APPLICATION NOTE

V850E2/MN4

UARTJ Control

R01AN0926EJ0100
Rev.1.00
Feb 07, 2012

Introduction

This application note explains how to set up the UARTJ serial interface (with FIFO) and also gives an outline of the operation and describes the procedures for using a sample program. The sample program sets the baud rate to 19200 bps and executes serial communication between the UARTJ1 and the UARTJ3. The UARTJ1 transmits data and the UARTJ3 receives the data. The internal RAM has a 16-byte user transmit array and a 16-byte user receive array.

Target Device

V850E2/MN4 Microcontrollers

Contents

1. Overview

This application note illustrates the usage examples of the UARTJ.

The sample program makes the basic initial settings of the V850E2/MN4 microcontrollers, such as the selection of the clock frequency and the setup of port I/O. The main processing after the end of initialization executes serial communication between the UARTJ1 and the UARTJ3. The UARTJ1 transmits data and the UARTJ3 receives the data.

The main points in the software are shown below.

See sections 4.1.2 and 4.1.3 for the details of the individual transfer processing.

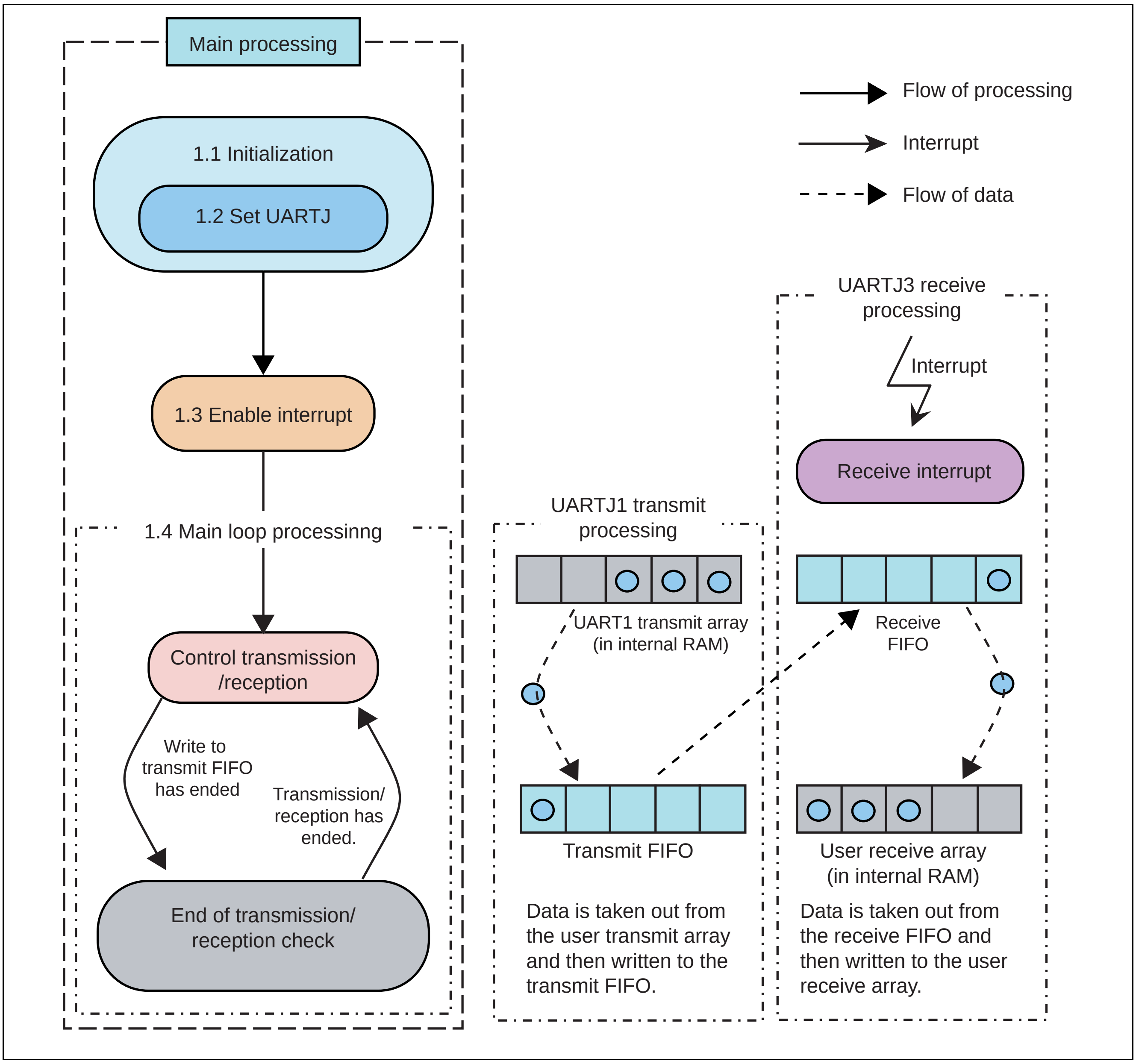

Main processing
1.1 Initialization
1.2 Set UARTJ
1.3 Enable interrupt
1.4 Main loop processinng
Control transmission /reception
Write to transmit FIFO has ended
Transmission/ reception has ended.
End of transmission/ reception check
Flow of processing
Interrupt
Flow of data
UARTJ3 receive processing
Interrupt
Receive interrupt
UARTJ1 transmit processing
UART1 transmit array (in internal RAM)
Transmit FIFO
Receive FIFO
User receive array (in internal RAM)
Data is taken out from the user transmit array and then written to the transmit FIFO.
Data is taken out from the receive FIFO and then written to the user receive array.

Figure 1.1 UARTJ Software Processing Flow

The basic communication specifications are shown below.

Receive I/F	UARTJ3
Transmit I/F	UARTJ1
Operation mode	Continuous transmission/reception using FIFOs
Transfer direction	LSB first
Parity bits during transmission/reception	No parity bit
Transmit data of 1 frame	8 bits
Stop bit	1 bit
Baud rate	19200 bps
Transmit/receive FIFO size	16 bytes

1.1 Initialization

The general registers and functional pins are initialized.

<Port setup>

- Port n function control expansion registers (PFCEn)
- Port n function control registers (PFCn)
- Port n mode control registers (PMCn)
- Port n mode registers (PMn)

1.2 UARTJ Setup

The registers listed below are set up to control the operation of the UARTJ. See section 4.2 for details.

<UART control setup>

- UARTJn control register 0 (URTJnCTL0)
- UARTJn control register 1 (URTJnCTL1)
- UARTJn control register 2 (URTJnCTL2)

<FIFO control setup>

- UARTJn macro FIFO control register 0 (URTJnFCTL0)
- UARTJn macro FIFO control register 1 (URTJnFCTL1)

1.3 Interrupt Enabling

- Interrupts are enabled by the EI instruction.

1.4 Main Loop Processing

The main loop processing is mainly divided into two. It repeatedly controls transmission/reception and awaits the end of transmission/reception. If transmit data is written to the URTJnFTX register after the operation of the UARTJ is set, the data is written to the transmit FIFO. The data stored in the transmit FIFO is transferred to the transmit shift register, after which serial transmission starts via the TXDnF pin.

In this sample program, the transmitting UARTJ1 transmits 16-byte data and the receiving UARTJ3 receives the 16-byte data from the UARJ1. The internal RAM has the 16-byte (one byte (one block data) 16) user transmit array to store transmit data and the 16-byte user receive array to store receive data.

When a transmit/receive interrupt or a status interrupt occurs during the main loop processing, the corresponding interrupt processing is executed. In this sample program, interrupts occur at the timings listed below. See section 4 for the details of the individual interrupt processing.

- Transmit interrupt: When the transmit FIFO becomes empty
- Receive interrupt: When the receive FIFO becomes full
- Status interrupt: When an error occurs during reception or transmission

2. Usage Environment

This section provides the circuit diagram and operating environment of the hardware on which this sample program is to run.

2.1 Circuit Diagram

See “V850E2/MN4 Target Board User Manual: QB-V850E2MN4DUAL-TB (R20UT0683XJ)” for the details of the circuit diagram.

This sample program performs serial communication between the UARTJ1 and the UARTJ3. The UARTJ1 transmits data and the UARTJ3 receives the data. The P13_4 pin and the P13_5 pin are used for the TXD1F pin and the RXD1F pin for the UARTJ1, respectively. The P4_6 pin and the P4_7 pin are used for the TXD3F pin and the RXD3F pin for the UARTJ3, respectively. The TXD1F pin is connected to the RXD1F pin and the RXD1F pin is connected to the TXD3F pin.

LED1 and LED2 are connected to port 13. The P13_7 pin is used for LED1. The P13_6 pin is used for LED2.

2.2 Development Environment

It is necessary to install the tools that are listed below to run the sample program.

- CubeSuite+
  The integrated development environment CubeSuite+ from Renesas Electronics provides various software development tools that are necessary for the user to develop applications. The user can use these tools seamlessly and easily in various development stages including coding, assembly, compilation, debugging using an emulator or simulator, and flash programming.

- MINICUBE
  MINICUBE is a general-purpose in-circuit emulator from Renesas Electronics which adopts the JTAG interface system. It allows the user to debug an onboard real processor and provides highly transparent and stable emulation functionalities. An adapter is required to connect a TB board to MINICUBE.

3. Software

This section describes the file organization of the sample program.

3.1 File Organization

The file organization of the sample program is summarized below.

File Name (Tool Structure)	Description	Common Source File	CubeSuite+ File
crtE.s	Hardware initialization processing		●
V850E2MN4.dir	Link/directive file		●
uartj.h	Variable and function declarations	●	
main.c	Main processing	●	
initial.c	Software initialization processing	●	
uartj_transmit.c	Transmit processing	●	
uartj_receive.c	Receive processing	●	
interrupt.c	Interrupt processing	●	

4. Sample Application

This section explains how to set up the UARTJ functions.

4.1 Flow Charts

The flow charts of this sample program are given below.

4.1.1 Main Processing

The main processing disables maskable interrupts first. After each setup ends, the maskable interrupts and transmit/receive status interrupts are enabled. The main loop processing repeatedly controls transmission/reception and awaits the end of FIFO transmission/reception.

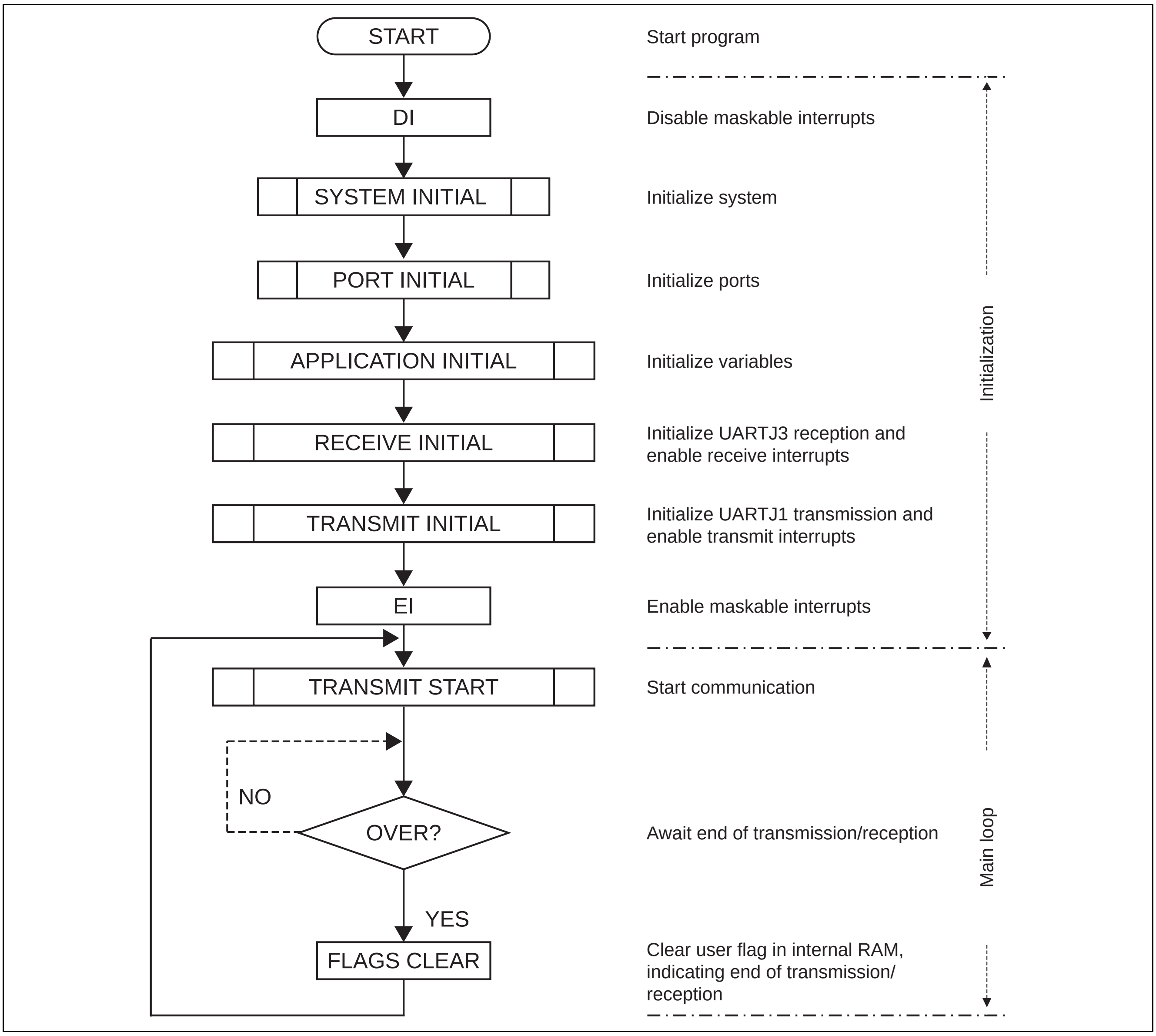

START
DI
SYSTEM INITIAL
PORT INITIAL
APPLICATION INITIAL
RECEIVE INITIAL
TRANSMIT INITIAL
EI
TRANSMIT START
NO
OVER?
YES
FLAGS CLEAR
Start program
Disable maskable interrupts
Initialize system
Initialize ports
Initialize variables
Initialize UARTJ3 reception and enable receive interrupts
Initialize UARTJ1 transmission and enable transmit interrupts
Enable maskable interrupts
Start communication
Await end of transmission/reception
Clear user flag in internal RAM, indicating end of transmission/ reception
Initialization
Main loop

Figure 4.1 Main Processing Flowchart

4.1.2 Receive Interrupt Processing

When data is received via the UARTJ, the data in the receive FIFO is read. If the stage of the receive FIFO matches the value set in the URTJnSLRP, a receive interrupt occurs.

During receive interrupt processing, data is read from the receive FIFO and the receive data is stored in the user receive array the number of times the data has been received. After all of the receive data is stored in the user receive array, a flag (regarded as a reception end flag) in the internal RAM is set to 1, indicating that all data (16 bytes) have been stored in the user receive array.

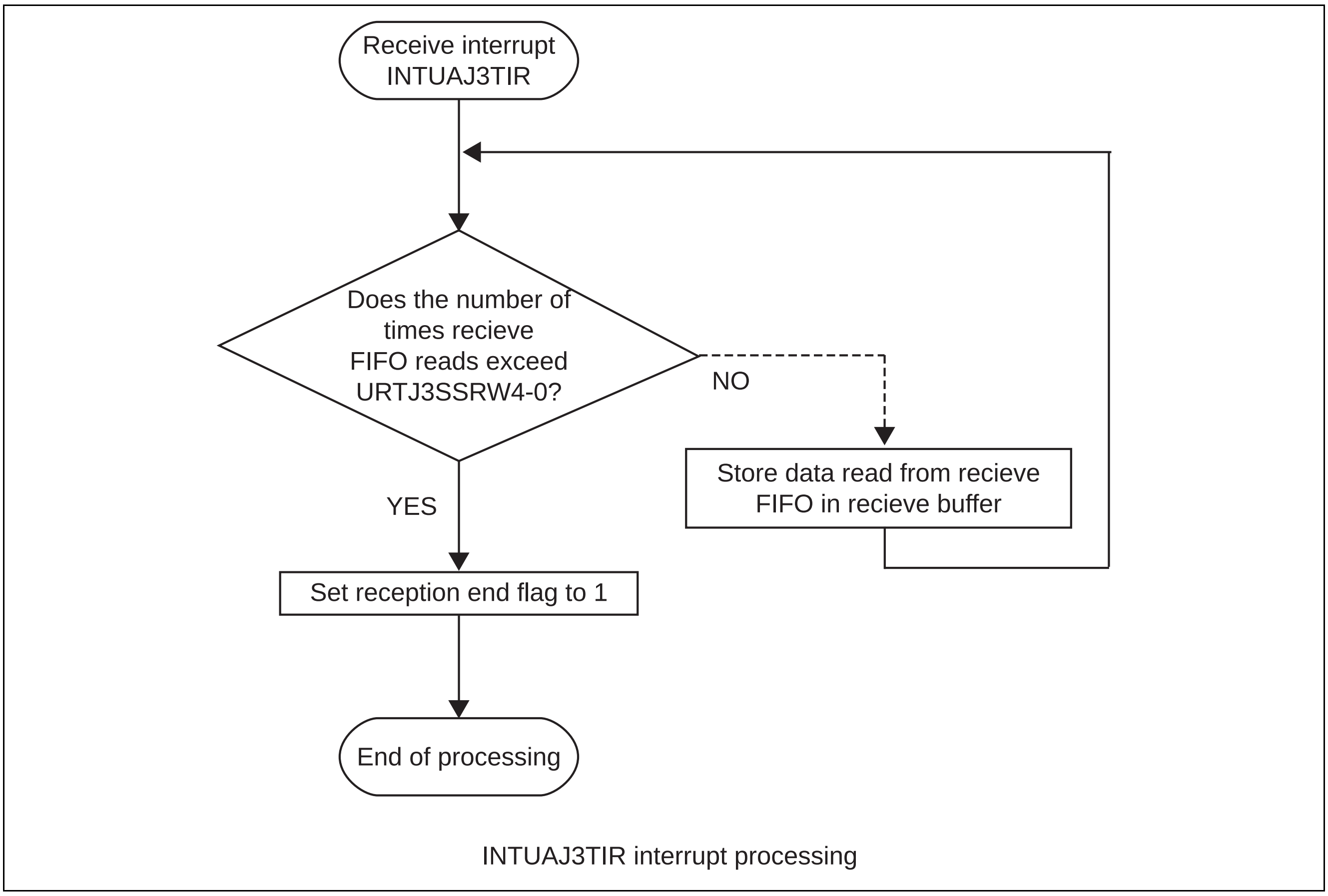

Receive interrupt
INTUAJ3TIR
Does the number of times recieve FIFO reads exceed URTJ3SSRW4-0?
NO
YES
Store data read from recieve FIFO in recieve buffer
Set reception end flag to 1
End of processing
INTUAJ3TIR interrupt processing

Figure 4.2 Receive Interrupt Processing

4.1.3 Transmit Interrupt Processing

If the fill stage of the transmit FIFO matches the value set in the URTJnSLTP and write access to the URTJ1FTX register is not executed, a transmit interrupt occurs.

During transmit interrupt processing, a flag (regarded as a transmit end flag) in the internal RAM is set to 1, indicating that all data (16 bytes) have been transferred to the transmit FIFO.

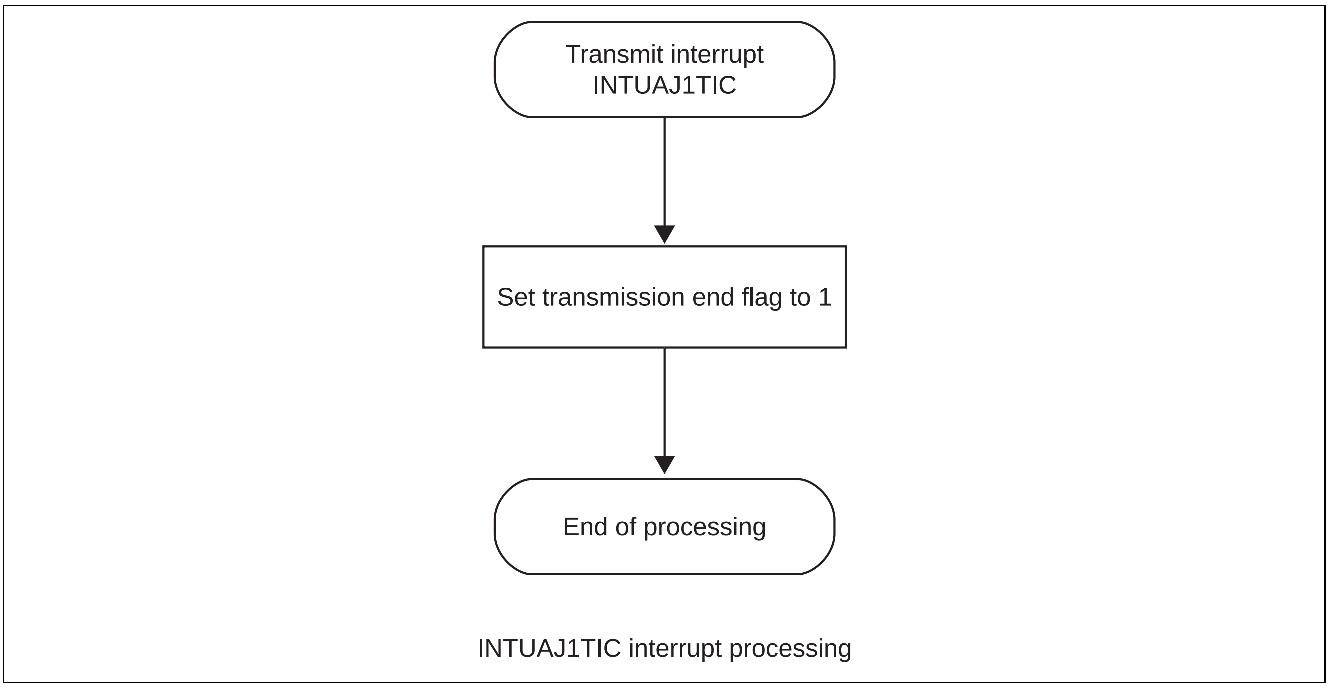

Transmit interrupt
INTUAJ1TIC
Set transmission end flag to 1
End of processing
INTUAJ1TIC interrupt processing

Figure 4.3 Transmit Interrupt Processing

4.1.4 Transmit/Receive Control Processing

When transmit data is written to the URTJ1FTX register after the initialization of the UARTJ, transmission starts. Receiving a start bit from the transmitter, the receiver starts reception.

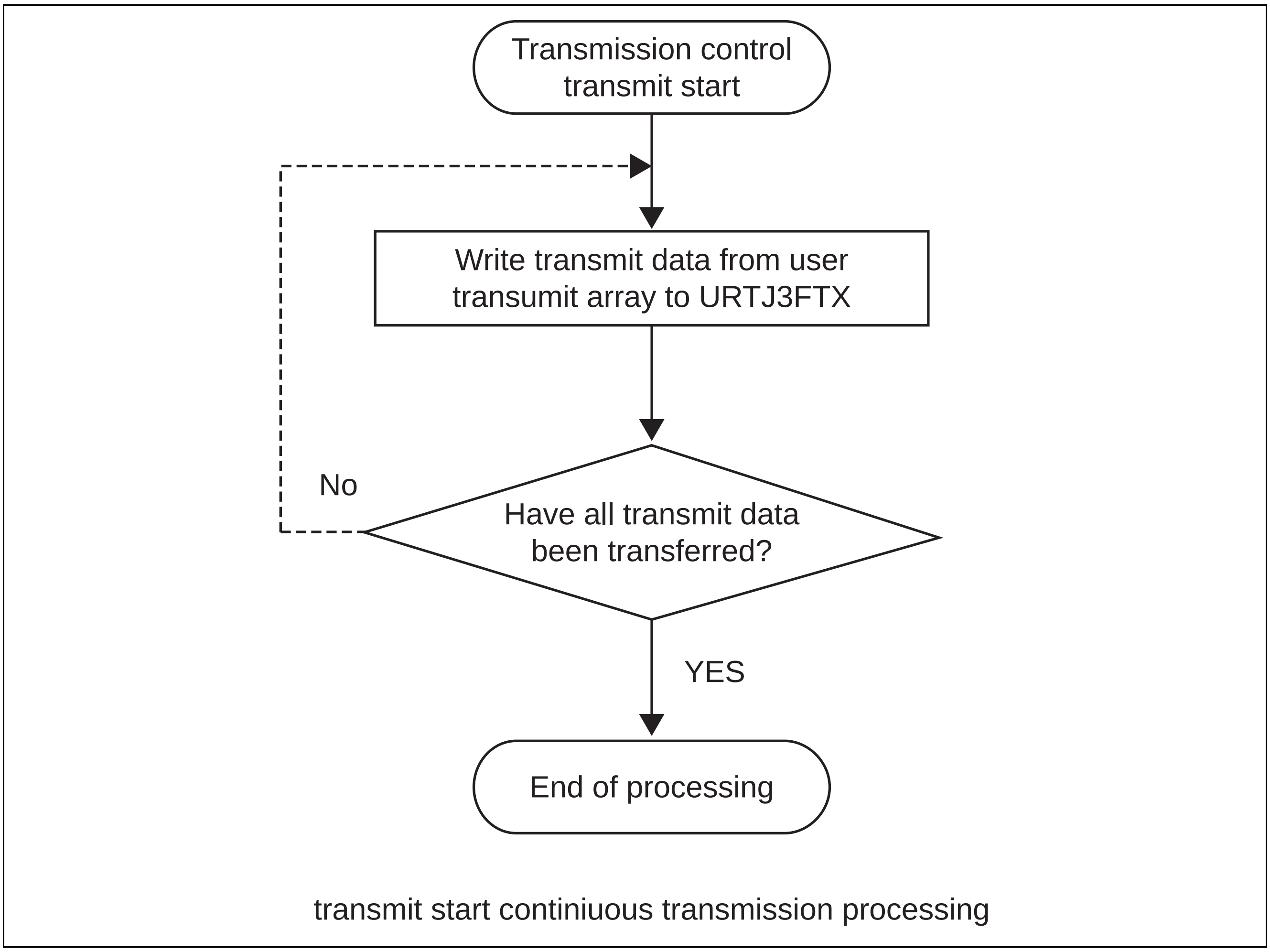

Transmission control
transmit start
Write transmit data from user
transumit array to URTJ3FTX
No
Have all transmit data
been transferred?
YES
End of processing
transmit start continiuous transmission processing

Figure 4.4 Transmit/Receive Control Processing

4.1.5 Status Interrupt Processing

A status interrupt request occurs when an error occurs during transmission/reception.

- Data consistency error
- Timeout error
- Framing error
- Parity error
- Overrun error
- Overflow error

When an error is detected during transmission/reception, the correspondent error flags in the URTJnSTR1 and URTJnFSTR1 registers are set to 1. Then, the UARTJ status interrupt processing clears the transmit/receive FIFO. The transmit FIFO is cleared by waiting for a period equivalent to the one required to transfer 16 bytes of data. The receive FIFO is cleared by reading all of the received data from it. The transmit flag and the receive flag in the internal RAM are set to 1. The SFR error flags are cleared via the URTJnSTC and URTJnFSTC registers and the transmit/receive interrupt request or the status interrupt request is also cleared. The UARTJ1 and UARTJ3 are temporarily stopped and return to the main loop (Transmission/reception is resumed in the main loop processing).

The status interrupt processing below is common to the UARTJ1 and the UARTJ3.

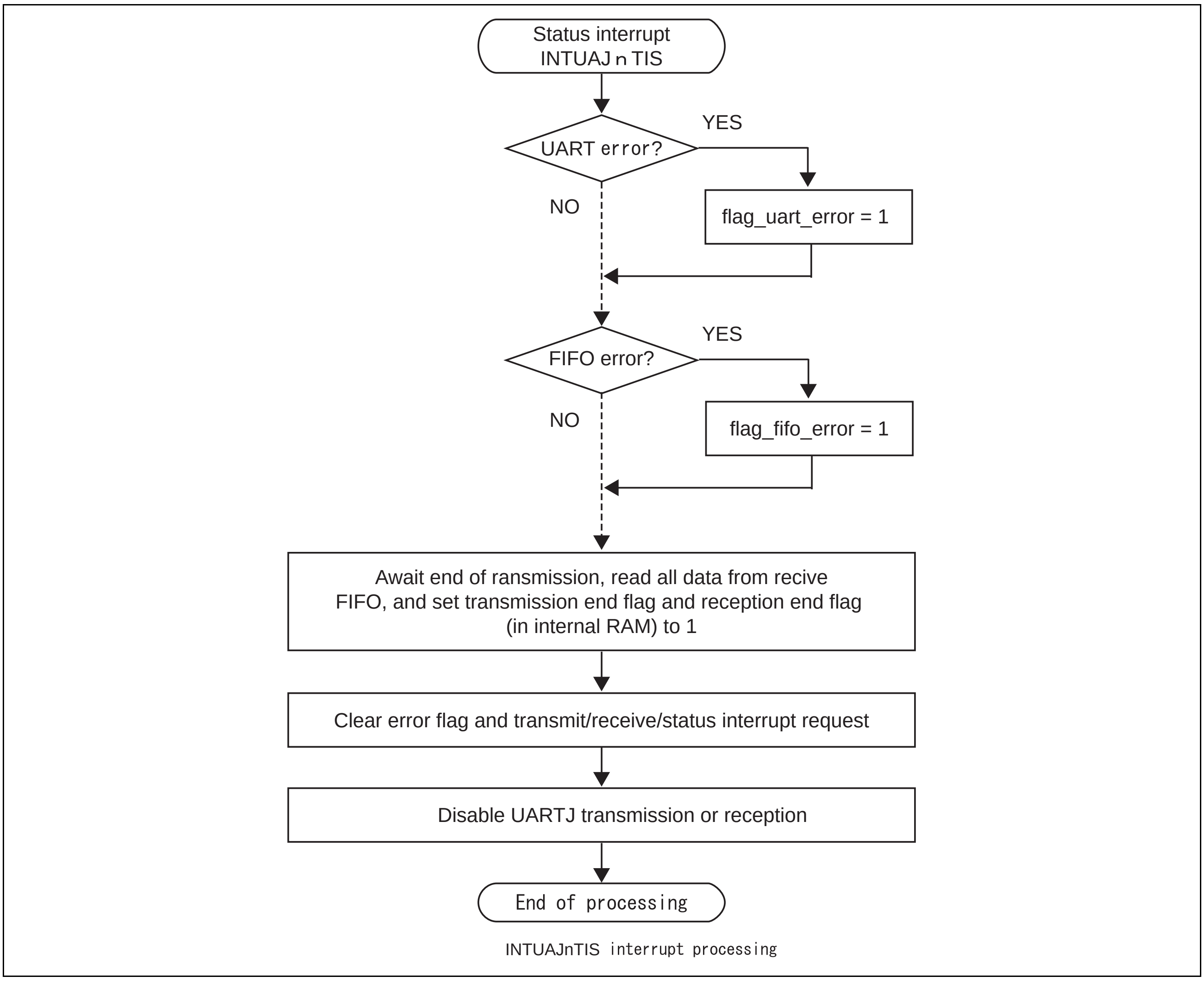

Status interrupt
INTUAJ n TIS
UART error?
YES
NO
flag_uart_error = 1
FIFO error?
YES
NO
flag_fifo_error = 1
Await end of ransmission, read all data from recive FIFO, and set transmission end flag and reception end flag (in internal RAM) to 1
Clear error flag and transmit/receive/status interrupt request
Disable UARTJ transmission or reception
End of processing
INTUAJnTIS interrupt processing

Figure 4.5 Status Interrupt Processing

4.2 Register Setup

This section explains how to set up the relevant registers according to the flow charts shown in section 4.1. The registers described below must be configured to control the UARTJ.

4.2.1 Port Setup

The program described in this application note executes serial transmission/reception by using two macros, the UARTJ1 and the UARTJ3. The relevant ports must be set up so that the pins for the UARTJ1 and the UARTJ3 are enabled.

The LEDs are connected to port 13. The P13_7 pin is used for LED1. The P13_6 pin is used for LED2.

Macro	Pin	PMC	PFCE	PFC	PM	Corresponding Function
UARTJ1	TXD1F	1	0	1	0	Alternative mode 2, output
	RXD1F	1	0	1	1	Alternative mode 2, input
UARTJ3	TXD3F	1	0	1	0	Alternative mode 2, output
	RXD3F	1	0	1	1	Alternative mode 2, input
PORT	P13_6	0	0	0	0	Port mode, output
	P13_7	0	0	0	0	Port mode, output

Setting examples

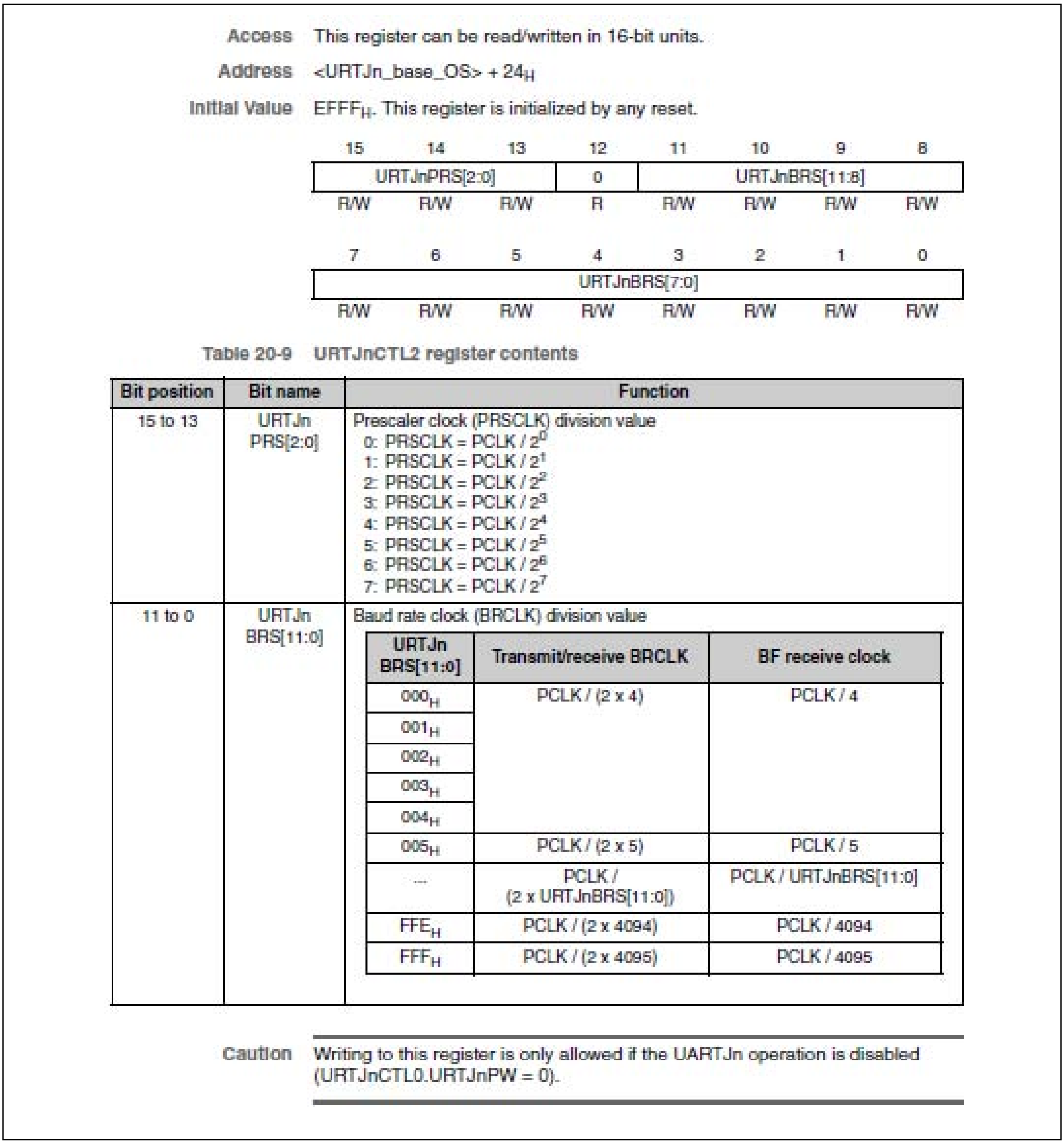
```
/* P4_6: URTJTA3RXD; alternative mode 2; input*/
/* P4_7: URTJTA3TXD; alternative mode 2; output*/
PFCE4 = 0x0000;
PFC4  = 0x00C0;
PMC4  = 0x00C0;
PM4   = 0x0040;

/* P13_6,7:LEDs; port mode; output*/
/* P13_5:URTJTA1RXD; alternative mode 2; input*/
/* P13_4:URTJTA1TXD; alternative mode 2; output*/
PFCE13 = 0x0000;
PFC13  = 0x0030;
PMC13  = 0x0030;
PM13   = 0x0020;
```

4.2.2 UARTJn Control Register 2 (URTJnCTL2)

The UARTJnCTL2 register defines the baud rates of the serial data transfers in which the UARTJn macros are used.

The transmit/receive clocks are generated from PCLK. The frequency of the clock signal from PCLK is divided by a prescaler and the clock of the specified baudrate is generated by the baudrate generator.

In this sample program, the baudrate is set to 19,200 bps.



Access This register can be read/written in 16-bit units.

Address <URTJn_base_OS> + 24_H

Initial Value $EFFF_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
URTJnPRS[2:0]			0	URTJnBRS[11:8]			
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
URTJnBRS[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-9 URTJnCTL2 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15 to 13</td><td>URTJn PRS[2:0]</td><td>Prescaler clock (PRSCLK) division value
0: PRSCLK = PCLK / 2^0
1: PRSCLK = PCLK / 2^1
2: PRSCLK = PCLK / 2^2
3: PRSCLK = PCLK / 2^3
4: PRSCLK = PCLK / 2^4
5: PRSCLK = PCLK / 2^5
6: PRSCLK = PCLK / 2^6
7: PRSCLK = PCLK / 2^7</td></tr>
<tr><td>11 to 0</td><td>URTJn BRS[11:0]</td><td>Baud rate clock (BRCLK) division value
<table>
<tr><th>URTJn BRS[11:0]</th><th>Transmit/receive BRCLK</th><th>BF receive clock</th></tr>
<tr><td>000_H</td><td rowspan="5">PCLK / (2 x 4)</td><td rowspan="5">PCLK / 4</td></tr>
<tr><td>001_H</td></tr>
<tr><td>002_H</td></tr>
<tr><td>003_H</td></tr>
<tr><td>004_H</td></tr>
<tr><td>005_H</td><td>PCLK / (2 x 5)</td><td>PCLK / 5</td></tr>
<tr><td>....</td><td>PCLK / (2 x URTJnBRS[11:0])</td><td>PCLK / URTJnBRS[11:0]</td></tr>
<tr><td>FFE_H</td><td>PCLK / (2 x 4094)</td><td>PCLK / 4094</td></tr>
<tr><td>FFF_H</td><td>PCLK / (2 x 4095)</td><td>PCLK / 4095</td></tr>
</table></td></tr>
</table>

Caution Writing to this register is only allowed if the UARTJn operation is disabled (URTJnCTL0.URTJnPW = 0).

Figure 4.6 URTJnCTL2 Register Format

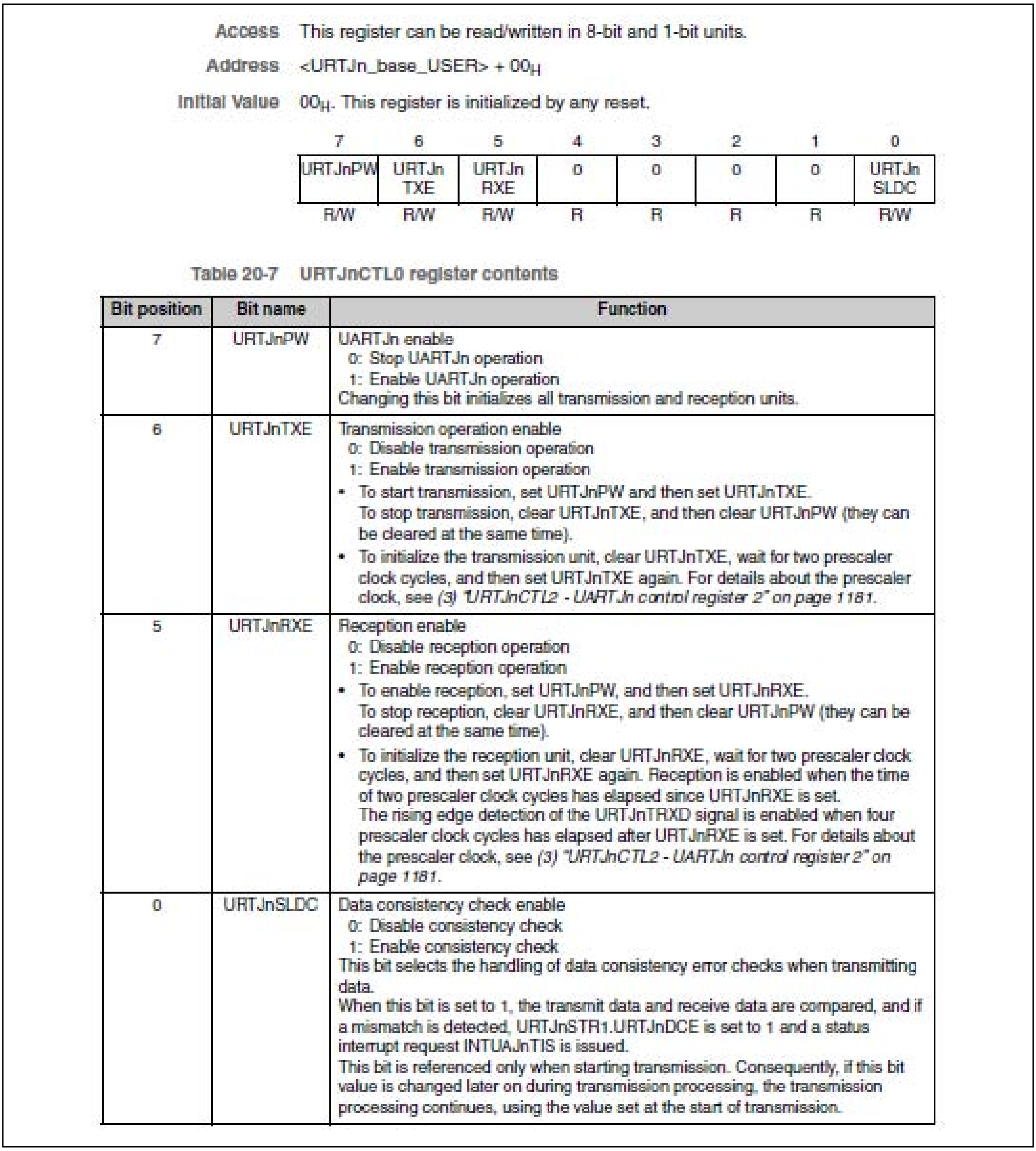
Setting example

```
URTJnCTL2 = 0x60D9;          /* Assume that PCLK is set to 66.667 MHz */
                             /* PRSCLK (1/8), baud rate clock (2*217) */
                             /* 19200≈PCLK/ (8*2*217) */
```

4.2.3 UARTJn Control Register 0 (URTJnCTL0)

The UARTJnCTL0 register controls the basic serial transfer operation of the UARTJn macros.

The sample program disables the macro operations, sets the required control registers, and then, enables the UARTJn operation.



Access This register can be read/written in 8-bit and 1-bit units.

Address <URTJn_base_USER> + 00_H

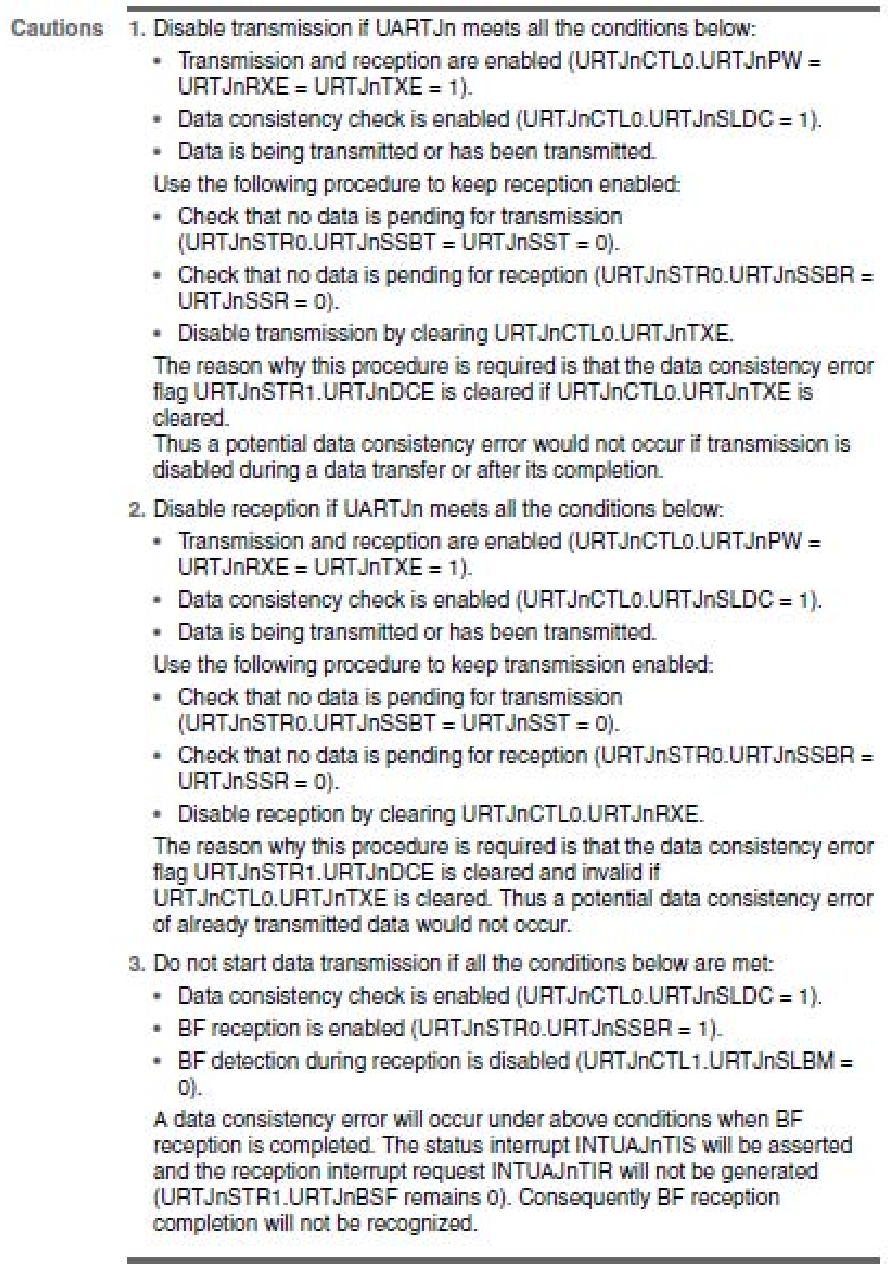
Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
URTJnPW	URTJn TXE	URTJn RXE	0	0	0	0	URTJn SLDC
R/W	R/W	R/W	R	R	R	R	R/W

Table 20-7 URTJnCTL0 register contents

Bit position	Bit name	Function
7	URTJnPW	UARTJn enable 0: Stop UARTJn operation 1: Enable UARTJn operation Changing this bit initializes all transmission and reception units.
6	URTJnTXE	Transmission operation enable 0: Disable transmission operation 1: Enable transmission operation • To start transmission, set URTJnPW and then set URTJnTXE. To stop transmission, clear URTJnTXE, and then clear URTJnPW (they can be cleared at the same time). • To initialize the transmission unit, clear URTJnTXE, wait for two prescaler clock cycles, and then set URTJnTXE again. For details about the prescaler clock, see *(3) "URTJnCTL2 - UARTJn control register 2" on page 1181*.
5	URTJnRXE	Reception enable 0: Disable reception operation 1: Enable reception operation • To enable reception, set URTJnPW, and then set URTJnRXE. To stop reception, clear URTJnRXE, and then clear URTJnPW (they can be cleared at the same time). • To initialize the reception unit, clear URTJnRXE, wait for two prescaler clock cycles, and then set URTJnRXE again. Reception is enabled when the time of two prescaler clock cycles has elapsed since URTJnRXE is set. The rising edge detection of the URTJnTRXD signal is enabled when four prescaler clock cycles has elapsed after URTJnRXE is set. For details about the prescaler clock, see *(3) "URTJnCTL2 - UARTJn control register 2" on page 1181*.
0	URTJnSLDC	Data consistency check enable 0: Disable consistency check 1: Enable consistency check This bit selects the handling of data consistency error checks when transmitting data. When this bit is set to 1, the transmit data and receive data are compared, and if a mismatch is detected, URTJnSTR1.URTJnDCE is set to 1 and a status interrupt request INTUAJnTIS is issued. This bit is referenced only when starting transmission. Consequently, if this bit value is changed later on during transmission processing, the transmission processing continues, using the value set at the start of transmission.

Figure 4.7 URTJnCTL0 Register Format (1/2)



Cautions

1. Disable transmission if UARTJn meets all the conditions below:
   - Transmission and reception are enabled (URTJnCTL0.URTJnPW = URTJnRXE = URTJnTXE = 1).
   - Data consistency check is enabled (URTJnCTL0.URTJnSLDC = 1).
   - Data is being transmitted or has been transmitted.

   Use the following procedure to keep reception enabled:
   - Check that no data is pending for transmission (URTJnSTR0.URTJnSSBT = URTJnSST = 0).
   - Check that no data is pending for reception (URTJnSTR0.URTJnSSBR = URTJnSSR = 0).
   - Disable transmission by clearing URTJnCTL0.URTJnTXE.

   The reason why this procedure is required is that the data consistency error flag URTJnSTR1.URTJnDCE is cleared if URTJnCTL0.URTJnTXE is cleared.
   Thus a potential data consistency error would not occur if transmission is disabled during a data transfer or after its completion.
2. Disable reception if UARTJn meets all the conditions below:
   - Transmission and reception are enabled (URTJnCTL0.URTJnPW = URTJnRXE = URTJnTXE = 1).
   - Data consistency check is enabled (URTJnCTL0.URTJnSLDC = 1).
   - Data is being transmitted or has been transmitted.

   Use the following procedure to keep transmission enabled:
   - Check that no data is pending for transmission (URTJnSTR0.URTJnSSBT = URTJnSST = 0).
   - Check that no data is pending for reception (URTJnSTR0.URTJnSSBR = URTJnSSR = 0).
   - Disable reception by clearing URTJnCTL0.URTJnRXE.

   The reason why this procedure is required is that the data consistency error flag URTJnSTR1.URTJnDCE is cleared and invalid if URTJnCTL0.URTJnTXE is cleared. Thus a potential data consistency error of already transmitted data would not occur.
3. Do not start data transmission if all the conditions below are met:
   - Data consistency check is enabled (URTJnCTL0.URTJnSLDC = 1).
   - BF reception is enabled (URTJnSTR0.URTJnSSBR = 1).
   - BF detection during reception is disabled (URTJnCTL1.URTJnSLBM = 0).

   A data consistency error will occur under above conditions when BF reception is completed. The status interrupt INTUAJnTIS will be asserted and the reception interrupt request INTUAJnTIR will not be generated (URTJnSTR1.URTJnBSF remains 0). Consequently BF reception completion will not be recognized.

Figure 4.8 URTJnCTL0 Register Format (2/2)

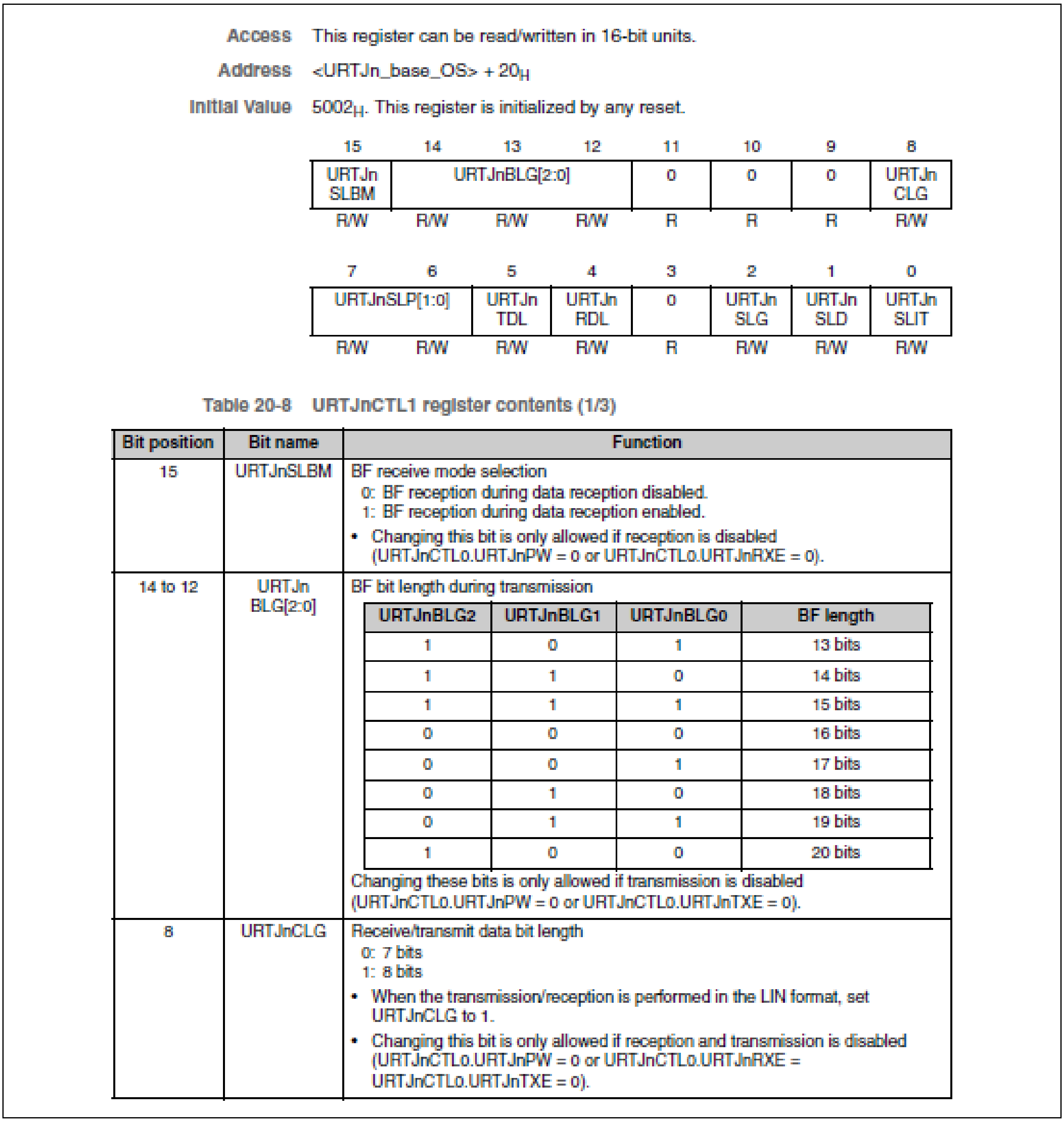
Setting examples

```
URTJnPW = 1;                    /* Enable UARTJn */
URTJnTXE = 1;                   /* Enable transmission */
URTJnRXE = 1;                   /* Enable reception */
URTJnBYTE = 0x00;               /* Disable UARTJn */
```

4.2.4 UARTJn Control Register 1 (URTJnCTL1)

The UARTJnCTL1 register defines the data frame properties of the serial data transfers in which the UARTJn macros are used.

This sample program does not use the functions related to the BF-related transmission/reception.



Access This register can be read/written in 16-bit units.

Address <URTJn_base_OS> + 20_H

Initial Value 5002_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
URTJn SLBM	URTJnBLG[2:0]			0	0	0	URTJn CLG
R/W	R/W	R/W	R/W	R	R	R	R/W

7	6	5	4	3	2	1	0
URTJnSLP[1:0]		URTJn TDL	URTJn RDL	0	URTJn SLG	URTJn SLD	URTJn SLIT
R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

Table 20-8 URTJnCTL1 register contents (1/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15</td><td>URTJnSLBM</td><td>BF receive mode selection
0: BF reception during data reception disabled.
1: BF reception during data reception enabled.
• Changing this bit is only allowed if reception is disabled (URTJnCTL0.URTJnPW = 0 or URTJnCTL0.URTJnRXE = 0).</td></tr>
<tr><td>14 to 12</td><td>URTJn
BLG[2:0]</td><td>BF bit length during transmission
<table>
<tr><th>URTJnBLG2</th><th>URTJnBLG1</th><th>URTJnBLG0</th><th>BF length</th></tr>
<tr><td>1</td><td>0</td><td>1</td><td>13 bits</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>14 bits</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>15 bits</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>16 bits</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>17 bits</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>18 bits</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>19 bits</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>20 bits</td></tr>
</table>
Changing these bits is only allowed if transmission is disabled (URTJnCTL0.URTJnPW = 0 or URTJnCTL0.URTJnTXE = 0).</td></tr>
<tr><td>8</td><td>URTJnCLG</td><td>Receive/transmit data bit length
0: 7 bits
1: 8 bits
• When the transmission/reception is performed in the LIN format, set URTJnCLG to 1.
• Changing this bit is only allowed if reception and transmission is disabled (URTJnCTL0.URTJnPW = 0 or URTJnCTL0.URTJnRXE = URTJnCTL0.URTJnTXE = 0).</td></tr>
</table>

Figure 4.9 URTJnCTL1 Register Format (1/3)

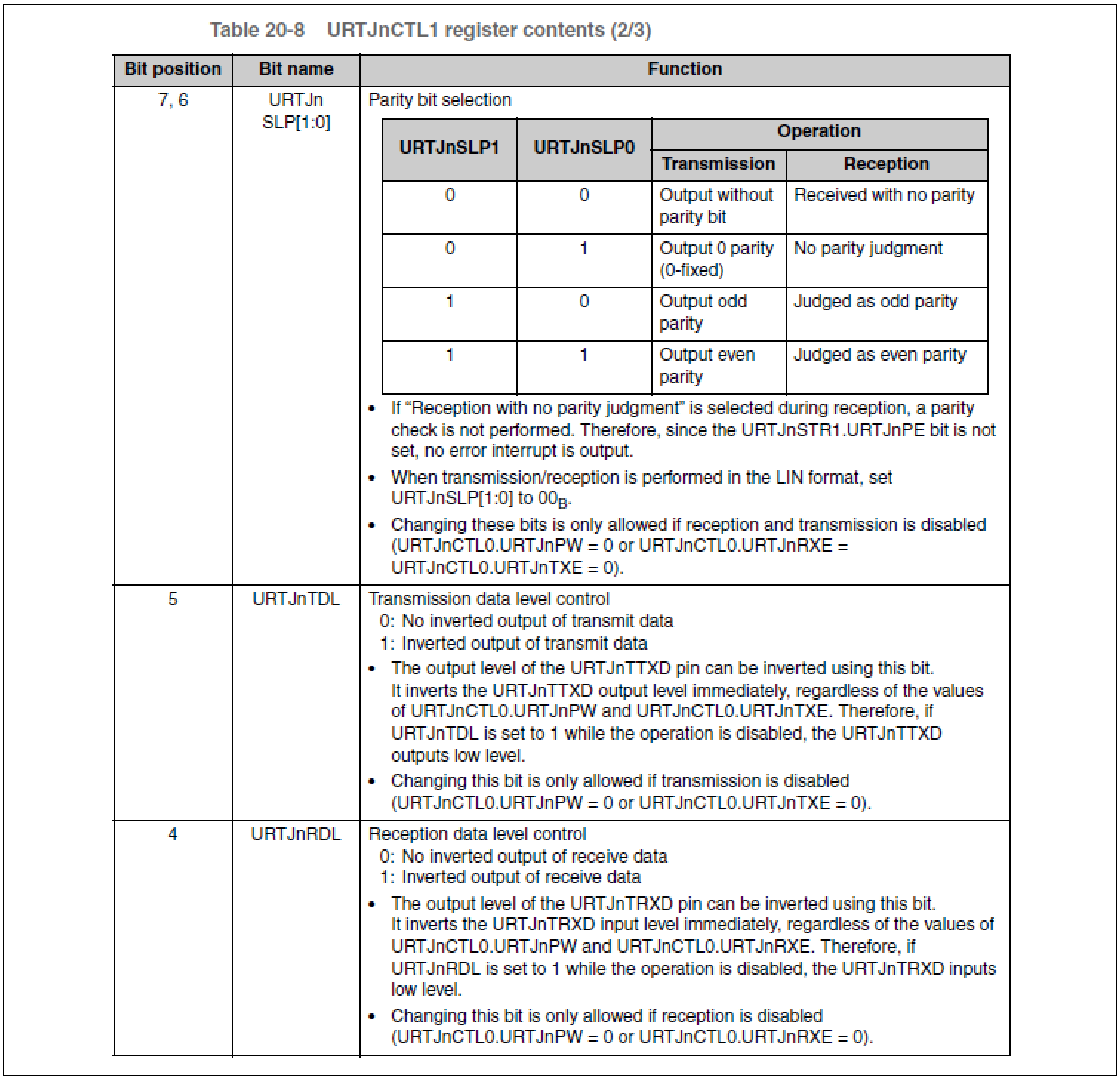


Table 20-8 URTJnCTL1 register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7, 6</td><td>URTJn
SLP[1:0]</td><td>Parity bit selection
<table>
<tr><th rowspan="2">URTJnSLP1</th><th rowspan="2">URTJnSLP0</th><th colspan="2">Operation</th></tr>
<tr><th>Transmission</th><th>Reception</th></tr>
<tr><td>0</td><td>0</td><td>Output without parity bit</td><td>Received with no parity</td></tr>
<tr><td>0</td><td>1</td><td>Output 0 parity (0-fixed)</td><td>No parity judgment</td></tr>
<tr><td>1</td><td>0</td><td>Output odd parity</td><td>Judged as odd parity</td></tr>
<tr><td>1</td><td>1</td><td>Output even parity</td><td>Judged as even parity</td></tr>
</table>
• If “Reception with no parity judgment” is selected during reception, a parity check is not performed. Therefore, since the URTJnSTR1.URTJnPE bit is not set, no error interrupt is output.
• When transmission/reception is performed in the LIN format, set URTJnSLP[1:0] to 00_B.
• Changing these bits is only allowed if reception and transmission is disabled (URTJnCTL0.URTJnPW = 0 or URTJnCTL0.URTJnRXE = URTJnCTL0.URTJnTXE = 0).</td></tr>
<tr><td>5</td><td>URTJnTDL</td><td>Transmission data level control
0: No inverted output of transmit data
1: Inverted output of transmit data
• The output level of the URTJnTTXD pin can be inverted using this bit. It inverts the URTJnTTXD output level immediately, regardless of the values of URTJnCTL0.URTJnPW and URTJnCTL0.URTJnTXE. Therefore, if URTJnTDL is set to 1 while the operation is disabled, the URTJnTTXD outputs low level.
• Changing this bit is only allowed if transmission is disabled (URTJnCTL0.URTJnPW = 0 or URTJnCTL0.URTJnTXE = 0).</td></tr>
<tr><td>4</td><td>URTJnRDL</td><td>Reception data level control
0: No inverted output of receive data
1: Inverted output of receive data
• The output level of the URTJnTRXD pin can be inverted using this bit. It inverts the URTJnTRXD input level immediately, regardless of the values of URTJnCTL0.URTJnPW and URTJnCTL0.URTJnRXE. Therefore, if URTJnRDL is set to 1 while the operation is disabled, the URTJnTRXD inputs low level.
• Changing this bit is only allowed if reception is disabled (URTJnCTL0.URTJnPW = 0 or URTJnCTL0.URTJnRXE = 0).</td></tr>
</table>

Figure 4.10 URTJnCTL1 Register Format (2/3)

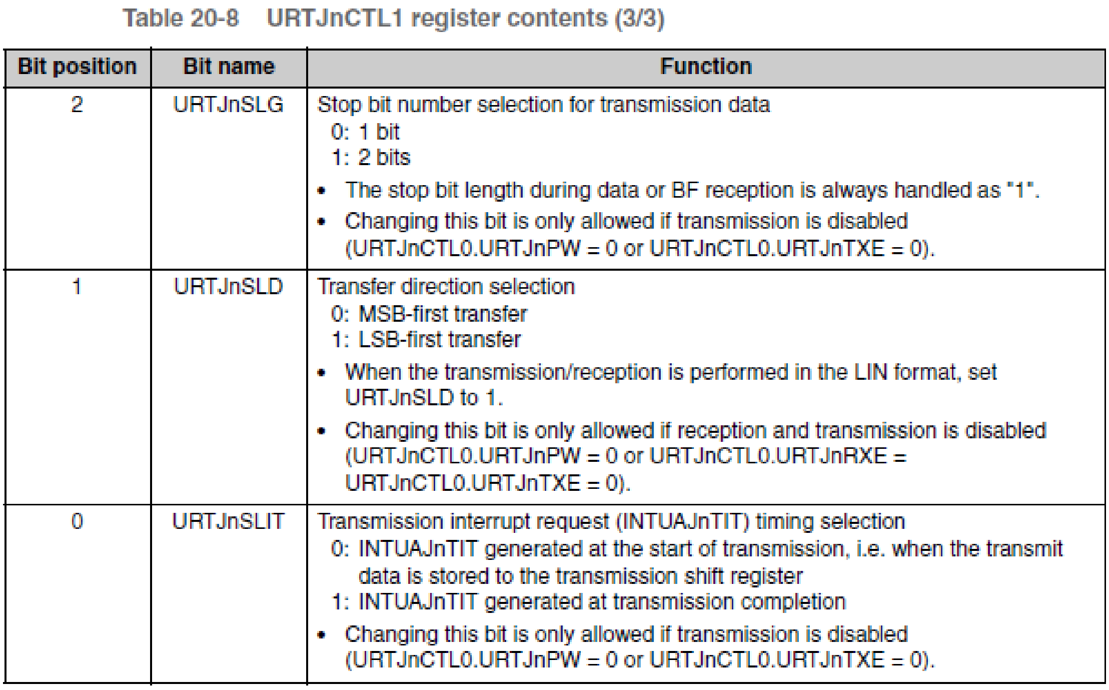


Table 20-8 URTJnCTL1 register contents (3/3)

Bit position	Bit name	Function
2	URTJnSLG	Stop bit number selection for transmission data 0: 1 bit 1: 2 bits • The stop bit length during data or BF reception is always handled as "1". • Changing this bit is only allowed if transmission is disabled (URTJnCTL0.URTJnPW = 0 or URTJnCTL0.URTJnTXE = 0).
1	URTJnSLD	Transfer direction selection 0: MSB-first transfer 1: LSB-first transfer • When the transmission/reception is performed in the LIN format, set URTJnSLD to 1. • Changing this bit is only allowed if reception and transmission is disabled (URTJnCTL0.URTJnPW = 0 or URTJnCTL0.URTJnRXE = URTJnCTL0.URTJnTXE = 0).
0	URTJnSLIT	Transmission interrupt request (INTUAJnTIT) timing selection 0: INTUAJnTIT generated at the start of transmission, i.e. when the transmit data is stored to the transmission shift register 1: INTUAJnTIT generated at transmission completion • Changing this bit is only allowed if transmission is disabled (URTJnCTL0.URTJnPW = 0 or URTJnCTL0.URTJnTXE = 0).

Figure 4.11 URTJnCTL1 Register Format (3/3)

Setting example:

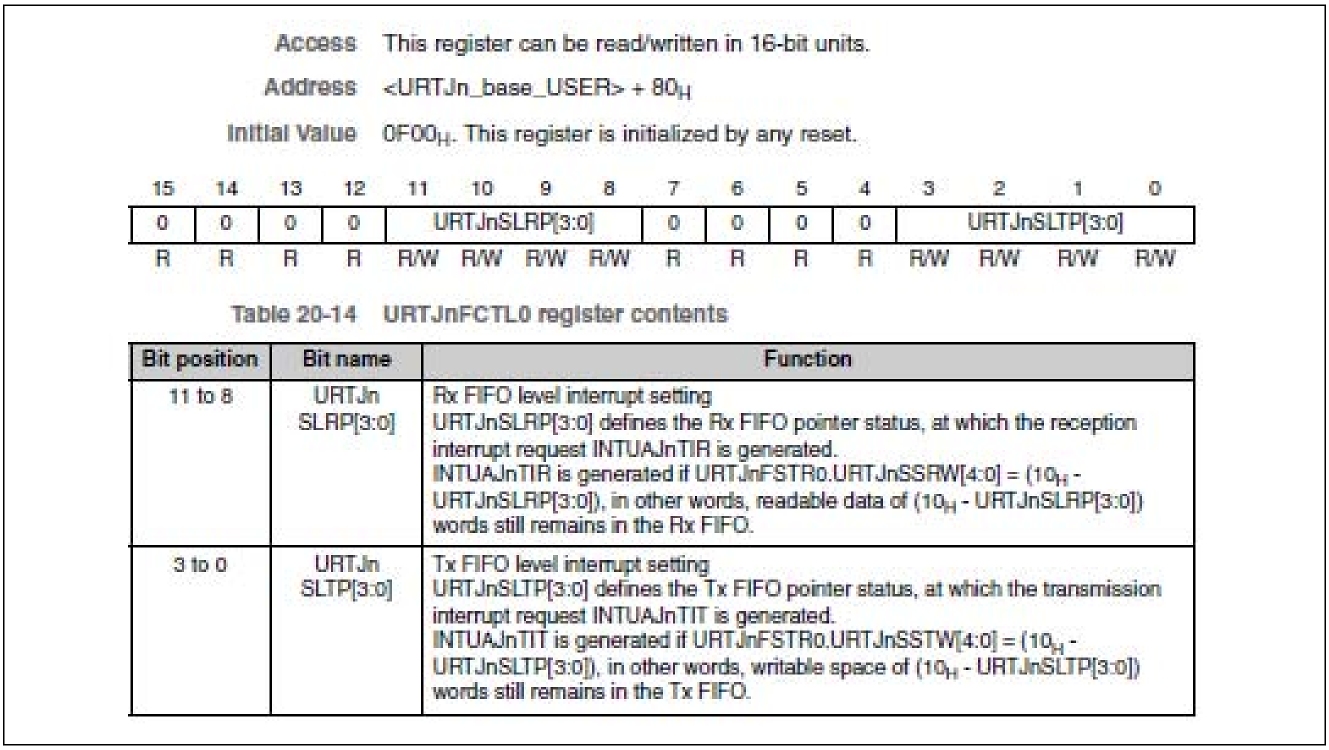
```
URTJnCTL1 = 0x5103;          /* Disable BF reception during data reception */
                             /* Receive/transmit data bit length: 8 bits */
                             /* No parity */
                             /* Transmit data output: No inversion, receive data input: No inversion */
                             /* Transmit data stop bit count: 1 bit */
                             /* Transfer direction select: LSB first transfer */
                             /* Transmit interrupt request generated at the end of transmission */
```

4.2.5 FIFO Control Register 0 (URTJnFCTL0)

The URTJnFCTL0 register defines the fill stages of the Rx FIFO and the Tx FIFO, at which the receive (INTUAJnTIR) and transmit (INTUAJnTIT) interrupts requests are generated.

A receive interrupt request is generated when the fill stage of the receive FIFO reaches the value specified in URTJnFCTL0.URTJnSLRP[3:0]. A transmit interrupt request is generated when the fill stage of the transmit FIFO reaches the value specified in URTJnFCTL0.URTJnSLTP[3:0].

The fill stages of the receive FIFO and the transmit FIFO need to be specified according to the baud rate. In this sample program, they are set to 16 bytes.



Access This register can be read/written in 16-bit units.

Address <URTJn_base_USER> + 80_H

Initial Value $0F00_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	URTJnSLRP[3:0]				0	0	0	0	URTJnSLTP[3:0]			
R	R	R	R	R/W	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

Table 20-14 URTJnFCTL0 register contents

Bit position	Bit name	Function
11 to 8	URTJn SLRP[3:0]	Rx FIFO level interrupt setting URTJnSLRP[3:0] defines the Rx FIFO pointer status, at which the reception interrupt request INTUAJnTIR is generated. INTUAJnTIR is generated if URTJnFSTR0.URTJnSSRW[4:0] = (10_H - URTJnSLRP[3:0]), in other words, readable data of (10_H - URTJnSLRP[3:0]) words still remains in the Rx FIFO.
3 to 0	URTJn SLTP[3:0]	Tx FIFO level interrupt setting URTJnSLTP[3:0] defines the Tx FIFO pointer status, at which the transmission interrupt request INTUAJnTIT is generated. INTUAJnTIT is generated if URTJnFSTR0.URTJnSSTW[4:0] = (10_H - URTJnSLTP[3:0]), in other words, writable space of (10_H - URTJnSLTP[3:0]) words still remains in the Tx FIFO.

Figure 4.12 URTJnFCTL0 Register Format

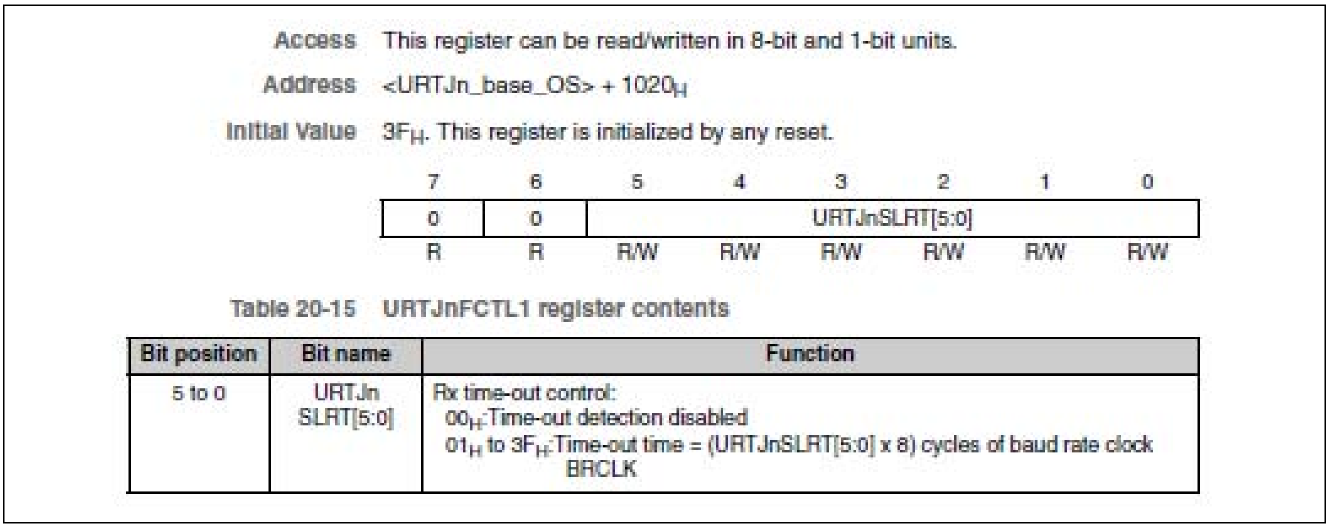
Setting example:

```
URTJnFCTL0 = 0x0000;           /* Receive FIFO interrupt level: 16 bytes */
                               /* Transmit FIFO interrupt level: 16 bytes */
```

4.2.6 FIFO Control Register 1 (URTJnFCTL1)

The URTJnFCTL1 register controls the Rx timeout detection.

A timeout error occurs, when the receive FIFO is not empty and when no receive data is stored in the receive FIFO or no data is read from the receive FIFO for a certain period of time.



Access This register can be read/written in 8-bit and 1-bit units.

Address <URTJn_base_OS> + 1020_H

Initial Value $3F_H$. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	URTJnSLRT[5:0]					
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-15 URTJnFCTL1 register contents

Bit position	Bit name	Function
5 to 0	URTJn SLRT[5:0]	Rx time-out control: 00_H:Time-out detection disabled 01_H to $3F_H$:Time-out time = (URTJnSLRT[5:0] x 8) cycles of baud rate clock BRCLK

Figure 4.13 URTJnFCTL1 Register Format

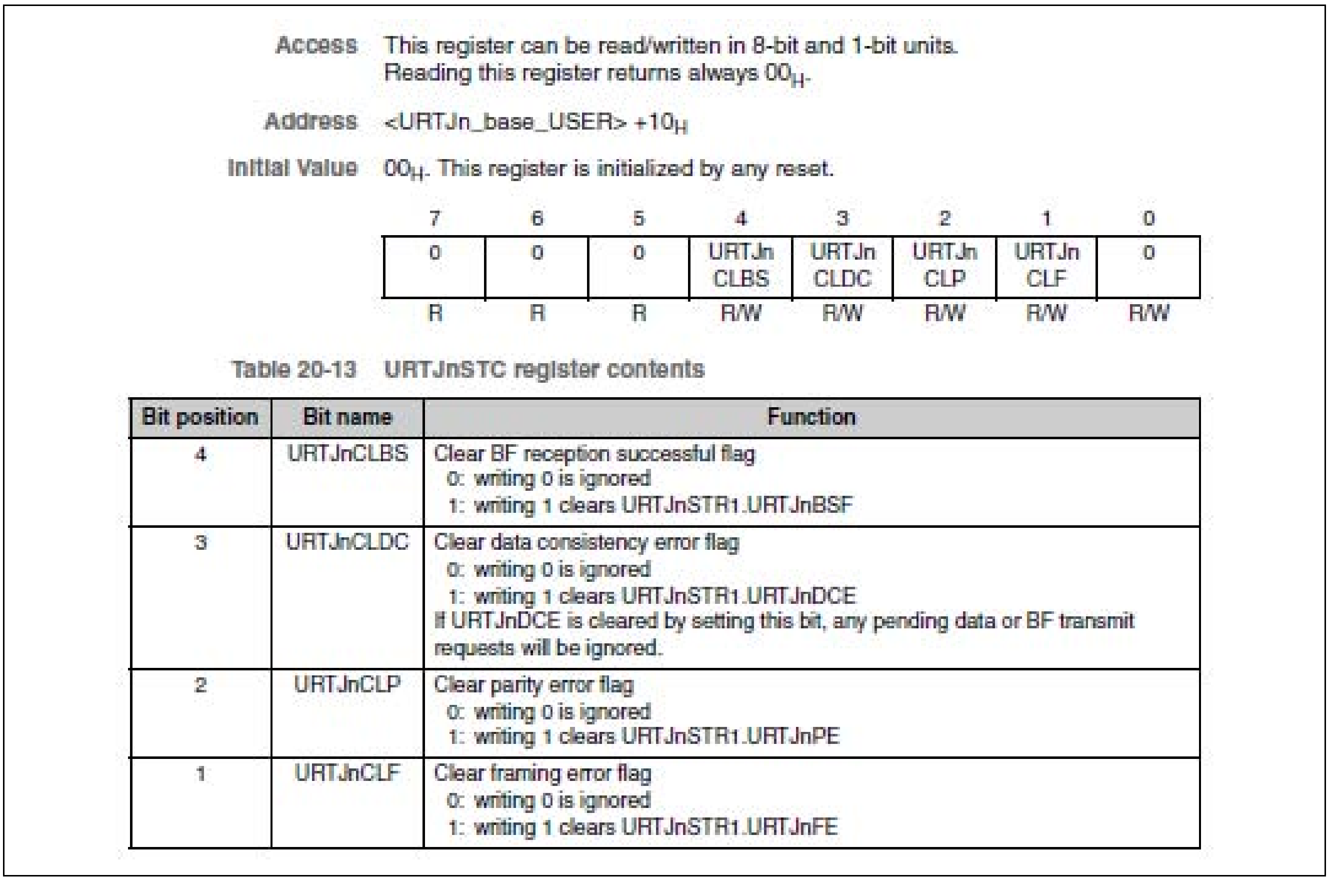
Setting example:

```
URTJnFCTL1 = 0x3F;        /* Detect timeout */
```

4.2.7 UARTJn Status Clear Register 0 (URTJnSTC)

The error flags in URTJnSTR1 are set when a data consistency error, a parity error, a framing error, and an overrun error occur. The URTJnSTC register can be used to clear the error flags in URTJnSTR1.

In this sample program, the corresponding error flags are cleared by using the URTJnSTC and URTJnFSTC registers when INTUAJnTIS is detected.



Access This register can be read/written in 8-bit and 1-bit units.
Reading this register returns always 00_H.

Address <URTJn_base_USER> +10_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	URTJn CLBS	URTJn CLDC	URTJn CLP	URTJn CLF	0
R	R	R	R/W	R/W	R/W	R/W	R/W

Table 20-13 URTJnSTC register contents

Bit position	Bit name	Function
4	URTJnCLBS	Clear BF reception successful flag 0: writing 0 is ignored 1: writing 1 clears URTJnSTR1.URTJnBSF
3	URTJnCLDC	Clear data consistency error flag 0: writing 0 is ignored 1: writing 1 clears URTJnSTR1.URTJnDCE If URTJnDCE is cleared by setting this bit, any pending data or BF transmit requests will be ignored.
2	URTJnCLP	Clear parity error flag 0: writing 0 is ignored 1: writing 1 clears URTJnSTR1.URTJnPE
1	URTJnCLF	Clear framing error flag 0: writing 0 is ignored 1: writing 1 clears URTJnSTR1.URTJnFE

Figure 4.14 URTJnSTC Register Format

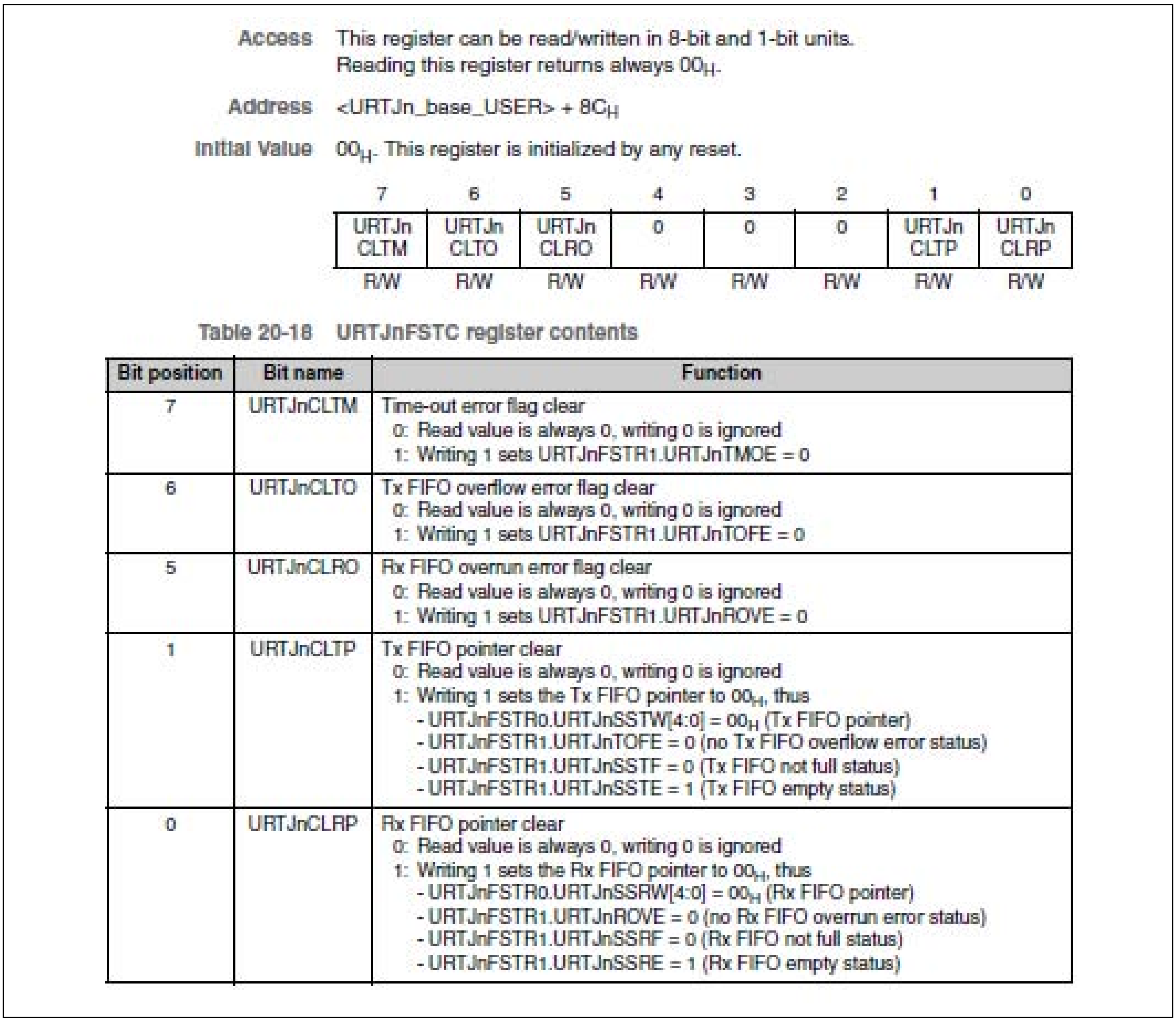
Setting example:

```
URTJnSTC = 0x1f;        /* Clear receive error flags */
                        /* Clear transmit/receive data consistency error flag */
                        /* Clear receive parity error flag */
                        /* Clear receive framing error flag */
                        /* Clear receive overrun error flag */
```

4.2.8 FIFO Status Clear Register (URTJnFSTC)

The error flags in URTJnFSTR1 are set when a timeout error, an overflow error, and an overrun error occur. The URTJnFSTC register can be used to clear the error flags in URTJnFSTR1. The transmit/receive FIFO pointers can also be cleared.

In this sample program, the corresponding error flags are cleared by using the URTJnSTC and URTJnFSTC registers when INTUAJnTIS is detected.



Access This register can be read/written in 8-bit and 1-bit units. Reading this register returns always 00_H.

Address <URTJn_base_USER> + $8C_H$

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
URTJn CLTM	URTJn CLTO	URTJn CLRO	0	0	0	URTJn CLTP	URTJn CLRP
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-18 URTJnFSTC register contents

Bit position	Bit name	Function
7	URTJnCLTM	Time-out error flag clear 0: Read value is always 0, writing 0 is ignored 1: Writing 1 sets URTJnFSTR1.URTJnTMOE = 0
6	URTJnCLTO	Tx FIFO overflow error flag clear 0: Read value is always 0, writing 0 is ignored 1: Writing 1 sets URTJnFSTR1.URTJnTOFE = 0
5	URTJnCLRO	Rx FIFO overrun error flag clear 0: Read value is always 0, writing 0 is ignored 1: Writing 1 sets URTJnFSTR1.URTJnROVE = 0
1	URTJnCLTP	Tx FIFO pointer clear 0: Read value is always 0, writing 0 is ignored 1: Writing 1 sets the Tx FIFO pointer to 00_H, thus - URTJnFSTR0.URTJnSSTW[4:0] = 00_H (Tx FIFO pointer) - URTJnFSTR1.URTJnTOFE = 0 (no Tx FIFO overflow error status) - URTJnFSTR1.URTJnSSTF = 0 (Tx FIFO not full status) - URTJnFSTR1.URTJnSSTE = 1 (Tx FIFO empty status)
0	URTJnCLRP	Rx FIFO pointer clear 0: Read value is always 0, writing 0 is ignored 1: Writing 1 sets the Rx FIFO pointer to 00_H, thus - URTJnFSTR0.URTJnSSRW[4:0] = 00_H (Rx FIFO pointer) - URTJnFSTR1.URTJnROVE = 0 (no Rx FIFO overrun error status) - URTJnFSTR1.URTJnSSRF = 0 (Rx FIFO not full status) - URTJnFSTR1.URTJnSSRE = 1 (Rx FIFO empty status)

Figure 4.15 URTJnFSTC Register Format

Setting example:

```
URTJnFSTC = 0xe3;            /* Clear receive error flags */
                             /* Clear timeout error flag */
                             /* Clear transmit FIFO overflow error flag */
                             /* Clear receive FIFO overrun error flag */
                             /* Clear transmit FIFO pointer */
                             /* Clear receive FIFO pointer */
```

4.3 Function Specifications

This section describes the specifications for the functions that are used by the sample program.

4.3.1 Main Processing (main.c)

[Function Name]	main()
[Function]	Calls necessary initialization functions before entering an infinite loop.
[Arguments]	None
[Return Value]	None
[Startup Method]	Enters the main function after hardware initialization.
[SFRs Used]	None
[Calling Function]	None
[Variables]	flag_transmit_over, flag_receive_over
[File Name]	main.c
[Notes]	None

4.3.2 Software Initialization Processing (initial.c)

[Function Name]	port_initial()
[Function]	Sets up ports and their mode.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	PFCE4, PFC4, PMC4, PM4, PFCE13, PFC13, PMC13, PM13
[Calling Function]	main()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	cg_initial()
[Function]	Initializes the special clock frequency control register.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	SFRCTL3
[Calling Function]	main()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	hbus_initial()
[Function]	Initializes the AHB bus.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	ETARCFG0, ETARADRS0, ETARMASK0
[Calling Function]	main()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	board_initial()
[Function]	Sets up the initial state of the LEDs.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	P13
[Calling Function]	main()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	ram_initial()
[Function]	Sets up the initial states of the receive buffer and flags.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	None
[Calling Function]	main()
[Variables]	txdata_urtj1[], rxdata_urtj3[], flag_transmit_over, flag_receive_over, flag_fifo_error, flag_uart_error
[File Name]	initial.c
[Notes]	None

[Function Name]	wait()
[Function]	Waits for a certain number of steps.
[Arguments]	int number
[Return Value]	None
[Startup Method]	Call according to the an argument setting.
[SFRs Used]	None
[Calling Function]	main(), int_urtj1ire(), int_urtj3ire()
[Variables]	None
[File Name]	initial.c
[Notes]	None

[Function Name]	display()
[Function]	Controls the LEDs according to the state of the relevant flags.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	P13
[Calling Function]	int_urtj3ir(),int_urtj3ire(), int_urtj1ire(),int_urtj1ic()
[Variables]	flag_transmit_over, flag_receive_over, flag_ error
[File Name]	initial.c
[Notes]	None

4.3.3 Receive Processing (uartj_receive.c)

[Function Name]	urtj3_receive_initial()
[Function]	Initializes the UARTJ3 macro.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	URTJ3CTL0, URTJ3CTL1, URTJ3CTL2, URTJ3FCTL0, URTJ3FCTL1, ICURTJ3IR, ICURTJ3IS
[Calling Function]	main()
[Variables]	None
[File Name]	uartj_receive.c
[Notes]	None

4.3.4 Transmit Processing (uartj_transmit.c)

[Function Name]	urtj1_transmit_initial()
[Function]	Initializes the UARTJ1 macro.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	URTJ1CTL0, URTJ1CTL1, URTJ1CTL2, URTJ1FCTL0, URTJ1FCTL1, ICURTJIC, ICURTJ1IS
[Calling Function]	main()
[Variables]	None
[File Name]	uartj_transmit.c
[Notes]	None

[Function Name]	urtj1_transmit_start()
[Function]	Performs the UARTJ1 macro transmission.
[Arguments]	None
[Return Value]	None
[Startup Method]	Call
[SFRs Used]	URTJ1FTX
[Calling Function]	main()
[Variables]	txdata_urtj1[]
[File Name]	uartj_transmit.c
[Notes]	None

4.3.5 Interrupt Processing (interrupt.c)

[Function Name]	int_urtj1ic ()
[Function]	Processes the UARTJ1 macro transmit interrupt.
[Arguments]	None
[Return Value]	None
[Startup Method]	Request INTUAJ1IC is present in an unmasked state.
[SFRs Used]	None
[Calling Function]	None
[Variables]	flag_transmit_over
[File Name]	interrupt.c
[Notes]	None

[Function Name]	int_urtj1ire ()
[Function]	Processes the UARTJ1 macro status interrupt.
[Arguments]	None
[Return Value]	None
[Startup Method]	Request INTUAJ1IS is present in an unmasked state.
[SFRs Used]	URTJ1STR1, URTJ1FSTR1, URTJ1STC, URTJ1FSTC, URTJ1CTL0, URTJ3FSTR0, URTJ3STC, URTJ3FSTC, URTJ3CTL0, ICURTJ1IC, ICURTJ1IS, ICURTJ3IR, ICURTJ3IS, P13
[Calling Function]	None
[Variables]	flag_uart_error, flag_fifo_error, flag_receive_over, flag_transmit_over, rxdata_urtj3[]
[File Name]	interrupt.c
[Notes]	None

[Function Name]	int_urtj3ir ()
[Function]	Processes the UARTJ3 macro receive interrupt.
[Arguments]	None
[Return Value]	None
[Startup Method]	Request INTUAJ3IR is present in an unmasked state.
[SFRs Used]	URTJ3FSTR0, URTJ3FRX
[Calling Function]	None
[Variables]	rxdata_urtj3[], flag_receive_over
[File Name]	interrupt.c
[Notes]	None

[Function Name]	int_urtj3ire ()
[Function]	Processes the UARTJ3 macro status interrupt.
[Arguments]	None
[Return Value]	None
[Startup Method]	Request INTAJ3IS is present in an unmasked state.
[SFRs Used]	URTJ3STR1, URTJ3FSTR1, URTJ1STC, URTJ1FSTC, URTJ1CTL0, URTJ3FSTR0, URTJ3STC, URTJ3FSTC, URTJ3CTL0, ICURTJ1IC, ICURTJ1IS, ICURTJ3IR, ICURTJ3IS, P13
[Calling Function]	None
[Variables]	flag_uart_error, flag_fifo_error, flag_receive_over, flag_transmit_over, rxdata_urtj3[]
[File Name]	interrupt.c
[Notes]	None

Website and Support

Renesas Electronics Website

http://www.renesas.com/

Inquiries

http://www.renesas.com/inquiry

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Feb 07, 2012	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins
   Handle unused pins in accord 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 one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

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

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

RENESAS

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

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

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

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

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

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

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

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

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

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

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

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

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.1