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

Troubleshooting for Serial Communications in Development

R01AN1622EJ0120
Rev. 1.20
Dec.15, 2017

Abstract

This application note describes troubles in development and their solutions regarding clock synchronous/ asynchronous serial communications 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

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

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 Using Clock Synchronous Serial I/O Mode	Chapter 1.
Troubles when Using Clock Asynchronous Serial I/O (UART) Mode	Chapter 2.
Troubles when Using the E8a Emulator Debugger	Chapter 3.

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

Proper Name/Meaning	Abbreviation/Acronym
Clock asynchronous serial I/O mode	UART mode
High-impedance	Hi-Z
High-performance Embedded Workshop	HEW
Device which the M16C MCU communicates with	Target device

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 product used.

Contents

1. Troubles when Using Clock Synchronous Serial I/O Mode

Table 1.1 lists 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	Transmit Data is not Output from the TXDi Pin	Transmission is not Enabled	1.1.1
		CTS Function is Enabled and CTSi Pin is Driven High	1.1.2
		Transmit Data was not Set	1.1.3
		TXD2 Pin is not Pulled Up when Using UART2	1.1.4
		TXDi Pin Used for N-Channel Open Drain Output is not Pulled Up	1.1.5
		Clock is not Provided when an External Clock is Selected	1.1.6
		Output Collides with the Other Peripheral Function	1.1.7
		UART6 or UART7 is Used in Memory Expansion Mode	1.1.8
		U4MR Register is not Set Properly when Using UART4	1.1.9
1.2	Unexpected Transmit Data is Output	Serial Interface is Disabled	1.2.1
		With an External Clock, CLKi Pin Level is Incorrect While Transmit/Receive Condition is Met	1.2.2
1.3	Missing Transmit Data	Empty Flag is Verified when Transmit Data is Set	1.3.1
		Transmit Buffer is Full when the Next Data is Set	1.3.2
1.4	Data cannot be Received Correctly in the Target Device	Target Device is not Properly Connected	1.4.1
		Communication Signal does not Satisfy the Specification of the Target Device	1.4.2
		Communication Format is Incorrect	1.4.3
		Bit Slippage Occurred	1.4.4
1.5	Transmission Cannot be Stopped even after Transmission is Disabled	Transmission is Disabled with the Transmit Enable Bit	1.5.1
1.6	Data Cannot be Received/Receive Interrupt does not Occur	Transmission is not Enabled	1.6.1
		Reception is not Enabled	1.6.2
		Transmit Data is not Set	1.6.3
		Target Device is Incorrectly Connected	1.6.4
		RXDi Pin is Set as an Output Port	1.6.5
		Receive Signal does not Satisfy the Electrical Characteristics	1.6.6
		Overrun Error Occurred	1.6.7
		CLKi Pin Level is Incorrect when the Transmit Condition is Met with an External Clock	1.6.8
		Noise or Incorrect Signal is Input to the CLKi pin	1.6.9
		Bit Slippage Occurred	1.6.10
1.7	Overrun Error Occurred	Receive Data is not Read in Time	1.7.1
1.8	Communication is not Available After a Communication Error Occurred	Serial Interface is not Initialized After a Communication Error Occurred	1.8.1
1.9	CLKi Pin Outputs Unexpected Low Level Signal	CLKi Pin is Set to N-channel Open Drain Output	1.9.1
—	When the Cause of the Issue Cannot be Found/Determined	—	5.

1.1 Transmit Data is not Output from the TXDi Pin

Example:
- TXDi pin level is not changed at all even if the transmit data is set.

1.1.1 Transmission is not Enabled

Check whether the transmit data was set to the UARTi transmit buffer register (UiTB) while transmission was disabled (TE bit in the UiC1 register is 0). Transmission must be enabled (TE bit is 1) when outputting transmit data.

Solution:
When outputting transmit data, set the TE bit in the UiC1 register to 1 (transmission enabled).

Application Note:
- Operation of Serial I/O (Transmission in Clock-Synchronous Serial I/O Mode) (R01AN0540)

❖ When you are not sure if this is the cause of the issue:
Check the setting value of the TE bit in the UiC1 register by a program or debugger such as the E8a emulator immediately before setting transmit data (write to the UiTB register).
When the TE bit is 0 (transmission disabled), this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.4 Examining Signals when Transmitting

1.1.2 CTS Function is Enabled and CTSi Pin is Driven High

Check whether the CTS function is enabled (CRD bit in the UiC0 register is 0) even if the function is not used. When the CTS function is used, also check whether the RTSi pin (1) of the target device which connects to the CTSi pin is driven low.
When the CTS function is enabled, the CTSi pin must be driven low as the transmission start condition.
The CTS function is enabled after a reset.

Solution:
When not using the CTS function, disable the CTS function (CRD bit in the UiC0 register is set to 1).
When using the CTS function, enable the CTS function (CRD bit in the UiC0 register is set to 0) and connect the CTSi pin to the RTSi pin (1) of the target device.

Application Note:
- Operation of Serial I/O (Transmission in Clock-Synchronous Serial I/O Mode) (R01AN0540)

❖ When you are not sure if this is the cause of the issue:
Check the setting value of the CRD bit in the UiC0 register by a program or debugger such as the E8a emulator immediately before setting transmit data (write to the UiTB register).
When the CRD bit is 0 (CTS/RTS function enabled), this item may be the cause of the issue. Then check the level of the CTSi pin using an oscilloscope or other tools. If the CTSi pin level is high, this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.4 Examining Signals when Transmitting

Note:
1. The pin name differs depending on the target device.

1.1.3 Transmit Data was not Set

Check whether transmit data is set to the UARTi transmit buffer register (UiTB).
When outputting transmit data, the TI bit in the UiC1 register must be set to 0 (data present in UiTB register). The TI bit automatically becomes 0 by setting transmit data to the UiTB register.

Solution:
When outputting transmit data, write transmit data to the UiTB register.

Application Note:
- Operation of Serial I/O (Transmission in Clock-Synchronous Serial I/O Mode) (R01AN0540)

❖ When you are not sure if this is the cause of the issue:
Check whether the program has the instruction to set transmit data to the UiTB register. If not, this is the cause of the issue.
Also refer to the following section in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting

1.1.4 TXD2 Pin is not Pulled Up when Using UART2

When using the M16C/63, 64A, 64C, 65, 65C, and 6C Group products, check whether the TXD2 pin of UART2 is pulled up. With these products, pins TXD2 and RXD2 of UART2 are N-channel open drain output dedicated pins. When transmitting or receiving data with pins TXD2 and RXD2, these pins must be pulled up with an external circuit.
With the M16C/5LD, 56D, 5L, 56, 5M, and 57 Group products, pins TXD2 and RXD2 of UART2 are CMOS output pins. Thus pull-up processing is not required.

Solution:
When transmitting or receiving data using N-channel open drain output pins TXD2 and RXD2 of UART2, pull-up these pins with an external circuit.

❖ When you are not sure if this is the cause of the issue:
Check pins TXD2 and RXD2 of UART2 when the MCU is reset using an oscilloscope or other tools. When pins TXD2 and RXD2 are driven low during the reset, pull-up may not be processed correctly. If pull-up is not performed correctly, this is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting
- 4.6 Checking Pull-Up for the N-Channel Open Drain Output Pin

1.1.5 TXDi Pin Used for N-Channel Open Drain Output is not Pulled Up

Check whether pins TXDi/SDAi and SCLi are pulled up when they are set as N-channel open drain output (NCH bit in the UiC0 register is 1).
N-channel open drain output does not output high. Therefore pins need to be pulled up with an external circuit.
For N-channel open drain output with pins TXDi/SDAi and SCLi by setting the NCH bit, the function only sets the P-channel transistor of the appropriate pin always off and it does not change pins TXDi/SDAi and SCLi to open drain output completely. For details on the allowable input voltage range, refer to the Electrical Characteristics chapter in the User's Manual: Hardware for the product used.

Solution:
When using pins TXDi/SDAi and SCLi as N-channel open drain output, pull-up these pins with an external circuit.

❖ When you are not sure if this is the cause of the issue:
Check pins TXDi/SDAi and SCLi when the MCU is reset using an oscilloscope or other tools. When pins TXDi/SDAi and SCLi are driven low during the reset, pull-up may not be processed correctly. If pull-up is not performed correctly, this is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting
- 4.6 Checking Pull-Up for the N-Channel Open Drain Output Pin

1.1.6 Clock is not Provided when an External Clock is Selected

Check whether a transmit/receive clock of the target device is provided to the CLKi pin when an external clock is selected (CKDIR bit in the UARTi transmit/receive mode register (UiMR) is 1). When an external clock is selected, transmission and reception are performed with reference to the clock provided to the CLKi pin. Thus if the clock is not provided to the CLKi pin, transmission and reception cannot be performed.

Solution:
When an external clock is selected, provide a transmit/receive clock to the CLKi pin.

❖ When you are not sure if this is the cause of the issue:
Check whether a transmit/receive clock is provided to the CLKi pin using an oscilloscope. If the clock is not provided, this is the cause of the issue.
Also refer to the following section in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting

1.1.7 Output Collides with the Other Peripheral Function

The output may collide with a pin output from the other peripheral function. Some pins can output signals for multiple functions (multi-function pin). For those pins, if multiple peripheral functions output signals simultaneously, the signals collide. In that case, transmit data may not output correctly from the TXDi pin.

Solution:
When transmit data is output from the TXDi pin, do not output from the other peripheral function which shares the pin with.

❖ When you are not sure if this is the cause of the issue:
Modify the program to disable the serial interface (set bits SMD2 to SMD0 in the UiMR register to 000b) and run it to see the TXDi pin using an oscilloscope or other tools. Then pull-up or pull-down the TXDi pin and check whether the pin state is Hi-Z. If the state is not Hi-Z, this may be the cause of the issue.
Also refer to the following section in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting

1.1.8 UART6 or UART7 is Used in Memory Expansion Mode

Check whether UART6 or UART7 is used in memory expansion mode or microprocessor mode. With memory expansion mode or microprocessor mode, pins assigned to UART6 or UART7 function as bus control pins. Thus these pins cannot be used for serial interface. It is the same when an 8-bit data bus is used (D8 to D15 not used) or when CS2/CS3 output is disabled.

Solution:
Do not use UART6 or UART7 in memory expansion mode or microprocessor mode.

❖ When you are not sure if this is the cause of the issue:
Check the setting values of bits PM01 and PM00 in the PM0 register by a program or debugger such as the E8a emulator immediately before setting transmit data (write to the UiTB register). When bits PM01 and PM00 are a value other than 00b (single-chip mode), the operating mode is set to memory expansion mode or microprocessor mode. In that case and if UART6 or UART7 is used, this is the cause of the issue.
Also refer to the following section in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting

1.1.9 U4MR Register is not Set Properly when Using UART4

When using UART4 with the M16C/5LD, 56D, 5L, 56, 5M, or 57 Group product, check whether the UART4 transmit/receive mode register (U4MR) is set properly. With the M16C/5LD, 56D, 5L, 56, 5M, and 57 Group products, the U4MR register is protected by the protect register. For details, refer to the Protection chapter in the User's Manual: Hardware for the product used.

Solution:
When setting the U4MR register in the M16C/5LD, 56D, 5L, 56, 5M, or 57 Group product, enable write operation with the protect register first.

❖ When you are not sure if this is the cause of the issue:
Check the setting value of U4MR register by a program or debugger such as the E8a emulator immediately after setting the register. If the value checked is not the value set, this is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.4 Examining Signals when Transmitting

1.2 Unexpected Transmit Data is Output

Examples:
- The transmit data output is not the data set in the UARTi transmit buffer register (UiTB).
- Transmit data is cut off.

1.2.1 Serial Interface is Disabled

Check whether the serial interface is disabled (bits SMD2 to SMD0 in the UiMR register are 000b) during transmission. If the serial interface is disabled, the transmission is canceled at that point.

Solution:
Do not disable the serial interface (set bits SMD2 to SMD0 to 000b) during transmission.

❖ When you are not sure if this is the cause of the issue:
Check whether the program disables the serial interface (bits SMD2 to SMD0 are set to 000b) at any point. In that case this item may be the cause of the issue.

1.2.2 With an External Clock, CLKi Pin Level is Incorrect While Transmit/Receive Condition is Met

Check if the conditions below are satisfied when an external clock is used and transmission/reception is started while no data is in the transmit register (TXEPT bit in UARTi transmit/receive control register 0 (UiC0) is 1).
- The CKPOL bit in the UiC0 register is 0 (transmit data is output at the falling edge of transmit/receive clock and receive data is input at the rising edge), and the CLKi pin is high.
- The CKPOL bit in the UiC0 register is 1 (transmit data is output at the rising edge of transmit/receive clock and receive data is input at the falling edge), and the CLKi pin is low.

The conditions above must be satisfied when an external clock is supplied to the CLKi pin. If the CLKi pin level is incorrect for the transmit/receive condition satisfied, bit slippage may occur.

Solution:
When selecting an external clock and starting transmission/reception while the TXEPT bit in the UiC0 register is 1 (no data present in transmit register):
When the CKPOL bit is 0, satisfy the transmit/receive condition while the CLKi pin is high.
When the CKPOL bit is 1, satisfy the transmit/receive condition while the CLKi pin is low.

❖ When you are not sure if this is the cause of the issue:
Halt the program immediately before the transmit condition is satisfied (e.g. before setting transmit data) by a program or debugger such as the E8a emulator. Then check the CLKi pin level using an oscilloscope.
If the CLK pin level is low when the CKPOL bit is 0, or the CLK pin level is high when the CKPOL bit is 1, this item may be the cause of the issue.
Also refer to the following section in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting

1.3 Missing Transmit Data

Example:

- When data is transmitted continuously, transmit data sometimes has 1-byte data missing.

1.3.1 Empty Flag is Verified when Transmit Data is Set

Check whether the transmit register empty flag (TXEPT bit in UARTi transmit/receive control register 0 (UiC0)) is verified before the next transmit data is set.
If these empty flags are verified immediately after the transmit data is set, they may indicate no data in the transmit register (TXEPT bit is 1) even though the transmit data is set. Then the next data cannot be set and may become missing data since the previous data is still present.

Solution:

When verifying that the transmission is completed , and setting transmit data, verify the interrupt request bit of the transmit interrupt first, and set the next transmit data.
The interrupt source of the transmit interrupt can be selected with the UARTi transmit interrupt source select bit (UiIRS) in UARTi transmit/receive control register 1 (UiC1).

❖ When you are not sure if this is the cause of the issue:
Check whether the transmit register empty flag and then set the next transmit data. In that case this item may be the cause of the issue.
Also refer to the following section in 4. Analysis Methods:

- 4.4 Examining Signals when Transmitting

1.3.2 Transmit Buffer is Full when the Next Data is Set

Check whether the next data is set to the transmit buffer register while the transmit buffer is full (TI bit in the UiC1 register is 0). If the value is set to the UARTi transmit buffer register (UiTB) while data is present in the UiTB register, the value in the register is overwritten and the overwritten data may become missing data.

Solution:

When verifying that the transmit buffer register is empty and setting transmit data, verify the interrupt request bit of the transmit interrupt first, and set the next data.
The interrupt source of the transmit interrupt can be selected with the UiIRS bit in the UiC1 register.

❖ When you are not sure if this is the cause of the issue:
Check whether the program verifies the IR bit for the transmit interrupt first after the value is set to the UiTB register and set the next transmit data. If the IR bit for the transmit interrupt is not verified, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.4 Examining Signals when Transmitting

1.4 Data cannot be Received Correctly in the Target Device

Example:

- The target device cannot receive the transmit data even though it is output correctly.

1.4.1 Target Device is not Properly Connected

Check whether pins TXDi, RXDi, and CLKi in the M16C MCU are connected to the reception pin, the transmission pin, and the clock pin in the target device, respectively.
Also when using the CTS/RTS function, check whether the CTSi/RTSi pin is connected to the corresponding pin.
For proper communication, make sure that pins are correctly connected between the M16C MCU and the target device.

Solution:
Connect pins correctly.

❖ When you are not sure if this is the cause of the issue:
Check the TXDi pin in the M16C MCU and the reception pin in the target device using an oscilloscope to see if the same waveforms appear on the TXDi pin and the reception pin when the transmit data is output.
Also check the RXDi pin in the M16C MCU and the transmission pin in the target device, the CLKi pin in the M16C MCU and the clock pin in the target device, and the CTSi/RTSi pins in both sides to see if same waveforms appear on each pair of pins.
If same waveforms do not appear, the connection may be incorrect or the circuit may be open. Then this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:

- 4.5 Examining Signals when Receiving

1.4.2 Communication Signal does not Satisfy the Specification of the Target Device

Check whether the width or level of the output signal satisfies the rated values, operating conditions, or the other characteristics of the target device.
For proper communication, the communication standards of the target device must be satisfied.

Solution:
Satisfy the communication standards of the target device.

❖ When you are not sure if this is the cause of the issue:
Check signals output from pins TXDi, CLKi, and RTSi using an oscilloscope or other tools. Also check signals input to the target device to see if signal delay occurs due to the capacity of the communication wiring, buffer IC, or others.
If the width or level of the output signal does not meet the communication standards of the target device, this item may be the cause of the issue.

1.4.3 Communication Format is Incorrect

Check whether the I/O polarity or bit order of transmit data being output is correct. If the data format differs between the M16C MCU and the target device, data cannot be received correctly.

Solution:
Specify the I/O polarity and bit order according to the communication format of the target device.

❖ When you are not sure if this is the cause of the issue:
Check the transmit data being output using an oscilloscope to see if the communication format matches the format in the target device. If the formats do not match, this item may be the cause of the issue.

1.4.4 Bit Slippage Occurred

Check whether the internal clock is selected (CKDIR bit in the UiMR register is set to 0) after the target device becomes ready for reception.
When the internal clock is selected for the UART clock, the CLKi pin state is Hi-Z as it is open until the mode is specified. The CLKi pin is driven high after the mode is specified. Therefore the target device may receive the rising edge of the CLKi pin incorrectly depending on the pin state when the CLKi pin is open and this may cause bit slippage.

Solution:
It is recommended to pull up the CLKi pin.
When the CLKi pin is not pulled up, enable reception in the target device after the CLKi pin becomes high.

❖ When you are not sure if this is the cause of the issue:
Halt the program immediately before the target device becomes ready for reception to see the CLKi pin state. If the pin level is low, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.5 Examining Signals when Receiving

1.5 Transmission Cannot be Stopped even after Transmission is Disabled

Example:
- Transmit data is still output even if transmission is disabled.

1.5.1 Transmission is Disabled with the Transmit Enable Bit

Check whether transmission is disabled (TE bit in the UiC1 register is set to 0) while transmit data is output.
Once transmission is started, if transmission is disabled while the transmit data is output, the transmission cannot be stopped. The transmission completes normally and transmission is disabled at the same time the transmit interrupt request is generated.

Solution:
When forcibly terminating the output of transmit data, disable the serial interface (bits SMD2 to SMD0 in the UiMR register are set to 000b).

❖ When you are not sure if this is the cause of the issue:
Check the program how it terminates the transmission during transmit operation. If the method of termination is not by disabling the serial interface (bits SMD2 to SMD0 are 000b), this item may be the cause of the issue.

1.6 Data Cannot be Received/Receive Interrupt does not Occur

Example:
- Data cannot be received even though the receive data is input.
- Reception cannot be completed.

1.6.1 Transmission is not Enabled

Check whether reception is performed while transmission is disabled (TE bit in the UiC1 register is 0). If the operation is reception only, transmission is required. Thus transmission must be enabled (TE bit is 1).

Solution:
When receive data is input, enable transmission (TE bit is set to 1).

Application Note:
- Operation of Serial I/O (Reception in Clock-Synchronous Serial I/O Mode) (R01AN0541)

❖ When you are not sure if this is the cause of the issue:
Check the TE bit immediately before the receive condition is met (receive data is input) by a program or debugger such as the E8a emulator. If the TE bit is 0 (transmission disabled), this item is the cause of the issue.
Also refer to the following section in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.5 Examining Signals when Receiving

1.6.2 Reception is not Enabled

Check whether reception is performed while reception is disabled (RE bit in the UiC1 register is 0). If reception is enabled, check whether the timing to enable reception is before receive data is transmitted by the target device.
Reception must be enabled (RE bit is set to 1) before the receive data is transmitted.

Solution:
When receive data is input, enable reception (RE bit is set to 1).

Application Note:
- Operation of Serial I/O (Reception in Clock-Synchronous Serial I/O Mode) (R01AN0541)

❖ When you are not sure if this is the cause of the issue:
Check the RE bit immediately before the receive condition is met (receive data is input) by a program or debugger such as the E8a emulator. If the RE bit is 0 (reception disabled), this item is the cause of the issue.
Also refer to the following section in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.5 Examining Signals when Receiving

1.6.3 Transmit Data is not Set

Check whether the transmit data is set to the UARTi transmit buffer register (UiTB). Even if the operation is only reception, transmission is required to generate the transmit/receive clock. Therefore transmit data must be set to the UiTB register and the TI bit in the UiC1 register must be set to 0 (data present in the UiTB register).

Solution:
When receive data is input, write transmit data to the UiTB register.

Application Note:
- Operation of Serial I/O (Reception in Clock-Synchronous Serial I/O Mode) (R01AN0541)

❖ When you are not sure if this is the cause of the issue:
Check whether the program sets transmit data to the UiTB register. If not, this item is the cause of the issue.
Also refer to the following section in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.5 Examining Signals when Receiving

1.6.4 Target Device is Incorrectly Connected

Check whether pins TXDi, RXDi, and CLKi are connected to the reception pin, transmission pin, and the clock pin in the target device, respectively.
When using the CTS/RTS function, also check whether the CTSi/RTSi pin is connected to the corresponding pin. For proper communication, make sure that pins are correctly connected between the M16C MCU and the target device.

Solution:
Connect pins correctly.

Refer to “1.4.1 Target Device is not Properly Connected” when you are not sure if this is the cause of the issue or for reference materials.

1.6.5 RXDi Pin is Set as an Output Port

Check whether the I/O port corresponding to the RXDi pin is set to output mode.
The RXDi pin can be used as an I/O port. When the port is set to output mode, the RXDi pin functions as an output port. Then a collision occurs with input data and the RXDi pin cannot receive input data correctly.

Solution:
When inputting receive data, set the bit in the port Pi register corresponding to the RXDi pin to input mode.

❖ When you are not sure if this is the cause of the issue:
Halt the program immediately before receive data is input by changing the program or a debugger such as the E8a emulator. Then pull up or pull down the RXDi pin to see if the pin state is Hi-Z. If not, this item is the cause of the issue.

1.6.6 Receive Signal does not Satisfy the Electrical Characteristics

Check whether the width or level of the receive signal input from the target device satisfies the standards of the electrical characteristics.
For proper communication, communication must be performed compliant with the standards of the electrical characteristics.

Solution:
Make the receive signal input from the target device satisfy the standards of the electrical characteristics.

❖ When you are not sure if this is the cause of the issue:
Check signals input to pins RXDi, CLKi, and CTSi using an oscilloscope or other tools. If the width or level of input signals do not satisfy standards of the electrical characteristics, this item may be the cause of the issue.

1.6.7 Overrun Error Occurred

Check whether an overrun error occurred by receiving the next data before reading data from the receive buffer after data reception.
When an overrun error occurs, receive data becomes undefined. Also the receive complete interrupt does not occur (IR bit in the SiRIC register does not change).

Solution:
After data reception, read the UARTi receive buffer register (UiRB) before receiving the next data.
If an overrun error occurs (OER bit in the UiRB register is 1), disable the serial interface or disable reception, and clear the overrun error. Before restarting reception, read the UiRB register to empty the receive register.

❖ When you are not sure if this is the cause of the issue:
Check whether an overrun error occurred (OER bit in the UiRB register is 1) when reading the UiRB register. If so, this item is the cause of the issue.

1.6.8 CLKi Pin Level is Incorrect when the Transmit Condition is Met with an External Clock

When using an external clock, check whether the conditions listed below are met when starting transmission/reception with no data in the transmit register (TXEPT bit in UARTi Transmit/Receive Control Register 0 register (UiC0) is 1).

- CKPOL bit in the UiC0 register is 0 (transmit data is output at the falling edge of transmit/receive clock and receive data is input at the rising edge) and the CLKi pin is driven high.
- CKPOL bit in the UiC0 register is 1 (transmit data is output at the rising edge of transmit/receive clock and receive data is input at the falling edge) and the CLKi pin is driven low.

When providing an external clock to the CLKi pin, the above conditions must be met. If the transmit condition is met but the CLKi pin level is not correct, bit slippage occurs.

Solution:

When using an external clock and starting transmission/reception while the TXEPT bit is 1 (no data present in transmit register):
When the CKPOL bit is 0, make sure the CLKi pin is driven high to satisfy the transmit/receive condition.
When the CKPOL bit is 1, make sure the CLKi pin is driven low to satisfy the transmit/receive condition.

Refer to “1.2.2 With an External Clock, CLKi Pin Level is Incorrect While Transmit/Receive Condition is Met” when you are not sure if this is the cause of the issue or for reference materials.

1.6.9 Noise or Incorrect Signal is Input to the CLKi pin

Check whether noise or an incorrect signal is input to the CLKi pin when an external clock is selected. If noise or an incorrect signal is input to the CLKi pin while the receive condition is met, the value according to the RXDi pin state at that point is input to the receive buffer and this may cause bit slippage.

Solution:

Design your board to avoid traces which could be a source such as noise or an incorrect signal.

❖ When you are not sure if this is the cause of the issue:
Examine the CLKi pin using an oscilloscope. If noise or an incorrect signal appears on the CLKi pin, this item may be the cause of the issue.
Also refer to the following section in 4. Analysis Methods:

- 4.5 Examining Signals when Receiving

1.6.10 Bit Slippage Occurred

Check whether an internal clock is selected (CKDIR bit in the UARTi transmit/receive mode register (UiMR) is 0) after the target device becomes ready for transmission.
When selecting an internal clock as the UART clock, the CLKi pin state is Hi-Z since it is open until the mode is set. The CLKi pin outputs high after the mode is set. According to the CLKi pin state when it is open, the target device may incorrectly start up and receive the rising edge signal. Then this may cause the bit slippage.

Solution:

It is recommended to pull up the CLKi pin.
When the CLKi pin is not pulled up, enable reception in the target device after the CLKi pin becomes high.

Refer to “1.4.4 Bit Slippage Occurred” when you are not sure if this is the cause of the issue or for reference materials.

1.7 Overrun Error Occurred

Example:

- An overrun occurs after the data is received.

1.7.1 Receive Data is not Read in Time

Check whether the UARTi receive buffer register (UiRB) is read before the next data is input after data reception.
An overrun error occurs if the last bit of the next data is received while data is present in the UiRB register (RI bit in the UiC1 register is 1).
When not using the receive interrupt, if the period from a reception to the next reception is longer than the period to input receive data, the receive register may not be read in time.
When the receive interrupt is used and multiple interrupts are enabled, if an interrupt which has higher priority level than the receive interrupt occurs, the receive register may not be read in time.
When the receive interrupt is used and multiple interrupts are disabled, if receive data is input while another interrupt is processed, and the interrupt handler requires quite a long time for its processing, the receive register may not be read in time.

Solution:

After data is received, read the UiRB register before receiving the next data.
Consider the timing and the method to read the UiRB register depending on the period to input receive data. Also consider using the DMAC for quick data access.

Application Notes:

- Procedure for successive serial I/O transmission/reception using the DMAC (REC05B0109)

❖ When you are not sure if this is the cause of the issue:
Add the program to invert the test port immediately before reading receive data.
Examine the test port and the RXDi pin using an oscilloscope.
If the period to input received data to the RXDi pin is shorter than the period to invert the test port, this item may be the cause of the issue.

1.8 Communication is not Available After a Communication Error Occurred

Example:

- Communication cannot be performed correctly once an error occurred.

1.8.1 Serial Interface is not Initialized After a Communication Error Occurred

Check whether the serial interface is initialized with proper steps after a communication error occurred. When communication is canceled or a communication error occurs, a problem such as bit slippage possibly occurs. Therefore the serial interface needs to be initialized.

Solution:

When canceling the communication or an error occurs, specify the settings as follows:

(1) Disable transmission/reception (set bits TE and RE in the UiC1 register to 0).
(2) Disable the serial interface (set bits SMD2 to SMD0 in the UiMR register to 000b).
(3) Reset bits SMD2 to SMD0.
(4) Enable transmission/reception (set bits TE and RE to 1).

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

Check the program to see if there is a problem with the processing when a communication error occurs.

If the serial interface is not initialized before restarting communication after the error occurred, this item may be the cause of the issue.

1.9 CLKi Pin Outputs Unexpected Low Level Signal

Example:

- Low pulse is output from the CLKi pin for a short time when the serial interface is initialized.

1.9.1 CLKi Pin is Set to N-channel Open Drain Output

With the M16C/64A or M16C/65 Group product, check whether the CLKi pin is set to N-channel open drain output and the port corresponding to the CLKi pin is set to low.
When the Px_x/CLKi pin is used as an input port, the pin state is Hi-Z. However when the following three conditions are all met, the Px_x/CLKi pin is driven low regardless of the value in the Px_x direction register.

- Bits SMD2 to SMD0 in the UiMR register are 000b (serial interface disabled)
- NODC bit in the UiSMR3 register is 1 (CLKi is N-channel open drain output)
- Px_x bit in the Px register is 0 (output level is 0)

Products applied this item:
M16C/64A, M16C/65
(Technical update: TN-16C-A178A/E)

Solution:

When switching from port Px_x to CLKi

(1) Select serial interface mode with bits SMD2 to SMD0 in the UiMR register (set value is other than 000b).
(2) Set the NODC bit in the UiSMR3 register to 1.

When switching from CLKi to port Px_x

(1) Set the NODC bit in the UiSMR3 register to 0.
(2) Disable the serial interface with bits SMD2 to SMD0 in the UiMR register (set value is 000b).

Technical Update:

- TN-16C-A178A/E

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

Check the procedures to set the NODC bit in the UiSMR3 register and bits SMD2 to SMD0 in the UiMR register.
When the port is not set or the port is driven low, if the NODC bit is set before clock synchronous serial I/O mode is selected with bits SMD2 to SMD0, this item may be the cause of the issue.
Also when the port is not set or the port is driven low, if the serial interface is disabled with bits SMD2 to SMD0 before setting the NODC bit to 0, this item may be the cause of the issue.
Also refer to the following section in 4. Analysis Methods:

- 4.1 Examining a Register Setting
- 4.5 Examining Signals when Receiving

2. Troubles when Using Clock Asynchronous Serial I/O (UART) Mode

Table 2.1 lists 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	Transmit Data is not Output from the TXDi Pin	Transmission is not Enabled	2.1.1
		CTS Function is Enabled and the CTSi Pin is Driven High	2.1.2
		Transmit Data is not Set	2.1.3
		TXD2 Pin is not Pulled Up when Using UART2	2.1.4
		TXDi Pin is not Pulled Up when Using It as N-Channel Open Drain Output	2.1.5
		Output Collides with the Other Peripheral Function	2.1.6
		UART6 or UART7 is Used in Memory Expansion Mode	2.1.7
		U4MR Register is not Set Properly when Using UART4	2.1.8
2.2	Unexpected Transmit Data is Output	Serial Interface is Disabled	2.2.1
		Bit Rate is Incorrectly Specified	2.2.2
		Operating Mode of the CPU Clock is Changed	2.2.3
		Division Ratio and Multiplying Factor of the PLL Clock is Incorrect	2.2.4
		TXDi Pin is Set to an Output Port	2.2.5
2.3	Missing Transmit Data	Empty Flag is Verified when Transmit Data is Set	2.3.1
		Transmit Buffer is Full when the Next Data is Set	2.3.2
2.4	Data cannot be Received Correctly in the Target Device	Target Device is not Properly Connected	2.4.1
		Communication Signal does not Satisfy the Specification of the Target Device	2.4.2
		Communication Format is not Correct	2.4.3
		Bit Rate is Incorrectly Specified	2.4.4
		Bit Rate is Outside the Allowable Margin of Error	2.4.5
		Transmissions are Performed Continuously with 1 Stop Bit	2.4.6
		UiBRG Register is Refreshed	2.4.7
2.5	Transmission cannot be Stopped Immediately	Transmission is Disabled during Transmit Operation	2.5.1
2.6	Data cannot be Received/Receive Interrupt does not Occur	Reception is not Enabled	2.6.1
		Target Device is not Properly Connected	2.6.2
		RXDi Pin is Set as an Output Port	2.6.3
		Receive Signal does not Satisfy the Electrical Characteristics	2.6.4
		Overrun Error Occurred	2.6.5
		Parity Error Occurred	2.6.6
		Framing Error Occurred	2.6.7
		Receive Data is Not Masked According to the Character Length	2.6.8
2.7	Overrun Error Occurred	Receive Data is not Read in Time	2.7.1
2.8	Parity Error Occurred	Parity Setting does not Match the Setting in the Target Device	2.8.1
2.9	Framing Error Occurred	Communication Formats do not Match	2.9.1
2.10	Communication is not Available After a Communication Error Occurred	Serial Interface is not Initialized After a Communication Error Occurred	2.10.1
—	When the Cause of the Issue Cannot be Found/Determined	—	5.

2.1 Transmit Data is not Output from the TXDi Pin

Example:
- The TXDi pin is not changed at all even though transmit data is set.

2.1.1 Transmission is not Enabled

Check whether transmission is performed while transmission is disabled (TE bit in the UiC1 register is 0).
When outputting transmit data, transmission must be enabled (TE bit is 1).

Solution:
When outputting transmit data, enable transmission (TE bit is 1).

Application Notes:
- Operation of Serial I/O (Transmission in UART Mode) (R01AN0542)

❖ When you are not sure if this is the cause of the issue:
Check the setting value of the TE bit in the UiC1 register by a program or debugger such as the E8a emulator immediately before setting transmit data (writing to the UiTB register). If the TE bit is 0 (transmission disabled), this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.4 Examining Signals when Transmitting

2.1.2 CTS Function is Enabled and the CTSi Pin is Driven High

Check whether the CTS function is enabled (CRD bit in the UiC0 register is 0) even though the function is not used. When using the CTS function, check whether the RTSi pin (1) in the target device, which is connected to the CTSi pin in the M16C MCU, is driven low.
When the CTS function is enabled, the CTSi pin must be driven low to output transmit data. The CTS function is enabled after a reset.

Solution:
When the CTS function is not used, disable the CTS function (CRD bit in the UiC0 register is 1).
When the CTS function is used, enable the CTS function (CRD bit in the UiC0 register is 0) and connect the CTSi pin to the RTSi pin (1) in the target device.

Application Notes:
- Operation of Serial I/O (Transmission in UART Mode) (R01AN0542)

❖ When you are not sure if this is the cause of the issue:
Check the setting value of the CRD bit in the UiC0 register by a program or debugger such as the E8a emulator immediately before setting transmit data (writing to the UiTB register). If the CRD bit is 0 (CTS/RTS function enabled), this item may be the cause of the issue. In that case examine the CTSi pin level using an oscilloscope or other tools. If the CTSi pin is driven high, this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.4 Examining Signals when Transmitting

Note:
1. The pin name differs depending on the target device.

2.1.3 Transmit Data is not Set

Check whether the transmit data is set to the UARTi transmit buffer register (UiTB).
When outputting transmit data, the TI bit in the UiC1 register must be set to 0 (data present in UiTB register). The TI bit automatically becomes 0 by setting transmit data to the UiTB register.

Solution:
When outputting transmit data, write transmit data to the UiTB register.

Application Notes:
- Operation of Serial I/O (Transmission in UART Mode) (R01AN0542)

❖ When you are not sure if this is the cause of the issue:
Check whether the program has the instruction to set transmit data to the UiTB register. If not, this item is the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.4 Examining Signals when Transmitting

2.1.4 TXD2 Pin is not Pulled Up when Using UART2

When the M16C/63, 64A, 64C, 65, 65C, or 6C Group product is used, check whether the TXD2 pin of UART2 is pulled up. For these products, pins TXD2 and RXD2 of UART2 are N-channel open drain output dedicated pins.
When transmission/reception is performed with pins TXD2 and RXD2, these pins must be pulled up with an external circuit.
With the M16C/5LD, 56D, 5L, 56, 5M, and 57 Group products, pins TXD2 and RXD2 of UART2 are CMOS output pins. Thus pull-up processing is not required.

Solution:
When transmitting or receiving data using pins TXD2 and RXD2 of UART2 which are N-channel open drain output pins, pull-up these pins with an external circuit.

Refer to “1.1.4 TXD2 Pin is not Pulled Up when Using UART2” when you are not sure if this is the cause of the issue or for reference materials.

Also refer to the following sections in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting
- 4.6 Checking Pull-Up for the N-Channel Open Drain Output Pin

2.1.5 TXDi Pin is not Pulled Up when Using It as N-Channel Open Drain Output

Check whether pins TXDi/SDAi and SCLi are pulled up when these pins are set to N-channel open drain output (NCH bit in the UiC0 register is 1).
N-channel open drain output does not output high. Therefore pins need to be pulled up with an external circuit.
For N-channel open drain output with pins TXDi/SDAi and SCLi by setting the NCH bit, the function only sets the P-channel transistor of the appropriate pin always off and it does not change pins TXDi/SDAi and SCLi to open drain output completely. For allowable input voltage range, refer to the Electrical Characteristics chapter in the User's Manual: Hardware for the product used.

Solution:
When using pins TXDi/SDAi and SCLi as N-channel open drain output, pull-up these pins with an external circuit.

Refer to "1.1.5 TXDi Pin Used for N-Channel Open Drain Output is not Pulled Up" when you are not sure if this is the cause of the issue or for reference materials.

Also refer to the following sections in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting
- 4.6 Checking Pull-Up for the N-Channel Open Drain Output Pin

2.1.6 Output Collides with the Other Peripheral Function

The output may collide with a pin output from the other peripheral function. Some pins can output signals for multiple functions (multi-function pin). For those pins, if multiple peripheral functions output a signal simultaneously, the signals collide. In that case, transmit data may not output correctly from the TXDi pin.

Solution:
When transmit data is output from the TXDi pin, do not output from the other peripheral function which shares the pin with.

Refer to "1.1.7 Output Collides with the Other Peripheral Function" when you are not sure if this is the cause of the issue or for reference materials.

2.1.7 UART6 or UART7 is Used in Memory Expansion Mode

Check whether UART6 or UART7 is used in memory expansion mode or microprocessor mode. With memory expansion mode or microprocessor mode, pins assigned to UART6 and UART7 function as bus control pins. Thus these pins cannot be used as serial interface. It is the same when 8-bit data bus is used (D8 to D15 not used) or when CS2/CS3 output is disabled.

Solution:
Do not use UART6 or UART7 in memory expansion mode or microprocessor mode.

Refer to "1.1.8 UART6 or UART7 is Used in Memory Expansion Mode" when you are not sure if this is the cause of the issue or for reference materials.

2.1.8 U4MR Register is not Set Properly when Using UART4

With the M16C/5LD, 56D, 5L, 56, 5M, or 57 product, check whether the UART4 transmit/receive mode register (U4MR) is set correctly when using UART4.
The U4MR register is protected by the protect register in the M16C/5LD, 56D, 5L, 56, 5M, and 57 products. For details, refer to the Protection chapter in the User's Manual: Hardware for the product used.

Solution:
When setting the U4MR register in the M16C/5LD, 56D, 5L, 56, 5M, or 57 Group product, enable write operation with the protect register first.

Refer to "1.1.9 U4MR Register is not Set Properly when Using UART4" when you are not sure if this is the cause of the issue or for reference materials.

2.2 Unexpected Transmit Data is Output

Example:
- Output width of 1-bit transmit data is not expected width.

2.2.1 Serial Interface is Disabled

Check whether the serial interface is disabled (bits SMD2 to SMD0 in the UiMR register are 000b) during transmission.
If the serial interface is disabled, the transmission is canceled at that point.

Solution:
Do not disable the serial interface (set bits SMD2 to SMD0 to 000b) during transmission.

Refer to “1.2.1 Serial Interface is Disabled” when you are not sure if this is the cause of the issue or for reference materials.

2.2.2 Bit Rate is Incorrectly Specified

Check whether the bit rate is the expected bit rate according to the count source frequency and setting in the UARTi bit rate register (UiBRG).
In UART mode, dividing the frequency, which is divided by the value set in the UiBRG register, by 16 becomes the bit rate.

Solution:
Specify the UiBRG register with the calculation referring the formula in the Bit Rate, Clock Asynchronous Serial I/O (UART) Mode of the Serial Interface UARTi chapter in the User’s Manual: Hardware for the product used.
If there is some margin of error between the M16C MCU and the target device, refer to the application note below to confirm the communication availability.

Application Note:
- Tolerance of Data Transfer Rate in Clock Asynchronous Serial I/O Mode (R01AN0746)

❖ When you are not sure if this is the cause of the issue:
Check transmit data output from the TXDi pin using an oscilloscope.
If the bit width of the transmit data is not output as the expected bit rate, 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 Frequency of Operating Peripheral Function Clock f1
- 4.4 Examining Signals when Transmitting

2.2.3 Operating Mode of the CPU Clock is Changed

Check whether the operating mode of the CPU clock is changed while transmit data is output. With the modes listed below, clock sources provided to the CPU clock and f1 are the same. Therefore if the clock source of the CPU clock is changed, the f1 frequency will be changed.

- High-speed mode
- Medium-speed mode
- PLL operating mode
- 40 MHz on-chip oscillator mode
- 125 kHz on-chip oscillator mode
- 125 kHz on-chip oscillator low power mode

The clock source of f1SIO, f2SIO, f8SIO, and f32SIO is f1. Therefore when the operating mode is changed, the transmit/receive clock rate is also changed.

Solution:

When f1SIO, f2SIO, f8SIO, or f32SIO is used, do not change the operating mode during transmission/reception.

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

Check whether the program has the point to change the operating mode of the CPU clock. If it does, check whether the timing is during transmit data being output. In that case this item may be the cause of the issue.

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

- 4.2 Checking Frequency of Operating Peripheral Function Clock f1
- 4.4 Examining Signals when Transmitting

2.2.4 Division Ratio and Multiplying Factor of the PLL Clock is Incorrect

Check whether the clock frequency divided by bits PLC05 and PLC04 (reference frequency counter set bit) in the PLC0 register is between 2 to 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 satisfies the recommended operating conditions in the Electrical Characteristics chapter in the User's Manual: Hardware. If the restrictions of the 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 and 5 MHz (6 MHz in the M16C/6C Group). Set the clock frequency multiplied by bits PLC02 to PLC00 to the value that satisfies the recommended operating conditions of the electrical characteristics.

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

Check the setting value in the PLC0 register by a program or debugger such as the E8a emulator. If the values do not satisfy 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 sections in 4. Analysis Methods:

- 4.2 Checking Frequency of Operating Peripheral Function Clock f1
- 4.4 Examining Signals when Transmitting

2.2.5 TXDi Pin is Set to an Output Port

Check whether the port corresponding to the TXDi pin is set to output mode. Then check if the port output level is set to low or not set.
The TXDi pin functions as a port until UART mode is selected by the UARTi transmit/receive mode register (UiMR). Therefore if the port corresponding to the TXDi pin is set to output port, it may output a signal to the target device unintentionally.

Solution:
It is recommended to set the port corresponding to the TXDi pin to input mode and pull it up.

❖ When you are not sure if this is the cause of the issue:
Check the setting value in the port direction register by a program or debugger such as the E8a emulator to see if the port corresponding to the TXDi pin is set to output mode. If so, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting

2.3 Missing Transmit Data

Example:

- When data is transmitted continuously, transmit data may have 1-byte data missing.

2.3.1 Empty Flag is Verified when Transmit Data is Set

Check whether the transmit register empty flag (TXEPT bit in UARTi transmit/receive control register 0 (UiC0)) is verified before the next transmit data is set.
If these empty flags are verified immediately after the transmit data is set, these flags may indicate no data in the transmit register (TXEPT bit is 1) even though the transmit data is set. Then the next data cannot be set and may become missing data since the previous data is still present.

Solution:

When verifying that the transmission is completed, and setting transmit data, verify the interrupt request bit of the transmit interrupt first, and set the next transmit data.
The interrupt source of the transmit interrupt can be selected with the UARTi transmit interrupt source select bit (UiIRS) in UARTi transmit/receive control register 1 (UiC1).

Refer to “1.3.1 Empty Flag is Verified when Transmit Data is Set” when you are not sure if this is the cause of the issue or for reference materials.

2.3.2 Transmit Buffer is Full when the Next Data is Set

Check whether the next data is set to the transmit buffer while the transmit buffer is full (TI bit in the UiC1 register is 0). If the value is set to the UARTi transmit buffer register (UiTB) while data is present in the UiTB register, the value in the register is overwritten and the overwritten data may become missing data.

Solution:

When verifying that the transmit buffer register is empty and setting transmit data, verify the interrupt request bit of the transmit interrupt first, and set the next transmit data.
The interrupt source of the transmit interrupt can be selected with the UiIRS bit in the UiC1 register.

Refer to “1.3.2 Transmit Buffer is Full when the Next Data is Set” when you are not sure if this is the cause of the issue or for reference materials.

2.4 Data cannot be Received Correctly in the Target Device

Example:

- The target device cannot receive the transmit data even though it is output correctly.

2.4.1 Target Device is not Properly Connected

Check whether pins TXDi and RXDi in the M16C MCU are connected to the reception pin and the transmission pin in the target device, respectively.
Also when using the CTS/RTS function, check whether the CTSi/RTSi pin is connected to the corresponding pin.
For proper communication, make sure that pins are correctly connected between the M16C MCU and the target device.

Solution:
Connect pins correctly.

Refer to “1.4.1 Target Device is not Properly Connected” when you are not sure if this is the cause of the issue or for reference materials.

2.4.2 Communication Signal does not Satisfy the Specification of the Target Device

Check whether the width or level of the output signal satisfies the rated values, operating conditions, or the other characteristics of the target device.
For proper communication, the communication standards of the target device must be satisfied.

Solution:
Satisfy the communication standards of the target device.

Refer to “1.4.2 Communication Signal does not Satisfy the Specification of the Target Device” when you are not sure if this is the cause of the issue or for reference materials.

2.4.3 Communication Format is not Correct

Check whether the I/O polarity, bit order, or bit rate of the transmit data being output is correct. If the data format or bit rate differs between the M16C MCU and the target device, data cannot be received correctly.

Solution:
Specify the I/O polarity, bit order, and bit rate according to the communication format of the target device.

❖ When you are not sure if this is the cause of the issue:
Check the transmit data being output using an oscilloscope to see if the communication format matches the format in the target device. If the formats do not match, this item may be the cause of the issue.

2.4.4 Bit Rate is Incorrectly Specified

Check whether the bit rate is the expected bit rate according to the count source frequency and setting in the UARTi bit rate register (UiBRG).
In UART mode, dividing the frequency, which is divided by the value set in the UiBRG register, by 16 becomes the bit rate.

Solution:
Specify the UiBRG register with the calculation referring the formula in the Bit Rate, Clock Asynchronous Serial I/O (UART) Mode of the Serial Interface UARTi chapter in the User's Manual: Hardware for the product used.
If there is some margin of error between the M16C MCU and the target device, refer to the application note below to confirm the communication availability.

Application Note:
- Tolerance of Data Transfer Rate in Clock Asynchronous Serial I/O Mode (R01AN0746)

Refer to "2.2.2 Bit Rate is Incorrectly Specified" when you are not sure if this is the cause of the issue or for reference materials.

2.4.5 Bit Rate is Outside the Allowable Margin of Error

Check whether the margin of error becomes 50% or more before all data are received.
In clock asynchronous communication, the clock does not synchronize with the target device. Therefore if the margin of error appears in the bit rate, data may not be received as expected.
The UART module performs data sampling at the rising edge of the transmit/receive clock (internal clock). Then if a margin of error becomes 50% or more before all data are received, a framing error may occur or expected data cannot be received between both communication devices.

Solution:
When one frame has 10 bits (start bit + 8 bits of character length + stop bit), adjust the bit rate so the margin of error does not become 5% or more for 1 bit and 50% or more for 10 bits.
Perform communication with enough margin of error.
For more information of the margin of error, refer to the application note below.

Application Note:
- Tolerance of Data Transfer Rate in Clock Asynchronous Serial I/O Mode (R01AN0746)

❖ When you are not sure if this is the cause of the issue:
Check the FER bit in the UiBR register. If the FER bit is 1 (framing error found), this item may be the cause of the issue.
Also check the transmit data output from the TXDi pin using an oscilloscope. If the margin of error for 1 bit is 5% or more, and the margin of error for 10 bits is 50% or more, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.2 Checking Frequency of Operating Peripheral Function Clock f1
- 4.4 Examining Signals when Transmitting

2.4.6 Transmissions are Performed Continuously with 1 Stop Bit

Check whether transmissions are performed continuously with 1 stop bit (STPS bit in the UiMR register is 0). When transmissions are performed continuously with 1 stop bit, the impact from a margin of error for the bit rate is accumulated in the next frame. If the accumulation becomes 50% of 1 bit or more, data cannot be received correctly by the target device. Then a framing error may occur or expected data may not be received. The impact can be avoided by using 2 stop bits (STPS bit in the UiMR register is 1).

Solution:
When performing transmission continuously, it is recommended to use 2 stop bits for communication. When using 1 stop bit, make sure to give enough of an interval between transmissions.

❖ When you are not sure if this is the cause of the issue:
Check the FER bit in the UiRB register. If the FER bit is 1 (framing error found), this item may be the cause of the issue.
Also, modify the program to give enough of an interval between transmissions or use 2 stop bits, and run it to see if the target device can receive data correctly. If it works, this item may be the cause of the issue.

2.4.7 UiBRG Register is Refreshed

Check whether the UiBRG register is refreshed or rewritten during transmission or reception. If so, the internal bit rate becomes unstable. Then data may not be transmitted or received correctly.

Solution:
Write to the UiBRG register while transmission or reception is stopped.

❖ When you are not sure if this is the cause of the issue:
Check whether the program has processing to refresh or rewrite the UiBRG register other than the serial interface initialization. If it does, check whether the processing is executed during transmission or reception. If so, this item may be the cause of the issue.
Also refer to the following sections in 4. Analysis Methods:
- 4.4 Examining Signals when Transmitting

2.5 Transmission cannot be Stopped Immediately

Example:

- Transmit data is still output even if the transmission is disabled.

2.5.1 Transmission is Disabled during Transmit Operation

Check whether transmission is disabled (TE bit in the UiC1 register is set to 0) while transmit data is output.
After a transmission starts, even if the transmission is disabled (TE bit in the UiC1 register is set to 0) during the transmit data being output, the transmission does not stop. When the transmission is completed and the transmit interrupt request is generated, transmission is disabled at the same time.

Solution:

When forcibly terminating the output of transmit data, disable the serial interface (bits SMD2 to SMD0 in the UiMR register are set to 000b).

Refer to “1.5.1 Transmission is Disabled with the Transmit Enable Bit” when you are not sure if this is the cause of the issue or for reference materials.

2.6 Data cannot be Received/Receive Interrupt does not Occur

Example:
- Receive data cannot be received even though it is input.
- Reception cannot be completed.

2.6.1 Reception is not Enabled

Check whether reception is performed with reception enabled (RE bit in the UiC1 register is 1).
Also check whether reception is enabled before receive data is transmitted.

Solution:
When receiving data, enable reception (RE bit in the UiC1 register is 1) of the UART module before the target device transmits data.

Application Note:
- Operation of Serial I/O (Reception in UART Mode) (R01AN0543)

❖ When you are not sure if this is the cause of the issue:
Check the RE bit immediately before the receive condition is met (receive data is input) by a program or debugger such as the E8a emulator. If the RE bit is 0 (reception disabled), this item is the cause of the issue.
Also refer to the following section in 4. Analysis Methods:
- 4.1 Examining a Register Setting
- 4.5 Examining Signals when Receiving

2.6.2 Target Device is not Properly Connected

Check whether pins TXDi and RXDi in the M16C MCU are connected to the reception pin and the transmission pin in the target device, respectively.
Also when using the CTS/RTS function, check whether the CTSi/RTSi pin is connected to the corresponding pin.
For proper communication, make sure that pins are correctly connected between the M16C MCU and the target device.

Solution:
Connect pins correctly.

Refer to “1.4.1 Target Device is not Properly Connected” when you are not sure if this is the cause of the issue or for reference materials.

2.6.3 RXDi Pin is Set as an Output Port

Check whether the I/O port corresponding to the RXDi pin is set as an output port.
The RXDi pin can be used as an I/O port. When the port is set to output mode, the RXDi pin functions as an output port. Then a signal collision occurs with input data and the RXDi pin cannot receive input data correctly.

Solution:
When inputting receive data, set the bit in the port Pi register corresponding to the RXDi pin as an input port.

Refer to “1.6.5 RXDi Pin is Set as an Output Port” when you are not sure if this is the cause of the issue or for reference materials.

2.6.4 Receive Signal does not Satisfy the Electrical Characteristics

Check whether the width or level of the receive signal input from the target device satisfies the standards of the electrical characteristics.
For proper communication, communication must be performed compliant with standards of the electrical characteristics.

Solution:
Make the receive signal input from the target device satisfy the standards of the electrical characteristics.

Refer to “1.6.6 Receive Signal does not Satisfy the Electrical Characteristics” when you are not sure if this is the cause of the issue or for reference materials.

2.6.5 Overrun Error Occurred

Check whether an overrun error occurred by receiving the next data before reading data from the receive buffer after data reception.
When an overrun error occurs, receive data becomes undefined. Also the receive complete interrupt does not occur (IR bit in the SiRIC register does not change).

Solution:
After data reception, read the UARTi receive buffer register (UiRB) before receiving the next data.
If an overrun error occurs (OER bit in the UiRB register is 1), disable the serial interface or disable the reception, and clear the overrun error. Before restarting the reception, read the UiRB register to clear the receive register.

Refer to “1.6.7 Overrun Error Occurred” when you are not sure if this is the cause of the issue or for reference materials.

2.6.6 Parity Error Occurred

Check whether the parity setting matches the setting in the target device.

Solution:
Make sure the parity setting is the same as the setting in the target device.

❖ When you are not sure if this is the cause of the issue:
Check whether a parity error occurred (PER bit in the UiRB register is 1) when reading the UiRB register. In that case this item is the cause of the issue.

2.6.7 Framing Error Occurred

Check whether the character length, parity enabled/disabled, I/O polarity, bit rate, and other settings are correctly specified in both communication devices. If settings in both devices do not match, a framing error may occur.

Solution:
Specify the communication settings according to the format in the target device.

❖ When you are not sure if this is the cause of the issue:
Check whether a framing error occurred (FER bit in the UiRB register is 1) when reading the UiRB register. In that case this item is the cause of the issue.

2.6.8 Receive Data is Not Masked According to the Character Length

Check whether only the required data are read following the selected character length. When the character length is 7 bits, b8 and b9 in the UiRB register are undefined. When the character length is 8 bits, b9 in the UiRB register is undefined.

Solution:
After reading the UiRB register, mask and read the required part of the data according to the selected character length.

❖ When you are not sure if this is the cause of the issue:
Check the read operation after data reception in the program to see if only required bits are used. If an undefined bit is used, this item may be the cause of the issue.

2.7 Overrun Error Occurred

Example:

- An overrun occurs when the data is received.

2.7.1 Receive Data is not Read in Time

Check whether the UARTi receive buffer register (UiRB) is read before the next data is input after the data reception.
An overrun error occurs if the last bit of the next data is received while data is present in the UiRB register (RI bit in the UiC1 register is 1).
When not using the receive interrupt, if the period from a reception to the next reception is longer than the period to input receive data, the receive register may not be read in time.
When the receive interrupt is used and multiple interrupts are enabled, if an interrupt which has a higher priority level than the receive interrupt occurs, the receive register may not be read in time.
When the receive interrupt is used and multiple interrupts are disabled, if receive data is input while another interrupt is processed, and the interrupt handler requires quite a long time for its processing, the receive register may not be read in time.

Solution:

After data is received, read the UiRB register before receiving the next data.
Consider the timing and the method to read the receive buffer according to the period to input receive data. Also consider using the DMAC for quick data access.

Refer to “1.7.1 Receive Data is not Read in Time” when you are not sure if this is the cause of the issue or for reference materials.

2.8 Parity Error Occurred

Example:

- The parity error flag becomes 1 when receiving data.

2.8.1 Parity Setting does not Match the Setting in the Target Device

Check whether the parity setting matches the setting in the target device.
With the UART module, the parity setting is selectable from parity disabled, even parity, and odd parity. The setting selected must match the setting in the target device.

Solution:

Make sure the parity setting is the same as the setting in the target device.

❖ When you are not sure if this is the cause of the issue:
Check whether the parity settings match in the program and the target device. If settings are not the same, this item is the cause of the issue.

2.9 Framing Error Occurred

Example:
- The framing error flag becomes 1 when receiving data.

2.9.1 Communication Formats do not Match

Check whether the character length, parity enabled/disabled, I/O polarity, bit rate, and other settings are correctly specified in both communication devices. If settings in both devices do not match, a framing error may occur.

Solution:
Specify the communication settings according to the format in the target device.

Application Note:
- Tolerance of Data Transfer Rate in Clock Asynchronous Serial I/O Mode (R01AN0746)

❖ When you are not sure if this is the cause of the issue:
Check the receive data being input to the RXDi pin using an oscilloscope. If the character length, parity enabled/disabled, I/O polarity, bit rate, and other settings do not match, this item may be the cause of the issue.

2.10 Communication is not Available After a Communication Error Occurred

Example:

- Communication cannot be performed correctly once an error occurred.

2.10.1 Serial Interface is not Initialized After a Communication Error Occurred

Check whether the serial interface is initialized after a communication error occurred. When communication is canceled or a communication error occurs, a problem such as bit slippage possibly occurs. Therefore the serial interface needs to be initialized.

Solution:

When canceling communication or when a communication error occurs, specify the settings as follows:

(1) Disable transmission/reception (set bits TE and RE in the UiC1 register to 0).
(2) Disable the serial interface (set bits SMD2 to SMD0 in the UiMR register to 000b).
(3) Reset bits SMD2 to SMD0.
(4) Enable transmission/reception (set bits TE and RE to 1).

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

Check the program whether processing on an communication error is correct.
If the serial interface is not initialized before restarting communication after the error occurred, this item may be the cause of the issue.

3. Troubles when Using the E8a Emulator Debugger

Table 3.1 lists 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	No Response can be Received from the MCU While Debugging	UART1 is Used	3.1.1
3.2	Communication is not Available when the Debugger is not Used	fOCO-F is Selected as the Count Source	3.2.1
3.3	Error does not Occur with Missing Receive Data/Clock is Output Incorrectly	Receive Buffer Register is Displayed in the Memory Window	3.3.1
—	When the Cause of the Issue Cannot be Found/Determined	—	5.

For information on the limitations of the E8a emulator, refer to the E8a emulator user's manual and additional document for user's manual from the following URL:
http://www.renesas.com/products/tools/emulation_debugging/onchip_debuggers/e8a/Documentation.jsp

3.1 No Response can be Received from the MCU While Debugging

Example:

- The system freezes after the serial interface initialization while debugging.

3.1.1 UART1 is Used

Check whether the program uses UART1 and it connects the E8a emulator in clock synchronous serial communication (communication performed with pins P6_4, P6_5, P6_6, and P6_7). In that case pins P6_4, P6_5, P6_6, and P6_7 in UART1 are occupied to control the MCU.
Also do not change bit 3 in the U1MR register since it is occupied by the E8a emulator debugger.

Solution:

When using the E8a emulator to debug the program with UART1, connect the E8a emulator in clock asynchronous serial communication with one wire.

❖ When you are not sure if this is the cause of the issue:
Check whether the program uses UART1. If it does, check whether it connects the E8a emulator in clock asynchronous serial communication with one wire. If not and if the program uses clock synchronous serial communication to connect the E8a emulator debugger, this item is the cause of the issue.

3.2 Communication is not Available when the Debugger is not Used

Example:

- Communication was available when using the debugger. However communication is not available when operating with the written program in the flash memory.

3.2.1 fOCO-F is Selected as the Count Source

Check whether fOCO-F is used as the count source. If so, check whether fOCO-F is oscillated.
When debugging with the E8a emulator, fOCO-F is used to control the MCU. Therefore the E8a emulator oscillates fOCO-F automatically.

Solution:
Oscillate fOCO-F when it is used as the count source.

❖ When you are not sure if this is the cause of the issue:
When using the fOCO-F as the count source, check whether the program oscillates fOCO-F. If not, this item is the cause of the issue.

3.3 Error does not Occur with Missing Receive Data/Clock is Output Incorrectly

Example:

- When using the debugger, an error does not occur even if data is missing in data reception.
- When using the debugger, the clock keeps being output from the CLKi pin.

3.3.1 Receive Buffer Register is Displayed in the Memory Window

Check whether the UARTi receive buffer register (UiRB) is displayed in the memory window.
The E8a emulator reads data regularly to display in the memory window. Then even if the user program cannot read data in time, an overrun error may not occur.
Also if clock synchronous serial I/O mode and continuous receive mode are used, the receive condition is met by reading data regularly, the transmit/receive clock may keep being output from the CLKi pin.

Solution:

When using the serial interface, do not display the UiRB register in the memory window.

❖ When you are not sure if this is the cause of the issue:
Close the memory window and check whether the same thing still happens. If not, this item is the cause of the issue.

4. Analysis Methods

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

- Integrated development environment: 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 value 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.

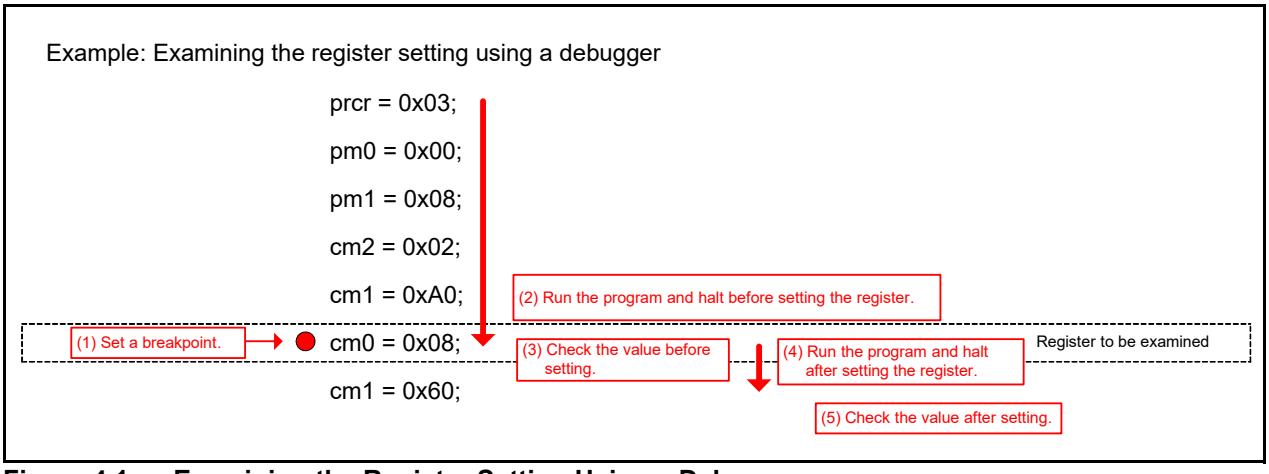

Figure 4.1 Examining the Register Setting Using a Debugger

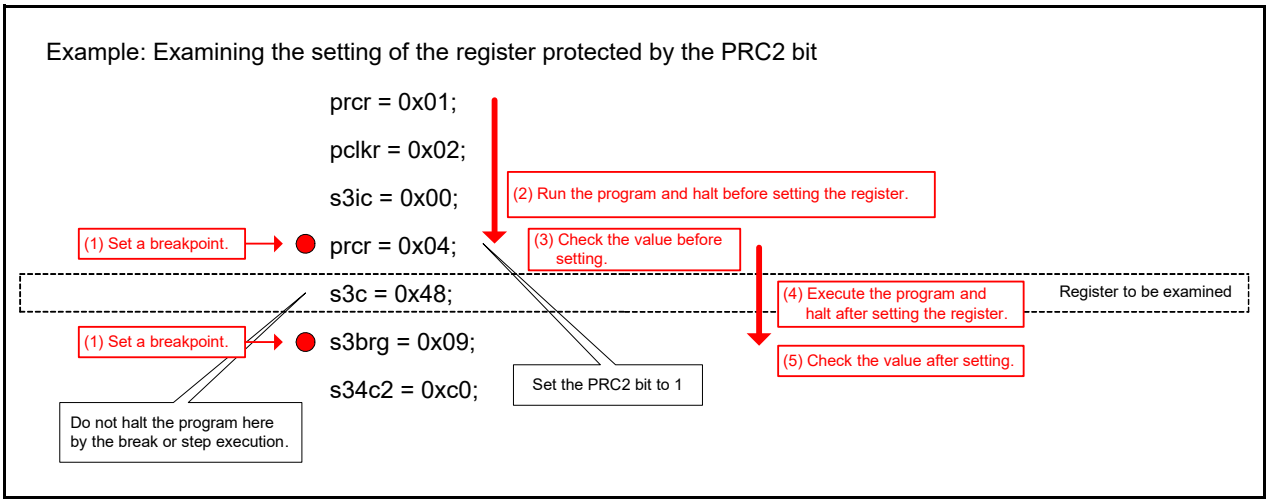

Figure 4.2 Examining the Setting of the Register Protected by the PRC2 Bit

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.

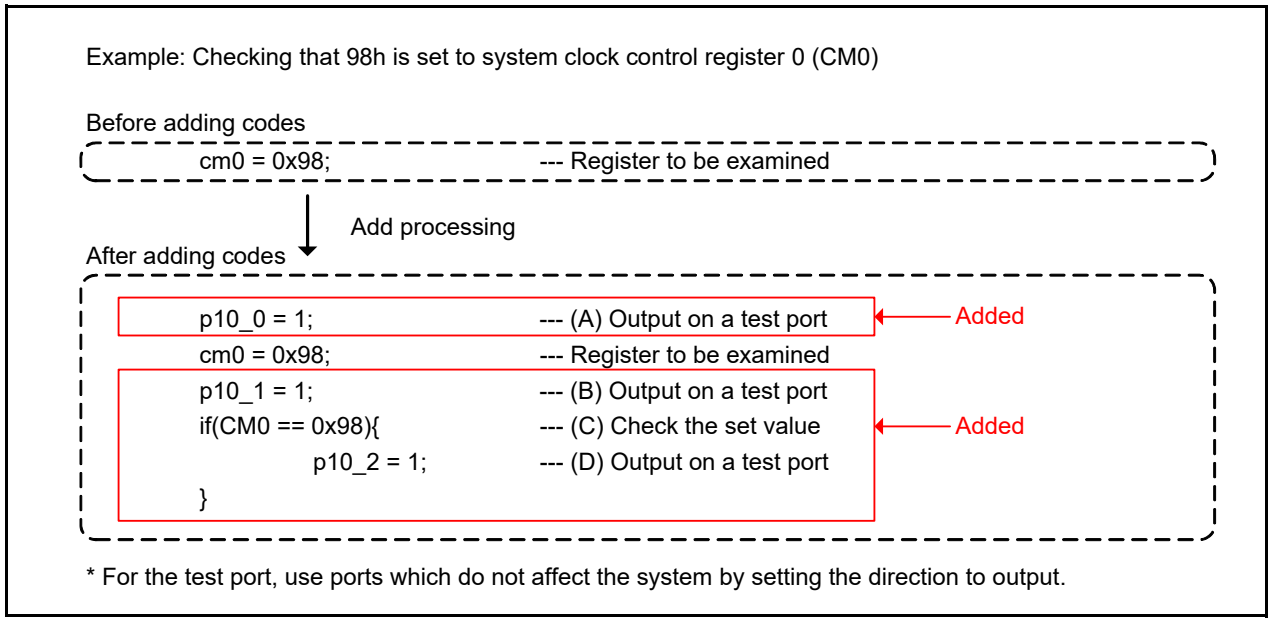

Figure 4.3 Example of Adding Test Codes

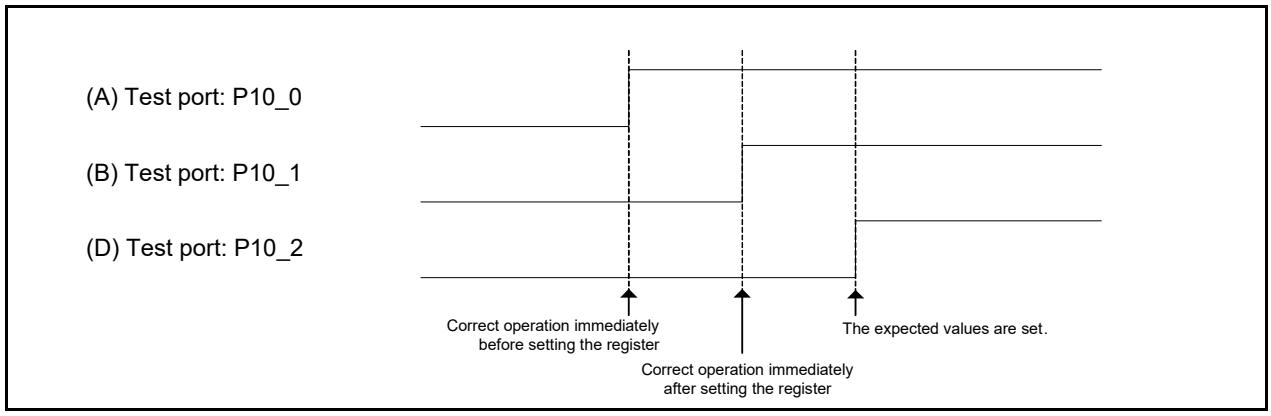

Figure 4.4 Waveform when the Value is Set as Expected

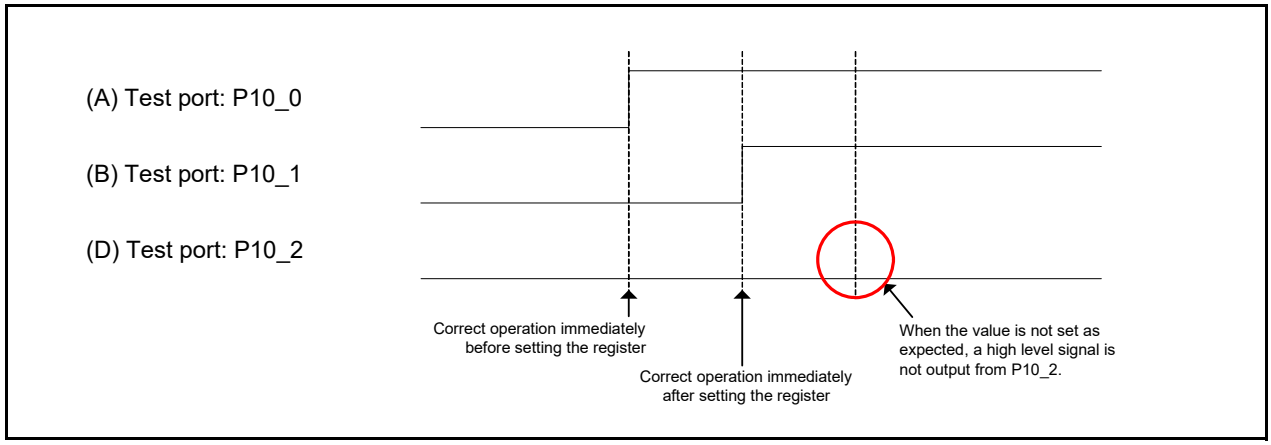

Figure 4.5 Waveform when the Value is not Set as Expected

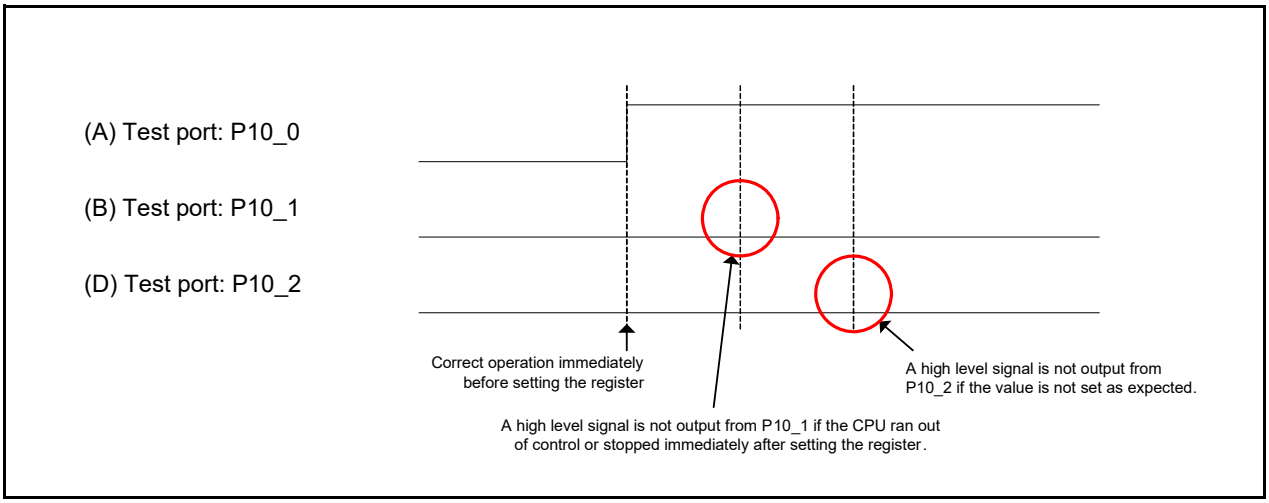

Figure 4.6 Waveform when the Program Stopped or Ran Out of Control after Setting a Register

4.2 Checking Frequency of Operating Peripheral Function Clock f1

This section describes methods to see whether peripheral function clock f1 is configured correctly.

4.2.1 Using the Clock Output Function and an Oscilloscope to Check f1 Output from CLKOUT

The clock output function is the function to output f1, f8, f32, or fC from the CLKOUT pin in single-chip mode. f1 is the same as the clock before the CPU clock passes the divider.
Specify the clock output function to output f1 from the CLKOUT pin and use an oscilloscope to check the frequency output from the CLKOUT pin.
The clock output from the CLKOUT pin should be 25 MHz or less. If it exceeds 25 MHz, use f8.
Make sure that the output from the CLKOUT pin does not affect the system.

Procedure

(1) Add codes to output f1 from the CLKOUT pin (set the PCLK5 bit in the PCLKR register to 1) to the program. When f1 exceeds 25 MHz, output f8 instead of f1 (set bits CM01 and CM00 in the CM0 register to 10b).
(2) Run the program to check the CLKOUT pin using an oscilloscope.

Explanation

With step (2) above, consider the checked frequency with the division selected. If the frequency is not as expected, check whether the registers associated with the system clock are set as expected. For details on the method, refer to “4.1 Examining a Register Setting”.

4.2.2 Checking Timer A Pulse Output Using an Oscilloscope

When f1 is used as the timer A count source and pulse is output with the counter value set to 0, half of the f1 frequency is output from the TAiOUT pin.
f1 is the same as the clock before the CPU clock passes the divider.
Use timer A to output half of the f1 frequency and check the frequency output from the TAiOUT pin with an oscilloscope.
Make sure that the output from the TAiOUT pin does not affect the system.

Procedure

(1) Add codes to output a half of f1 frequency using timer A.
Setting for timer A
- Operating mode: Timer mode
- Pulse output function: Pulse output
- Count source: f1
- Timer register: 0000h
- Count operation: Started

(2) Run the program and check the TAiOUT (i = channel to output the timer) pin using an oscilloscope.

Explanation

With step (2) above, double the checked frequency and consider it with the division selected. If the frequency is not as expected, check whether the registers associated with the system clock are set as expected. For the method, refer to “4.1 Examining a Register Setting”.

4.3 Examining Codes where an Interrupt Occurs

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

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

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

Procedure

(1) Set a breakpoint on the first code of the interrupt handler 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:

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

4.3.2 Examining Codes Using a Debugger

By stepping through return processing (REIT) from the interrupt handler, 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 handler.
(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

It is determined that 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:

- 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, a 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 Examining Signals when Transmitting

This section describes methods to examine output signals of transmit data or the clock.
The methods are described for each serial interface mode in the following order.
- Clock synchronous serial I/O mode with an internal clock
- Clock synchronous serial I/O mode with an external clock
- UART mode

4.4.1 Examining Signals in Clock Synchronous Serial I/O Mode with an Internal Clock Using an Oscilloscope

Add codes to invert a signal on the test port immediately before the code to set the transmit data. The inversion is used as a trigger for oscilloscope to capture the waveform when transmitting.
Examine the data output from the TXDi pin and the clock output from the CLKi pin at the timing to invert a signal on the test port using an oscilloscope. By doing this, problems you have encountered can be speculated.
For the test port, use a port which does not affect the system by setting the direction to output.

Procedure

(1) Add codes to invert a signal on the test port immediately before the code to set the transmit data to the UiTB register.
(2) Run the program to see the test port, TXDi pin, and CLKi pin states using an oscilloscope.
  • Set an inversion of a signal on the test port as the trigger to capture the waveform.

Explanation

Examine changes of the test port, TXDi pin, and CLKi pin in step (2) above with an oscilloscope.

This section introduces a normal pattern and error patterns of signals when operating in clock synchronous serial I/O mode with an internal clock. With error patterns, possible causes of issues are described.
Signal patterns introduced here are based on the following conditions:
- Internal clock (CKDIR bit in the UiMR register is 0)
- LSB first (UFORM bit in the UiC0 register is 0)
- Data logic not inverted (UiLCH bit in the UiC1 register is 0)
- Transmission at the falling edge of the clock and reception at the rising edge of the clock (CKPOL bit in the UiC0 register is 0)

- **Signal pattern in a normal operation [A-1]**

  Transmit data is output from the TXDi pin and the clock is output from the CLKi pin immediately after a signal on the test port is inverted.
  When the transmit data from the TXDi pin is as the data set in the UiTB register and the clock is output from the CLKi pin in an expected period, output signals are normal. Thus this is not the cause of the issue.
  Make sure signals satisfy the operating conditions (rated values) in both communication devices.

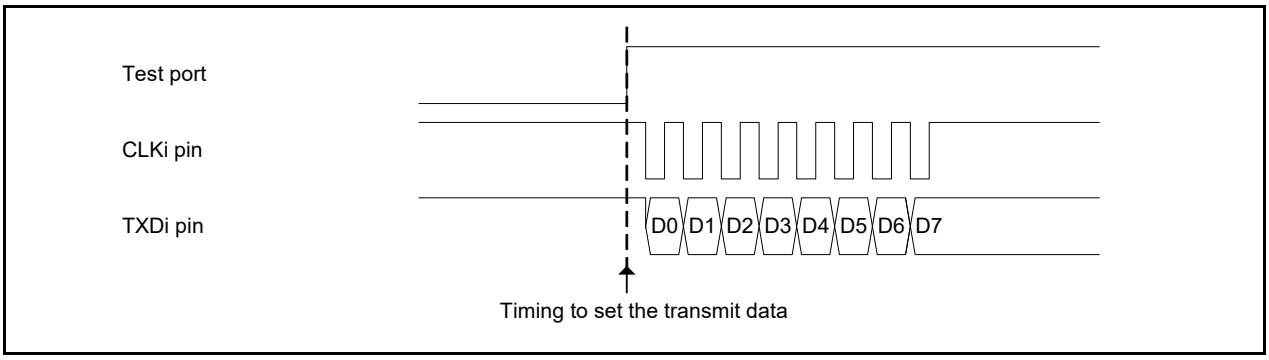

Figure 4.7 Signal pattern in a normal operation [A-1]

- **Signal pattern in an error operation [A-2]**

  Signals on the pins TXDi and CLKi remain high after a signal on the test port is inverted. In this case the transmit condition may not be met.
  Check whether transmission is enabled (TE bit in the UiC1 register is 1) and the CTS function is enabled (bits CRS and CRD in the UiC0 register are 0). When the CTS function is used, also check if the CTSi pin is driven low.

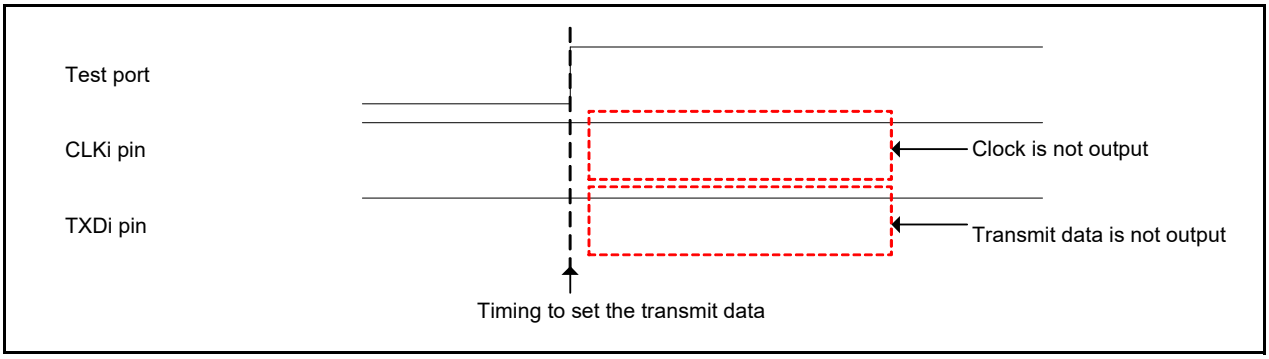

Figure 4.8 Signal pattern in an error operation [A-2]

- **Signal pattern in an error operation [A-3]**

  Signals on pins TXDi and CLKi remain low after a signal on the test port is inverted. In this case the UiMR register may not specified correctly (pins TXDi and CLKi are driven high when clock synchronous serial I/O mode is selected by the UiMR register), or possibly pins TXDi and CLKi are used as the N-channel open drain output pins and not pulled up.

  Specify the UiMR register. If pins TXDi and CLKi are used as the N-channel open drain output pins, pull them up with an external circuit.

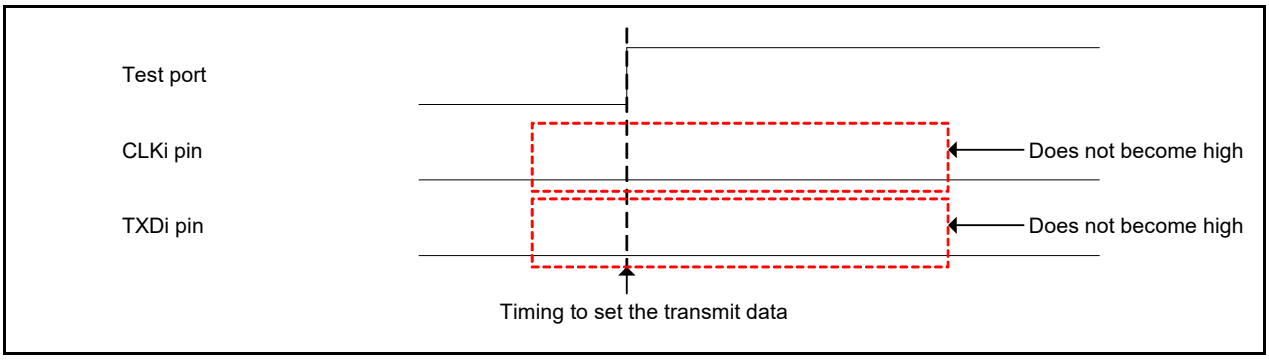

Figure 4.9 Signal pattern in an error operation [A-3]

- **Signal pattern in an error operation [A-4]**

  Signals on pins TXDi and CLKi are driven low until a certain point. In this case these pins are driven low probably while the serial interface is disabled (until clock synchronous serial I/O mode is selected by the UiMR register). If the CLKi pin changes from low to high, the target device may recognize the change as the transmit/receive clock and bit slippage may occur.

  Before the target device becomes ready for communication, set the UiMR register and drive the CLKi pin high, or pull up pins TXDi and CLKi with an external circuit, so that the pin levels do not change from low to high.

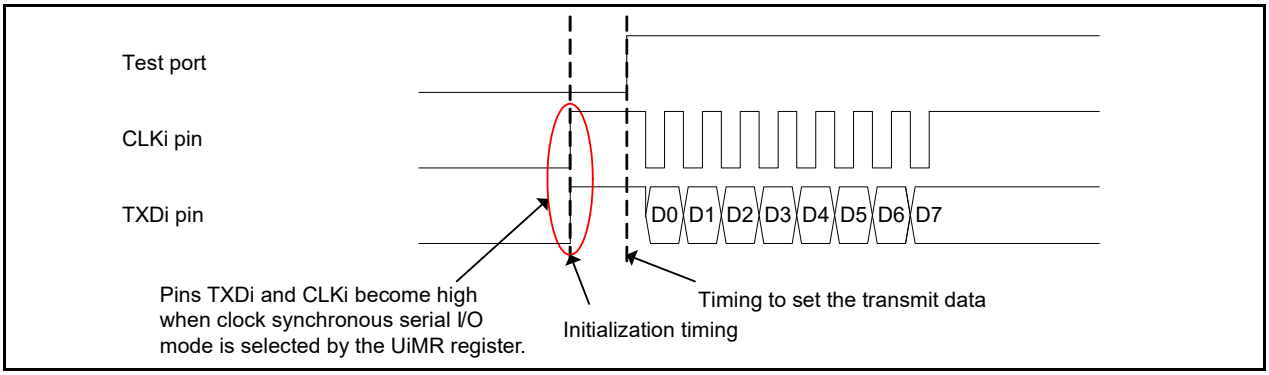

Figure 4.10 Signal pattern in an error operation [A-4]

- **Signal pattern in an error operation [A-5]**

  Short low pulses appear on pins TXDi and CLKi. In this case the serial interface may be disabled or noise may be introduced from peripheral circuits or external sources. If a pulse appears on the CLKi pin, bit slippage may occur in the target device.
  Do not disable the serial interface every transmission. To avoid bit slippage, disable the serial interface when transmission is disabled in the target device or while the CLKi pin is pulled up.
  When noise occurs, consider the design with less noise by adding a capacitor or moving the signal lines farther from the cause of the noise.

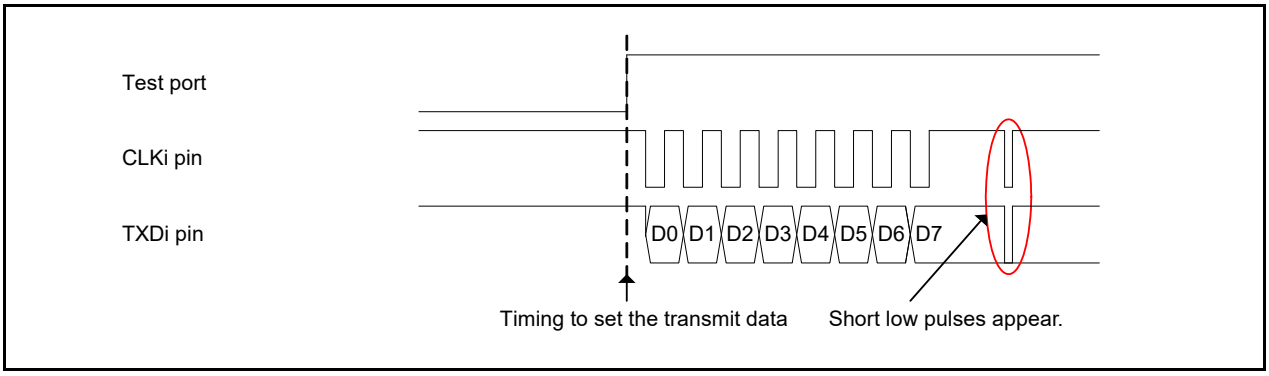

Figure 4.11 Signal pattern in an error operation [A-5]

- **Signal pattern in an error operation [A-6]**

  Low level output from pins TXDi and CLKi does not fall down to 0 V or high level does not rise up to VCC. In this case other pins connected to pins TXDi and CLKi or the target device output signals and signal collision may occur. Or the operation conditions VIH or VIL in the target device are not satisfied and low and high levels may not be recognized correctly.
  Examine the circuit and settings whether signals are output from pins connected to pins TXDi and CLKi or the target device.

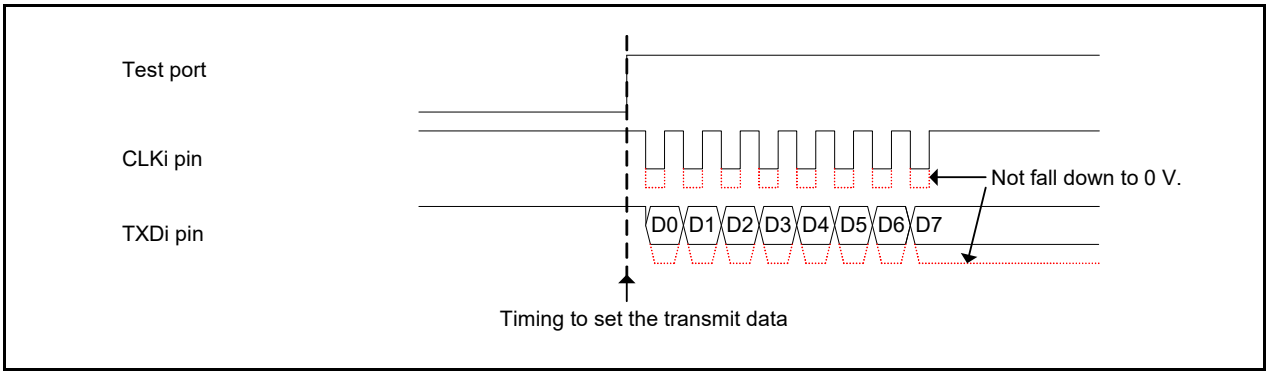

Figure 4.12 Signal pattern in an error operation [A-6]

- **Signal pattern in an error operation [A-7]**

  The test port is inverted multiple times during 1-byte transmit data is output. In this case the next data is set to the UiTB register while the transmit buffer is full (TI bit in the UiC1 register is 0) and a data may be dropped.
  First confirm the transmit buffer is empty or the transmission is completed, then set the next transmit data.

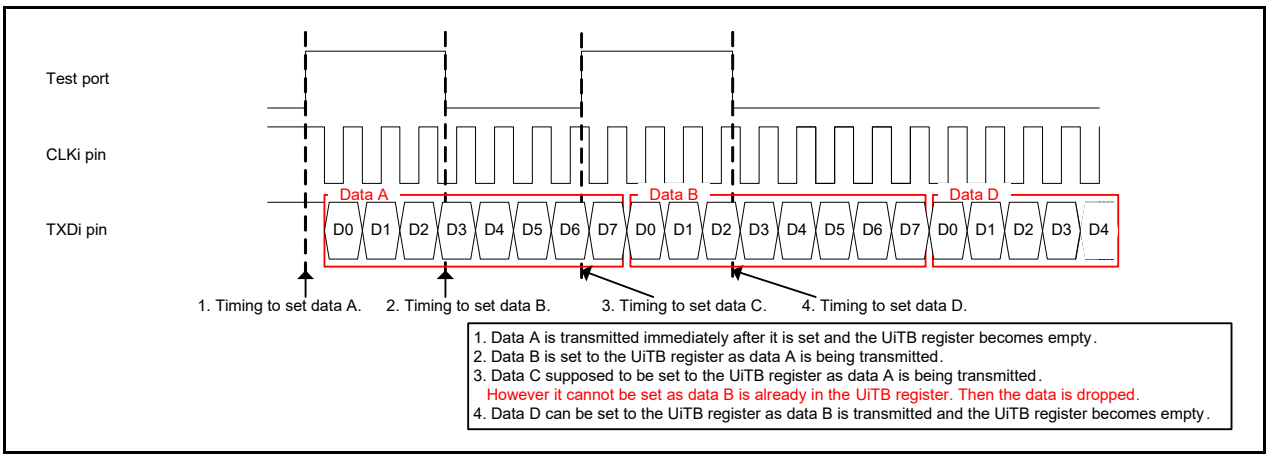

Figure 4.13 Signal pattern in an error operation [A-7]

4.4.2 Examining Signals in Clock Synchronous Serial I/O Mode with an External Clock Using an Oscilloscope

Add codes to invert a signal on the test port in the beginning of the interrupt handler on completion of the transmission. The inversion is used as a trigger for an oscilloscope to capture the waveform when transmitting.
Examine the data output from the TXDi pin and the clock output from the CLKi pin at the timing to invert a signal on the test port. By doing this, problems you have encountered can be speculated.
For the test port, use a port which does not affect the system by setting the direction to output.

Procedure

(1) Add codes to invert a signal on the test port in the beginning of the interrupt handler on completion of the transmission.
(2) Run the program to see the test port, TXDi pin, and CLKi pin states using an oscilloscope.
- Set an inversion of a signal on the test port as the trigger to capture the waveform.
- If an inversion cannot be seen on the test port, specify to invert a signal on the CLKi pin and use it as the trigger.

Explanation

Examine changes of the test port, TXDi pin, and CLKi pin in step (2) above with an oscilloscope.

This section introduces a normal pattern and error patterns of signals when operating in clock synchronous serial I/O mode with an external clock. With error patterns, possible causes of issues are described.
Signal patterns introduced here are based on the following conditions:

- External clock (CKDIR bit in the UiMR register is 1)
- LSB first (UFORM bit in the UiC0 register is 0)
- Data logic not inverted (UiLCH bit in the UiC1 register is 0)
- Transmission at the falling edge of the clock and reception at the rising edge of the clock (CKPOL bit in the UiC0 register is 0)
- Interrupt request generated when the UiTB register is empty (UiIRS bit in the UCON register or the UiC1 register is 0).

• **Signal pattern in a normal operation [B-1]**

The test port is inverted on the first falling edge of the clock input to the CLKi pin. Transmit data is output from the TXDi pin synchronizing with the clock input to the CLKi pin.
When the transmit data from the TXDi pin is as the data set in the UiTB register and the clock is input to the CLKi pin in an expected period, I/O signals are normal. Thus this is not the cause of the issue.
Make sure signals satisfy the operating conditions (rated values) in both communication devices.

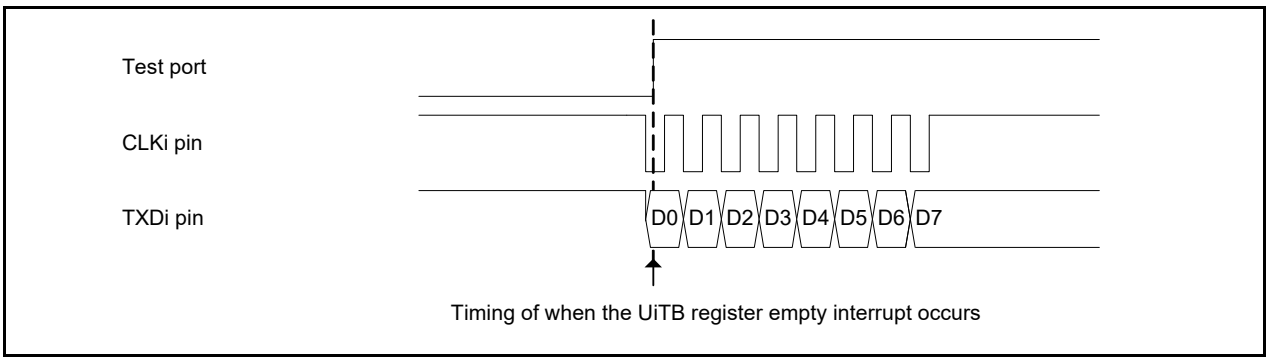

Figure 4.14 Signal pattern in a normal operation [B-1]

• **Signal pattern in an error operation [B-2]**

A signal on the test port is not inverted. The clock is not input to the CLKi pin. In this case the target device is possibly not configured correctly and it may not operate. Check the settings in the target device.

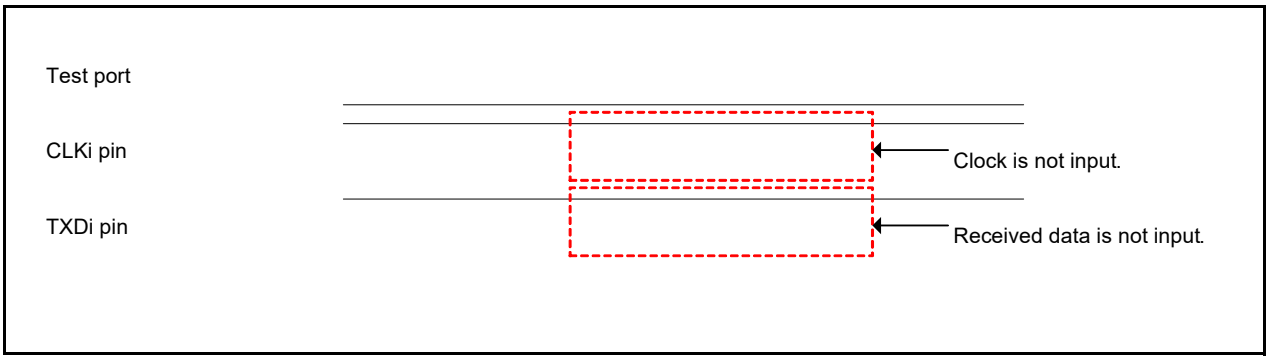

Figure 4.15 Signal pattern in an error operation [B-2]

- **Signal pattern in an error operation [B-3]**

  The test port is not inverted. Transmit data is not output from the TXDi pin even though the clock is input to the CLKi pin. In this case the transmit condition may not be satisfied.
  Check whether the transmission is enabled (TE bit in the UiC1 register is 1). Also check whether the CTS function is enabled (bits CRS and CRD in the UiC0 register are 0). When the CTS function is used, check whether the CTSi pin is driven low.

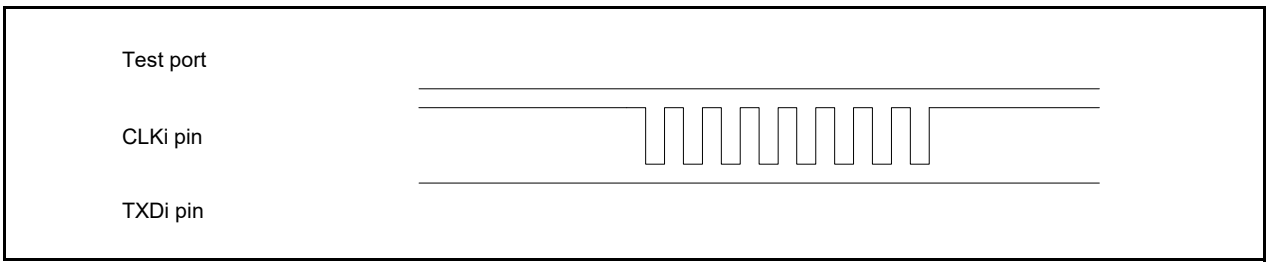

Figure 4.16 Signal pattern in an error operation [B-3]

- **Signal pattern in an error operation [B-4]**

  Signals on the test port, TXDi pin, and CLKi pin remain low. In this case the UiMR register may not be specified correctly (pins TXDi and CLKi are driven high when clock synchronous serial I/O mode is selected by the UiMR register), or possibly pins TXDi and CLKi are used as N-channel open drain output pins and not pulled up.
  Specify the UiMR register. If pins TXDi and CLKi are used as N-channel open drain output pins, pull them up with an external circuit.

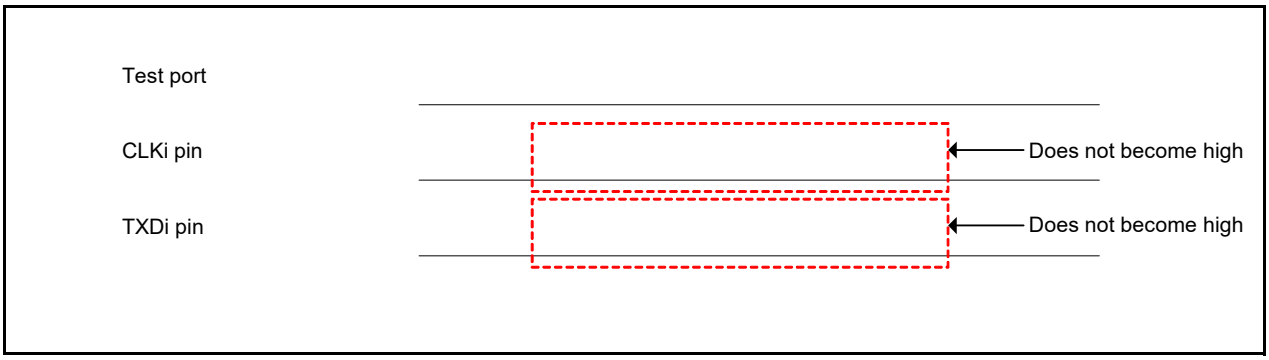

Figure 4.17 Signal pattern in an error operation [B-4]

- **Signal pattern in an error operation [B-5]**

  Low level output from the TXDi pin and input to the CLKi pin does not fall down to 0 V or high level does not rise up to VCC. In this case other pins connected to pins TXDi and CLKi or the target device may output signals and signal collision may occur. Or the operation conditions VIH or VIL in the target device are not satisfied and low and high levels may not be recognized correctly.
  Examine the circuit and settings whether signals are output from pins connected to pins TXDi and CLKi or the target device.

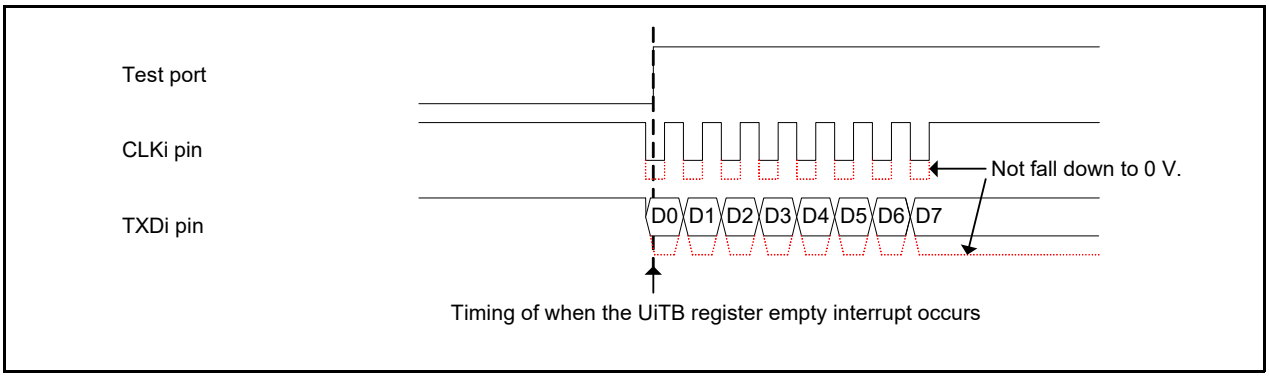

Figure 4.18 Signal pattern in an error operation [B-5]

- **Signal pattern in an error operation [B-6]**

  Short low pulses appear on pins TXDi and CLKi. In this case noise may be introduced from peripheral circuits or external sources. If a pulse appears on the CLKi pin, bit slippage may occur in the target device.
  Do not disable the serial interface every transmission. To avoid bit slippage, disable the serial interface when transmission is disabled in the target device or while the CLKi pin is pulled up.

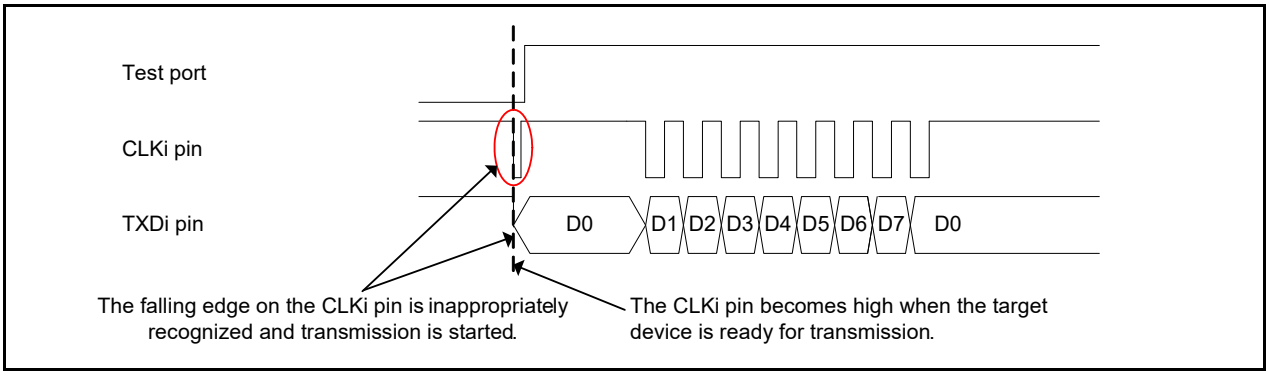

Figure 4.19 Signal pattern in an error operation [B-6]

- **Signal pattern in an error operation [B-7]**

  The test port is inverted multiple times during 1-byte transmit data is output. In this case the next data is set to the UiTB register while the transmit buffer is full (TI bit in the UiC1 register is 0) and a transmit data may be dropped.
  When setting the next transmit data, make sure the transmit buffer is empty or the transmission is completed.

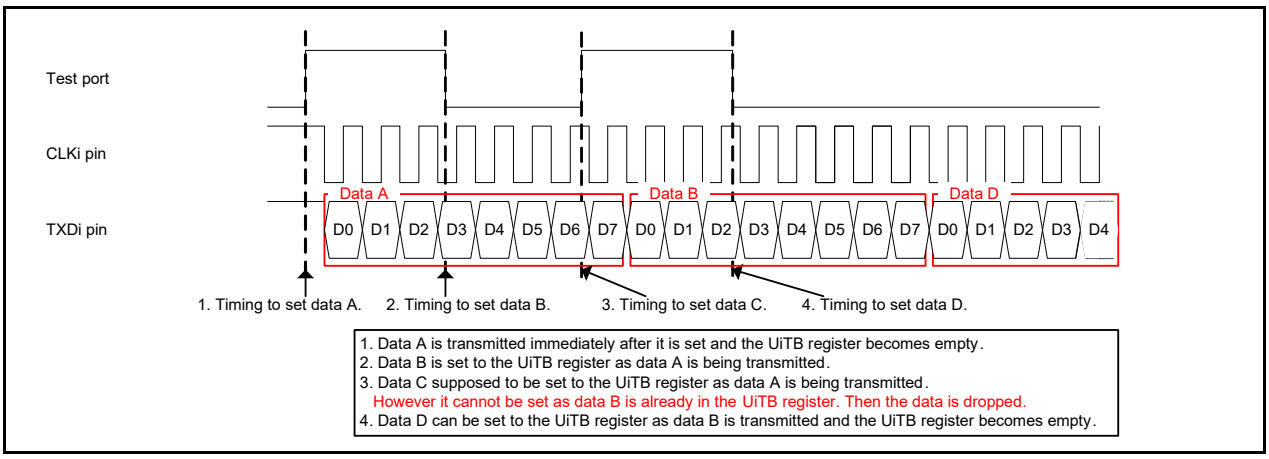

Figure 4.20 Signal pattern in an error operation [B-7]

4.4.3 Examining a Signal in UART Mode Using an Oscilloscope

Add codes to invert a signal on the test port immediately before the code to set the transmit data. The inversion is used as a trigger for oscilloscope to capture the waveform when transmitting.
Examine the data output from the TXDi pin at the timing to invert a signal on the test port. By doing this, problems you have encountered can be speculated.
For the test port, use a port which does not affect the system by setting the direction to output.

Procedure

(1) Add codes to invert a signal on the test port immediately before the code to set the transmit data to the UiTB register.
(2) Run the program to see the test port and the TXDi pin states using an oscilloscope.
  • Set an inversion of a signal on the test port as the trigger to capture the waveform.

Explanation

Examine changes of the test port and the TXDi pin in step (2) above with an oscilloscope.

This section introduces a normal pattern and error patterns of signals when operating in UART mode. With error patterns, possible causes of issues are described.
Signal patterns introduced here are based on the following conditions:

- Internal clock (CKDIR bit in the UiMR register is 0)
- LSB first (UFORM bit in the UiC0 register is 0)
- Data logic not inverted (UiLCH bit in the UiC1 register is 0)
- Transmission at the falling edge of the clock and reception at the rising edge of the clock (CKPOL bit in the UiC0 register is 0)

• Signal pattern in a normal operation [C-1]

Transmit data is output from the TXDi pin immediately after a signal on the test port is inverted.
When the transmit data from the TXDi pin is as the data set in the UiTB register, the output signal is normal. Thus this is not the cause of the issue.
Make sure signals satisfy the operating conditions (rated values) in both communication devices.

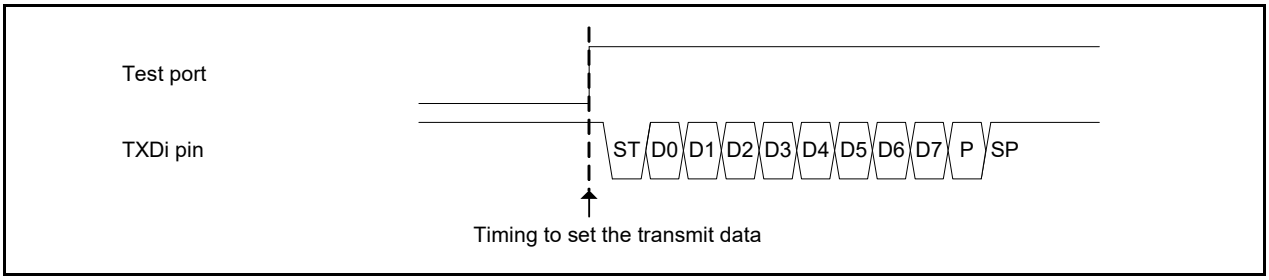

Figure 4.21 Signal pattern in a normal operation [C-1]

• **Signal pattern in an error operation [C-2]**

Signals on the TXDi pin remain high after a signal on the test port is inverted. In this case the transmit condition may not be satisfied.

Check whether transmission is enabled (TE bit in the UiC1 register is 1) and the CTS function is enabled (bits CRS and CRD in the UiC0 register are 0). When the CTS function is used, also check if the CTSi pin is driven low.

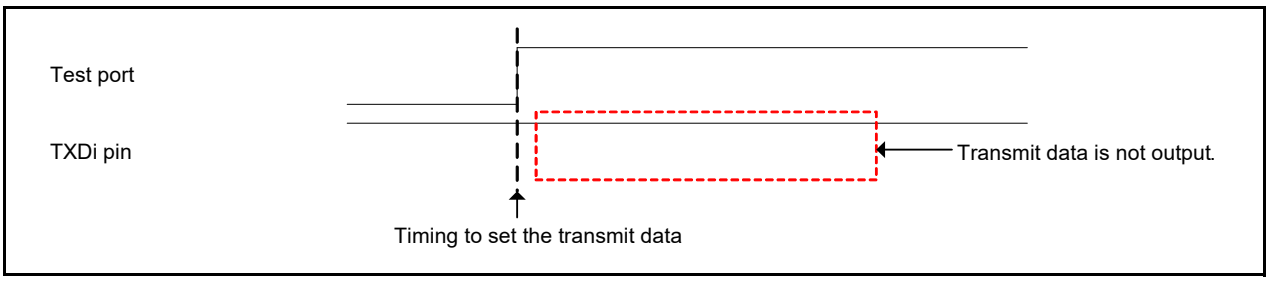

Figure 4.22 Signal pattern in an error operation [C-2]

• **Signal pattern in an error operation [C-3]**

Signals on the TXDi pin remain low after a signal on the test port is inverted. In this case the UiMR register may not be specified correctly (TXDi pin is driven high when UART mode is selected by the UiMR register), or possibly the pin is used as the N-channel open drain output pin and not pulled up.

Specify the UiMR register. If the TXDi pin is used as the N-channel open drain output pin, pull it up with an external circuit.

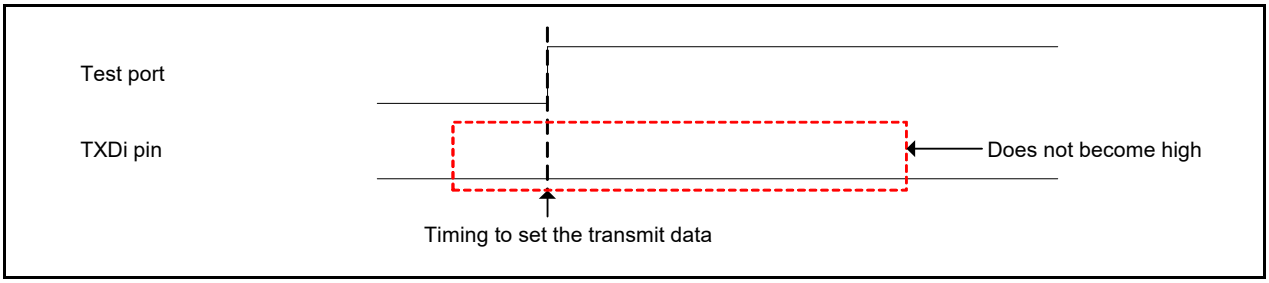

Figure 4.23 Signal pattern in an error operation [C-3]

• **Signal pattern in an error operation [C-4]**

The TXDi pin is driven low until a certain point. In this case the pin is driven low probably while the serial interface is disabled (until UART mode is selected by the UiMR register). The target device may inappropriately recognize the start bit, and incorrect data may be received.

Before the target device becomes ready for communication, set the UiMR register and drive the TXDi pin high, or pull up the TXDi pin with an external circuit, so that the pin is not driven low.

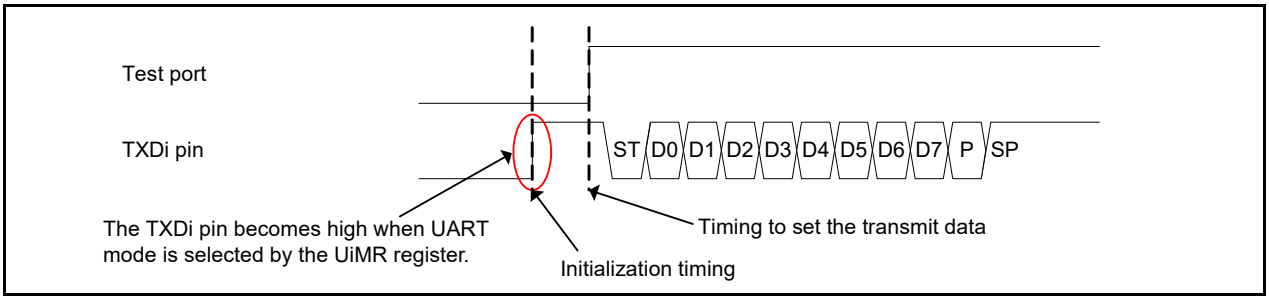

Figure 4.24 Signal pattern in an error operation [C-4]

- **Signal pattern in an error operation [C-5]**

  A short low pulse appears on the TXDi pin. In this case the serial interface may be disabled or noise may be introduced from peripheral circuits or external sources. If a pulse appears on the TXDi pin, incorrect data may be received in the target device.
  Do not disable the serial interface every transmission. Disable the serial interface when reception is disabled in the target device or while the TXDi pin is pulled up so that the target device can receive correct data.
  When noise occurs, consider the design with less noise by adding a capacitor or moving the signal lines farther from the cause of the noise.

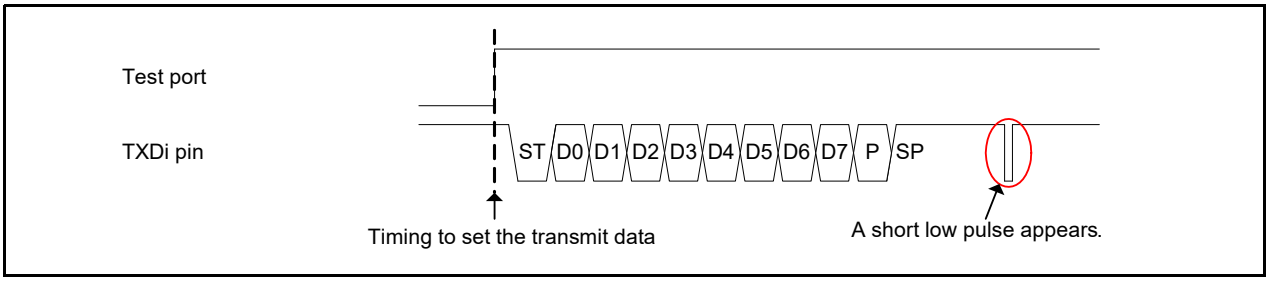

Figure 4.25 Signal pattern in an error operation [C-5]

- **Signal pattern in an error operation [C-6]**

  Low level output from the TXDi pin does not fall down to 0 V or high level does not rise up to VCC. In this case other pins connected to the TXDi pin or the target device may output signals and signal collision may occur. Or the operation conditions for VIH or VIL in the target device are not satisfied and low and high levels may not be recognized correctly.
  Examine the circuit and settings whether signals are output from pins connected to the TXDi pin or the target device.

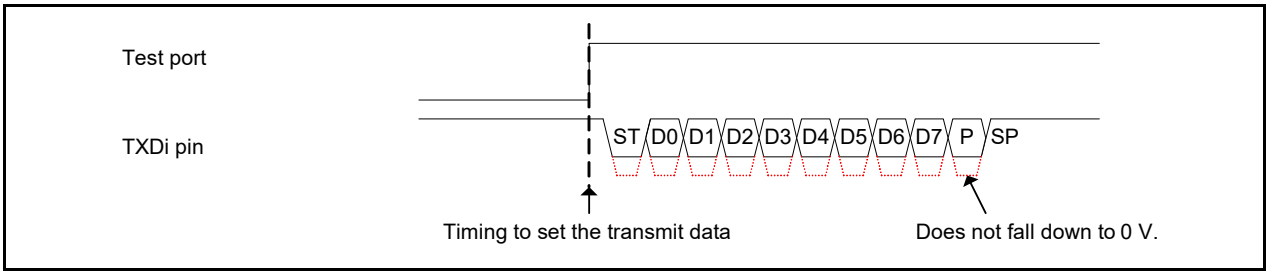

Figure 4.26 Signal pattern in an error operation [C-6]

- **Signal pattern in an error operation [C-7]**

  Width of a signal output from the TXDi pin for each bit is shorter or longer than the width expected. In this case the bit rate may not be specified correctly.
  Check whether the setting for the UiBRG register, the CPU clock, or the count source is correctly specified.

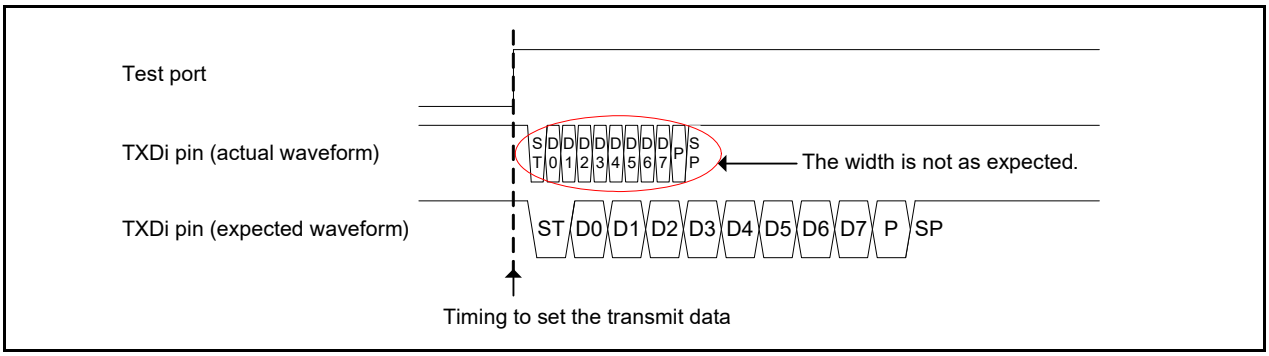

Figure 4.27 Signal pattern in an error operation [C-7]

- **Signal pattern in an error operation [C-8]**

  Width of a signal output from the TXDi pin for a bit is shorter or longer than the width expected. In this case the UiBRG register may be refreshed or rewritten. If the register is refreshed or rewritten during transmission, the bit rate may not be held internally and data cannot be transmitted or received correctly.
  Do not refresh or rewrite the UiBRG register during transmission.

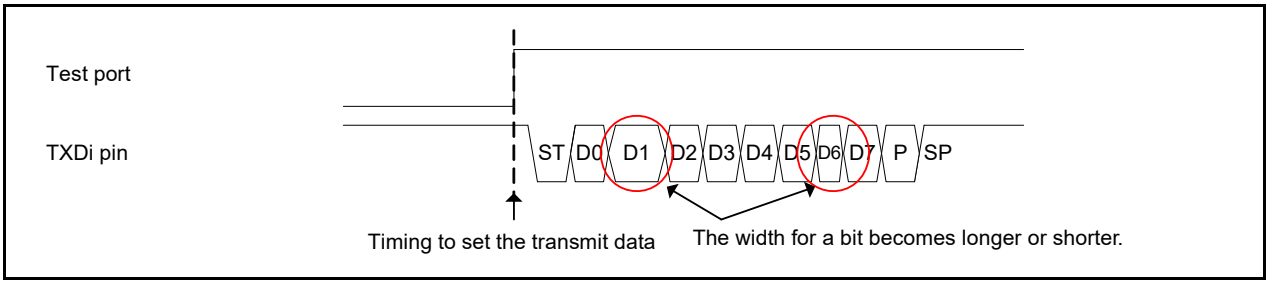

Figure 4.28 Signal pattern in an error operation [C-8]

- **Signal pattern in an error operation [C-9]**

  The test port is inverted multiple times while 1-byte transmit data is output. In this case the next data is set to the UiTB register while the transmit buffer is full (TI bit in the UiC1 register is 0) and a data may be dropped.

  Confirm the transmit buffer is empty or the transmission is completed before setting the next transmit data.

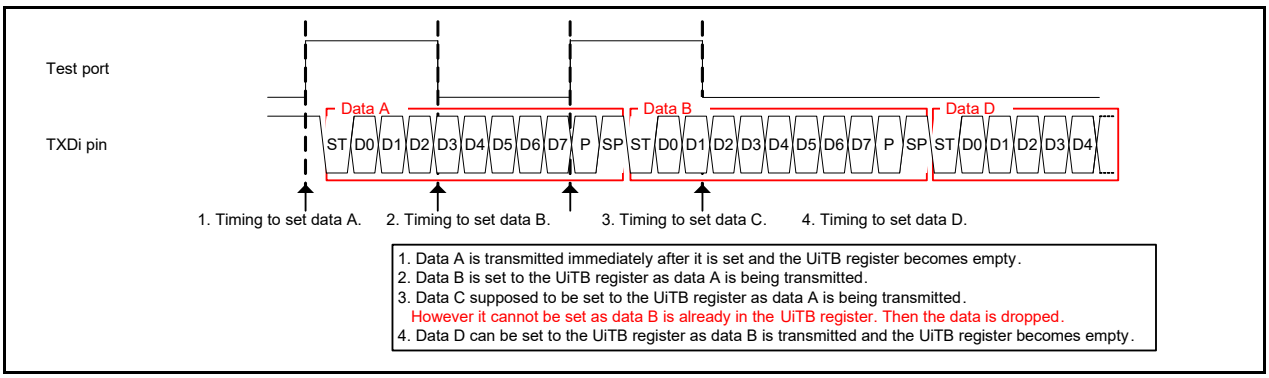

Figure 4.29 Signal pattern in an error operation [C-9]

4.5 Examining Signals when Receiving

This section introduces methods to examine whether signals of the receive data or the clock being input are correct. The methods are described for each serial interface mode in the following order.

- Clock synchronous serial I/O mode with an internal clock
- Clock synchronous serial I/O mode with an external clock
- UART mode

4.5.1 Examining Signals in Clock Synchronous Serial I/O Mode with an Internal Clock Using an Oscilloscope

Add codes to invert a signal on the test port immediately before setting dummy data. The inversion is used as a trigger for an oscilloscope to capture the waveform when receiving.
Examine the data input to the RXDi pin and the clock output from the CLKi pin at the timing to invert a signal on the test port with an oscilloscope. By doing this, problems you have encountered can be speculated.
For the test port, use a port which does not affect the system by setting the direction to output.

Procedure

(1) Add codes to invert a signal on the test port immediately before setting dummy data to the UiTB register.
(2) Run the program to see the test port, RXDi pin, and CLKi pin states using an oscilloscope.
  - Set an inversion of a signal on the test port as the trigger to capture the waveform.

Explanation

Examine changes of the test port, RXDi pin, and CLKi pin in step (2) above with an oscilloscope.

This section introduces a normal pattern and error patterns of signals when operating in clock synchronous serial I/O mode with an internal clock. With error patterns, possible causes of issues are described.
Signal patterns introduced here are based on the following conditions:

- Internal clock (CKDIR bit in the UiMR register is 0)
- LSB first (UFORM bit in the UiC0 register is 0)
- Data logic not inverted (UiLCH bit in the UiC1 register is 0)
- Transmission at the falling edge of the clock and reception at the rising edge of the clock (CKPOL bit in the UiC0 register is 0)

• **Signal pattern in a normal operation [D-1]**

The clock is output from the CLKi pin immediately after a signal on the test port is inverted.
The receive data is input to the RXDi pin synchronizing with the clock output from the CLKi pin.
When the receive data to the RXDi pin is the expected data and the clock is output from the CLKi pin in an expected period, signals on reception are normal. Thus this is not the cause of the issue.
Make sure signals satisfy the operating conditions (rated values) in both communication devices.

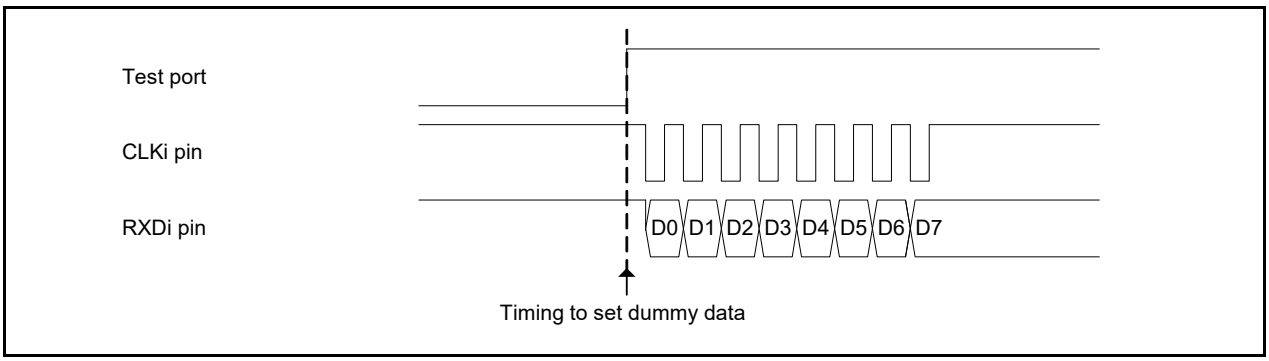

Figure 4.30 Signal pattern in a normal operation [D-1]

• **Signal pattern in an error operation [D-2]**

Signals on pins RXDi and CLKi remain high after a signal on the test port is inverted. In this case the receive condition may not be satisfied.
Check whether transmission is enabled (TE bit in the UiC1 register is 1) and the reception is enabled (RE bit in the UiC1 register is 1). Check whether dummy data is set in the UiTB register. Also check whether the CTS function is enabled (bits CRS and CRD in the UiC0 register are 0) though the function is not used. When the CTS function is used, check if the CTSi pin is driven low.

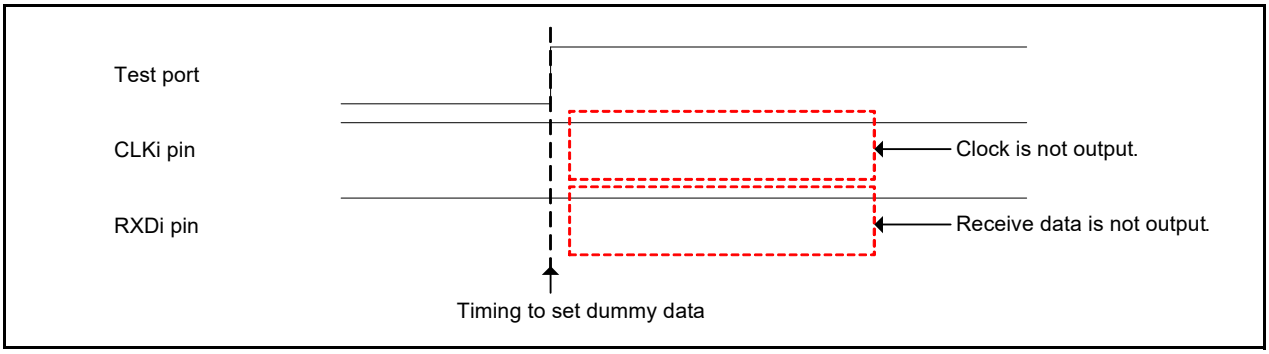

Figure 4.31 Signal pattern in an error operation [D-2]

- **Signal pattern in an error operation [D-3]**

  Signals on pins RXDi and CLKi remain low after a signal on the test port is inverted. In this case the UiMR register may not be specified correctly (the CLKi pin is driven high when clock synchronous serial I/O mode is selected by the UiMR register), the target device may not be ready for communication, or possibly pins RXDi and CLKi are used as N-channel open drain output pins and not pulled up.
  Specify the UiMR register. Confirm that the target device operates correctly. If pins RXDi and CLKi are used as N-channel open drain output pins, pull them up with an external circuit.

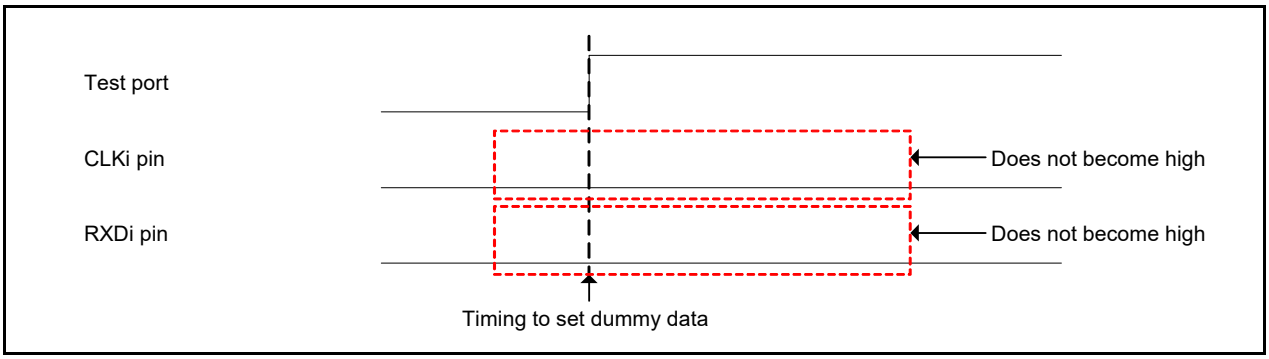

Figure 4.32 Signal pattern in an error operation [D-3]

- **Signal pattern in an error operation [D-4]**

  The RXDi pin does not change the level and remains high or low though the clock is output from the CLKi pin immediately after the test port is inverted. In this case the target device may not be ready for transmission or may not operate correctly.
  Check whether the start timing of communication or settings on the target device are correct.

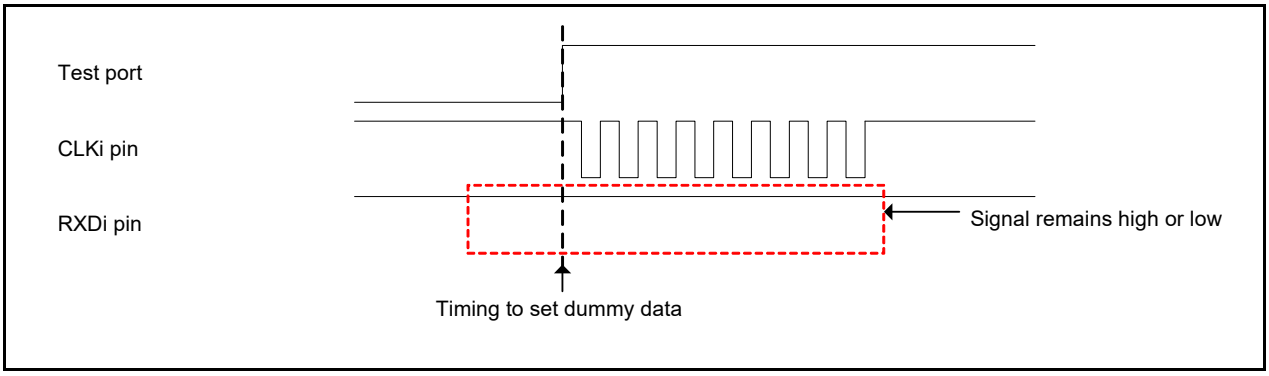

Figure 4.33 Signal pattern in an error operation [D-4]

• **Signal pattern in an error operation [D-5]**

The CLKi pin is driven low until a certain point. In this case the pin is driven low probably while the serial interface is disabled (until clock synchronous serial I/O mode is selected by the UiMR register). If the CLKi pin changes from low to high, the target device recognizes the change as the transmit/receive clock and bit slippage may occur.

Before the target device becomes ready for communication, set the UiMR register and drive the CLKi pin high, or pull up the CLKi pin with an external circuit, so that the pin level does not change from low to high.

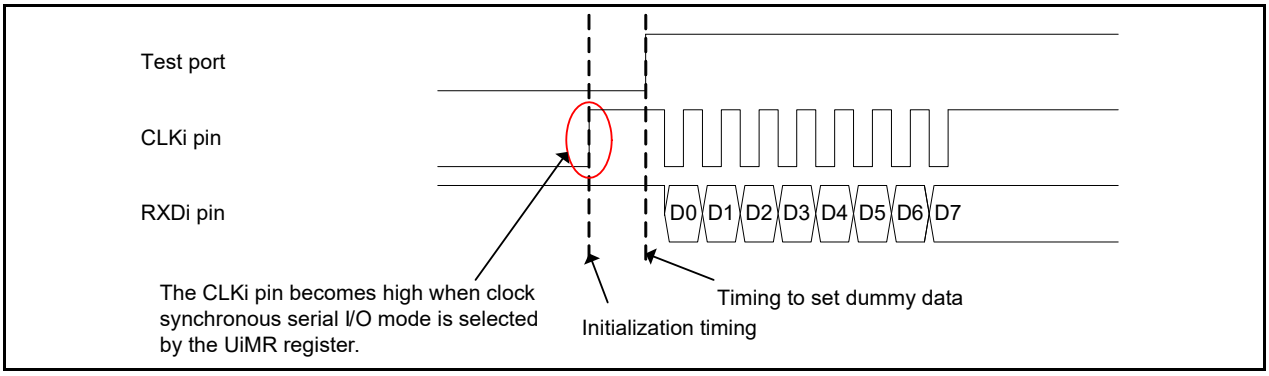

Figure 4.34 Signal pattern in an error operation [D-5]

• **Signal pattern in an error operation [D-6]**

Short low pulses appear on pins RXDi and CLKi. In this case the serial interface may be disabled or noise may be introduced from peripheral circuits or external sources. If a pulse appears on the CLKi pin, bit slippage may occur in the target device.

Do not disable the serial interface every reception. Disable the serial interface when transmission/reception is disabled in the target device or while the CLKi pin is pulled up to avoid bit slippage.

When noise occurs, consider the design with less noise by adding a capacitor or moving the signal lines farther from the cause of the noise.

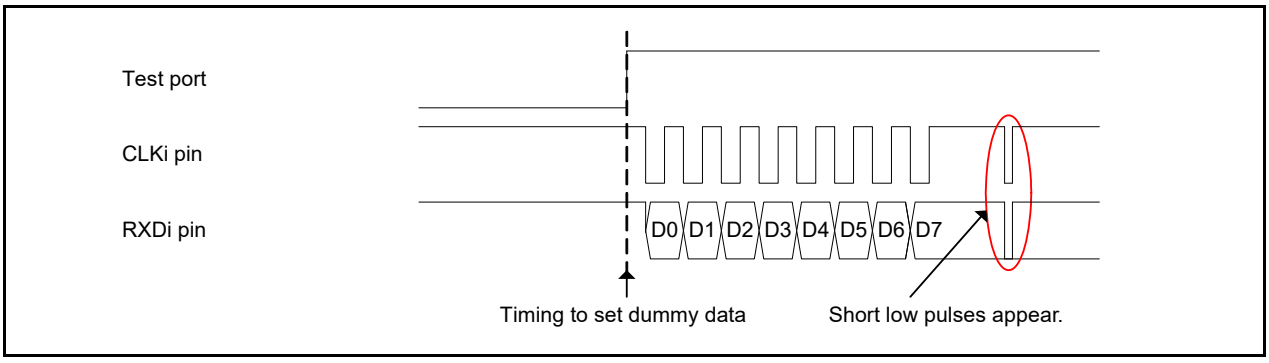

Figure 4.35 Signal pattern in an error operation [D-6]

- **Signal pattern in an error operation [D-7]**

  Low level output from the CLKi pin or input to the RXDi pin does not fall down to 0 V or high level does not rise up to VCC. In this case other pins connected to pins RXDi and CLKi or the target device may output signals and signal collision may occur. Or the operation conditions for VIH or VIL are not satisfied and low and high levels may not be recognized correctly.
  Examine the circuit and settings whether signals are output from pins connected to pins RXDi and CLKi or the target device.

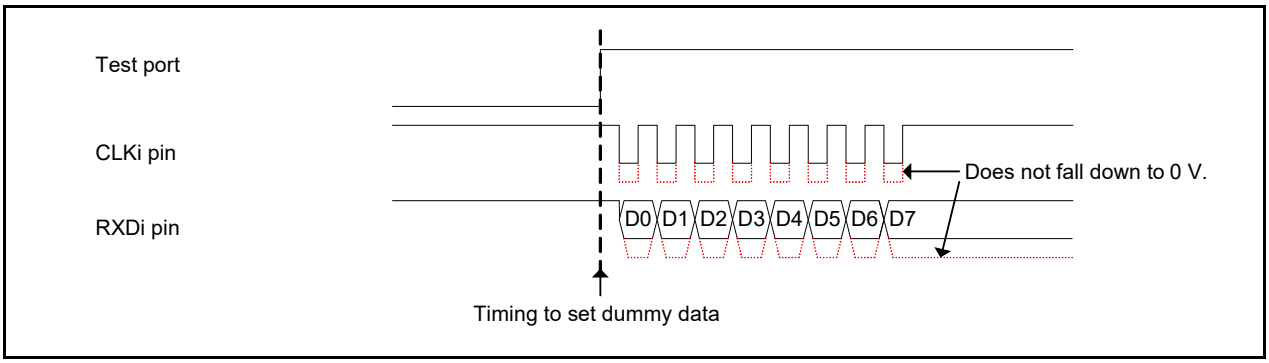

Figure 4.36 Signal pattern in an error operation [D-7]

4.5.2 Examining Signals in Clock Synchronous Serial I/O Mode with an External Clock Using an Oscilloscope

Add codes to invert a signal on the test port at the beginning of the receive interrupt handler. The inversion is used as a trigger for an oscilloscope to capture the waveform when receiving.
Examine the data input to the RXDi pin and the clock input to the CLKi pin at the timing to invert a signal on the test port with an oscilloscope. By doing this, problems you have encountered can be speculated.
For the test port, use a port which does not affect the system by setting the direction to output.

Procedure

(1) Add codes to invert a signal on the test port at the beginning of the receive interrupt handler.
(2) Run the program to see the test port, RXDi pin, and CLKi pin states using an oscilloscope.
- Set an inversion of a signal on the test port as the trigger to capture the waveform.
- When the inversion cannot be seen on the test port, set an inversion of a clock on the CLKi pin as the trigger.

Explanation

Examine changes of the test port, RXDi pin, and CLKi pin in step (2) above with an oscilloscope.

This section introduces a normal pattern and error patterns of signals when operating in clock synchronous serial I/O mode with an external clock. With error patterns, possible causes of issues are described.
Signal patterns introduced here are based on the following conditions:

- External clock (CKDIR bit in the UiMR register is 1)
- LSB first (UFORM bit in the UiC0 register is 0)
- Data logic not inverted (UiLCH bit in the UiC1 register is 0)
- Transmission at the falling edge of the clock and reception at the rising edge of the clock (CKPOL bit in the UiC0 register is 0)

- **Signal pattern in a normal operation [E-1]**

  A signal on the test port is inverted at the eighth rising timing of the clock input to the CLKi pin. The received data is input to the RXDi pin synchronizing with the clock input to the CLKi pin. In this case the receive data to the RXDi pin is the expected data and the clock is input to the CLKi pin in an expected period. Then input signals can be determined as normal. Thus this is not the cause of the issue.
  Make sure signals satisfy the operating conditions (rated values) in both communication devices.

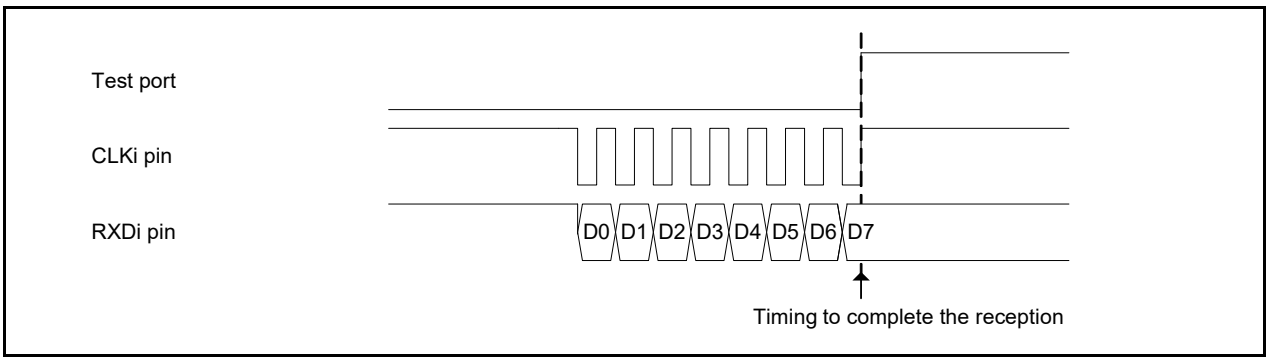

Figure 4.37 Signal pattern in a normal operation [E-1]

- **Signal pattern in an error operation [E-2]**

  A signal on the test port is not inverted. Also the receive data and clock are not input to pins RXDi and CLKi, respectively. In this case the target device may not be configured and it may not operate correctly.
  Check whether the target device is configured correctly.

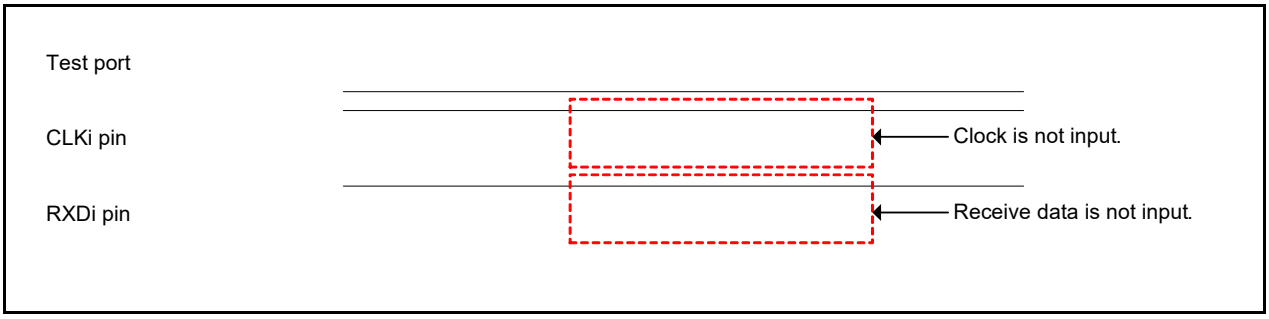

Figure 4.38 Signal pattern in an error operation [E-2]

- **Signal pattern in an error operation [E-3]**

  A signal on the test port is not inverted. The receive data and clock are input to pins RXDi and CLKi, respectively. In this case the condition for the reception may not be satisfied.
  Check whether the transmission is enabled (TE bit in the UiC1 register is 1) and the reception is enabled (RE bit in the UiC1 register is 1). Check whether dummy data is set to the UiTB register. Also check whether the CTS function is enabled (bits CRS and CRD in the UiC0 register are 0) though the function is not used. When the CTS function is used, check if the CTSi pin is driven low.

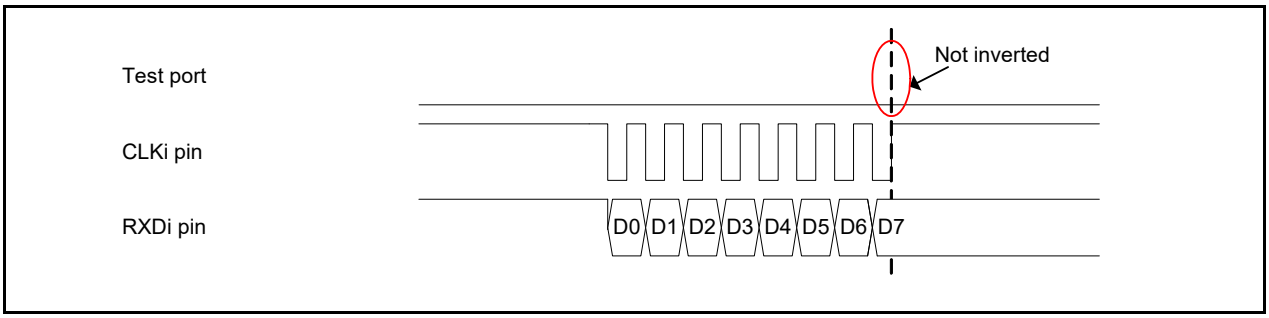

Figure 4.39 Signal pattern in an error operation [E-3]

- **Signal pattern in an error operation [E-4]**

  Signals on the test port, RXDi pin, and CLKi pin do not change and remain low. In this case the target device may not be ready for communication, or pins RXDi and CLKi may be used as the N-channel open drain output pins and not pulled up.
  Confirm that the target device operates correctly. If pins RXDi and CLKi are used as the N-channel open drain output pins, pull them up with an external circuit.

Figure 4.40 Signal pattern in an error operation [E-4]

- **Signal pattern in an error operation [E-5]**

  A signal on the test port is inverted other than at the last (eighth) rising timing of the clock input to the CLKi pin. In this case bit slippage may occur.
  Check whether noise occurs on the CLKi pin or the receive condition is met while the CLKi pin is driven low.

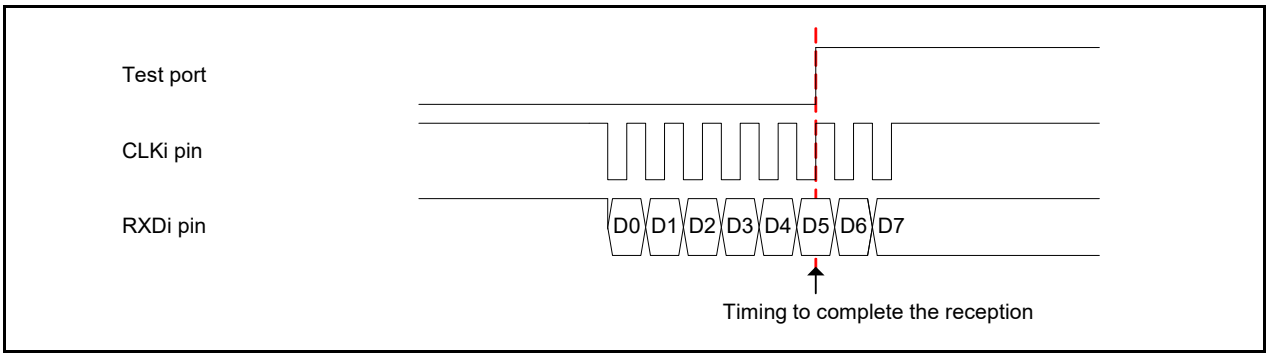

Figure 4.41 Signal pattern in an error operation [E-5]

- **Signal pattern in an error operation [E-6]**

  Low level input to pins CLKi and RXDi does not fall down to 0 V or high level does not rise up to VCC. In this case other pins connected to pins RXDi and CLKi or the target device may output signals and signal collision may occur. Or the operation conditions for VIH or VIL in the M16C MCU may not be satisfied and low and high levels may not be recognized correctly.
  Examine the circuit and settings whether signals are output from pins connected to pins RXDi and CLKi or the target device.

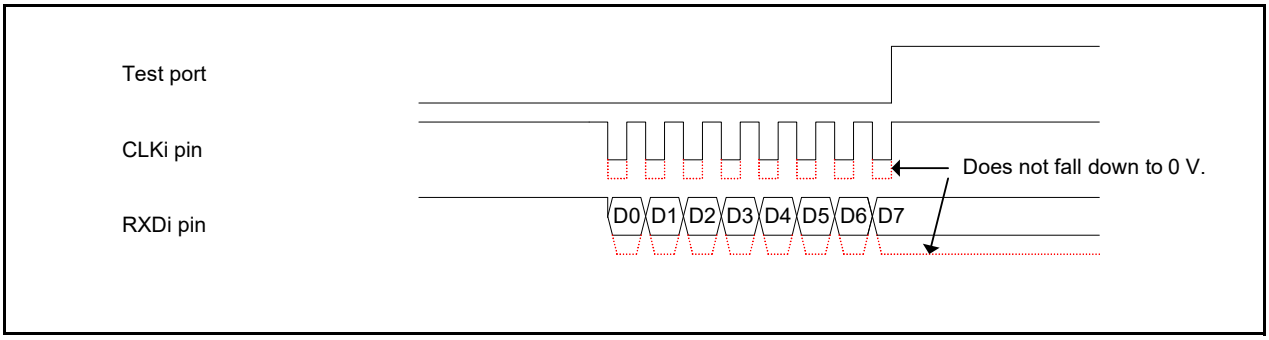

Figure 4.42 Signal pattern in an error operation [E-6]

- **Signal pattern in an error operation [E-7]**

  A signal on the test port is inverted when the first byte reception is completed but not inverted when the second byte reception is completed. In this case the receive buffer may not be read when a reception is completed. If the next data is received before reading the received data, an overrun error occurs. However the receive interrupt request bit does not change, thus an interrupt does not occur.
  After data reception, read the receive buffer before the next data is received.

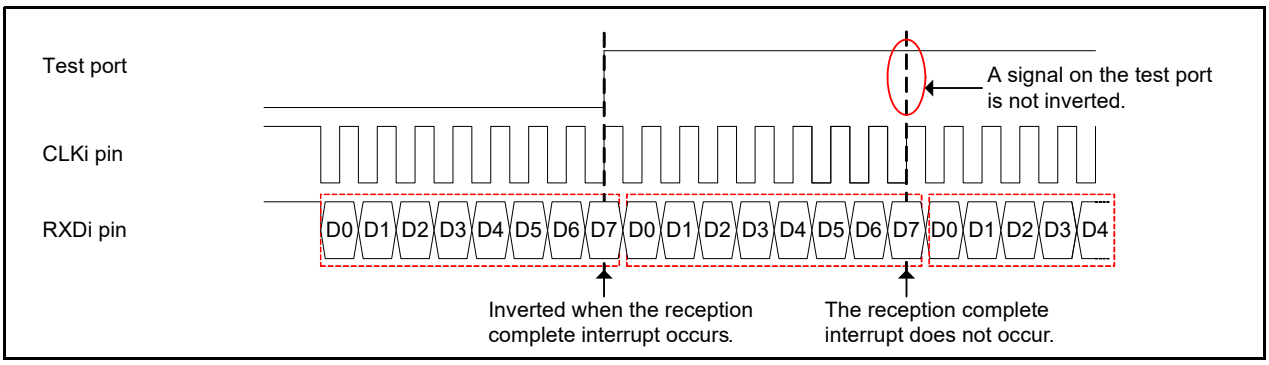

Figure 4.43 Signal pattern in an error operation [E-7]

4.5.3 Examining a Signal in UART Mode Using an Oscilloscope

Add codes to invert a signal on the test port at the beginning of the receive complete interrupt handler. The inversion is used as a trigger for oscilloscope to capture the waveform when receiving.
Examine the data input to the RXDi pin at the timing to invert a signal on the test port with an oscilloscope. By doing this, problems you have encountered can be speculated.
For the test port, use a port which does not affect the system by setting the direction to output.

Procedure

(1) Add codes to invert a signal on the test port at the beginning of the receive complete interrupt handler.
(2) Run the program to see the test port and RXDi pin states using an oscilloscope.
- Set an inversion of a signal on the test port as the trigger to capture the waveform.
- When the inversion cannot be seen on the test port, set an inversion of a clock on the RXDi pin as the trigger.

Explanation

Examine changes of the test port and RXDi pin in step (2) above with an oscilloscope.

This section introduces a normal pattern and error patterns of signals when operating in UART mode. With error patterns, possible causes of issues are described.
Signal patterns introduced here are based on the following conditions:
- Internal clock (CKDIR bit in the UiMR register is 0)
- LSB first (UFORM bit in the UiC0 register is 0)
- Data logic not inverted (UiLCH bit in the UiC0 register is 0)
- Transmission at the falling edge of the clock and reception at the rising edge of the clock (CKPOL bit in the UiC0 register is 0)

- **Signal pattern in a normal operation [F-1]**

Receive data is input to the RXDi pin and a signal on the test port is inverted at the timing where the stop bit for the received data is input.
When the receive data is input to the RXDi pin in the expected bit rate and the received data is as expected, the input signal is normal. Thus this is not the cause of the issue.
Make sure signals satisfy the operating conditions (rated values) in both communication devices.

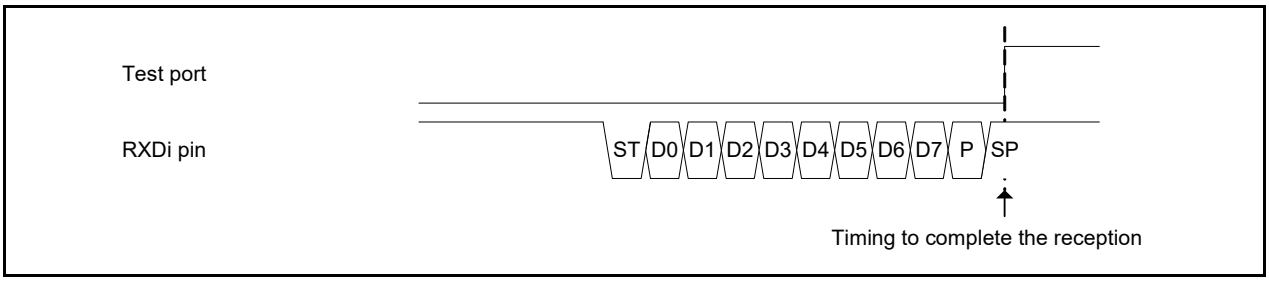

Figure 4.44 Signal pattern in a normal operation [F-1]

- **Signal pattern in an error operation [F-2]**

  A signal on the test port is not inverted. Also the receive data is not input to the RXDi pin. In this case the target device may not be configured and it may not operate correctly.
  Check whether the target device is configured correctly.

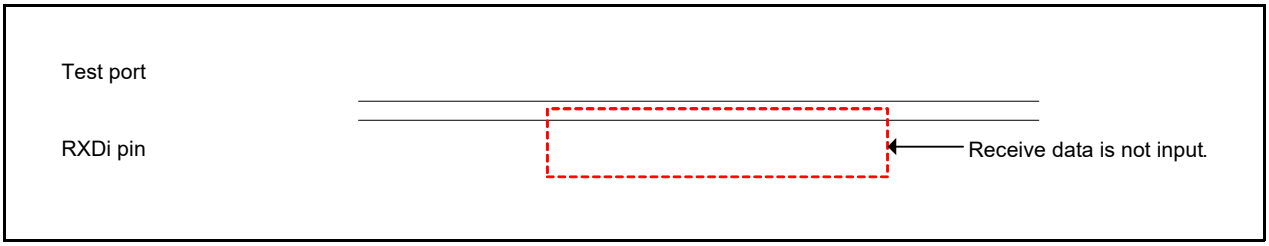

Figure 4.45 Signal pattern in an error operation [F-2]

- **Signal pattern in an error operation [F-3]**

  Signals on the test port and RXDi pin do not change and remain low. In this case the target device may not be ready for communication, or the RXDi pin may be used as the N-channel open drain output pin and not pulled up.
  Confirm that the target device operates correctly. If the RXDi pin is used as the N-channel open drain output pin, pull it up with an external circuit.

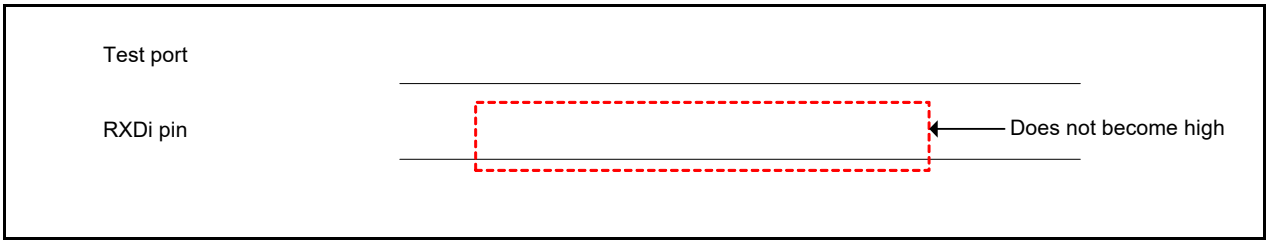

Figure 4.46 Signal pattern in an error operation [F-3]

- **Signal pattern in an error operation [F-4]**

  The test port is inverted during the receive data being input to the RXDi pin. In this case the bit rate in the M16C may differ from the one in the target device.
  Check whether settings for the UiBRG register, CPU clock, and count source are correctly specified. Also check whether the bit rate in the target device is correct.

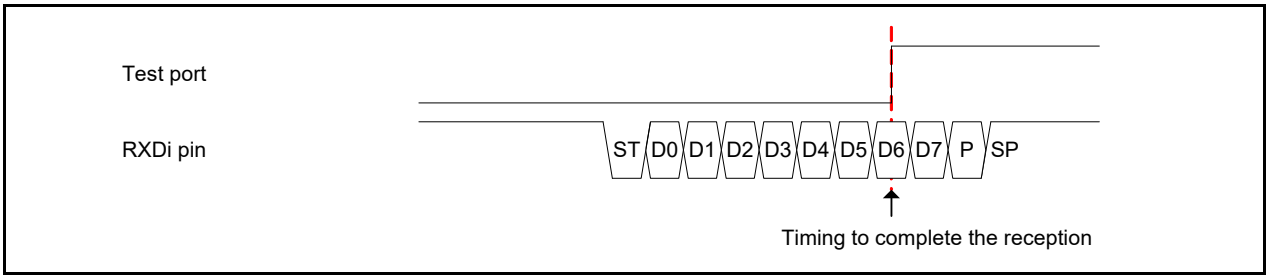

Figure 4.47 Signal pattern in an error operation [F-4]

- **Signal pattern in an error operation [F-5]**

  A short low pulse appears before the receive data is input to the RXDi pin. A signal on the test port is inverted during the receive data being input to the RXDi pin. In this case the short pulse may be recognized as the start bit. Also the low pulse may be noise introduced from peripheral circuits or external sources.

  When noise occurs, consider the design with less noise by adding a capacitor or moving the signal lines farther from the cause of the noise.

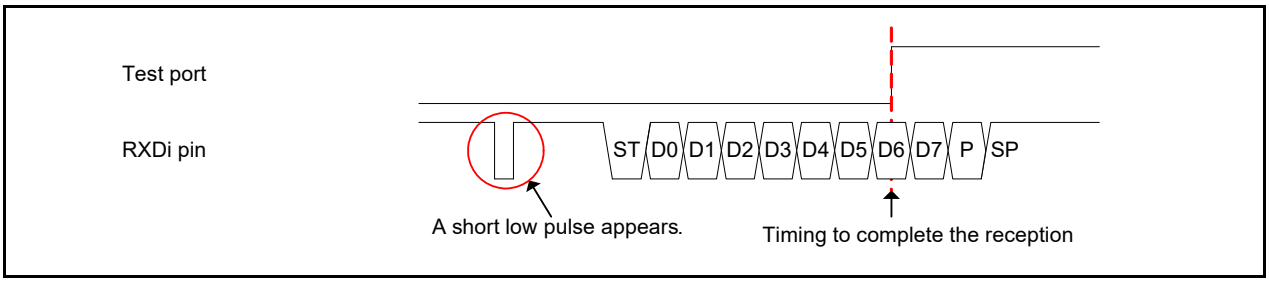

Figure 4.48 Signal pattern in an error operation [F-5]

- **Signal pattern in an error operation [F-6]**

  Low level input to the RXDi pin does not fall down to 0 V or high level does not rise up to VCC. In this case other pins connected to the RXDi pin or the target device may output signals and signal collision may occur. Or the operation conditions for VIH or VIL are not satisfied and low and high levels may not be recognized correctly.

  Examine the circuit and settings whether signals are output from pins connected to the RXDi pin or the target device.

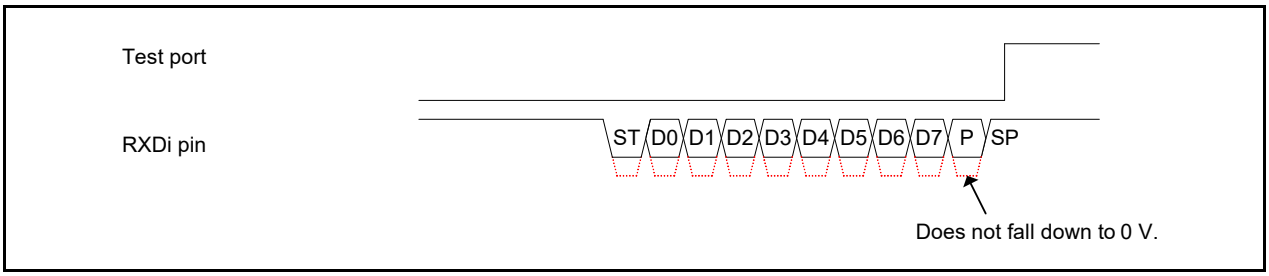

Figure 4.49 Signal pattern in an error operation [F-6]

- **Signal pattern in an error operation [F-7]**

  A signal on the test port is inverted when the first byte reception is completed but not inverted when the second byte reception is completed. In this case the receive buffer may not be read when a reception is completed. If the next data is received before reading the received data, an overrun error occurs. However the receive interrupt request bit does not change, thus an interrupt does not occur.
  After data reception, read the receive buffer before the next data is received.

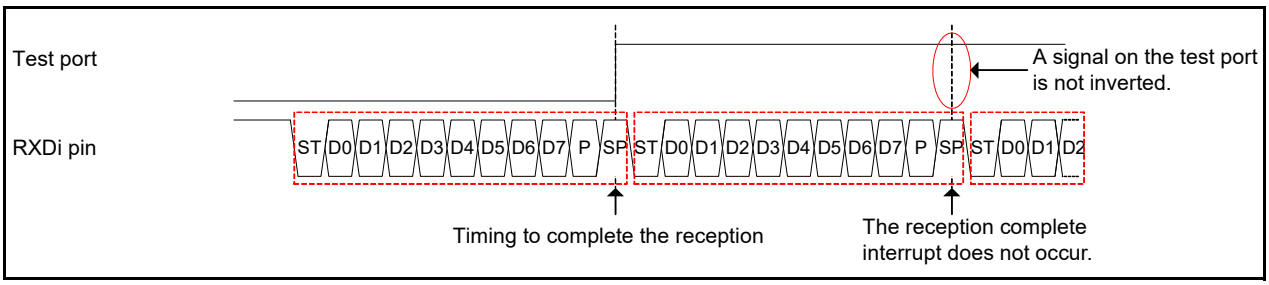

Figure 4.50 Signal pattern in an error operation [F-7]

4.6 Checking Pull-Up for the N-Channel Open Drain Output Pin

This section describes a method to check if the N-channel open drain output pin is correctly pulled up.

4.6.1 Checking with an Oscilloscope

Use an oscilloscope to check whether the N-channel open drain output pin is pulled up correctly.

Procedure

(1) Connect the pin to be checked to VSS (pull-down) through the same resistor as the one used for pull-up.
(2) Run the program to see the pin state with an oscilloscope.

Explanation

The possible cases are provided below to determine the result from performing step (2) above.

When the N-channel open drain output pin is pulled up correctly:
When the pin does not output (during reset), the pin level becomes half of the VCC. When the pin outputs high, the level becomes half of the VCC, and when the pin outputs low, the level becomes VSS. In this case the N-channel open drain output pin is pulled up correctly.

When communicating with an external device, the communication may not be performed correctly since the high level of the waveform output from the N-channel open drain output pin is half of the VCC. When checking communication with an external device, remove the pull-down resistor connected to the N-channel open drain output pin.

When the N-channel open drain output pin is not pulled up correctly:
When the pin does not output (during reset), the pin level becomes VSS.
When the pin outputs high, the level becomes VSS, and when the pin outputs low, the level becomes VSS.

Check the circuit if the pin pulled up is correct.
A low signal may be input from the external device. If so, avoid the low output by resetting the external device, for instance, then check the pin level again.

When the pin is not the N-channel open drain output pin:
When the pin does not output (during reset), the pin level becomes VSS.
When the pin outputs high, the level becomes VCC, and when the pin outputs low, the level becomes VSS.

The pin may be the CMOS output pin or switching to the N-channel open drain output pin may fail. Check if the pin tested is the right pin. Also examine the program whether the setting for N-channel open drain output is specified correctly.

4.7 Investigating the Cause of Interrupt Problems

This section describes how to investigate what causes an interrupt to be skipped or not to be generated.

4.7.1 Examining the Program Using ICE (E100)

The following two may cause an interrupt to be skipped or not to be generated:

- Address 00000h is read.
- When an interrupt request is generated, the interrupt request is cleared before transition to the interrupt handler is made.

This section introduces the method to examine the program if address 00000h is read.

Procedure

(1) Specify the HEW to have a break point when address 00000h is read.
- Select View >> Event >> Hardware break from the menu bar. The Hardware Break window opens.
- In the Hardware Break window, click the Add button.
- In the Event window, specify the settings as follows and click OK.
  - Event type: Data access
  - Access type: MCU bus, CPU, and DMAC selected
  - Address condition: Specified value (=) Start: 0000h
  - Read/write: Read/Write
- Click the Apply button in the Hardware Break window to apply the settings.

(2) Run the program to see if the break occurs.

Explanation

Check whether the break occurs in step (2) above. If the break occurred, address 00000h is read. Open the trace window and determine the point where address 00000h is read in the program.

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) Reproducibility
(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
- Communication mode
- Transfer rate

(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 Serial Communications in Development

Rev.	Date	Description	
		Page	Summary
1.10	Apr. 1, 2014	—	First edition issued
1.20	Dec.15,2017	11	1.3.1 Delete description about send buffer empty flag from sentences
		29	2.3.1 Delete description about send buffer empty flag from sentences

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit 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 Microprocessing unit or Microcontroller unit products 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. 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 shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended 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; industrial robots; etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, 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.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

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

(Rev.4.0-1 November 2017)

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.
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

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: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 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 Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, 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 1207, 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 India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.0