To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

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

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

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

Application Notes

8

38K0/38K2 Group

EXB Application Notes

Renesas Single-Chip Microcomputers

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (http://www.renesas.com).

Renesas Electronics
www.renesas.com

VER.1.1 2003.06

Keep safety first in your circuit designs!

* RENESAS TECHNOLOGY CORPORATION puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of non-flammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

* These materials are intended as a reference to assist our customers in the selection of the RENESAS semiconductor product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to RENESAS TECHNOLOGY CORPORATION or a third party.
* RENESAS TECHNOLOGY CORPORATION assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
* All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by RENESAS TECHNOLOGY CORPORATION without notice due to product improvements or other reasons. It is therefore recommended that customers contact RENESAS TECHNOLOGY CORPORATION or an authorized RENESAS Semiconductor product distributor for the latest product information before purchasing a product listed herein. The information described here may contain technical inaccuracies or typographical errors. RENESAS TECHNOLOGY CORPORATION assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors. Please also pay attention to information published by RENESAS TECHNOLOGY CORPORATION by various means, including the RENESAS Semiconductor home page (http://www.renesas.com/eng/).
* When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. RENESAS TECHNOLOGY CORPORATION assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
* RENESAS TECHNOLOGY CORPORATION semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact RENESAS TECHNOLOGY CORPORATION or an authorized RENESAS Semiconductor product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical aerospace, nuclear, or undersea repeater use.
* The prior written approval of RENESAS TECHNOLOGY CORPORATION is necessary to reprint or reproduce in whole or in part these materials.
* If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination. Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
* Please contact RENESAS TECHNOLOGY CORPORATION or an authorized RENESAS Semiconductor product distributor for further details on these materials or the products contained therein.

Revision History	38K0/38K2 Group EXB Application Notes

Rev.No.	Date	Contents
0.0	2001/10/18	Draft issued
0.9	2001/12/05	First edition issued
1.0	2002/11/26	First edition is revised.
1.1	2003/01/31	Correction of errors

- Related Materials

  *RENESAS Microcomputers 38K0(38K2) Group Data Sheet

  *38K0(38K2) Group USB Application Note

- **RENESAS** MCU Technical Information

  ¨ **http://www.renesas.com/eng/products/mpumcu/specific/usb_mcu/index.html**

- Table of contents -

Chapter 1
EXB Outline

1.1 Introduction

This document provides detailed information concerning the external bus function “EXB” for the RENESAS Microprocessor 38K0/38K2 Group. For information concerning USB functions please refer to the 38K0 (38K2) Group USB Application Notes.

1.2 What is “EXB?”

38K0/38K2 features an external bus function, referred to as EXB, that performs parallel data transfer to/from the system (external) main MCU through an 8-bit bus. 38K0/38K2 can be accessed as an external device by the system main MCU, such as an external memory or peripheral device. 38K0/38K2 also performs as a slave device with a theoretical transfer speed of internal clock ø ÷ 5 bytes/sec (1.6 Mbytes/s ø = 8MHz).

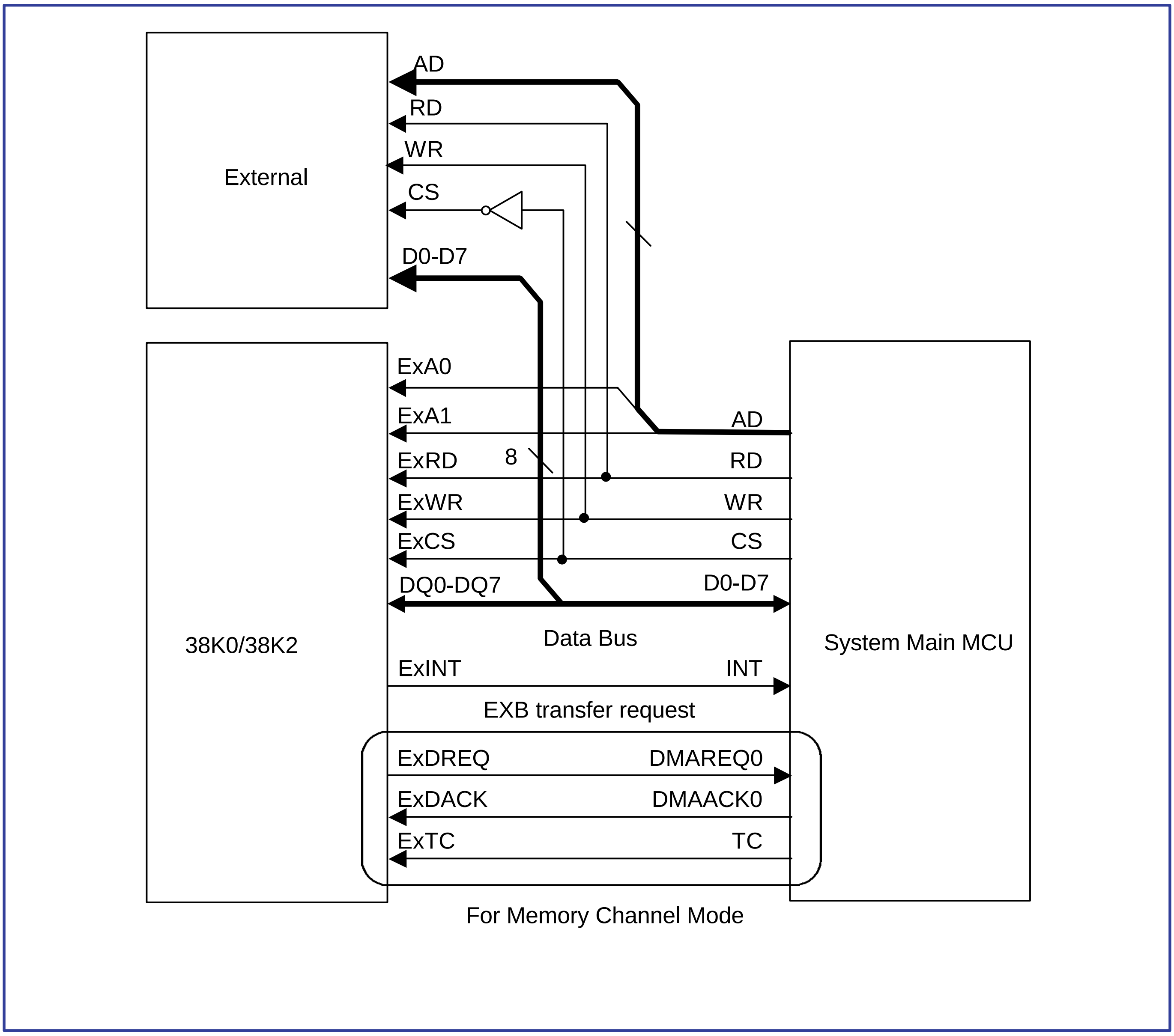


Figure 1.2 (1) EXB External Configuration

EXB can be transmit and receive data in two modes: CPU channel mode and memory channel mode. The features of each mode are described below.

●CPU Channel Mode

- One-byte transfers of data that can be read-accessed by the 38K0/38K2 CPU.
- Interrupt (CPU channel interrupt) generation in 38K0/38K2 for every transfer
- Applicable for one-byte control code transfers

●Memory Channel Mode

- Continuous data transfers to 38K0/38K2 RAM area that is specified by start and end addresses.
- Interrupt generation (memory channel interrupt) in the CPU after specified number of bytes is transferred.
- Simple DMA transfer (system main MCU does not need to be prepared for DMA)
  Applicable for block data transfer

38K0/38K2 can also be applied as a data converter device with the combined use of EXB and USB functions. The transfer rate between USB and EXB(note 1) can be more than 640 Kbytes/s when ø = 8MHz in BULK transfers, although it depends on system, firmware and OS configurations (Note 2).

Note 1) EXB and USB functions are not connected in hardware. Bridging operation is required.

Note 2) Transfer rate varies according to the system main MCU speed, 38K0/38K2 internal clock ø, and OS (driver) polling frequency. Theoretical value is approx. 900Kbytes/sec.

Figure 1.2 (2) shows a EXB transfer model. In the CPU channel mode, the CPU transfers data with its register; in the memory channel mode, the memory channel controller performs the transfers.

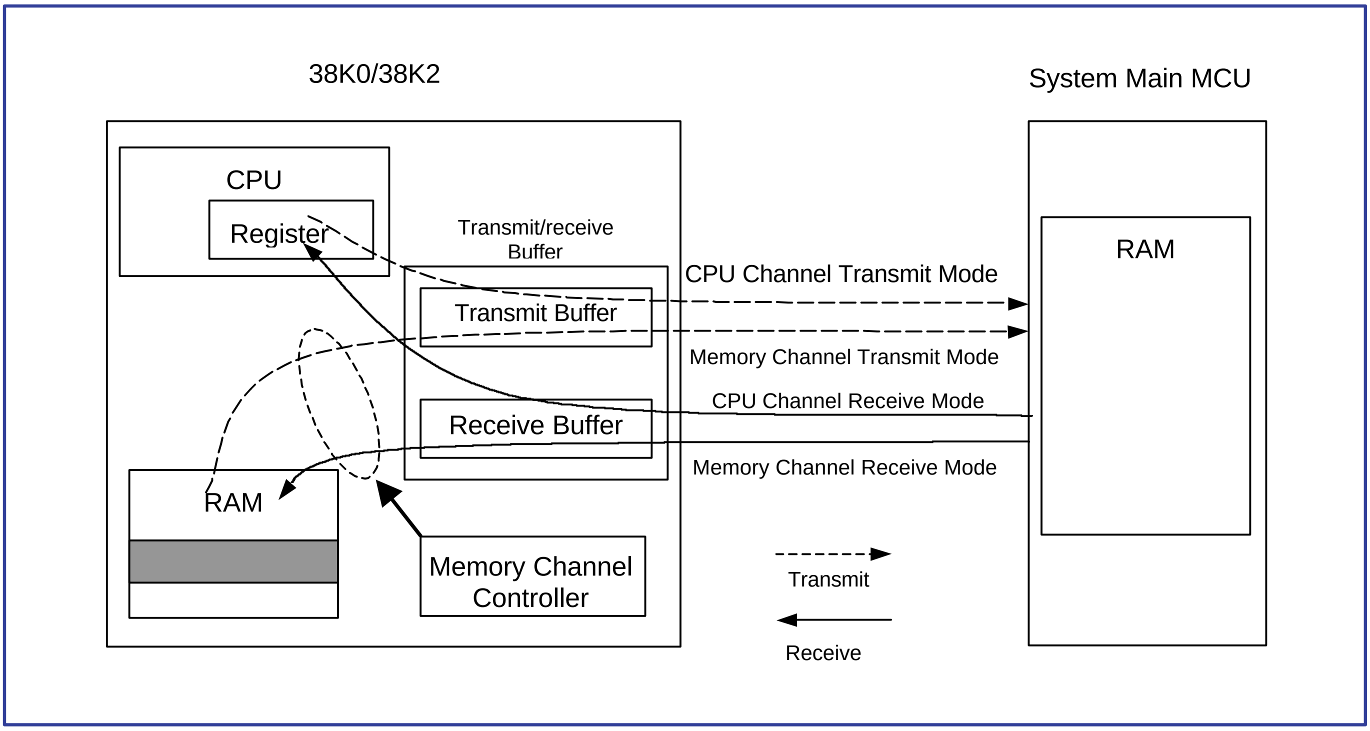


Figure 1.2 (2) EXB Transfer Model

Chapter 2
EXB Basic Operations

2.1 Pin Descriptions

38K0/38K2 has a total of 17 pins dedicated for the EXB function, including 8 data bus pins. Table 2.1 provides descriptions of each pin function. EXB requires a minimum of 13 pins, as indicated in the white area in the table.

Figure 2.1 38K0/38K2 EXB Pin Function Descriptions

Pin No.	Pin Name	Function
ExA1 (54)	Status Read Select Input	●This is the input control pin for the system main MCU to read the register that determines the interrupt factor when the CPU channel interrupt request and memory channel interrupt request are both assigned to the external EXB interrupt pin (EXINT).
ExINT (58)	External EXB Interrupt Request Output	●This is the EXB interrupt request output pin for 38K0/38K2. This pin outputs "L" when the CPU channel is ready or the memory channel request is generated. This enables the system main MCU to execute the transfer in sync with 38K0/38K2.
ExCS (59)	External Chip Select Input	●This is the chip select pin for the system main MCU.
		When ExCS = "L", data is read/written from/to 38K0/38K2.
ExWR (60)	External Write Signal Input	●This is the write signal input pin for the system main MCU.
ExRD (61)	External Read Signal Input	●This is the read signal input pin for the system main MCU.
ExA0 (62)	External Address Input	●This is the address input pin for the system main MCU to access either the CPU channel mode register or memory channel mode register (See Note 1)
		When ExA0 = "H", data bus operates in the CPU channel mode.
		When ExA0 = "L", data bus operates in the memory channel mode.
		When the system main MCU accesses 38K0/38K2 with an ExRD strobe and ExA1 = "H", ExCS = "L", the system main MCU can read the EXB interrupt factor register (EXBREQ), confirming the EXB interrupt factor.
DQ0-DQ7(63-6)	External Data Bus	●This is the EXB data bus (8 bits).
ExDREQ (51)	DMA Request Output	○This is the request output pin for DMA. 38K0/38K2 outputs a memory channel request to the system main MCU.
ExDACK (52)	DMA Acknowledge Input	○This is the acknowledge input pin for DMA. It receives an acknowledgement of the DMA request from the system main MCU. (See Note 2)
ExTC (53)	Terminal Count Input	○This is the terminal count input pin for DMA. When the system main MCU applies this signal to 38K0/38K2, 38K0/38K2 terminates the on-going transfer in the memory channel operation and generates a memory channel interrupt.

Note 1: Before controlling the address input ExS0 pin, the EXB interrupt factor register (EXBIREQ) must be checked using the ExA1 pin.

Note 2: In the memory channel mode, the system main MCU needs to apply ExRD/ExWR strobes with ExCS = "L" and ExA0 = "L". However, with a special setting for external configuration register H, the ExDACK pin can replace the functions of these 3 pins

2.2 EXB Registers

38K0/38K2 has 13 registers dedicated for the EXB function, including the index register and register window. Eight of the registers are mapped in the same register window (same address space), and are selected in the index register. There are multiple EXB interrupt enable bits and associated EXB interrupt factors bits in the interrupt vector area. This section explains the EXB registers and bits in detail.

Table 2.2 (1) provides a list of EXB registers and Table 2.2 (2) shows the register sets that can be switched by the index register.

Table 2.2 (1) EXB Registers

Address	Symbol	Register Name
0030H	EXBICON	EXB interrupt factor enable register
0031H	EXBIREQ	EXB interrupt factor register
0032H		
0033H	EXBINDEX	Index register
0034H	EXBREG1	Register window 1 (LOW)
0035H	EXBREG2	Register window 2 (HIGH)

003CH	IREQ1	Interrupt request register 1
003EH	ICON1	Interrupt control register 1

Can be read from external bus when ExA1 = "H"

Table 2.2 (2) Registers Selected in Index Register

Index Register (0033H)	Address (Window)	Symbol	Register Name
00H	0034H	EXBCFGL	External I/O configuration register L
	0035H	EXBCFGH	External I/O configuration register H
01H	0034H	RXBUF/TXBUF	Transmit/receive buffer register
	0035H		
02H	0034H	MCHMOD	Memory channel operation mode register
	0035H		
03H	0034H	MEMADL	Memory address counter L
	0035H	MEMADH	Memory address counter H (lower 3 bits only)
04H	0034H	ENDADL	End address register L
	0035H	ENDADH	End address register H (lower 3 bits only)

Setting pin ExA1 = "H" shows the contents of EXB interrupt factor register (0031H9) on the external data bus (DQ0 to DQ7). The system main MCU reads the contents of the EXB interrupt factor register (0031H) with an ExRD strobe.

2.2.1 Interrupt Control Register 1

Figure 2.2.1 shows the bit configurations of interrupt control register 1 (in the interrupt vector area). Refer to the 38K0 (38K2) Group Data Sheet for information on other interrupts.

● **Interrupt Control Register 1** **ICON1** **[003EH]**

Bit	Bit Name	Function	Reset	R/W
0	USB bus reset interrupt enable bit	0: interrupt disabled 1: interrupt enabled	0	R/W
1	USB SOF interrupt enable bit	0: interrupt disabled 1: interrupt enabled	0	R/W
2	USB device interrupt enable bit	0: interrupt disabled 1: interrupt enabled	0	R/W
3	EXB interrupt enable bit	0: interrupt disabled 1: interrupt enabled	0	R/W
4	INT0 interrupt enable bit	0: interrupt disabled 1: interrupt enabled	0	R/W
5	Timer X interrupt enable bit	0: interrupt disabled 1: interrupt enabled	0	R/W
6	Timer 1 interrupt enable bit	0: interrupt disabled 1: interrupt enabled	0	R/W
7	Timer 2 interrupt enable bit	0: interrupt disabled 1: interrupt enabled	0	R/W

Figure 2.2.1 Bit Configurations of Interrupt Control Register 1

2.2.2 Interrupt Request Register 1

Figure 2.2.2 shows the bit configurations of interrupt request register 1 (in the interrupt vector area).

● **Interrupt Request Register 1** **IREQ1** **[003CH]**

Bit	Bit Name	Function	Reset	R/W
0	USB bus reset interrupt request bit	0: no interrupt request 1: interrupt request	0	R/W
1	USB SOF interrupt request bit	0: no interrupt request 1: interrupt request	0	R/W
2	USB Device interrupt request bit	0: no interrupt request 1: interrupt request	0	R/W
3	EXB interrupt request bit	0: no interrupt request 1: interrupt request	0	R/W
4	INT0 interrupt request bit	0: no interrupt request 1: interrupt request	0	R/W
5	Timer X interrupt request bit	0: no interrupt request 1: interrupt request	0	R/W
6	Timer 1 interrupt request bit	0: no interrupt request 1: interrupt request	0	R/W
7	Timer 2 interrupt request bit	0: no interrupt request 1: interrupt request	0	R/W

Note 1) This register is only write-valid for “0”.

Figure 2.2.2 Bit Configurations of Interrupt Request Register 1

2.2.3 Interrupt Factor Enable Register

Figure 2.2.3 shows the bit configurations of the EXB interrupt factor enable register.

● **EXB Interrupt Factor Enable Register EXBICON** **[0030H]**

Bit	Bit Name	Function	Reset	R/W
0	CPU channel receive enable bit (RXB_ENB)	0: Interrupt disabled 1: Interrupt enabled (receive buffer full interrupt)	0	R/W
1	CPU channel transmit enable bit (TXB_ENB)	0: Interrupt disabled 1: Interrupt enabled (transmit buffer empty interrupt)	0	R/W
2	Memory channel operation enable bit (MC_ENB)	0: Interrupt disabled 1: Interrupt enabled (memory channel operation completed interrupt)	0	R/W
7-3	Unused	Set to “0” when writing.	-	-

Note 1) To enable each mode, set the corresponding bit to “0” first, then to “1”.

Note 2) Only one of the 3 bits bit can be set to “1”.

Figure 2.2.3 Bit Configurations of EXB Interrupt Factor Enable Register

(1) CPU Channel Receive Enable Bit (RXB_ENB)

When this bit is “1” (interrupt enabled), 38K0/38K2 latches the value of the external bus to the receive buffer register at the rising edge of the ExWR strobe with ExA0 = “H” and ExCS = “L”, and then generates an EXB interrupt (receive buffer full interrupt).

(2) CPU Channel Transmit Enable Bit (TXB_ENB)

When this bit is “1” (interrupt enabled), 38K0/38K2 outputs the value of the transmit buffer register to the external bus at the falling edge of the ExRD strobe with ExA0 = “H” and ExCS = “L”, and then generates an EXB interrupt (transmit buffer empty interrupt). However, when this bit is reset from “0” to “1”, the transmit/receive buffer register is recognized as empty, causing the first EXB interrupt factor to be generated (regardless of ExRD).

(3) Memory Channel Operation Enable Bit (MC_ENB)

In the memory channel mode, 38K0/38K2 increments the memory address counter (start address of RAM area) by “1” each time one byte of data is transferred. When this memory address counter reaches the value of the end address register of RAM buffer area) plus 1, 38K0/38K2 completes the memory channel operation. If the MC_ENB bit is “1” (interrupt enabled) at this time, 38K0/38K2 generates an EXB interrupt (memory channel operation completed interrupt).

2.2.4 EXB Interrupt Factor Register

Figure 2.2.4 shows the bit configurations for the EXB interrupt factor register. Note that if 38K0/38K2 detects ExRD when ExA1 = "H" and ExCS = "L", it outputs the value of this register to the external bus (status read function).

● **EXB Interrupt Factor Register** **EXBIREQ** **[0031H]**

Bit	Bit Name	Function	Reset	R/W
0	Receive buffer full bit (RXB_FULL)	0: No interrupt request 1: Receive buffer full (generates interrupt)	0 Note 2	- Note 1
1	Transmit buffer empty bit (TXB_EMPTY)	0: No interrupt request 1: Transmit buffer empty (generates interrupt)	0 Note 3	- Note 1
3-2	Memory channel status bit (MC_STS)	00: Memory channel operation stopped 01: Memory channel operation (no access occurred) 10: Memory channel operation (accessed) 11: Memory channel operation completed (Generates memory channel operation completed interrupt)	00	- Note 1
7-4	Unused	Set to "0" when writing.	-	-

Note 1) This register is write-invalid for "0" or "1".

Note 2) This bit is set to "0" when reading the receive buffer or disabling the CPU channel receive operation.

Note 3) This bit is set to "0" when writing to the transmit buffer or when disabling the CPU channel transmit operation.

Figure 2.2.4 Bit Configurations of EXB Interrupt Factor Register

The EXB interrupt factor register is a reference register for the system main MCU, and 38K0/38K2 cannot set it to either "0" or "1" using the connection firmware. The receive buffer full bit and the transmit buffer empty bit are set to "0" when reading the receive buffer or writing to the transmit buffer, respectively. In addition, all bits of the EXB interrupt factor register go to "0" when writing "0" to bit 1 of the EXB interrupt factor enable register (EXBICON).

Note that the EXB interrupt request bit (bit 3 of IREQ1) can be written to "0" by 38K0/38K2 firmware.

(1) Memory Channel Status Bits (MC_STS)

These status bits indicate the status of the memory channel mode.

Table 2.2.4 shows the transitions of the memory channel status bits (MC_STS).

Table 2.2.4 Memory Channel Status Bit (MC_STS) Transitions

MC_STS	Status
00	Memory channel operation disabled
01	Memory channel operation enabled, no ExRD/ExWR access from system main MCU occurred.
10	Memory channel operation enabled, at least one ExRD/ExWR access from system main MCU, but all transfers not completed yet.
11	Memory channel operation enabled, all transfers completed.

2.2.5 External I/O Configuration Register (Index: 00H)

Figure 2.2.5 shows the bit configurations of external configuration register L.

● External I/O Configuration Register L EXBCFGL [0034H]

Bit	Bit Name	Function	Reset	R/W
0	EXB port control bit (P10-P17,P33-P37) (EXB_CTR)	0: General-use port 1: EXB function port (P33 = ExINT/P34 = ExCS) /P35=ExWR/P36=ExRD/P37=ExA0)	0	R/W
3-1	ExINT port control bits (P33) (INT_CTR)	xx1: Receive buffer empty output x1x: Transmit buffer full output Note 1 1xx: DMA request output (memory channel)	000	R/W
4	ExA1 port control bit (P43) (A1_CTR)	0: General-use port 1: A1 Input (EXBIREQ read control port)	0	R/W
7-5	Unused	Set to “0” when writing.	-	-

Note 1) ExINT output is AND for all port outputs (at least one ready signal is “L” and output is “L”).

Figure 2.2.4 Bit Configurations of External I/O Configuration Register L

(1) EXB Port Control Bit (EXB_CTR)

When this bit is “1”, P10 to P17 become the external bus DQ0 to DQ7; P33, ExINT; P34, ExCS; P35, ExWR; P36, ExRD; and P37, ExA0.

(2) ExINT Port Control Bits (INT_CTR)

These three bits set the ready/request signal (low active) output factor for the ExINT port. The factor for all three bits can be set simultaneously and the output signal is AND (at least one ready signal is “L”, output is “L”)

(3) ExA1 Port Control Bit (A1_CTR)

When this bit is “1”, P43 becomes the ExA1 (input) port and behaves as the EXBIREQ read control port. If 38K0/38K2 detects an ExRD strobe when ExA1 = “H” and ExCS = “L”, the EXB interrupt factor register (EXBIREQ) is output to the external bus.

2.2.6 External I/O Configuration Register H (Index: 00H)

Figure 2.2.6 shows the bit configurations of external I/O configuration register H.

● **External I/O Configuration Register H EXBCFGH [0035H]**

Bit	Bit Name	Function	Reset	R/W
1-0	ExDREQ/ExRxD port control bit (P40) (DRQ_CTR)	00: General-use port 01: Not available 10: Receive buffer empty output 11: DMA request output (memory channel)	00	R/W
3-2	ExDACK port control bit (P41) (DAK_CTR)	00: General-use port 01: Not available 10: Acknowledge input (shared with ExRD/ExWR signal) 11: Acknowledge input (ExRD/ExWR signal unnecessary) Note 1	00	R/W
4	ExTC port control bit (P42) (TC_CTR)	0: General-use port 1: ExTC (terminal count) input	0	R/W
7-5	Unused	Set to “0” when writing.	-	-

Note 1) 11: Acknowledge Input - in the ‘ExRD/ExWR signal unnecessary’ mode, ExDACK can perform ExRD/ExWR if 38K0/38K2 is the only MCU connected to the system main MCU.

Figure 2.2.6 Bit Configurations of External I/O Configuration Register H

(1) ExDREQ/ExRxD Port Control Bits (DRQ_CTR)

When these bits are “10” in the CPU channel receive mode, 38K0/38K2 assigns the receive buffer empty output to this port. When “11” in the memory channel mode, the DMA request output is assigned to this port.

(2) ExDACK Port Control Bits (DAK_CTR)

When these bits are “10” in the memory channel mode, the system main MCU must read/write data according to the ExRD/ExWR signal. When “11”, the system main MCU can read/write data according to the ExDACK signal.

(3) ExTC Port Control Bit (TC_CTR)

When this bit is “1” in the memory channel mode, the system main MCU can end a transfer using the ExTC signal.

2.2.7 Transmit/Receive Buffer Registers (Index: 01H)

Figure 2.2.7 shows the bit configurations of the transmit and receive buffer registers. Both registers are assigned to the same address although their polarities differ.

● **Transmit Buffer Register** **RXBUF** **[0034H]**

Bit	Bit Name	Function	Reset	R/W
7-0	TXBUF		00H	W

● **Receive Buffer Register** **TXBUF** **[0034H]**

Bit	Bit Name	Function	Reset	R/W
7-0	RXBUF	Data is received at the ExWR rising edge.	00H	R

Note 1) These buffers also use the memory channel controller in the memory channel mode. Therefore, transfers in the memory channel mode may be affected if the address is read/write accessed in the CPU channel mode during memory channel operations

Figure 2.2.7 Bit Configurations of Transmit/Receive Buffer Registers

These registers enable data to be accessed to/from the EXB by the CPU in the CPU channel mode or by the memory channel controller in the memory channel mode.

In the CPU channel receive mode, if 38K0/38K2 detects a rising edge of ExWR when ExA0 = “H” and ExCS = “L”, the value of the external bus is latched by the receive buffer. However, if the CPU channel receive interrupt is not enabled, this event (interrupt) is not notified to the CPU.

In the CPU channel transmit mode, if 38K0/38K2 detects a falling edge of ExRD when ExA0 = “H” and ExCS = “L”, the value in the transmit buffer is output to the external bus.

Note that when the CPU channel transmit mode is not enabled in 38K0/38K2, data cannot be written to the transmit buffer with firmware. In addition, when this interrupt is not necessary for generating the factor for the first interrupt (transmit buffer empty interrupt) in an interrupt enable event, set the EXB interrupt request flag to “0” after writing data to the transmit buffer (the transmit buffer empty bit cannot be set to “0” directly).

2.2.8 Memory Channel Operation Mode Register (Index: 02H)

Figure 2.2.8 shows the bit configurations of the memory channel operation mode register.

● **Memory Channel Operation Mode EXBCFGH [0034H]**

Bit	Bit Name	Function	Reset	R/W
1-0	Memory channel direction control bit (MC_DIR)	00: Operation disabled 01: Receive mode 10: Transmit mode 11: Not available	00	R/W
2	Burst bit (BURST)	0: Cycle mode 1: Burst mode	0	R/W
7-3	Unused	Set to “0” when writing.	-	-

Figure 2.2.8 Bit Configurations of Memory Channel Operation Mode Register

(1) Memory Channel Direction Control Bit (MC_DIR)

These bits set the direction (transmit/receive) for data transfer in the memory channel mode.

(2) Burst Bit (BURST)

This bit sets the outburst type of DMA request (ExDREQ port) in the memory channel mode. The output type changes according to the timing of the mode: when the bit is “0”, DMA requests are output in the cycle mode (ExDREQ port goes to “H” each time 1 byte is output); when “1”, requests are output in the burst mode (ExDREQ goes to “H” when all transfers are completed or when 38K0/38K2 detects the terminal count (ExTC)).

2.2.9 Memory Address Counter L/H (Index: 03H)

Figure 2.2.9 shows the bit configurations of the memory address counter (L/H).

● Memory Address Counter L MEMADL [0034H]

Bit	Bit Name	Function	Reset	R/W
7-0	(IM_A)	Sets the lower bits of the start address for memory channel operations. Note 1 Value increases by “1” for each RAM access.	00H	R/W

● Memory Address Counter H MEMADH [0035H]

Bit	Bit Name	Function	Reset	R/W
2-0	(IM_A)	Sets the upper 3 bits of the start address for memory channel operations. Note 1	000	R/W
7-3	Unused	Set to “0” when writing.	-	-

Note 1) Only specify a RAM area address to this register.

Figure 2.2.9 Bit Configurations of Memory Address Counter L/H

The memory address counter L/H register specifies the start address for the transfer area in the memory channel mode. This register is incremented by “1” each time 1 byte of data is transferred in the memory channel mode, so it is necessary to reset the value every time a new transfer is started.

Do not specify an address (such as an SFR address) for this register that is not within the RAM area.

2.2.10 End Address Register L/H (Index: 04H)

Figure 2.2.10 shows the bit configurations of end address register L/H.

● End Address Register L ENDADL [0034H]

Bit	Bit Name	Function	Reset	R/W
7-0	(END_A)	Sets the lower bits of the end address for the memory channel mode. Note 1	00H	R/W

● End Address Register H ENDADH [0035H]

Bit	Bit Name	Function	Reset	R/W
2-0	(END_A)	Sets the upper 3 bits of the end address for the memory channel mode. Note 1	000	R/W
7-3	Unused	Set to “0” when writing.	-	-

Note 1) Only specify a RAM area address to this register.

Figure 2.2.10 Bit Configurations of End Address Register L/H

The end address register L/H specifies the end address of the memory channel mode transfer area. The value of this register does not vary, as it does in the memory address counter L/H. After the initial value is set it does not need to be reset even after an interrupt occurs. Memory channel operations are completed when the memory address counter reaches the value of this register +1.

Do not specify an address (such as an SFR address) for this register that is not within the RAM area.

2.3 CPU Channel Mode

In the CPU channel mode, the CPU transfers data from the transmit/receive buffer register in 1-byte units to the system main MCU. 38K0/38K2 generates an interrupt (CPU channel interrupt) after each 1-byte transfer.

Figure 2.3 shows an example of an EXB transfer in the CPU channel mode.

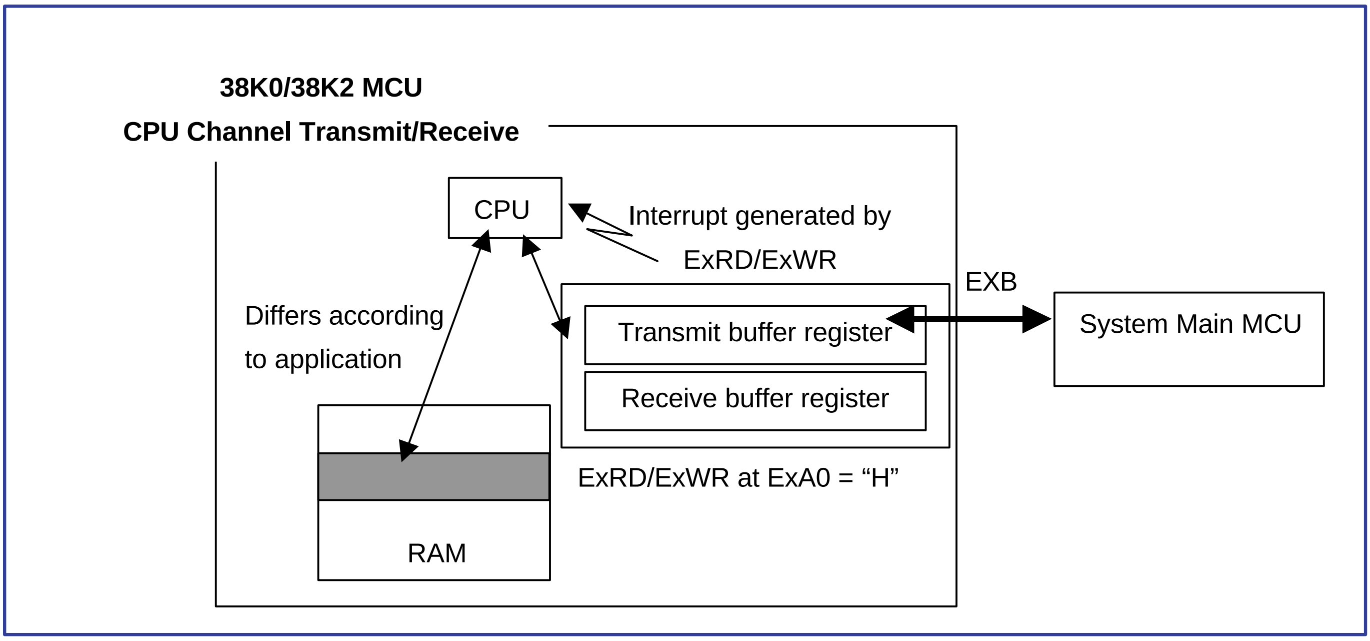
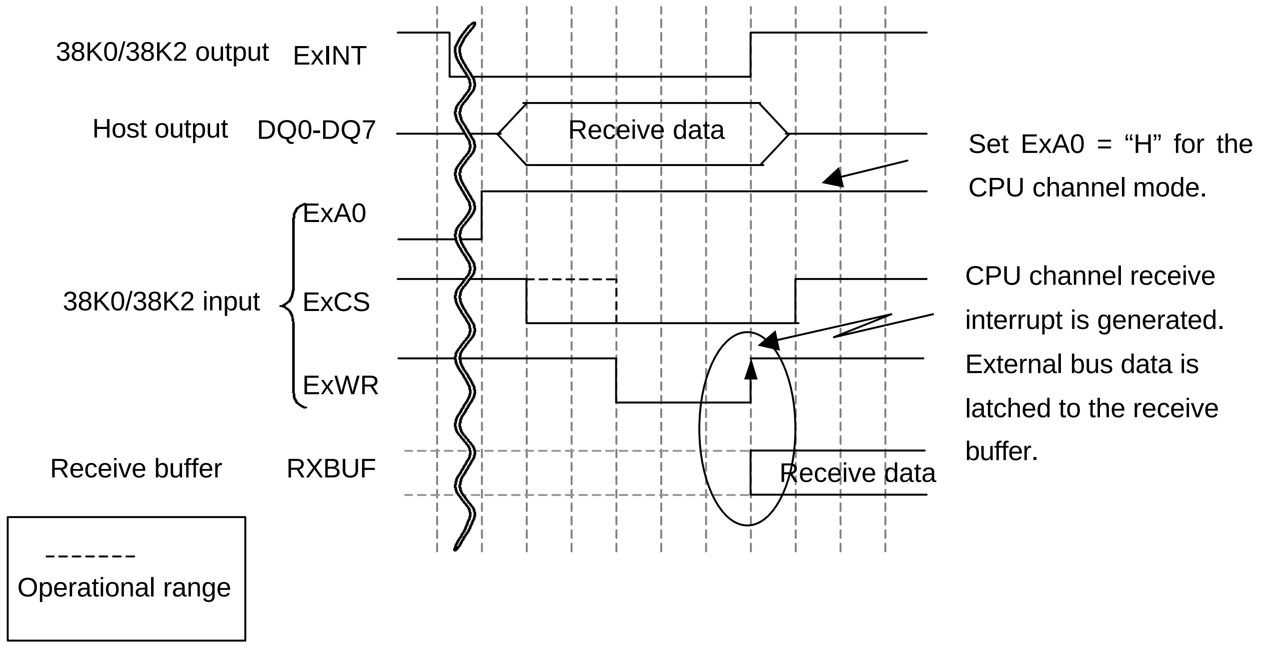


Figure 2.3 EXB Transfer Example

2.3.1 CPU Channel Receive Mode

In the CPU channel receive mode, the CPU processes 1-byte units of data that was written to the receive buffer register from 38K0/38K2.

As shown in Figure 2.3.1 (1), when the receive buffer is ready (receive buffer empty), and ExINT is “L” (depending on the setting of external I/O configuration register L; 38K0/38K2 triggers an ExWR strobe to the system main MCU. After this, if the system main MCU raises the ExWR when ExA0 = “H” and ExCS = “L”, 38K0/38K2 latches one byte of DQ0-DQ7 to the receive buffer register, raises the ExINT, and generates a CPU channel receive interrupt.


38K0/38K2 output
ExINT
Host output
DQ0-DQ7
Receive data
ExA0
38K0/38K2 input
ExCS
ExWR
Receive buffer
RXBUF
Receive data

Operational range
Set ExA0 = “H” for the CPU channel mode.
CPU channel receive interrupt is generated. External bus data is latched to the receive buffer.

For a more detailed timing chart, refer to 38K0 (38K2) Group Data Sheet.

Figure 2.3.1 (1) Timing Waveform in CPU Channel Receive Mode

Figure 2.3.1 (2) shows settings for registers related to the CPU channel receive mode.

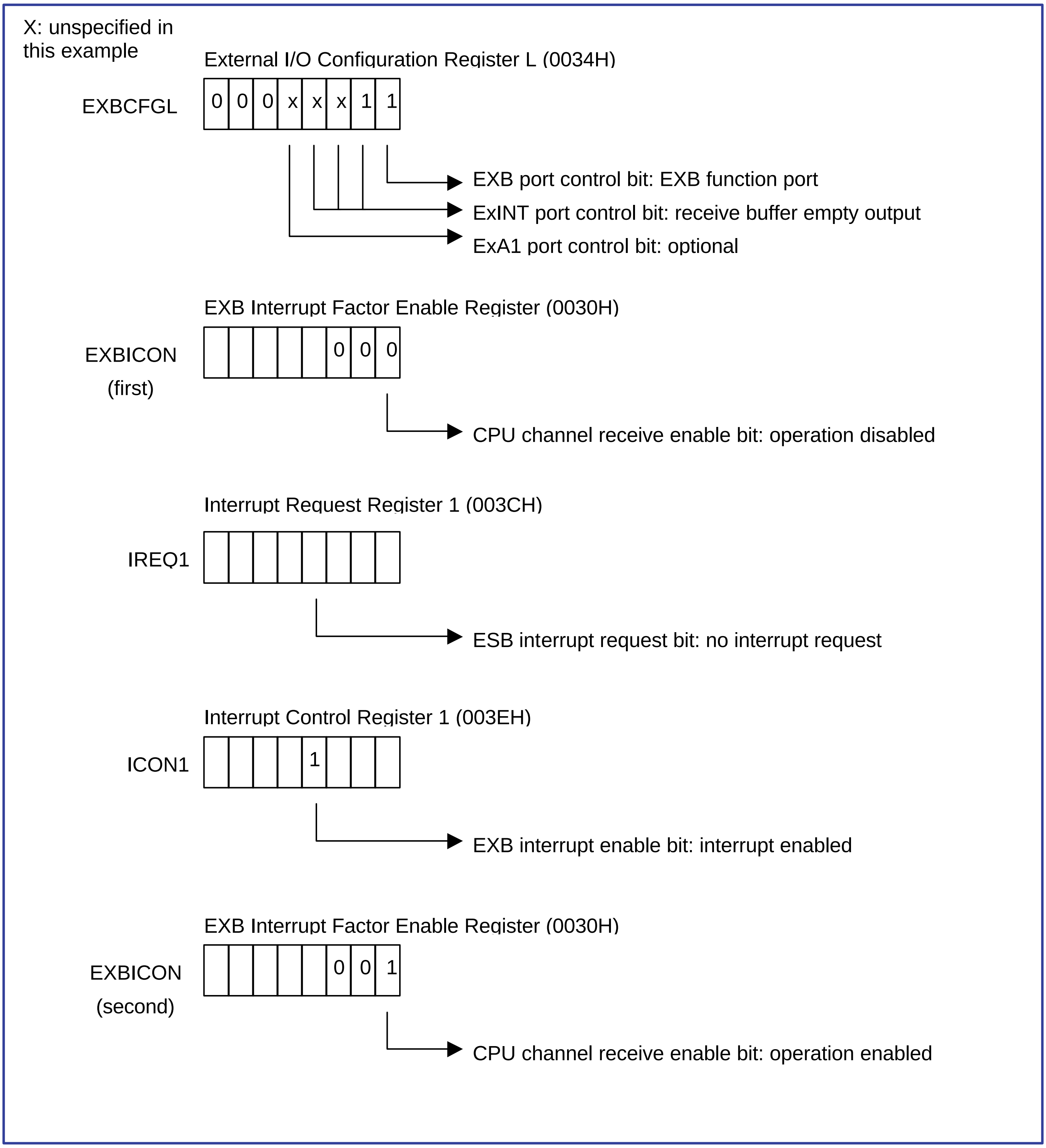
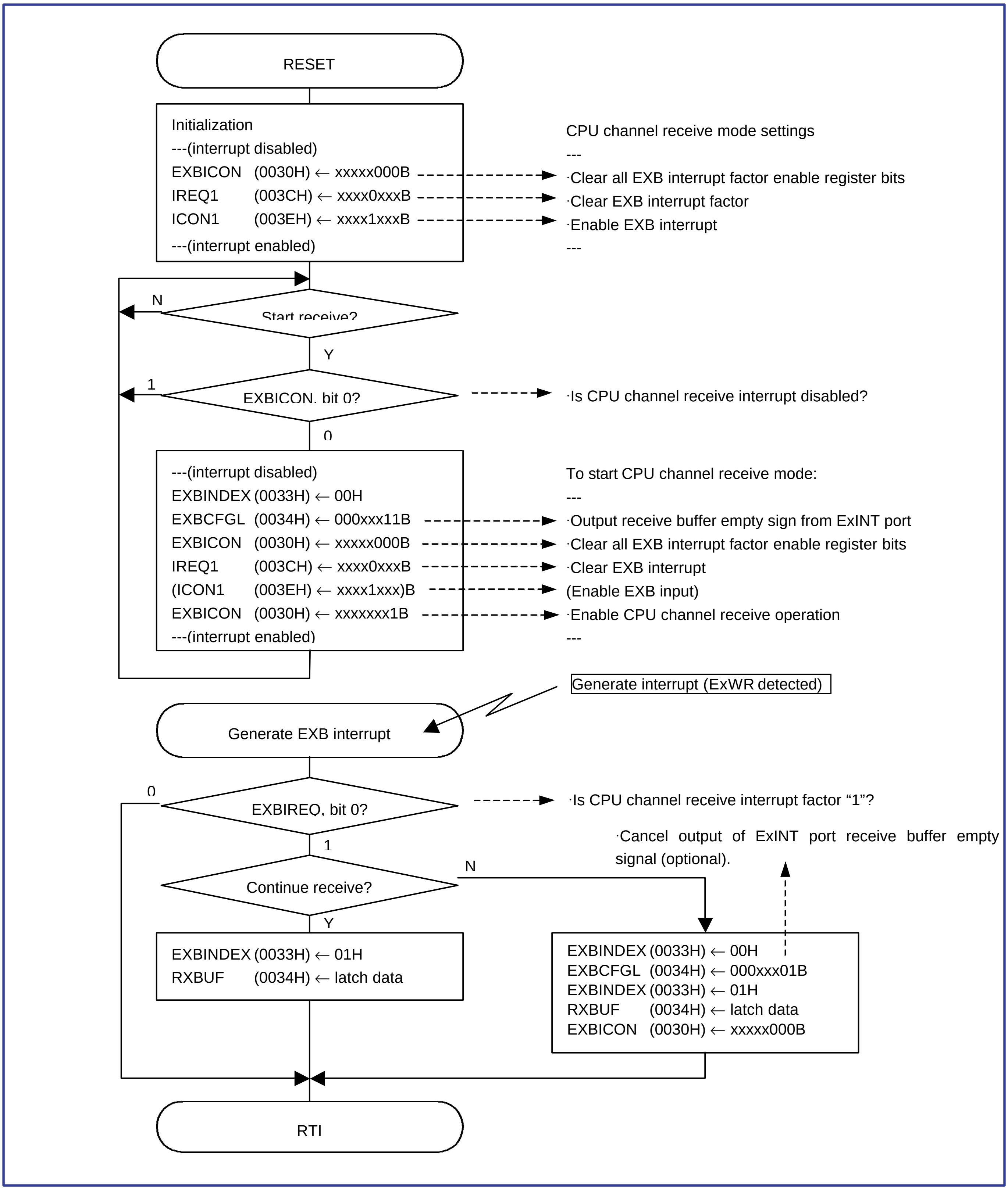

X: unspecified in this example
External I/O Configuration Register L (0034H)
EXBCFGL
0 0 0 x x x 1 1
EXB port control bit: EXB function port
ExINT port control bit: receive buffer empty output
ExA1 port control bit: optional
EXB Interrupt Factor Enable Register (0030H)
EXBICON
(first)
0 0 0
CPU channel receive enable bit: operation disabled
Interrupt Request Register 1 (003CH)
IREQ1
ESB interrupt request bit: no interrupt request
Interrupt Control Register 1 (003EH)
ICON1
1
EXB interrupt enable bit: interrupt enabled
EXB Interrupt Factor Enable Register (0030H)
EXBICON
(second)
0 0 1
CPU channel receive enable bit: operation enabled

Figure 2.3.1 (2) CPU Channel Receive Mode Registers

Figure 2.3.1 (3) shows a control example in the CPU channel receive mode.


RESET
Initialization
---(interrupt disabled)
EXBICON (0030H) ← xxxxx000B
IREQ1 (003CH) ← xxxx0xxxB
ICON1 (003EH) ← xxxx1xxxB
---(interrupt enabled)
CPU channel receive mode settings

·Clear all EXB interrupt factor enable register bits
·Clear EXB interrupt factor
·Enable EXB interrupt

N
Start receive?
Y
1
EXBICON, bit 0?
·Is CPU channel receive interrupt disabled?
0
---(interrupt disabled)
EXBINDEX (0033H) ← 00H
EXBCFGL (0034H) ← 000xxx11B
EXBICON (0030H) ← xxxxx000B
IREQ1 (003CH) ← xxxx0xxxB
(ICON1 (003EH) ← xxxx1xxx)B
EXBICON (0030H) ← xxxxxxx1B
---(interrupt enabled)
To start CPU channel receive mode:

·Output receive buffer empty sign from ExINT port
·Clear all EXB interrupt factor enable register bits
·Clear EXB interrupt
(Enable EXB input)
·Enable CPU channel receive operation

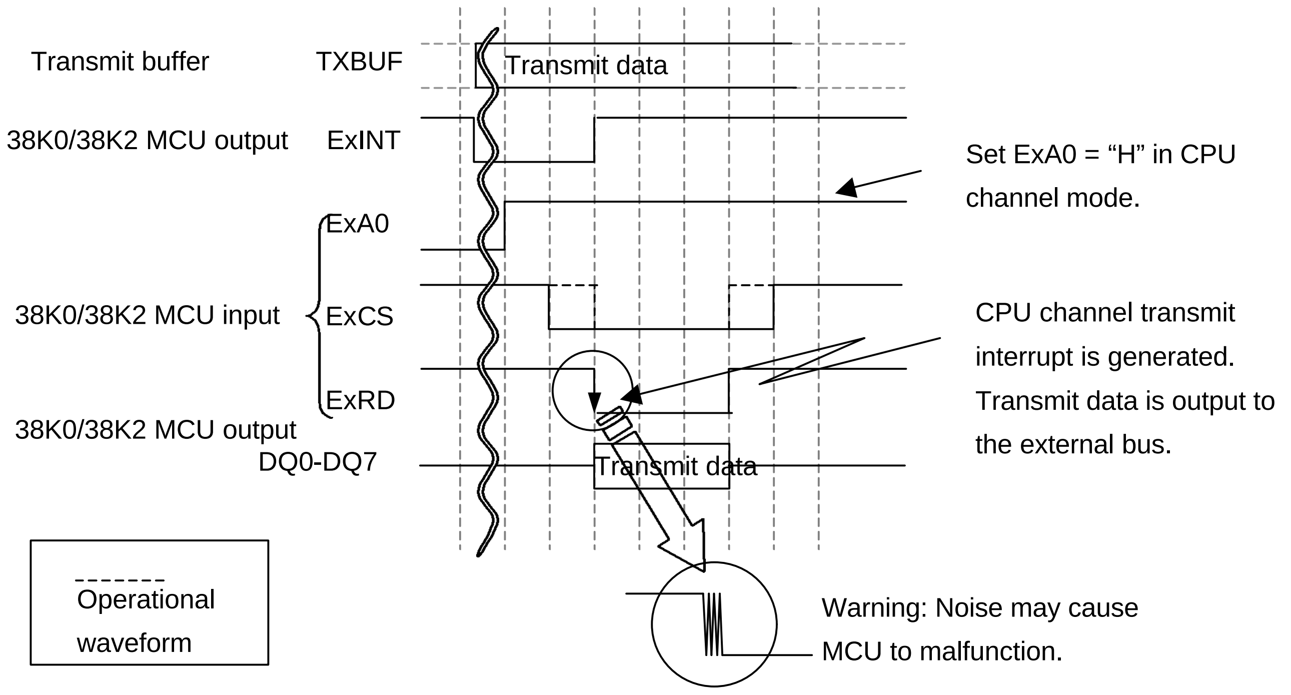
Generate interrupt (ExWR detected)
Generate EXB interrupt
0
EXBIREQ, bit 0?
·Is CPU channel receive interrupt factor "1"?
1
·Cancel output of ExINT port receive buffer empty signal (optional).
N
Continue receive?
Y
EXBINDEX (0033H) ← 01H
RXBUF (0034H) ← latch data
EXBINDEX (0033H) ← 00H
EXBCFGL (0034H) ← 000xxx01B
EXBINDEX (0033H) ← 01H
RXBUF (0034H) ← latch data
EXBICON (0030H) ← xxxxx000B
RTI

Figure 2.3.1 (3) Control Example in CPU Channel Receive mode

2.3.2 CPU Channel Transmit Mode

In the CPU channel transmit mode, the CPU writes the data to be sent to the system main MCU in one-byte units to the transmit buffer register.

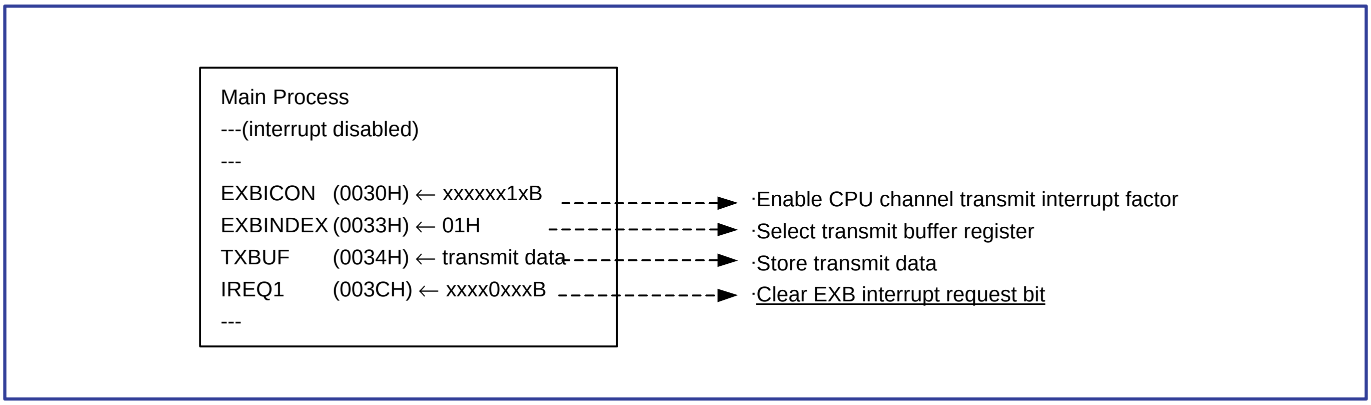
As shown in Figure 2.3.2 (1), when transmit data is written to the transmit buffer, the transmit buffer goes to the ready state, ExINT is output as “L” (depending on the setting of external I/O configuration register L), and 38K0/38K2 then triggers an ExRD strobe to the system main MCU. After this, if the system main MCU lowers the ExRD when ExA0 = “H” and ExCS = “L”, 38K0/38K2 outputs one byte of the transmit buffer register to DQ0-DQ7, raises the ExINT, and generates a CPU channel transmit interrupt.


Transmit buffer
TXBUF
Transmit data
38K0/38K2 MCU output
ExINT
Set ExA0 = “H” in CPU channel mode.
ExA0
38K0/38K2 MCU input
ExCS
ExRD
CPU channel transmit interrupt is generated. Transmit data is output to the external bus.
38K0/38K2 MCU output
DQ0-DQ7
Transmit data
Operational waveform
Warning: Noise may cause MCU to malfunction.

For a more detailed timing chart, refer to 38K0 (38K2) Group Data Sheet.

Figure 2.3.1 (2) Timing Waveform in CPU Channel Transmit Mode

When a CPU channel transmit interrupt factor is enabled in the CPU channel transmit mode, 38K0/38K2 immediately generates the first (internal) interrupt request (regardless of ExRD output). If this interrupt is not necessary, set the EXB interrupt request bit to “0” after enabling the interrupt factor, as shown in Figure 2.3.2 (2). (Note that the transmit buffer empty bit cannot be set to “0” directly.)



```
Main Process
---(interrupt disabled)
---
EXBICON   (0030H) ← xxxxxx1xB       ·Enable CPU channel transmit interrupt factor
EXBINDEX  (0033H) ← 01H             ·Select transmit buffer register
TXBUF     (0034H) ← transmit data   ·Store transmit data
IREQ1     (003CH) ← xxxx0xxxB       ·Clear EXB interrupt request bit
---
```

Figure 2.3.2 (2) Clearing First Interrupt Factor in CPU Channel Transmit Mode

Figure 2.3.2 (3) shows register settings for the CPU channel transmit mode.

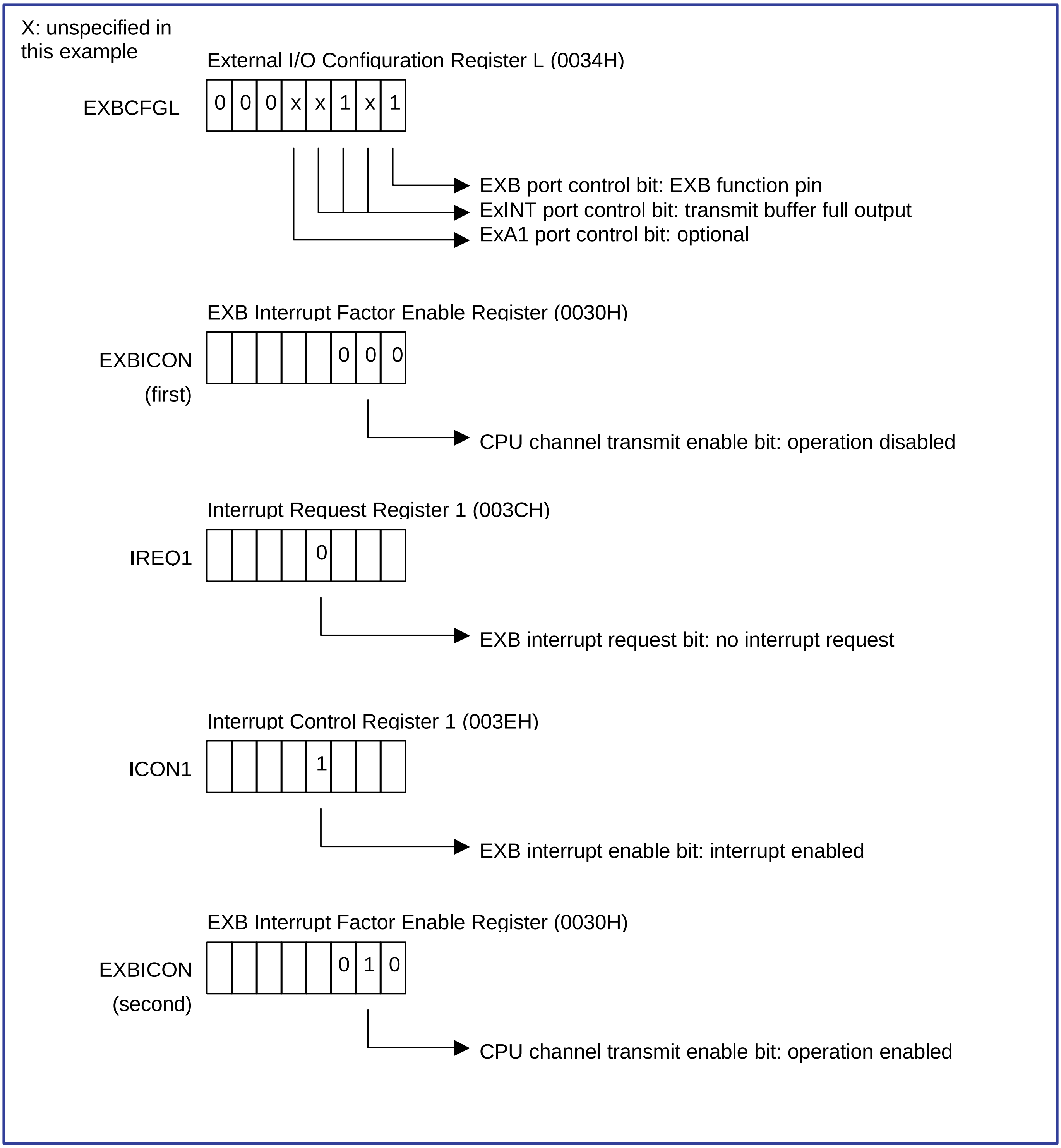
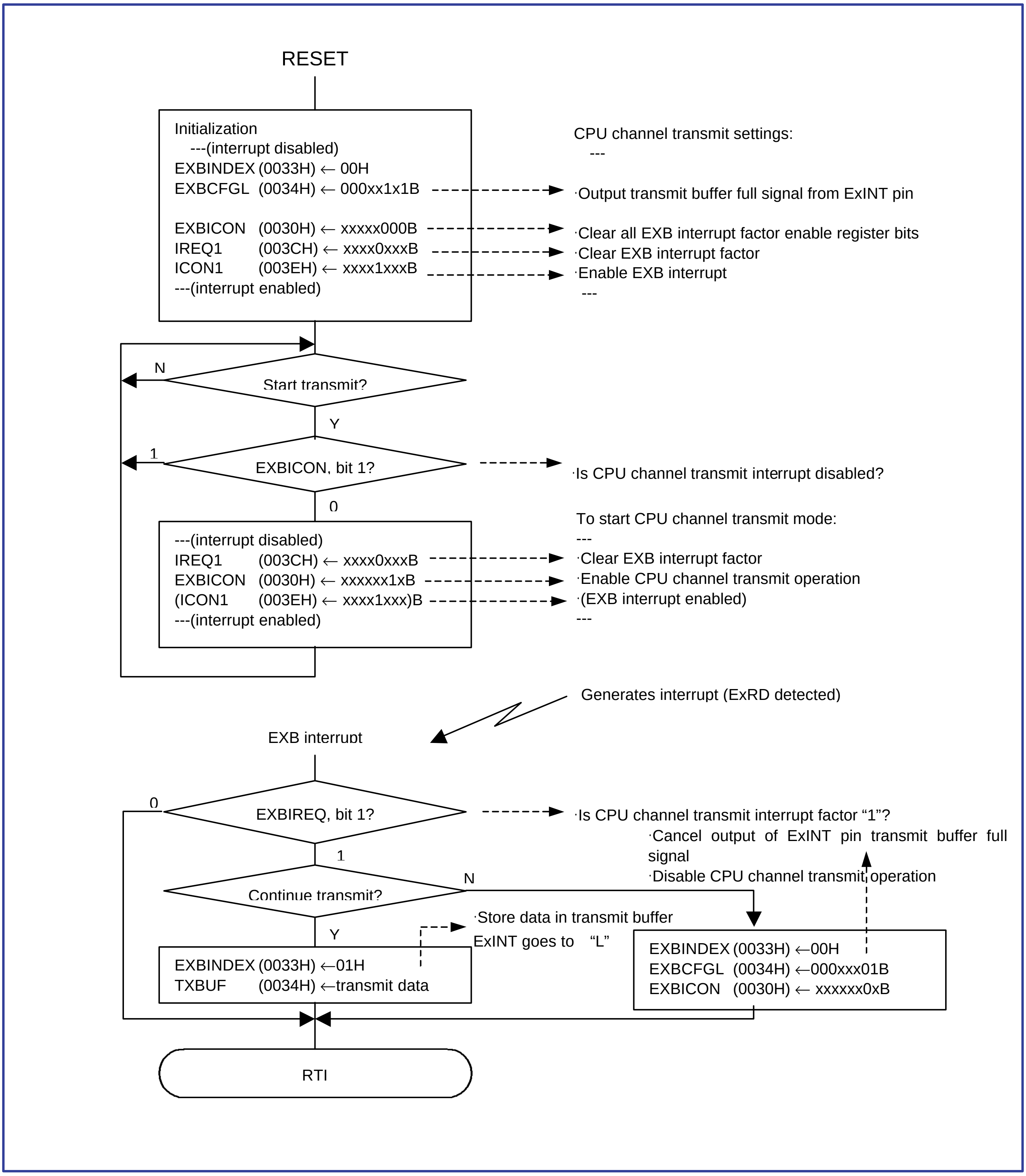

X: unspecified in this example
External I/O Configuration Register L (0034H)
EXBCFGL
0 0 0 x x 1 x 1
EXB port control bit: EXB function pin
ExINT port control bit: transmit buffer full output
ExA1 port control bit: optional
EXB Interrupt Factor Enable Register (0030H)
EXBICON (first)
0 0 0
CPU channel transmit enable bit: operation disabled
Interrupt Request Register 1 (003CH)
IREQ1
0
EXB interrupt request bit: no interrupt request
Interrupt Control Register 1 (003EH)
ICON1
1
EXB interrupt enable bit: interrupt enabled
EXB Interrupt Factor Enable Register (0030H)
EXBICON (second)
0 1 0
CPU channel transmit enable bit: operation enabled

Figure 2.3.2 (3) CPU Channel Transmit Mode Registers

Figure 2.3.2 (4) shows a control example in the CPU channel transmit mode.


RESET
Initialization
---(interrupt disabled)
EXBINDEX (0033H) ← 00H
EXBCFGL (0034H) ← 000xx1x1B
EXBICON (0030H) ← xxxxx000B
IREQ1 (003CH) ← xxxx0xxxB
ICON1 (003EH) ← xxxx1xxxB
---(interrupt enabled)
CPU channel transmit settings:

·Output transmit buffer full signal from ExINT pin
·Clear all EXB interrupt factor enable register bits
·Clear EXB interrupt factor
·Enable EXB interrupt

N
Start transmit?
Y
1
EXBICON, bit 1?
0
·Is CPU channel transmit interrupt disabled?
To start CPU channel transmit mode:

---(interrupt disabled)
IREQ1 (003CH) ← xxxx0xxxB
EXBICON (0030H) ← xxxxxx1xB
(ICON1 (003EH) ← xxxx1xxx)B
---(interrupt enabled)
·Clear EXB interrupt factor
·Enable CPU channel transmit operation
·(EXB interrupt enabled)

Generates interrupt (ExRD detected)
EXB interrupt
0
EXBIREQ, bit 1?
·Is CPU channel transmit interrupt factor "1"?
1
Continue transmit?
N
·Cancel output of ExINT pin transmit buffer full signal
·Disable CPU channel transmit operation
Y
·Store data in transmit buffer
ExINT goes to "L"
EXBINDEX (0033H) ←01H
TXBUF (0034H) ←transmit data
EXBINDEX (0033H) ←00H
EXBCFGL (0034H) ←000xxx01B
EXBICON (0030H) ← xxxxxx0xB
RTI

Figure 2.3.2 (4) Control Example in CPU Channel Transmit Mode

2.4 Memory Channel Mode

The memory channel mode enables continuous transmission of RAM area data without passing through the 38K0/38K2 CPU. The main purpose of this mode is continuous USB data transfer to the system main CPU (from the system main MCU).

Figure 2.4 (1) shows an example of an EXB transfer of USB data in the memory channel mode.

(1)38K0/38K2 uses a dedicated RAM area for the USB endpoint buffer.

(2)38K0/38K2 generates a USB device interrupt when the USB transfer is completed.

(3)The CPU (F/W) receives the interrupt, specifies the RAM address of the transfer area, and implements the memory channel mode.

(4)Once the memory channel mode is implemented, the memory channel controller accesses the transfer data in the transmit/receive buffer register, and transfers data to/from the system main MCU, accordingly.

(5)38K0/38K2 generates a memory channel interrupt after all data is transferred to/from the specified RAM address (Note 1).

Note 1) The RAM address can also be specified as an area other than the area in which the USB data is stored.

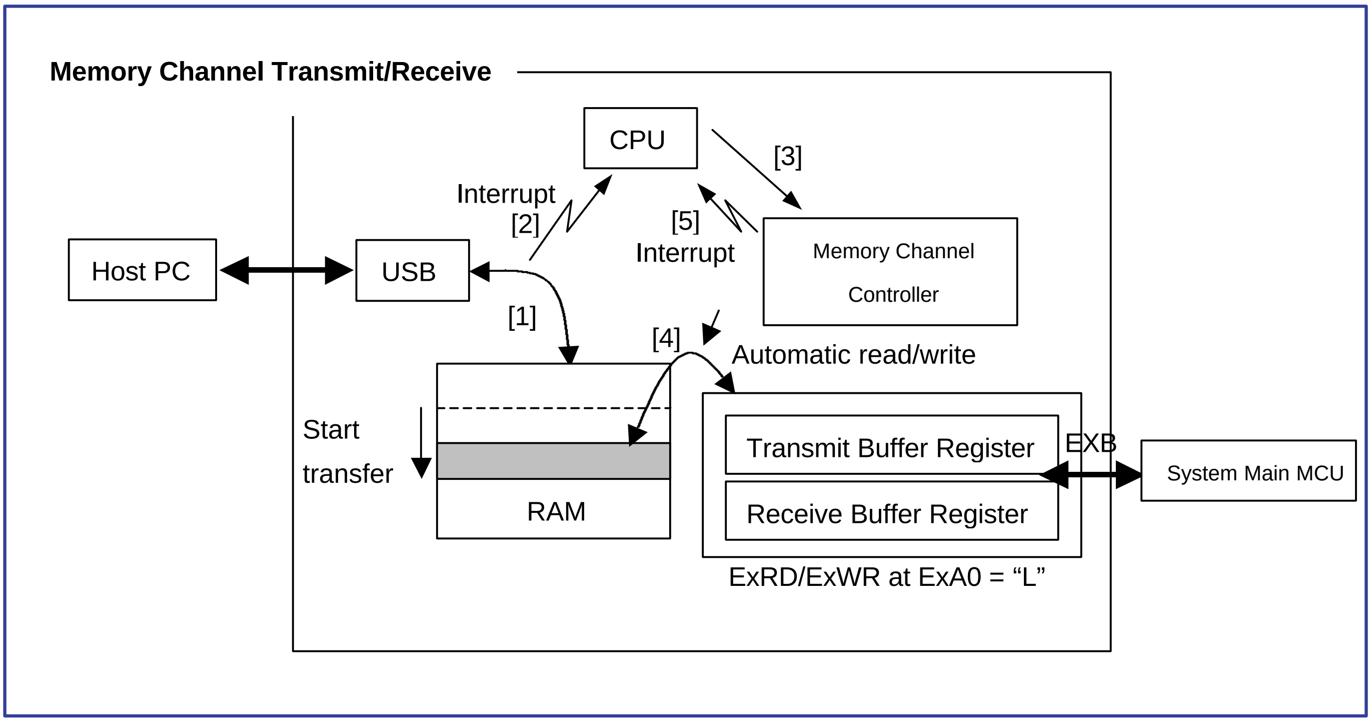


Figure 2.4 (1) Example of EXB Transfer of USB Data

Figure 2.4 (2) shows the RAM expansion operation for memory channel receive data.

The memory channel controller writes the receive data in the receive buffer to the RAM area byte-by-byte. At this time, the start address set in the memory address counter is incremented by 1 for each byte transferred. When the memory address counter reaches the end address register value plus 1, 38K0/38K2 generates a memory channel interrupt (reverse process is executed for memory channel transmit operation).

In this manner, the start and end addresses for the transfer, as well as the transfer initiations must be set before executing the memory channel mode.

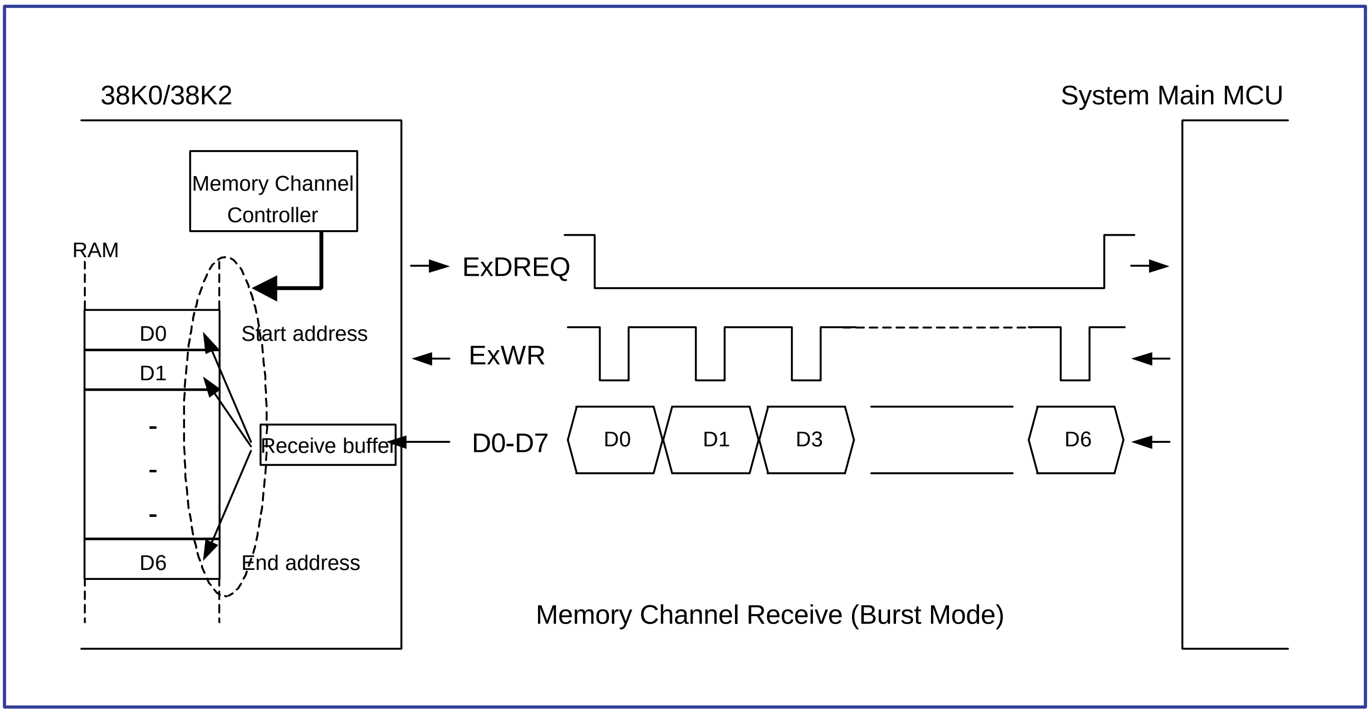


Figure 2.4 (2) RAM Expansion Operation for Memory Channel Receive Data

In the memory channel mode transfer, 38K0/38K2 outputs a request with the ExDREQ signal and the system main MCU returns a DMA response through ExDACK (it is not always necessary for the system main CPU to comply with the DMA.)

Figure 2.4 (3) shows the operational waveform when a memory channel transfer occurs in DMA format.

The DMA-type EXB operates in two modes: the burst mode and the cycle mode. During data transmission in the burst mode, ExDREQ is held at “L” but in the cycle mode, ExDREQ is raised to “H” each time one byte of data is transferred. In both modes, after the last byte of data is transferred, or when an ExTC strobe (terminal count input) is detected, a memory channel interrupt is generated.

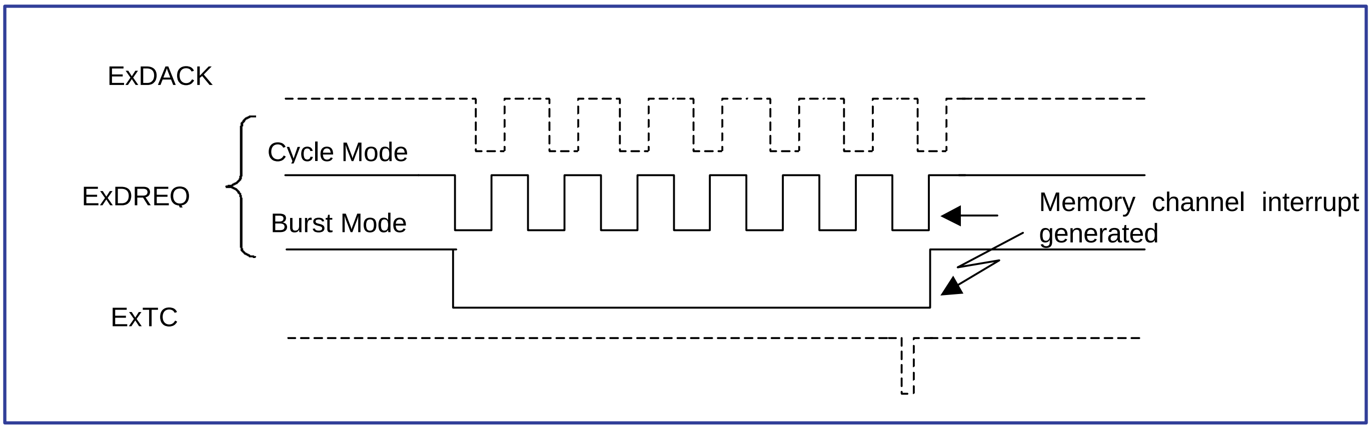


Figure 2.4 (3) Waveform in Burst Mode and Cycle Mode

2.4.1 Limitations on RAM Arbitration and Internal Clock ø

38K0/38K2 Group MCUs access the RAM in 3 blocks: USB, CPU, EXB. However, as there is only one entrance/exit to the RAM, access by the three blocks is controlled through arbitration. The RAM arbitration circuit performs the arbitration process, as diagramed in Figure 2.4.1 (1).

USB and EXB data are temporarily stored in the appropriate block buffer and are accessed according to the RAM access order. However, because the access cycle is synchronized with interrupt clock ϕ, when the transfer process of an unsynchronized block is faster than internal clock ϕ, the RAM access won't be able to keep up with the process speed and the data may be destroyed. To avoid this problem, always access the RAM with an internal clock ϕ of 6MHz or higher (required for USB accesses) and an EXB access size of 5ϕ ntervals or higher per byte, as shown in Figure 2.4.1 (2). Theoretically, EXB in these conditions can reach a transfer rate of 1.2Mbytes/s (1/5 of ϕ = 6MHz).

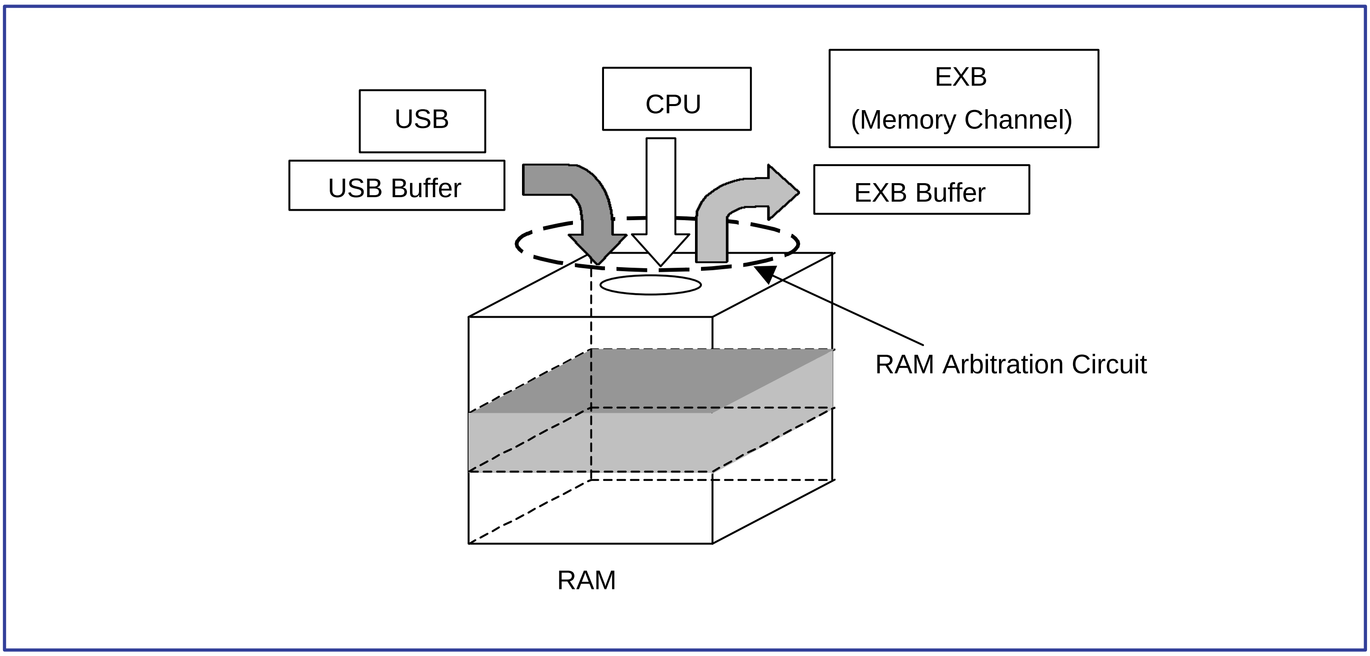


Figure 2.4.1 (1) RAM Arbitration Circuit Image

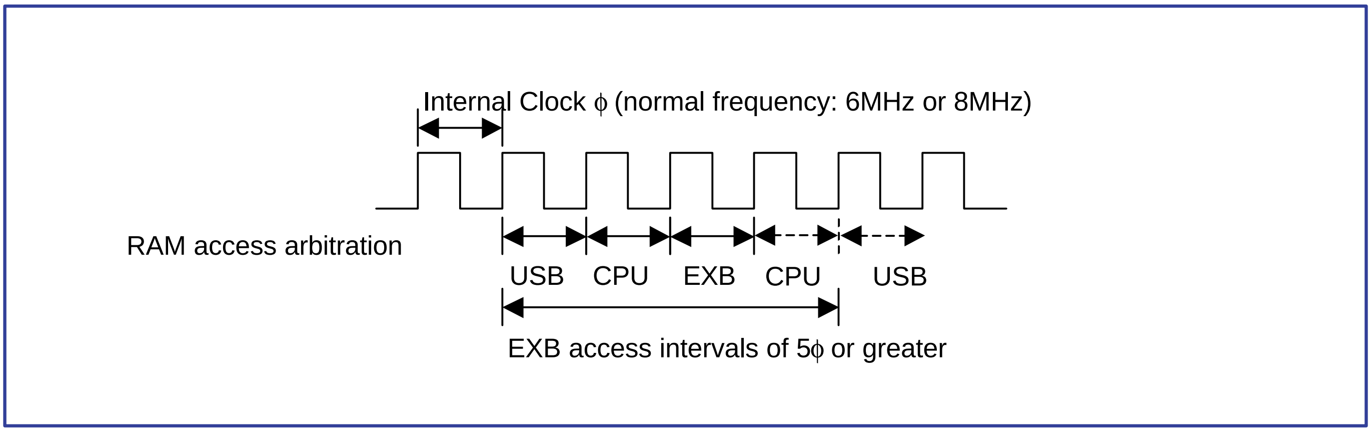

Internal Clock ϕ (normal frequency: 6MHz or 8MHz)
RAM access arbitration
USB
CPU
EXB
CPU
USB
EXB access intervals of 5ϕ or greater

Figure 2.4.1 (2) EXB Access Intervals

●Limitations of Internal Clockf for USB Data Transfer

The USB (SIE: serial interface engine) request is synchronized with internal clockϕ and is sent to the RAM arbitration circuit. The write-access to the RAM is then completed and a one-clock ACK is returned to the USB (SIE). The next write-access to the USB RAM occurs after the acknowledge. In other words, as shown in Figure 2.4.1 (3), a write-access to the USB RAM requires up to 3 internal clockϕ intervals.

38K0/38K2 processes transfers in byte units for each of the 3 blocks in order to provide a baud rate of 12MHz/8 = 1.5M bytes/s for USB transfers. Therefore, internal clockϕ must be set to three times the USB transfer baud rate, or ≥ 1.5 x 3 = 4.5MHz. Consequently, the choice of internal clockϕ frequency is limited to 6MHz or 8MHz for USB functions.

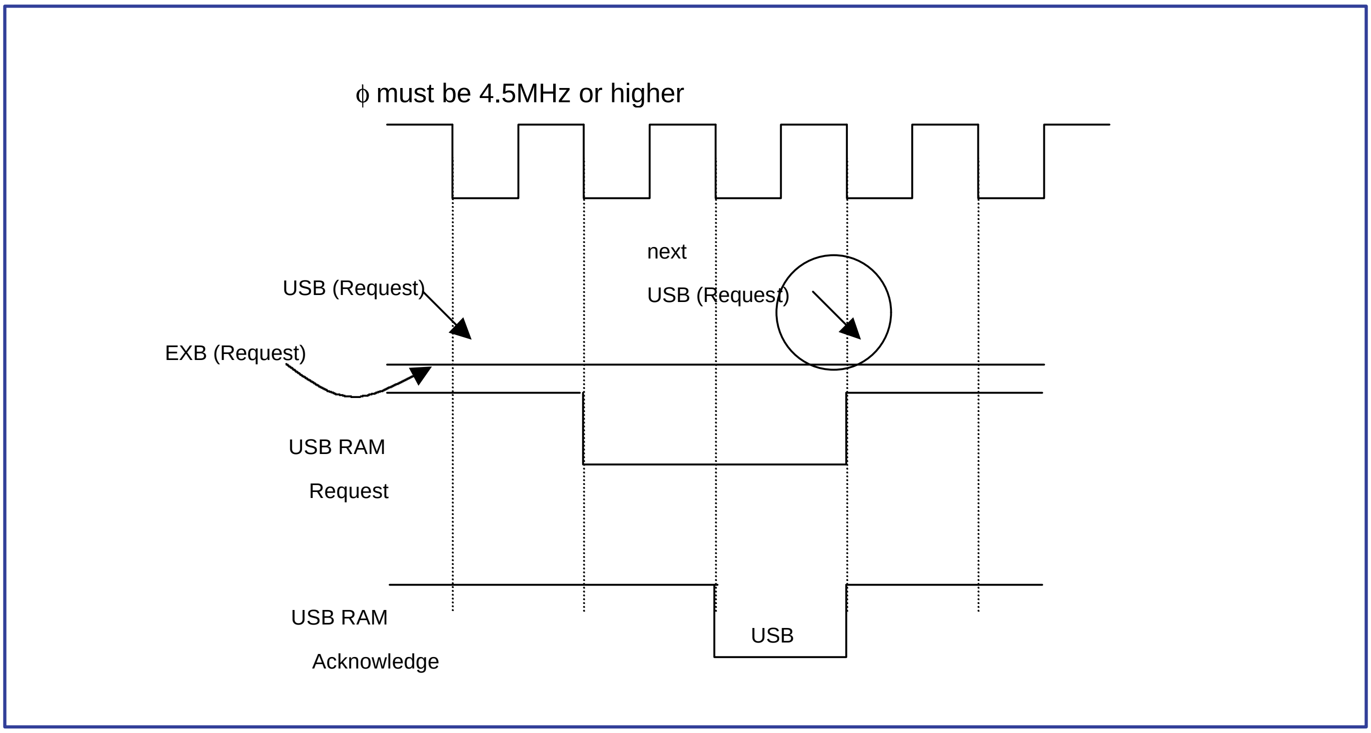
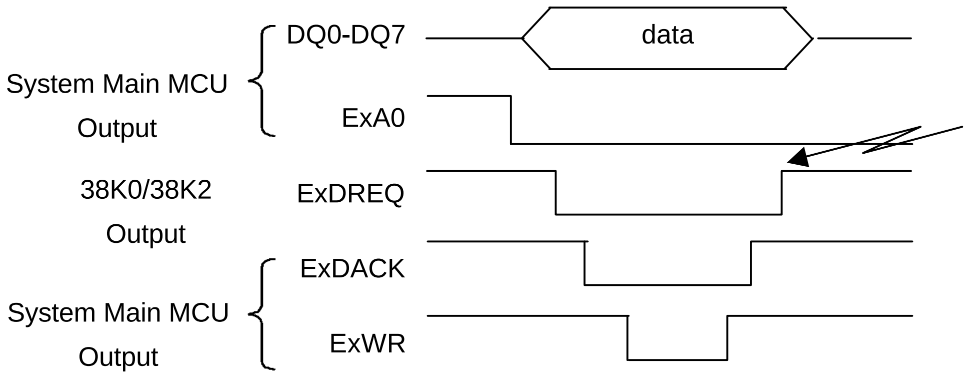


Figure 2.4.1 (3) Minimum Frequency for Internal Clockϕ When Using only USB

2.4.2 Memory Channel Receive Mode

In the memory channel receive mode, the memory channel controller automatically assigns the data received from the system main MCU to the RAM area, and 38K0/38K2 generates a memory channel operation completed interrupt when all data is transferred.

As shown in Figure 2.4.2 (1), DMA format can be used for memory channel transfers. If 38K0/38K2 detects the rising edge of ExWR when ExA0 = “L”, ExDREQ = “L” and ExDACK = “L”, it temporarily latches one byte of DQ0 – DQ7 to the receive buffer register and then writes the data to the RAM in the order determined by the arbitration circuit. 38K0/38K2 repeats this operation several times until all data is transferred, and then generates a memory channel operation completed interrupt.



The memory channel interrupt is generated after ExDREQ goes high and the last receive data is stored in the RAM buffer.

Figure 2.4.2 (1) Memory Channel Receive Waveform

The RAM area for the memory channel receive operation is specified by setting the memory address counter and end address counter. The memory address counter indicates the start (first) address and the end address counter indicates the end (last) address of the receive area. Both counters are 11-bit registers and can be set within the following RAM area ranges: Mask version = 0040H to 043FH, Flash version = 0040H to 07FFH.

Figure 2.4.2 (2) shows the relation between the two registers. After the RAM transfer areas have been set for both registers and a memory channel receive operation is executed, the memory address counter is incremented by 1 for each byte of data transferred. 38K0/38K2 repeats this operation until it reaches the end address counter value plus 1, then completes the operation and generates a memory channel interrupt.

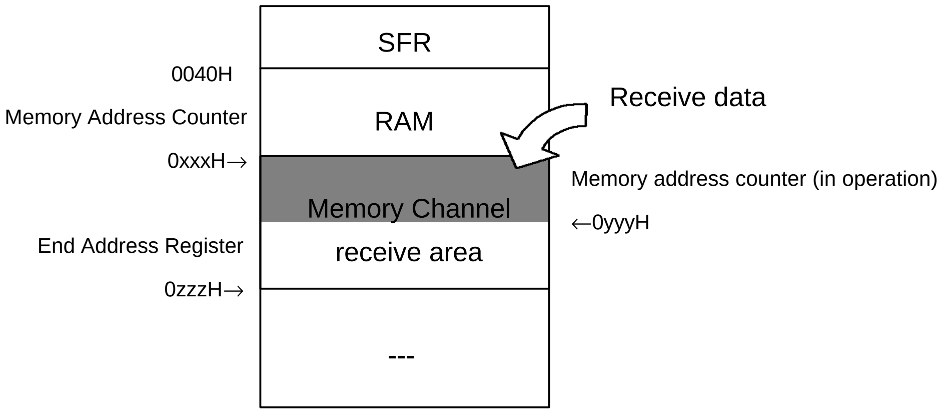
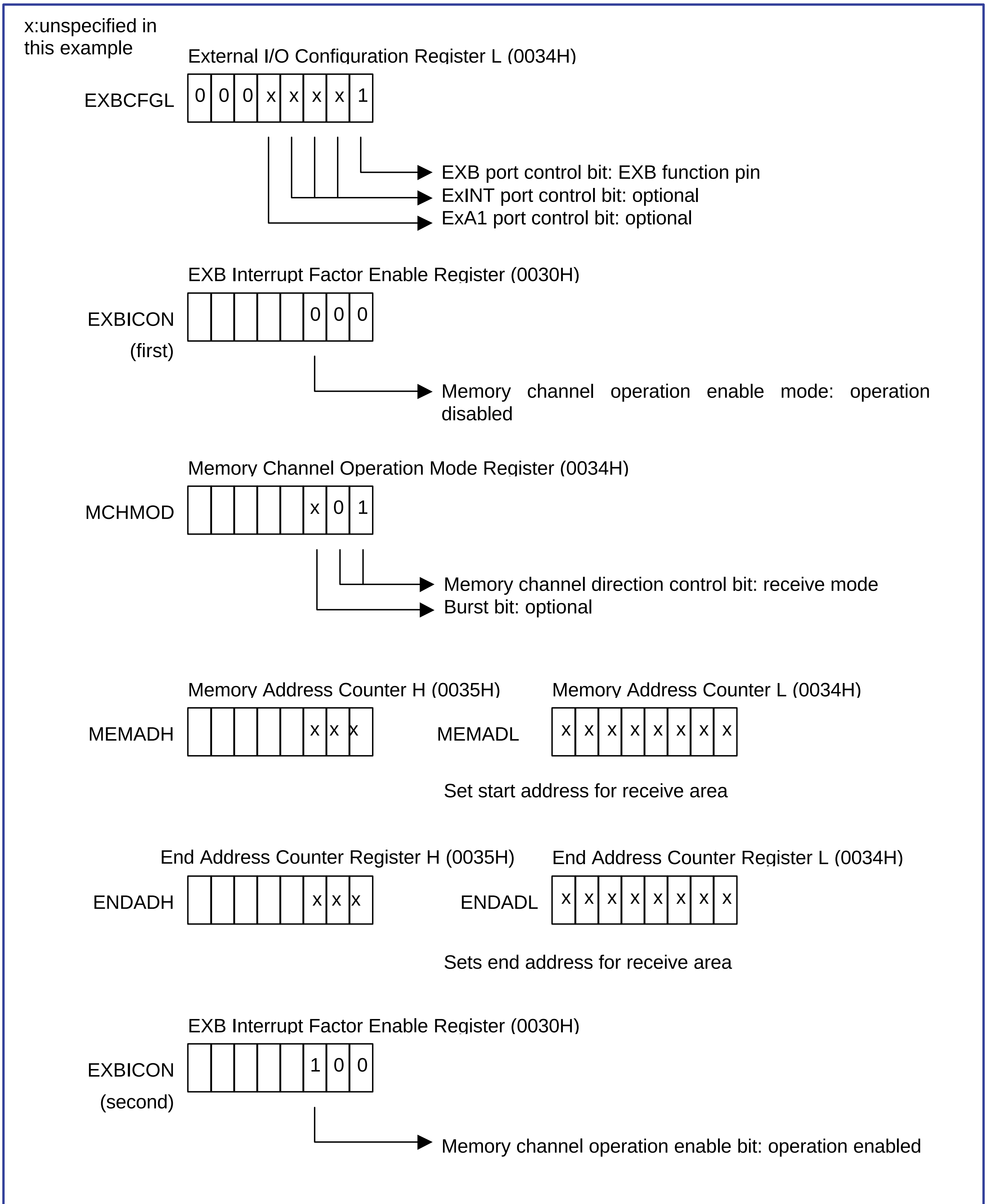


Figure 2.4.2 (2) Memory Channel Address Settings

Figures 2.4.2 (2) and (3) show the settings for memory channel mode registers.


x:unspecified in this example
External I/O Configuration Register L (0034H)
EXBCFGL
0 0 0 x x x x 1
EXB port control bit: EXB function pin
ExINT port control bit: optional
ExA1 port control bit: optional
EXB Interrupt Factor Enable Register (0030H)
EXBICON (first)
0 0 0
Memory channel operation enable mode: operation disabled
Memory Channel Operation Mode Register (0034H)
MCHMOD
x 0 1
Memory channel direction control bit: receive mode
Burst bit: optional
Memory Address Counter H (0035H)
MEMADH
x x x
Memory Address Counter L (0034H)
MEMADL
x x x x x x x x
Set start address for receive area
End Address Counter Register H (0035H)
ENDADH
x x x
End Address Counter Register L (0034H)
ENDADL
x x x x x x x x
Sets end address for receive area
EXB Interrupt Factor Enable Register (0030H)
EXBICON (second)
1 0 0
Memory channel operation enable bit: operation enabled

[Continued. on next page]

Figure 2.4.2 (2) Memory Channel Receive Mode Registers

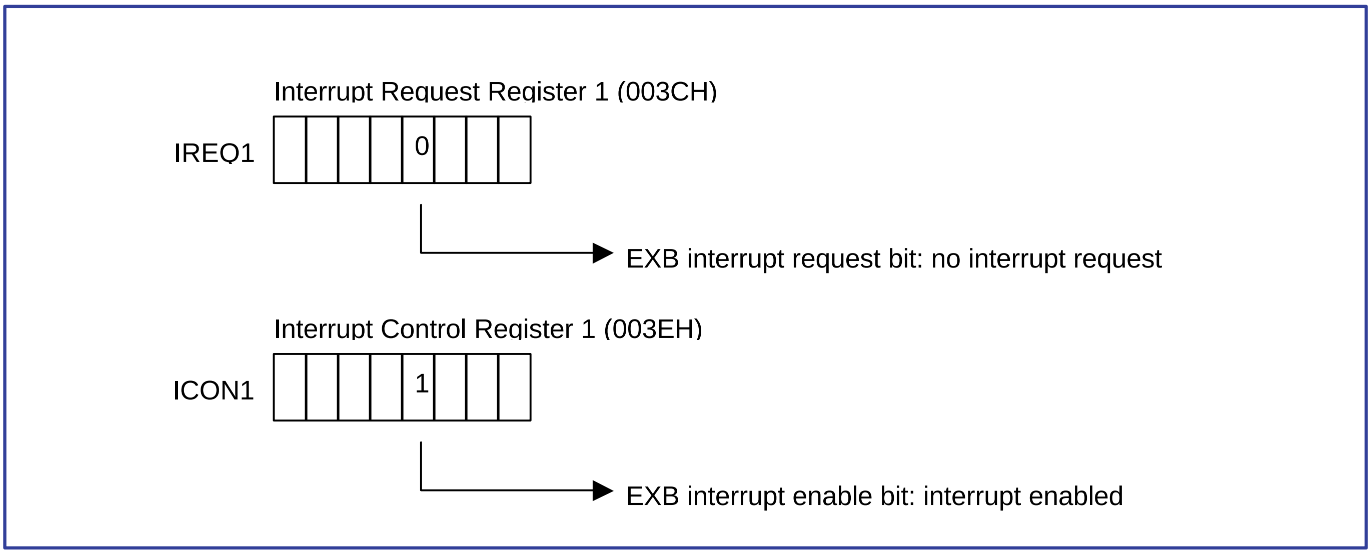
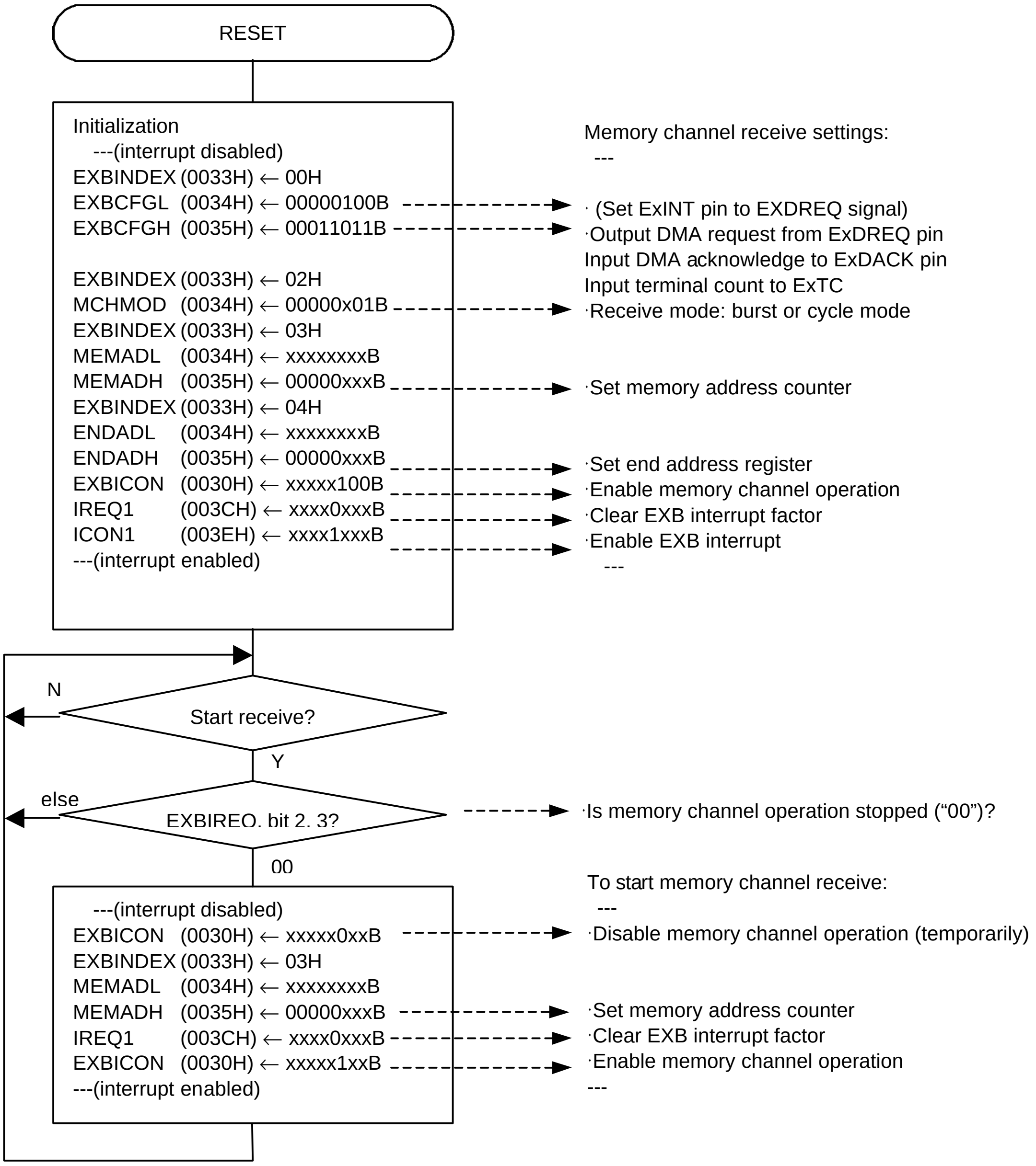

Interrupt Request Register 1 (003CH)
IREQ1
0
EXB interrupt request bit: no interrupt request
Interrupt Control Register 1 (003EH)
ICON1
1
EXB interrupt enable bit: interrupt enabled

Figure 2.4.2 (3) Memory Channel Receive Mode Registers

Figures 2.4.2 (4) and (5) show a control example in the memory channel receive mode.



```
Initialization
   ---(interrupt disabled)
EXBINDEX (0033H) ← 00H
EXBCFGL  (0034H) ← 00000100B
EXBCFGH  (0035H) ← 00011011B

EXBINDEX (0033H) ← 02H
MCHMOD   (0034H) ← 00000x01B
EXBINDEX (0033H) ← 03H
MEMADL   (0034H) ← xxxxxxxxB
MEMADH   (0035H) ← 00000xxxB
EXBINDEX (0033H) ← 04H
ENDADL   (0034H) ← xxxxxxxxB
ENDADH   (0035H) ← 00000xxxB
EXBICON  (0030H) ← xxxxx100B
IREQ1    (003CH) ← xxxx0xxxB
ICON1    (003EH) ← xxxx1xxxB
---(interrupt enabled)
```

Memory channel receive settings:

· (Set ExINT pin to EXDREQ signal)
·Output DMA request from ExDREQ pin
Input DMA acknowledge to ExDACK pin
Input terminal count to ExTC
·Receive mode: burst or cycle mode
·Set memory address counter
·Set end address register
·Enable memory channel operation
·Clear EXB interrupt factor
·Enable EXB interrupt

·Is memory channel operation stopped (“00”)?

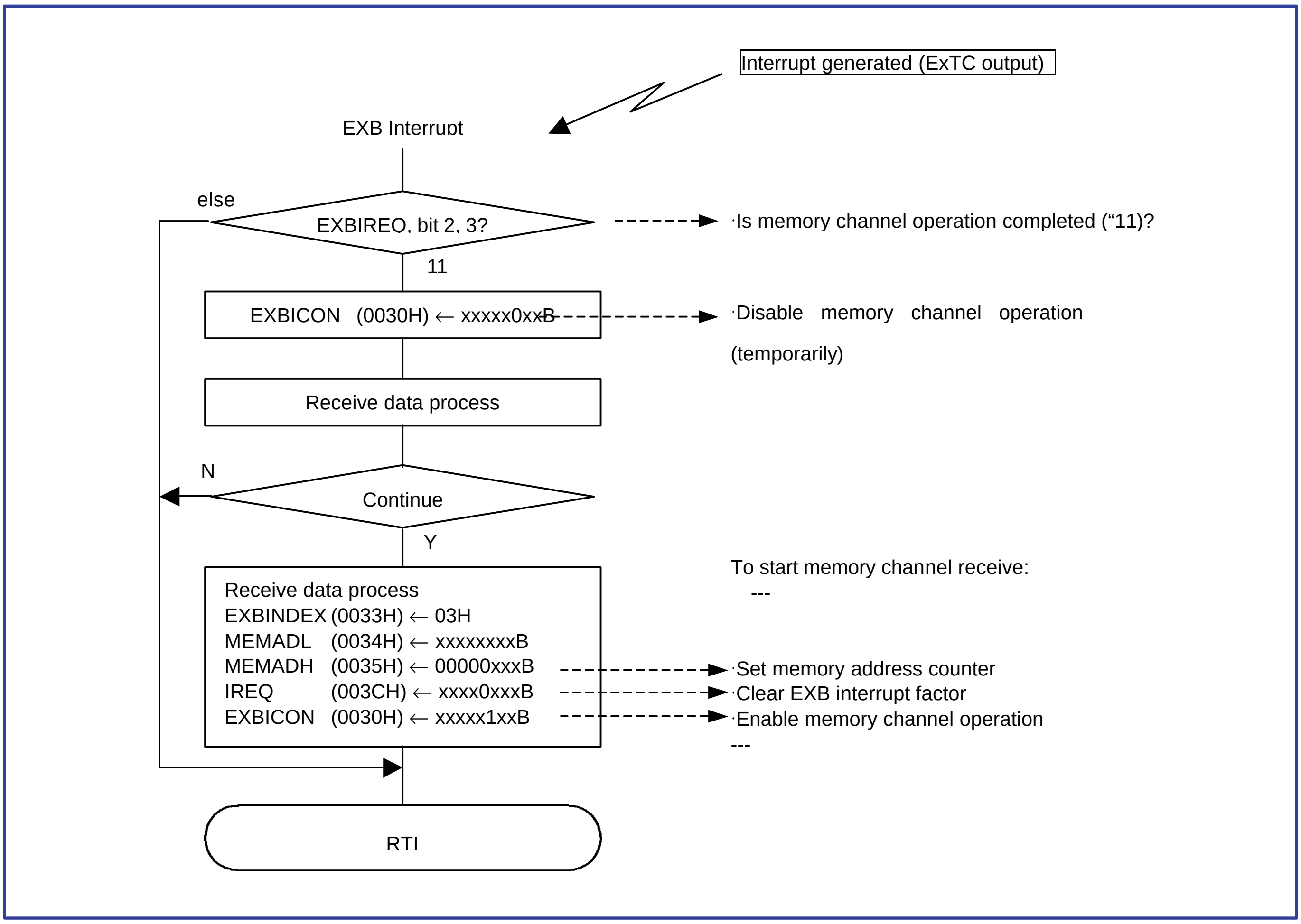
```
   ---(interrupt disabled)
EXBICON  (0030H) ← xxxxx0xxB
EXBINDEX (0033H) ← 03H
MEMADL   (0034H) ← xxxxxxxxB
MEMADH   (0035H) ← 00000xxxB
IREQ1    (003CH) ← xxxx0xxxB
EXBICON  (0030H) ← xxxxx1xxB
---(interrupt enabled)
```

To start memory channel receive:

·Disable memory channel operation (temporarily)
·Set memory address counter
·Clear EXB interrupt factor
·Enable memory channel operation

[Continued on next page]

Figure 2.4.2 (4) Control Example in Memory Channel Receive Mode


Interrupt generated (ExTC output)
EXB Interrupt
else
EXBIREQ, bit 2, 3?
·Is memory channel operation completed (“11)?
11
EXBICON (0030H) ← xxxxx0xxB
·Disable memory channel operation (temporarily)
Receive data process
N
Continue
Y
To start memory channel receive:

Receive data process
EXBINDEX (0033H) ← 03H
MEMADL (0034H) ← xxxxxxxxB
MEMADH (0035H) ← 00000xxxB
IREQ (003CH) ← xxxx0xxxB
EXBICON (0030H) ← xxxxx1xxB
·Set memory address counter
·Clear EXB interrupt factor
·Enable memory channel operation

RTI

Figure 2.4.2 (5) Control Example in Memory Channel Receive Mode

2.4.3 Memory Channel Transmit Mode

In the memory channel transmit mode, the memory channel controller automatically sends multiple data from the RAM area to the system main MCU, and 38K0/38K2 generates a memory channel operation completed interrupt when all data has been transferred.

As shown in Figure 2.4.3 (1), DMA format can be used for memory channel transfers. If 38K0/38K2 detects a falling edge of ExRD when ExAO = "L", ExDREQ = "L" and ExDACK = "L", the data stored in the transmit buffer register up to that point is output to DQ0 to DQ7 byte-by-byte. 38K0/38K2 repeats this operation and, when all data is transferred, generates a memory channel operation completed interrupt.

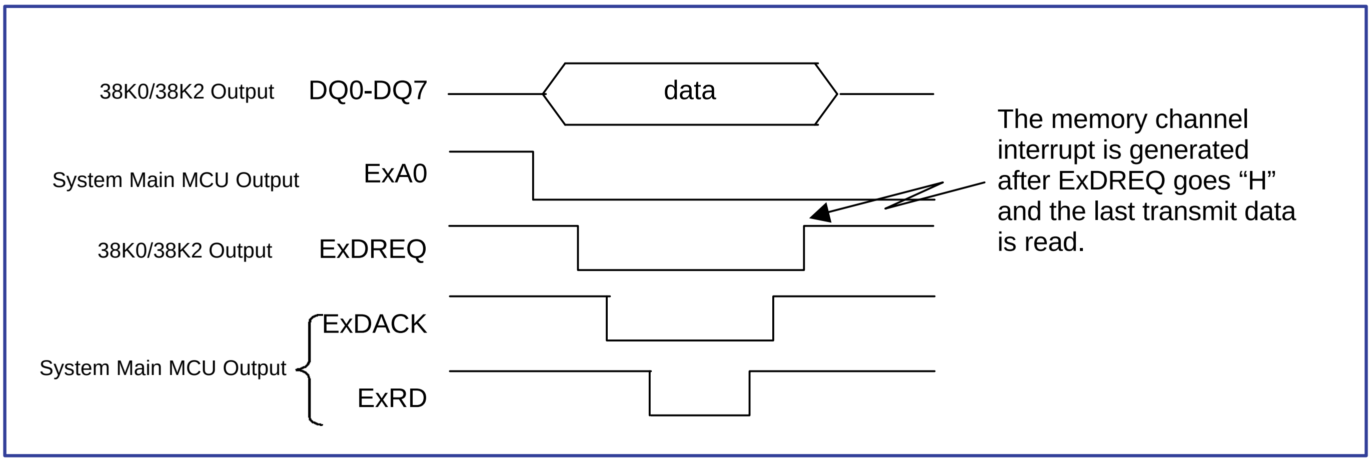


Figure 2.4.3 (1) Memory Channel Transmit Waveform

The RAM area for the memory channel transmit operation is specified by setting the memory address counter and end address counter. The memory address counter indicates the start address and the end address counter indicates the end address of the transmit area. Both counters are 11-bit registers and can be set within the following RAM area ranges: Mask version = 0040H to 043FH, Flash version = 0040H to 07FFH.

Figure 2.4.3 (2) shows the relation between the two registers. After the RAM transfer areas have been set for both registers, and a memory channel transmit operation is executed, the memory address counter is incremented by 1 for each byte of data transferred. 38K0/38K2 repeats this operation until it reaches the end address counter value plus 1, then completes the operation and generates a memory channel interrupt.

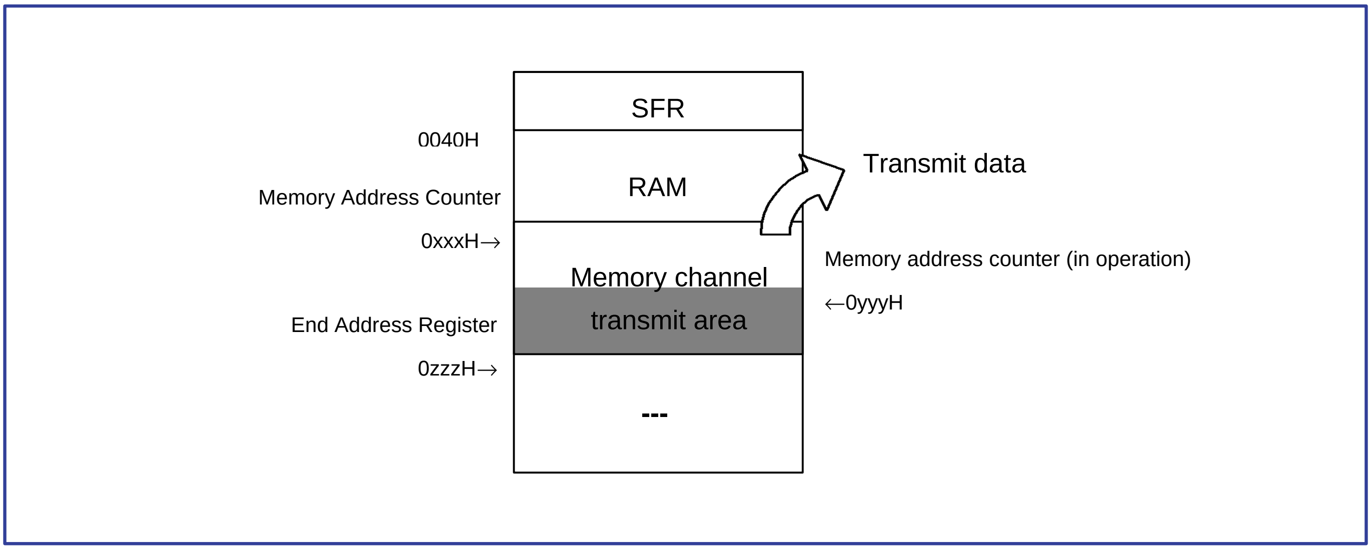
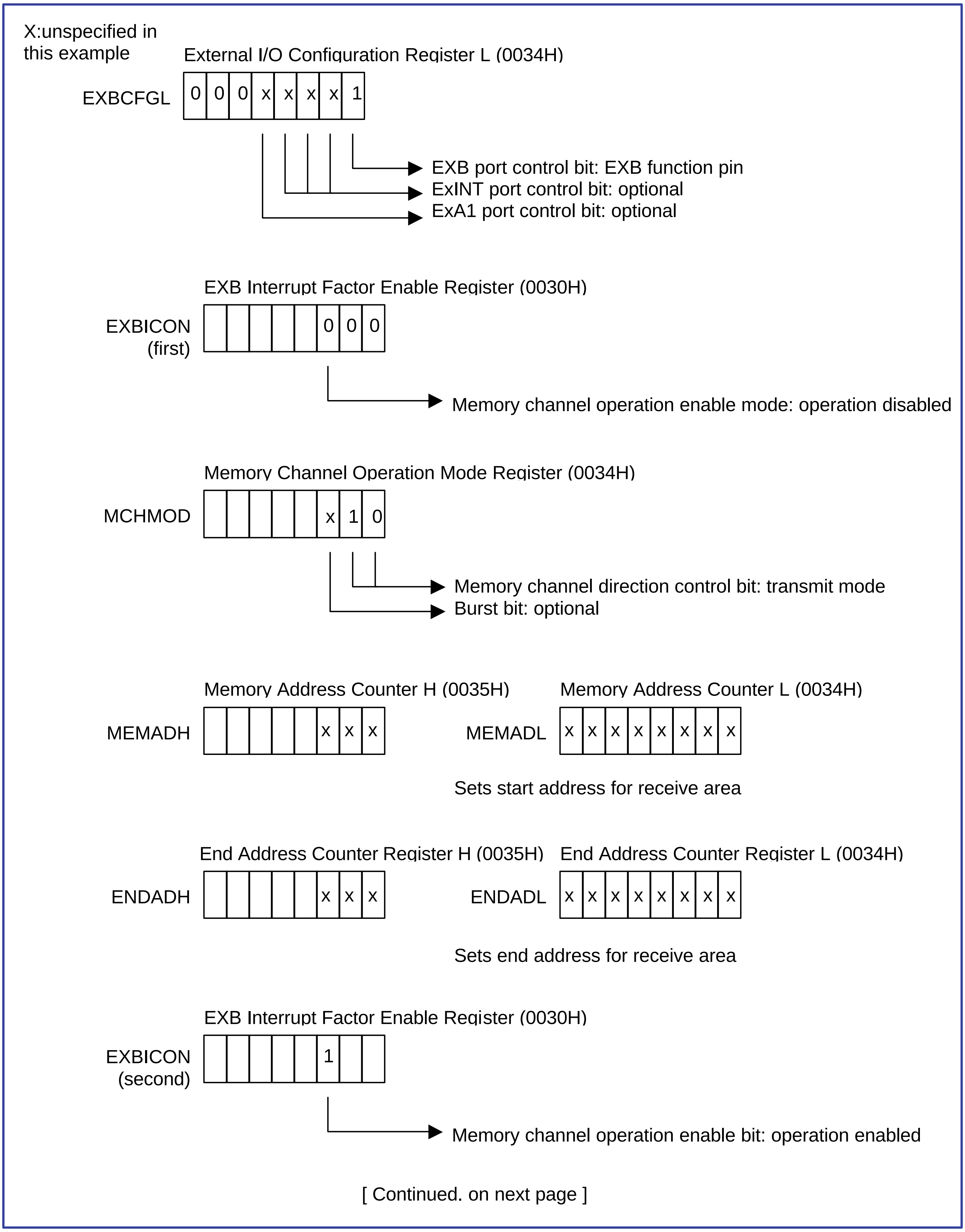


Figure 2.4.3 (2) Memory Channel Address Settings

Figures 2.4.3 (3) and (4) show the settings for memory channel mode registers.


X:unspecified in this example
External I/O Configuration Register L (0034H)
EXBCFGL
0 0 0 x x x x 1
EXB port control bit: EXB function pin
ExINT port control bit: optional
ExA1 port control bit: optional
EXB Interrupt Factor Enable Register (0030H)
EXBICON (first)
0 0 0
Memory channel operation enable mode: operation disabled
Memory Channel Operation Mode Register (0034H)
MCHMOD
x 1 0
Memory channel direction control bit: transmit mode
Burst bit: optional
Memory Address Counter H (0035H)
MEMADH
x x x
Memory Address Counter L (0034H)
MEMADL
x x x x x x x x
Sets start address for receive area
End Address Counter Register H (0035H)
ENDADH
x x x
End Address Counter Register L (0034H)
ENDADL
x x x x x x x x
Sets end address for receive area
EXB Interrupt Factor Enable Register (0030H)
EXBICON (second)
1
Memory channel operation enable bit: operation enabled

[Continued. on next page]

Figure 2.4.3 (3) Memory Channel Transmit Mode Registers

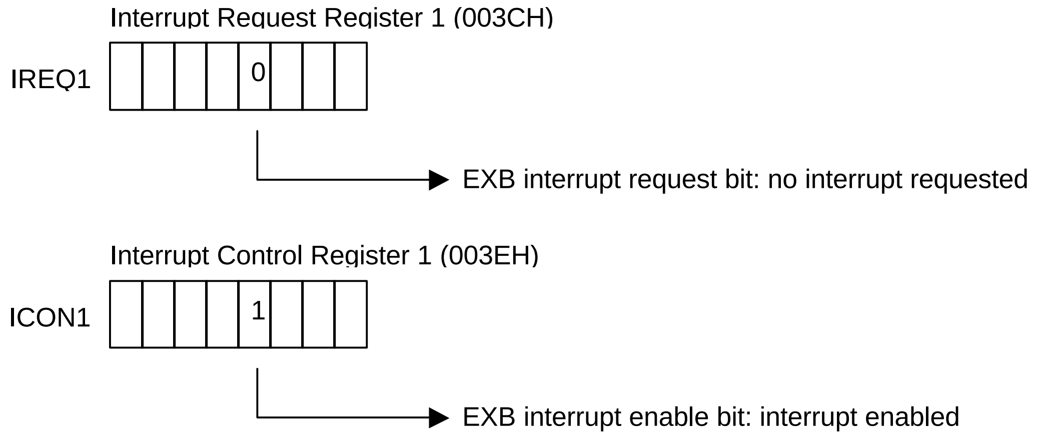
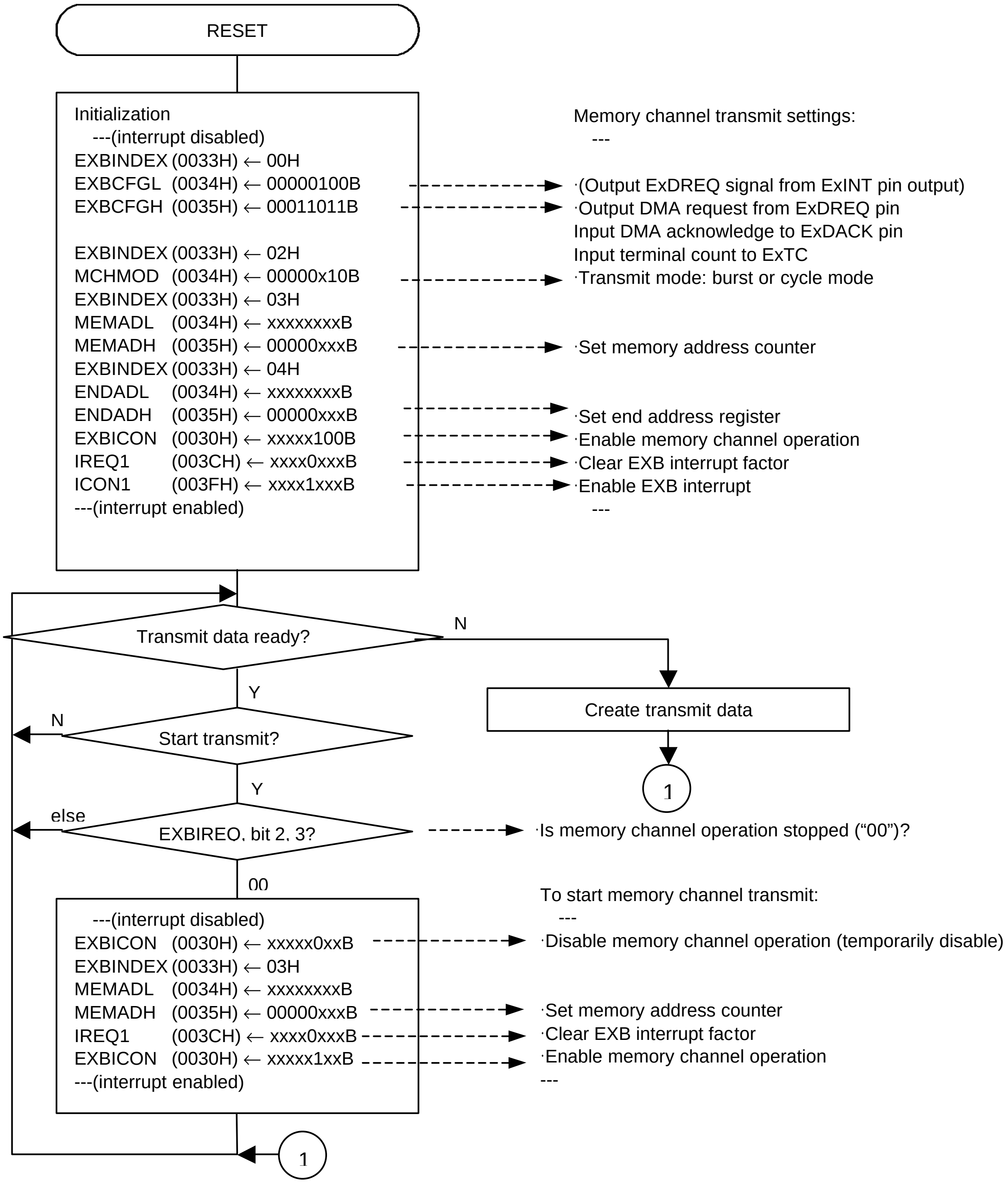

Interrupt Request Register 1 (003CH)
IREQ1
0
EXB interrupt request bit: no interrupt requested
Interrupt Control Register 1 (003EH)
ICON1
1
EXB interrupt enable bit: interrupt enabled

Figure 2.4.3 (4) Memory Channel Transmit Mode Registers

Figures 2.4.3 (5) and (6) show a control example in the memory channel transmit mode


RESET
Initialization
---(interrupt disabled)
EXBINDEX (0033H) ← 00H
EXBCFGL (0034H) ← 00000100B
EXBCFGH (0035H) ← 00011011B
EXBINDEX (0033H) ← 02H
MCHMOD (0034H) ← 00000x10B
EXBINDEX (0033H) ← 03H
MEMADL (0034H) ← xxxxxxxxB
MEMADH (0035H) ← 00000xxxB
EXBINDEX (0033H) ← 04H
ENDADL (0034H) ← xxxxxxxxB
ENDADH (0035H) ← 00000xxxB
EXBICON (0030H) ← xxxxx100B
IREQ1 (003CH) ← xxxx0xxxB
ICON1 (003FH) ← xxxx1xxxB
---(interrupt enabled)
Memory channel transmit settings:

·(Output ExDREQ signal from ExINT pin output)
·Output DMA request from ExDREQ pin
Input DMA acknowledge to ExDACK pin
Input terminal count to ExTC
·Transmit mode: burst or cycle mode
·Set memory address counter
·Set end address register
·Enable memory channel operation
·Clear EXB interrupt factor
·Enable EXB interrupt

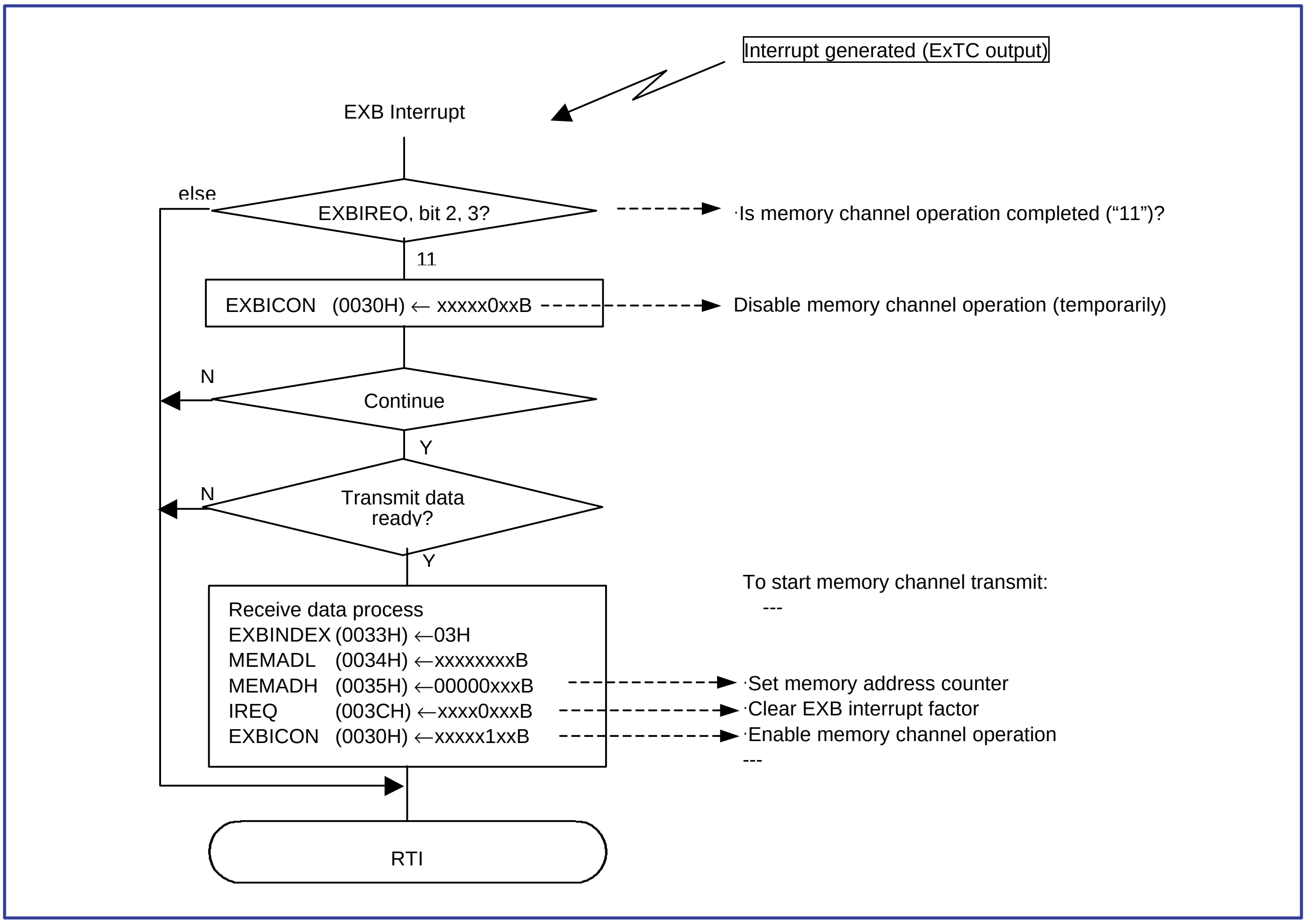
Transmit data ready?
N
Y
Create transmit data
1
N
Start transmit?
Y
else
EXBIREQ, bit 2, 3?
·Is memory channel operation stopped (“00”)?
00
To start memory channel transmit:

---(interrupt disabled)
EXBICON (0030H) ← xxxxx0xxB
EXBINDEX (0033H) ← 03H
MEMADL (0034H) ← xxxxxxxxB
MEMADH (0035H) ← 00000xxxB
IREQ1 (003CH) ← xxxx0xxxB
EXBICON (0030H) ← xxxxx1xxB
---(interrupt enabled)
·Disable memory channel operation (temporarily disable)
·Set memory address counter
·Clear EXB interrupt factor
·Enable memory channel operation

1

[Continued on next page]

Figure 2.4.3 (5) Control Example in Memory Channel Transmit Mode


Interrupt generated (ExTC output)
EXB Interrupt
else
EXBIREQ, bit 2, 3?
·Is memory channel operation completed (“11”)?
11
EXBICON (0030H) ← xxxxx0xxB
Disable memory channel operation (temporarily)
N
Continue
Y
N
Transmit data ready?
Y
To start memory channel transmit:

Receive data process
EXBINDEX (0033H) ←03H
MEMADL (0034H) ←xxxxxxxxB
MEMADH (0035H) ←00000xxxB
IREQ (003CH) ←xxxx0xxxB
EXBICON (0030H) ←xxxxx1xxB
·Set memory address counter
·Clear EXB interrupt factor
·Enable memory channel operation

RTI

Figure 2.4.3 (6) Control Example in Memory Channel Transmit Mode

2.5 EXBIREQ (Status) Read Function

As described in Table 2.5 (1), the ExA0 pin must be set by the system main MCU for both CPU channel mode and memory channel mode for EXB transfers. Data may be destroyed or lost by 38K0/38K2 if the system main MCU performs ExRD/ExWR operations under incorrect settings.

To confirm a read/write request, the EXB is equipped with the EXBIREQ status read function which reads the status register, as described by Table 2.5 (2) and Figure 2.5. The status register is the same register as the EXB interrupt factor register [EXBIREQ] (0031H). If 38K0/38K2 detects the ExRD signal when ExCS = “L”, the EXB interrupt factor register is output to the external bus as the status register.

This function enables the system main MCU to determine the current operation mode of 38K0/38K2 after detecting the ExINT signal.

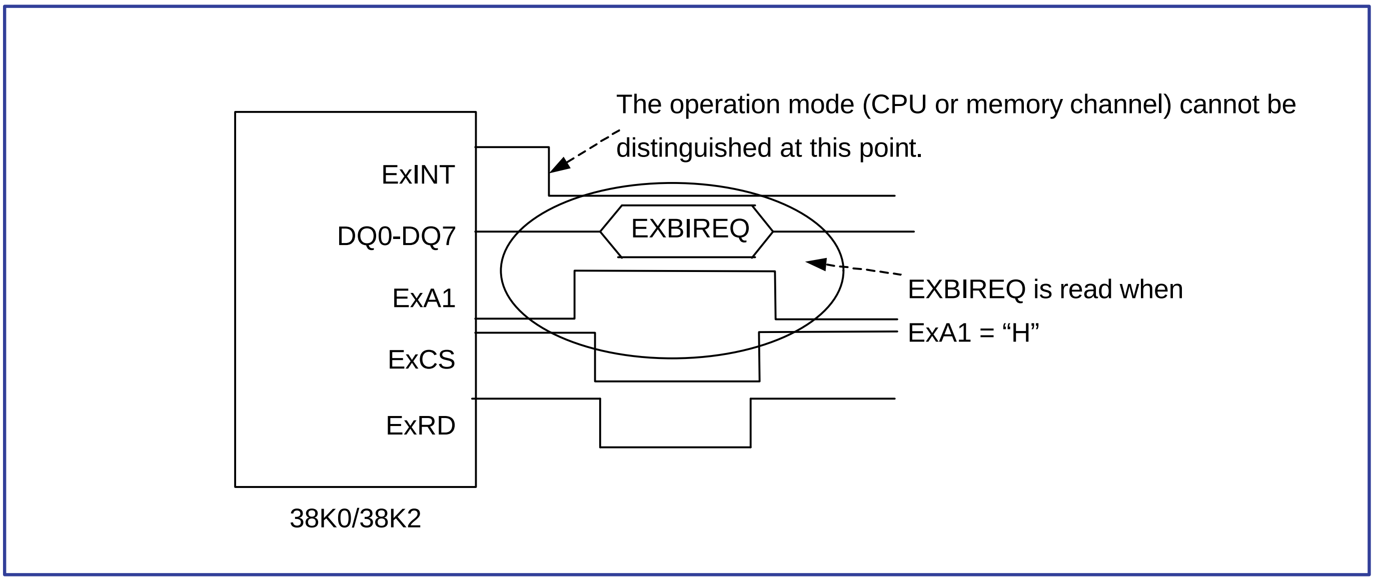
Table 2.5 (1) Correspondence of ExA0 Pin and Operation Mode

ExA0	Mode
L	Memory channel mode
H	CPU channel mode

Incorrect mode setting may result in faulty operations.

Table 2.5 (2) Correspondence of ExA1 Pin and DQ0 to DQ7

ExA1	DQ0-DQ7
L	Transmit/receive data
H	EXB interrupt factor register



The operation mode (CPU or memory channel) cannot be distinguished at this point.

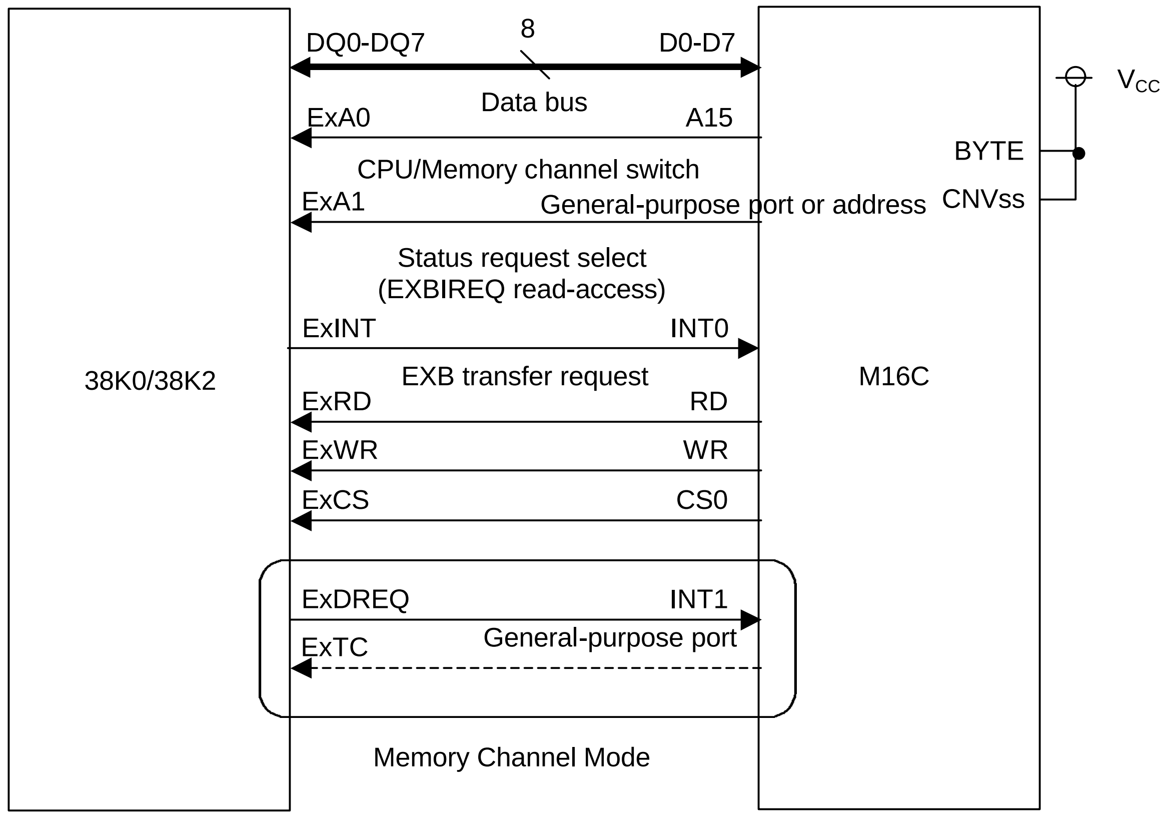
EXBIREQ is read when ExA1 = “H”

Figure 2.5 ExA1 Read-Access of EXBIREQ

Chapter 3
Appendix

3.1 Connection Example

Figure 3.1 (1) shows a connection example for EXB and M16C, a RENESAS 16-bit MCU.



● Required pins   ()optional

Pin Name		CPU Channel	Memory Channel
DQ0-DQ7	Data line	○	○
ExA0	Address *	(○)	(○)
ExA1	Status read select	(○)	×
ExINT	EXB interrupt output	(○)	(○)
ExRD	Read	(○transmit)	(○)
ExWR	Write	(○receive)	(○)
ExCS	Chip select	○	(○)
ExDREQ	DMA request	×	○
ExDACK	DMA acknowledge	×	(○)
ExTC	Terminal count	×	(○)

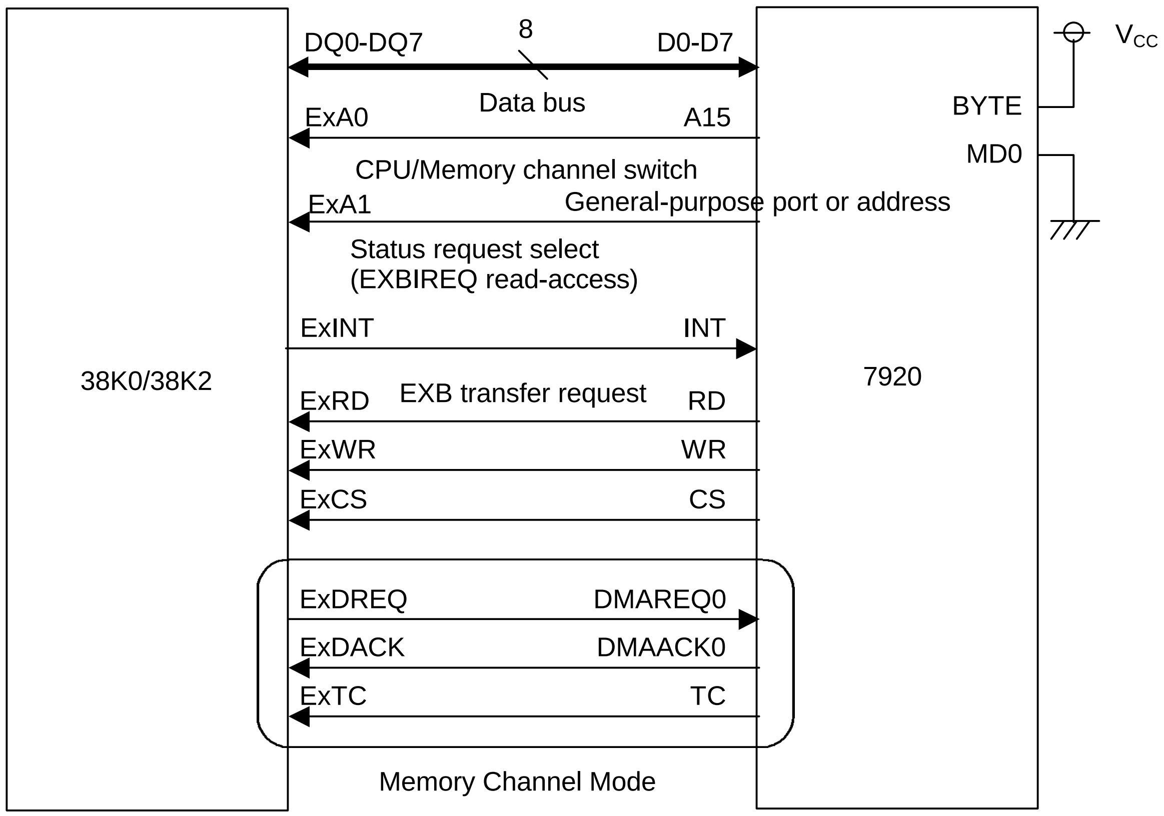
Required when not using ExDACK

The usage conditions for these pins vary according to system and firmware.

*When using ExA0 for only one mode, fix to “H” or “L”, accordingly.

Figure 3.2 (1) M16C Connection Example

Figure 3.1 (2) shows a connection example for EXB and 7920, a RENESAS 16-bit MCU.



● Required pins ()optional

Pin Name		CPU Channel	Memory Channel
DQ0-DQ7	Data line	○	○
ExA0	Address *	(○)	(○)
ExA1	Status read select	(○)	×
ExINT	EXB interrupt output	(○)	(○)
ExRD	Read	(○transmit)	(○)
ExWR	Write	(○receive)	(○)
ExCS	Chip select	○	(○)
ExDREQ	DMA request	×	○
ExDACK	DMA acknowledge	×	(○)
ExTC	Terminal count	×	(○)

Required when not using ExDACK

The usage conditions for these pins vary according to system and firmware

*When using ExA0 for only one mode, fix to “H” or “L”, accordingly.

Figure 3.1 (2) 7920 Connection Example

3.2 USB (Endpoints 01/02/03) to System Main MCU Transfer Control Example

The 38K0/38K2 Group USB function does not include a dedicated FIFO memory. Therefore, data transfers to/from the Host are directly transferred to/from the RAM (a specified USB buffer area). In other words, USB data is handled in the same manner as normal RAM data, but when the data is transferred to/from the system main MCU, special firmware is required to complete the transfer. However, as USB polling in the Host occurs irregularly, the direct transfer rate may be affected for EXB transfer processing using firmware.

Figure 3.2 shows the basic flow of the USB (Endpoints 01/02/03) to system main MCU transfer using the 38K0/38K2 EXB function. After each endpoint completes a transfer, 38K0/38K2 generates a buffer ready interrupt. This interrupt triggers the firmware to initiate an EXB transfer to/from the system main MCU.

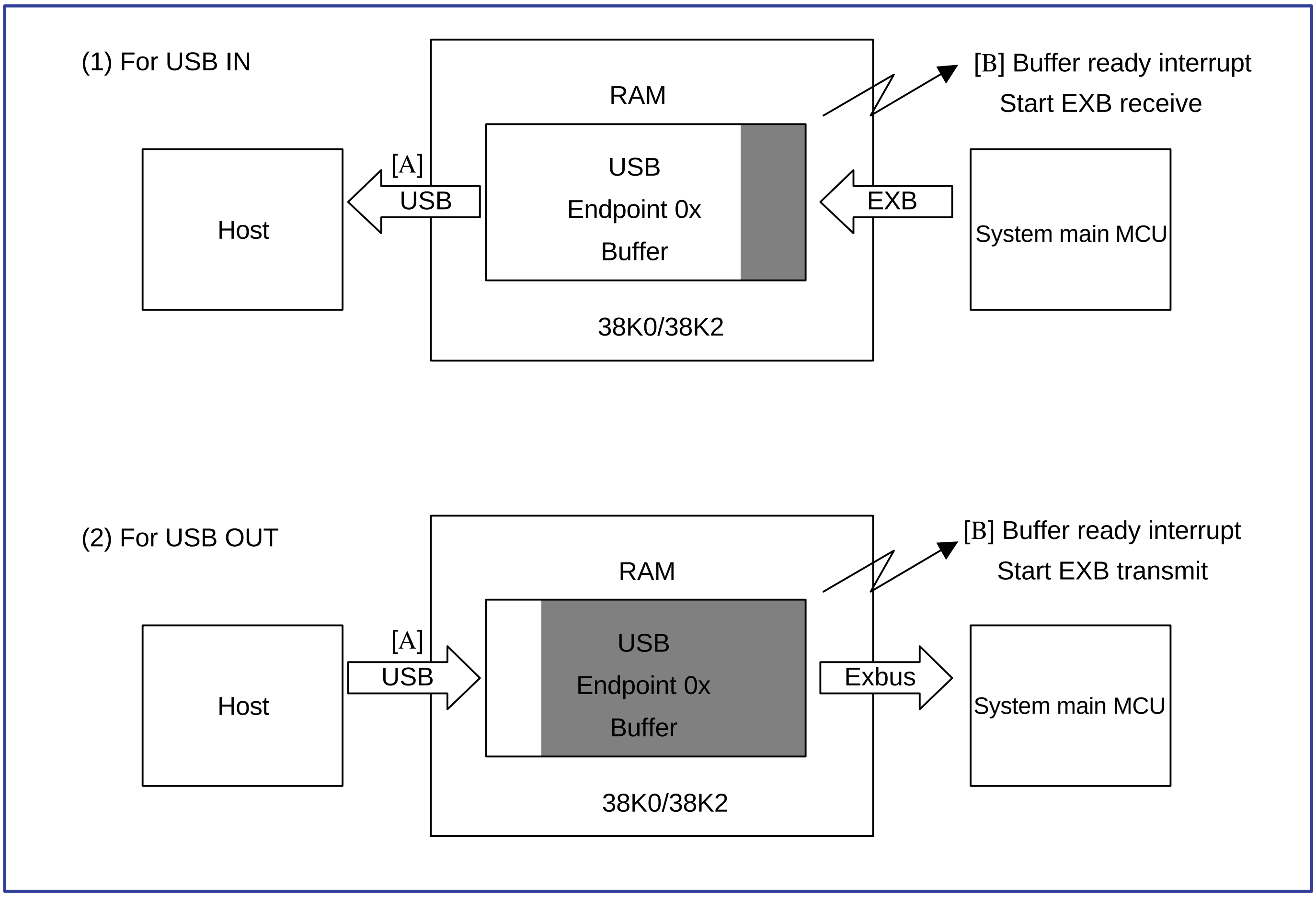


Figure 3.2 Flow for USB (Endpoints 01/02/03) to System Main MCU Data Transfer

(1) USB IN

[A] The data in Endpoint0x (in RAM) is sent to the Host in a USB transfer.

[B] When the Endpoint0x buffer is empty, an Endpointx buffer ready interrupt is generated. The EXB confirms this event and then receives data from the system main MCU.

(2) USB OUT

[A] Endpoint0x buffer (in RAM) receives data from the Host.

[B] When the Endpoint0x buffer is full, an Endpointx buffer ready interrupt is generated. The EXB confirms this event and sends data to the system main MCU.

3.2.1 USB Buffer Settings

The USB buffer area in the RAM is set in the EP0x buffer area setup register (EP0xBUF)[0FEDH] of each endpoint, as shown in Figure 3.2.1 (1) This register specifies the start address of each endpoint buffer area.

● EPx Buffer Area Setup Register EP0xBUF [0FEDH]

Bit	Bit Name	Function	Reset	R/W
4-0	EP0x start address setup bit (BADD0x)	Sets the setup value X 32 bytes (20H) as the absolute address. Example: 00100 EP0x start address is set to 04H × 20H = 0080H	00000	R/W
7-5	Unused	Set to “0” when writing.	-	-

Figure 3.2.1 (1) EP0x Buffer Area Setup

In addition, up to 2 buffers (64 bytes each) can be set for each endpoint (called “double buffer”). When selecting a double buffer, setup each endpoint, including transfer type, in the EP0x setup register (EP0xCFG) [0019H], as shown in Figure 3.2.1 (2).

● EP0x Setup Register EP0xCFG [0019H]

Bit	Bit Name	Function	Reset	R/W
1-0	Double buffer start address setup bit (BSIZE0x)	00: 8-byte gap 01: 16-byte gap 10: 64-byte gap 11: 128-byte gap	00	R/W
2	Buffer mode setup bit (DBLB0x)	0: single buffer mode 1: double buffer mode	0	R/W
3	Sequence toggle bit clear bit (SQCL0x)	0: toggle bit clear disabled 1: DATA0 at next PID	0	R/W
4	Interrupt toggle mode setup bit (ITMD0x)	0: normal mode 1: continuous toggle mode (interrupt IN only)	0	R/W
5	Transfer direction bit (DIR0x)	0: OUT 1: IN	0	R/W
7-6	Transfer type bit (TYP0x)	00: transfer disabled 01: BULK transfer 10: interrupt transfer	0	R/W

Figure 3.2.1 (2) EP0x Setup Register

(1) Double Buffer Start Address Setup Bits (BSIZE0x)

These two bits are used to set the difference between the start addresses of the first and second buffers (Buffer 0 and Buffer 1) when an endpoint in the RAM is set to the double buffer mode (buffer mode setup bit (bit 2) = “1”).

Figure 3.2.1 (3) shows an example of the EP0x buffer area when the double buffer start address setup bit is set to “11”.

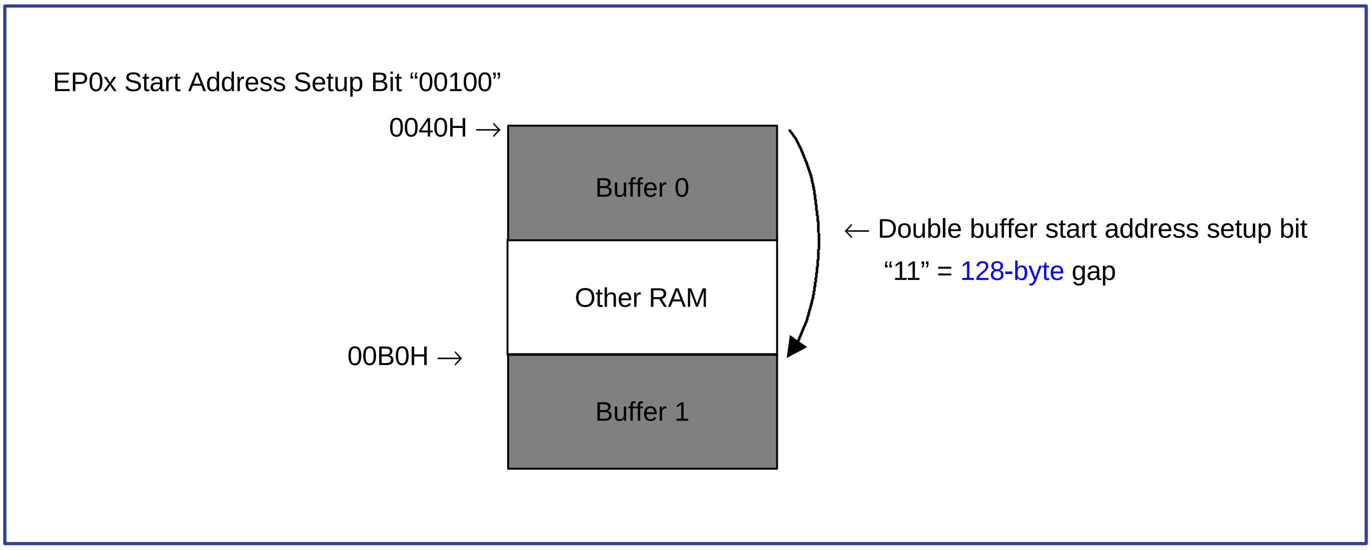

EP0x Start Address Setup Bit “00100”
0040H →
Buffer 0
Other RAM
00B0H →
Buffer 1
← Double buffer start address setup bit
“11” = 128-byte gap

Figure 3.2.1 (3) EP0x Buffer Area Setup

(2) Buffer mode setup bit (DBLD0x)

When this bit is “1”, the corresponding endpoint operates in the double buffer mode.

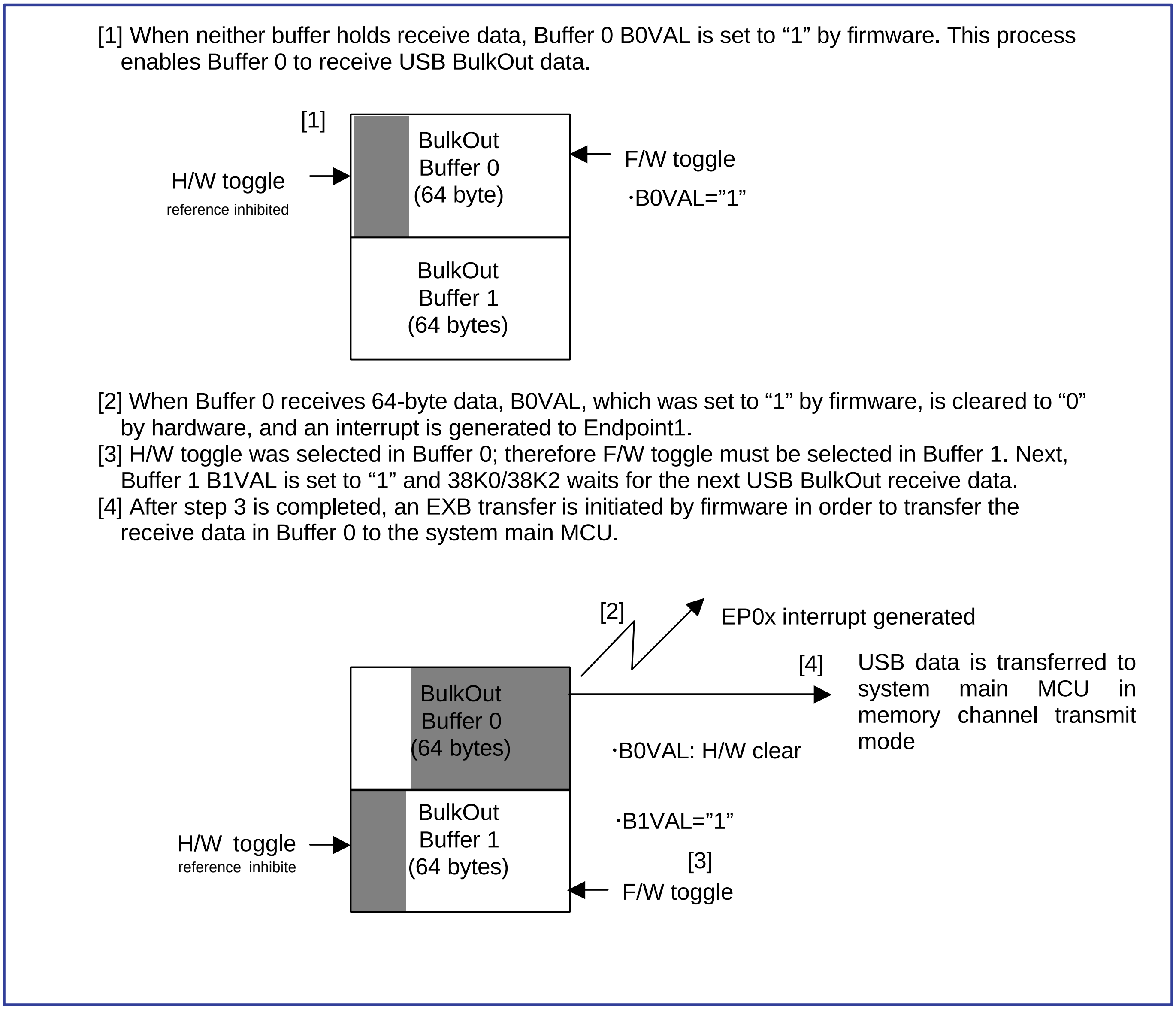
3.2.2 Example Double Buffer Operation

In the double buffer mode, the USB endpoints generate an interrupt for every data transfer to/from Buffer 0 and Buffer 1 (Buffer 0 ready interrupt (B0RDY0x) or Buffer 1 ready interrupt (B0RDY0x) of the EP0x interrupt factor register. 38K0/38K2 confirms the interrupt and then initiates an EXB transfer to/from the system main MCU. In addition, after all data has been transferred to/from the buffers, Buffer 0 enable bit (B0VAL0x) and Buffer 1 enable bit (B1VAL0x) of EP0x control register 2 (EP0xCON2) and EP0x control register 3 (EP0xCON3), respectively, are set to “1”. This process enables new USB data to be written to the same buffers

Figure 3.2.2 shows an example of a BulkOut transfer in the double buffer mode. Note that the access order for Buffer 0 and Buffer 1 is “H/W toggle” (toggle referenced by H/W) followed by “F/W toggle” (toggle referenced by F/W).

Note:

The hardware toggle (next buffer to be accessed by USB) cannot be directly referenced by firmware. Therefore, a firmware toggle that coincides with a hardware toggle must be controlled separately. The hardware toggle is set to “0” (access from Buffer 0) by the sequence toggle bit clear bit (SQCL0x).



[1] When neither buffer holds receive data, Buffer 0 B0VAL is set to “1” by firmware. This process enables Buffer 0 to receive USB BulkOut data.

[2] When Buffer 0 receives 64-byte data, B0VAL, which was set to “1” by firmware, is cleared to “0” by hardware, and an interrupt is generated to Endpoint1.

[3] H/W toggle was selected in Buffer 0; therefore F/W toggle must be selected in Buffer 1. Next, Buffer 1 B1VAL is set to “1” and 38K0/38K2 waits for the next USB BulkOut receive data.

[4] After step 3 is completed, an EXB transfer is initiated by firmware in order to transfer the receive data in Buffer 0 to the system main MCU.

Figure 3.2.2 Double Buffer Operation Example

RENESAS Single-Chip Microcomputer
38K0/38K2 Group
EXB Application Notes VER.1.1
RENESAS TECHNOLOGY CORPORATION.
RENESAS SOLUTIONS CORPORATION.