M16C/63, 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Groups

Troubleshooting for Clocks in Development

R01AN1350EJ0110
Rev. 1.10
Dec. 2, 2013

Abstract

This application note describes troubles that developers have encountered and their solutions for the M16C/63, 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Groups. The application note is designed to be a guide for solving issues during development.

Products

M16C/63, 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Groups

The table below lists examples of trouble introduced in the application note.

Examples of Trouble Introduced in the Application Note

Trouble Example	Chapter
Troubles when Switching the CPU Clock	Chapter 1.
Trouble when Entering/Exiting Wait Mode	Chapter 2.
Trouble when Entering/Exiting Stop Mode	Chapter 3.

How to use this application note

- Trouble examples and their reference sections are provided with hyperlinks in the beginning of each chapter. Click the hyperlink of the appropriate trouble example to see details.
- To go back to the previous page from the linked page, press ALT + ← keys.
- Reference application notes are introduced for M16C/65C as a representative of products. Visit the Renesas Electronics website to see if the appropriate application note is available for the required product.

Abbreviations and acronyms are used to indicate some circuits, modes, and signals in the application note. The table below lists those abbreviations and acronyms.

Abbreviations and Acronyms Used in The Application Note

On-chip oscillator	OCO
40 MHz on-chip oscillator	40 MHz OCO
125 kHz on-chip oscillator	125 kHz OCO
40 MHz on-chip oscillator mode	40 MHz OCO mode
125 kHz on-chip oscillator mode	125 kHz OCO mode
125 kHz on-chip oscillator low power mode	125 kHz OCO low power mode
125 kHz on-chip oscillator clock	125 kHz OCO clock
PLL frequency synthesizer	PLL
High-performance Embedded Workshop	HEW
Flash Development Toolkit	FDT

Contents

1. Troubles when Switching the CPU Clock

Table 1.1 lists the Examples of Trouble and Possible Causes. Refer to the sections in the Refer to column for detailed explanations and troubleshooting.

Table 1.1 Examples of Trouble and Possible Causes

Section	Trouble	Possible Cause	Refer to
1.1	CPU Clock is not Switched as per the Settings	Write Operation is not Enabled in the Protect Register	1.1.1
		System Clock Protection Function is Enabled	1.1.2
		CM21 Bit in the CM2 Register is not Set	1.1.3
1.2	Clock does not Oscillate/Clock Stops	Procedure to Start Sub Clock Oscillation is not Correct	1.2.1
		Drive Capacity is Set to Low when Oscillation is Unstable	1.2.2
		Matching is not Achieved when the Drive Capacity is Low	1.2.3
1.3	Program Runs Out of Control/Program Stops	CPU Clock is not Switched Following the Procedure in the Manual	1.3.1
		Recommended Condition of the CPU Clock is not Satisfied	1.3.2
		Oscillation is not Stable when the Clock is Switched	1.3.3
		Matching of a Crystal/Ceramic Resonator is not Achieved	1.3.4
		40 MHz OCO Starts Oscillation or PLL Starts Operation when the CPU Clock Source is fOCO-S	1.3.5
		Clock Switched To is not Oscillating	1.3.6
		External Clock is Stopped	1.3.7
1.4	Unexpected Reset Occurrence	CPU Clock is not Switched Following the Procedure in the Manual	1.4.1
		Watchdog Timer is Started Automatically	1.4.2
		Counter is not Refreshed after the Watchdog Timer Interrupt Occurs	1.4.3
		40 MHz OCO Starts Oscillation or PLL Starts Operation during Power-on Reset or Voltage Detector Being Used	1.4.4
		40 MHz OCO Starts Oscillation or PLL Starts Operation During the Watchdog Timer Being Used	1.4.5
1.5	Unexpected Interrupt Occurrence	40 MHz OCO Starts Oscillation or PLL Starts Operation when Using the Voltage Detector	1.5.1
		40 MHz OCO Starts Oscillation or PLL Starts Operation When Using the Watchdog Timer	1.5.2
1.6	Output Pulse Width is Changed	Operation Mode of the CPU Clock is Changed	1.6.1
		Restrictions of the PLL Clock Division Ratio and Multiplying Factor are not Satisfied	1.6.2
—	When the Cause of the Issue Cannot be Found/Determined	—	5.

1.1 CPU Clock is not Switched as per the Settings

Examples:

- The CPU clock is not switched to the sub clock.
- MCU cannot enter stop mode.

1.1.1 Write Operation is not Enabled in the Protect Register

Check whether the system clock related registers (CM0, CM1, CM2, PCLKR, and so on) are rewritten with write operation disabled (the PRC0 bit in the PRCR register is 0).
System clock related registers are protected by the protect register (PRCR) so they are not easily rewritten in the event that a program runs out of control. For details, refer to the Protection chapter in the User's Manual: Hardware of each product.

Solution:

Enable writing to the system clock related registers by the protect register before rewriting to them.

Application Notes:

- Procedure for Using PLL Clock as CPU Clock Source (R01AN0404EJ)
- Transition between High-Speed Mode and Low Power Mode (Using Low Current Consumption Read Mode) (REJ05B1448)

❖ When you are not sure if this is the cause of the issue:

After setting a system clock related register, read the register by a program or debugger such as E8a emulator to see the expected value is set in the register.
When the read value is not the expected value, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.1 Examining a Register Setting
- 4.3 Checking the Frequency of the Operating CPU Clock
- 4.8 Checking Transition to Stop Mode

1.1.2 System Clock Protection Function is Enabled

Check whether the system clock related registers are rewritten with system clock protection function enabled (PM21 bit in the PM2 register is 1).
When the system clock protection function is enabled, the following bits remain unchanged even if they are written.

- Bits CM02, CM05 and CM07 in the CM0 register
- Bits CM10 and CM11 in the CM1 register
- The CM20 bit in the CM2 register
- All bits in the PLC0 register

Once the system clock protection function is enabled, the function cannot be disabled by a program.

Solution:
When using the system clock protection function, set the system clock related registers first, then enable the system clock protection function.
If the system requires the change of the system clock during operation, consider using the protect function (clock protection using the PRCR register) instead of the system clock protection function.

❖ When you are not sure if this is the cause of the issue:
Read the PM21 bit in the PM2 register by a program or debugger such as E8a emulator immediately before setting the system clock related register. When the PM21 bit is 1, this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.1 Examining a Register Setting
- 4.3 Checking the Frequency of the Operating CPU Clock
- 4.8 Checking Transition to Stop Mode

1.1.3 CM21 Bit in the CM2 Register is not Set

Check whether the CM21 bit (system clock select bit 2) in the CM2 register is set. The CPU clock source differs after reset depending on the product used.
The CPU clock source after reset is fOCO-S (CM21 bit is 1) in the M16C/63, 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Groups. The main clock becomes the CPU clock source (CM21 bit is 0) after reset in some existing products such as the M16C/62P Group. Some MCUs do not have the CM2 register. Refer to the Clock Generator chapter in the User’s Manual: Hardware for the appropriate product for details.

Solution:
Do not use the CM21 bit with the default value. The CM21 bit always needs to be set.

Application Notes:

- Procedure for Using PLL Clock as CPU Clock Source (R01AN0404EJ)
- Transition between High-Speed Mode and Low Power Mode (Using Low Current Consumption Read Mode) (REJ05B1448)

❖ When you are not sure if this is the cause of the issue:
Read the CM21 bit by a program or debugger such as E8a emulator immediately before setting a system clock related register.
When the main clock or PLL clock is used, if the CM21 bit is 1, this is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.1 Examining a Register Setting
- 4.3 Checking the Frequency of the Operating CPU Clock

1.2 Clock does not Oscillate/Clock Stops

Examples:

- The sub clock does not oscillate.
- The program stops when the drive capacity is set to low.

1.2.1 Procedure to Start Sub Clock Oscillation is not Correct

Check whether the procedure to start sub clock oscillation follows the user's manual. The sub clock is stopped during and after a reset. The procedure to start sub clock oscillation differs between products in the M16C/63 Group and products in the M16C/64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Groups.

For M16C/63 Group products, while the CM04 bit in the CM0 register is 0 (XCIN-XCOUT pin is used as an I/O port), if the CM03 bit in the CM0 register is written to 0 (sub clock on), the CM03 bit does not become 0.

For 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Group products, while the CM03 bit in the CM0 register is 0 (XCIN-XCOUT drive capacity is low), if the CM04 bit in the CM0 register is set to 1 (XCIN-XCOUT oscillation function), the sub clock may not oscillate.

Solution:

For M16C/63 Group products, set the CM04 bit to 1 (XCIN-XCOUT oscillation function) first, then set the CM03 bit to 0.

For 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Group products, set the CM04 bit to 1 while the CM03 bit is 1 (XCIN-XCOUT drive capacity is high).

Refer to the Clock Generator chapter in the User's Manual: Hardware for details.

Application Notes:

- M16C/63 Group Transition between High-Speed Mode and Low Power Mode (Using Low Current Consumption Read Mode) (REJ05B1445)
- M16C/65 Group Transition between High-Speed Mode and Low Power Mode (Using Low Current Consumption Read Mode) (REJ05B1448)

❖ When you are not sure if this is the cause of the issue:

For the M16C/63 Group, check the program to see if the CM03 bit is written to 0 while the CM04 bit is 0. In that case this item is the cause of the issue.

For 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Group products, check the program to see if the CM04 bit is written to 1 while the CM03 bit is 0. In that case this item may be the cause of the issue

Also refer to the following sections in 4. Analysis Methods:

- 4.1 Examining a Register Setting
- 4.2 Checking Oscillation of Crystals/Ceramic Resonators

1.2.2 Drive Capacity is Set to Low when Oscillation is Unstable

Check whether the clock oscillation is stable when the drive capacity is set to low.
If the drive capacity is changed while the oscillation is unstable, the program may run out of control or stop.

Solution:
Change the drive capacity after the clock oscillation becomes stable.

❖ When you are not sure if this is the cause of the issue:
Check whether the program operates correctly when the drive capacity remains high without changing it to low. When the program operates correctly with the drive capacity high, and the oscillation is stopped when the drive capacity is changed to low, this item may be the cause of the issue.
Also refer to the following section in 4. Analysis Methods:

- 4.2 Checking Oscillation of Crystals/Ceramic Resonators

1.2.3 Matching is not Achieved when the Drive Capacity is Low

Check whether matching of a crystal/ceramic resonator is achieved when the drive capacity is low.
If matching is not achieved when the drive capacity is low even though matching is achieved when the drive capacity is high, the program may run out of control or stop.

Solution:
When switching the drive capacities of the main clock and sub clock between low and high, matching must be achieved for both high and low drive capacities. For an evaluation of matching, contact the crystal/ceramic resonator manufacturer.

❖ When you are not sure if this is the cause of the issue:
Check whether the program operates correctly when the drive capacity remains high without changing it to low. When the program operates correctly with the drive capacity high, and the oscillation is stopped when the drive capacity is changed to low, this item may be the cause of the issue.
Also refer to the following section in 4. Analysis Methods:

- 4.2 Checking Oscillation of Crystals/Ceramic Resonators

1.3 Program Runs Out of Control/Program Stops

Examples:

- The program operates incorrectly when changing the CPU clock division from divide-by-8 to divide-by-1.
- The program may not operate correctly when the clock is switched to the sub clock.

1.3.1 CPU Clock is not Switched Following the Procedure in the Manual

Check whether the CPU clock is switched following the procedure in the user's manual.

When entering another mode from PLL operating mode, high-speed mode, medium-speed mode, 40 MHz OCO mode, or 125 kHz OCO mode, or entering these modes from another mode, the clocks of these modes must be divided by 8 or 16 before transition. Also the CPU clock needs to be switched by changing the division in a step-by-step manner for products in the M16C/64A, 64C, 65, 65C, 6C, 5LD, and 56D Groups. For example, when the division is switched from divide-by-8 to divide-by-1, switch the division from divide-by-8 to divide-by-4, divide-by-4 to divide-by-2, then divide-by-2 to divide-by-1.

If the switching procedure described above is not followed, the program may operate incorrectly. In that case the program may refer to where no instruction is placed, and often executes 00h or FFh as an instruction. 00h is the BRK instruction and FFh is an undefined instruction. Then when either of the instructions is executed, an interrupt for the instruction occurs. If the address set in the interrupt vectors for the interrupt is same as the address for the reset vector, these interrupts occur as if a reset had occurred.

Furthermore, when the watchdog timer is used, the watchdog timer counter is not refreshed. Thus the watchdog timer reset occurs.

Solution:

Switch the CPU clock following the procedures described in the Clock Generator and Power Control chapters in the User's Manual: Hardware.

Application Notes:

- Procedure for Using PLL Clock as CPU Clock Source (R01AN0404EJ)
- Transition between High-Speed Mode and Low Power Mode (Using Low Current Consumption Read Mode) (REJ05B1448)

❖ When you are not sure if this is the cause of the issue:

Check whether an interrupt for the BRK instruction or an undefined instruction, or a reset has occurred when switching the CPU clock.

If a reset occurred, read the RSTFR register (reset source determine register) to determine the reset source.

If an interrupt for the BRK instruction or an undefined instruction occurred, or the reset source is the watchdog timer, this item may be the cause of the issue.

Also refer to the following sections in 4. Analysis Methods:

- 4.5 Investigating the Cause when the System Operates Incorrectly
- 4.6 Determining the Reset Source

1.3.2 Recommended Condition of the CPU Clock is not Satisfied

Check whether a crystal/ceramic resonator connected to the main clock oscillator, or an external clock frequency meets the recommended operating condition in the Electrical Characteristics chapter in the User’s Manual: Hardware. If the condition is not satisfied, the program may run out of control or stop.

Solution:
Connect a crystal/ceramic resonator or supply a frequency to the main clock oscillator that satisfies the recommended operating condition in the Electrical Characteristics chapter in the User’s Manual: Hardware.

❖ When you are not sure if this is the cause of the issue:
Check whether the recommended operating condition in Electrical Characteristics chapter in the User’s Manual: Hardware is satisfied.
Also refer to the following sections in 4. Analysis Methods:

- 4.2 Checking Oscillation of Crystals/Ceramic Resonators
- 4.3 Checking the Frequency of the Operating CPU Clock
- 4.5 Investigating the Cause when the System Operates Incorrectly

1.3.3 Oscillation is not Stable when the Clock is Switched

Check whether the clock oscillation becomes stable after the clock starts oscillation when switching the CPU clock. If the CPU clock is switched while the oscillation of the clock switched to is not stabilized, the program may run out of control or stop.

Solution:
When the clock starts oscillation, wait until the clock oscillation stabilizes, then switch the clock. Refer to the Electrical Characteristics chapter in the User’s Manual: Hardware for details on the wait time until the OCO oscillation stabilizes and PLL stabilizes.
Wait time until main clock or sub clock oscillation stabilizes differs depending on a crystal/ceramic resonator. Contact the crystal/ceramic resonator manufacturer for details.

❖ When you are not sure if this is the cause of the issue:
Start clock oscillation, then check whether the clock oscillation becomes stable before switching the clock. If the oscillation is not stable, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.2 Checking Oscillation of Crystals/Ceramic Resonators
- 4.5 Investigating the Cause when the System Operates Incorrectly

1.3.4 Matching of a Crystal/Ceramic Resonator is not Achieved

Check whether matching of a crystal/ceramic resonator is achieved. Also check whether the matching is confirmed in the user system. If matching is not achieved, a crystal/ceramic resonator may not oscillate or the oscillation may stop in the middle of the operation. The electrical characteristics of the oscillation depends largely on the board layout used, thus matching needs to be confirmed in the user system.

Solution:
Contact the crystal/ceramic resonator manufacturer for the matching evaluation. When performing the matching evaluation, the actual system must be used as the evaluation is affected by peripheral circuits and so on. When the system is changed, the characteristics may change accordingly. Therefore the matching evaluation is required again.

❖ When you are not sure if this is the cause of the issue:
If the matching evaluation is not performed, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.2 Checking Oscillation of Crystals/Ceramic Resonators
- 4.5 Investigating the Cause when the System Operates Incorrectly

1.3.5 40 MHz OCO Starts Oscillation or PLL Starts Operation when the CPU Clock Source is fOCO-S

Check whether the 40 MHz OCO starts oscillation or PLL is enabled to operate when in 125 kHz OCO mode or 125 kHz OCO low power mode, and also operating with divide-by-1, divide-by-2, or divide-by-4. In that case the MCU may operate incorrectly.

Products applied this item to:
R5F3650ENFA/FB, R5F3650EDFA/FB, R5F36506NFA/FB, R5F36506DFA/FB,
R5F364AENFA/FB, R5F364AEDFA /FB, R5F364A6NFA/FB, R5F364A6DFA/FB
(Technical update: TN-16C-A177A/E)

Solution:
When operating in 125 kHz OCO mode or 125 kHz OCO low power mode, change the division to divide-by-8 or divide-by-16, then start 40 MHz OCO oscillation or enable PLL operation.

Technical Update:

- TN-16C-A177A/E

❖ When you are not sure if this is the cause of the issue:
Check whether the program has codes that the 40 MHz OCO starts oscillation or PLL is enabled to operate when the CPU clock source is fOCO-S.
If such codes are found and the oscillation or operation is not started as described in the Solution above, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.2 Checking Oscillation of Crystals/Ceramic Resonators
- 4.5 Investigating the Cause when the System Operates Incorrectly

1.3.6 Clock Switched To is not Oscillating

Check whether the CPU clock is switched while the clock switched to is not oscillating. In that case the program stops.

Solution:
When switching the CPU clock, oscillate the clock switched to, wait until the clock oscillation stabilizes, and then switch the clock.

❖ When you are not sure if this is the cause of the issue:
Check whether the clock switched to is oscillating immediately before switching the CPU clock.

- Check the XOUT pin when switching to the main clock and the XCOUT pin when switching to the sub clock using an oscilloscope.
- When switching to the 125 kHz OCO clock, check whether the FRA01 bit in the FRA0 register is 0 (125 kHz on-chip oscillator) and also the CM14 bit in the CM1 register is 0 (125 kHz on-chip oscillator on).
- When switching to the 40 MHz OCO clock, check whether the FRA00 bit in the FRA0 register is 1 (40 MHz on-chip oscillator on) and also the FRA01 bit in the FRA0 register is 1 (40 MHz on-chip oscillator).

If the clock switched to stops, this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.1 Examining a Register Setting
- 4.2 Checking Oscillation of Crystals/Ceramic Resonators
- 4.5 Investigating the Cause when the System Operates Incorrectly

1.3.7 External Clock is Stopped

Check whether the external clock is stopped when it is connected to the XIN pin.

Solution:
Do not stop the external clock when it is connected to the XIN pin and the main clock is selected as the CPU clock source.

Technical Update:

- M16C-109-0309

❖ When you are not sure if this is the cause of the issue:
Check whether the external clock is stopped. In that case this item may be the cause of the issue.

1.4 Unexpected Reset Occurrence

Examples:

- Initialization is repeatedly performed every few seconds after reset.
- Reset occurs periodically.

1.4.1 CPU Clock is not Switched Following the Procedure in the Manual

Check whether the CPU clock is switched following the procedure in the user's manual.
When entering another mode from PLL operating mode, high-speed mode, medium-speed mode, 40 MHz on-chip oscillator mode, or 125 kHz on-chip oscillator mode, or entering these modes from another mode, the clocks of these modes must be divided by 8 or 16 before transition. Also the CPU clock needs to be switched by changing the division in a step-by-step manner for products in the M16C/64A, 64C, 65, 65C, 6C, 5LD, and 56D Groups. For example, when the division is switched from divide-by-8 to divide-by-1, switch the division from divide-by-8 to divide-by-4, divide-by-4 to divide-by-2, then divide-by-2 to divide-by-1.
If the switching procedure described above is not followed, the program may operate incorrectly. In that case the program may refer to where no instruction is placed, and often executes 00h or FFh as an instruction. 00h is the BRK instruction and FFh is an undefined instruction. Then when either of instructions is executed, an interrupt for the instruction occurs. If the address set in the interrupt vectors for the interrupt is same as the address for the reset vector, these interrupts occur as if a reset had occurred.
Furthermore, when the watchdog timer is used, the watchdog timer counter is not refreshed. Thus the watchdog timer reset occurs.

Solution:
Switch the CPU clock following the procedures described in the Clock Generator and Power Control chapters in the User's Manual: Hardware.

Refer to “1.3.1 CPU Clock is not Switched Following the Procedure in the Manual” when you are not sure if this is the cause of the issue or for reference materials.

1.4.2 Watchdog Timer is Started Automatically

Check whether the watchdog timer is automatically started.
Products in the M16C/63, 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Groups have the function to start the watchdog timer automatically after reset.

Solution:
When not using the watchdog timer, set bit 0 in address FFFFFh to 1 (watchdog timer is stopped after reset).

Application Note:

- Watchdog Timer (REJ05B1170)

❖ When you are not sure if this is the cause of the issue:
Read the mot file with simulator debugger of HEW or FDT and check whether bit 0 in address FFFFFh is 0 (watchdog timer starts automatically after reset) with a memory window and so on.
If bit 0 is 0 in address FFFFFh when the watchdog timer is not used, this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.4 Examining Codes where an Interrupt Occurs
- 4.5 Investigating the Cause when the System Operates Incorrectly
- 4.6 Determining the Reset Source

1.4.3 Counter is not Refreshed after the Watchdog Timer Interrupt Occurs

Check whether the watchdog timer counter is refreshed with the WDTR register after the watchdog timer interrupt occurs.
When the watchdog timer counter is not refreshed, if the watchdog timer underflows again, the program may operate incorrectly.

Solution:
Refresh the watchdog timer counter with the WDTR register after the watchdog timer interrupt occurs.
When the watchdog timer interrupt occurs, the program is possibly running out of control. Refer to 1.3 Program Runs Out of Control/Program Stops to see if the cause of the program runaway is in the program.

Application Note:

- Watchdog Timer (REJ05B1170)

❖ When you are not sure if this is the cause of the issue:
Check whether the watchdog timer interrupt occurs, and also check whether the watchdog timer counter is refreshed after the watchdog timer interrupt occurred.
When the watchdog timer interrupt occurs and the watchdog timer counter is not refreshed, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.5 Investigating the Cause when the System Operates Incorrectly
- 4.6 Determining the Reset Source

1.4.4 40 MHz OCO Starts Oscillation or PLL Starts Operation during Power-on Reset or Voltage Detector Being Used

Check whether the 40 MHz OCO starts oscillation or PLL is enabled to operate when using the power-on reset or voltage detector.
While using the voltage detector, when the 40 MHz OCO starts oscillating or PLL is enabled to operate, some error may be introduced temporarily into the detection voltage of the voltage detector, and a reset or an interrupt may occur on a voltage outside the range of the electrical characteristics.

Products applied this item to:
R5F3650ENFA/FB, R5F3650EDFA/FB, R5F36506NFA/FB, R5F36506DFA/FB,
R5F364AENFA/FB, R5F364AEDFA/FB, R5F364A6NFA/FB, R5F364A6DFA/FB
(Technical update: TN-16C-A177A/E)

Solution:
Do not change the FRA00 bit in the VCR2 register from 0 to 1 when any bit from VC25 to VC27 in the VCR2 register is 1. When using the power-on reset or voltage detector, change the FRA00 bit as follows:

(1) Set bits VC25, VC26, and VC27 all to 0 (voltage detector disabled).
(2) Change the FRA00 bit (or PLC07 bit) from 0 to 1.
(3) Wait for 1 ms.
(4) Set the required bit from VC25 to VC27 to 1 (voltage detector enabled).

Technical Update:

- TN-16C-A177A/E

❖ When you are not sure if this is the cause of the issue:
When using the power-on reset or voltage detector, check whether the program has codes that 40 MHz OCO starts oscillation or PLL is enabled to operate. In that case, and 40 MHz OCO oscillation or PLL operation is not started following the procedure in the Solution above, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.4 Examining Codes where an Interrupt Occurs
- 4.5 Investigating the Cause when the System Operates Incorrectly
- 4.6 Determining the Reset Source

1.4.5 40 MHz OCO Starts Oscillation or PLL Starts Operation During the Watchdog Timer Being Used

Check whether the 40 MHz OCO starts oscillation or PLL is enabled to operate when fOCO-S is selected as the count source of the watchdog timer.
When the fOCO-S is selected as the count source of the watchdog timer, if the 40 MHz OCO starts oscillation or PLL is enabled to operate, the watchdog timer counter may count incorrectly.

Products applied this item to:
R5F3650ENFA/FB, R5F3650EDFA/FB, R5F36506NFA/FB, R5F36506DFA/FB,
R5F364AENFA/FB, R5F364AEDFA/FB, R5F364A6NFA/FB, R5F364A6DFA/FB
(Technical update: TN-16C-A177A/E)

Solution:
To start the 40 MHz OCO oscillation or enable the PLL operation, follow the procedure below:

(1) Write 00h and then FFh to the WDTR register (refresh the watchdog timer).
(2) Change the FRA00 bit (or PLC07 bit) from 0 to 1.
(3) Wait for 1 ms.
(4) Write 00h and then FFh to the WDTR register (refresh the watchdog timer).

Technical Update:

- TN-16C-A177A/E

❖ When you are not sure if this is the cause of the issue:
Check whether the program has codes that the 40 MHz OCO starts oscillation or PLL is enabled to operate when using the watchdog timer with fOCO-S as the count source. In that case, and 40 MHz OCO oscillation or PLL operation is not started following the procedure in the Solution above, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.4 Examining Codes where an Interrupt Occurs
- 4.5 Investigating the Cause when the System Operates Incorrectly
- 4.6 Determining the Reset Source

1.5 Unexpected Interrupt Occurrence

Example:

- An interrupt occurs when switching the CPU clock from the main clock to the PLL clock.

1.5.1 40 MHz OCO Starts Oscillation or PLL Starts Operation when Using the Voltage Detector

Check whether the 40 MHz OCO starts oscillation or PLL is enabled to operate when using the voltage detector.
While using the voltage detector, when the 40 MHz OCO starts oscillation or PLL is enabled to operate, some error may be introduced temporarily into the detection voltage of the voltage detector, and a reset or an interrupt may occur on a voltage outside the range of the electrical characteristics.

Products applied this item to:
R5F3650ENFA/FB, R5F3650EDFA/FB, R5F36506NFA/FB, R5F36506DFA/FB,
R5F364AENFA/FB, R5F364AEDFA/FB, R5F364A6NFA/FB, R5F364A6DFA/FB
(Technical update: TN-16C-A177A/E)

Solution:

Do not change the FRA00 bit in the VCR2 register from 0 to 1 when any bit from VC25 to VC27 in the VCR2 register is 1. When using the voltage detector, change the FRA00 bit as follows:

(1) Set bits VC25 to VC27 to 0 (voltage detector disabled).
(2) Change the FRA00 bit (or PLC07 bit) from 0 to 1.
(3) Wait for 1 ms.
(4) Set the required bit from VC25 to VC27 to 1 (voltage detector enabled).

Refer to “1.4.4 40 MHz OCO Starts Oscillation or PLL Starts Operation during Power-on Reset or Voltage Detector Being Used” when you are not sure if this is the cause of the issue or for reference materials.

1.5.2 40 MHz OCO Starts Oscillation or PLL Starts Operation When Using the Watchdog Timer

Check whether the 40 MHz OCO starts oscillation or PLL is enabled to operate when the fOCO-S is selected as the count source of the watchdog timer.
When the fOCO-S is selected as the count source of the watchdog timer, if the 40 MHz OCO starts oscillation or PLL is enabled to operate, the watchdog timer counter may count incorrectly.

Products applied this item to:
R5F3650ENFA/FB, R5F3650EDFA/FB, R5F36506NFA/FB, R5F36506DFA/FB,
R5F364AENFA/FB, R5F364AEDFA/FB, R5F364A6NFA/FB, R5F364A6DFA/FB
(Technical update: TN-16C-A177A/E)

Solution:

To start the 40 MHz OCO oscillation or enable PLL to operate, follow the procedure below:

(1) Write 00h and then FFh to the WDTR register (refresh the watchdog timer).
(2) Change the FRA00 bit (or PLC07 bit) from 0 to 1.
(3) Wait for 1 ms.
(4) Write 00h and then FFh to the WDTR register (refresh the watchdog timer).

Refer to “1.4.5 40 MHz OCO Starts Oscillation or PLL Starts Operation During the Watchdog Timer Being Used” when you are not sure if this is the cause of the issue or for reference materials.

1.6 Output Pulse Width is Changed

Example:

- Timer pulse width being output is changed when switching the CPU clock from the PLL clock to the main clock.

1.6.1 Operation Mode of the CPU Clock is Changed

Check whether the operating mode is changed, for example, the CPU clock source is switched from the PLL clock to the main clock.
In the following operating mode, the clock source provided to the CPU clock and f1 is the same. Thus if the CPU clock source is changed, the f1 frequency is changed accordingly.

High-speed mode, medium-speed mode, PLL operating mode, 40 MHz OCO mode,
125 kHz OCO mode, 125 kHz OCO low power mode

Therefore if the operating mode is changed for peripheral functions that select f1 as the count source or operating clock (hereinafter this is also referred to as count source), the f1 frequency is changed, then the output pulse width or operating frequency is changed.

Solution:

When changing the CPU clock operating mode, reset peripheral functions which select f1 as the count source depending on the changed operating mode.
Or consider using fOCO-S, fOCO-F, fOCO40M, fC32, or fC as the count source. When one of these is selected as the count source, the CPU clock operating mode does not affect the count source. Thus the output pulse width is not changed by changing the operating mode.

In the M16C/65C and M16C/64C Groups, the main clock can be selected as the timer A or timer B count source that is independent of the CPU clock source.
Count sources that can be provided to peripheral functions differ depending on products. For details, refer to the User’s Manual: Hardware for the product used.

Application Note:

- M16C/64C Group, M16C/65C Group Providing a Timer Clock Independent of the CPU Clock (R01AN0711EJ)

❖ When you are not sure if this is the cause of the issue:

Use the oscilloscope and so on to see whether the timing for the CPU clock to be switched and the timing for output pulse width to be changed are same. If these timings are same, this item may be the cause of the issue.
Also refer to the following section in 4. Analysis Methods:

- 4.3 Checking the Frequency of the Operating CPU Clock

1.6.2 Restrictions of the PLL Clock Division Ratio and Multiplying Factor are not Satisfied

Check whether the clock frequency divided by bits PLC05 and PLC04 (reference frequency counter set bit) in the PLC0 register is between 2 MHz and 5 MHz (6 MHz in the M16C/6C Group). Also check whether the clock frequency multiplied by bits PLC02 to PLC00 (PLL multiplying factor select bit) in the PLC0 register met the recommended operating conditions in the Electrical Characteristics chapter in the User’s Manual: Hardware.
If the restrictions of PLL division ratio and multiplying factor are not satisfied, the generated clock is unstable and the output pulse width or operating cycle may become incorrect.

Solution:
Set the clock frequency divided by bits PLC05 and PLC04 to be between 2 MHz and 5 MHz (6 MHz in the M16C/6C Group). Set the clock frequency multiplied by bits PLC02 to PLC00 to meet the recommended operating conditions of the electrical characteristics.

Application Notes:

- Procedure for Using PLL Clock as CPU Clock Source (R01AN0404EJ)
- Transition between High-Speed Mode and Low Power Mode (Using Low Current Consumption Read Mode) (REJ05B1448)

❖ When you are not sure if this is the cause of the issue:
Check the set value in the PLC0 register. If the values do not meet the restrictions of the PLL clock division ratio and multiplying factor, this item may be the cause of the issue.
Also refer to the following section in 4. Analysis Methods:

- 4.3 Checking the Frequency of the Operating CPU Clock

2. Trouble when Entering/Exiting Wait Mode

Table 2.1 lists the Examples of Trouble and Possible Causes. Refer to the sections in the Refer to column for detailed explanations and troubleshooting.

Table 2.1 Examples of Trouble and Possible Causes

Section	Trouble	Possible Cause	Refer to
2.1	MCU cannot Enter Wait Mode	CM02 Bit in the CM0 Register is Set to 1 while the Oscillation Stop/Restart Detection Function is Used	2.1.1
2.2	MCU cannot Exit Wait mode	Interrupt Used for Exit is not Enabled	2.2.1
		Interrupt is Enabled Immediately before the Wait Instruction	2.2.2
		Clock is Unstable when Switching the Clock after Exiting Wait Mode	2.2.3
		CM02 Bit in the CM0 Register is Set to 1	2.2.4
		Transition is Made from Low Current Consumption Read Mode to Wait Mode	2.2.5
2.3	Current Consumption cannot be Reduced	PLC07 Bit in the PLC0 Register is Set to 1	2.3.1
		NMI Interrupt is Enabled and the NMI Pin is Low	2.3.2
		Unused Pins are not Processed	2.3.3
2.4	Unexpected Reset Occurrence	Watchdog Timer is Used with Count Source Protection Mode Enabled	2.4.1
—	When the Cause of the Issue Cannot be Found/Determined	—	5.

2.1 MCU cannot Enter Wait Mode

Example:

- MCU cannot enter wait mode even if the wait instruction is executed.

2.1.1 CM02 Bit in the CM0 Register is Set to 1 while the Oscillation Stop/Restart Detection Function is Used

Check whether the oscillation stop/restart detection function is used and the CM02 bit in the CM0 register is set to 1 (peripheral function clock f1 stops in wait mode).
When the oscillation stop/restart detection function is used and the CM02 bit is 1, if wait mode is entered, the clock stops oscillation and the oscillator stop/restart detect interrupt occurs. Then the timing of exiting wait mode becomes same as the timing of entering wait mode. Thus wait mode cannot be entered.

Solution:

When entering wait mode while the oscillation stop/restart detection function is used, set the CM02 bit to 0.

❖ When you are not sure if this is the cause of the issue:

Read the CM02 bit by a program or debugger such as E8a emulator immediately before the wait instruction is executed. When the oscillation stop/restart detection function is used and the CM02 bit is 1, this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.4 Examining Codes where an Interrupt Occurs
- 4.7 Checking Transition to Wait Mode

2.2 MCU cannot Exit Wait mode

Example:

- MCU cannot exit wait mode even if an interrupt for exit is input.

2.2.1 Interrupt Used for Exit is not Enabled

Check whether an interrupt used to exit wait mode is enabled.
The program stops after the wait instruction is executed, thus an interrupt to exit wait mode needs to be enabled before executing the wait instruction.

Solution:
Enable an interrupt to exit wait mode before the wait instruction is executed. When using the maskable interrupt, set the interrupt priority level for an interrupt used for exit to 1 or higher, and set the I flag to 1 (maskable interrupt enabled) immediately before the wait instruction is executed.

Application Note:

- Wait Mode Setup (REJ05B1172)

❖ When you are not sure if this is the cause of the issue:
Read the I flag status and the set value in the interrupt control register for an interrupt used by a program or debugger such as E8a emulator immediately before the wait instruction is executed. If the interrupt is not enabled in either of settings, this item is the cause of the issue.

- 4.1 Examining a Register Setting

2.2.2 Interrupt is Enabled Immediately before the Wait Instruction

Check if the I flag is set to 1 (maskable interrupt enabled) immediately before the wait instruction.
When the I flag is not set to 1 immediately before the wait instruction, an interrupt may occur before the wait instruction is executed. If the system allows an interrupt for exit wait mode to occur only once, the system cannot exit wait mode.

Solution:
Set the I flag to 1 immediately before the wait instruction. Then an interrupt request will not be accepted before being in wait mode.

Application Note:

- Wait Mode Setup (REJ05B1172)

❖ When you are not sure if this is the cause of the issue:
Check whether the program has an instruction to set the I flag to 1 immediately before the wait instruction. If not, this item may be the cause of the issue.

2.2.3 Clock is Unstable when Switching the Clock after Exiting Wait Mode

Check whether the clock oscillation becomes stable when switching the clock after exiting from wait mode. The CPU clock stops during wait mode, thus the clock must be confirmed to become stable when switching the clock immediately after exiting from wait mode.

Solution:
When switching the clock after exiting from wait mode, wait until the clock oscillation stabilizes, then switch the clock. Refer to the Electrical Characteristics chapter in the User's Manual: Hardware for details on wait time until the OCO oscillation stabilizes and PLL stabilizes.
Wait time until the main clock or sub clock oscillation stabilizes differs depending on a crystal/ceramic resonator. Contact the crystal/ceramic resonator manufacturer for details.

❖ When you are not sure if this is the cause of the issue:
Check whether the clock is stable when switching the clock after exiting from wait mode. If not, this item may be the cause of the issue.

- 4.2 Checking Oscillation of Crystals/Ceramic Resonators

2.2.4 CM02 Bit in the CM0 Register is Set to 1

Check whether the CM02 bit in the CM0 register is set to 1 (peripheral function clock f1 stops in wait mode).
When the CM02 bit is set to 1, f1 that is provided to peripheral functions stops during wait mode. Thus the peripheral functions that use f1 as the count source cannot be used to exit wait mode.

Solution:
When exiting from wait mode using an interrupt of a peripheral function that uses f1 as the count source, set the CM02 bit to 0 (peripheral function clock f1 does not stop in wait mode).
Also set the CM02 bit to 0 when entering wait mode while the oscillation stop/restart detection function is used.

Application Note:

- Wait Mode Setup (REJ05B1172)

❖ When you are not sure if this is the cause of the issue:
Read the CM02 bit and set value for the count source of the peripheral function used for exit by a program or debugger such as E8a emulator immediately before the wait instruction is executed. If the CM02 bit is 1 and the count source is f1, this item may be the cause of the issue.

2.2.5 Transition is Made from Low Current Consumption Read Mode to Wait Mode

Check whether the transition is made from low current consumption read mode to wait mode.
If entering wait mode from low current consumption read mode, the program may operate incorrectly when exiting from wait mode.

Solution:
Set the FMR23 bit in the FMR2 register to 0 (low current consumption read mode disabled), set the mode to slow read mode, then enter wait mode.

❖ When you are not sure if this is the cause of the issue:
Read the set value in the system clock related register by a program or debugger such as E8a emulator immediately before entering wait mode. If the operating mode is set to low current consumption read mode, this item may be the cause of the issue.

2.3 Current Consumption cannot be Reduced

Example:

- Current consumption cannot be reduced even if entering wait mode.

2.3.1 PLC07 Bit in the PLC0 Register is Set to 1

Check whether wait mode is entered while the PLC07 bit in the PLC0 register is 1 (PLL on).
When the PLC07 bit is 1, current consumption cannot be reduced even if entering wait mode.

Solution:

Set the PLC07 bit to 0 (PLL off) first, then enter wait mode.

Application Notes:

- Wait Mode Setup (REJ05B1172)
- Concept of the Power Control (MAEC-MCU-M16C-102-0302)

❖ When you are not sure if this is the cause of the issue:
Read the PLC07 bit by a program or debugger such as E8a emulator immediately before the wait instruction is executed. If the PLC07 bit is 1, this item is the cause of the issue.

2.3.2 NMI Interrupt is Enabled and the NMI Pin is Low

Check whether the NMI interrupt is enabled and a low level signal is input to the NMI pin.
When the wait instruction is executed while the NMI interrupt is enabled and a low level signal is input to the NMI pin, the CPU clock does not stop despite the CPU stopping. Thus current consumption cannot be reduced.

Solution:

Set the NMI pin to be high when entering wait mode. When the NMI interrupt is not used, disable the NMI interrupt.

❖ When you are not sure if this is the cause of the issue:
Use an oscilloscope and so on to see the NMI pin level when the wait instruction is executed.
When the NMI interrupt is enabled and the NMI pin is low, this item is the cause of the issue.

2.3.3 Unused Pins are not Processed

Check whether unused pins are set to input mode with a pull-up or pull-down resistor, or released by setting them to output mode. If any of them are not performed for unused pins, peripheral circuits or noise affect them and levels of those pins may become midpoint potential. When the unused pins have the midpoint potential, the shoot-through current flows and current consumption increases.

Solution:

Process unused pins by performing one of the following:

- Set unused pins to input mode with a pull-up or pull-down resistor.
- Release unused pins by setting them to output mode.

Application Note:

- Power Control Low Current Consumption Setting (R01AN0409EJ)

❖ When you are not sure if this is the cause of the issue:

Check if there are any unused pins that have not been processed. In that case, this item may be the cause of the issue.

2.4 Unexpected Reset Occurrence

Example:

- A reset occurs when entering wait mode.

2.4.1 Watchdog Timer is Used with Count Source Protection Mode Enabled

Check whether the watchdog timer is used with count source protection mode enabled.
When the watchdog timer is used with count source protection mode enabled, the watchdog timer counter does not stop counting even if entering wait mode.

Solution:

When using the watchdog timer in wait mode and count source protection mode simultaneously, exit wait mode and refresh the watchdog timer counter periodically.
Or consider a way to use the watchdog timer with count source protection disabled.

❖ When you are not sure if this is the cause of the issue:
Read the CSPRO bit (count source protection mode select bit) in the CSPR register and the CSPROINI bit (after-reset count source protection mode select bit) in address FFFFFh (optional function select address 1). If their settings are to use count source protection mode, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.5 Investigating the Cause when the System Operates Incorrectly
- 4.6 Determining the Reset Source

3. Trouble when Entering/Exiting Stop Mode

Table 3.1 lists the Examples of Trouble and Possible Causes. Refer to the sections in the Refer to column for explanations and troubleshooting.

Table 3.1 Examples of Trouble and Possible Causes

Section	Trouble	Possible Cause	Refer to
3.1	MCU cannot Enter Stop Mode	Write Operation is Enabled in the Protect Register	3.1.1
		System Clock Protection Function is Enabled	3.1.2
		PLC07 Bit in the PLC0 Register is Set to 1	3.1.3
		NMI Interrupt is Enabled and the NMI Pin is Low	3.1.4
		Watchdog Timer is Used with Count Source Protection Mode Enabled	3.1.5
3.2	MCU cannot Exit Stop Mode	Interrupt Used for Exit is not Enabled	3.2.1
		Interrupt is not Enabled Immediately Before the Instruction to Set the CM10 Bit to 1	3.2.2
		Clock Oscillation is Unstable when Switching the Clock after Exiting from Stop Mode	3.2.3
3.3	Instruction Immediately After Exiting Stop Mode is not Performed Correctly	JMP and NOP Instructions are not Inserted Properly after the Instruction to Set the CM10 Bit to 1	3.3.1
3.4	Unexpected Reset Occurrence	Transition is Made from 40 MHz OCO Mode to Stop Mode when Using the Voltage Detector	3.4.1
3.5	Unexpected Interrupt Occurrence	Transition is made from 40 MHz OCO Mode to Stop Mode when Using the Voltage Detector	3.5.1
—	When the Cause of the Issue Cannot be Found/Determined	—	5.

3.1 MCU cannot Enter Stop Mode

Example:

- MCU cannot enter stop mode even if the CM10 bit is set to 1.

3.1.1 Write Operation is Enabled in the Protect Register

Check whether the system clock related registers (CM0, CM1, CM2, PCLKR, and so on) are rewritten with write operation disabled (PRC0 bit in the PRCR register is 0).
The system clock related registers are protected with the protect register (PRCR) so that they are not easily rewritten in the event that a program runs out of control. For details, refer to the Protection chapter in the User's Manual: Hardware for each product.

Solution:
Enable writing to the system clock related registers by the protect register before rewriting to them.

Refer to "1.1.1 Write Operation is not Enabled in the Protect Register" when you are not sure if this is the cause of the issue or for reference materials.

3.1.2 System Clock Protection Function is Enabled

Check whether the system clock related registers are rewritten with system clock protection function enabled (PM21 bit in the PM2 register is 1).
When the system clock protection function is enabled, the following bits remain unchanged even if they are written.

- Bits CM02, CM05 and CM07 in the CM0 register
- Bits CM10 and CM11 in the CM1 register
- CM20 bit in the CM2 register
- All bits in the PLC0 register

Once the system clock protection function is enabled, the function cannot be disabled by a program.

Solution:
When using the system clock protection function, set the system clock related registers, then enable the system clock protection function.
If the system requires to change the system clock during operation, consider using the protect function (clock protection with the PRCR register) instead of the system clock protection function.

Refer to "1.1.2 System Clock Protection Function is Enabled" when you are not sure if this is the cause of the issue or for reference materials.

3.1.3 PLC07 Bit in the PLC0 Register is Set to 1

Check whether the stop mode is set when the PLC07 bit in the PLC0 register is 1 (PLL on).
When the PLC07 bit is 1, stop mode cannot be entered.

Solution:
Set a mode other than PLL operating mode first, then set the mode to stop mode. For example enter medium-speed mode, then set the PLC07 bit to 0 (PLL off).

❖ When you are not sure if this is the cause of the issue:
Read the PLC07 bit by a program or debugger such as E8a emulator immediately before executing an instruction to set the CM10 bit in the CM1 register to 1 (all clocks off (stop mode)). If the PLC07 bit is 1, this item is the cause of the issue.

3.1.4 NMI Interrupt is Enabled and the NMI Pin is Low

Check whether the NMI interrupt is enabled and a low level signal is input to the NMI pin.
When the NMI interrupt is enabled and a low level signal is input to the NMI pin, stop mode cannot be entered as the CM10 bit in the CM1 register is fixed to 0.

Solution:
When entering stop mode, set the NMI pin to be high or disable the NMI interrupt.

❖ When you are not sure if this is the cause of the issue:
Use an oscilloscope and so on to check the NMI pin level when executing the instruction to set the CM10 bit to 1 (all clocks off (stop mode)). If the NMI interrupt is enabled and the NMI pin is low, this item is the cause of the issue.

3.1.5 Watchdog Timer is Used with Count Source Protection Mode Enabled

Check whether the watchdog timer is used with count source protection mode enabled.
When the watchdog timer is used with count source protection mode enabled, stop mode cannot be entered.

Solution:
When using both the watchdog timer and stop mode simultaneously, consider a way to use the watchdog timer with count source protection disabled.

❖ When you are not sure if this is the cause of the issue:
Read the CSPRO bit (count source protection mode select bit) in the CSPR register and the CSPROINI bit (after-reset count source protection mode select bit) in address FFFFFh (optional function select address 1). If their settings are to use count source protection mode, this item may be the cause of the issue.

3.2 MCU cannot Exit Stop Mode

Example:

- MCU cannot exit stop mode even if an interrupt for exit is input.

3.2.1 Interrupt Used for Exit is not Enabled

Check whether an interrupt used to exit stop mode is enabled.
The program stops after executing the instruction to set the CM10 bit in the CM1 register to 1 (all clocks off (stop mode)), thus an interrupt to exit stop mode needs to be enabled before executing the instruction.

Solution:
Enable an interrupt to exit stop mode before the instruction to set the CM10 bit to 1 is executed. When using the maskable interrupt, set the interrupt priority level for an interrupt used for exit to 1 or higher, and set the I flag to 1 (maskable interrupt enabled) immediately before the instruction to set the CM10 bit to 1.

Application Note:

- Stop Mode Setup (REJ05B1171)

❖ When you are not sure if this is the cause of the issue:
Read the I flag status and the set value in the interrupt control register for an interrupt used by a simulator, emulator, or program immediately before the instruction to set the CM10 bit to 1 is executed. If the interrupt is not enabled in either of settings, then this item is the cause of the issue.
Also refer to the following section in 4. Analysis Methods:

- 4.1 Examining a Register Setting

3.2.2 Interrupt is not Enabled Immediately Before the Instruction to Set the CM10 Bit to 1

Check if the I flag is set to 1 (maskable interrupt enabled) immediately before executing the instruction to set the CM10 bit in the CM1 register to 1 (all clocks off (stop mode)).
When the I flag is not set to 1 immediately before the instruction to set the CM10 bit to 1, an interrupt may occur before the instruction is executed. If the system allows an interrupt for exiting from stop mode to occur only once, the system cannot exit stop mode.

Solution:
Set the I flag to 1 immediately before the instruction to set the CM10 bit to 1. Then an interrupt request will not be accepted before entering stop mode.

Application Note:

- Stop Mode Setup (REJ05B1171)

❖ When you are not sure if this is the cause of the issue:
Check whether the program has an instruction to set the I flag to 1 immediately before the instruction to set the CM10 bit to 1. If not, this item may be the cause of the issue.

3.2.3 Clock Oscillation is Unstable when Switching the Clock after Exiting from Stop Mode

Check whether the clock oscillation becomes stable when switching the clock after exiting from stop mode. The CPU clock stops during stop mode, thus the clock switched to must be confirmed to become stable when switching the clock immediately after exiting from stop mode.

Solution:
When switching the clock after exiting from stop mode, wait until the clock oscillation stabilizes, then switch the clock. Refer to the Electrical Characteristics chapter in the User's Manual: Hardware for details on wait time until the OCO oscillation stabilizes and PLL stabilizes.
Wait time until the main clock or sub clock oscillation stabilizes differs depending on a crystal/ceramic resonator. Contact the crystal/ceramic resonator manufacturer for details.

❖ When you are not sure if this is the cause of the issue:
Check whether the clock is stable when switching the clock after exiting from stop mode. If not, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.2 Checking Oscillation of Crystals/Ceramic Resonators

3.3 Instruction Immediately After Exiting Stop Mode is not Performed Correctly

Example:

- When an interrupt for exit is input, the program runs out of control.

3.3.1 JMP and NOP Instructions are not Inserted Properly after the Instruction to Set the CM10 Bit to 1

Check whether the JMP.B instruction is inserted immediately after the instruction to set the CM10 bit to 1 (stop mode), then four or more NOP instructions are inserted.
The instruction queue prefetches instructions after the instruction to set the CM10 bit to 1. Therefore when entering stop mode, a pre-fetched instruction may be executed before entering stop mode or before an interrupt handler for exiting from stop mode.

Solution:

When entering stop mode, insert the JMP.B instruction immediately after the instruction to set the CM10 bit to 1, then insert four or more NOP instructions.

The following is an example program for entering stop mode:

```
Program Example:  FSET   I
                  BSET   0, CM1   ;  Enter stop mode
                  JMP.B  L2       ;  Insert a JMP.B instruction
         L2:
                  NOP             ;  At least four NOP instructions
                  NOP
                  NOP
                  NOP
```

As shown in the example program, put the instruction to set the I flag to 1 immediately before the instruction to set the CM10 bit to 1, then an interrupt request will not be accepted before entering stop mode.

Application Note:

- Stop Mode Setup (REJ05B1171)

❖ When you are not sure if this is the cause of the issue:
Check the program before and after the instruction to set the CM10 bit to 1. If the program does not follow the procedure described in the Solution above, this item may be the cause of the issue.

3.4 Unexpected Reset Occurrence

Example:

- A reset occurs when exiting from stop mode if entering stop mode from 40 MHz OCO mode.

3.4.1 Transition is Made from 40 MHz OCO Mode to Stop Mode when Using the Voltage Detector

Check whether transition is made from 40 MHz OCO mode to stop mode when using the voltage detector.
When using the voltage detector, if exiting from stop mode and entering 40 MHz OCO mode, some error may be introduced temporarily into the detection voltage of the voltage detector, and a reset or an interrupt may occur on a voltage outside the range of the electrical characteristics.

Products applied this item to:
R5F3650ENFA/FB, R5F3650EDFA/FB, R5F36506NFA/FB, R5F36506DFA/FB
(Technical update: TN-16C-A177A/E)

Solution:
Do not enter stop mode when using the voltage detector and 40 MHz OCO mode. When entering stop mode, enter any mode other than 40 MHz OCO mode first, then enter stop mode.

Technical Update:

- TN-16C-A177A/E

❖ When you are not sure if this is the cause of the issue:
When using the voltage detector, read the set value in the system clock related register to see the CPU clock operating mode by a program or debugger such as E8a emulator immediately before executing the instruction to set the CM10 bit to 1 (stop mode). If the operating mode is 40 MHz OCO mode when the voltage detector is used, this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.4 Examining Codes where an Interrupt Occurs
- 4.5 Investigating the Cause when the System Operates Incorrectly
- 4.6 Determining the Reset Source

3.5 Unexpected Interrupt Occurrence

Example:

- When entering stop mode from 40 MHz OCO mode and exiting from stop mode, an interrupt occurs.

3.5.1 Transition is made from 40 MHz OCO Mode to Stop Mode when Using the Voltage Detector

Check whether transition is made from 40 MHz OCO mode to stop mode when using the voltage detector.
When using the voltage detector, if exiting from stop mode and entering 40 MHz OCO mode, some error may be introduced temporarily into the detection voltage of the voltage detector, and a reset or an interrupt may occur on a voltage outside the range of the electrical characteristics.

Products applied this item to:
R5F3650ENFA/FB, R5F3650EDFA/FB, R5F36506NFA/FB, R5F36506DFA/FB
(Technical update: TN-16C-A177A/E)

Solution:
Do not enter stop mode when using the voltage detector and 40 MHz OCO mode. When entering stop mode, enter any mode other than 40 MHz OCO mode first, then enter stop mode.

Refer to “3.4.1 Transition is Made from 40 MHz OCO Mode to Stop Mode when Using the Voltage Detector” for reference materials.

4. Analysis Methods

This chapter describes the analysis methods with the following development environment:

- Integrated development environment: High-performance Embedded Workshop (HEW)
- Debugger: E8a emulator, E100 emulator

4.1 Examining a Register Setting

This section introduces methods to confirm that a register or variable is set as expected.

4.1.1 Examining a Register Setting Using a Debugger

With a debugger, the value in a register or variable can be checked at a given point. Examine a values in a register before and after setting the register to see if the value is set as expected.

Procedure

(1) Set a breakpoint on the code for setting the register to be examined.
However if there is a code to set the PRC2 bit in the PRCR register to 1 before the code for setting a register, set breakpoints on the code for setting the PRC2 bit and after the code for setting the register.

(2) Run the program and halt at the breakpoint.

(3) Use the memory widow or I/O window to check the value before setting the register.

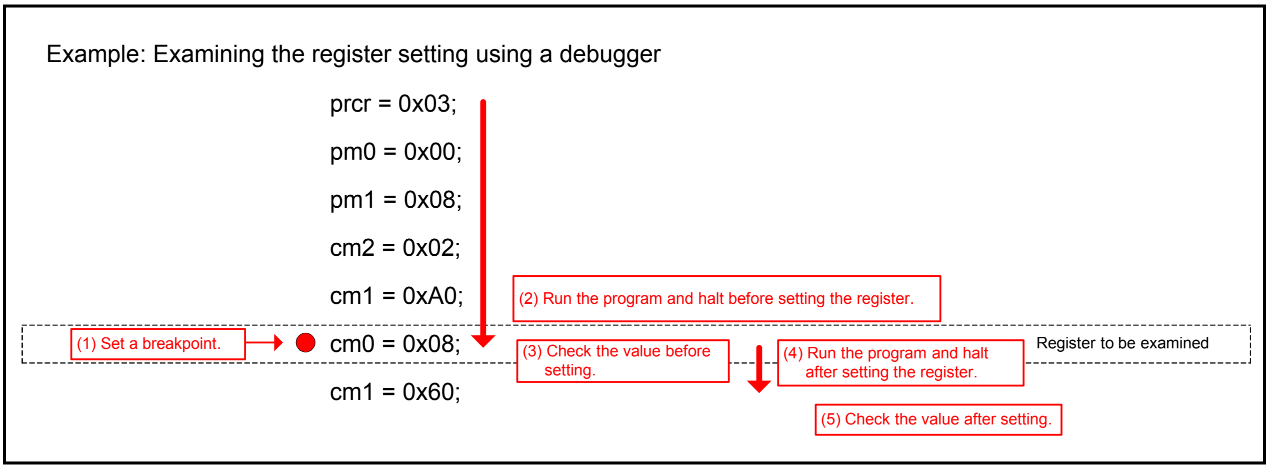
(4) Step through codes to set the register.
However if the register is protected with the PRC2 bit, the register setting cannot be checked with this procedure. Do not halt the program between the code to set the PRC2 bit and the next code.

(5) Check the value after setting the register using the memory widow or I/O window to see if the value is set as expected.

Figure 4.1 shows the Examining the Register Setting Using a Debugger and Figure 4.2 shows Examining the Setting of the Register Protected by the PRC2 Bit.

Explanation

With step (5) above, if the value is not set as expected, the register may be write protected or the specific setting procedure may have to be followed. Refer to the User's Manual: Hardware to confirm the condition for the register setting and its setting procedure.

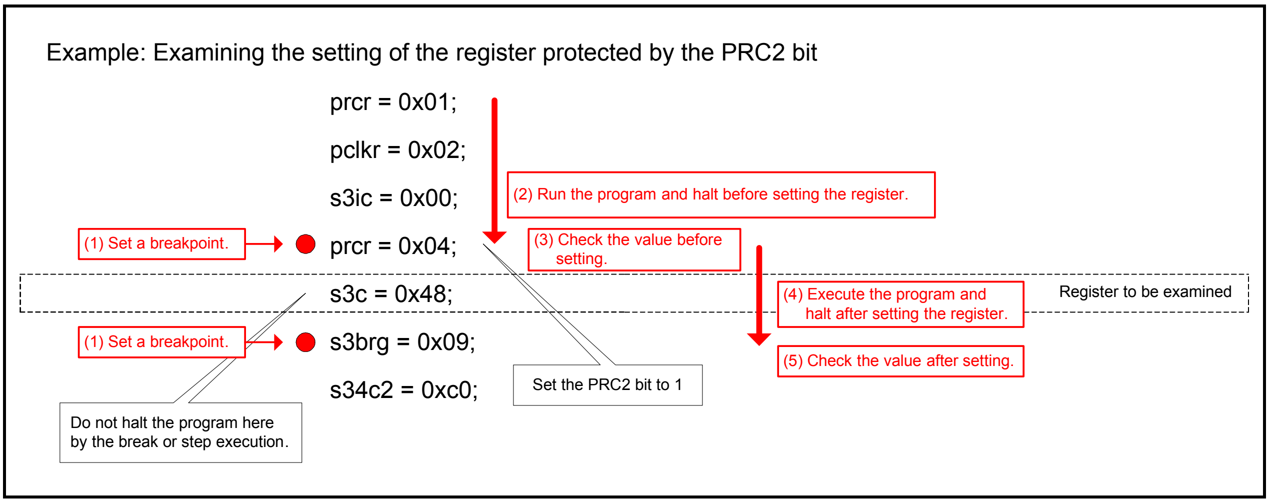


Example: Examining the register setting using a debugger

```
prcr = 0x03;
pm0 = 0x00;
pm1 = 0x08;
cm2 = 0x02;
cm1 = 0xA0;
cm0 = 0x08;
cm1 = 0x60;
```

(1) Set a breakpoint.
(2) Run the program and halt before setting the register.
(3) Check the value before setting.
(4) Run the program and halt after setting the register.
Register to be examined
(5) Check the value after setting.

Figure 4.1 Examining the Register Setting Using a Debugger


Example: Examining the setting of the register protected by the PRC2 bit
(1) Set a breakpoint.
(2) Run the program and halt before setting the register.
(3) Check the value before setting.
Register to be examined
(4) Execute the program and halt after setting the register.
(1) Set a breakpoint.
(5) Check the value after setting.
Set the PRC2 bit to 1
Do not halt the program here by the break or step execution.

```
prcr = 0x01;
pclkr = 0x02;
s3ic = 0x00;
prcr = 0x04;
s3c = 0x48;
s3brg = 0x09;
s34c2 = 0xc0;
```

Figure 4.2 Examining the Setting of the Register Protected by the PRC2 Bit

4.1.2 Examining a Register Setting Using an Oscilloscope

Add test codes to read the register after the code for setting the register, and if the register value read is an expected value, then output a high level signal from the test port. Check the test port state with the oscilloscope to see if an expected value is set. For test ports, use ports which do not affect the system by setting the direction to output.

Procedure

(1) Add codes to output the test result on the test ports in the user program.
Figure 4.3 shows an Example of Adding Test Codes.

(2) Run the program and check the test port states with an oscilloscope.

Explanation

Examine changes of test ports in step (2) above with an oscilloscope.
The following are criteria for judging the result when executing test codes shown in Figure 4.3.

When the output is as expected:
High level signals are output from all test ports as shown in Figure 4.4. The register is set correctly. Thus this is not the cause of the issue.

When the output is not as expected:
A high level signal is not output from test port (D) as shown in Figure 4.5. The register may be write protected or the specific setting procedure may have to be followed. Refer to the User's Manual: Hardware to confirm the condition for the register setting and its setting procedure.

When the program stopped or ran out of control after setting the register
High level signals may not output from test ports (B) and (D) as shown in Figure 4.6. Refer to the User's Manual: Hardware to check whether the register setting procedure, recommended operating conditions of electrical characteristics, and related notes are correctly followed.

Example: Checking that 98h is set to system clock control register 0 (CM0)

Before adding codes

```
        cm0 = 0x98;                         --- Register to be examined
```

Add processing

After adding codes

```
        p10_0 = 1;                          --- (A) Output on a test port      Added
        cm0 = 0x98;                         --- Register to be examined
        p10_1 = 1;                          --- (B) Output on a test port
        if(CM0 == 0x98){                    --- (C) Check the set value        Added
                p10_2 = 1;                  --- (D) Output on a test port
        }
```

* For the test port, use ports which do not affect the system by setting the direction to output.

Figure 4.3 Example of Adding Test Codes

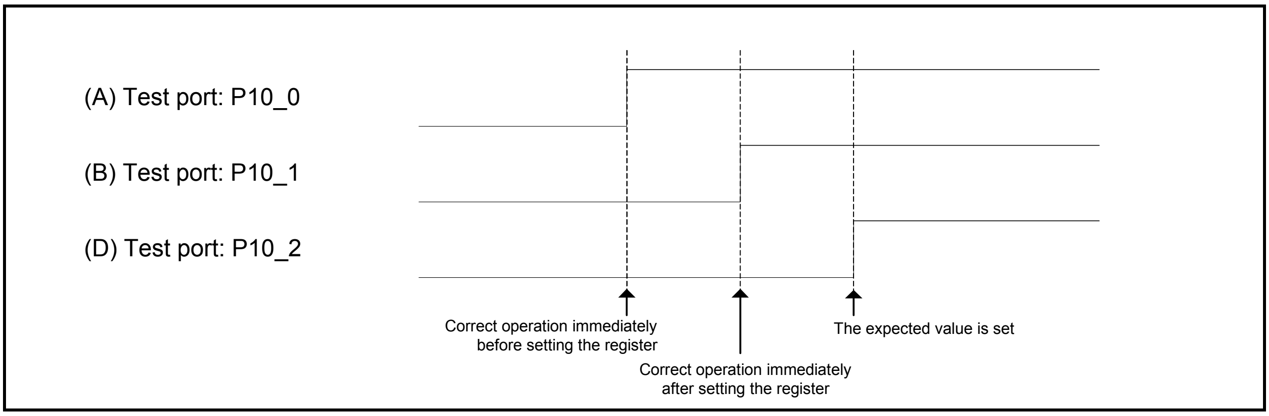

(A) Test port: P10_0
(B) Test port: P10_1
(D) Test port: P10_2
Correct operation immediately before setting the register
The expected value is set
Correct operation immediately after setting the register

Figure 4.4 Waveform when the Value is Set as Expected

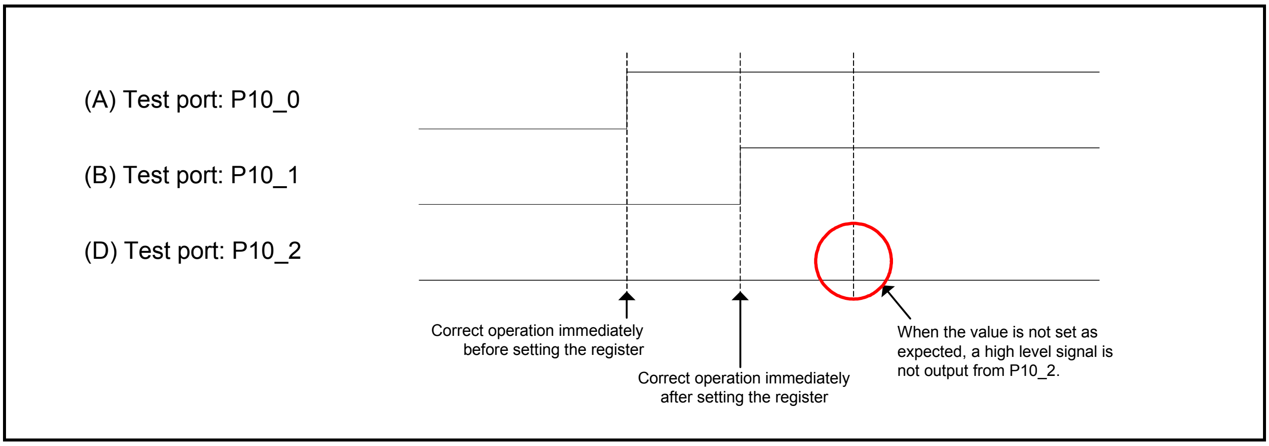

(A) Test port: P10_0
(B) Test port: P10_1
(D) Test port: P10_2
Correct operation immediately before setting the register
When the value is not set as expected, a high level signal is not output from P10_2.
Correct operation immediately after setting the register

Figure 4.5 Waveform when the Value is not Set as Expected

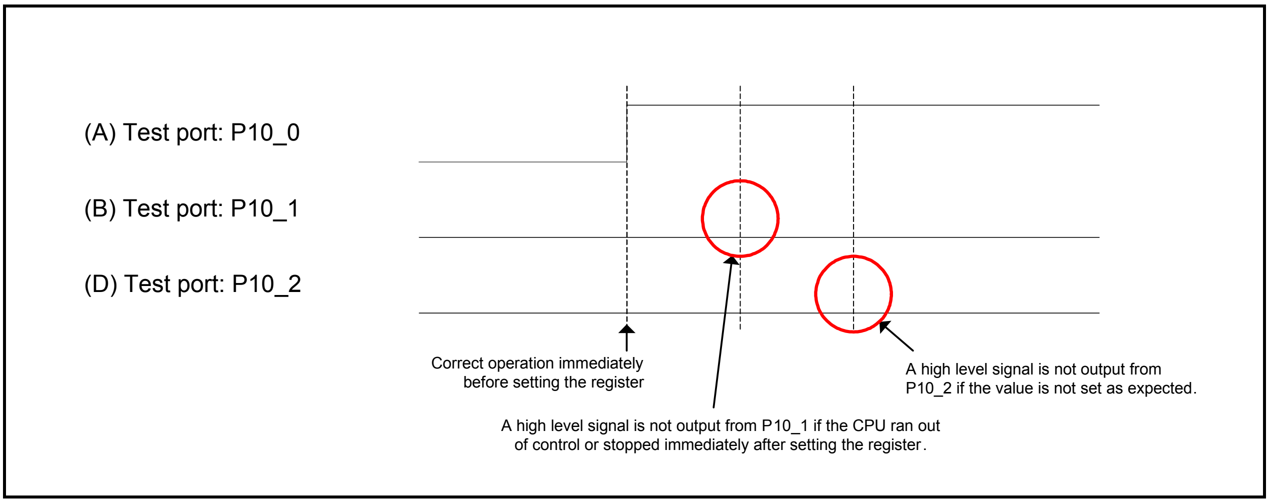

(A) Test port: P10_0
(B) Test port: P10_1
(D) Test port: P10_2
Correct operation immediately before setting the register
A high level signal is not output from P10_2 if the value is not set as expected.
A high level signal is not output from P10_1 if the CPU ran out of control or stopped immediately after setting the register.

Figure 4.6 Waveform when the Program Stopped or Ran Out of Control after Setting a Register

4.2 Checking Oscillation of Crystals/Ceramic Resonators

This section describes the method to examine oscillation of crystals/ceramic resonators.

4.2.1 Checking XOUT/XCOUT Using an Oscilloscope

Check the XOUT pin for the main clock and the XCOUT pin for the sub clock to see whether the connected crystals/ceramic resonators oscillate.

Procedure

(1) Run the program and check the XOUT pin for the main clock and the XCOUT pin for the sub clock using an oscilloscope.

Explanation

By doing step (1) above, if the oscillation did not appear correctly, the procedure to oscillate the clock may be wrong. Refer to the User’s Manual: Hardware to check whether the register setting procedure, recommended operating conditions of the electrical characteristics, and related notes are followed correctly.

When the cause is still not determined, matching may not achieved. Contact your crystal/ceramic resonator manufacturer for the appropriate oscillation parameters.

4.3 Checking the Frequency of the Operating CPU Clock

This section describes methods to see whether the CPU clock is configured correctly.

4.3.1 Checking BCLK in Memory Expansion Mode Using an Oscilloscope

When using memory expansion mode, the frequency output from the BCLK pin is same as the frequency of the CPU clock. Set the processor mode to memory expansion mode and see the frequency output from the BCLK pin using an oscilloscope.

When in memory expansion mode, signals will be output from the other external bus pins as well. Make sure this does not affect your system before changing the mode.

Procedure

(1) Add codes to set the mode to memory expansion mode (set bits PM01 and PM00 in the PM0 register to 01b) in the user program.

(2) Run the program and check the BCLK pin with an oscilloscope.

Explanation

By doing step (2) above, if the frequency is not as expected, check whether the system clock related registers are set correctly. To check registers, refer to 4.1 Examining a Register Setting.

4.3.2 Checking f1 on CLKOUT Using an Oscilloscope

The clock output function can be used in single-chip mode and output f1, f8, f32, or fC from the CLKOUT pin. f1 is the clock same as the CPU clock before passing through the divider.

Use the clock output function to output f1 from the CLKOUT pin and check the frequency output from the CLKOUT pin using an oscilloscope.

The clock output from the CLKOUT pin must be up to 25 MHz. If the clock exceeds 25 MHz, use f8 (set bits CM01 and CM00 in the CM0 register to 10b).

Make sure that output from the CLKOUT pin does not affect your system before performing this method.

Procedure

(1) Add codes to output f1 from the CLKOUT pin (set the PCLK5 bit in the PCLKR register to 1) in the user program. When f1 exceeds 25 MHz, output f8 instead of f1.

(2) Run the program and check the CLKOUT pin with an oscilloscope.

Explanation

With the result of step (2) above, take the specified division into consideration. Then if the frequency is not as expected, check whether the system clock related registers are set correctly. To check registers, refer to 4.1 Examining a Register Setting.

4.3.3 Checking Timer A Pulse Output Using an Oscilloscope

When outputting pulses with f1 as the timer A count source and 0 as the count value, a half of the f1 frequency is output from the TAiOUT pin (i is the channel used for timer output). f1 is the clock same as the CPU clock before passing through the divider. Use timer A to output a half of the f1 frequency and check the frequency output from the TAiOUT pin using an oscilloscope.
Make sure that output from the TAiOUT pin does not affect your system before performing this method.

Procedure

(1) Add codes to output a half of the f1 frequency with timer A.
Timer A settings
- Operating mode: Timer mode
- Pulse output function: Output
- Count source: f1
- Timer register: 0000h
- Count operation: Start

(2) Run the program and check the TAiOUT pin with an oscilloscope.

Explanation

Double the frequency checked in step (2) above. With the frequency, take the specified division into consideration. Then if the frequency is not as expected, check whether the system clock related registers are set correctly. To check registers, refer to 4.1 Examining a Register Setting.

4.4 Examining Codes where an Interrupt Occurs

This section describes methods to examine codes to see if an unexpected interrupt occurs.

4.4.1 Examining Codes Using the Trace Function of the ICE (E100)

The ICE can trace codes when running the program. Examine the traced codes where an interrupt occurred.

Procedure

(1) Set a breakpoint on the first code of the interrupt handling to be examined.
(2) Run the program and halt at the breakpoint.
(3) Open the trace window and see if the timing the interrupt occurred is as expected.

Explanation

With step (3) above, if the interrupt occurred at an unexpected point, check the following things:

- A few instructions immediately before the interrupt occurred may have a problem. Refer to the User's Manual: Hardware to check the conditions and the procedure for setting the register.
- When using an interrupt that occurs from an external signal being received, an signal may be input at an unexpected timing. Use an oscilloscope to see if an unexpected signal is input on the receive pin.

4.4.2 Examining Codes Using a Debugger

By stepping through return processing (REIT) from the interrupt handling, the program can be halted at the next code of the code where the interrupt occurred. Then the point where the interrupt occurred can be determined.

Procedure

(1) Set a breakpoint on the return instruction (REIT) of the interrupt handling.
(2) Run the program and halt at the breakpoint.
(3) Step through the return processing (REIT) to check the code immediately before the address to return.

Explanation

The interrupt occurred immediately after the code checked in step (3) above was executed. If the point where the interrupt occurred is an unexpected point, check the following things:

- A few instructions immediately before the interrupt occurred may have a problem. Refer to the User's Manual: Hardware to check the conditions and the procedure for setting the register.
- If using an interrupt which occurs when an external signal is received, an signal may be input at an unexpected timing. Use oscilloscope to see if an unexpected signal is input on the receive pin.

4.5 Investigating the Cause when the System Operates Incorrectly

This section describes methods to examine codes to see if the CPU runs out of control or if a reset occurs when the system does not operate correctly.

4.5.1 Examining Codes for Unexpected Program Operation with ICE (E100)

When the program runs out of control, it often refers an address where no program is placed and executes 00h or FFh as an instruction. 00h is the BRK instruction and FFh is an undefined instruction. Then when either of the instructions is executed, an interrupt for the instruction occurs.
Examine codes to see if the program runs out of control due to an interrupt for the BRK instruction or an undefined instruction.

Procedure

(1) Set breakpoints on the first codes of the interrupt handling for the BRK instruction and an undefined instruction to be examined.
(2) Run the program.

Explanation

After performing step (2) above, if the program stops at either of the breakpoint set, the program may run out of control. Refer to the User’s Manual: Hardware to check whether the register setting procedure, recommended operating conditions of the electrical characteristics, and related notes are followed correctly.

4.5.2 Examining Codes for Unexpected Program Operation with an Oscilloscope

When the program runs out of control, it often refers an address where no program is placed and executes 00h or FFh as an instruction. 00h is the BRK instruction and FFh is an undefined instruction. Then when either of the instructions is executed, an interrupt for the instruction occurs.
Add codes to output a high level signal on the test port to the start of the interrupt handling for the BRK instruction and an undefined instruction. Determine whether the program runs out of control by checking the output on the test port when running the program.
Or add codes to invert a signal on the test port to the main routine executed regularly. Determine if the output from the test port is inverted continuously when running the program.
For the test port, use a port which does not affect the system by setting the direction to output.

Procedure

(1) Add codes to output a high level signal on the test port to the start of the interrupt handling for the BRK instruction and an undefined instruction.
Or add codes to invert a signal on the test port to the main routine.
(2) Run the program and check the test port state with an oscilloscope.

Explanation

With step (2) above, if an interrupt for the BRK instruction or an undefined instruction occurred and a high level signal was output from the test port, or if a signal on the test port was not inverted in cycles of the main routine execution, the program may run out of control. Refer to the User’s Manual: Hardware to check whether the register setting procedure, recommended operating conditions of the electrical characteristics, and related notes are correctly followed.

4.5.3 Checking a Reset in the Program Using the Oscilloscope

When a reset occurs, the port direction register is reset to 00h (value after reset) and becomes an input port. Set the test port to output a high level signal and check whether the port becomes an input port using an oscilloscope.

For the test port, use a port which does not affect the system by setting the direction to output.

Procedure

(1) Pull down the pin used as the test port.

(2) Add codes which output a high level signal from the test port immediately after the reset start.

(3) Run the program and check the test port state with the oscilloscope.

Explanation

With step (3) above, check if a low pulse is output from the test port. If a low pulse is output as shown in Figure 4.7, then a reset occurred. In that case, refer to 4.6.1 Checking the Reset Source Determine Register Using an Oscilloscope to determine the reset source.

Example: Waveforms when a reset occurs

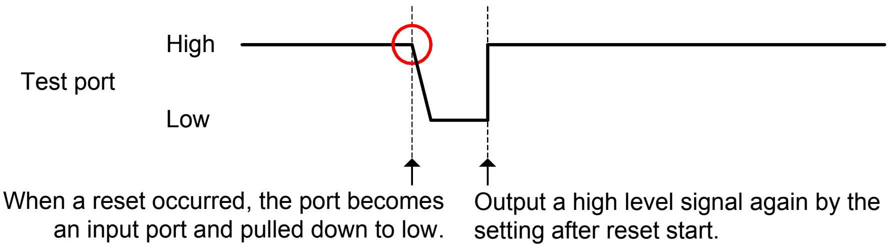

High
Test port
Low
When a reset occurred, the port becomes an input port and pulled down to low.
Output a high level signal again by the setting after reset start.

Figure 4.7 Waveform on the Test Port when a Reset Occurs

4.6 Determining the Reset Source

4.6.1 Checking the Reset Source Determine Register Using an Oscilloscope

When a reset occurs, the flag in the reset source determine register changes according to the reset source. The following is reset sources which can be determined by the reset source determine register.

- Power-on reset or voltage monitor 0 reset
- Hardware reset
- Software reset
- Watchdog timer reset
- Voltage monitor 1 reset
- Voltage monitor 2 reset
- Oscillator stop detect reset

Output the value in the reset source determine register on the test port immediately after the reset start to determine the reset source.
For the test port, use a port which does not affect the system by setting the direction to output.

Procedure

(1) Add the following codes immediately after the reset start:
- Code to output the value in the reset source determine register on the test port.
- Code to set the cold start/warm start discrimination flag in the reset source determine register to 1.

(2) Run the program and check the test port state with the oscilloscope.

Explanation

With step (2) above, determine the reset source.

4.7 Checking Transition to Wait Mode

This section describes a method to check if the transition to wait mode is made.

4.7.1 Checking Current Consumption

After entering wait mode, current consumption is reduced. After executing the wait instruction, measure current consumption to see if it is close to the expected value. This can be a criterion to determine if the transition to wait mode has been made.

Procedure

(1) Run the program until the wait instruction is executed.
(2) Measure current consumption.

When measuring current consumption, stop the peripheral function used to exit wait mode or disable interrupts used by that peripheral function, so the MCU does not exit wait mode.

Explanation

If the measured current consumption in step (2) above is not as expected, transition to wait mode may not be made.

4.8 Checking Transition to Stop Mode

This section describes a method to check if the transition to stop mode is made.

4.8.1 Checking Current Consumption

After entering stop mode, current consumption is reduced. Set the CM10 bit in the CM1 register to 1 (all clocks off (stop mode)) and measure current consumption to see if it is close to the expected value. This can be a criterion to determine if the transition to stop mode has been made.

Procedure

(1) Run the program until the code to set the CM10 bit to 1 is executed.
(2) Measure current consumption.

When measuring current consumption, stop the peripheral function used to exit stop mode or disable interrupts used by that peripheral function, so the MCU does not exit stop mode.

Explanation

If the measured current consumption in step (2) above is not as expected, the transition to stop mode may not be made.

4.8.2 Checking XOUT/XCOUT Using the Oscilloscope

After entering stop mode, the clock stops oscillating. Check pins XOUT and XCOUT with an oscilloscope to see if crystals/ceramic resonators connected to these pins stop. This can be a criterion to determine if the transition to stop mode has been made.

Procedure

(1) Run the program until the code to set the CM10 bit in the CM1 register to 1 (all clocks off (stop mode)) is executed.
(2) Check pins XOUT and XCOUT with an oscilloscope.

When checking pins XOUT and XCOUT, stop the peripheral function used to exit stop mode or disable interrupts used by that peripheral function, so that the MCU does not exit stop mode.

Explanation

With step (2) above, if the clock did not stop oscillating, the transition to stop mode was not made.

5. When the Cause of the Issue Cannot be Found/Determined

5.1 When No Solution can be Found

When you cannot find any solution for the encountered issue, contact the Renesas distributor or sales representative in your area. The support team will investigate the cause of the issue based on the information provided.

Please provide the following information when making an inquiry:

(1) MCU part number
(2) Goal to be achieved
(3) Issue encountered
(4) Operating frequency (CPU clock)
(5) Frequency of crystals/ceramic resonators connected
(6) Power-supply voltage
(7) Temperature
(8) Reproductivity
(9) Dependencies (voltage dependence, frequency dependence, board dependence)
(10)Number of chips with the issue (pcs/pcs)
(11)Frequency of the issue occurrence (times/hour)
(12)Peripheral function in which the issue occurred
(13)Development phase (development, mass production)
(14)Setting values of related registers
(15)Simulator and emulator information
(16)Compiler version

6. Reference Documents

M16C/63 Group User’s Manual: Hardware Rev.2.20
M16C/64A Group User’s Manual: Hardware Rev.2.10
M16C/64C Group User’s Manual: Hardware Rev.1.10
M16C/65 Group User’s Manual: Hardware Rev.2.10
M16C/65C Group User’s Manual: Hardware Rev.1.10
M16C/6C Group User’s Manual: Hardware Rev.2.10
M16C/5LD Group, M16C/56D Group User’s Manual: Hardware Rev.1.20
M16C/5L Group, M16C/56 Group User’s Manual: Hardware Rev.1.10
M16C/5M Group, M16C/57 Group User’s Manual: Hardware Rev.1.10
The latest versions can be downloaded from the Renesas Electronics website.

Technical Update/Technical News
The latest information can be downloaded from the Renesas Electronics website.

Website and Support

Renesas Electronics website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry

Revision History	M16C/63, 64A, 64C, 65, 65C, 6C, 5LD, 56D, 5L, 56, 5M, and 57 Groups Troubleshooting for Clocks in Development

Rev.	Date	Description	
		Page	Summary
1.10	Dec. 2, 2013	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics 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 appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 LanGao Rd., Putuo District, Shanghai, China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 3.0