RX Family, M16C Family

Migrating from the M16C Family to the RX Family: Simple I²C Mode

R01AN1928EJ0100
Rev. 1.00
Aug. 18, 2014

Abstract

This document describes migrating from special mode 1 (I²C mode) for the serial I/O in the M16C Family to the simple I²C mode for the SCI in the RX Family.

Products

RX Family, M16C Family

As an example of migrating from the M16C to the RX, the explanation in this document uses the RX210 Group in the RX Family and the M16C/65C Group in the M16C Family. When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Differences in Terminology

Item	RX Family	M16C Family
Serial communication interface (SCI)	SCI	Serial I/O
Simple I²C mode	Simple I²C mode	Special mode 1 (I²C mode)
SDA pin	SSDAi pin	SDAi pin
SCL pin	SSCLi pin	SCLi pin
SCI operating clock (clock source)	Clock source	Count source
Peripheral function operating clock	Peripheral module clocks: PCLKA, PCLKB, PCLKC, PCLKD	Peripheral function clocks: f1, fOCO40M, fOCO-F, fOCO-S, fC32
Transmit buffer	TDR register	UiTB register
Transmit shift register	TSR register	UART transmit shift register
Receive buffer	RDR register	UiRB register
Interrupt after generating the start condition and stop condition	STI interrupt	Start condition/stop condition detection interrupt *1
Transmit interrupt	TXI interrupt	UARTi transmit interrupt (transmit buffer empty)
Receive interrupt	RXI interrupt	UARTi receive interrupt
Function to select I/O of peripheral functions for pins	MPC *2	Function select register, input function select register *3

Note 1. In the M16C Family, an interrupt request occurs when a start condition, restart condition, or stop condition is generated.
Note 2. The MPC is not available in some groups.
Note 3. Only available in the M32C Group and R32C Group.

Contents

1. Differences in Simple I2C Mode

Table 1.1 lists the Differences in I2C Mode.

Table 1.1 Differences in I2C Mode

Item	RX (RX210)	M16C (M16C/65C)
Operating mode	Master *1	Master, slave
Data length	8 bits	8 bits
Data format	MSB first fixed	MSB first fixed
Interrupt sources	Transmit, NACK (TXI) interrupt Receive, ACK detection (RXI) interrupt Completion of generating a start, restart, or stop condition interrupt (STI interrupt)	Transmit, NACK (TXI) interrupt Receive, ACK detection (RXI) interrupt Start condition, stop condition detection interrupt
Error detection	None	Overrun error, arbitration lost
Noise cancellation	The signal paths from input on the SSCLn and SSDAn pins incorporate digital noise filters	Not available
SSDAn digital delay	0 to 31 cycle delay of the clock signal selectable	No digital delay, or delay of 2 to 8 cycles of UiBRG count source selectable
Clock phase setting	Clock delay available	Clock delay available or no clock delay selectable

Note 1. Use the RIIC when performing slave operation with the RX Family.

2. Peripheral Functions Used

Table 2.1 lists Peripheral Functions and Modes Used When Using Simple I2C Mode.

Table 2.1 Peripheral Functions and Modes Used When Using Simple I2C Mode

No.	Operating Example	RX		M16C		Reference
		Peripheral Function	Mode	Peripheral Function	Mode	
1	Master transmit operation in simple I2C mode	SCI	Simple I2C mode	Serial I/O	Special mode 1 (I2C mode)	Section 3.1
2	Master receive operation in simple I2C mode					Section 3.2

3. Differences in Simple I2C Mode

This section describes the differences in simple I2C mode with an example using the conditions listed in Table 3.1 Conditions for Simple I2C Communication.

Table 3.1 Conditions for Simple I2C Communication

Item	Conditions for Transmission and Reception
Peripheral function operating clock	16 MHz
Transfer rate	100 kbps
Channels used	RX Family: SCI0 M16C Family: UART0

3.1 Differences in Transmission

This section describes the differences in master transmission for simple I²C mode.

3.1.1 Differences in Timing During Transmission

Figure 3.1 shows the Differences in Timing When Transmitting 3 Bytes at a Time.

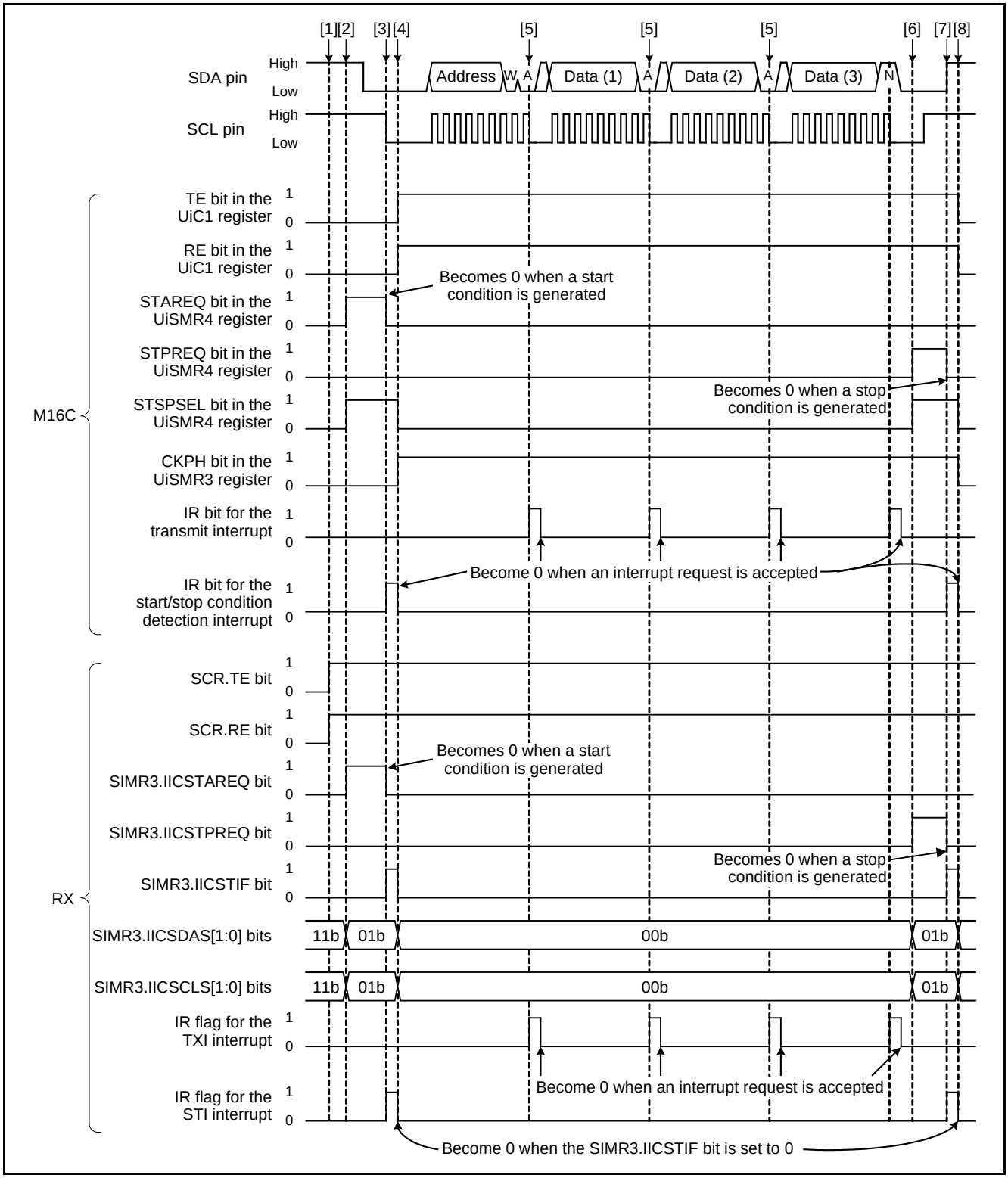

Figure 3.1 Differences in Timing When Transmitting 3 Bytes at a Time

Table 3.2 lists Differences in Operation at Various Timings and Processing During Master Transmission.

Table 3.2 Differences in Operation at Various Timings and Processing During Master Transmission

Timing		RX (RX210)	M16C (M16C/65C)
[1]	Before transmission starts	The SCR.TE bit is set to 1 (serial transmission is enabled), the RE bit is set to 1 (serial reception is enabled), the TIE bit is set to 1 (TXI interrupt request is enabled), and the TEIE bit is set to 1 (STI interrupt request is enabled).	Bits ILVL2 to ILVL0 in the UiBCNIC register are set and the start condition/stop condition interrupt is enabled.
[2]	Start condition output	At the same time the SIMR3.IICSTAREQ bit is set to 1 (start condition is generated), the IICSCLS[1:0] bits and the IICSDAS[1:0] bits are set to 01b (generate a start, restart, or stop condition).	After the STAREQ bit is set to 1, the STSPSEL bit is set to 1 (select start condition/stop condition generate circuit).
[3]	Start condition detection interrupt generation	The IICSTAREQ bit is set to 0 (start condition is not generated), the IICSTIF bit is set to 1 (all request generation has been completed), and the IR flag for the STI interrupt becomes 1.	The STAREQ bit becomes 0, and the IR bit for the start condition/stop condition detection interrupt becomes 1.
[4]	Start condition detection interrupt handling	The IICSTIF bit is set to 0, the IICSCLS[1:0] bits are set to 00b (serial clock output), and the IICSDAS[1:0] bits are set to 00b (serial data output). When the IICSTIF bit is set to 0, the IR flag for the STI interrupt becomes 0. Then, the first byte of transmit data (slave address and R/W bit) is written to the transmit buffer.	The CKPH bit is set to 1 (with clock delay), the TE bit is set to 1 (transmission enabled), the RE bit is set to 1 (reception enabled), and the STSPSEL bit is set to 0 (select serial I/O circuit). In addition, bits ILVL2 to ILVL0 in the SiTIC register are set and the transmit interrupt is enabled. Then, the first byte of transmit data (slave address and W bit) is written to the transmit buffer.
[5]	When transmission is complete	The IR flag (IR bit) for the transmit interrupt (TXI interrupt) becomes 1, and the transmit interrupt is generated. In the transmit interrupt handling, after ACK or NACK is confirmed, the second byte and successive data is written to the transmit buffer.	
[6]	Transmit interrupt handling after last data is output	At the same time the IICSTPREQ bit is set to 1 (stop condition is generated), the IICSCLS[1:0] bits and IICSDAS[1:0] bits are set to 01b.	After the STPREQ bit is set to 1, the STSPSEL bit is set to 1.
[7]	Stop condition detection interrupt generation	The IICSTPREQ bit becomes 0 (stop condition is not generated), the IICSTIF bit becomes 1, and the IR flag for the STI interrupt becomes 1.	The STPREQ bit becomes 0, and the IR bit for the start condition/stop condition detection interrupt becomes 1.
[8]	Stop condition detection interrupt handling	The IICSTIF bit is set to 0, and bits IICSCLS[1:0] and IICSDAS[1:0] are set to 11b (high-impedance). When the IICSTIF bit is set to 0, the IR flag for the STI interrupt becomes 0.	The TE bit is set to 0 (transmission disabled), the RE bit is set to 0 (reception disabled), the CKPH bit is set to 0 (no clock delay), the STSPSEL bit is set to 0, and bits SMD2 to SMD0 in the UiMR register are set to 000b.

3.1.2 Differences in Setting Procedures During Master Transmission

The tables in this section list the differences during transmission.

Table 3.3 Differences in the Initial Setting Procedures During Master Transmission

Step		RX (RX210)	M16C (M16C/65C)
1	Exiting the module-stop state *1	SYSTEM.PRCR.WORD = 0xA502; MSTP(SCI0) = 0; SYSTEM.PRCR.WORD = 0xA500;	N/A (no module-stop function)

2	Setting the I/O port functions *2	PORT2.ODR0.BYTE = 0x05;
		PORT2.PMR.BIT.B0 = 0;
		PORT2.PMR.BIT.B1 = 0;
		MPC.PWPR.BIT.B0WI = 0;
		MPC.PWPR.BIT.PFSWE = 1;
		MPC.P20PFS.BYTE = 0x0A;
		MPC.P21PFS.BYTE = 0x0A;
		MPC.PWPR.BYTE = 0x80;
		PORT2.PMR.BIT.B0 = 1;
		PORT2.PMR.BIT.B1 = 1;
			nch_u0c0 = 1;
			p6 = 0x0C;
			pd6 = 0x00;
			See Note 3

3	Setting the transmit/receive mode, etc.	SCI0.SIMR3.BYTE = 0xF0;
		SCI0.SMR.BYTE = 0x00;
		SCI0.SCMR.BYTE = 0x08;
			uclksel0 = 0x00;
			prc0 = 1;
			pclk1 = 1;
			prc0 = 0;
			u0smr = 0x01;
			u0mr = 0x02;
			u0smr2 = 0x03;
			u0smr3 = 0xE0;
			u0smr4 = 0x70;
			u0c0 = 0xB0;

4	Setting the bit rate *4	SCI0.BRR = 4;	u0brg = 79;
5	Setting the transmit/receive mode, etc.	SCI0.SEMR.BYTE = 0x00; SCI0.SNFR.BYTE = 0x00; SCI0.SIMR1.BYTE = 0x51; SCI0.SIMR2.BYTE = 0x23; SCI0.SPMR.BYTE = 0x00;	ucon = 0x01;

6	Setting the interrupt priority level	IPR(SCI0,) =0x01;	ifsr26 = 1;
			u0bcnic = 0x00;
			s0tic = 0x00;

7	Clear the interrupt requests	IR(SCI0,RXI0) = 0; IR(SCI0,TXI0) = 0;	
8	Enable peripheral function interrupt requests *5	SCI0.SCR.BYTE |= 0xB4; /* Note 6 */ IEN(SCI0,TEI0) = 1; IEN(SCI0,TXI0) = 1;	N/A
9	Enable transmission/reception		N/A *7
10	Enable interrupt requests *5		N/A (no processing)
11	Enable maskable interrupts	setpsw_i();	asm("fset i");

Note 1. For details on the module stop function, refer to section 4.1.3 Module Stop Function.

Note 2. In the RX Family, pin settings for peripheral functions are configured in the MPC. Refer to section 4.1.2 I/O ports for details.

Note 3. In the M32C/80 Series and R32C/100 Series, select the pin function in the function select register.

Note 4. The bit rate calculation method differs between the RX Family and M16C Family. Refer to section 3.3 Calculating the Bit Rate for details.

Note 5. The method of enabling interrupt requests differs. Refer to section 4.1.1 Interrupts for details.

Note 6. Set the SCR.TE bit to 1 (serial transmission is enabled) and the RE bit to 1 (serial reception is enabled) simultaneously.

Note 7. In the M16C Family, transmission is enabled after a start condition occurs.

Table 3.4 Differences in the Start Condition Output Processing During Master Transmission

Step		RX (RX210)	M16C (M16C/65C)
1	Check to see if the bus is free	N/A *1	if (bbs_u0smr ==0) {

2	Initial setting prior to start condition output *2	N/A	u0bcnic = 0x01:
			u0smr4 = 0x70;
			u0mr = 0x02;
			u0brg = 0;
			u0smr2 = 0x03;
			u0brg = 79;

3	Start condition output	SCI0.SIMR3.BYTE = 0xF0;
		SCI0.SIMR3.BYTE = 0x51;
			u0smr4 = 0x71;
			asm(" "); /* Optimization measure */
			u0smr4 = 0x09;
			}

Note 1. The RX Family does not have the bus busy flag. Use software when necessary.
Note 2. In the M16C Family, the countermeasure in the TN-16C-A130B/E technical update is used.

Table 3.5 Differences in the Start/Stop Condition Generation Interrupt During Master Transmission

Step		RX (RX210)	M16C (M16C/65C)

1	Check the source of the interrupt request	do
		{
		while ((0 != SCI0.SIMR3.BIT.IICSTIF))
		{
			N/A

2	Confirm start condition generation	if(Was the interrupt the one generated when the start condition was generated?) {	if (bbs_u0smr ==1) {
3	Settings prior to starting transmission	N/A	u0smr3 = 0xE2; u0c1 = 0x05;
4	Stop start condition output, output serial data	SCI0.SIMR3.BYTE = 0x00;	u0smr4 = 0x00;
5	Set the arbitration lost detection flag to 0	N/A	u0rb = 0x0000;
6	Transmit the slave address and R/W bit	/* Write the slave address and R/W bit to the SCI0.TDR register */	/* Write the slave address and R/W bit to the U0TB register */
7	Set the IR flag to 0	N/A	u0bcnic = 0x01;
8	Enable transmit interrupts	N/A	s0tic = (s0tic & 0x08) |0x01;

9	Confirm stop condition generation	}
		else if (Was the interrupt the one generated when the stop condition was generated?)
		{
			}
			else
			{

10	Settings after transmission is complete	N/A	u0c1 = 0x00;
			u0mr = 0x00;
			u0smr3 = 0xE0;

11	Stop condition output, leave pins open	SCI0.SIMR3.BYTE = 0xF0; }	u0smr4 = 0x70; u0mr = 0x02;

12	Disable interrupts	N/A	s0tic = 0x00;
			u0bcnic = 0x00;
			}

13	Read the IR flag	} }while(0 != IR(SCI0, TEI0));	N/A

Table 3.6 Differences in Transmit Interrupt Handling During Master Transmission

Step		RX (RX210)	M16C (M16C/65C)
1	Confirm stop condition output	if (Has the last data been output or has NACK been received (SISR.IICACKR = 1) ?) {	if (Has the last data been output or has NACK been received (bit 8 of U0RB register = 1) ?) {

2	Output stop condition (when last data has already been output, or when NACK is received)	SCI0.SIMR3.BYTE = 0x00;
		SCI0.SIMR3.BYTE = 0x54;
		}
			u0smr4 = 0x04;
			asm("");
			u0smr4 = 0x3C;
			}

3	Write transmit data (before last data is output, and when ACK is received)	else
		{
		/* Write transmit data to the SCI0.TDR register */;
		}
			else
			{
			/* Write transmit data to the U0TB register (set bit 8 to 1) */;
			}

3.2 Differences in Master Reception

This section describes the differences in master reception for simple I²C mode.

3.2.1 Differences in Timing During Reception

Figure 3.2 shows Differences in Timing When Receiving 3 Bytes at a Time.

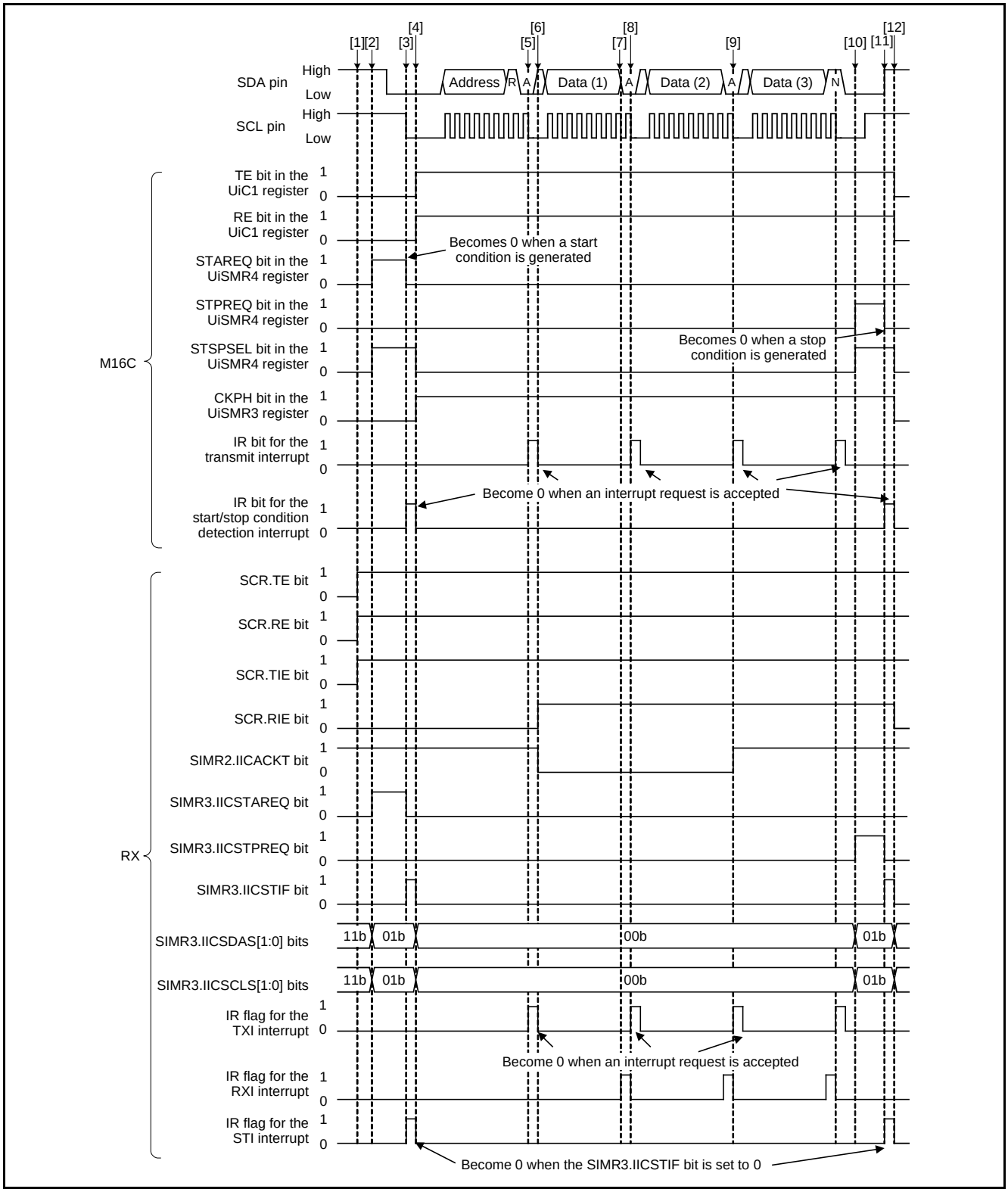

Figure 3.2 Differences in Timing When Receiving 3 Bytes at a Time

Table 3.7 Differences in Operation at Various Timing and Processing During Master Reception (1/2)

Timing		RX (RX210)	M16C (M16C/65C)
[1]	Before transmission starts	The SCR.TE bit is set to 1 (serial transmission is enabled), the RE bit is set to 1 (serial reception is enabled), the TIE bit is set to 1 (TXI interrupt request is enabled), and the TEIE bit is set to 1 (STI interrupt request is enabled).	Bits ILVL2 to ILVL0 in the UiBCNIC register are set and the start condition/stop condition interrupt is enabled.
[2]	Start condition output	At the same time the SIMR3.IICSTAREQ bit is set to 1 (start condition is generated), the IICSCLS[1:0] bits and the IICSDAS[1:0] bits are set to 01b (generate a start, restart, or stop condition).	After the STAREQ bit is set to 1, the STSPSEL bit is set to 1 (select start condition/stop condition generate circuit).
[3]	Start condition output detection interrupt generation	The IICSTAREQ bit is set to 0 (start condition is not generated), the IICSTIF bit is set to 1 (all request generation has been completed), and the IR flag for the STI interrupt becomes 1.	The STAREQ bit becomes 0, and the IR bit for the start condition/stop condition detection interrupt becomes 1.
[4]	Start condition detection interrupt handling	The IICSTIF bit is set to 0, the IICSCLS[1:0] bits are set to 00b (serial clock output), and the IICSDAS[1:0] bits are set to 00b (serial data output). When the IICSTIF bit is set to 0, the IR flag for the STI interrupt becomes 0. Then, the first byte of transmit data (slave address and R/W bit) is written to the transmit buffer.	The CKPH bit is set to 1 (with clock delay), the TE bit is set to 1 (transmission enabled), the RE bit is set to 1 (reception enabled), and the STSPSEL bit is set to 0 (select serial I/O circuit). Then, the first byte of transmit data (slave address and W bit) is written to the transmit buffer.
[5]	When address transmission is complete	The IR flag (IR bit) for the transmit interrupt (TXI interrupt) becomes 1, and the transmit interrupt is generated.	
[6]	Transmit interrupt handling after address is transmitted	The SCR.RIE bit is set to 1 (RXI interrupt request is enabled), the SIMR2.IICACKT bit is set to 0 (ACK transmission), and dummy data (0xFF) is written to the transmit buffer.	0x00FF is written as dummy data to the transmit buffer.
[7]	Data reception	The IR flag for the RXI interrupt becomes 1. Data received is read in the RXI interrupt handling.	N/A
[8]	ACK transmission complete	The IR flag for the TXI interrupt becomes 1. Dummy data (0xFF) is written to the transmit buffer in the TXI interrupt handling.	The IR bit for the transmit interrupt becomes 1. After the transmit interrupt is used to read the receive data, dummy data (0x00FF) is written to the transmit buffer.
[9]	Transmit complete interrupt handling prior to last data being written	The IICACKT bit is set to 1 (NACK transmission), and dummy data (0xFF) is written to the transmit buffer.	After reading the data received, 0x01FF is written to the transmit buffer as dummy data.
[10]	Transmit interrupt handling after the last data is output	At the same time the IICSTPREQ bit is set to 1 (stop condition is generated), the IICSCLS[1:0] bits and the IICSDAS[1:0] bits are set to 01b (generate a start, restart, or stop condition).	After the STPREQ bit is set to 1, the STSPSEL bit is set to 1.
[11]	Stop condition detection interrupt generation	The IICSTPREQ bit becomes 0 (stop condition is not generated), the IICSTIF bit becomes 1, and the IR flag for the STI interrupt becomes 1.	The STPREQ bit becomes 0, and the IR bit for the start condition/stop condition detection interrupt becomes 1.
[12]	Stop condition detection interrupt handling	The SCR.RIE bit is set to 0 (RXI interrupt request is disabled), the SIMR3.IICSTIF bit is set to 0, and bits IICSCLS[1:0] and IICSDAS[1:0] are set to 11b (high-impedance). When the IICSTIF bit is set to 0, the IR flag for the STI interrupt becomes 0.	The TE bit is set to 0 (transmission disabled), the RE bit is set to 0 (reception disabled), the CKPH bit is set to 0 (no clock delay), the STSPSEL bit is set to 0, and bits SMD2 to SMD0 in the UiMR register are set to 000b.

3.2.2 Differences in Setting Procedures During Master Reception

Table 3.8 lists the Differences in the Initial Setting Procedures When Transmitting Data, Table 3.9 lists the Differences in Transmit Interrupt Handling When Transmitting Data, and Table 3.10 lists the Differences in Transmit End Interrupt Handling When Transmitting Data.

Table 3.8 Differences in the Initial Setting Procedures for Transmission and Reception During Master Reception

Step		RX (RX210)	M16C (M16C/65C)
1	Exiting the module-stop state *1	SYSTEM.PRCR.WORD = 0xA502; MSTP(SCI0) = 0; SYSTEM.PRCR.WORD = 0xA500;	N/A (no module-stop function)

2	Setting the I/O port functions *2	PORT2.ODR0.BYTE = 0x05;
		PORT2.PMR.BIT.B0 = 0;
		PORT2.PMR.BIT.B1 = 0;
		MPC.PWPR.BIT.B0WI = 0;
		MPC.PWPR.BIT.PFSWE = 1;
		MPC.P20PFS.BYTE = 0x0A;
		MPC.P21PFS.BYTE = 0x0A;
		MPC.PWPR.BYTE = 0x80;
		PORT2.PMR.BIT.B0 = 1;
		PORT2.PMR.BIT.B1 = 1;
			nch_u0c0 = 1; /* See Note 1 */
			p6 = 0x0C;
			pd6 = 0x00;

3	Setting the transmit/receive mode, etc.	SCI0.SIMR3.BYTE = 0xF0;
		SCI0.SMR.BYTE = 0x00;
		SCI0.SCMR.BYTE = 0x08;
			uclksel0 = 0x00;
			prc0 = 1;
			pclk1 = 1;
			prc0 = 0;
			u0smr = 0x01;
			u0mr = 0x02;
			u0smr2 = 0x03;
			u0smr3 = 0xE0;
			u0smr4 = 0x70;
			u0c0 = 0xB0;

4	Setting the bit rate *4	SCI0.BRR = 4;	u0brg = 79;
5	Setting the transmit/receive mode, etc.	SCI0.SEMR.BYTE = 0x00; SCI0.SNFR.BYTE = 0x00; SCI0.SIMR1.BYTE = 0x51; SCI0.SIMR2.BYTE = 0x23; SCI0.SPMR.BYTE = 0x00;	ucon = 0x01;

6	Setting the interrupt priority level	IPR(SCI0,) =0x01	ifsr26 = 1;
			u0bcnic = 0x00;
			s0tic = 0x00;

7	Clear the interrupt requests	IR(SCI0,RXI0) = 0; IR(SCI0,TXI0) = 0;	
8	Enable peripheral function interrupt requests *5	SCI0.SCR.BYTE |= 0xF4; /* Note 6 */ IEN(SCI0,RXI0) = 1; IEN(SCI0,TEI0) = 1; IEN(SCI0,TXI0) = 1;	
9	Enable transmission/reception		N/A *7
10	Enable interrupt requests *5		N/A (no processing)
11	Enable maskable interrupts	setpsw_i();	asm("fset i");

Note 1. For details on the module stop function, refer to section 4.1.3 Module Stop Function.

Note 2. In the RX Family, pin settings for peripheral functions are configured in the MPC. Refer to section 4.1.2 I/O ports for details.

Note 3. In the M32C/80 Series and R32C/100 Series, select the pin function in the function select register.

Note 4. The bit rate calculation method differs between the RX Family and M16C Family. Refer to section 3.3 Calculating the Bit Rate for details.

Note 5. The method of enabling interrupt requests differs. Refer to section 4.1.1 Interrupts for details.

Note 6. Set the SCR.TE bit to 1 (serial transmission is enabled) and the RE bit to 1 (serial reception is enabled) simultaneously.

Note 7. In the M16C Family, transmission and reception are enabled after a start condition is generated.

Table 3.9 Differences in Start Condition Output Processing During Master Reception

Step		RX (RX210)	M16C (M16C/65C)
1	Check to see if the bus is free	N/A *1	if (bbs_u0smr ==0) {

2	Initial setting prior to start condition output *2	N/A	u0bcnic = 0x01:
			u0smr4 = 0x70;
			u0mr = 0x02;
			u0brg = 0;
			u0smr2 = 0x03;
			u0brg = 79;

3	Start condition output	SCI0.SIMR3.BYTE = 0xF0;
		SCI0.SIMR3.BYTE = 0x51;
			u0smr4 = 0x71;
			asm(" "); /* Optimization measure */
			u0smr4 = 0x09;
			}

Note 1. The RX Family does not have the bus busy flag. Use software when necessary.
Note 2. In the M16C Family, the countermeasure in the TN-16C-A130B/E technical update is used.

Table 3.10 Differences in the Start/Stop Condition Generation Interrupt During Master Reception

Step		RX (RX210)	M16C (M16C/65C)

1	Check the source of the interrupt request	do
		{
		while ((0 != SCI0.SIMR3.BIT.IICSTIF))
		{
			N/A

2	Confirm start condition generation	if(Was the interrupt the one generated when the start condition was generated?) {	if (bbs_u0smr ==1) {
3	Settings prior to starting transmission	N/A	u0smr3 = 0xE2; u0c1 = 0x05;
4	Stop start condition output, output serial data	SCI0.SIMR3.BYTE = 0x00;	u0smr4 = 0x00;
5	Set the arbitration lost detection flag to 0	N/A	u0rb = 0x0000;
6	Transmit the slave address and R/W bit	/* Write the slave address and R/W bit to the SCI0.TDR register */	/* Write the slave address and R/W bit to the U0TB register */
7	Set the IR flag to 0	N/A	u0bcnic = 0x01;
8	Enable transmit interrupts	N/A	s0tic = (s0tic & 0x08) |0x01;

9	Confirm stop condition generation	}
		else if (Was the interrupt the one generated when the stop condition was generated?)
		{
			}
			else
			{

10	Settings after transmission is complete	N/A	u0c1 = 0x00;
			u0mr = 0x00;
			u0smr3 = 0xE0;

11	Stop condition output, leave pins open	SCI0.SIMR3.BYTE = 0xF0; }	u0smr4 = 0x70; u0mr = 0x02;

12	Disable interrupts	SCI0.SCR.BIT.RIE = 0;	s0tic = 0x00;
			u0bcnic = 0x00;
			}

13	Read the IR flag	} }while(0 != IR(SCI0, TEI0));	N/A

Table 3.11 Differences in Transmit Interrupt Handling During Master Reception

Step		RX (RX210)	M16C (M16C/65C)

1	Read the data received	N/A *1	if (When the data is received?)
			{
			/* Read data from the U0RB register */;
			}

2	Confirm stop condition output	if (Has the last data been output or has NACK been received (SISR.IICACKR = 1) ?) {	if (Has the last data been output or has NACK been received (bit 8 of U0RB register = 1) ?) {

3	Output stop condition (when last data has already been output, or when NACK is received)	SCI0.SIMR3.BYTE = 0x00;
		SCI0.SIMR3.BYTE = 0x54;
		}
			u0smr4 = 0x04;
			asm("");
			u0smr4 = 0x3C;
			}

4	Write transmit data (before last data is output, and when ACK is received)	else
		{
		if (When the slave address is transmitted?)
		{
		SCI0.SIMR2.BIT.IICACKT = 0;
		SCI0.SCR.BIT.RIE = 1;
		}
		if (Is the next data the last data?)
		{
		/* NACK output setting */
		SCI0.SIMR2.BIT.IICACKT = 1;
		}
			N/A

5	Set ACK/NACK transmission (prior to receiving the last data)		else
			{
			if (Is the next data the last data?)
			{
			/* NACK output setting */
			u0tb = 0x01FF;
			}
			else
			{
			/* ACK output setting */
			u0tb = 0x00FF;
			}
			}

6	Write transmit data (dummy data) (prior to receiving the last data)	SCI0.TDR = 0xFF; }	

Note 1. In the RX Family, received data is read by the RXI interrupt.

Table 3.12 Differences in the Receive Interrupt Handling During Master Reception

Step		RX (RX210)	M16C (M16C/65C)
1	Read the receive buffer	/* Read the data in the SCI0.RDR register */	N/A *1

Note 1. In the M16C Family, the receive buffer is read by the transmit complete interrupt.

3.3 Calculating the Bit Rate

There are differences in calculating the bit rate between the RX Family and M16C Family.

Table 3.13 Differences in Calculating the Bit Rate

Item	RX (RX210)	M16C (M16C/65C)
Calculating the bit rate using the internal clock	Clock source ÷ 32(*N* + 1) *1 where the clock source is PCLK, PCLK/4, PCLK/16, or PCLK/64, and *N* is the BRR register setting value	Clock source ÷ 2(*n* + 1) where the clock source is f1SIO, f2SIO, f8SIO, or f32SIO, and *n* is the UiBRG setting value

Note 1. Based on the "Relationships between N Setting in BRR and Bit Rate B" in the UMH:

$B = PCLK \div (64 \times 2^{2n-1} \times (N + 1))$

$= PCLK \div (32 \times 2^{2n} \times (N + 1))$

$= (PCLK \div 2^{2n}) \div (32 \times (N + 1))$

$= \text{Clock source} \div (32 \times (N + 1))$

4. Appendix

4.1 Points on Migrating from the M16C to the RX

This chapter explains points on migrating from the M16C to the RX.

4.1.1 Interrupts

In the RX Family, if an interrupt request is received when all of the following conditions are met, the interrupt can be accepted.

- The I flag (PSW.I bit) is 1.
- Registers IER and IPR in the ICU are set to "interrupt enabled".
- The interrupt request is enabled by the interrupt request enable bits for the peripheral function.

Table 4.1 lists the Comparison of Conditions for Generating an Interrupt.

Table 4.1 Comparison of Conditions for Generating an Interrupt

Item	RX	M16C
I flag	When the I flag is set to 1 (enabled), the maskable interrupts are accepted.	
Interrupt request flag	When there is an interrupt request from a peripheral function, the interrupt request flag becomes 1 (interrupt requested).	
Interrupt priority level	Selected by setting the IPR[3:0] bits.	Selected by setting bits ILVL2 to ILVL0.
Interrupt request enable	Specified by setting the IER register.	N/A
Interrupt enable for peripheral functions	Interrupt enable or disable can be specified in each peripheral function.	N/A

For more information, refer to sections Interrupt Controller (ICU), CPU, and sections for other peripheral functions used in the UMH.

4.1.2 I/O ports

In the RX Family, the MPC must be configured in order to assign I/O signals of peripheral functions to pins. Before controlling the input and output pins in the RX Family, the following two items must be set.

- In the MPC.PFS register, select the peripheral functions that are assigned to the appropriate pins.
- In the PORT.PMR register, select the function for the pin to be used as a general I/O port or I/O port for a peripheral function.

Table 4.2 lists Comparison of I/O Settings for Peripheral Function Pins.

Table 4.2 Comparison of I/O Settings for Peripheral Function Pins

Function	RX (RX210)	M16C (M16C/65C)
Select the pin function	A pin to which a peripheral I/O is assigned can be selected from multiple pins. The PFS register is used to select a peripheral function I/O that is assigned to a pin used.	These are not available in the M16C Group. *1 When a mode of a peripheral function is selected, an appropriate pin is assigned as an I/O pin for the peripheral function.
Switch between general I/O port and peripheral function	With the PMR register, the pin function can be selected as a general I/O port or a peripheral function.	

Note 1. The M32C Group and R32C Group have registers to assign a peripheral function to a pin.

For more information, refer to the Multi-Function Pin Controller (MPC) and I/O port sections in the UMH.

4.1.3 Module Stop Function

The RX Family has the ability to stop each peripheral module individually. By placing unused peripheral modules in the module stop state, power consumption can be reduced. After a reset is released, all modules (with a few exceptions) are in the module stop state. Registers for modules in the module stop state cannot be read or written.

For more information, refer to the Low Power Consumption section in the UMH.

4.2 I/O Register Macros

Macro definitions listed in Table 4.3 can be found in the RX I/O register definitions (iodefine.h).

The readability of programs can be achieved with these macro definitions.

Table 4.3 lists Examples of Macros.

Table 4.3 Macros Used

Macro	Usage Example
IR("module name", "bit name")	**IR(MTU0,TGIA0)** = 0 ; The IR bit corresponding to MTU0.TGIA0 is set to 0 (no interrupt request is generated).
IEN("module name", "bit name")	**IEN(MTU0, TGIA0)** = 1 ; The IEN bit corresponding to MTU0.TGIA0 is set to 1 (interrupt request enabled).
IPR("module name", "bit name")	**IPR(MTU0, TGIA0)** = 0x02 ; The IPR[3:0] bits corresponding to MTU0.TGIA0 are set to 0010b (interrupt priority level 2).
MSTP("module name")	**MSTP(MTU)** = 0 ; The MTU0 module stop bit is set to 0 (module stop state is canceled).
VECT("module name", "bit name")	#pragma interrupt (Excep_MTU0_TGIA0 (vect=**VECT(MTU0, TGIA0)**) The interrupt function corresponding to MTU0.TGIA0 is declared.

4.3 Intrinsic Functions

The RX has intrinsic functions for setting control registers and special instructions. When using intrinsic functions, include machine.h.

Table 4.4 lists examples of Descriptions of Special Instructions and Control Register Settings.

Table 4.4 Examples of Differences in the Descriptions of Special Instructions and Control Register Settings

Item	Description	
	RX	M16C
Set the I flag to 1	setpsw_i (); *1	asm("fset i");
Set the I flag to 0	clrpsw_i (); *1	asm("fclr i");
Expanded into the WAIT instruction	wait(); *1	asm("wait");
Expanded into the NOP instruction	nop(); *1	asm("nop");

Note 1. "machine.h" must be included.

5. Reference Documents

User's Manual: Hardware
RX210 Group User's Manual: Hardware Rev.1.50 (R01UH0037EJ)
M16C/65C Group User's Manual: Hardware Rev.1.10 (R01UH0093)
Refer to the corresponding UMH when using products other than the RX210 Group and M16C/65C Group.
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.

User's Manual: Development Tools
RX Family C/C++ Compiler Package V.1.01 User's Manual Rev.1.00 (R20UT0570EJ)
M16C Series, R8C Family C Compiler Package V5.45
C Compiler User's Manual Rev.3.00
The latest versions can be downloaded from the Renesas Electronics website.

Website and Support

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

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

REVISION HISTORY	RX Family, M16C Family Application Note Migrating From the M16C Family to the RX Family: Simple I2C Mode

Rev.	Date	Description	
		Page	Summary
1.00	Aug. 18, 2014	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

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

1. Handling of Unused Pins

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

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

2. Processing at Power-on

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

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

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

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

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

RENESAS

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

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

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, 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: +49-211-6503-0, 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, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022/9044

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

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Ku, Seoul, 135-920, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 4.0