

To our customers,

---

## Old Company Name in Catalogs and Other Documents

---

On April 1<sup>st</sup>, 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 1<sup>st</sup>, 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.

# 38K2 Group

User's Manual

RENESAS 8-BIT SINGLE-CHIP  
MICROCOMPUTER  
740 FAMILY / 38000 SERIES

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>).

## Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, 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 products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (<http://www.renesas.com> )
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
  - (1) artificial life support devices or systems
  - (2) surgical implantations
  - (3) healthcare intervention (e.g., excision, administration of medication, etc.)
  - (4) any other purposes that pose a direct threat to human lifeRenesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas 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 applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

## General Precautions in the Handling of MPU/MCU Products

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

### 1. Handling of Unused Pins

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

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

### 2. Processing at Power-on

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

- The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.

In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

### 3. Prohibition of Access to Reserved Addresses

Access to reserved addresses is prohibited.

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

### 4. Clock Signals

After applying a reset, only release the reset line after the operating clock signal has become stable.

When switching the clock signal during program execution, wait until the target clock signal has stabilized.

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

### 5. Differences between Products

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

- The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

# BEFORE USING THIS MANUAL

This user's manual consists of the following three chapters. Refer to the chapter appropriate to your conditions, such as hardware design or software development. Chapter 3 also includes necessary information for systems development. You must refer to that chapter.

## 1. Organization

### ● CHAPTER 1 HARDWARE

This chapter describes features of the microcomputer and operation of each peripheral function.

### ● CHAPTER 2 APPLICATION

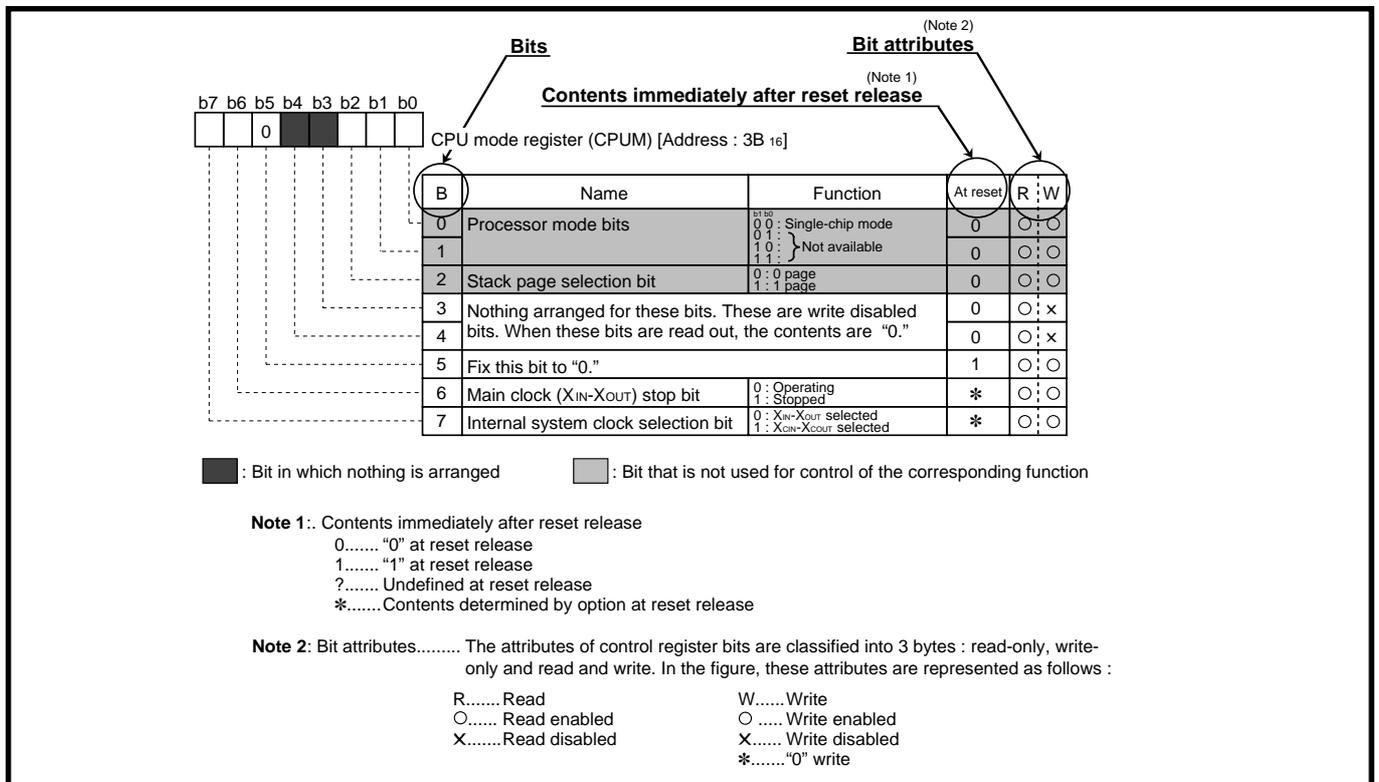
This chapter describes usage and application examples of peripheral functions, based mainly on setting examples of relevant registers.

### ● CHAPTER 3 APPENDIX

This chapter includes necessary information for systems development using the microcomputer, such as the electrical characteristics, the list of registers.

## 2. Structure of register

The figure of each register structure describes its functions, contents at reset, and attributes as follows :



## 3. Supplementation

For details of software, refer to the "740 FAMILY SOFTWARE MANUAL."

For details of development support tools, refer to the "Renesas Technology" Homepage (<http://www.renesas.com>).

# Table of contents

## CHAPTER 1 HARDWARE

<b>DESCRIPTION</b> .....	<b>2</b>
<b>FEATURES</b> .....	<b>2</b>
<b>PIN CONFIGURATION</b> .....	<b>2</b>
<b>FUNCTIONAL BLOCK</b> .....	<b>3</b>
<b>PIN DESCRIPTION</b> .....	<b>4</b>
<b>PART NUMBERING</b> .....	<b>5</b>
<b>GROUP EXPANSION</b> .....	<b>6</b>
Memory Type .....	6
Memory Size .....	6
Packages .....	6
<b>FUNCTIONAL DESCRIPTION</b> .....	<b>7</b>
Central Processing Unit (CPU) .....	7
Memory .....	11
I/O Ports .....	13
Interrupts .....	17
Timers .....	20
Serial Interface .....	22
USB Function .....	26
HUB Function .....	58
External Bus Interface (EXB) .....	70
Multichannel RAM .....	89
A/D Converter .....	91
Watchdog Timer .....	93
Reset Circuit .....	94
PLL Circuit (Frequency Synthesizer) .....	95
Clock Generating Circuit .....	97
Flash Memory Mode .....	100
<b>NOTES ON PROGRAMMING</b> .....	<b>126</b>
<b>NOTES ON USAGE</b> .....	<b>128</b>
<b>DATA REQUIRED FOR MASK ORDERS</b> .....	<b>128</b>
<b>FUNCTIONAL DESCRIPTION SUPPLEMENT</b> .....	<b>129</b>

## CHAPTER 2 APPLICATION

<b>2.1 I/O port</b> .....	<b>2</b>
2.1.1 Memory map .....	2
2.1.2 Related registers .....	3
2.1.3 Handling of unused pins .....	5
2.1.4 Notes on input and output pins .....	6
2.1.5 Termination of unused pins .....	7
<b>2.2 Interrupt</b> .....	<b>8</b>
2.2.1 Memory map .....	8
2.2.2 Related registers .....	8
2.2.3 Interrupt source .....	11
2.2.4 Interrupt operation .....	12
2.2.5 Interrupt control .....	15
2.2.6 INT interrupt .....	18

2.2.7 Key input interrupt .....	19
2.2.8 Notes on interrupts .....	21
<b>2.3 Timer .....</b>	<b>23</b>
2.3.1 Memory map .....	23
2.3.2 Related registers .....	23
2.3.3 Timer application examples .....	28
2.3.4 Notes on timer .....	39
<b>2.4 Serial I/O .....</b>	<b>40</b>
2.4.1 Memory map .....	40
2.4.2 Related registers .....	41
2.4.3 Serial I/O connection examples .....	45
2.4.4 Setting of serial I/O transfer data format .....	47
2.4.5 Serial I/O application examples .....	48
2.4.6 Notes on serial I/O .....	66
<b>2.5 USB function .....</b>	<b>69</b>
<b>2.6 HUB function .....</b>	<b>70</b>
<b>2.7 External bus interface(EXB) .....</b>	<b>71</b>
<b>2.8 A/D converter .....</b>	<b>72</b>
2.8.1 Memory map .....	72
2.8.2 Related registers .....	72
2.8.3 A/D converter application examples .....	75
2.8.4 Notes on A/D converter .....	77
<b>2.9 Watchdog timer .....</b>	<b>78</b>
2.9.1 Memory map .....	78
2.9.2 Related registers .....	78
2.9.3 Watchdog timer application examples .....	80
2.9.4 Notes on watchdog timer .....	81
<b>2.10 Reset .....</b>	<b>82</b>
2.10.1 Connection example of reset IC .....	82
2.10.2 Notes on RESET pin .....	83
<b>2.11 Frequency synthesizer (PLL) .....</b>	<b>84</b>
2.11.1 Memory map .....	84
2.11.2 Related registers .....	84
2.11.3 Functional description .....	86
2.11.4 Notes on PLL .....	89
<b>2.12 Clock generating circuit .....</b>	<b>90</b>
2.12.1 Memory map .....	90
2.12.2 Related registers .....	90
2.12.3 Oscillation control .....	92
<b>2.13 Standby function .....</b>	<b>95</b>
2.13.1 Memory map .....	95
2.13.2 Related registers .....	95
2.13.3 Stop mode .....	96
2.13.4 Wait mode .....	100
2.13.5 Notes on stand-by function .....	102
<b>2.14 Flash memory .....</b>	<b>103</b>
2.14.1 Overview .....	103
2.14.2 Memory map .....	103
2.14.3 Related registers .....	104
2.14.4 Parallel I/O mode .....	105
2.14.5 Standard serial I/O mode .....	105
2.14.6 CPU rewrite mode .....	106

2.14.7 Flash memory mode application examples .....	107
2.14.8 Notes on CPU rewrite mode .....	112

## CHAPTER 3 APPENDIX

<b>3.1 Electrical characteristics .....</b>	<b>2</b>
3.1.1 Absolute maximum ratings .....	2
3.1.2 Recommended operating conditions (L.Ver) .....	3
3.1.3 Electrical characteristics (L.Ver) .....	5
3.1.4 A/D converter characteristics (L.Ver) .....	7
3.1.5 Timing requirements (L.Ver) .....	8
3.1.6 Switching characteristics (L.Ver) .....	11
<b>3.2 Notes on use .....</b>	<b>20</b>
3.2.1 Notes on input and output ports .....	20
3.2.2 Termination of unused pins .....	21
3.2.3 Notes on interrupts .....	22
3.2.4 Notes on timer .....	23
3.2.5 Notes on serial I/O .....	24
3.2.6 Notes on USB function .....	26
3.2.7 Notes on A/D converter .....	27
3.2.8 Notes on watchdog timer .....	27
3.2.9 Notes on $\overline{\text{RESET}}$ pin .....	27
3.2.10 Notes on PLL .....	27
3.2.11 Notes on stand-by function .....	28
3.2.12 Notes on CPU rewrite mode .....	28
3.2.13 Notes on programming .....	29
3.2.14 Notes on flash memory version .....	31
3.2.15 Electric Characteristic Differences Between Mask ROM and Flash Memory Version MCUs .....	31
<b>3.3 Countermeasures against noise .....</b>	<b>32</b>
3.3.1 Shortest wiring length .....	32
3.3.2 Connection of bypass capacitor across $V_{SS}$ line and $V_{CC}$ line .....	34
3.3.3 Wiring to analog input pins .....	35
3.3.4 Oscillator concerns .....	36
3.3.5 Setup for I/O ports .....	37
3.3.6 Providing of watchdog timer function by software .....	38
<b>3.4 List of registers .....</b>	<b>39</b>
<b>3.5 Package outline .....</b>	<b>83</b>
<b>3.6 Machine instructions .....</b>	<b>86</b>
<b>3.7 List of instruction code .....</b>	<b>97</b>
<b>3.8 SFR memory map .....</b>	<b>98</b>
<b>3.9 Pin configurations .....</b>	<b>99</b>

# List of figures

## CHAPTER 1 HARDWARE

Fig. 1	Pin configuration of 38K2 group .....	2
Fig. 2	Functional block diagram .....	3
Fig. 3	Part numbering .....	5
Fig. 4	Memory expansion plan .....	6
Fig. 5	740 Family CPU register structure .....	7
Fig. 6	Register push and pop at interrupt generation and subroutine call .....	8
Fig. 7	Structure of CPU mode register .....	10
Fig. 8	Memory map diagram .....	11
Fig. 9	Memory map of special function register (SFR) .....	12
Fig. 10	Port block diagram (1) .....	14
Fig. 11	Port block diagram (2) .....	15
Fig. 12	Structure of port I/O-related registers .....	16
Fig. 13	Interrupt control .....	18
Fig. 14	Structure of interrupt-related registers .....	18
Fig. 15	Connection example when using key input interrupt and port P0 block diagram ..	19
Fig. 16	Structure of timer X mode register .....	20
Fig. 17	Timer block diagram .....	21
Fig. 18	Block diagram of clock synchronous serial I/O .....	22
Fig. 19	Operation of clock synchronous serial I/O function .....	22
Fig. 20	Block diagram of UART serial I/O .....	23
Fig. 21	Operation of UART serial I/O function .....	23
Fig. 22	Structure of serial I/O control registers .....	25
Fig. 23	USB function overview .....	26
Fig. 24	USB Function Control Circuit (USBFCC) block diagram .....	27
Fig. 25	USB port external circuit (D0+, D0-, USBV <sub>REF</sub> , TrON) block diagram (4.0V ≤ V <sub>CC</sub> ≤ 5.25V) .....	28
Fig. 26	USB port external circuit (D0+, D0-, USBV <sub>REF</sub> , TrON) block diagram (3.0V ≤ V <sub>CC</sub> ≤ 4.0V) .....	28
Fig. 27	Example setting of buffer area beginning address .....	29
Fig. 28	Examples of interrupt source dependant buffer area offset address .....	29
Fig. 29	USB device interrupt control .....	31
Fig. 30	USB related registers .....	32
Fig. 31	Structure of USB control register .....	33
Fig. 32	Structure of USB function/HUB enable register .....	33
Fig. 33	Structure of USB function address register .....	34
Fig. 34	Structure of USB HUB address register .....	34
Fig. 35	Structure of Frame number register Low .....	34
Fig. 36	Structure of Frame number register High .....	34
Fig. 37	Structure of USB interrupt source enable register .....	35
Fig. 38	Structure of USB interrupt source register .....	36
Fig. 39	Structure of Endpoint index register .....	37
Fig. 40	Structure of EP00 stage register .....	38
Fig. 41	Structure of EP00 control register 1 .....	38
Fig. 42	Structure of EP00 control register 2 .....	38
Fig. 43	Structure of EP00 control register 3 .....	39

Fig. 44	Structure of EP00 interrupt source register .....	39
Fig. 45	Structure of EP00 byte number register .....	40
Fig. 46	Structure of EP00 buffer area set register .....	40
Fig. 47	Structure of EP01 set register .....	41
Fig. 48	Structure of EP01 control register 1 .....	41
Fig. 49	Structure of EP01 control register 2 .....	42
Fig. 50	Structure of EP01 control register 3 .....	42
Fig. 51	Structure of EP01 interrupt source register .....	42
Fig. 52	Structure of EP01 byte number register 0 .....	43
Fig. 53	Structure of EP01 byte number register 1 .....	43
Fig. 54	Structure of EP01 MAX. packet size register .....	43
Fig. 55	Structure of EP01 buffer area set register .....	44
Fig. 56	Structure of EP02 set register .....	45
Fig. 57	Structure of EP02 control register 1 .....	45
Fig. 58	Structure of EP02 control register 2 .....	46
Fig. 59	Structure of EP02 control register 3 .....	46
Fig. 60	Structure of EP02 interrupt source register .....	46
Fig. 61	Structure of EP02 byte number register 0 .....	47
Fig. 62	Structure of EP02 byte number register 1 .....	47
Fig. 63	Structure of EP02 MAX. packet size register .....	47
Fig. 64	Structure of EP02 buffer area set register .....	48
Fig. 65	Structure of EP03 set register .....	49
Fig. 66	Structure of EP03 control register 1 .....	49
Fig. 67	Structure of EP03 control register 2 .....	50
Fig. 68	Structure of EP03 control register 3 .....	50
Fig. 69	Structure of EP03 interrupt source register .....	50
Fig. 70	Structure of EP03 byte number register 0 .....	51
Fig. 71	Structure of EP03 byte number register 1 .....	51
Fig. 72	Structure of EP03 MAX. packet size register .....	51
Fig. 73	Structure of EP03 buffer area set register .....	52
Fig. 74	Structure of EP10 stage register .....	53
Fig. 75	Structure of EP10 control register 1 .....	53
Fig. 76	Structure of EP10 control register 2 .....	53
Fig. 77	Structure of EP10 control register 3 .....	54
Fig. 78	Structure of EP10 interrupt source register .....	54
Fig. 79	Structure of EP10 byte number register .....	55
Fig. 80	Structure of EP10 buffer area set register .....	55
Fig. 81	Structure of EP11 set register .....	56
Fig. 82	Structure of EP11 control register 1 .....	56
Fig. 83	Structure of EP11 control register 2 .....	56
Fig. 84	Structure of EP11 interrupt source register .....	57
Fig. 85	Structure of EP11 byte number register .....	57
Fig. 86	Structure of EP11 buffer area set register .....	57
Fig. 87	HUB functions .....	58
Fig. 88	HUB function control circuit block diagram .....	59
Fig. 89	Block diagram of USB down-port peripheral circuits (D1+, D1-) .....	60
Fig. 90	Block diagram of USB down-port peripheral circuits (D2+, D2-) .....	60
Fig. 91	USB HUB interrupt control .....	61
Fig. 92	HUB related registers .....	62
Fig. 93	Structure of HUB interrupt source enable register .....	63
Fig. 94	Structure of HUB interrupt source register .....	63
Fig. 95	Structure of HUB downstream port index register .....	64

Fig. 96	Structure of DP1 interrupt source register .....	65
Fig. 97	Structure of DP1 control register .....	66
Fig. 98	Structure of DP1 status register .....	66
Fig. 99	Structure of DP2 interrupt source register .....	67
Fig. 100	Structure of DP2 control register .....	68
Fig. 101	Structure of DP2 status register .....	68
Fig. 102	Structure of Downstream port control register .....	69
Fig. 103	External bus interface .....	70
Fig. 104	Data transfer timing of memory channel .....	70
Fig. 105	External bus interface (EXB) pin assignment .....	71
Fig. 106	Block diagram of external bus interface (EXB) .....	72
Fig. 107	EXB related registers (1) .....	76
Fig. 108	EXB related registers (2) .....	76
Fig. 109	Structure of EXB interrupt source enable register .....	77
Fig. 110	Structure of EXB interrupt source register .....	77
Fig. 111	Structure of EXB index register .....	78
Fig. 112	Structure of Register window 1 .....	78
Fig. 113	Structure of Register window 2 .....	78
Fig. 114	Index00[low]; Structure of External I/O configuration register .....	79
Fig. 115	Index00[high]; Structure of External I/O configuration register .....	79
Fig. 116	Index01[low]; Structure of Transmit/Receive buffer register .....	80
Fig. 117	Index02[low]; Structure of Memory channel operation mode register .....	80
Fig. 118	Index03[low]; Structure of Memory address counter .....	80
Fig. 119	Index03[high]; Structure of Memory address counter .....	81
Fig. 120	Index04[low]; Structure of End address register .....	81
Fig. 121	Index04[high]; Structure of End address register .....	81
Fig. 122	CPU channel receiving operation .....	82
Fig. 123	CPU channel transmitting operation .....	83
Fig. 124	Memory channel receiving operation (1) .....	84
Fig. 125	Memory channel receiving operation (2) .....	85
Fig. 126	Memory channel receiving operation (3) .....	86
Fig. 127	Memory channel transmitting operation (1) .....	87
Fig. 128	Memory channel transmitting operation (2) .....	88
Fig. 129	Multichannel RAM timing diagram (no wait) .....	89
Fig. 130	Multichannel RAM timing diagram (one wait) .....	89
Fig. 131	Multichannel RAM operation example .....	90
Fig. 132	Structure of AD control register .....	91
Fig. 133	10-bit A/D mode reading .....	91
Fig. 134	A/D converter block diagram .....	92
Fig. 135	Block diagram of Watchdog timer .....	93
Fig. 136	Structure of Watchdog timer control register .....	93
Fig. 137	Example of reset circuit .....	94
Fig. 138	Reset sequence .....	94
Fig. 139	Block diagram of PLL circuit .....	95
Fig. 140	Structure of PLL control register .....	96
Fig. 141	Ceramic resonator or quartz-crystal oscillator circuit .....	98
Fig. 142	External clock input circuit .....	98
Fig. 143	Structure of MISRG .....	98
Fig. 144	System clock generating circuit block diagram (single-chip mode) .....	98
Fig. 145	State transitions of clock .....	99
Fig. 146	Block diagram of built-in flash memory .....	101
Fig. 147	Structure of flash memory control register .....	102

Fig. 148	CPU rewrite mode set/release flowchart .....	103
Fig. 149	Program flowchart .....	105
Fig. 150	Erase flowchart .....	106
Fig. 151	Full status check flowchart and remedial procedure for errors .....	108
Fig. 152	Structure of ROM code protect control register .....	109
Fig. 153	ID code store addresses .....	110
Fig. 154	Pin connection diagram in standard serial I/O mode (1) .....	114
Fig. 155	Timing for page read .....	116
Fig. 156	Timing for reading status register .....	116
Fig. 157	Timing for clear status register .....	117
Fig. 158	Timing for page program .....	117
Fig. 159	Timing for erase all blocks .....	118
Fig. 160	Timing for download .....	119
Fig. 161	Timing for version information output .....	120
Fig. 162	Timing for Boot ROM area output .....	120
Fig. 163	Timing for ID check .....	121
Fig. 164	ID code storage addresses .....	121
Fig. 165	Full status check flowchart and remedial procedure for errors .....	124
Fig. 166	Example circuit application for standard serial I/O mode .....	125
Fig. 167	Definition of A/D conversion accuracy .....	127
Fig. 168	A/D conversion equivalent circuit .....	130
Fig. 169	A/D conversion timing chart .....	130

## CHAPTER 2 APPLICATION

Fig. 2.1.1	Memory map of registers related to I/O port .....	2
Fig. 2.1.2	Structure of Port Pi (i = 0 to 6) .....	3
Fig. 2.1.3	Structure of Port Pi direction register (i = 0 to 6) .....	3
Fig. 2.1.4	Structure of Port P0 pull-up control register .....	4
Fig. 2.1.5	Structure of Port P5 pull-up control register .....	4
Fig. 2.2.1	Memory map of registers related to interrupt .....	8
Fig. 2.2.2	Structure of Interrupt request register 1 .....	8
Fig. 2.2.3	Structure of Interrupt request register 2 .....	9
Fig. 2.2.4	Structure of Interrupt control register 1 .....	9
Fig. 2.2.5	Structure of Interrupt control register 2 .....	10
Fig. 2.2.6	Structure of Interrupt edge selection register .....	10
Fig. 2.2.7	Interrupt operation diagram .....	12
Fig. 2.2.8	Changes of stack pointer and program counter upon acceptance of interrupt request .....	13
Fig. 2.2.9	Time up to execution of interrupt processing routine .....	14
Fig. 2.2.10	Timing chart after acceptance of interrupt request .....	14
Fig. 2.2.11	Interrupt control diagram .....	15
Fig. 2.2.12	Example of multiple interrupts .....	17
Fig. 2.2.13	Connection example and port P0 block diagram when using key input interrupt .....	19
Fig. 2.2.14	Registers setting related to key input interrupt (corresponding to Figure 2.2.13) .....	20
Fig. 2.2.15	Sequence of changing relevant register .....	21
Fig. 2.2.16	Sequence of check of interrupt request bit .....	22
Fig. 2.3.1	Memory map of registers related to timers .....	23
Fig. 2.3.2	Structure of Prescaler 12, Prescaler X .....	23
Fig. 2.3.3	Structure of Timer 1 .....	24
Fig. 2.3.4	Structure of Timer 2, Timer X .....	24

Fig. 2.3.5 Structure of Timer X mode register.....	25
Fig. 2.3.6 Structure of Interrupt request register 1 .....	26
Fig. 2.3.7 Structure of Interrupt request register 2 .....	26
Fig. 2.3.8 Structure of Interrupt control register 1 .....	27
Fig. 2.3.9 Structure of Interrupt control register 2 .....	27
Fig. 2.3.10 Timers connection and setting of division ratios .....	29
Fig. 2.3.11 Related registers setting .....	29
Fig. 2.3.12 Control procedure.....	30
Fig. 2.3.13 Peripheral circuit example.....	31
Fig. 2.3.14 Timers connection and setting of division ratios .....	31
Fig. 2.3.15 Related registers setting .....	32
Fig. 2.3.16 Control procedure.....	32
Fig. 2.3.17 Judgment method of valid/invalid of input pulses .....	33
Fig. 2.3.18 Related registers setting .....	34
Fig. 2.3.19 Control procedure.....	35
Fig. 2.3.20 Timers connection and setting of division ratios .....	36
Fig. 2.3.21 Related registers setting .....	37
Fig. 2.3.22 Control procedure.....	38
Fig. 2.4.1 Memory map of registers related to Serial I/O .....	40
Fig. 2.4.2 Structure of Transmit/Receive buffer register .....	41
Fig. 2.4.3 Structure of Serial I/O status register .....	41
Fig. 2.4.4 Structure of Serial I/O control register .....	42
Fig. 2.4.5 Structure of UART control register .....	42
Fig. 2.4.6 Structure of Baud rate generator .....	43
Fig. 2.4.7 Structure of Interrupt edge selection register .....	43
Fig. 2.4.8 Structure of Interrupt request register 2 .....	44
Fig. 2.4.9 Structure of Interrupt control register 2 .....	44
Fig. 2.4.10 Serial I/O connection examples (1) .....	45
Fig. 2.4.11 Serial I/O connection examples (2) .....	46
Fig. 2.4.12 Serial I/O transfer data format .....	47
Fig. 2.4.13 Connection diagram .....	48
Fig. 2.4.14 Timing chart .....	48
Fig. 2.4.15 Registers setting related to transmitting side .....	49
Fig. 2.4.16 Registers setting related to receiving side .....	50
Fig. 2.4.17 Control procedure of transmitting side.....	51
Fig. 2.4.18 Control procedure of receiving side .....	52
Fig. 2.4.19 Connection diagram .....	53
Fig. 2.4.20 Timing chart .....	53
Fig. 2.4.21 Registers setting related to Serial I/O .....	54
Fig. 2.4.22 Setting of serial I/O transmission data .....	54
Fig. 2.4.23 Control procedure of Serial I/O .....	55
Fig. 2.4.24 Connection diagram .....	56
Fig. 2.4.25 Timing chart .....	57
Fig. 2.4.26 Related registers setting .....	57
Fig. 2.4.27 Control procedure of master unit.....	58
Fig. 2.4.28 Control procedure of slave unit .....	59
Fig. 2.4.29 Connection diagram (Communication using UART) .....	60
Fig. 2.4.30 Timing chart (using UART) .....	60
Fig. 2.4.31 Registers setting related to transmitting side .....	62
Fig. 2.4.32 Registers setting related to receiving side .....	63
Fig. 2.4.33 Control procedure of transmitting side.....	64
Fig. 2.4.34 Control procedure of receiving side .....	65
Fig. 2.4.35 Sequence of setting serial I/O control register again .....	67

Fig. 2.8.1 Memory map of registers related to A/D converter .....	72
Fig. 2.8.2 Structure of AD control register .....	72
Fig. 2.8.3 Structure of AD conversion register 1 .....	73
Fig. 2.8.4 Structure of AD conversion register 2 .....	73
Fig. 2.8.5 Structure of Interrupt request register 2 .....	74
Fig. 2.8.6 Structure of Interrupt control register 2 .....	74
Fig. 2.8.7 Connection diagram .....	75
Fig. 2.8.8 Related registers setting .....	75
Fig. 2.8.9 Control procedure for 8-bit read .....	76
Fig. 2.8.10 Control procedure for 10-bit read .....	76
Fig. 2.9.1 Memory map of registers related to watchdog timer .....	78
Fig. 2.9.2 Structure of Watchdog timer control register .....	78
Fig. 2.9.3 Structure of CPU mode register .....	79
Fig. 2.9.4 Watchdog timer connection and division ratio setting .....	80
Fig. 2.9.5 Related registers setting .....	81
Fig. 2.9.6 Control procedure .....	81
Fig. 2.10.1 Example of poweron reset circuit .....	82
Fig. 2.10.2 RAM backup system .....	82
Fig. 2.11.1 Memory map of registers related to PLL .....	84
Fig. 2.11.2 Structure of USB control register .....	84
Fig. 2.11.3 Structure of CPU mode register .....	85
Fig. 2.11.4 Structure of PLL control register .....	85
Fig. 2.11.5 Block diagram for frequency synthesizer circuit .....	86
Fig. 2.11.6 Related registers setting when hardware reset .....	87
Fig. 2.11.7 Related registers setting when stop mode .....	88
Fig. 2.11.8 Related registers setting when recovery from stop mode .....	89
Fig. 2.12.1 Memory map of registers related to clock generating circuit .....	90
Fig. 2.12.2 Structure of USB control register .....	90
Fig. 2.12.3 Structure of CPU mode register .....	91
Fig. 2.12.4 Structure of PLL control register .....	91
Fig. 2.12.5 Related registers setting .....	92
Fig. 2.12.6 Related registers setting .....	94
Fig. 2.13.1 Memory map of registers related to standby function .....	95
Fig. 2.13.2 Structure of MISRG .....	95
Fig. 2.13.3 Oscillation stabilizing time at restoration by reset input .....	97
Fig. 2.13.4 Execution sequence example at restoration by occurrence of INT <sub>0</sub> interrupt request .....	99
Fig. 2.13.5 Reset input time .....	101
Fig. 2.14.1 Memory map of flash memory version for 38K2 Group .....	103
Fig. 2.14.2 Memory map of registers related to flash memory .....	104
Fig. 2.14.3 Structure of Flash memory control register .....	104
Fig. 2.14.4 Rewrite example of built-in flash memory in standard serial I/O mode .....	107
Fig. 2.14.5 Connection example in standard serial I/O mode (1) .....	108
Fig. 2.14.6 Connection example in standard serial I/O mode (2) .....	108
Fig. 2.14.7 Connection example in standard serial I/O mode (3) .....	109
Fig. 2.14.8 Example of rewrite system for built-in flash memory in CPU rewrite mode ....	110
Fig. 2.14.9 CPU rewrite mode beginning/release flowchart .....	111

## CHAPTER 3 APPENDIX

Fig. 3.1.1 Output switching characteristics measurement circuit .....	11
Fig. 3.1.2 USB output switching characteristics measurement circuit (1) for D0-, D1+/D2+ (low-speed), D1-/D2- (full-speed) .....	13

Fig. 3.1.3 SB output switching characteristics measurement circuit (2) for D0+, D1+/D2+ (full-speed), D1-/D2- (low-speed) .....	13
Fig. 3.1.4 Timing chart (1) .....	14
Fig. 3.1.5 Timing chart (2) .....	15
Fig. 3.1.6 Timing chart (3) .....	16
Fig. 3.1.7 Timing chart (4) .....	17
Fig. 3.1.8 Timing chart (5) .....	18
Fig. 3.1.9 Timing chart (6) .....	19
Fig. 3.2.1 Sequence of changing relevant register .....	22
Fig. 3.2.2 Sequence of check of interrupt request bit .....	23
Fig. 3.2.3 Sequence of setting serial I/O control register again .....	25
Fig. 3.2.4 Initialization of processor status register .....	29
Fig. 3.2.5 Sequence of PLP instruction execution .....	29
Fig. 3.2.6 Stack memory contents after PHP instruction execution .....	29
Fig. 3.2.7 Status flag at decimal calculations .....	30
Fig. 3.3.1 Selection of packages .....	32
Fig. 3.3.2 Wiring for the RESET pin .....	32
Fig. 3.3.3 Wiring for clock I/O pins .....	33
Fig. 3.3.4 Wiring for CNV <sub>SS</sub> pin .....	33
Fig. 3.3.5 Wiring for the V <sub>PP</sub> pin of the flash memory version .....	34
Fig. 3.3.6 Bypass capacitor across the V <sub>SS</sub> line and the V <sub>CC</sub> line .....	34
Fig. 3.3.7 Analog signal line and a resistor and a capacitor .....	35
Fig. 3.3.8 Wiring for a large current signal line .....	36
Fig. 3.3.9 Wiring of RESET pin .....	36
Fig. 3.3.10 V <sub>SS</sub> pattern on the underside of an oscillator .....	37
Fig. 3.3.11 Setup for I/O ports .....	37
Fig. 3.3.12 Watchdog timer by software .....	38
Fig. 3.4.1 Structure of Port Pi .....	39
Fig. 3.4.2 Structure of Port Pi direction register .....	39
Fig. 3.4.3 Structure of USB control register .....	40
Fig. 3.4.4 Structure of USB function/HUB enable register .....	40
Fig. 3.4.5 Structure of USB function address register .....	40
Fig. 3.4.6 Structure of USB HUB address register .....	41
Fig. 3.4.7 Structure of Frame number register Low .....	41
Fig. 3.4.8 Structure of Frame number register High .....	41
Fig. 3.4.9 Structure of USB interrupt source enable register .....	41
Fig. 3.4.10 Structure of USB interrupt source register .....	42
Fig. 3.4.11 Structure of Endpoint index register .....	42
Fig. 3.4.12 Structure of EP00 stage register .....	43
Fig. 3.4.13 Structure of EP01 set register .....	43
Fig. 3.4.14 Structure of EP02 set register .....	44
Fig. 3.4.15 Structure of EP03 set register .....	44
Fig. 3.4.16 Structure of EP10 stage register .....	45
Fig. 3.4.17 Structure of EP11 set register .....	45
Fig. 3.4.18 Structure of EP00 control register 1 .....	45
Fig. 3.4.19 Structure of EP01 control register 1 .....	46
Fig. 3.4.20 Structure of EP02 control register 1 .....	46
Fig. 3.4.21 Structure of EP03 control register 1 .....	46
Fig. 3.4.22 Structure of EP10 control register 1 .....	47
Fig. 3.4.23 Structure of EP11 control register 1 .....	47
Fig. 3.4.24 Structure of EP00 control register 2 .....	47
Fig. 3.4.25 Structure of EP01 control register 2 .....	48
Fig. 3.4.26 Structure of EP02 control register 2 .....	48

Fig. 3.4.27 Structure of EP03 control register 2 .....	48
Fig. 3.4.28 Structure of EP10 control register 2 .....	49
Fig. 3.4.29 Structure of EP11 control register 2 .....	49
Fig. 3.4.30 Structure of EP00 control register 3 .....	49
Fig. 3.4.31 Structure of EP01 control register 3 .....	50
Fig. 3.4.32 Structure of EP02 control register 3 .....	50
Fig. 3.4.33 Structure of EP03 control register 3 .....	50
Fig. 3.4.34 Structure of EP10 control register 3 .....	51
Fig. 3.4.35 Structure of EP00 interrupt source register .....	51
Fig. 3.4.36 Structure of EP01 interrupt source register .....	52
Fig. 3.4.37 Structure of EP02 interrupt source register .....	52
Fig. 3.4.38 Structure of EP03 interrupt source register .....	53
Fig. 3.4.39 Structure of EP10 interrupt source register .....	54
Fig. 3.4.40 Structure of EP11 interrupt source register .....	54
Fig. 3.4.41 Structure of EP00 byte number register .....	55
Fig. 3.4.42 Structure of EP01 byte number register 0 .....	55
Fig. 3.4.43 Structure of EP02 byte number register 0 .....	55
Fig. 3.4.44 Structure of EP03 byte number register 0 .....	56
Fig. 3.4.45 Structure of EP10 byte number register .....	56
Fig. 3.4.46 Structure of EP11 byte number register 0 .....	56
Fig. 3.4.47 Structure of EP01 byte number register 1 .....	57
Fig. 3.4.48 Structure of EP02 byte number register 1 .....	57
Fig. 3.4.49 Structure of EP03 byte number register 1 .....	57
Fig. 3.4.50 Structure of Prescaler12, Prescaler X .....	58
Fig. 3.4.51 Structure of Timer 1 .....	58
Fig. 3.4.52 Structure of Timer 2, Timer X .....	59
Fig. 3.4.53 Structure of Timer X mode register .....	59
Fig. 3.4.54 Structure of Transmit/Receive buffer register .....	60
Fig. 3.4.55 Structure of Serial I/O status register .....	60
Fig. 3.4.56 Structure of HUB interrupt source enable register .....	61
Fig. 3.4.57 Structure of HUB interrupt source register .....	61
Fig. 3.4.58 Structure of HUB downstream port index register .....	61
Fig. 3.4.59 Structure of DP1 interrupt source register .....	62
Fig. 3.4.60 Structure of DP2 interrupt source register .....	63
Fig. 3.4.61 Structure of DP1 control register .....	64
Fig. 3.4.62 Structure of DP2 control register .....	64
Fig. 3.4.63 Structure of DP1 status register .....	65
Fig. 3.4.64 Structure of DP2 status register .....	65
Fig. 3.4.65 Structure of EXB interrupt source enable register .....	65
Fig. 3.4.66 Structure of EXB interrupt source register .....	66
Fig. 3.4.67 Structure of EXB index register .....	66
Fig. 3.4.68 Structure of Register window 1 .....	67
Fig. 3.4.69 Index00[low]; Structure of External I/O configuration register .....	67
Fig. 3.4.70 Index01[low]; Structure of Transmit/Receive buffer register .....	68
Fig. 3.4.71 Index02[low]; Structure of Memory channel operation mode register .....	68
Fig. 3.4.72 Index03[low]; Structure of Memory address counter .....	68
Fig. 3.4.73 Index04[low]; Structure of End address register .....	69
Fig. 3.4.74 Structure of Register window 2 .....	69
Fig. 3.4.75 Index00[high]; Structure of External I/O configuration register .....	69
Fig. 3.4.76 Index03[high]; Structure of Memory address counter .....	70
Fig. 3.4.77 Index04[high]; Structure of End address register .....	70
Fig. 3.4.78 Structure of AD control register .....	70
Fig. 3.4.79 Structure of AD conversion register 1 .....	71

---

Fig. 3.4.80 Structure of AD conversion register 2 .....	71
Fig. 3.4.81 Structure of Watchdog timer control register .....	72
Fig. 3.4.82 Structure of CPU mode register .....	72
Fig. 3.4.83 Structure of Interrupt request register 1 .....	73
Fig. 3.4.84 Structure of Interrupt request register 2 .....	73
Fig. 3.4.85 Structure of Interrupt control register 1 .....	74
Fig. 3.4.86 Structure of Interrupt control register 2 .....	74
Fig. 3.4.87 Structure of Serial I/O control register .....	75
Fig. 3.4.88 Structure of UART control register .....	75
Fig. 3.4.89 Structure of Baud rate generator .....	76
Fig. 3.4.90 Structure of EP01 MAX. packet size register .....	76
Fig. 3.4.91 Structure of EP02 MAX. packet size register .....	76
Fig. 3.4.92 Structure of EP03 MAX. packet size register .....	77
Fig. 3.4.93 Structure of EP00 buffer area set register .....	77
Fig. 3.4.94 Structure of EP01 buffer area set register .....	77
Fig. 3.4.95 Structure of EP02 buffer area set register .....	78
Fig. 3.4.96 Structure of EP03 buffer area set register .....	78
Fig. 3.4.97 Structure of EP10 buffer area set register .....	78
Fig. 3.4.98 Structure of EP11 buffer area set register .....	79
Fig. 3.4.99 Structure of Port P0 pull-up control register .....	79
Fig. 3.4.100 Structure of Port P5 pull-up control register .....	80
Fig. 3.4.101 Structure of Interrupt edge selection register .....	80
Fig. 3.4.102 Structure of PLL control register .....	81
Fig. 3.4.103 Structure of Downstream port control register .....	81
Fig. 3.4.104 Structure of MISRG .....	82
Fig. 3.4.105 Structure of Flash memory control register .....	82

# List of tables

## CHAPTER 1 HARDWARE

Table 1	Pin description .....	4
Table 2	List of 38K2 group products (L version) .....	6
Table 3	Push and pop instructions of accumulator or processor status register .....	8
Table 4	Set and clear instructions of each bit of processor status register .....	9
Table 5	I/O ports functions .....	13
Table 6	Interrupt vector addresses and priority .....	17
Table 7	USB interrupt sources .....	30
Table 8	HUBinterrupt sources .....	61
Table 9	Summary of 38K2 group's flash memory version .....	100
Table 10	List of software commands (CPU rewrite mode) .....	105
Table 11	Definition of each bit in status register .....	107
Table 12	Description of pin function (Standard Serial I/O Mode) .....	113
Table 13	Software commands (Standard serial I/O mode) .....	115
Table 14	Status register (SRD) .....	122
Table 15	Status register 1 (SRD1) .....	123
Table 16	Relative formula for a reference voltage VREF of A/D converter and Vref .....	129
Table 17	Change of AD conversion register during A/D conversion .....	129

## CHAPTER 2 APPLICATION

Table 2.1.1	Handling of unused pins .....	5
Table 2.2.1	Interrupt sources, vector addresses and priority of 38K2 group .....	11
Table 2.2.2	List of interrupt bits according to interrupt source .....	16
Table 2.3.1	CNTR <sub>0</sub> active edge selection bit function .....	25
Table 2.4.1	Setting examples of Baud rate generator values and transfer bit rate values .....	61
Table 2.11.1	PLL operation mode selection bits setting example .....	86
Table 2.11.2	USB clock division ratio selection bits setting example .....	87
Table 2.12.1	Example of internal clock f(f) generation using main clock f(X <sub>IN</sub> ) .....	92
Table 2.12.2	Example of internal clock f(f) generation using fSYN .....	93
Table 2.13.1	State in stop mode .....	96
Table 2.13.2	State in wait mode .....	100
Table 2.14.1	Setting of programmers when parallel programming .....	105
Table 2.14.2	Connection example to flash programmer when serial programming (4 wires) .....	105
Table 2.14.3	Setting condition in serial I/O mode .....	107

## CHAPTER 3 APPENDIX

Table 3.1.1	Absolute maximum ratings .....	2
Table 3.1.2	Recommended operating conditions (V <sub>CC</sub> = 3.00 to 5.25 V, V <sub>SS</sub> = 0 V, T <sub>a</sub> = -20 to 85°C, unless otherwise noted) .....	3
Table 3.1.3	Recommended operating conditions (V <sub>CC</sub> = 3.00 to 5.25 V, V <sub>SS</sub> = 0 V, T <sub>a</sub> = -20 to 85°C, unless otherwise noted) .....	4
Table 3.1.4	Electrical characteristics (1) (V <sub>CC</sub> = 3.00 to 5.25 V, V <sub>SS</sub> = 0 V, T <sub>a</sub> = -20 to 85°C, unless otherwise noted) .....	5
Table 3.1.5	Electrical characteristics (2) (V <sub>CC</sub> = 3.00 to 5.25 V, V <sub>SS</sub> = 0 V, T <sub>a</sub> = -20 to 85°C, unless otherwise noted) .....	6

Table 3.1.6 A/D Converter characteristics ( $V_{CC} = 3.00$ to $5.25$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	7
Table 3.1.7 Timing requirements (1) ( $V_{CC} = 4.00$ to $5.25$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	8
Table 3.1.8 Timing requirements (2) ( $V_{CC} = 3.00$ to $4.00$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	8
Table 3.1.9 Timing requirements of external bus interface (EXB) (1) ( $V_{CC} = 4.00$ to $5.25$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	9
Table 3.1.10 Timing requirements of external bus interface (EXB) (2) ( $V_{CC} = 3.00$ to $4.00$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	10
Table 3.1.11 Switching characteristics (1) ( $V_{CC} = 4.00$ to $5.25$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	11
Table 3.1.12 Switching characteristics (2) ( $V_{CC} = 3.00$ to $4.00$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	11
Table 3.1.13 Switching characteristics of external bus interface (EXB) (1) ( $V_{CC} = 4.00$ to $5.25$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	12
Table 3.1.14 Switching characteristics of external bus interface (EXB) (2) ( $V_{CC} = 3.00$ to $4.00$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	12
Table 3.1.15 Switching characteristics (USB ports) ( $V_{CC} = 3.00$ to $5.25$ V, $V_{SS} = 0$ V, $T_a = -20$ to $85^\circ\text{C}$ , unless otherwise noted) .....	13

THIS PAGE IS BLANK FOR REASONS OF LAYOUT.

# CHAPTER 1

---

## HARDWARE

DESCRIPTION  
FEATURES  
PIN CONFIGURATION  
FUNCTIONAL BLOCK  
PIN DESCRIPTION  
PART NUMBERING  
GROUP EXPANSION  
FUNCTIONAL DESCRIPTION  
NOTES ON PROGRAMMING  
NOTES ON USAGE  
DATA REQUIRED FOR MASK ORDERS  
FUNCTIONAL DESCRIPTION  
SUPPLEMENT

**DESCRIPTION**

The 38K2 group is the 8-bit microcomputer based on the 740 family core technology.

The 38K2 group has the USB function, an 8-bit bus interface, a Serial Interface, three 8-bit timers, and an 8-channel 10-bit A/D converter, which are available for the PC peripheral I/O device.

The various microcomputers in the 38K2 group include variations of internal memory size and packaging. For details, refer to the section on part numbering.

**FEATURES**

- Basic machine-language instructions ..... 71
- The minimum instruction execution time ..... 0.25 μs  
(at 8 MHz system clock\*)
- System clock\*: Reference frequency to internal circuit except USB function
- Memory size
  - ROM ..... 16 K to 32 K bytes
  - RAM ..... 1024 to 2048 bytes
- Programmable input/output ports ..... 44
- Software pull-up resistors
- Interrupts ..... 16 sources, 16 vectors
- USB function (Full-Speed USB2.0 specification) ..... 4 endpoints
- USB HUB function (Full-Speed USB2.0 specification) .... 2 down ports
- External bus interface ..... 8-bit X 1 channel

- Timers ..... 8-bit X 3
- Watchdog timer ..... 16-bit X 1
- Serial Interface
  - Serial I/O ..... 8-bit X 1 (UART or Clock-synchronized)
- A/D converter ..... 10-bit X 8 channels  
(8-bit reading available)
- LED direct drive port ..... 4
- Clock generating circuit  
(connect to external ceramic resonator or quartz-crystal oscillator)
- Power source voltage (L version)
  - System clock/Internal clock division mode
  - At 12 MHz/2-divide mode(φ = 6 MHz) ..... 4.00 to 5.25 V
  - At 8 MHz/Through mode (φ = 8 MHz) ..... 4.00 to 5.25 V
  - At 6 MHz/Through mode (φ = 6 MHz) ..... 3.00 to 5.25 V
- Power dissipation
  - At 5 V power source voltage ..... 125 mW (typ.)  
(at 8 MHz system clock, in through mode)
  - At 3.3 V power source voltage ..... 30 mW (typ.)  
(at 6 MHz system clock, in through mode)
- Operating temperature range ..... -20 to 85°C
- Packages
  - FP ..... PLQP0064GA-A (64-pin 14 X 14 mm LQFP)
  - HP ..... PLQP0064KB-A (64-pin 10 X 10 mm LQFP)

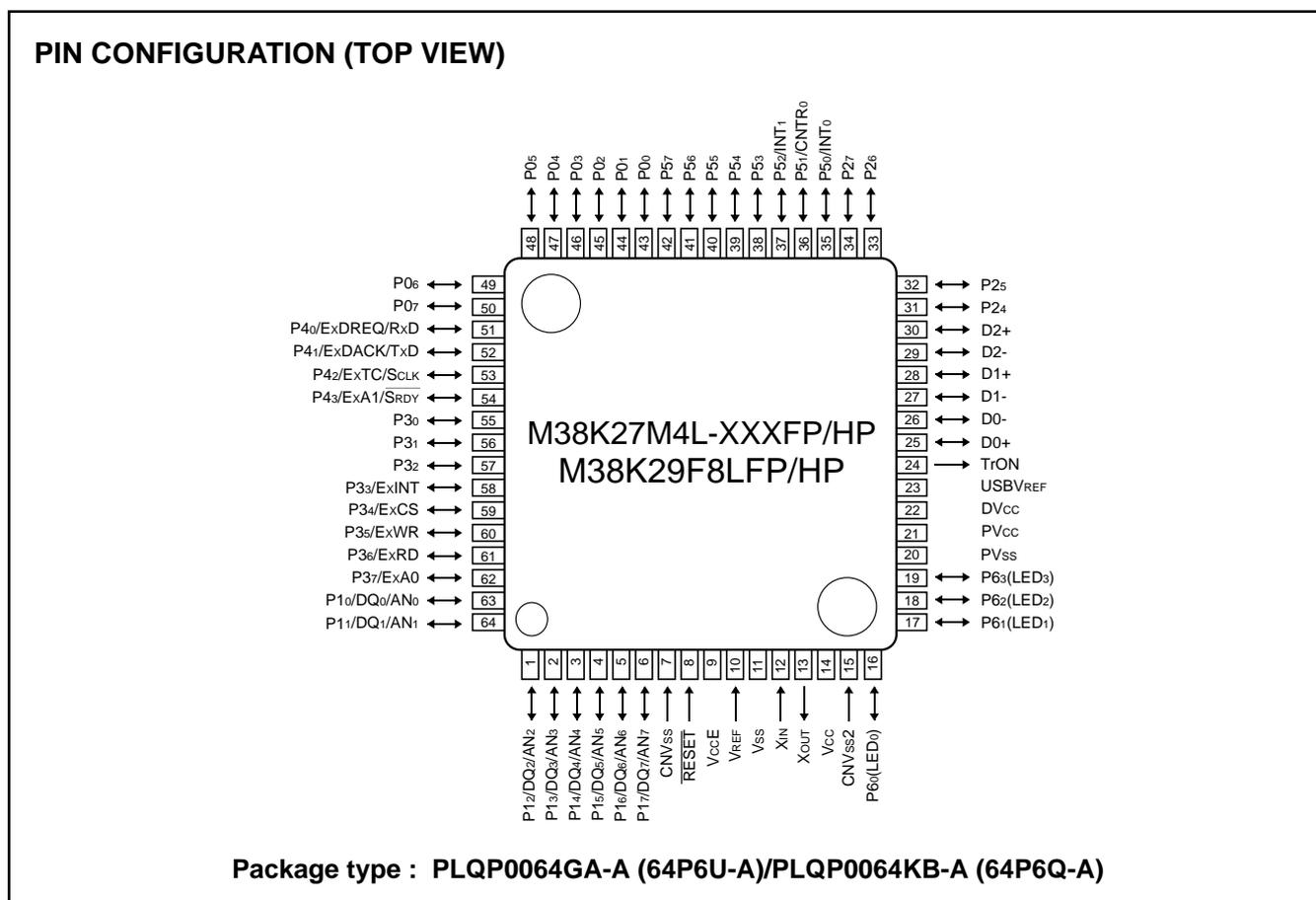


Fig. 1 Pin configuration of 38K2 group

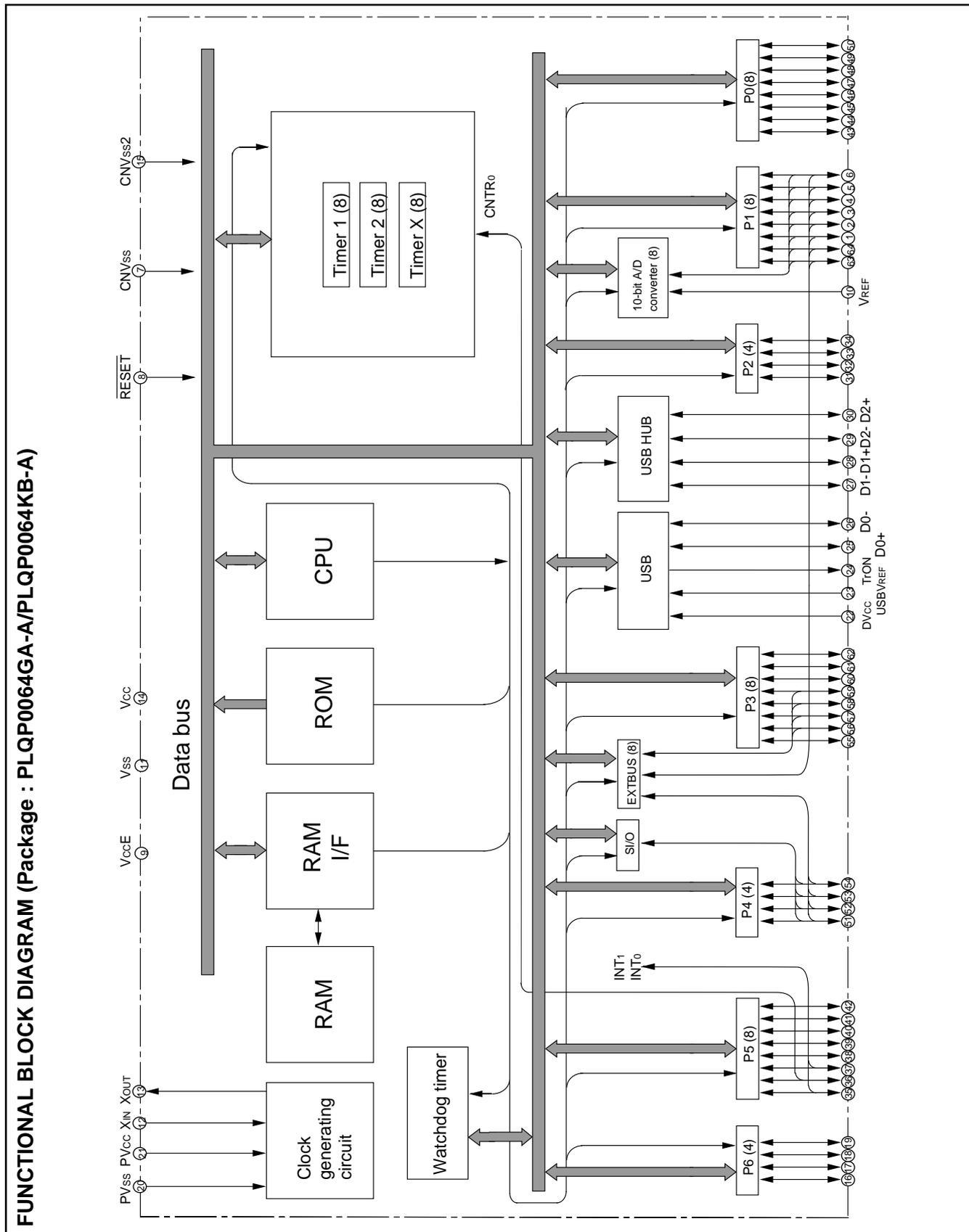


Fig. 2 Functional block diagram

## PIN DESCRIPTION

Table 1. Pin description

Pin	Name	Function	Function except a port function
VCC, VSS	Power source	<ul style="list-style-type: none"> <li>Apply voltage of 3.0 V – 5.25 V (L version) to VCC, and 0 V to VSS.</li> </ul>	
VCC_E	Analog power source	<ul style="list-style-type: none"> <li>Power source pin for ports P1, P3, P4 and analog circuit. Connect this pin to VCC.</li> </ul>	
CNVSS	CNVSS	<ul style="list-style-type: none"> <li>This pin controls the operation mode of the chip. Connect this pin to VSS. In the flash memory mode, this pin becomes VPP power source input pin.</li> </ul>	
CNVSS2	CNVSS2	<ul style="list-style-type: none"> <li>This pin controls the operation mode of the chip. Connect this pin to VSS.</li> </ul>	
VREF	Analog reference voltage input	<ul style="list-style-type: none"> <li>Reference voltage input pin for A/D converter.</li> </ul>	
DVCC PVCC, PVSS	Analog power source	<ul style="list-style-type: none"> <li>Power source pin for analog circuit.</li> <li>Connect the DVCC and PVCC pins to VCC, and the PVSS pin to VSS.</li> </ul>	
RESET	Reset input	<ul style="list-style-type: none"> <li>Reset input pin for active “L”</li> </ul>	
XIN	Clock input	<ul style="list-style-type: none"> <li>Input and output pins for the main clock generating circuit.</li> <li>Connect a ceramic resonator or a quartz-crystal oscillator between the XIN and XOUT pins to set the oscillation frequency.</li> </ul>	
XOUT	Clock output	<ul style="list-style-type: none"> <li>If an external clock is used, connect the clock source to the XIN pin and leave the XOUT pin open.</li> </ul>	
USBVREF	USB reference power source	<ul style="list-style-type: none"> <li>Power source pin for USB port circuit.</li> <li>In Vcc = 4.00 to 5.25 V use the built-in USB reference voltage circuit. In Vcc = 3.60 to 4.00 V apply 3.3 V power supply from the external because use of the built-in USB reference voltage circuit is prohibited in this voltage range. In Vcc = 3.00 to 3.60 V connect this pin to VCC because use of the built-in USB reference voltage circuit is prohibited in this voltage range.</li> </ul>	
TrON	USB reference voltage output	<ul style="list-style-type: none"> <li>Output pin to pull-up D0+ by 1.5 kΩ external resistor.</li> </ul>	
D0+, D0-	USB upstream I/O	<ul style="list-style-type: none"> <li>USB upstream I/O port</li> <li>USB input level</li> <li>USB output level output structure</li> </ul>	
D1+, D1-, D2+, D2-	USB downstream I/O	<ul style="list-style-type: none"> <li>USB downstream I/O port</li> <li>USB input level</li> <li>USB output level output structure</li> </ul>	
P00–P07	I/O port P0	<ul style="list-style-type: none"> <li>8-bit I/O port</li> <li>I/O direction register allows each pin to be individually programmed as either input or output.</li> <li>CMOS compatible input level</li> <li>CMOS 3-state output structure</li> <li>Pull-up control is enabled.</li> </ul>	<ul style="list-style-type: none"> <li>Key input pins (key-on wake up interrupt)</li> </ul>
P10/DQ0/AN0– P17/DQ7/AN7	I/O port P1	<ul style="list-style-type: none"> <li>8-bit I/O port</li> <li>I/O direction register allows each pin to be individually programmed as either input or output.</li> <li>CMOS compatible input level</li> <li>CMOS 3-state output structure</li> </ul>	<ul style="list-style-type: none"> <li>A/D converter input pins</li> <li>External bus interface function pins</li> </ul>
P24–P27	I/O port P2	<ul style="list-style-type: none"> <li>4-bit I/O port</li> <li>I/O direction register allows each pin to be individually programmed as either input or output.</li> <li>CMOS compatible input level</li> <li>CMOS 3-state output structure</li> </ul>	
P30–P32	I/O port P3	<ul style="list-style-type: none"> <li>8-bit I/O port</li> <li>I/O direction register allows each pin to be individually programmed as either input or output.</li> <li>CMOS compatible input level</li> <li>CMOS 3-state output structure</li> </ul>	<ul style="list-style-type: none"> <li>External bus interface function pins</li> </ul>
P33/ExINT P34/ExCS P35/ExWR P36/ExRD P37/ExA0			
P40/ExDREQ/RxD P41/ExDACK/TxD P42/ExTC/SCLK P43/ExA1/SRDY	I/O port P4	<ul style="list-style-type: none"> <li>4-bit I/O port</li> <li>I/O direction register allows each pin to be individually programmed as either input or output.</li> <li>CMOS compatible input level</li> <li>CMOS 3-state output structure</li> </ul>	<ul style="list-style-type: none"> <li>Serial I/O function pins</li> <li>External bus interface function pins</li> </ul>
P50/INT0	I/O port P5	<ul style="list-style-type: none"> <li>8-bit I/O port</li> <li>I/O direction register allows each pin to be individually programmed as either input or output.</li> <li>CMOS compatible input level</li> <li>CMOS 3-state output structure</li> </ul>	<ul style="list-style-type: none"> <li>Interrupt input pin</li> </ul>
P51/CNTR0			<ul style="list-style-type: none"> <li>Timer X function pin</li> </ul>
P52/INT1			<ul style="list-style-type: none"> <li>Interrupt input pin</li> </ul>
P53–P57			
P60–P63	I/O port P6	<ul style="list-style-type: none"> <li>4-bit I/O port; • I/O direction register allows each pin to be individually programmed as either input or output.; • CMOS compatible input level• CMOS 3-state output structure;</li> <li>Output large current for LED drive is enabled.</li> </ul>	

## PART NUMBERING

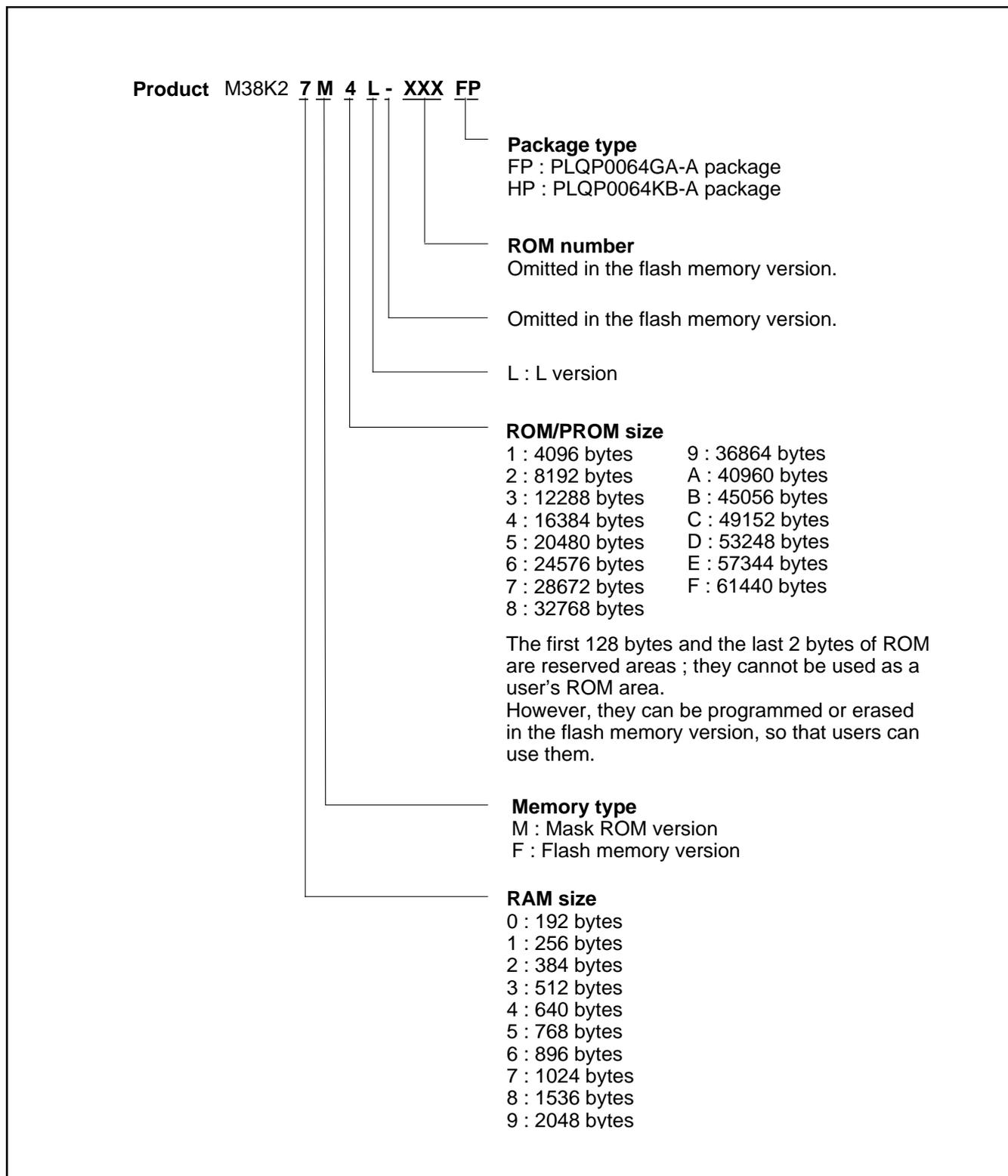


Fig. 3 Part numbering

**GROUP EXPANSION**

Mitsubishi plans to expand the 38K2 group as follows.

**Memory Type**

Support for mask ROM and flash memory versions.

**Memory Size**

Flash memory size ..... 32 Kbytes

Mask ROM size ..... 16 Kbytes

RAM size ..... 1024 to 2048 bytes

**Packages**

PLQP0064GA-A ..... 0.8 mm-pitch plastic molded LQFP

PLQP0064KB-A ..... 0.5 mm-pitch plastic molded LQFP

100D0M ..... 0.65 mm-pitch metal seal PIGGY BACK

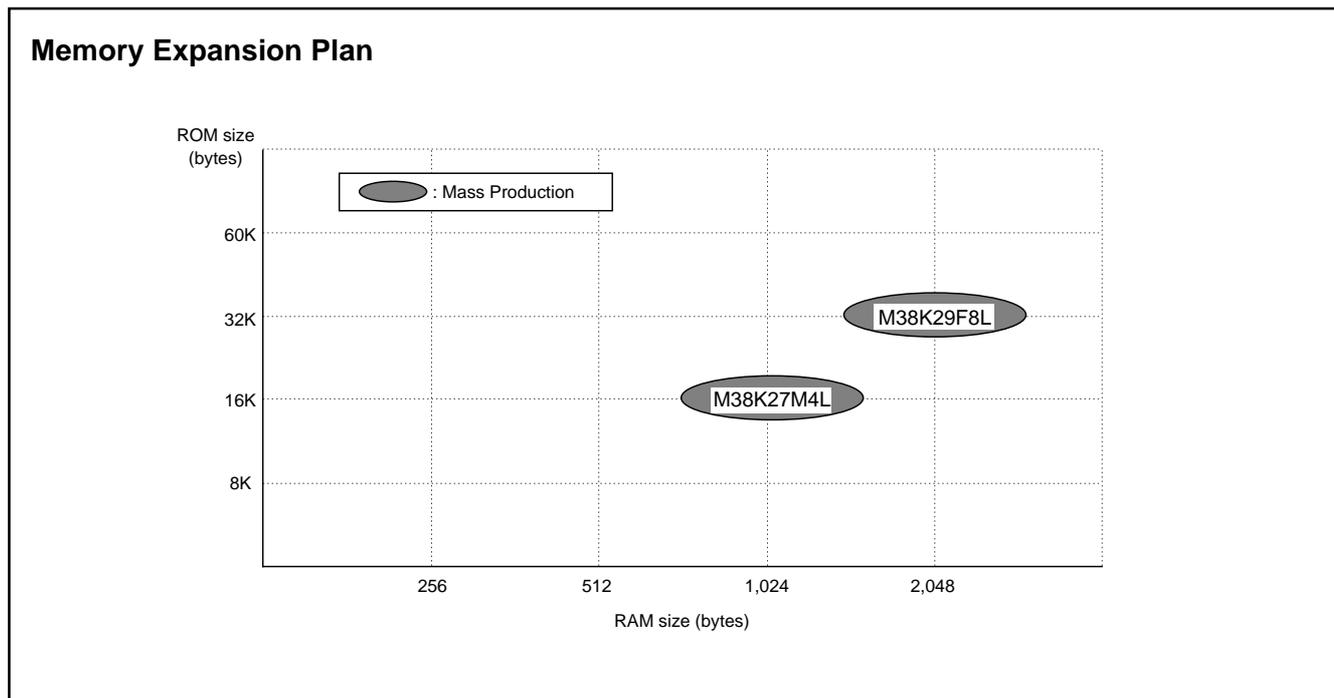


Fig. 4 Memory expansion plan

Currently products are listed below.

Table 2. List of 38K2 group products (L version)

As of October 2006

Product	ROM size (bytes) ROM size for User in ( )	RAM size (bytes)	Package	Remarks
M38K27M4L-XXXFP	16384	1024	PLQP0064GA-A	Mask ROM version
M38K27M4L-XXXHP	(16254)		PLQP0064KB-A	
M38K29F8LFP	32768	2048	PLQP0064GA-A	Flash memory version
M38K29F8LHP	(32638)		PLQP0064KB-A	
M38K29RFS	—	2048	100D0M	

## FUNCTIONAL DESCRIPTION CENTRAL PROCESSING UNIT (CPU)

The 38K2 group uses the standard 740 family instruction set. Refer to the table of 740 family addressing modes and machine instructions or the 740 Family Software Manual for details on the instruction set.

Machine-resident 740 family instructions are as follows:

The FST and SLW instruction cannot be used.

The STP, WIT, MUL, and DIV instruction can be used.

The CPU has the 6 registers. The register structure is shown in Figure 5.

### [Accumulator (A)]

The accumulator is an 8-bit register. Data operations such as data transfer, etc., are executed mainly through the accumulator.

### [Index Register X (X)]

The index register X is an 8-bit register. In the index addressing modes, the value of the OPERAND is added to the contents of register X and specifies the real address.

### [Index Register Y (Y)]

The index register Y is an 8-bit register. In partial instruction, the value of the OPERAND is added to the contents of register Y and specifies the real address.

### [Stack Pointer (S)]

The stack pointer is an 8-bit register used during subroutine calls and interrupts. This register indicates start address of stored area (stack) for storing registers during subroutine calls and interrupts. The low-order 8 bits of the stack address are determined by the contents of the stack pointer. The high-order 8 bits of the stack address are determined by the stack page selection bit. If the stack page selection bit is "0", the high-order 8 bits becomes "0016". If the stack page selection bit is "1", the high-order 8 bits becomes "0116".

Figure 6 shows the store and the return movement into the stack. If there are registers other than those described in Figure 5, the users need to store them with the program.

### [Program Counter (PC)]

The program counter is a 16-bit counter consisting of two 8-bit registers PCH and PCL. It is used to indicate the address of the next instruction to be executed.

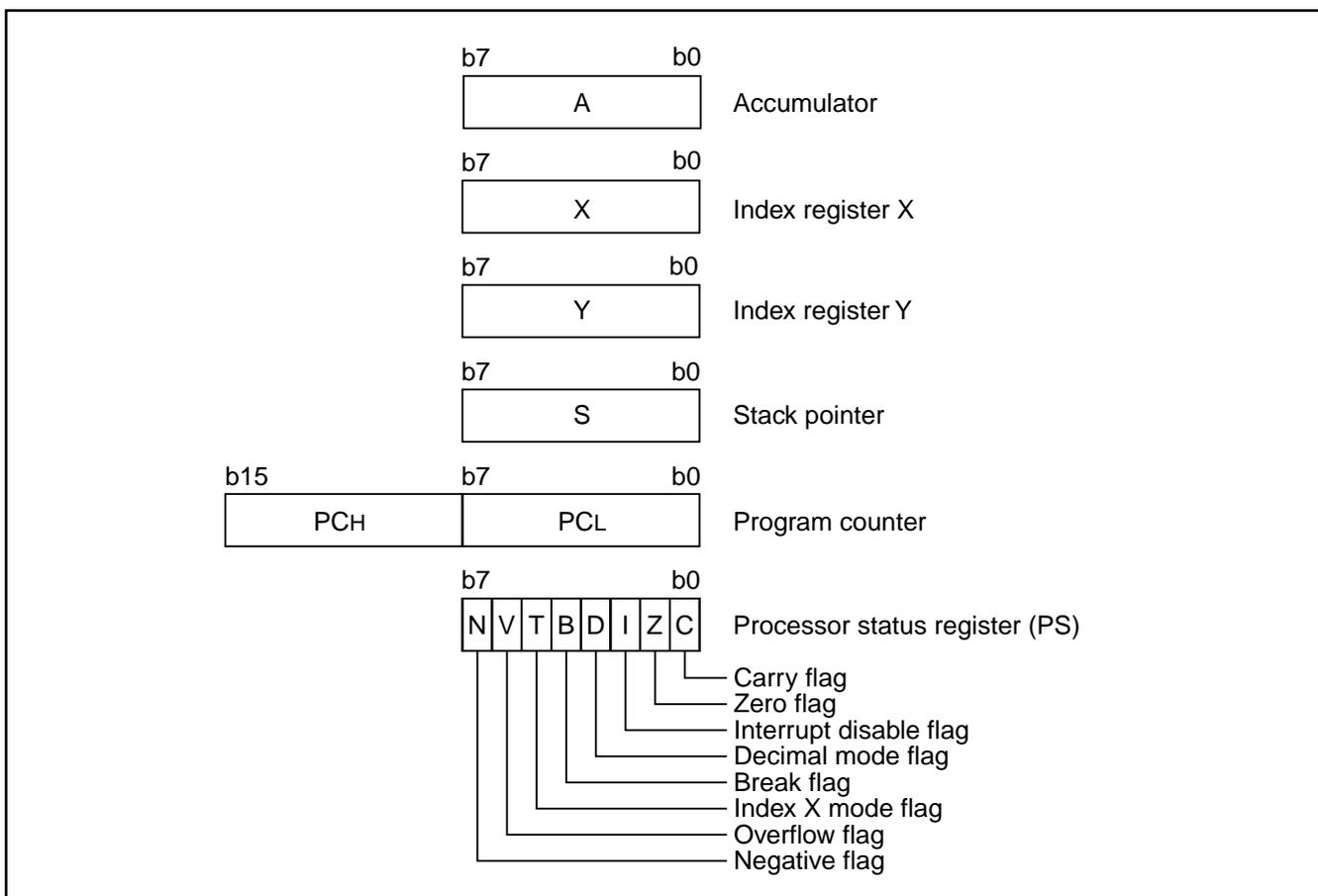


Fig. 5 740 Family CPU register structure

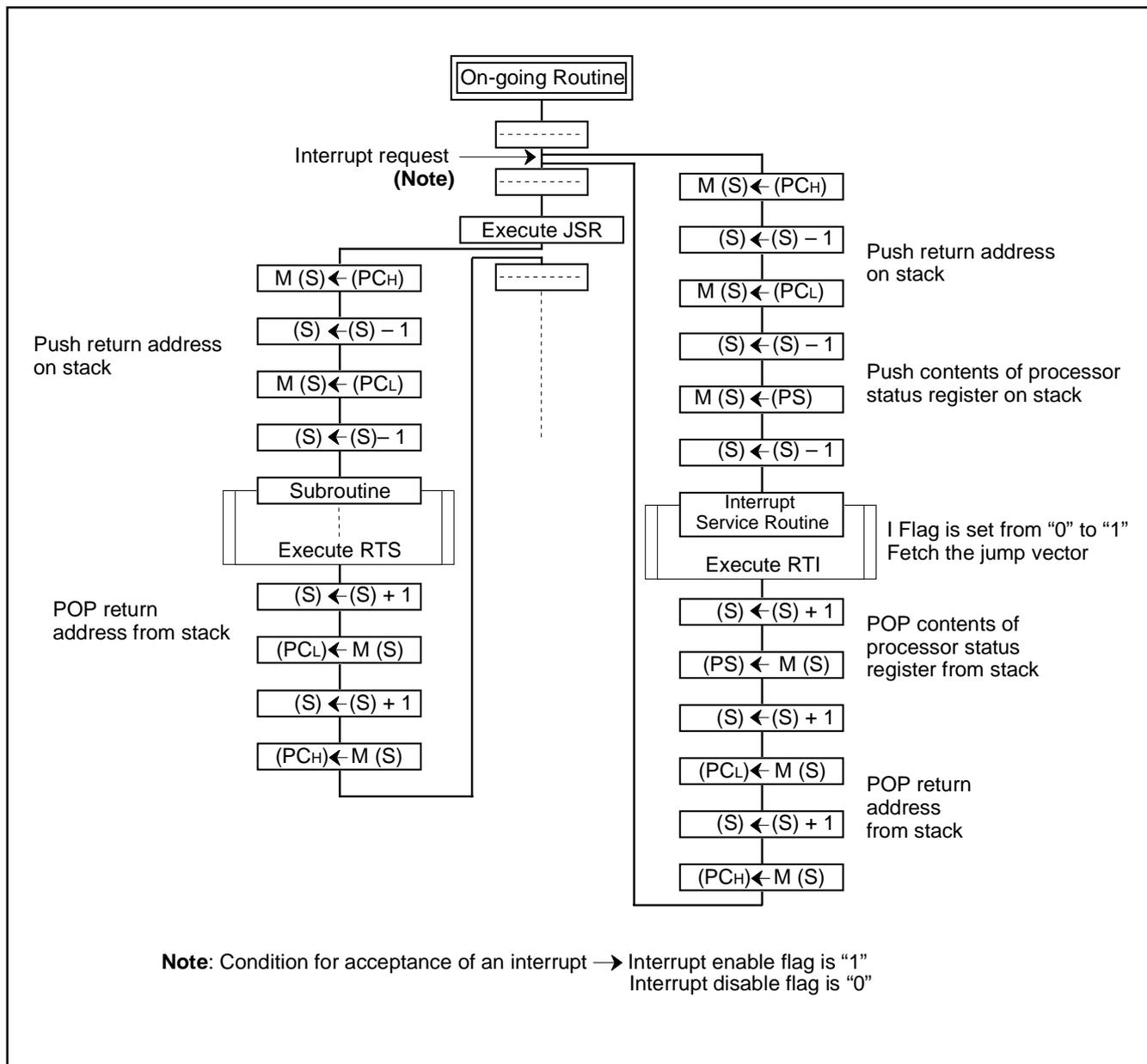


Fig. 6 Register push and pop at interrupt generation and subroutine call

Table 3 Push and pop instructions of accumulator or processor status register

	Push instruction to stack	Pop instruction from stack
Accumulator	PHA	PLA
Processor status register	PHP	PLP

### [Processor status register (PS)]

The processor status register is an 8-bit register consisting of 5 flags which indicate the status of the processor after an arithmetic operation and 3 flags which decide MCU operation. Branch operations can be performed by testing the Carry (C) flag, Zero (Z) flag, Overflow (V) flag, or the Negative (N) flag. In decimal mode, the Z, V, N flags are not valid.

•Bit 0: Carry flag (C)

The C flag contains a carry or borrow generated by the arithmetic logic unit (ALU) immediately after an arithmetic operation. It can also be changed by a shift or rotate instruction.

•Bit 1: Zero flag (Z)

The Z flag is set if the result of an immediate arithmetic operation or a data transfer is "0", and cleared if the result is anything other than "0".

•Bit 2: Interrupt disable flag (I)

The I flag disables all interrupts except for the interrupt generated by the BRK instruction.

Interrupts are disabled when the I flag is "1".

•Bit 3: Decimal mode flag (D)

The D flag determines whether additions and subtractions are executed in binary or decimal. Binary arithmetic is executed when this flag is "0"; decimal arithmetic is executed when it is "1". Decimal correction is automatic in decimal mode. Only the ADC and SBC instructions can execute decimal arithmetic.

•Bit 4: Break flag (B)

The B flag is used to indicate that the current interrupt was generated by the BRK instruction. The BRK flag in the processor status register is always "0". When the BRK instruction is used to generate an interrupt, the processor status register is pushed onto the stack with the break flag set to "1".

•Bit 5: Index X mode flag (T)

When the T flag is "0", arithmetic operations are performed between accumulator and memory. When the T flag is "1", direct arithmetic operations and direct data transfers are enabled between memory locations.

•Bit 6: Overflow flag (V)

The V flag is used during the addition or subtraction of one byte of signed data. It is set if the result exceeds +127 to -128. When the BIT instruction is executed, bit 6 of the memory location operated on by the BIT instruction is stored in the overflow flag.

•Bit 7: Negative flag (N)

The N flag is set if the result of an arithmetic operation or data transfer is negative. When the BIT instruction is executed, bit 7 of the memory location operated on by the BIT instruction is stored in the negative flag.

**Table 4 Set and clear instructions of each bit of processor status register**

	C flag	Z flag	I flag	D flag	B flag	T flag	V flag	N flag
Set instruction	SEC	–	SEI	SED	–	SET	–	–
Clear instruction	CLC	–	CLI	CLD	–	CLT	CLV	–

**[CPU Mode Register (CPUM)] 003B16**

The CPU mode register contains the stack page selection bit and the internal system clock selection bit.

The CPU mode register is allocated at address 003B16.

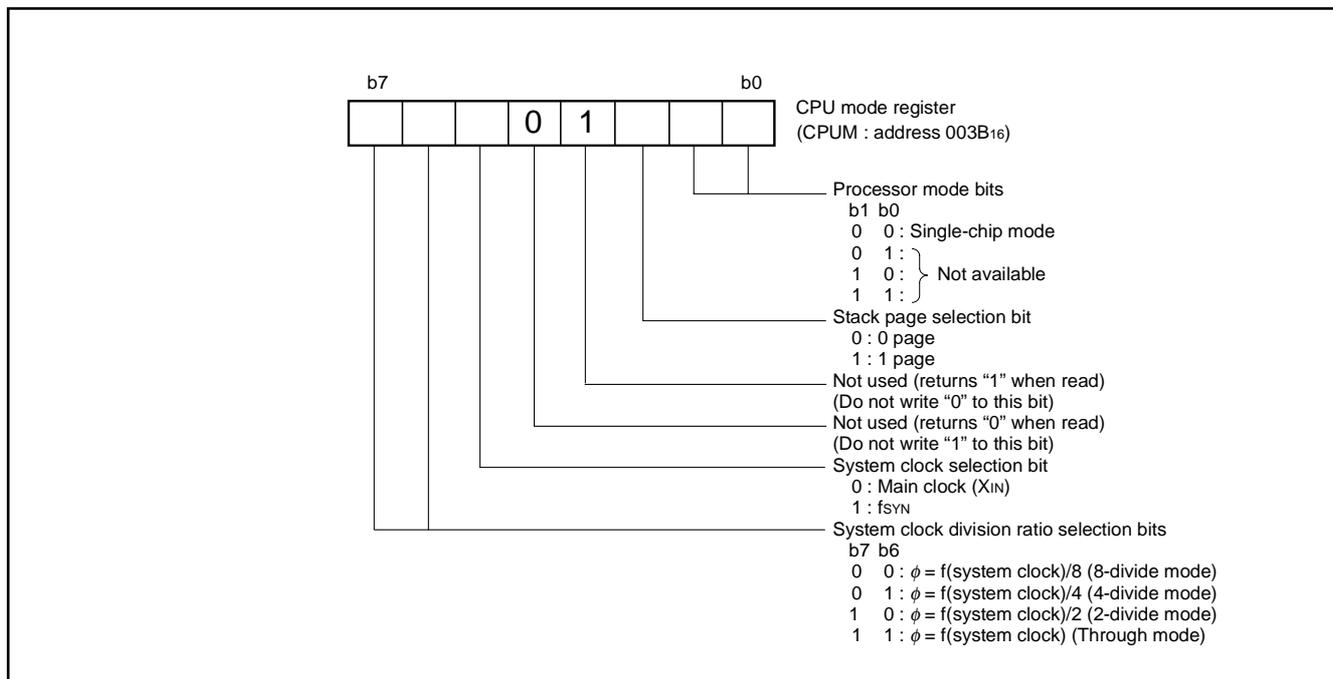


Fig. 7 Structure of CPU mode register

**MEMORY****Special Function Register (SFR) Area**

The Special Function Register area in the zero page contains control registers such as I/O ports and timers.

**RAM**

RAM is used for data storage and for stack area of subroutine calls and interrupts.

**ROM**

The first 128 bytes and the last 2 bytes of ROM are reserved for device testing and the rest is user area for storing programs. In the flash memory version, program and erase can be performed in the reserved area.

**Interrupt Vector Area**

The interrupt vector area contains reset and interrupt vectors.

**Zero Page**

The 256 bytes from addresses  $0000_{16}$  to  $00FF_{16}$  are called the zero page area. The internal RAM and the special function registers (SFR) are allocated to this area.

The zero page addressing mode can be used to specify memory and register addresses in the zero page area. Access to this area with only 2 bytes is possible in the zero page addressing mode.

**Special Page**

The 256 bytes from addresses  $FF00_{16}$  to  $FFFF_{16}$  are called the special page area. The special page addressing mode can be used to specify memory addresses in the special page area. Access to this area with only 2 bytes is possible in the special page addressing mode.

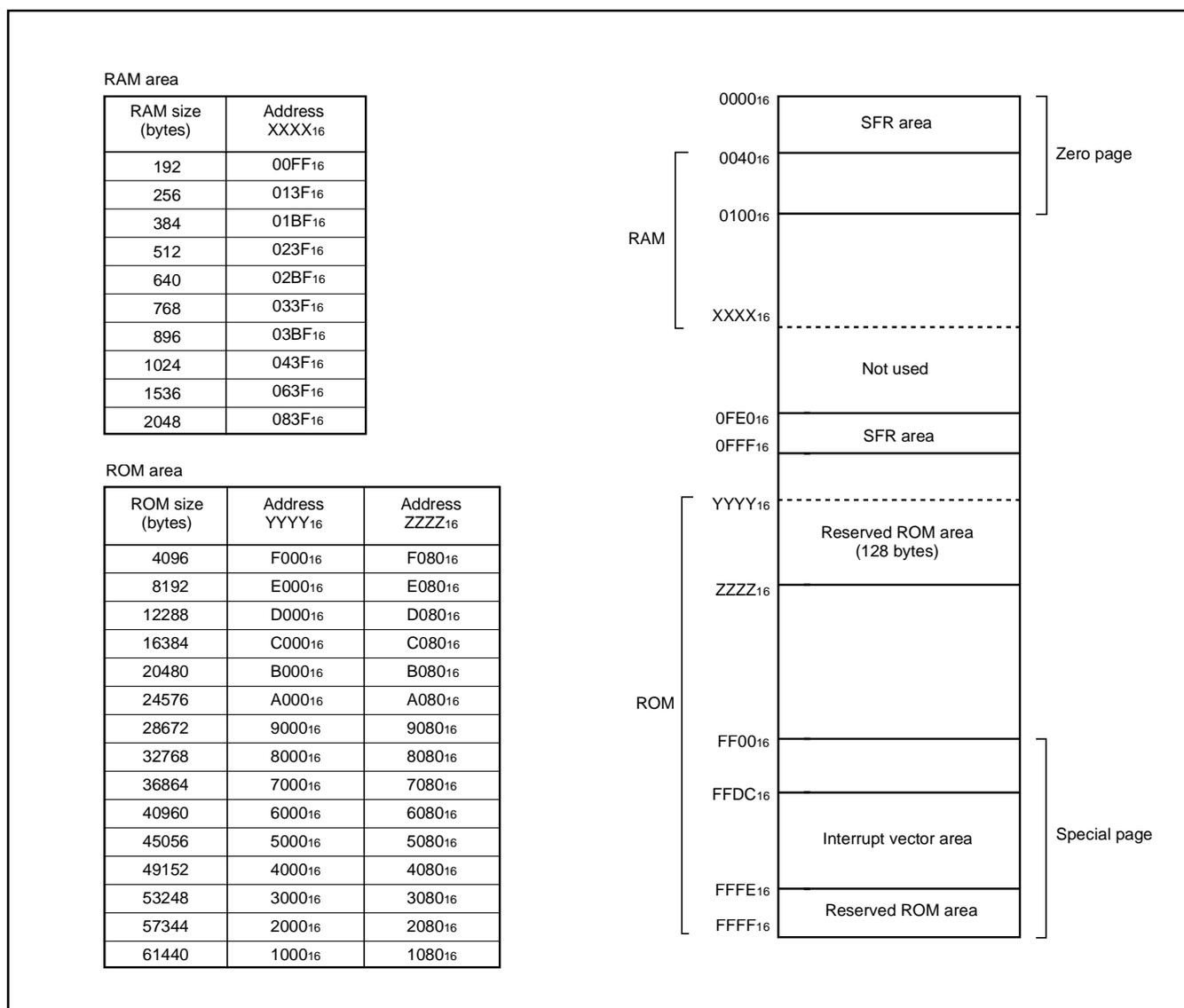


Fig. 8 Memory map diagram

0000 <sub>16</sub>	Port P0 (P0)	0020 <sub>16</sub>	Prescaler 12 (PRE12)
0001 <sub>16</sub>	Port P0 direction register (P0D)	0021 <sub>16</sub>	Timer 1 (T1)
0002 <sub>16</sub>	Port P1 (P1)	0022 <sub>16</sub>	Timer 2 (T2)
0003 <sub>16</sub>	Port P1 direction register (P1D)	0023 <sub>16</sub>	Timer X mode register (TM)
0004 <sub>16</sub>	Port P2 (P2)	0024 <sub>16</sub>	Prescaler X (PREX)
0005 <sub>16</sub>	Port P2 direction register (P2D)	0025 <sub>16</sub>	Timer X (TX)
0006 <sub>16</sub>	Port P3 (P3)	0026 <sub>16</sub>	Transmit/Receive buffer register (TB/RB)
0007 <sub>16</sub>	Port P3 direction register (P3D)	0027 <sub>16</sub>	Serial I/O status register (SIOSTS)
0008 <sub>16</sub>	Port P4 (P4)	0028 <sub>16</sub>	HUB interrupt source enable register (HUBICON)
0009 <sub>16</sub>	Port P4 direction register (P4D)	0029 <sub>16</sub>	HUB interrupt source register (HUBIREQ)
000A <sub>16</sub>	Port P5 (P5)	002A <sub>16</sub>	HUB down stream port index register (HUBINDEX)
000B <sub>16</sub>	Port P5 direction register (P5D)	002B <sub>16</sub>	HUB port field register 1 (DPXREG1)
000C <sub>16</sub>	Port P6 (P6)	002C <sub>16</sub>	HUB port field register 2 (DPXREG2)
000D <sub>16</sub>	Port P6 direction register (P6D)	002D <sub>16</sub>	HUB port field register 3 (DPXREG3)
000E <sub>16</sub>	Reserved ( <b>Note</b> )	002E <sub>16</sub>	Reserved ( <b>Note</b> )
000F <sub>16</sub>	Reserved ( <b>Note</b> )	002F <sub>16</sub>	Reserved ( <b>Note</b> )
0010 <sub>16</sub>	USB control register (USBCON)	0030 <sub>16</sub>	EXB interrupt source enable register (EXBICON)
0011 <sub>16</sub>	USB function/Hub enable register (USBAE)	0031 <sub>16</sub>	EXB interrupt source register (EXBIREQ)
0012 <sub>16</sub>	USB function address register (USBA0)	0032 <sub>16</sub>	Reserved ( <b>Note</b> )
0013 <sub>16</sub>	USB HUB address register (USBA1)	0033 <sub>16</sub>	EXB index register (EXBINDEX)
0014 <sub>16</sub>	Frame number register Low (FNUML)	0034 <sub>16</sub>	Register window 1 (EXBREG1)
0015 <sub>16</sub>	Frame number register High (FNUMH)	0035 <sub>16</sub>	Register window 2 (EXBREG2)
0016 <sub>16</sub>	USB interrupt source enable register (USBICON)	0036 <sub>16</sub>	AD control register (ADCON)
0017 <sub>16</sub>	USB interrupt source register (USBIREQ)	0037 <sub>16</sub>	AD conversion register 1 (AD1)
0018 <sub>16</sub>	Endpoint index register (USBINDEX)	0038 <sub>16</sub>	AD conversion register 2 (AD2)
0019 <sub>16</sub>	Endpoint field register 1 (EPXXREG1)	0039 <sub>16</sub>	Watchdog timer control register (WDTCN)
001A <sub>16</sub>	Endpoint field register 2 (EPXXREG2)	003A <sub>16</sub>	Reserved ( <b>Note</b> )
001B <sub>16</sub>	Endpoint field register 3 (EPXXREG3)	003B <sub>16</sub>	CPU mode register (CPUM)
001C <sub>16</sub>	Endpoint field register 4 (EPXXREG4)	003C <sub>16</sub>	Interrupt request register 1(IREQ1)
001D <sub>16</sub>	Endpoint field register 5 (EPXXREG5)	003D <sub>16</sub>	Interrupt request register 2(IREQ2)
001E <sub>16</sub>	Endpoint field register 6 (EPXXREG6)	003E <sub>16</sub>	Interrupt control register 1(ICON1)
001F <sub>16</sub>	Endpoint field register 7 (EPXXREG7)	003F <sub>16</sub>	Interrupt control register 2(ICON2)
0FE0 <sub>16</sub>	Serial I/O control register (SIOCON)	0FF0 <sub>16</sub>	Port P0 pull-up control register (PULL0)
0FE1 <sub>16</sub>	UART control register (UARTCON)	0FF1 <sub>16</sub>	Reserved ( <b>Note</b> )
0FE2 <sub>16</sub>	Baud rate generator (BRG)	0FF2 <sub>16</sub>	Port P5 pull-up control register (PULL5)
0FE3 <sub>16</sub>	Reserved ( <b>Note</b> )	0FF3 <sub>16</sub>	Interrupt edge selection register (INTEDGE)
0FE4 <sub>16</sub>	Reserved ( <b>Note</b> )	0FF4 <sub>16</sub>	Reserved ( <b>Note</b> )
0FE5 <sub>16</sub>	Reserved ( <b>Note</b> )	0FF5 <sub>16</sub>	Reserved ( <b>Note</b> )
0FE6 <sub>16</sub>	Reserved ( <b>Note</b> )	0FF6 <sub>16</sub>	Reserved ( <b>Note</b> )
0FE7 <sub>16</sub>	Reserved ( <b>Note</b> )	0FF7 <sub>16</sub>	Reserved ( <b>Note</b> )
0FE8 <sub>16</sub>	Reserved ( <b>Note</b> )	0FF8 <sub>16</sub>	PLL control register (PLLCON)
0FE9 <sub>16</sub>	Reserved ( <b>Note</b> )	0FF9 <sub>16</sub>	Downstream port control register (DPCTL)
0FEA <sub>16</sub>	Reserved ( <b>Note</b> )	0FFA <sub>16</sub>	Reserved ( <b>Note</b> )
0FEB <sub>16</sub>	Reserved ( <b>Note</b> )	0FFB <sub>16</sub>	MISRG
0FEC <sub>16</sub>	Endpoint field register 8 (EPXXREG8)	0FFC <sub>16</sub>	Reserved ( <b>Note</b> )
0FED <sub>16</sub>	Endpoint field register 9 (EPXXREG9)	0FFD <sub>16</sub>	Reserved ( <b>Note</b> )
0FEE <sub>16</sub>	Reserved ( <b>Note</b> )	0FFE <sub>16</sub>	Flash memory control register (FMCR)
0FEF <sub>16</sub>	Reserved ( <b>Note</b> )	0FFF <sub>16</sub>	Reserved ( <b>Note</b> )

**Note:** Do not write any data to these addresses, because these areas are reserved.

Fig. 9 Memory map of special function register (SFR)

## I/O PORTS

The I/O ports have direction registers which determine the input/output direction of each individual pin. Each bit in a direction register corresponds to one pin, and each pin can be set to be input port or output port.

When "0" is written to the bit corresponding to a pin, that pin becomes an input pin. When "1" is written to that bit, that pin becomes an output pin.

If data is read from a pin set to output, the value of the port output latch is read, not the value of the pin itself. Pins set to input are floating. If a pin set to input is written to, only the port output latch is written to and the pin remains floating.

**Table 5 I/O ports functions**

Pin	Name	Input/Output	I/O Format	Non-Port Function	Related SFRs	Diagram No.
P00–P07	Port P0	Input/output, individual bits	CMOS compatible input level CMOS 3-state output	Key-on wake up	Port P0 pull-up control register	(1)
P10–P17	Port P1			A/D conversion input External bus interface function I/O	AD control register EXB control register	(2)
P24–P27	Port P2			—————	—————	(3)
P30–P32	Port P3			—————	—————	(4)
P33/ExINT				External bus interface function output	EXB control register	(5)
P34/ExCS P35/ExWR P36/ExRD P37/ExA0				External bus interface function input	EXB control register	(6)
P40/RxD/ ExDREQ				Serial I/O input External bus interface function output	Serial I/O control register EXB control register	(7)
P41/TxD/ ExDACK	Serial I/O output External bus interface function input			Serial I/O control register EXB control register	(8)	
P42/SCLK/ ExTC	Serial I/O I/O External bus interface function input			Serial I/O control register EXB control register	(9)	
P43/SRDY/ ExA1	Serial I/O output External bus interface function input			Serial I/O control register EXB control register	(10)	
P50/INT0 P52/INT1	Port P5			External interrupt input	Port P5 pull-up control register Interrupt edge selection register	(11)
P51/CNTR0				Timer X function I/O	Timer X mode register	(12)
P53–P57				—————	—————	(13)
P60–P63	Port P6			—————	—————	(14)

**Note:** Make sure that the input level at each pin is either 0 V or V<sub>CC</sub> during execution of the STP instruction. When an input level is at an intermediate potential, a current will flow from V<sub>CC</sub> to V<sub>SS</sub> through the input-stage gate.

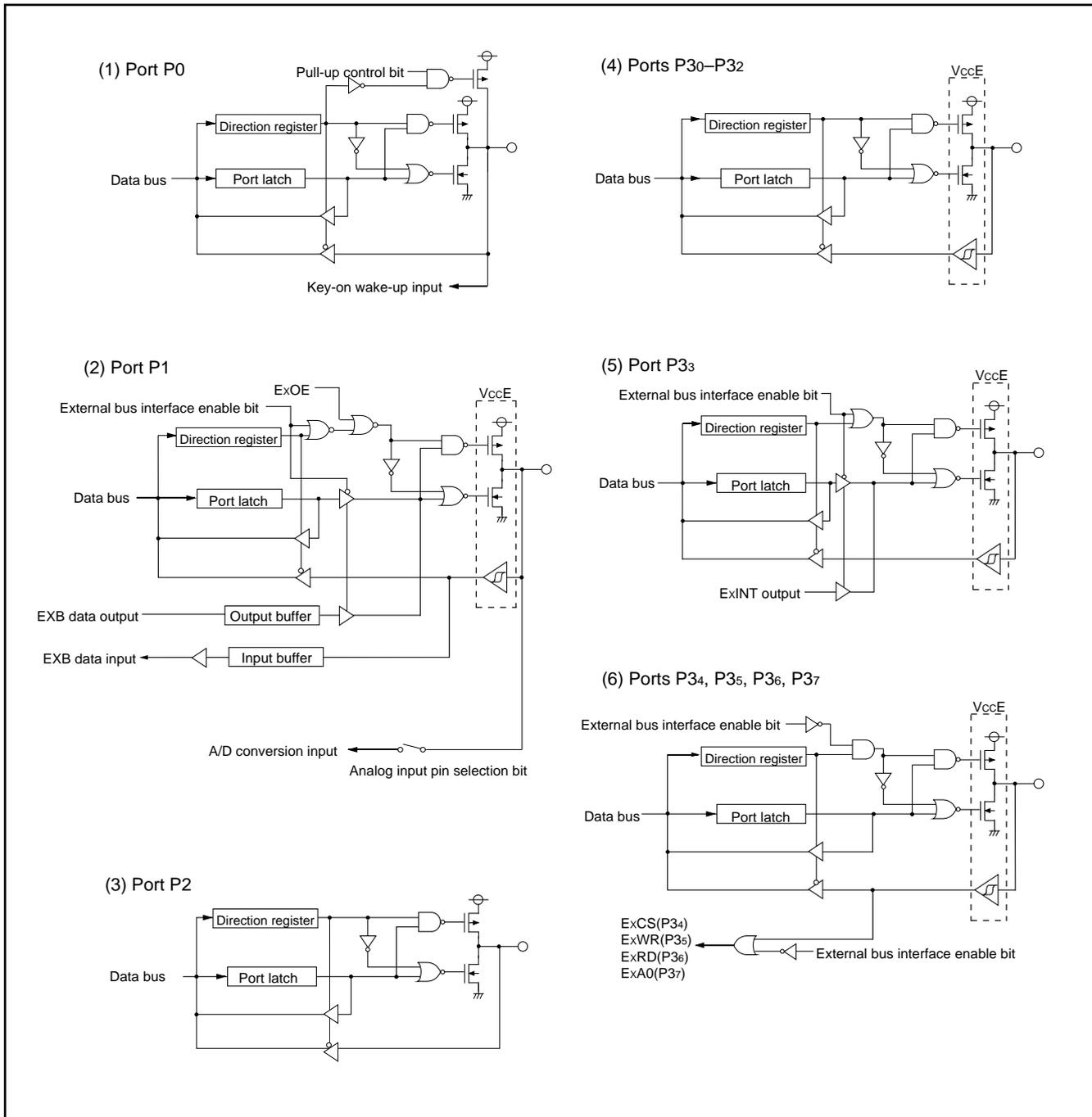


Fig. 10 Port block diagram (1)

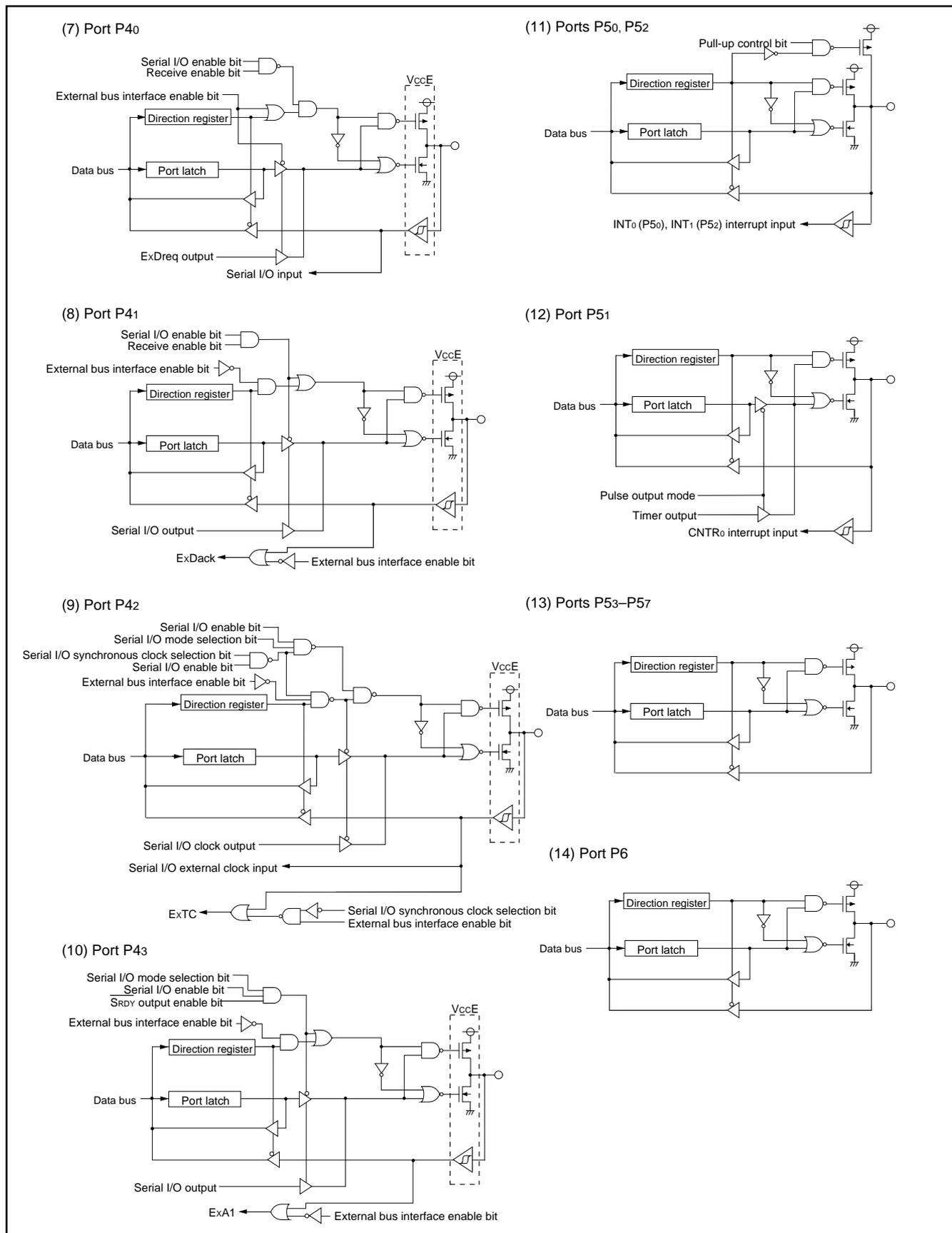


Fig. 11 Port block diagram (2)

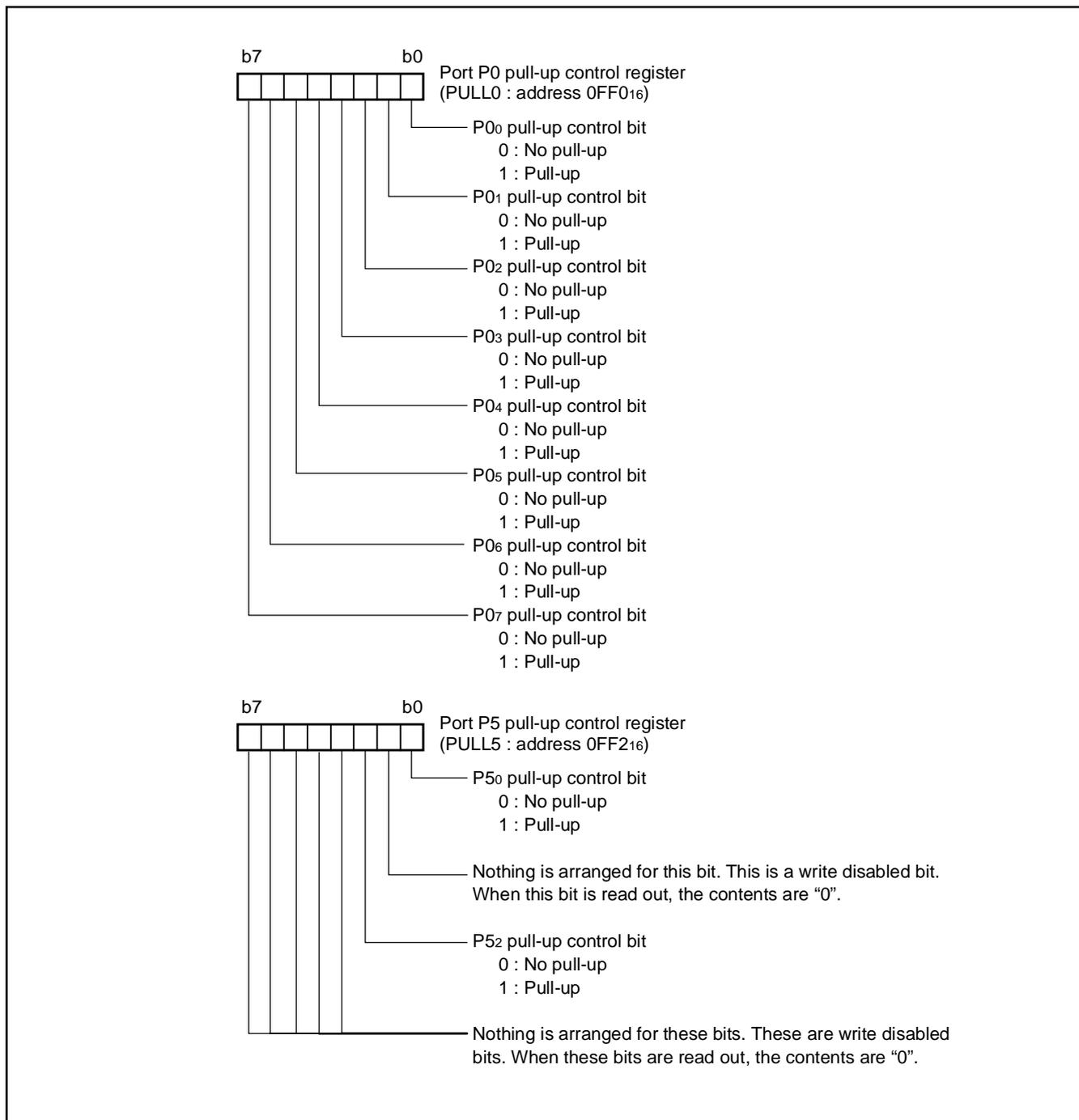


Fig. 12 Structure of port I/O-related registers

## INTERRUPTS

Interrupts occur by sixteen sources: four external, eleven internal, and one software.

### Interrupt Control

Each interrupt is controlled by an interrupt request bit, an interrupt enable bit, and the interrupt disable flag except for the software interrupt set by the BRK instruction. An interrupt occurs if the corresponding interrupt request and enable bits are "1" and the interrupt disable flag is "0".

Interrupt enable bits can be set or cleared by software.

Interrupt request bits can be cleared by software, but cannot be set by software.

The BRK instruction cannot be disabled with any flag or bit. The I flag disables all interrupts except the BRK instruction interrupt.

When several interrupts occur at the same time, the interrupts are received according to priority.

### Interrupt Operation

By acceptance of an interrupt, the following operations are automatically performed:

1. The contents of the program counter and the processor status register are automatically pushed onto the stack.
2. The interrupt disable flag is set and the corresponding interrupt request bit is cleared.
3. The interrupt jump destination address is read from the vector table into the program counter.

**Table 6 Interrupt vector addresses and priority**

Interrupt Source	Priority	Vector Addresses (Note 1)		Interrupt Request Generating Conditions
		High	Low	
Reset (Note 2)	1	FFFD <sub>16</sub>	FFFC <sub>16</sub>	At reset
USB bus reset	2	FFFB <sub>16</sub>	FFFA <sub>16</sub>	At detection of USB bus reset signal (2.5 $\mu$ s interval SE0)
USB SOF	3	FFF9 <sub>16</sub>	FFF8 <sub>16</sub>	At detection of USB SOF signal
USB device	4	FFF7 <sub>16</sub>	FFF6 <sub>16</sub>	At detection of resume signal (K state or SE0) or suspend signal (3 ms interval bus idle), or at completion of transaction
External bus	5	FFF5 <sub>16</sub>	FFF4 <sub>16</sub>	At completion of reception or transmission or at completion of DMA transmission
INT <sub>0</sub>	6	FFF3 <sub>16</sub>	FFF2 <sub>16</sub>	At detection of either rising or falling edge of INT <sub>0</sub> input
Timer X	7	FFF1 <sub>16</sub>	FFF0 <sub>16</sub>	At timer X underflow
Timer 1	8	FFEF <sub>16</sub>	FFEE <sub>16</sub>	At timer 1 underflow
Timer 2	9	FFED <sub>16</sub>	FFEC <sub>16</sub>	At timer 2 underflow
INT <sub>1</sub>	10	FFEB <sub>16</sub>	FFEA <sub>16</sub>	At detection of either rising or falling edge of INT <sub>1</sub> input
USB HUB	11	FFE9 <sub>16</sub>	FFE8 <sub>16</sub>	At detection of USB HUB downport's state switch
Serial I/O reception	12	FFE7 <sub>16</sub>	FFE6 <sub>16</sub>	At completion of serial I/O data reception
Serial I/O transmission	13	FFE5 <sub>16</sub>	FFE4 <sub>16</sub>	At completion of serial I/O data transmission
CNTR <sub>0</sub>	14	FFE3 <sub>16</sub>	FFE2 <sub>16</sub>	At detection of either rising or falling edge of CNTR <sub>0</sub> input
Key-on wake up	15	FFE1 <sub>16</sub>	FFE0 <sub>16</sub>	At falling of conjunction of input level for port P0 (at input mode)
A/D conversion	16	FFDF <sub>16</sub>	FFDE <sub>16</sub>	At completion of A/D conversion
BRK instruction	17	FFDD <sub>16</sub>	FFDC <sub>16</sub>	At BRK instruction execution

**Notes 1:** Vector addresses contain interrupt jump destination addresses.

**2:** Reset function in the same way as an interrupt with the highest priority.

### ■Notes on interrupts

When setting the followings, the interrupt request bit may be set to "1".

- When switching external interrupt active edge

Related register: Interrupt edge selection register (address 0FF3<sub>16</sub>), Timer X mode register (address 0023<sub>16</sub>)

When not requiring for the interrupt occurrence synchronized with these setting, take the following sequence.

- ①Set the corresponding interrupt enable bit to "0" (disabled).
- ②Set the interrupt edge select bit (active edge switch bit).
- ③Set the corresponding interrupt request bit to "0" after 1 or more instructions have been executed.
- ④Set the corresponding interrupt enable bit to "1" (enabled).

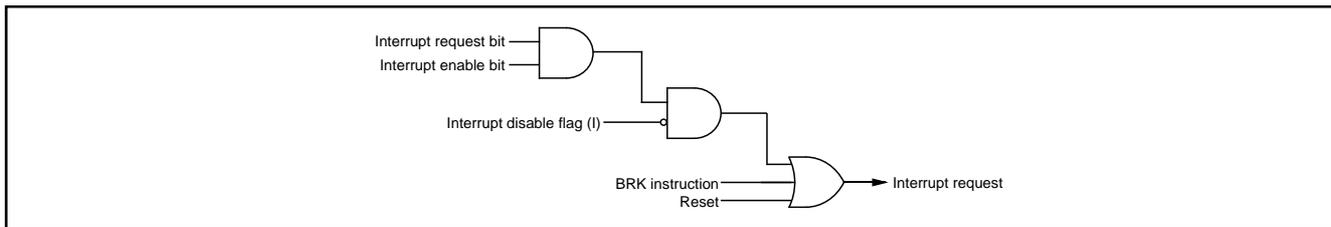


Fig. 13 Interrupt control

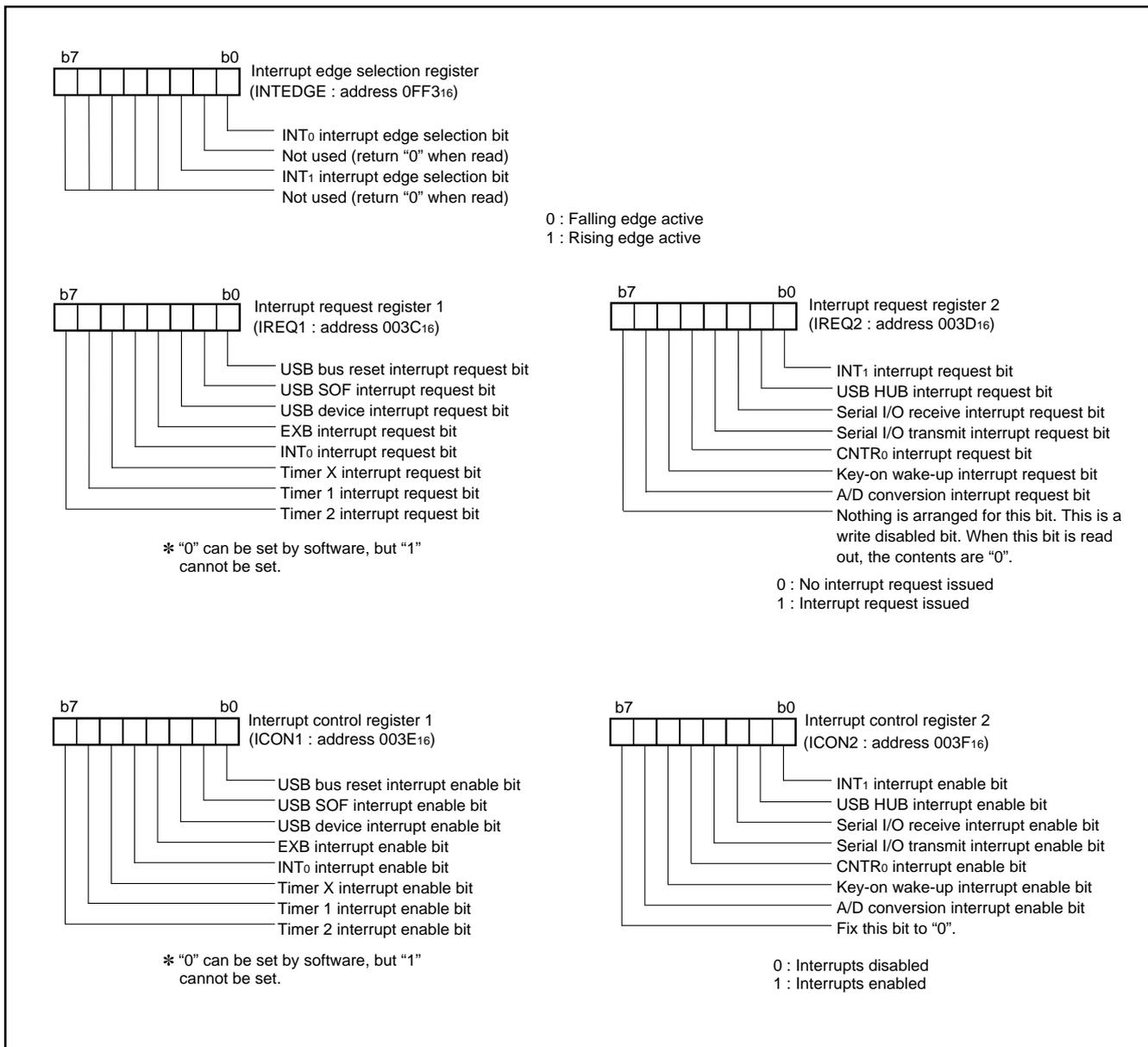


Fig. 14 Structure of interrupt-related registers

### Key Input Interrupt (Key-on Wake Up)

A Key-on wake up interrupt request is generated by applying a falling edge to any pin of port P0 that have been set to input mode. In other words, it is generated when AND of input level goes from

"1" to "0". An example of using a key input interrupt is shown in Figure 15, where an interrupt request is generated by pressing one of the keys consisted as an active-low key matrix which inputs to ports P00-P03.

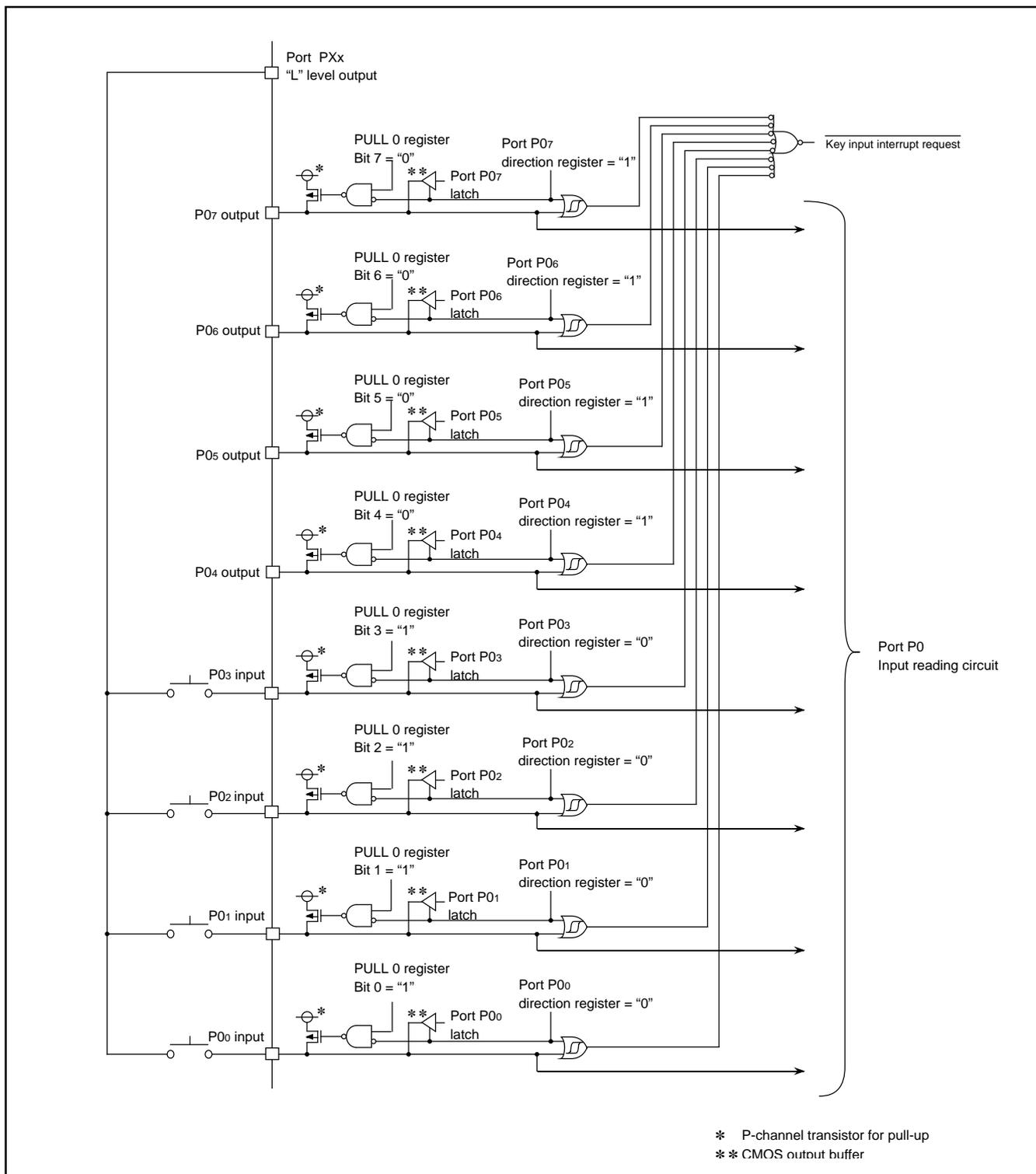


Fig. 15 Connection example when using key input interrupt and port P0 block diagram

## TIMERS

The 38K2 group has three timers: timer X, timer 1, and timer 2. The division ratio of each timer or prescaler is given by  $1/(n + 1)$ , where  $n$  is the value in the corresponding timer or prescaler latch. All timers are down count timers. When the timer reaches "0016", an underflow occurs at the next count pulse and the corresponding timer latch is reloaded into the timer and the count is continued. When a timer underflows, the interrupt request bit corresponding to that timer is set to "1".

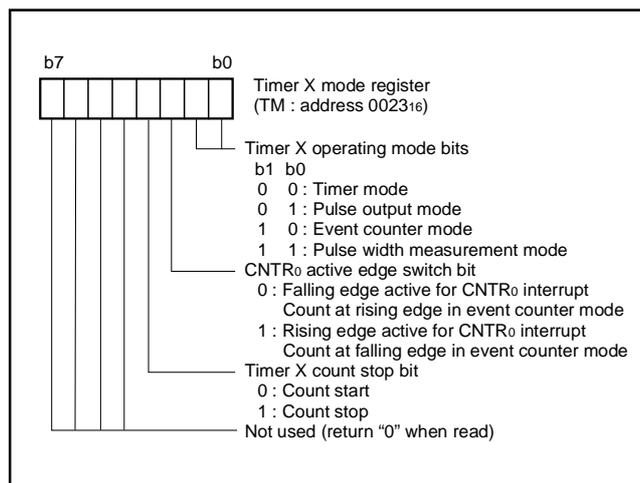


Fig. 16 Structure of timer X mode register

## Timer 1 and Timer 2

The count source of prescaler 12 is the system clock divided by 16. The output of prescaler 12 is counted by timer 1 and timer 2, and a timer underflow periodically sets the interrupt request bit.

## Timer X

Timer X can each select in one of four operating modes by setting the timer X mode register.

### (1) Timer Mode

The timer counts the count source selected by timer count source selection bit.

### (2) Pulse Output Mode

The timer counts the system clock divided by 16. Whenever the contents of the timer reach "0016", the signal output from the CNTR<sub>0</sub> pin is inverted. If the CNTR<sub>0</sub> active edge selection bit is "0", output begins at "H".

If it is "1", output starts at "L". When using a timer in this mode, set the corresponding port P5<sub>1</sub> direction register to output mode.

### (3) Event Counter Mode

Operation in event counter mode is the same as in timer mode, except that the timer counts signals input through the CNTR<sub>0</sub> pin. When the CNTR<sub>0</sub> active edge selection bit is "0", the rising edge of the CNTR<sub>0</sub> pin is counted.

When the CNTR<sub>0</sub> active edge selection bit is "1", the falling edge of the CNTR<sub>0</sub> pin is counted.

### (4) Pulse Width Measurement Mode

If the CNTR<sub>0</sub> active edge selection bit is "0", the timer counts the system clock divided by 16 while the CNTR<sub>0</sub> pin is at "H". If the CNTR<sub>0</sub> active edge selection bit is "1", the timer counts it while the CNTR<sub>0</sub> pin is at "L".

The count can be stopped by setting "1" to the timer X count stop bit in any mode. The corresponding interrupt request bit is set each time a timer underflows.

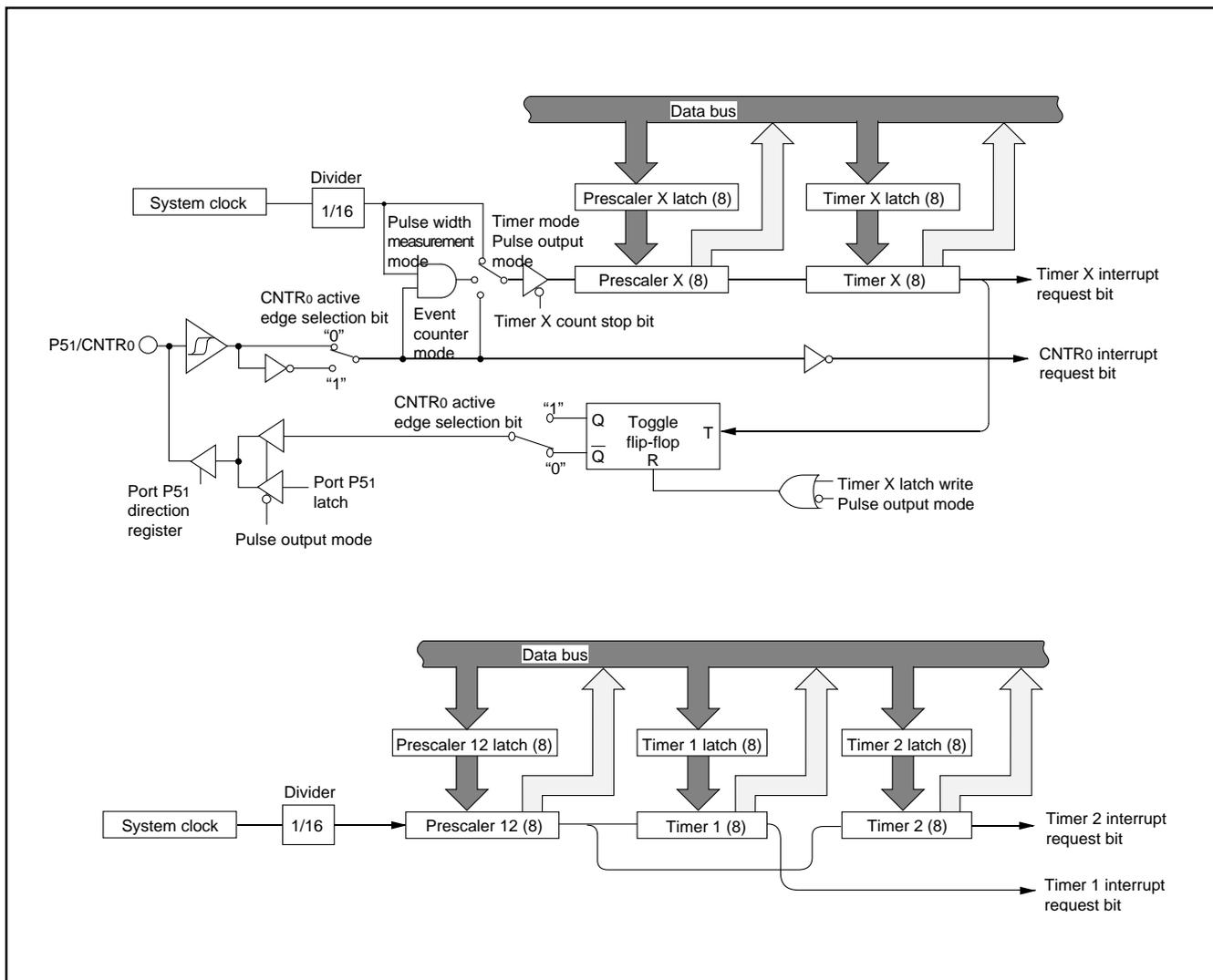


Fig. 17 Timer block diagram

## SERIAL INTERFACE Serial I/O

Serial I/O can be used as either clock synchronous or asynchronous (UART) serial I/O. A dedicated timer (baud rate generator) is also provided for baud rate generation.

### (1) Clock Synchronous Serial I/O Mode

Clock synchronous serial I/O mode can be selected by setting the mode selection bit of the serial I/O control register (bit 6 of address 0FE0<sub>16</sub>) to "1".

For clock synchronous serial I/O, the transmitter and the receiver must use the same clock. If an internal clock is used, transfer is started by a write signal to the Trancemit/Receive buffer register.

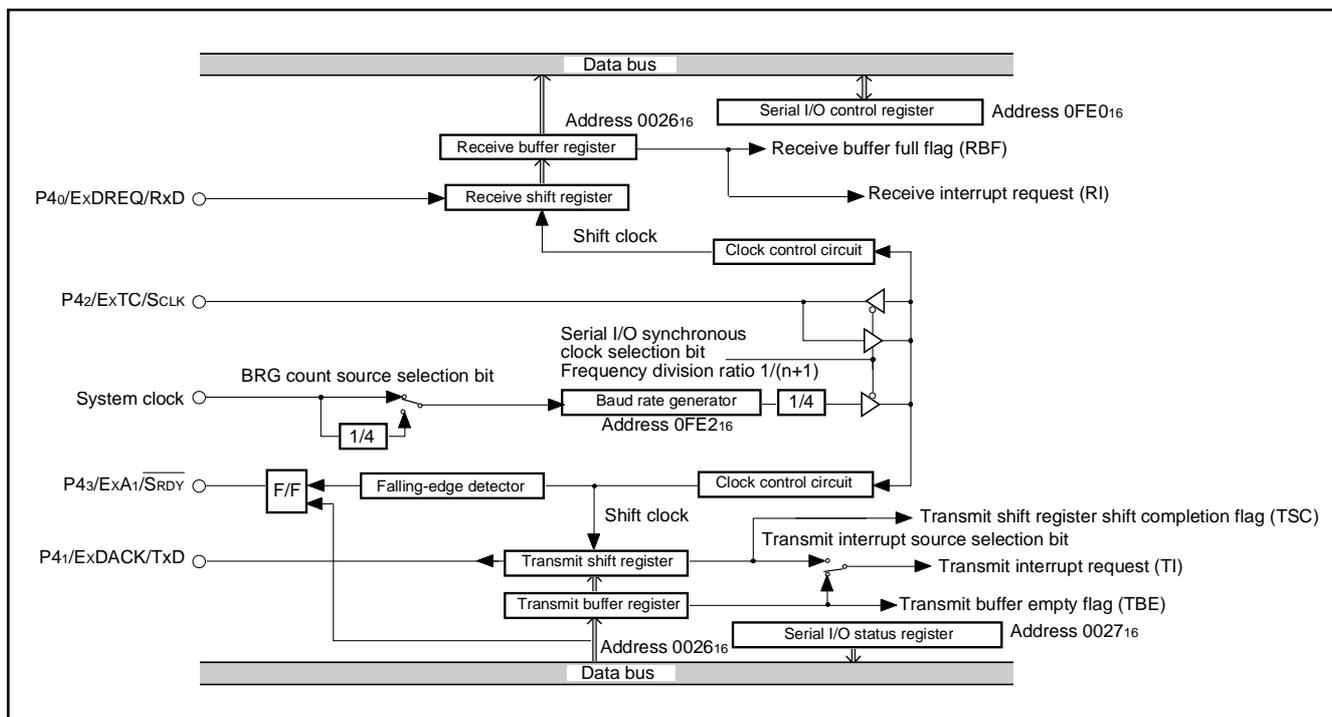


Fig. 18 Block diagram of clock synchronous serial I/O

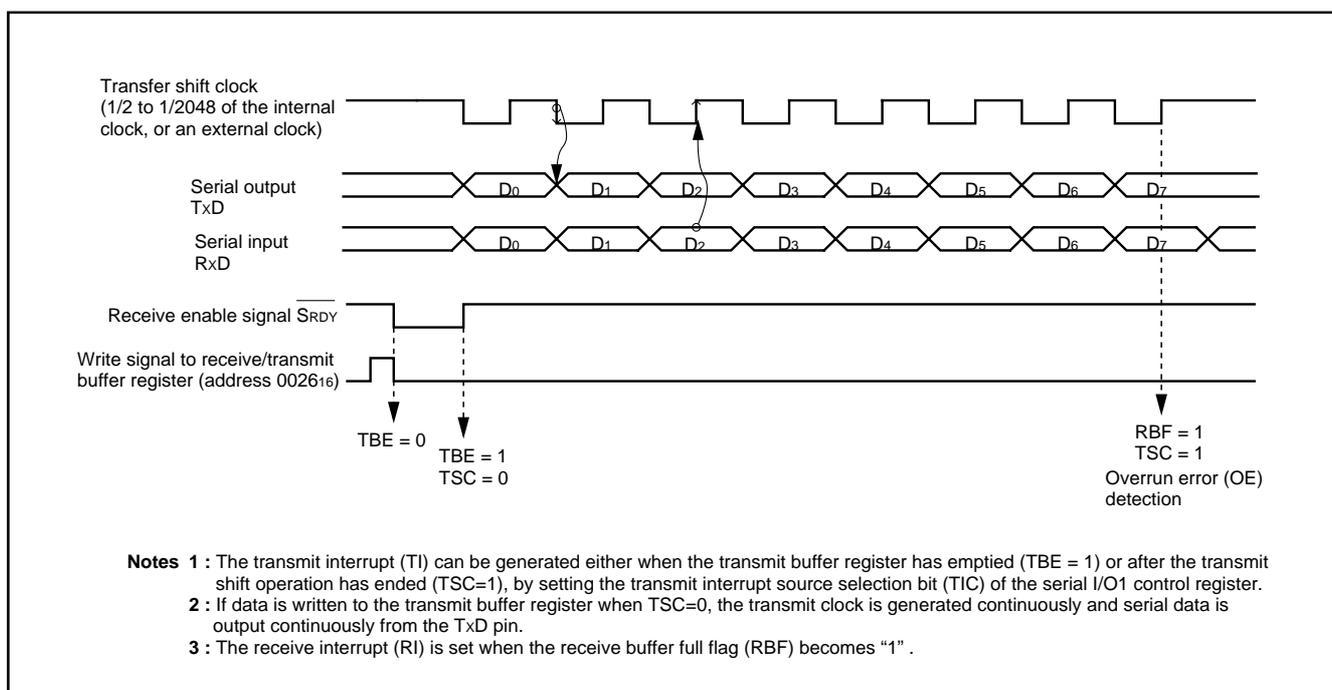


Fig. 19 Operation of clock synchronous serial I/O function

## (2) Asynchronous Serial I/O (UART) Mode

Clock asynchronous serial I/O mode (UART) can be selected by setting the serial I/O mode selection bit of the serial I/O control register to "0".

Eight serial data transfer formats can be selected, and the transfer formats used by a transmitter and receiver must be identical.

The transmit and receive shift registers each have a buffer regis-

ter, but the two buffers have the same address in memory. Since the shift register cannot be written to or read from directly, transmit data is written to the transmit buffer, and receive data is read from the receive buffer.

The transmit buffer can also hold the next data to be transmitted, and the receive buffer register can hold a character while the next character is being received.

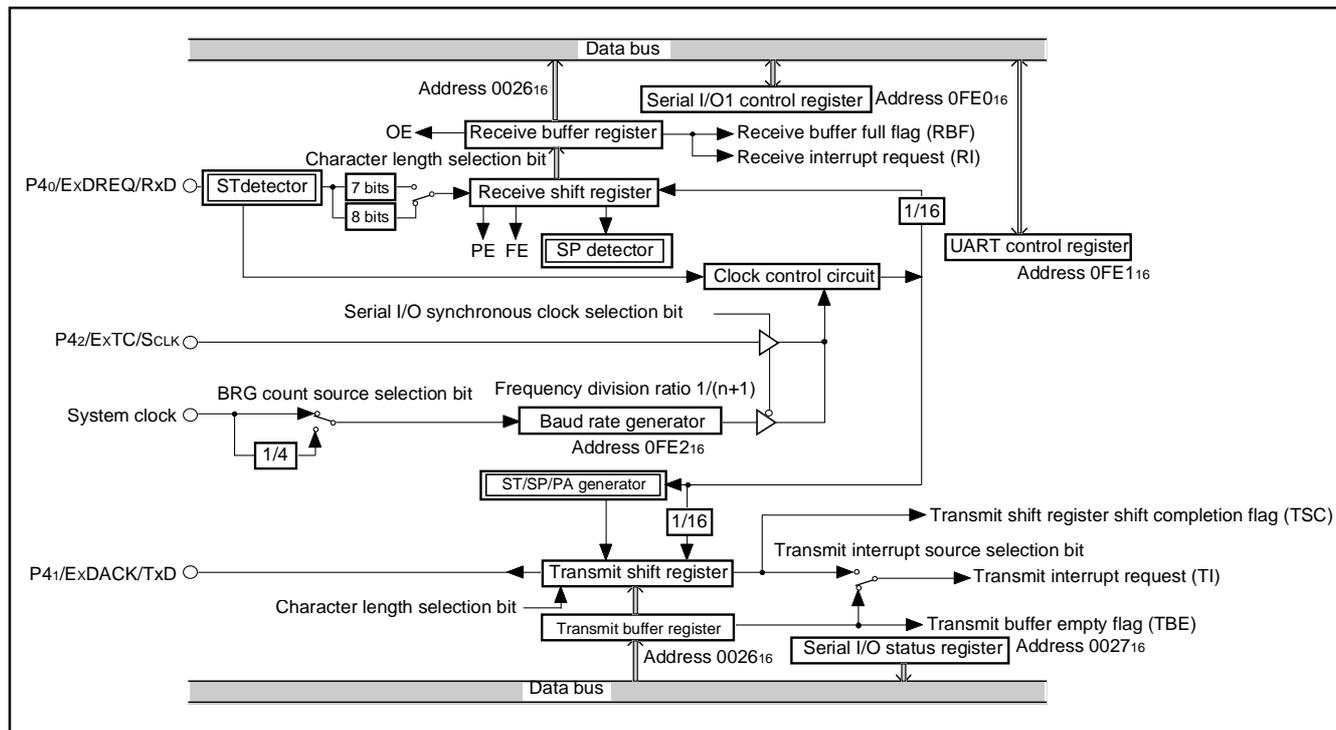


Fig. 20 Block diagram of UART serial I/O

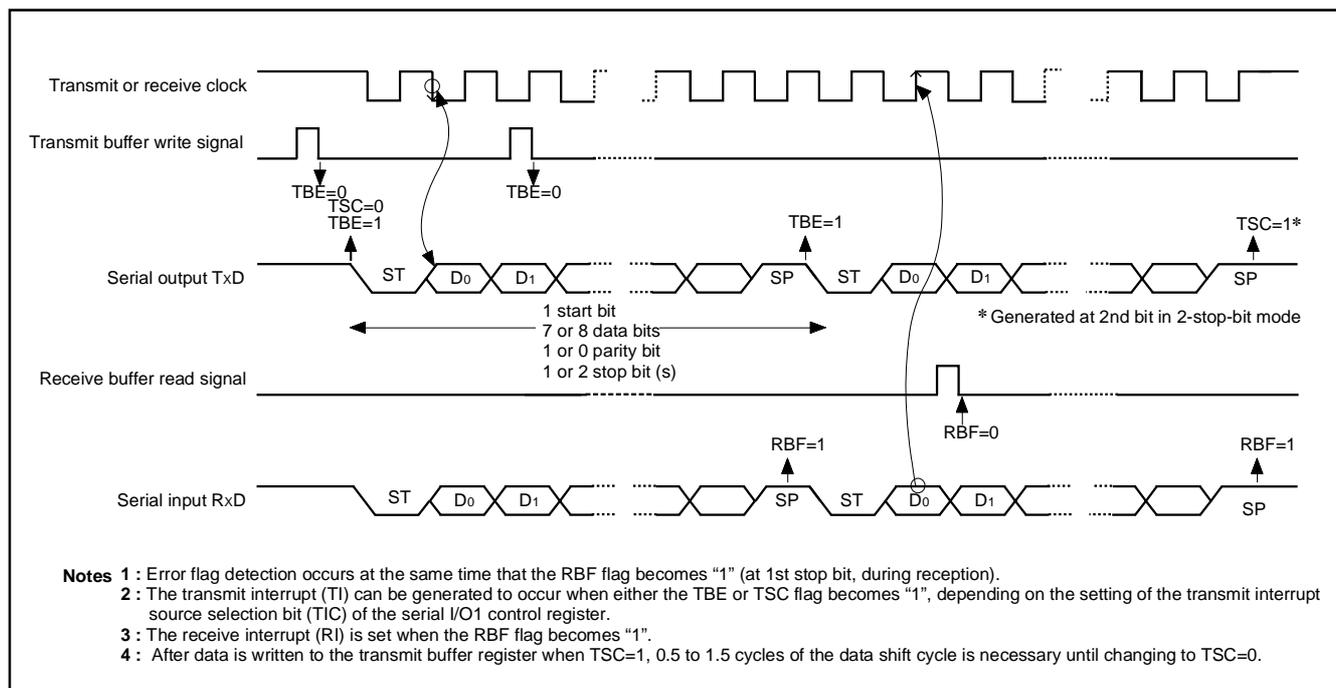


Fig. 21 Operation of UART serial I/O function

**[Serial I/O Control Register (SIOCON)] 0FE0<sub>16</sub>**

The serial I/O control register contains eight control bits for the serial I/O function.

**[UART Control Register (UARTCON)] 0FE1<sub>16</sub>**

The UART control register consists of four control bits (bits 0 to 3) which are valid when asynchronous serial I/O is selected and set the data format of an data transfer.

**[Serial I/O Status Register (SIOSTS)] 0027<sub>16</sub>**

The read-only serial I/O status register consists of seven flags (bits 0 to 6) which indicate the operating status of the serial I/O function and various errors.

Three of the flags (bits 4 to 6) are valid only in UART mode.

The receive buffer full flag (bit 1) is cleared to "0" when the receive buffer is read.

If there is an error, it is detected at the same time that data is transferred from the receive shift register to the receive buffer register, and the receive buffer full flag is set. A write to the serial I/O status register clears all the error flags OE, PE, FE, and SE (bit 3 to bit 6, respectively). Writing "0" to the serial I/O enable bit SIOE (bit 7 of the serial I/O control register) also clears all the status flags, including the error flags.

All bits of the serial I/O status register are initialized to "0" at reset, but if the transmit enable bit (bit 4) of the serial I/O control register has been set to "1", the transmit shift register shift completion flag (bit 2) and the transmit buffer empty flag (bit 0) become "1".

**[Transmit Buffer/Receive Buffer Register (TB/RB)] 0026<sub>16</sub>**

The transmit buffer register and the receive buffer register are located at the same address. The transmit buffer register is write-only and the receive buffer register is read-only. If a character bit length is 7 bits, the MSB of data stored in the receive buffer register is "0".

**[Baud Rate Generator (BRG)] 0FE2<sub>16</sub>**

The baud rate generator determines the baud rate for serial transfer.

The baud rate generator divides the frequency of the count source by  $1/(n + 1)$ , where  $n$  is the value written to the baud rate generator.

**■Notes on serial I/O**

When setting the transmit enable bit to "1", the serial I/O transmit interrupt request bit is automatically set to "1". When not requiring the interrupt occurrence synchronized with the transmission enabled, take the following sequence.

- ①Set the serial I/O transmit interrupt enable bit to "0" (disabled).
- ②Set the transmit enable bit to "1".
- ③Set the serial I/O transmit interrupt request bit to "0" after 1 or more instructions have been executed.
- ④Set the serial I/O transmit interrupt enable bit to "1" (enabled).

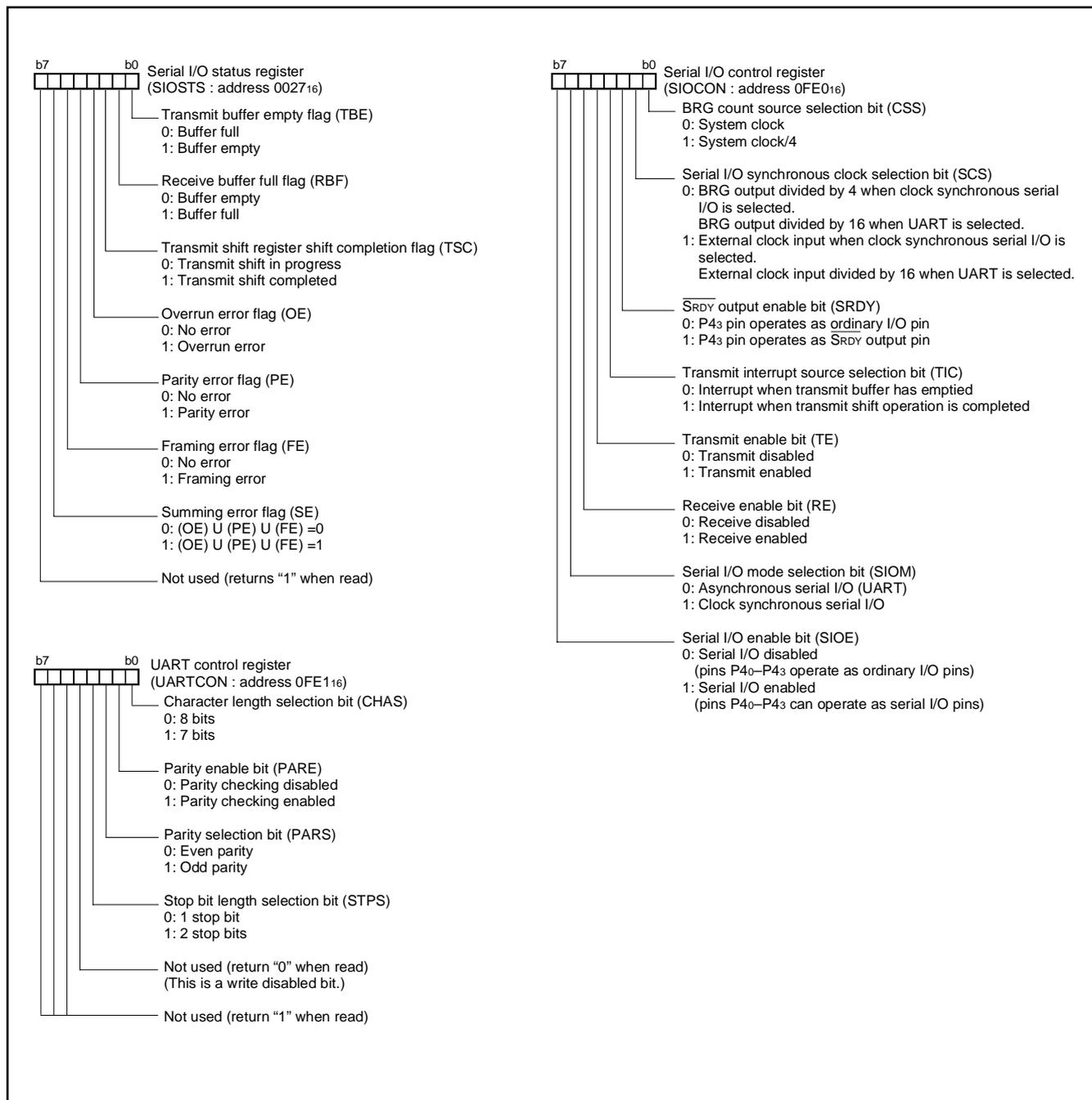


Fig. 22 Structure of serial I/O control registers

## USB FUNCTION

38K2 Group is equipped with a USB function control circuit (USBFCC) that enables effective interfacing with the host-PC. This circuit is in compliance with USB2.0's Full-Speed Transfer Mode (12 Mbps, equivalent to USB1.1). This circuit also supports all four transfer-types specified in the standard USB specification. The USBFCC has two USB addresses and 6 endpoints, enabling separate control of the HUB functions and peripheral functions. The USB address for HUB functions is equipped with two endpoints. Each endpoint is fixed to a specified transfer type: Endpoint 0 is fixed to Control Transfer and Endpoint 1 is fixed to Interrupt Transfer.

The USB address for peripheral functions is equipped with four endpoints that can select its transfer type. Although Endpoint 0 is fixed to Control Transfer, the Endpoints 1 to 3 can be set to Interrupt Transfer, Bulk Transfer, or Isochronous Transfer.

A dedicated circuit automatically performs stage management for Control Transfer and packet management for transactions, which are necessary for matching of data transmit/receive timing, error detection, and retry after error. This dedicated control circuit enables the user to develop a program or timing design very easily. Each endpoint can be programmed for data transfer conditions so that the endpoints are adaptive for all USB device class transfer systems.

The data buffer of each endpoint can be assigned to any area in the multi-channel RAM. This feature offers highly efficient memory usage by avoiding re-buffering and enabling simple data modification.

The transmit/receive data is directly transferred to the data buffer via the control circuit (direct RAM access type) without disturbing the CPU operation. This mechanism enables the CPU to transfer data smoothly with no drop in performance. In addition to this buffer function, a double-buffer setting will keep a re-buffering stall at a minimum and increase the overall data throughput (max. 64 bytes X 2 channels).

As other special signals control, the endpoints have detection functions for the USB bus reset signal, resume signal, suspend signal, and SOF signal, and also have a remote wake-up signal transmit function.

When completing data transfer or receiving a special signal, the endpoint generates the corresponding interrupt to the CPU (3 vectors/24 factors).

With all this essential yet comprehensive built-in hardware, your system using the 38K2 group will be ready for any USB application that comes its way.

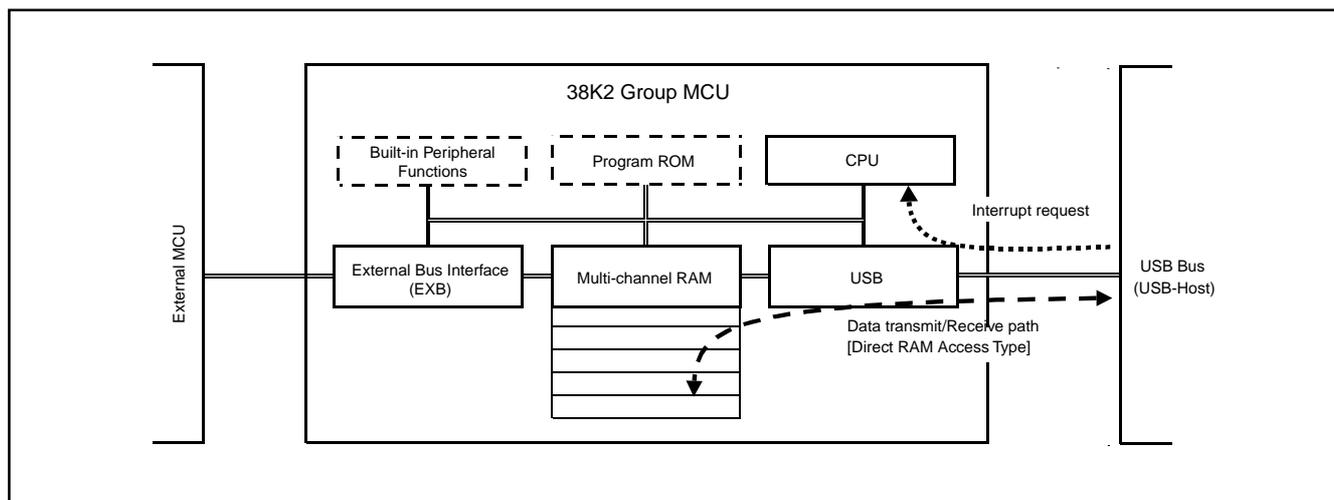


Fig. 23 USB function overview

## USB Data Transfer

The USB specification promises 12 Mbps data transfer in the full-speed mode, that is equivalent to 1.5 M bytes per second of data transactions.

However, in USB data transfer, bit-stuffing may be executed depending on the bit patterns of the transfer data, possibly resulting in 1-byte data (normally 8 bits) handled as up to 10 bits.

Because USB uses asynchronous transfers, the clock cycle of the USB internal reference clock may change to adjust to the clock phase. Therefore, the access timing of the USBFCC for the multi-channel RAM will change owing to the frequency of internal clock  $\phi$ :

When the USBFCC is operating at  $\phi = 8$  MHz, access for a normal

transfer is performed every 5 to 6 cycles and access for a bit-stuffing transfer is performed in up to 7 cycles.

If the EXB function is enabled in the above conditions, this function generates a maximum wait of 1 clock cycle, so that the access is performed every 4 to 8 cycles.

When operating at  $\phi = 6$  MHz, a normal access is performed every 4 cycles. If the clock-phase correction of the reference clock occurs, access is performed every 3 to 5 cycles.

If bit stuffing occurs at this clock rate, the access cycle will be extended to up to 6 cycles. When the EXB function that generates a maximum 1-wait cycle is used in this condition, the access cycle will be 2 (min.) to 7 (max.) cycles.

## USB Function Control Circuit (USB FCC) Block Diagram

The following diagram shows the USB FCC block diagram. The circuit comprises:

- (1) Serial Interface Engine (SIE)
- (2) Device Control Unit (DCU)
- (3) Internal Memory Interface (MIF)
- (4) CPU Interface (CIF)

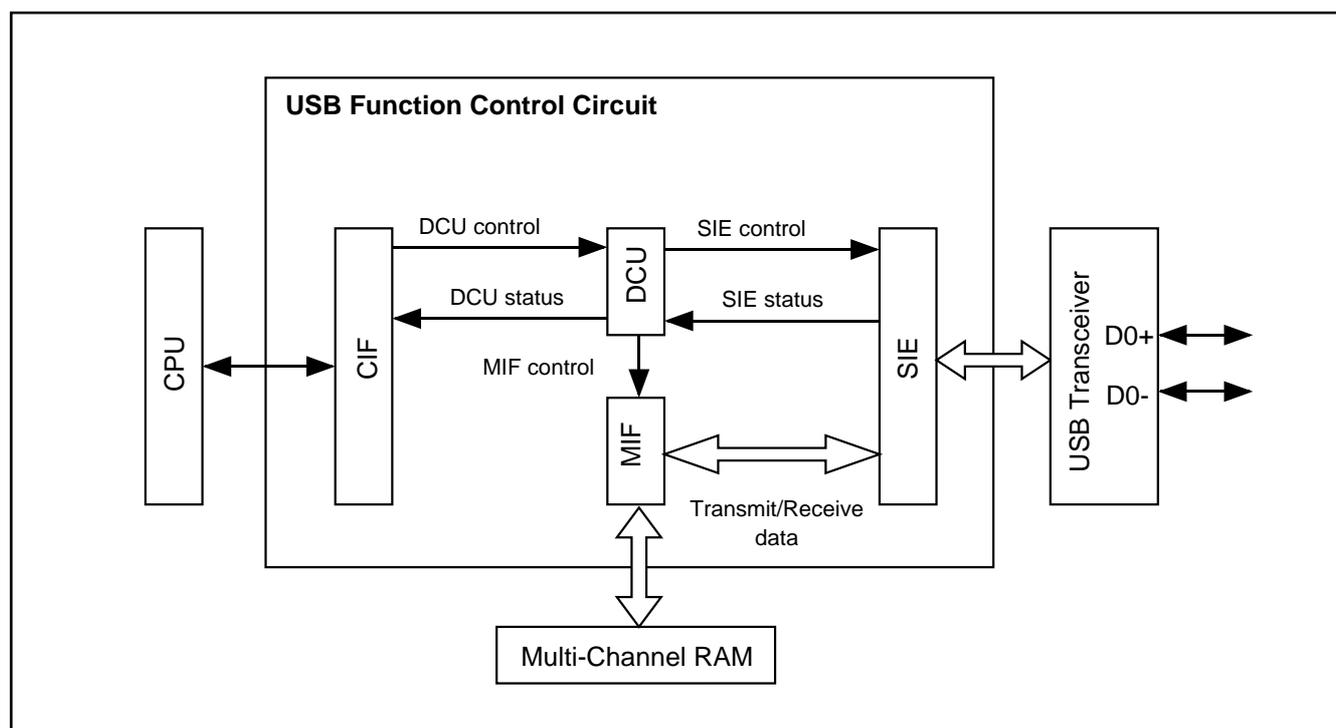


Fig. 24 USB Function Control Circuit (USB FCC) block diagram

### (1) Serial Interface Engine (SIE)

The SIE performs the following USB lower-layer protocols (packets, transactions):

- Sampling of receive data and clock, generation of transmit clock
- Serial-to-parallel conversion of transmit/receive data
- NRZI (Non Return Zero Invert) encode/decode
- Bit stuffing/unstuffing
- SYNC (Synchronization Pattern) detection, EOP (End of Packet) detection
- USB address detection, endpoint detection
- CRC (Cyclic Redundancy Check) generation and checking

### (2) Device Control Unit (DCU)

The DCU manages the following USB upper-layer protocols (address/endpoint and control-transfer sequence):

- Status control for each endpoint
- Control-transfer sequence control
- Memory interface status control

### (3) Memory Interface (MIF)

The MIF controls the flow of data transfer between the SIE and the multi-channel RAM under the management of the DCU.

### (4) CPU Interface (CIF)

The CIF performs the following functions:

- Mode setting via registers, DCU control signal generation, DCU status signal reading
- Interrupt signal generation
- Internal bus interface control.

### USB Port External Circuit Configuration

The operation mode of the USB port driver circuit can be configured by USB control register (address 001016).

Figure 25 and Figure 26 show the USB port external circuit block diagram.

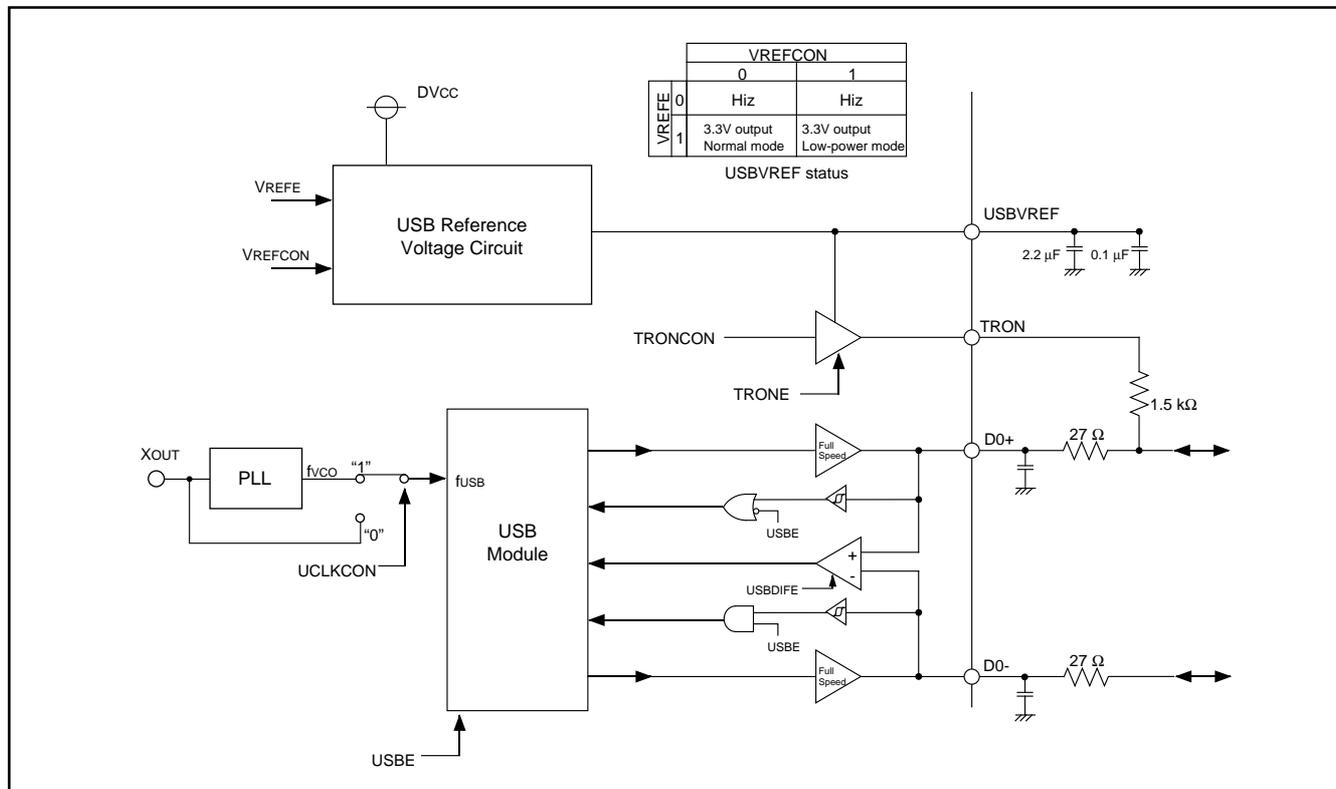


Fig. 25 USB port external circuit (D0+, D0-, USBVREF, TrON) block diagram (4.0V ≤ Vcc ≤ 5.25V)

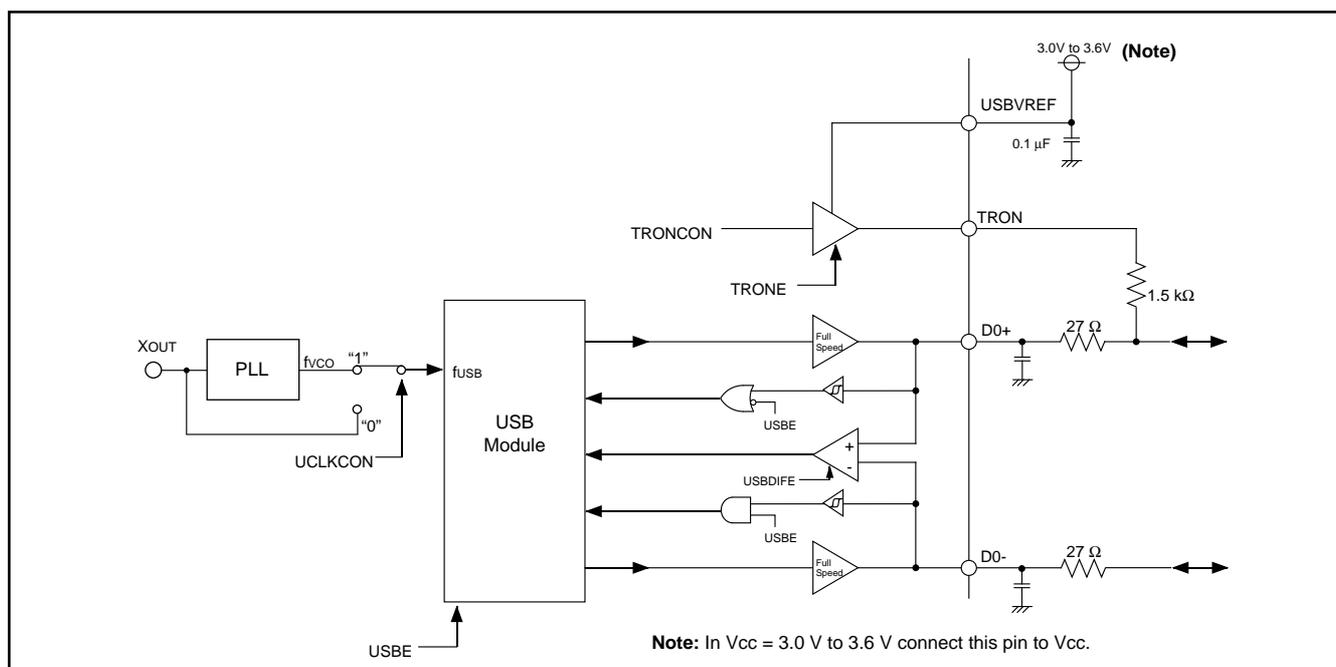


Fig. 26 USB port external circuit (D0+, D0-, USBVREF, TrON) block diagram (3.0V ≤ Vcc ≤ 4.0V)

### Endpoint Buffer Area Setting

The buffer area used in data transfer can be assigned to any area of the multi-channel RAM for each endpoint.

#### ●Buffer area beginning address

The buffer area configuration register (address 0FED16) defines the beginning address of the buffer area (every 32 bytes) for each Endpoint. However, the only RAM area is configurable.

- 00h [Address 000016], 01h [Address 002016]: Not configurable
- 02h [Address 004016] to 1Fh [Address 03E016]: Configurable

#### ●Interrupt-source dependant buffer area offset address

An offset value is added to the beginning address of each source, which is specified by the interrupt source register (address 001D16), for each endpoint.

This section describes in detail the beginning address specified by the buffer area set register as offset address 00h, according to each endpoint.

#### (1) Endpoint 00

Endpoint 00 has two kinds of interrupt sources for accessing the buffer. The respective address offsets are:

- BSRDY00 (SETUP Buffer Ready Interrupt): Offset address = 00h
- BRDY00 (OUT or IN Buffer Ready Interrupt): Offset address = 08h

#### (2) Endpoint 01

The buffer area offset address for each interrupt source for of Endpoint 01 varies according to the contents of the EP01 set register (address 001916).

- In single buffer mode (DBLB01 = "0"): Endpoint 01 has only one interrupt source for accessing the buffer. B0RDY01 (Buffer 0 Ready Interrupt): Offset address = 00h
- In double buffer mode (DBLB01 = "1"): Endpoint 01 has two kinds of interrupt sources for accessing the buffer. B0RDY01 (Buffer 0 Ready Interrupt): Offset address = 00h

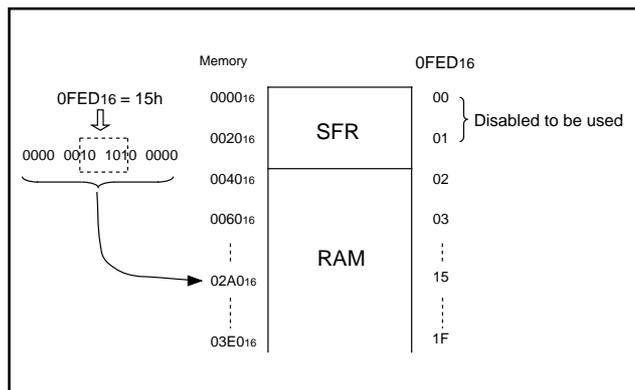


Fig. 27 Example setting of buffer area beginning address

#### B1RDY01 (Buffer 1 Ready Interrupt):

The offset address varies according to the double buffer beginning address set bit (BSIZ01).

- Offset address = 08h when BSIZ01 = 00
- Offset address = 10h when BSIZ01 = 01
- Offset address = 40h when BSIZ01 = 10
- Offset address = 80h when BSIZ01 = 11

#### (3) Endpoints 02 and 03

Same as Endpoint 01.

#### (4) Endpoint 10

Same as Endpoint 00.

#### (5) Endpoint 11

Endpoint 11 has only one interrupt source for accessing the buffer. B0RDY11 (Buffer 0 Ready Interrupt): Offset address = 00h

### Notes

The selected RAM area must be within addresses 004016 to 03FF16.

Make sure the buffer area beginning address is set in agreement with the offset address and the number of transmit/receive data bytes.

This is particularly important when in the double buffer mode or when handling 64-byte data.

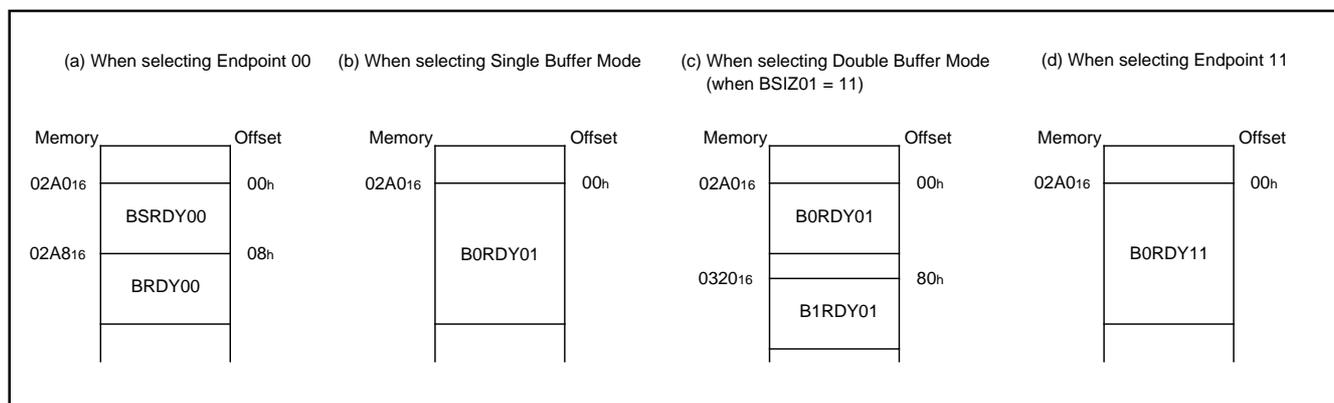


Fig. 28 Examples of interrupt source dependant buffer area offset address

## USB Interrupt Function

USB Interrupt Control Circuit (USBINTCON) has 3 requests and 22 USB-device interrupt request sources. Each interrupt source register enables the user to easily determine which interrupt has occurred.

Table 7 shows the list of USB interrupt sources.

Table 7 USB interrupt sources

Interrupt request bit (IREQ1: Address 003C <sub>16</sub> )	USB interrupt bit (USBIREQ: Address 0017 <sub>16</sub> )	Interrupt source
USB bus reset	—	At USB bus reset signal detection: After enabling the USB module (USBE = "1"), an interrupt request occurs when 2.5 μs SE0 state is detected in D0+/D0- port. (Equivalent to 120-clock length when f <sub>USB</sub> = 48 MHz)
USB SOF	—	At SOF packet receive: After enabling the USB module (USBE = "1"), an interrupt request occurs when SOF packet is detected in D0+/D0- port. Its occurrence does not depend on frame-time or CRC value after SOF packet is transferred. (Normally, SOF packet detection occurs only when f <sub>USB</sub> = 48 MHz)
USB device	EP00	At Endpoint 00 data transfer complete: •Buffer ready (read/write enabled state) •Control transfer completed •Status stage transition •SETUP buffer ready (read enabled state) •Control transfer error
	EP01	At Endpoint 01 data transfer complete: •Buffer 0 ready (read/write enabled state) •Buffer 1 ready (read/write enabled state) •Transfer error
	EP02	At Endpoint 02 data transfer complete: •Buffer 0 ready (read/write enabled state) •Buffer 1 ready (read/write enabled state) •Transfer error
	EP03	At Endpoint 03 data transfer complete: •Buffer 0 ready (read/write enabled state) •Buffer 1 ready (read/write enabled state) •Transfer error
	EP10	At Endpoint 10 data transfer complete: •Buffer ready (read/write enabled state) •Control transfer completed •Status stage transition •SETUP buffer ready (read enabled state) •Control transfer error
	EP11	At Endpoin 11 data transfer complete: •Buffer 0 ready (write enabled state)
	SUS	At suspend signal detection: After enabling the USB module (USBE = "1"), an interrupt request occurs when 3 ms J state is detected in D0+/D0- port. (Equivalent to 144,000 clock-length when f <sub>USB</sub> = 48MHz)
	RSM	At resume signal detection: After enabling the USB module (USBE = "1") and resume interrupt (RSME = "1"), an interrupt request occurs when a bus state change (J state to SE0 or K state) is detected in D0- port.

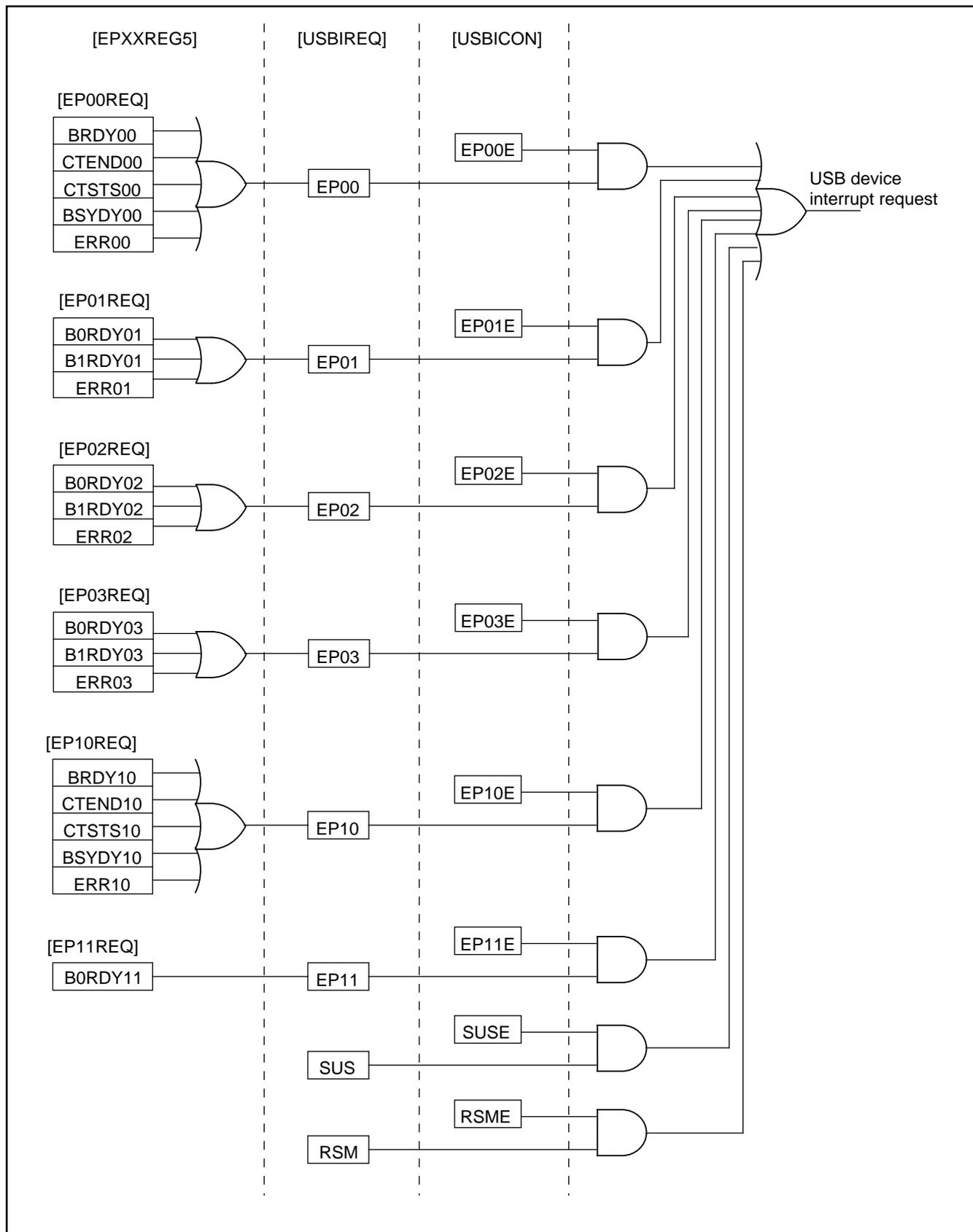


Fig. 29 USB device interrupt control

### USB Register List

The USB register list is shown below.

Address	Register Name	SYMBOL	USB SFR							
			bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0010 <sub>16</sub>	USB control register	USBCON	USBE	UCLKCON	USBDIFE	VREFE	VREFCON	TRONE	TRONCON	WKUP
0011 <sub>16</sub>	USB Function/Hub enable register	USBAE							AD1E	AD0E
0012 <sub>16</sub>	USB function address register	USBA0							USBADD0[6:0]	
0013 <sub>16</sub>	USB HUB address register	USBA1							USBADD1[6:0]	
0014 <sub>16</sub>	Frame number register Low	FNUML							FNUM[7:0]	
0015 <sub>16</sub>	Frame number register High	FNUMH							FNUM[10:8]	
0016 <sub>16</sub>	USB interrupt source enable register	USBCON	RSME	SUSE	EP11E	EP10E	EP03E	EP02E	EP01E	EP00E
0017 <sub>16</sub>	USB interrupt source register	USBREQ	RSM	SUS	EP11	EP10	EP03	EP02	EP01	EP00
0018 <sub>16</sub>	Endpoint index register	USBINDEX						ADIDX	EPIDX[1:0]	
0019 <sub>16</sub>	Endpoint field register 1	EPXXREG1								
001A <sub>16</sub>	Endpoint field register 2	EPXXREG2								
001B <sub>16</sub>	Endpoint field register 3	EPXXREG3								
001C <sub>16</sub>	Endpoint field register 4	EPXXREG4								
001D <sub>16</sub>	Endpoint field register 5	EPXXREG5								
001E <sub>16</sub>	Endpoint field register 6	EPXXREG6								
001F <sub>16</sub>	Endpoint field register 7	EPXXREG7								
0FEC <sub>16</sub>	Endpoint field register 8	EPXXREG8								
0FED <sub>16</sub>	Endpoint field register 9	EPXXREG9								
(1) Endpoint 00										
0019 <sub>16</sub>	EP00 stage register	EP00STG								SETUP00
001A <sub>16</sub>	EP00 control register 1	EP00CON1							PID00[1:0]	
001B <sub>16</sub>	EP00 control register 2	EP00CON2								BVAL00
001C <sub>16</sub>	EP00 control register 3	EP00CON3								CTENDE00
001D <sub>16</sub>	EP00 interrupt source register	EP00REQ			ERR00	BSRDY00	CTSTS00	CTEND00		BRDY00
001E <sub>16</sub>	EP00 byte number register	EP00BYT					BBYT00[3:0]			
001F <sub>16</sub>										
0FEC <sub>16</sub>										
0FED <sub>16</sub>	EP00 buffer area set register	EP00BUF					BADD00[4:0]			
(2) Endpoint 01										
0019 <sub>16</sub>	EP01 set register	EP01CFG	TYP01[1:0]		DIR01	ITMD01	SQCL01	DBLB01	BSIZ01[1:0]	
001A <sub>16</sub>	EP01 control register 1	EP01CON1							PID01[1:0]	
001B <sub>16</sub>	EP01 control register 2	EP01CON2								BOVAL01
001C <sub>16</sub>	EP01 control register 3	EP01CON3								B1VAL01
001D <sub>16</sub>	EP01 interrupt source register	EP01REQ					ERR01	B1RDY01	B0RDY01	
001E <sub>16</sub>	EP01 byte number register 0	EP01BYT0				B0BYT01[6:0]				
001F <sub>16</sub>	EP01 byte number register 1	EP01BYT1				B1BYT01[6:0]				
0FEC <sub>16</sub>	EP01 MAX. packet size register	EP01MAX				MXPS01[6:0]				
0FED <sub>16</sub>	EP01 buffer area set register	EP01BUF					BADD01[4:0]			
(3) Endpoint 02										
0019 <sub>16</sub>	EP02 set register	EP02CFG	TYP02[1:0]		DIR02	ITMD02	SQCL02	DBLB02	BSIZ02[1:0]	
001A <sub>16</sub>	EP02 control register 1	EP02CON1							PID02[1:0]	
001B <sub>16</sub>	EP02 control register 2	EP02CON2								BOVAL02
001C <sub>16</sub>	EP02 control register 3	EP02CON3								B1VAL02
001D <sub>16</sub>	EP02 interrupt source register	EP02REQ					ERR02	B1RDY02	B0RDY02	
001E <sub>16</sub>	EP02 byte number register 0	EP02BYT0				B0BYT02[6:0]				
001F <sub>16</sub>	EP02 byte number register 1	EP02BYT1				B1BYT02[6:0]				
0FEC <sub>16</sub>	EP02 MAX. packet size register	EP02MAX				MXPS02[6:0]				
0FED <sub>16</sub>	EP02 buffer area set register	EP02BUF					BADD02[4:0]			
(4) Endpoint 03										
0019 <sub>16</sub>	EP03 set register	EP03CFG	TYP03[1:0]		DIR03	ITMD03	SQCL03	DBLB03	BSIZ03[1:0]	
001A <sub>16</sub>	EP03 control register 1	EP03CON1							PID03[1:0]	
001B <sub>16</sub>	EP03 control register 2	EP03CON2								BOVAL03
001C <sub>16</sub>	EP03 control register 3	EP03CON3								B1VAL03
001D <sub>16</sub>	EP03 interrupt source register	EP03REQ					ERR03	B1RDY03	B0RDY03	
001E <sub>16</sub>	EP03 byte number register 0	EP03BYT0				B0BYT03[6:0]				
001F <sub>16</sub>	EP03 byte number register 1	EP03BYT1				B1BYT03[6:0]				
0FEC <sub>16</sub>	EP03 MAX. packet size register	EP03MAX				MXPS03[6:0]				
0FED <sub>16</sub>	EP03 buffer area set register	EP03BUF					BADD03[4:0]			
(5) Endpoint 10										
0019 <sub>16</sub>	EP10 set register	EP10STG								SETUP10
001A <sub>16</sub>	EP10 control register 1	EP10CON1							PID10[1:0]	
001B <sub>16</sub>	EP10 control register 2	EP10CON2								BVAL10
001C <sub>16</sub>	EP10 control register 3	EP10CON3								CTENDE10
001D <sub>16</sub>	EP10 interrupt source register	EP10REQ			ERR10	BSRDY10	CTSTS10	CTEND10	BRDY10	
001E <sub>16</sub>	EP10 byte number register	EP10BYT					BBYT10[3:0]			
001F <sub>16</sub>										
0FEC <sub>16</sub>										
0FED <sub>16</sub>	EP10 buffer area set register	EP10BUF					BADD10[4:0]			
(6) Endpoint 11										
0019 <sub>16</sub>	EP11 set register	EP11CFG	TYP11		DIR11		SQCL11		PID11[1:0]	
001A <sub>16</sub>	EP11 control register 1	EP11CON1							PID11[1:0]	
001B <sub>16</sub>	EP11 control register 2	EP11CON2								BOVAL11
001C <sub>16</sub>										
001D <sub>16</sub>	EP11 interrupt source register	EP11REQ								B0RDY11
001E <sub>16</sub>	EP11 byte number register	EP11BYT0								B0BYT11
001F <sub>16</sub>										
0FEC <sub>16</sub>										
0FED <sub>16</sub>	EP11 buffer area set register	EP11BUF					BADD11[4:0]			

 : Not used

Fig. 30 USB related registers

## USB Related Registers

The USB related registers are shown below.

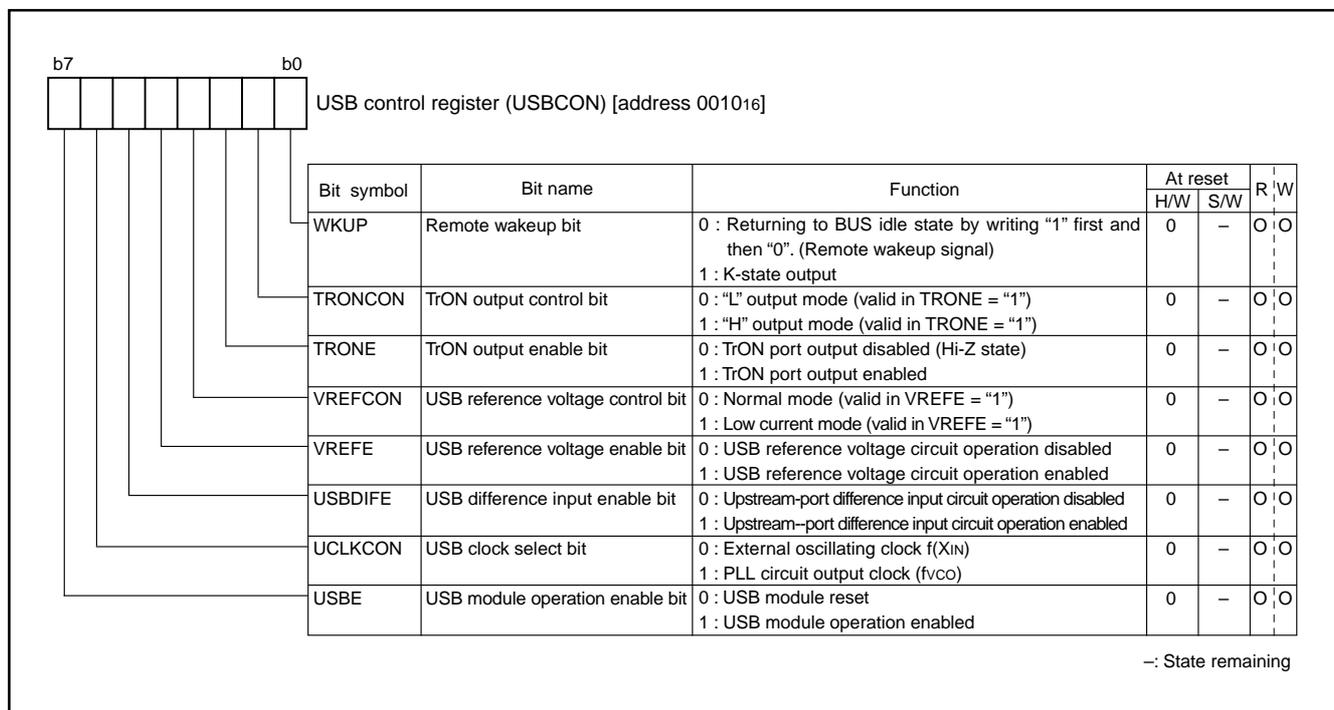


Fig. 31 Structure of USB control register

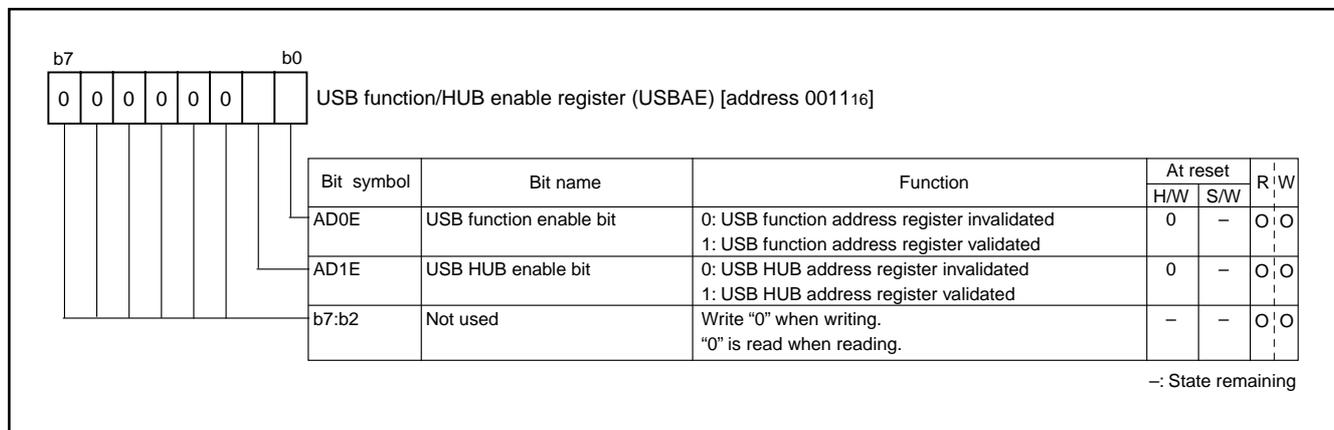


Fig. 32 Structure of USB function/HUB enable register

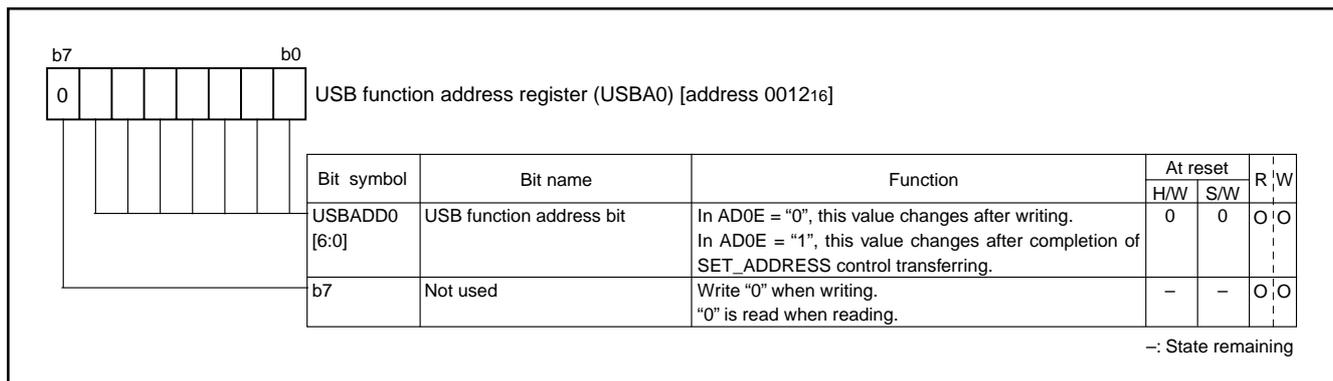


Fig. 33 Structure of USB function address register

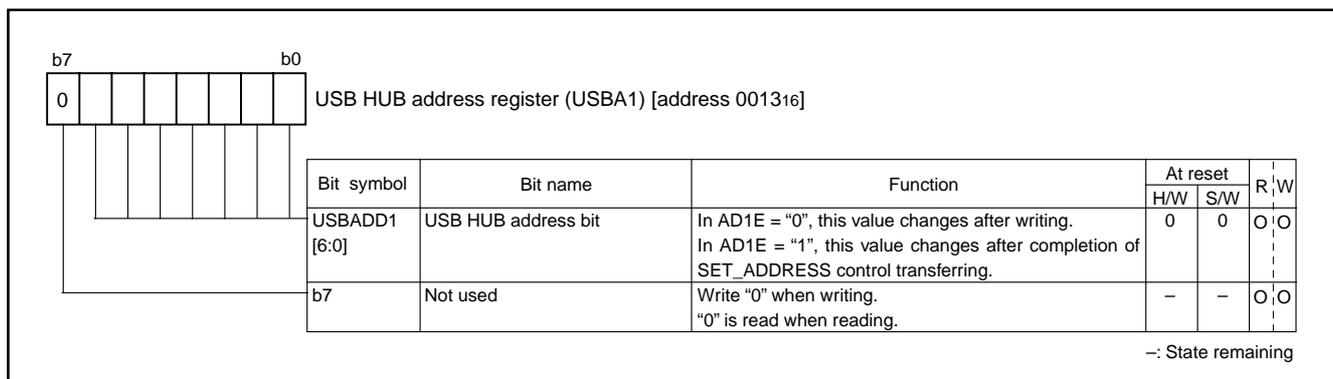


Fig. 34 Structure of USB HUB address register

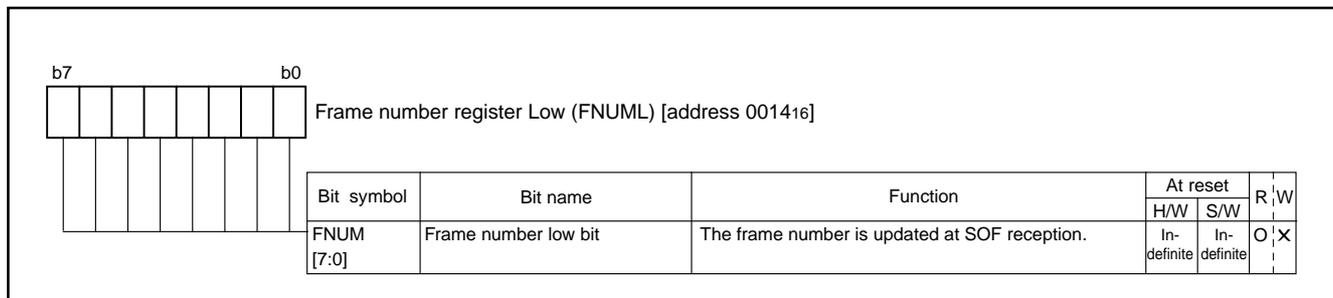


Fig. 35 Structure of Frame number register Low

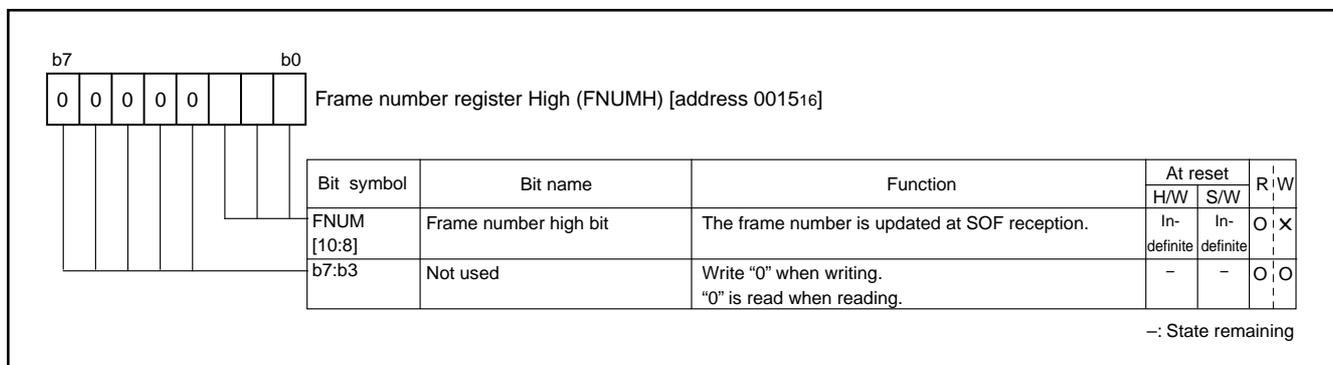


Fig. 36 Structure of Frame number register High

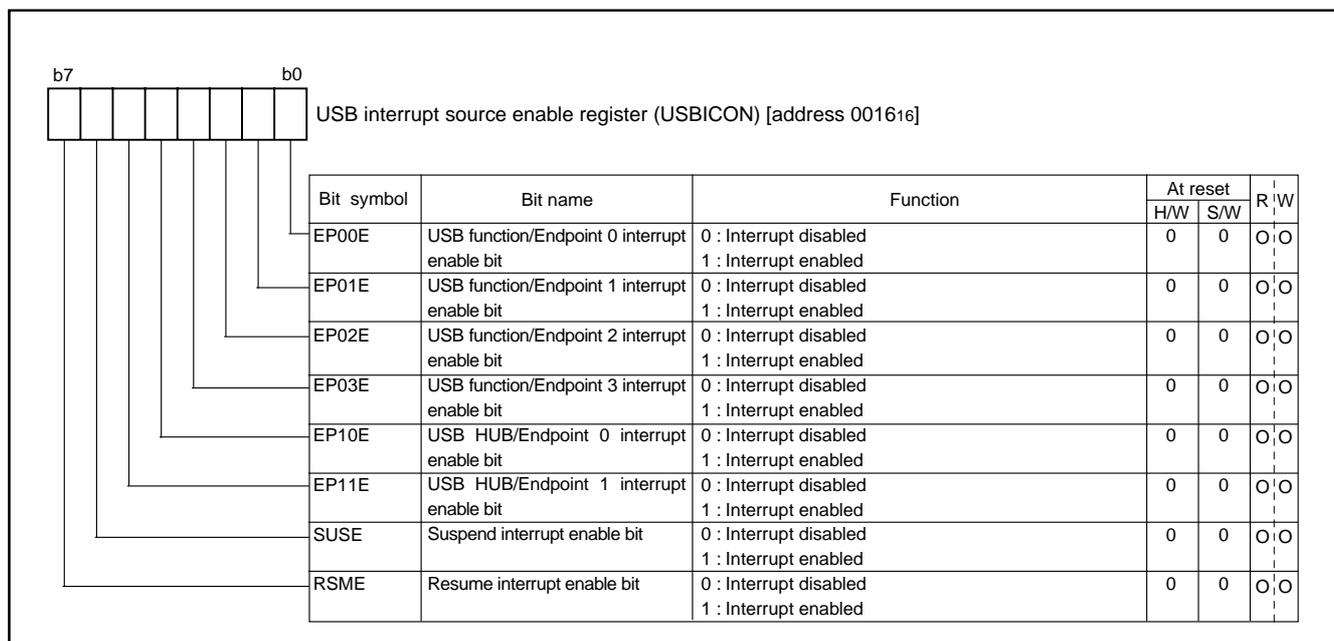


Fig. 37 Structure of USB interrupt source enable register

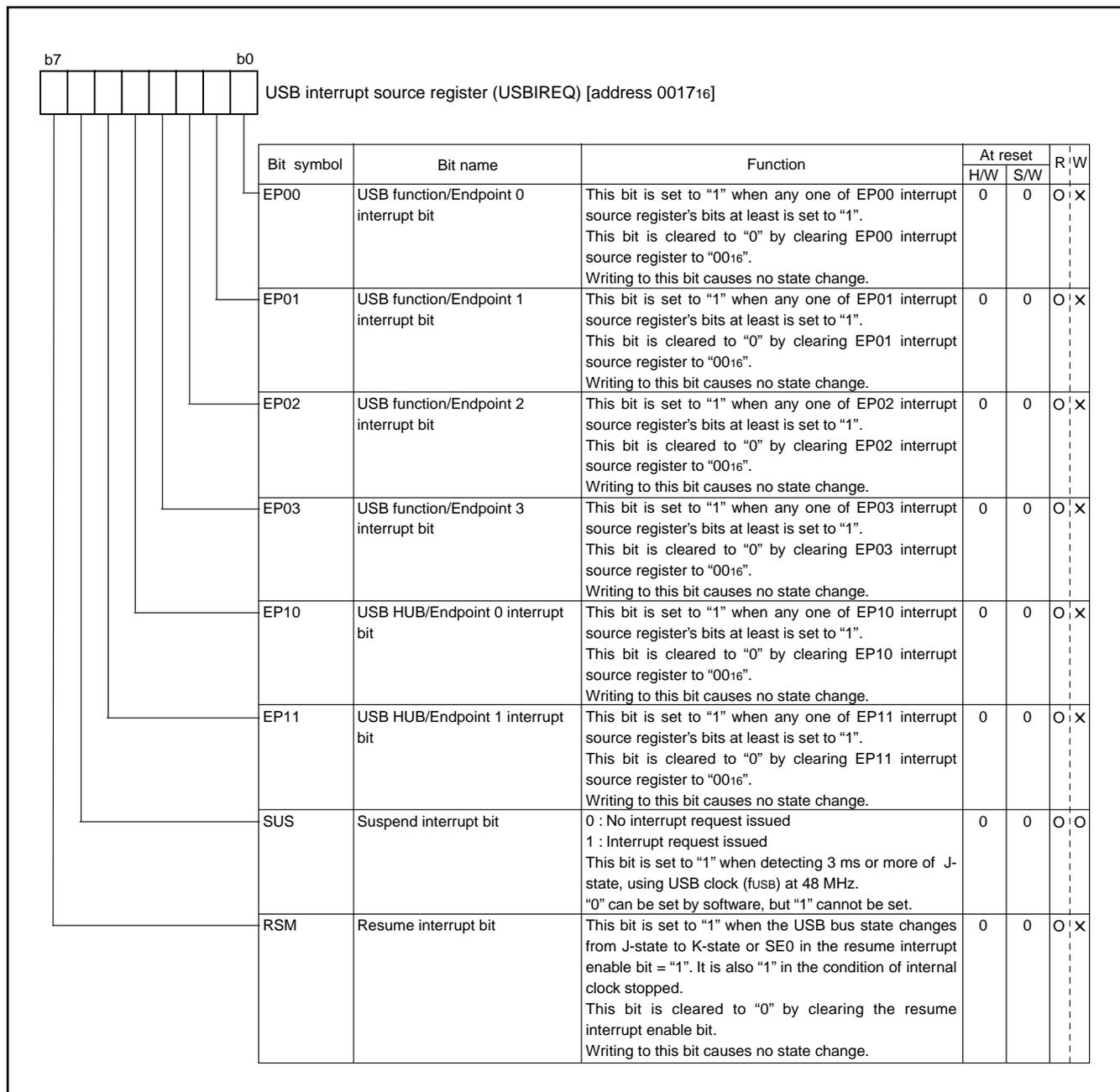


Fig.38 Structure of USB interrupt source register

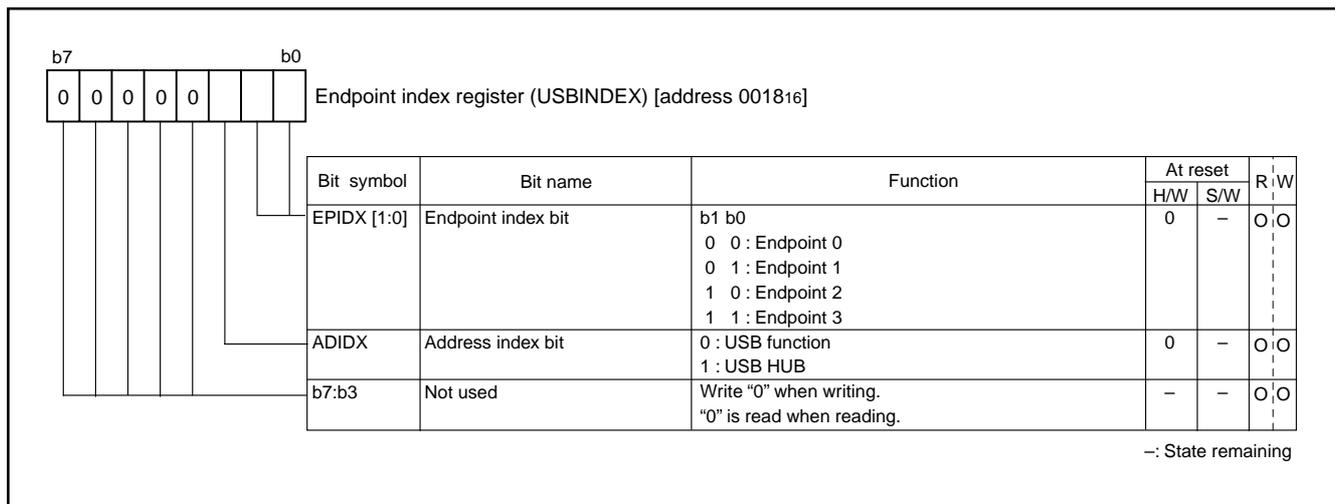


Fig. 39 Structure of Endpoint index register

## (1) Endpoint 00

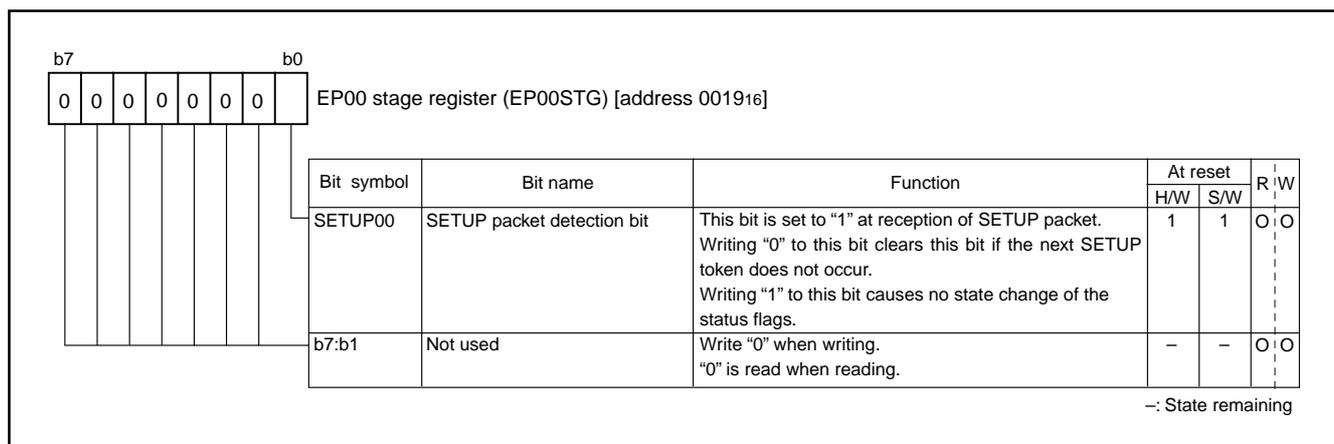


Fig. 40 Structure of EP00 stage register

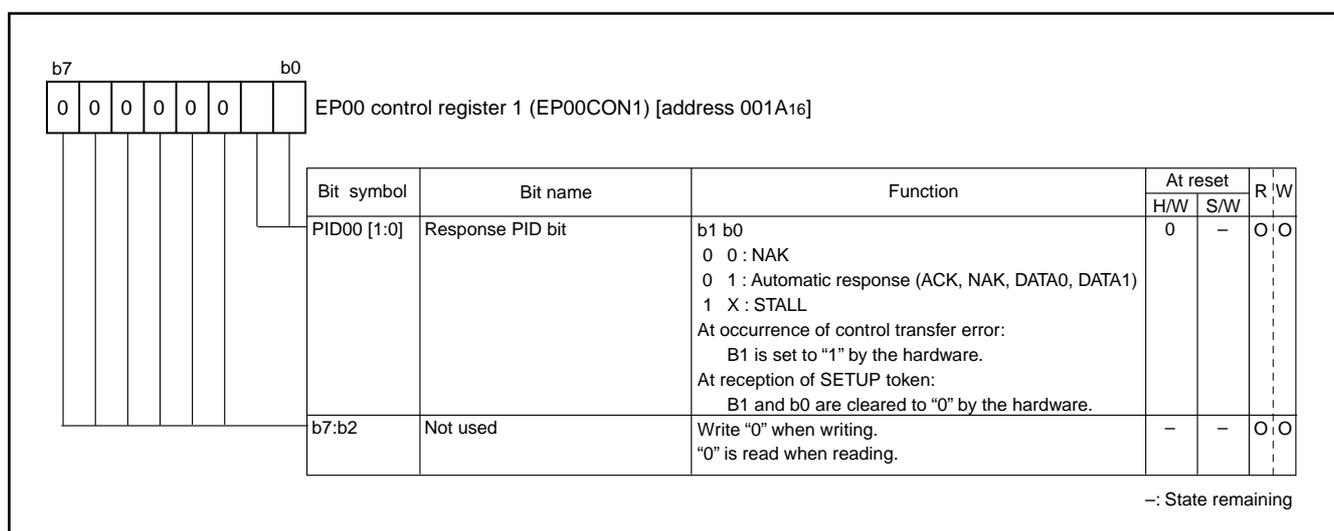


Fig. 41 Structure of EP00 control register 1

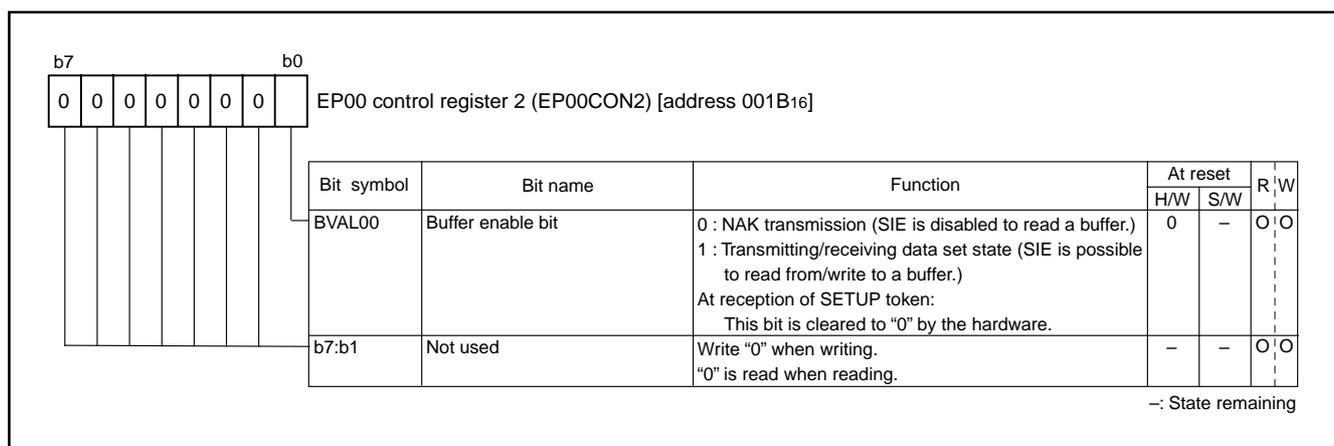


Fig. 42 Structure of EP00 control register 2

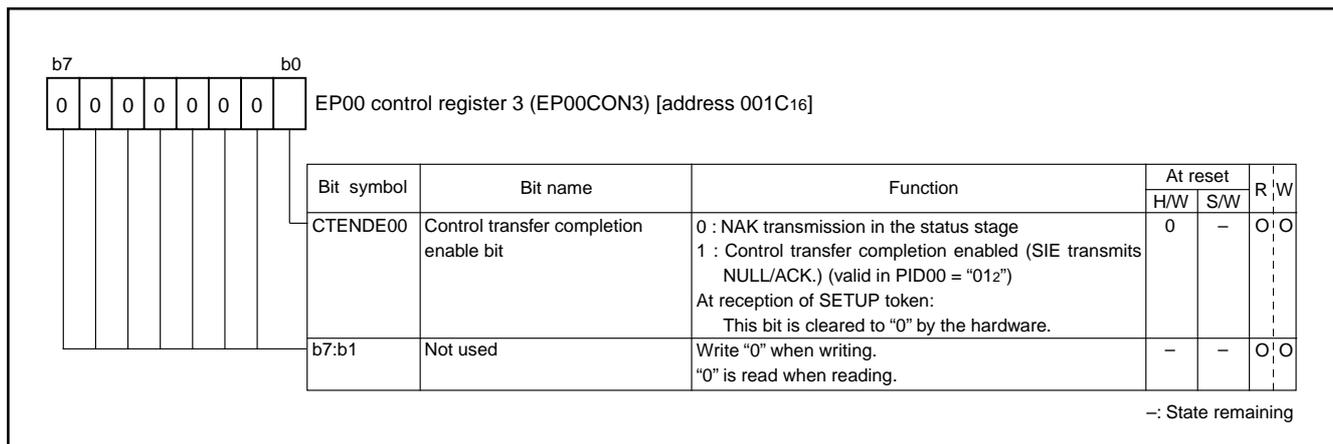


Fig. 43 Structure of EP00 control register 3

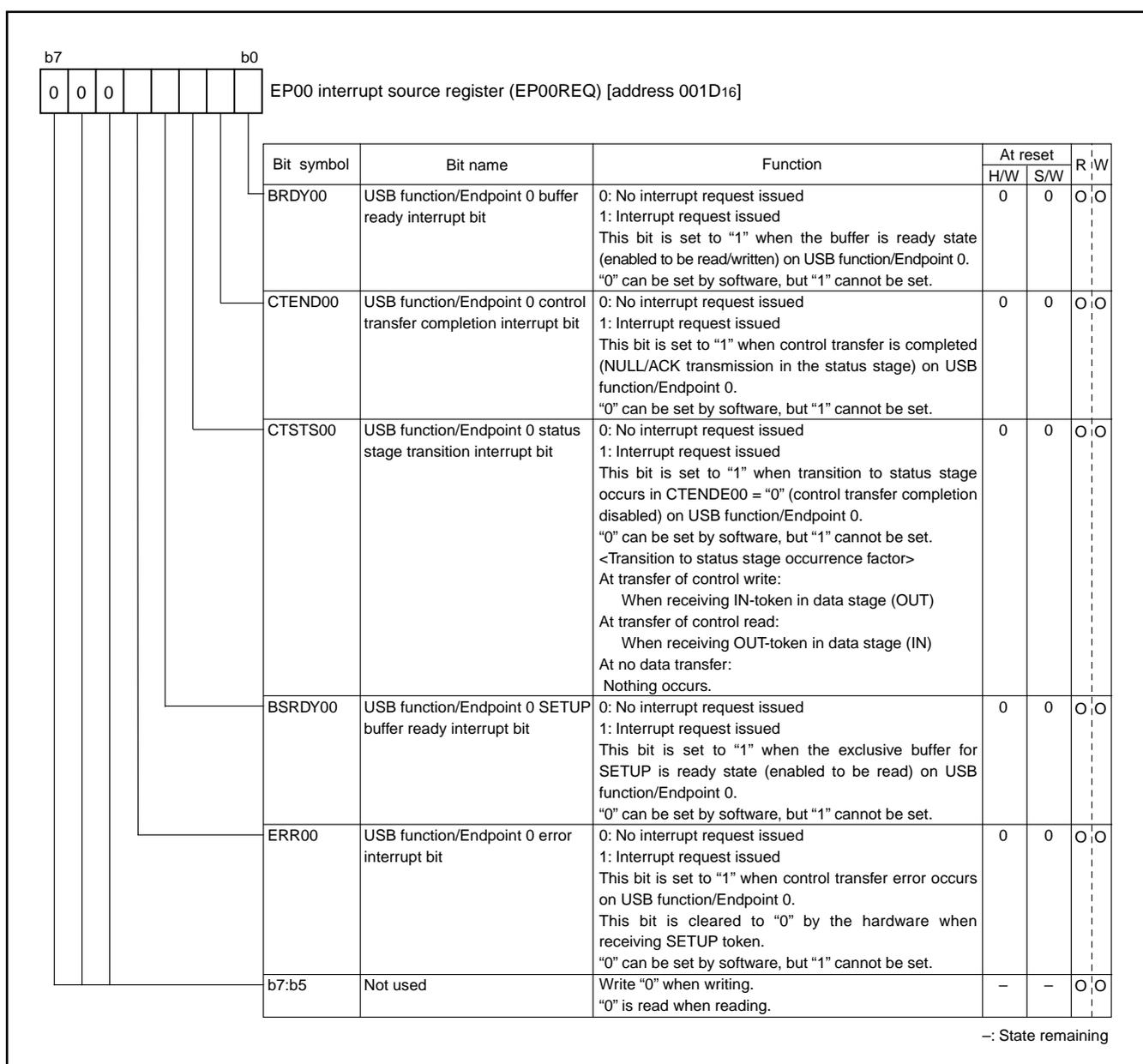


Fig. 44 Structure of EP00 interrupt source register

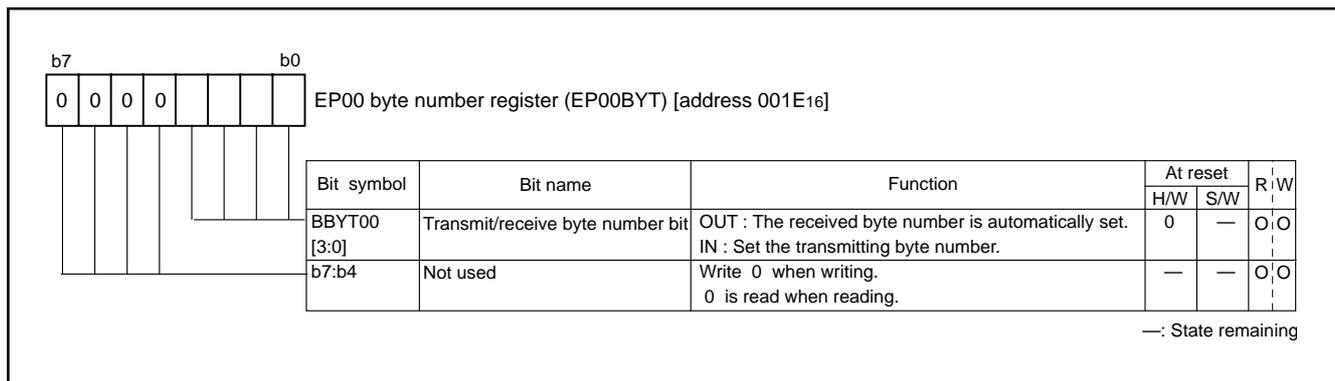


Fig. 45 Structure of EP00 byte number register

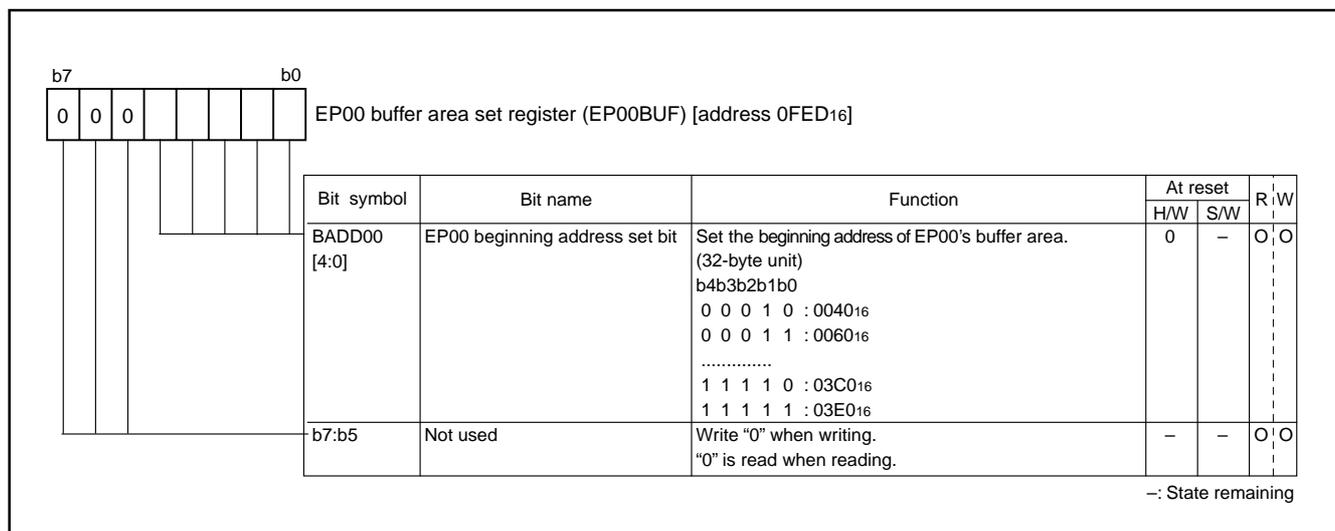


Fig. 46 Structure of EP00 buffer area set register

## (2) Endpoint 01

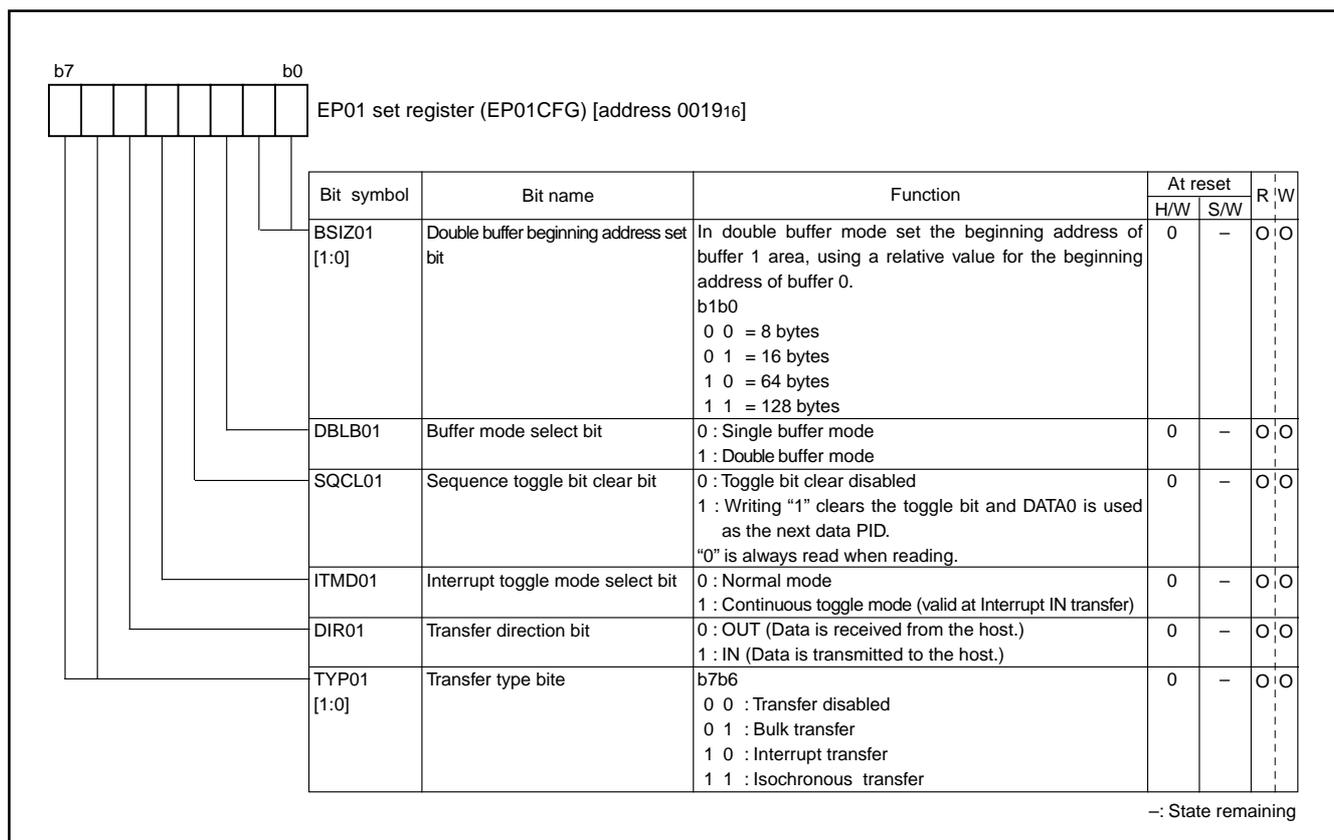


Fig. 47 Structure of EP01 set register

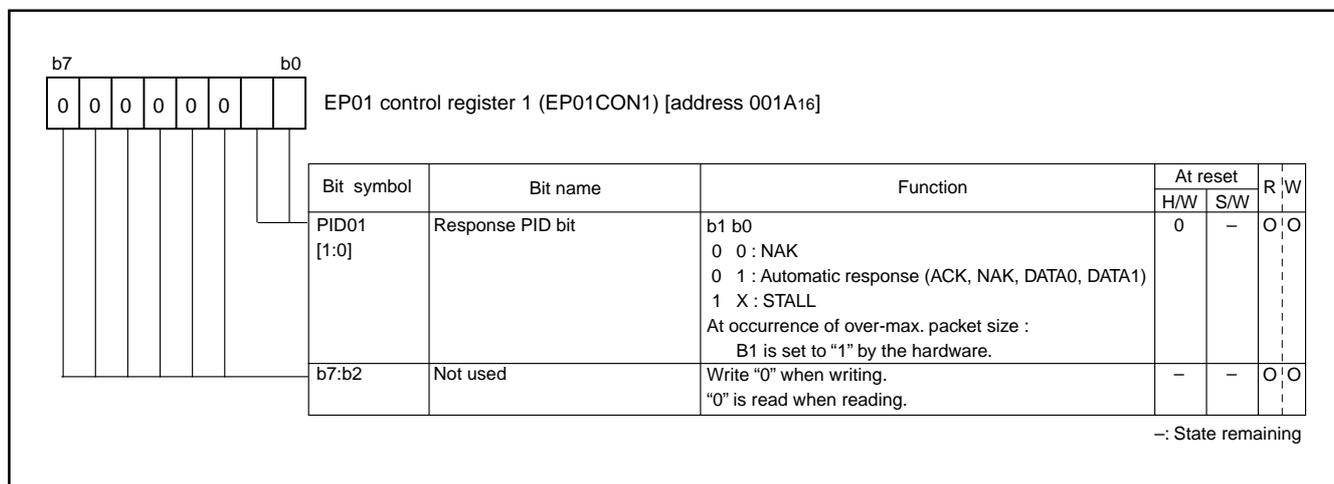


Fig. 48 Structure of EP01 control register 1

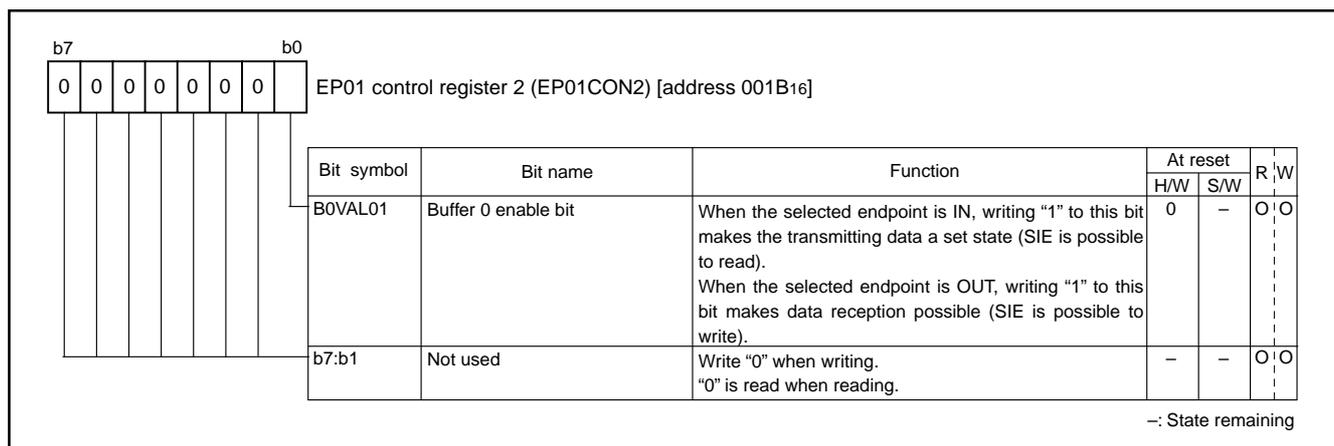


Fig. 49 Structure of EP01 control register 2

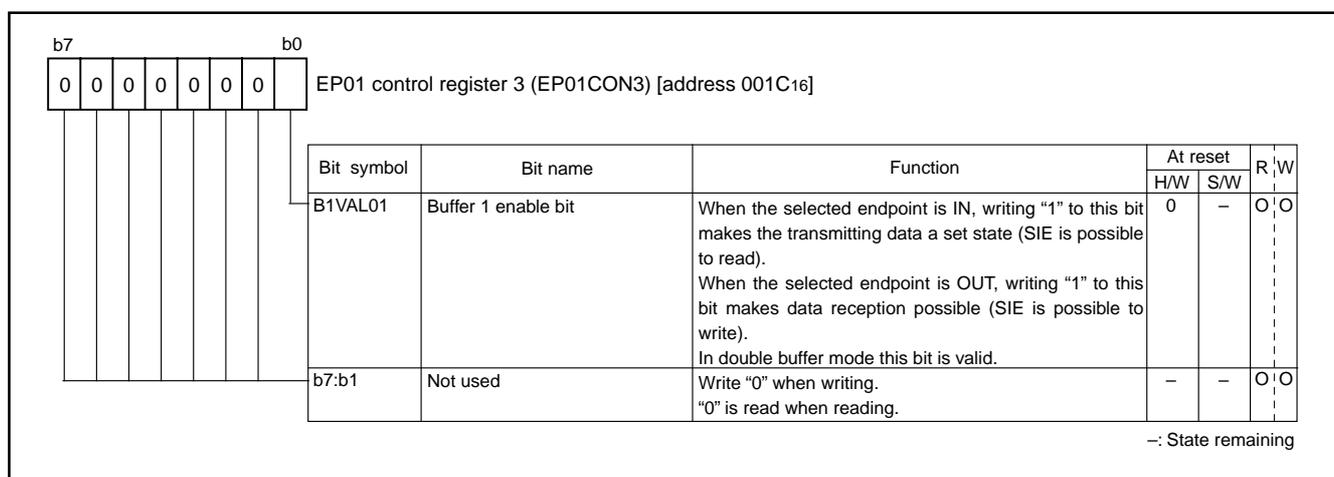


Fig.50 Structure of EP01 control register 3

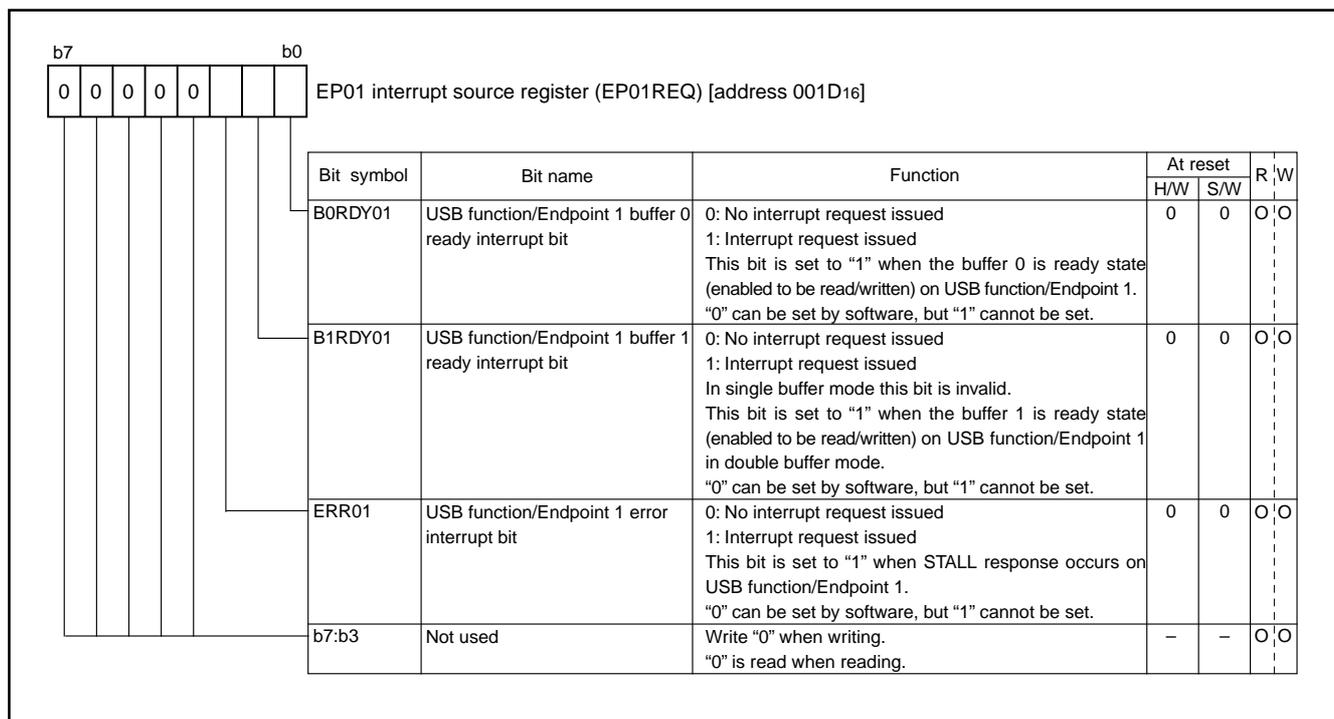


Fig. 51 Structure of EP01 interrupt source register

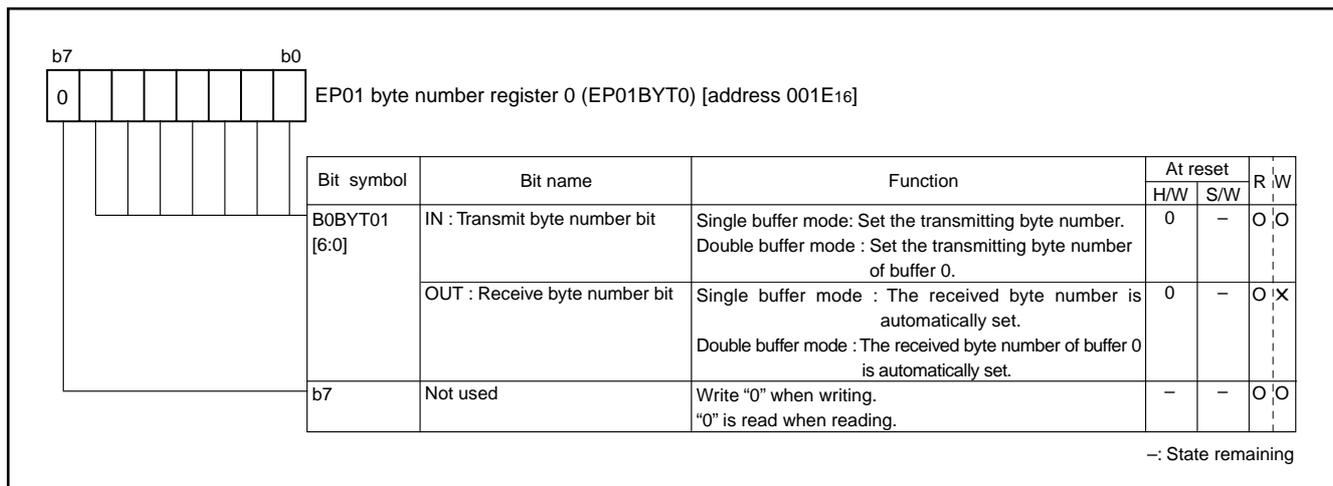


Fig. 52 Structure of EP01 byte number register 0

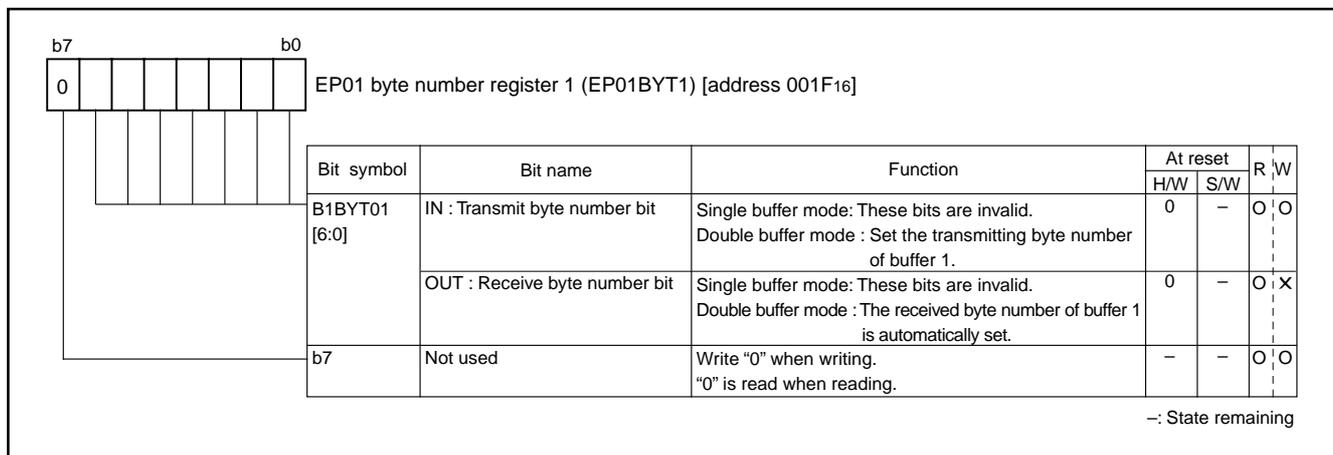


Fig. 53 Structure of EP01 byte number register 1

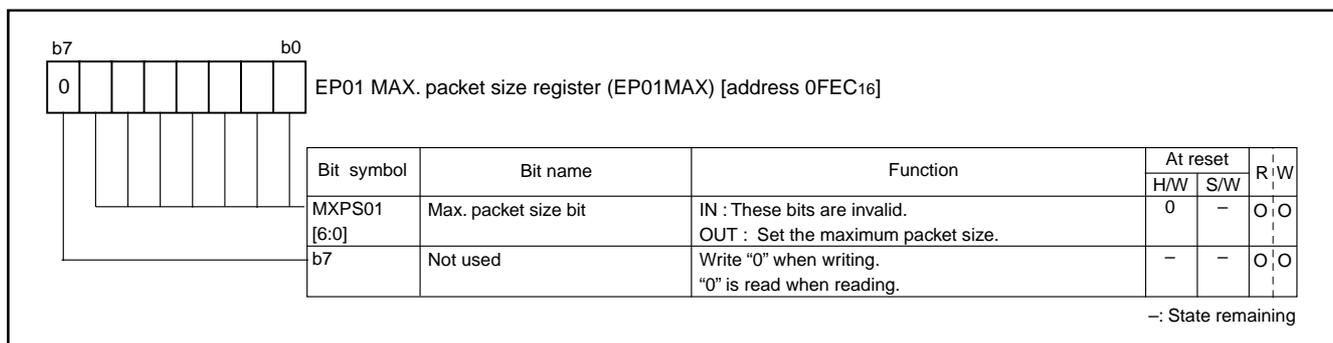


Fig. 54 Structure of EP01 MAX. packet size register

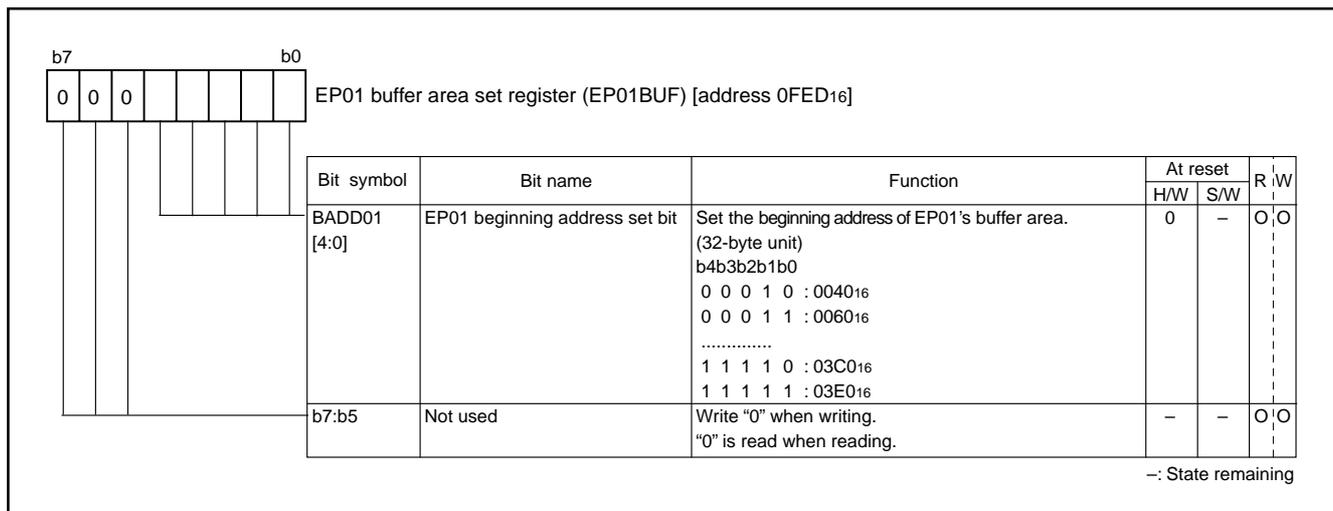


Fig. 55 Structure of EP01 buffer area set register

## (3) Endpoint 02

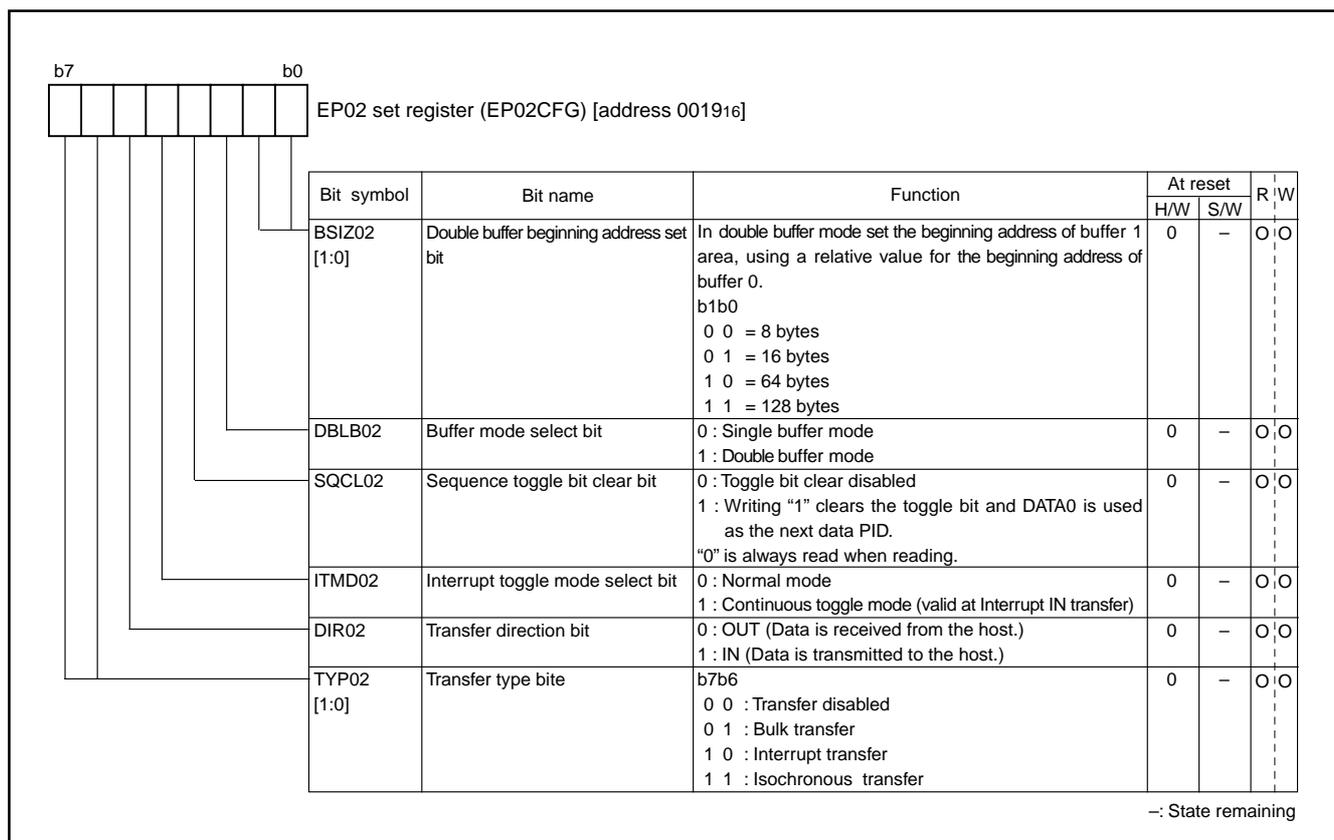


Fig. 56 Structure of EP02 set register

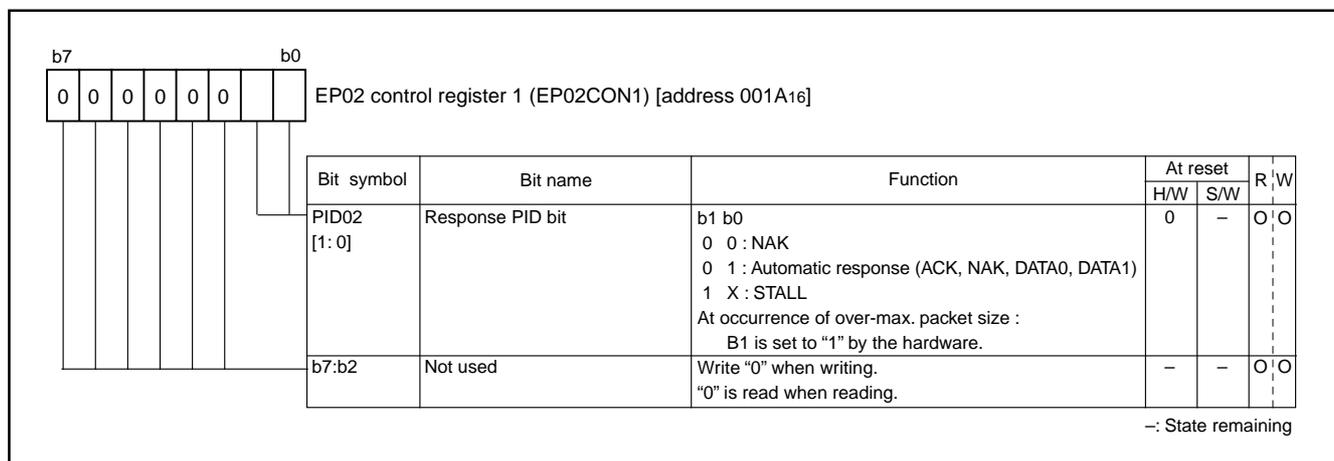


Fig. 57 Structure of EP02 control register 1

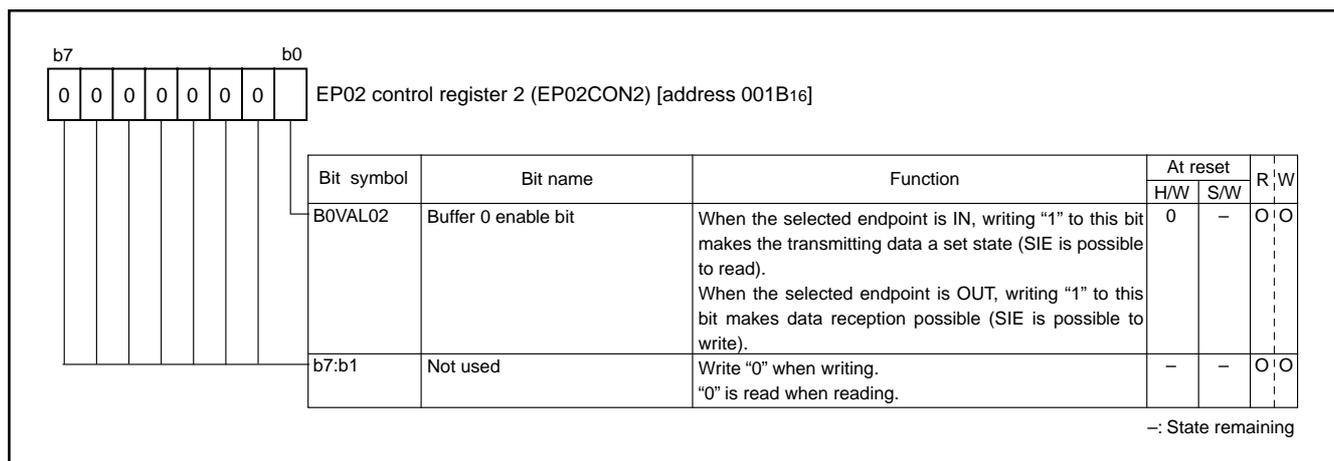


Fig. 58 Structure of EP02 control register 2

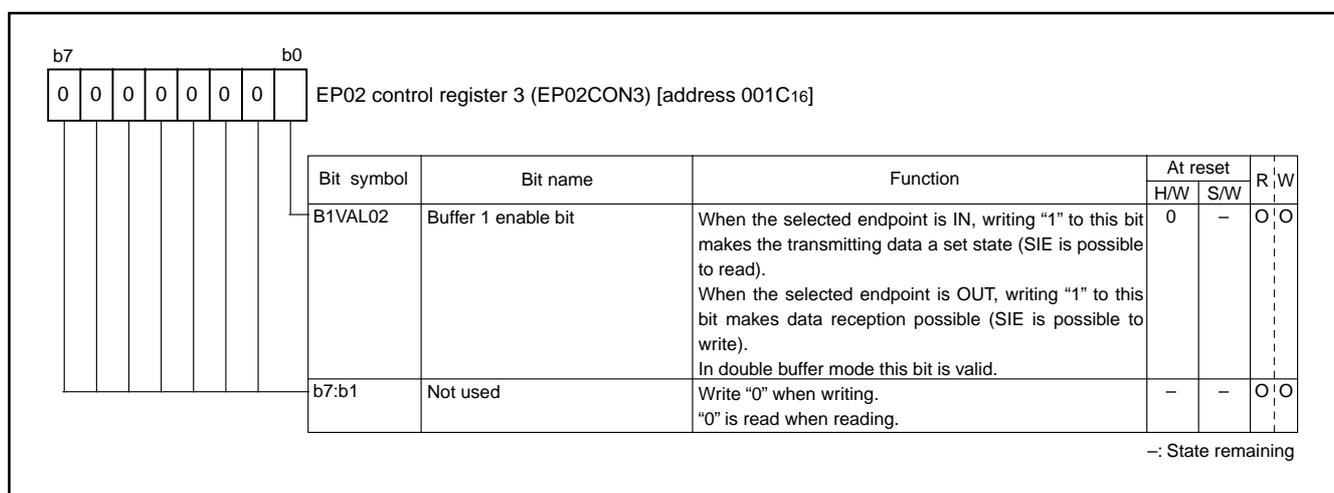


Fig. 59 Structure of EP02 control register 3

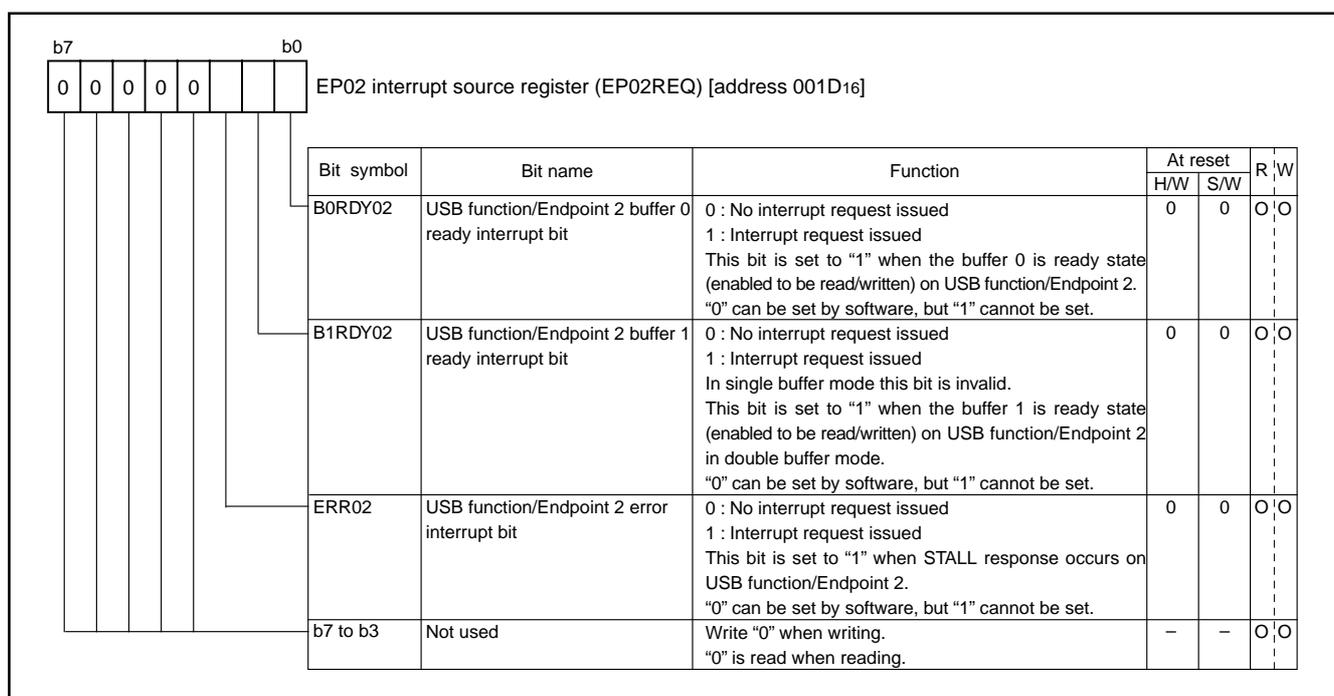


Fig. 60 Structure of EP02 interrupt source register

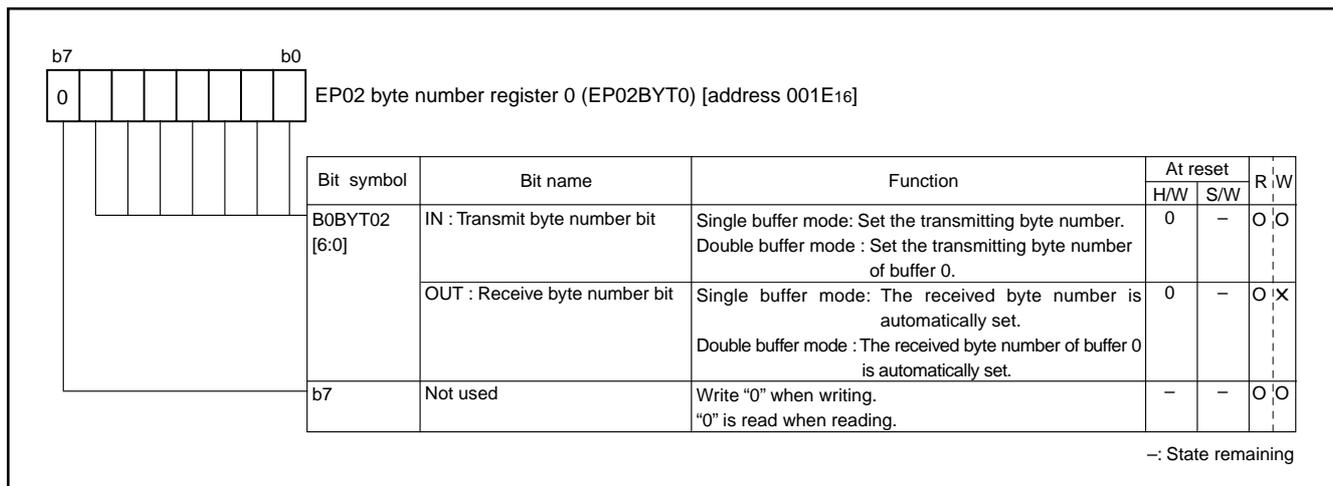


Fig. 61 Structure of EP02 byte number register 0

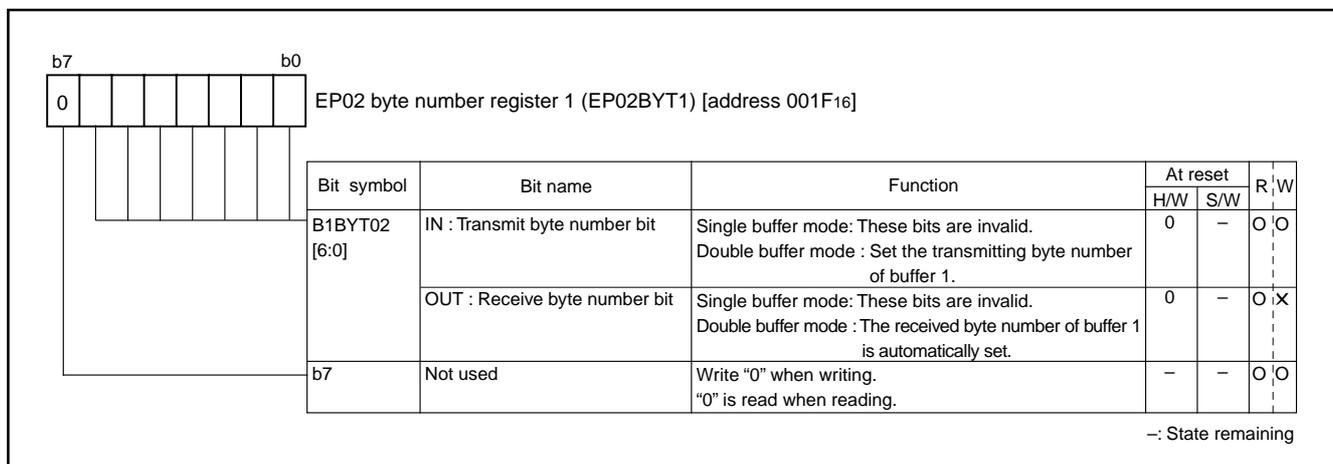


Fig. 62 Structure of EP02 byte number register 1

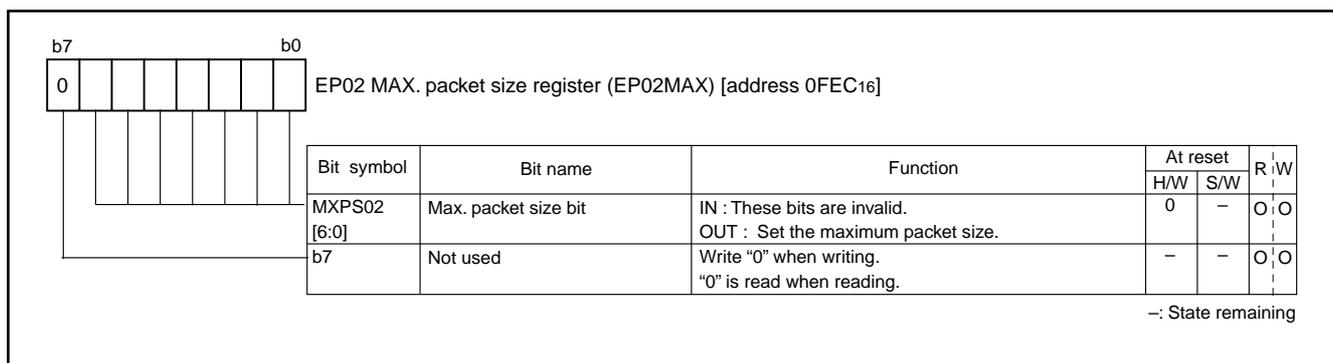


Fig. 63 Structure of EP02 MAX. packet size register

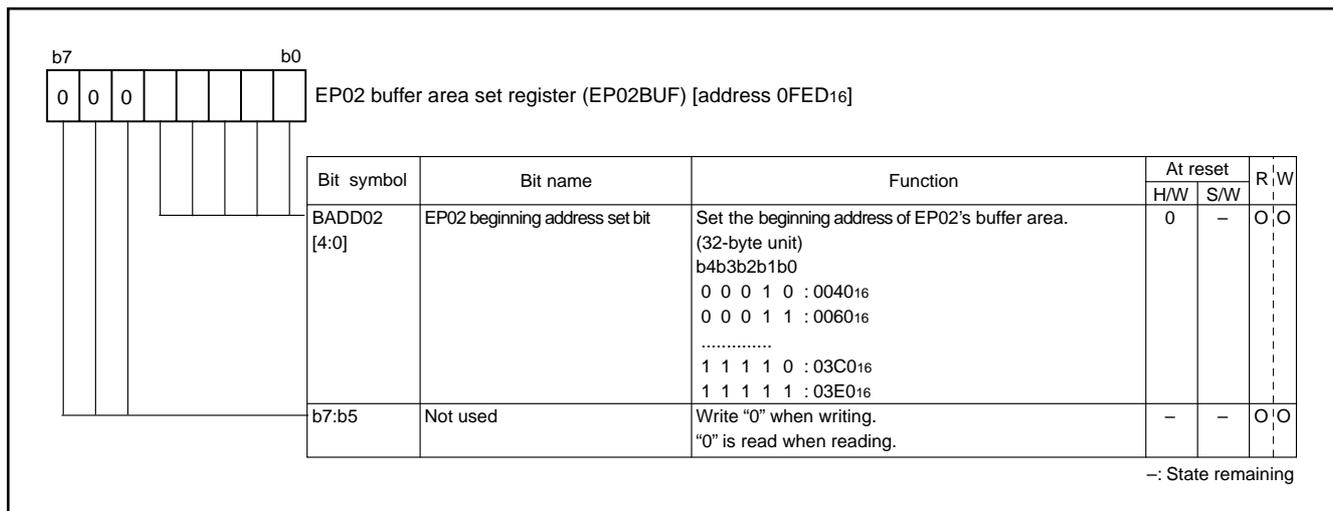


Fig. 64 Structure of EP02 buffer area set register

## (4) Endpoint 03

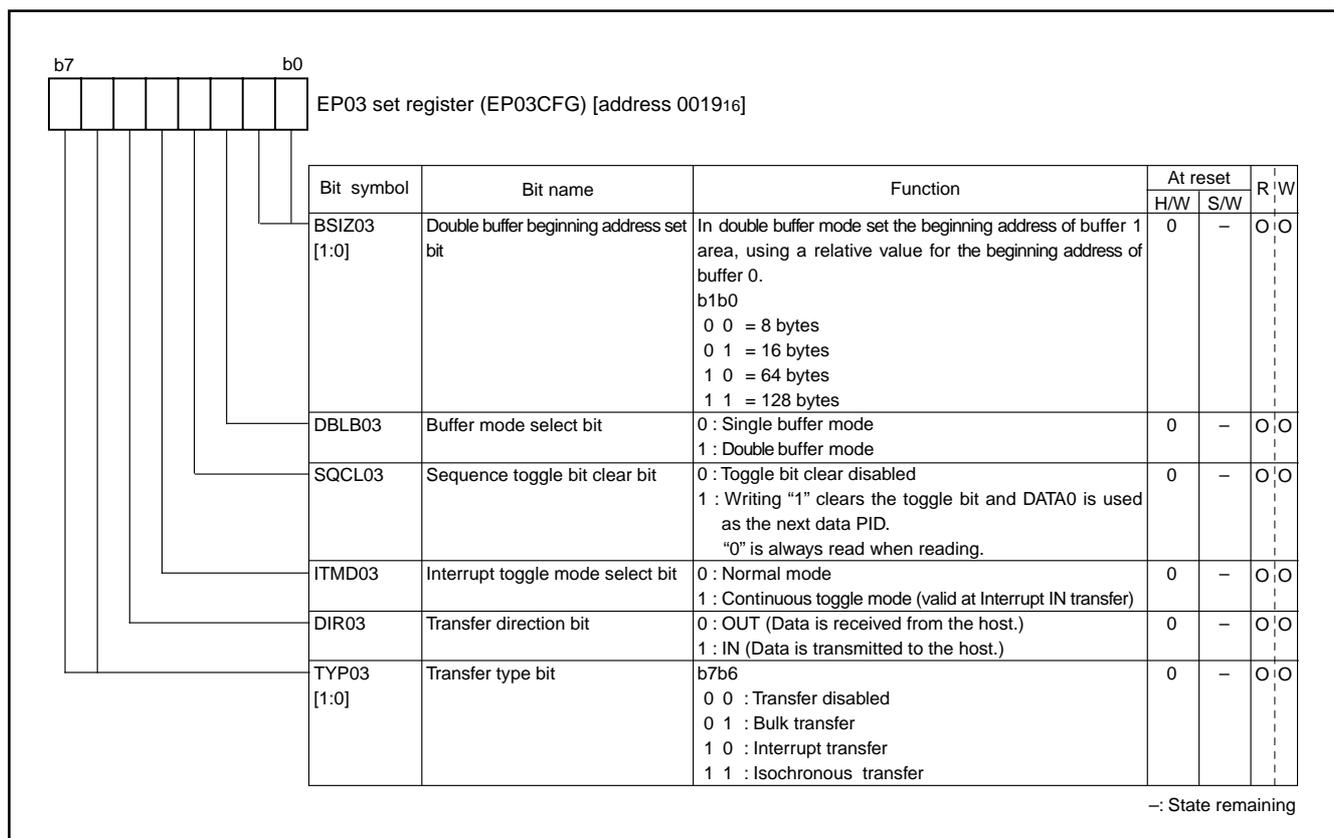


Fig. 65 Structure of EP03 set register

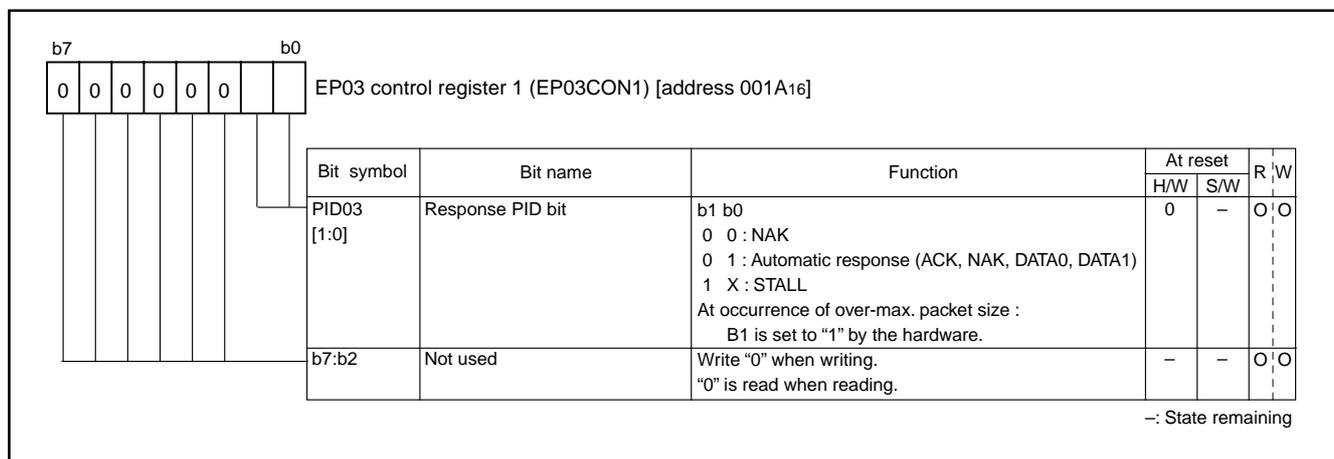


Fig. 66 Structure of EP03 control register 1

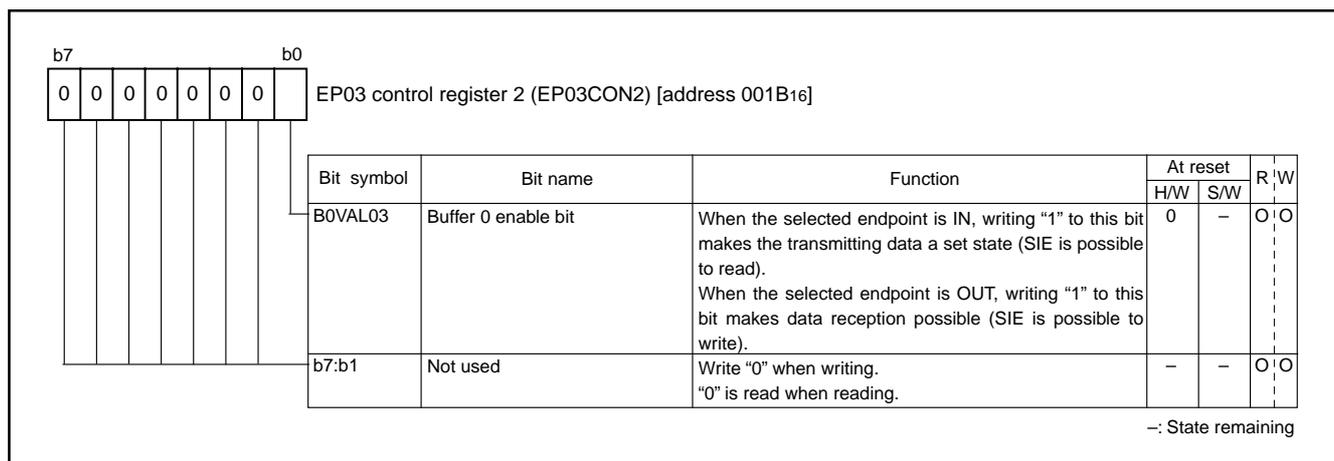


Fig. 67 Structure of EP03 control register 2

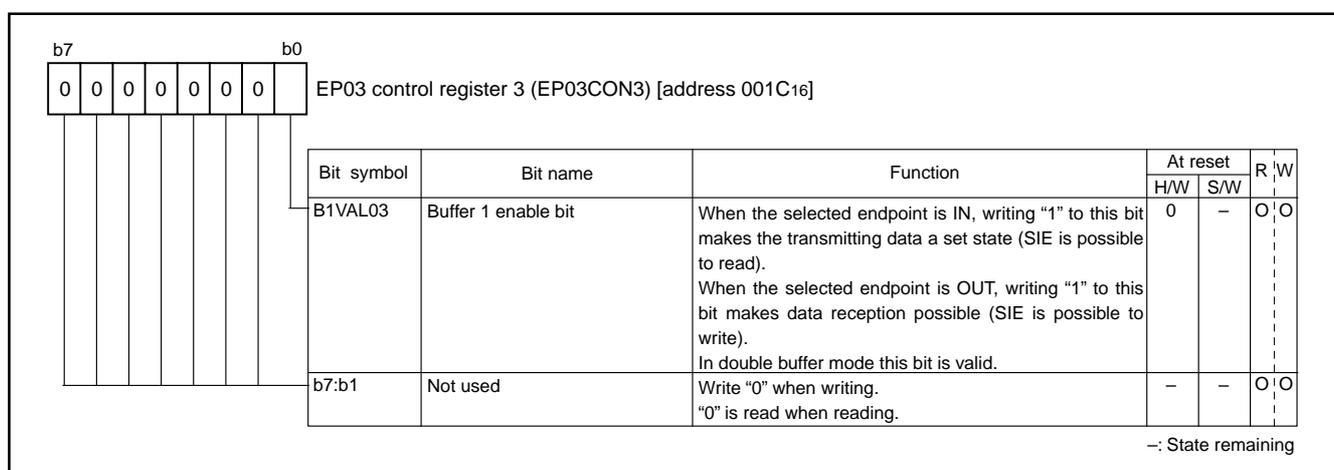


Fig. 68 Structure of EP03 control register 3

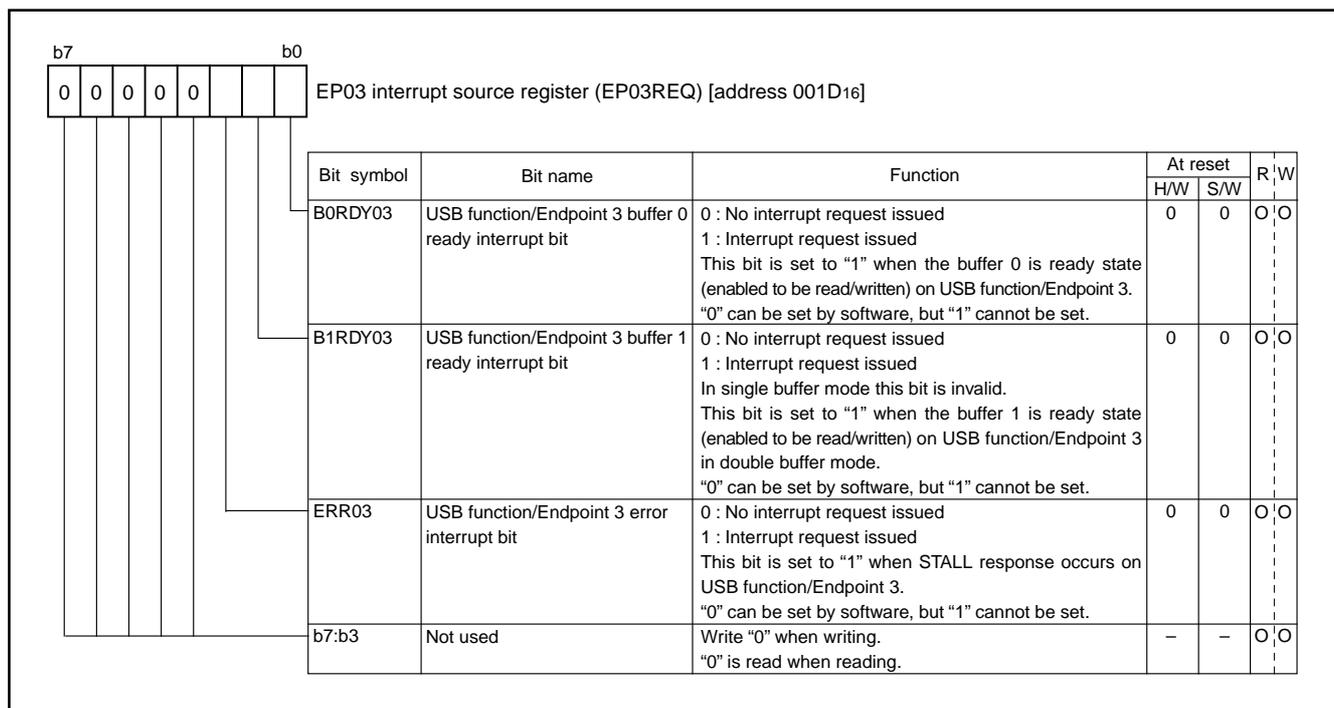


Fig. 69 Structure of EP03 interrupt source register

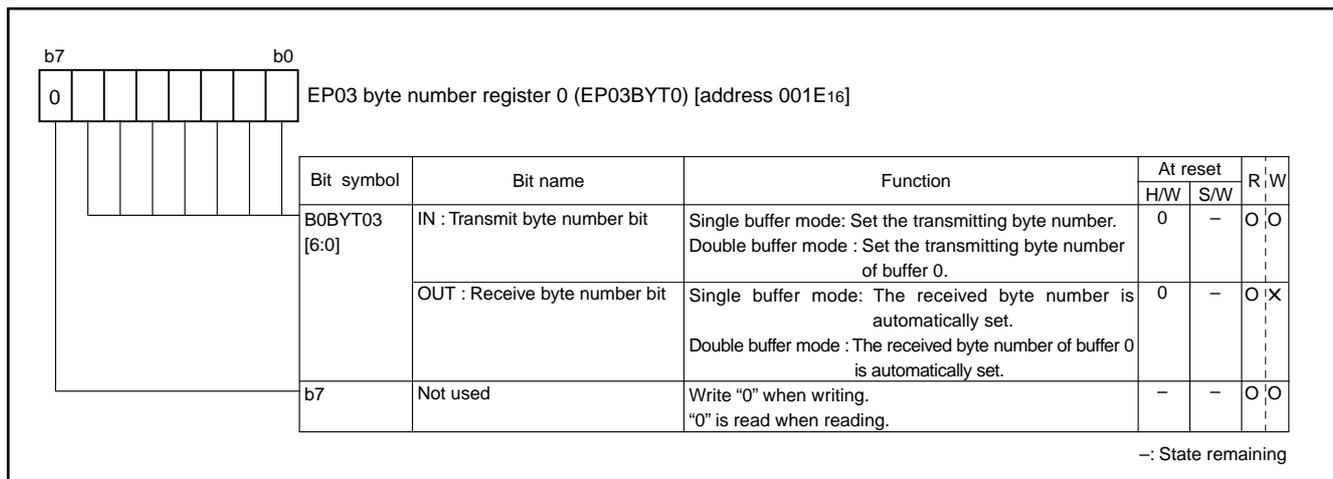


Fig. 70 Structure of EP03 byte number register 0

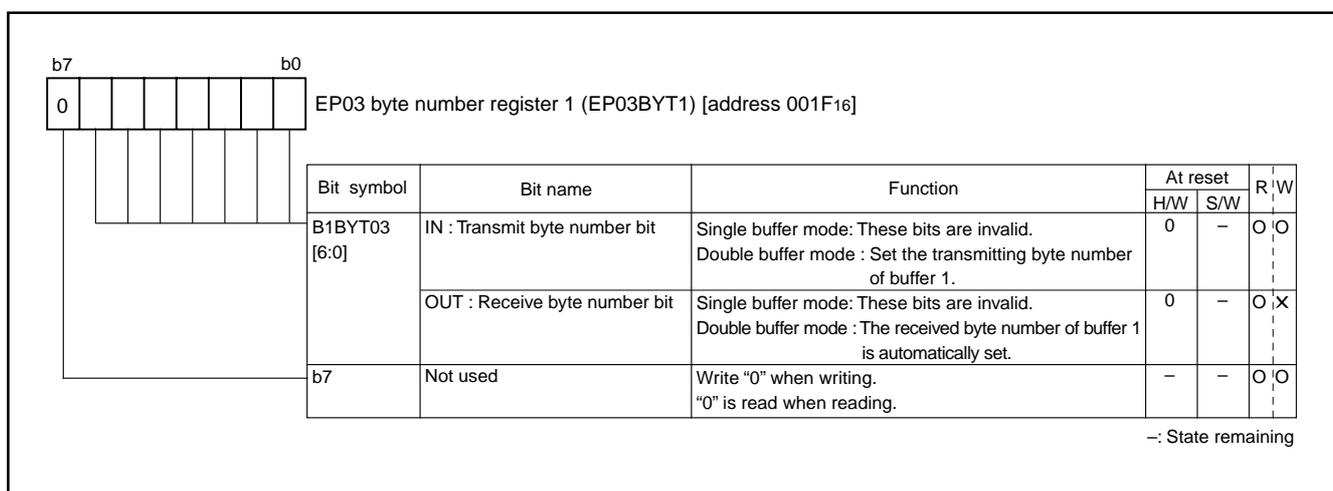


Fig. 71 Structure of EP03 byte number register 1

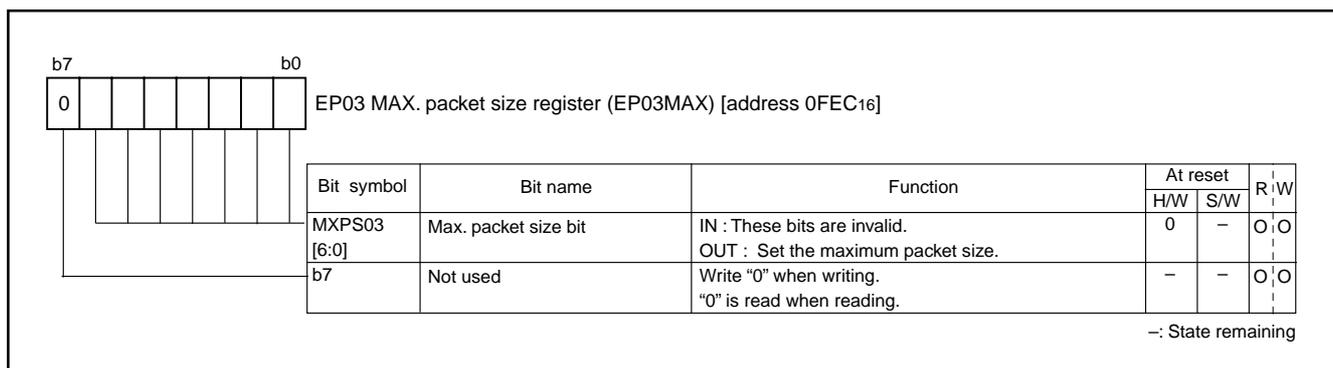


Fig. 72 Structure of EP03 MAX. packet size register

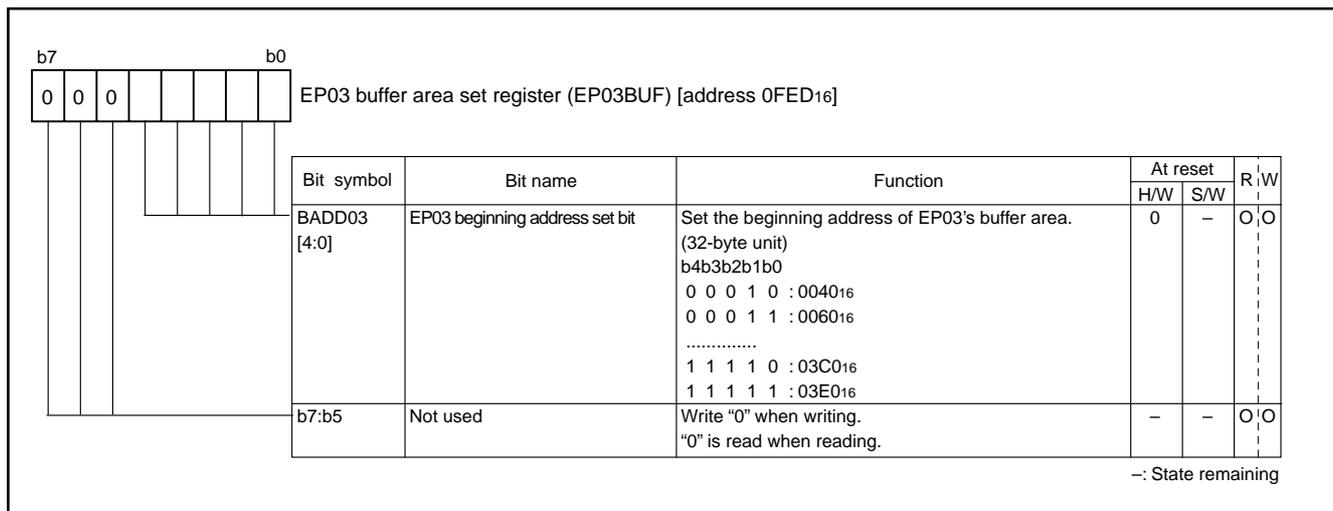


Fig. 73 Structure of EP03 buffer area set register

## (5) Endpoint 10

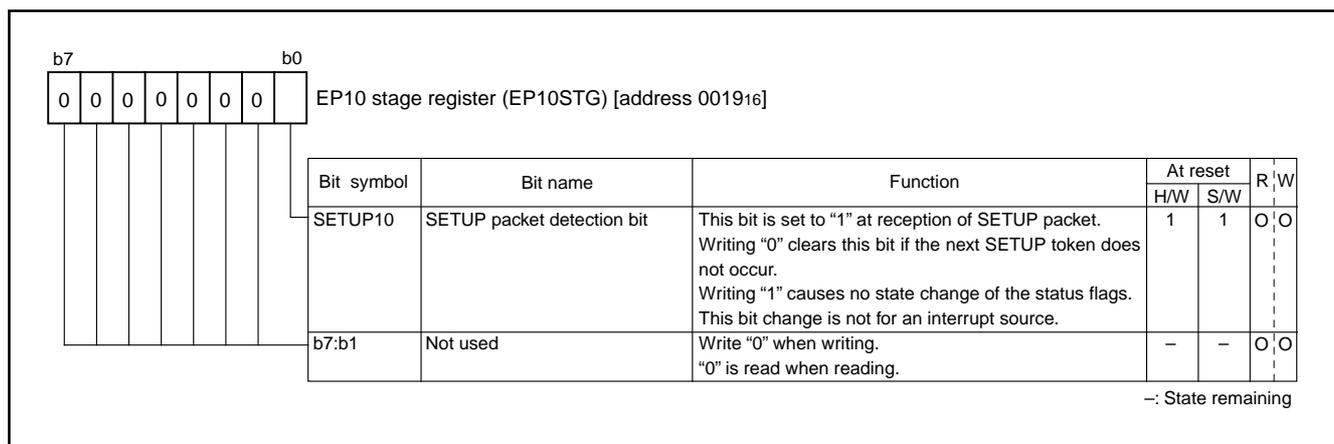


Fig. 74 Structure of EP10 stage register

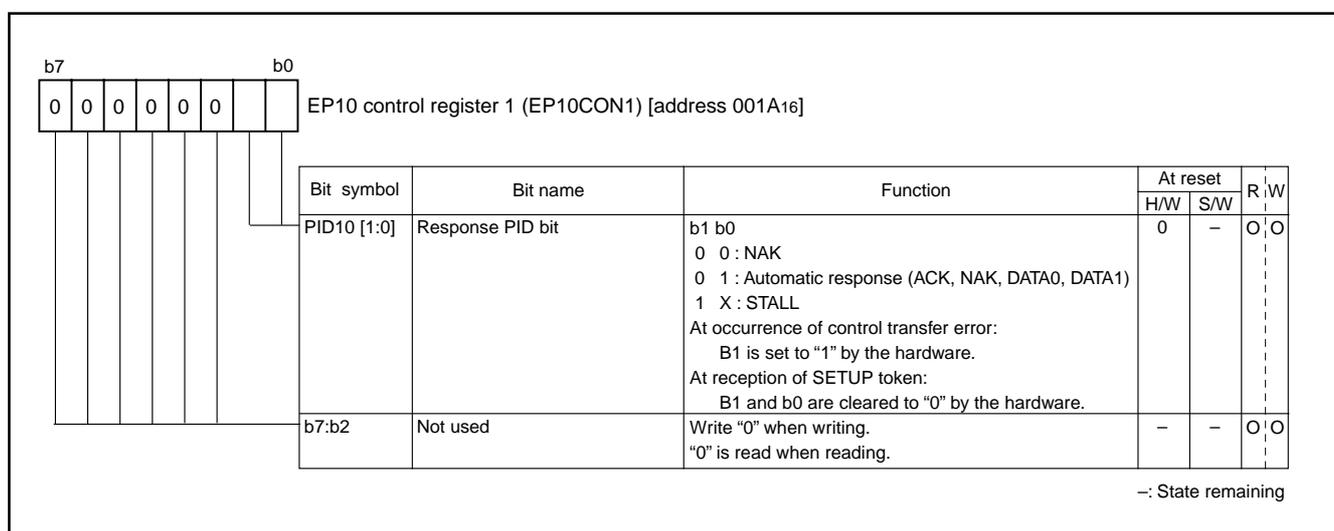


Fig. 75 Structure of EP10 control register 1

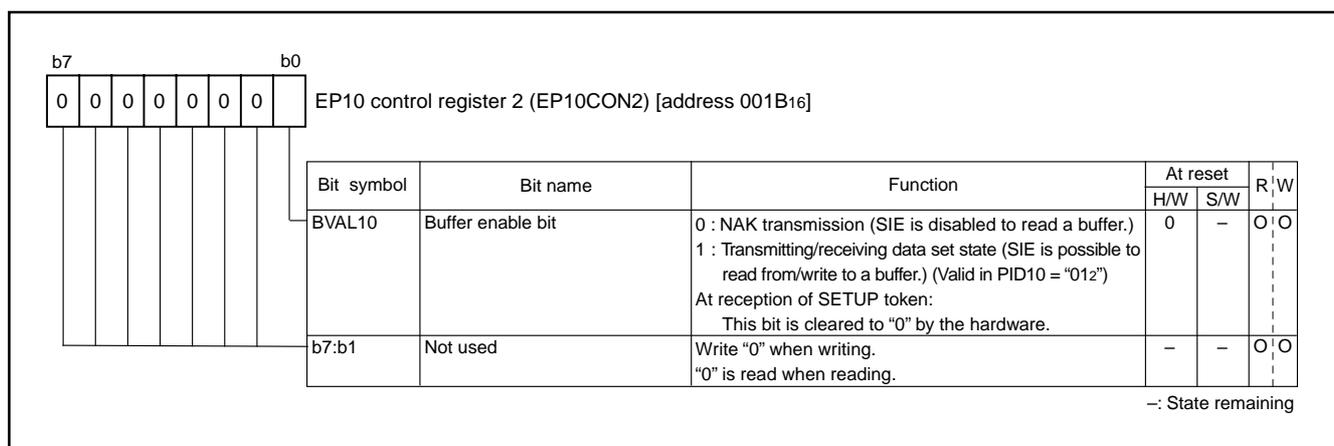


Fig. 76 Structure of EP10 control register 2

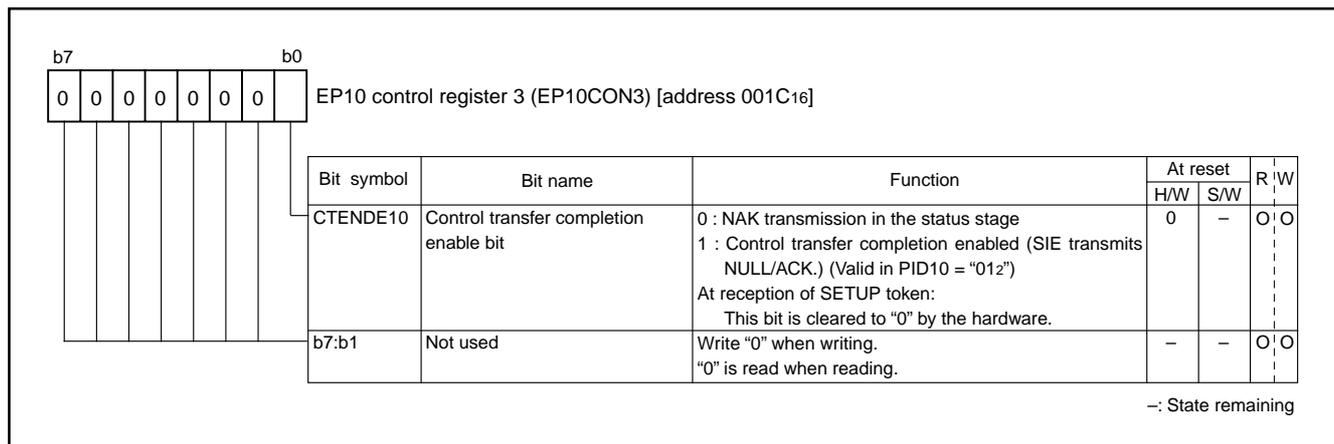


Fig. 77 Structure of EP10 control register 3

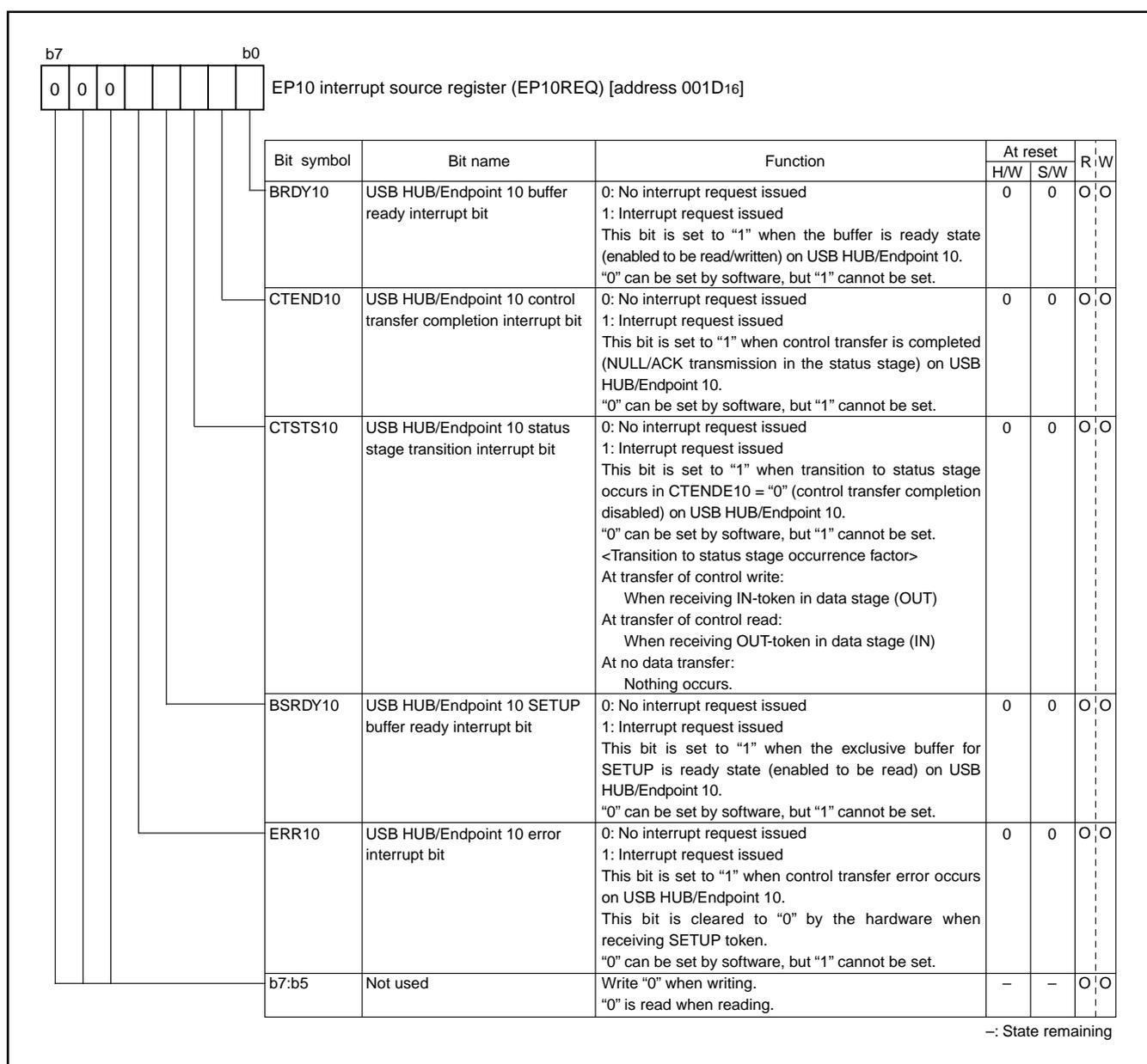


Fig. 78 Structure of EP10 interrupt source register

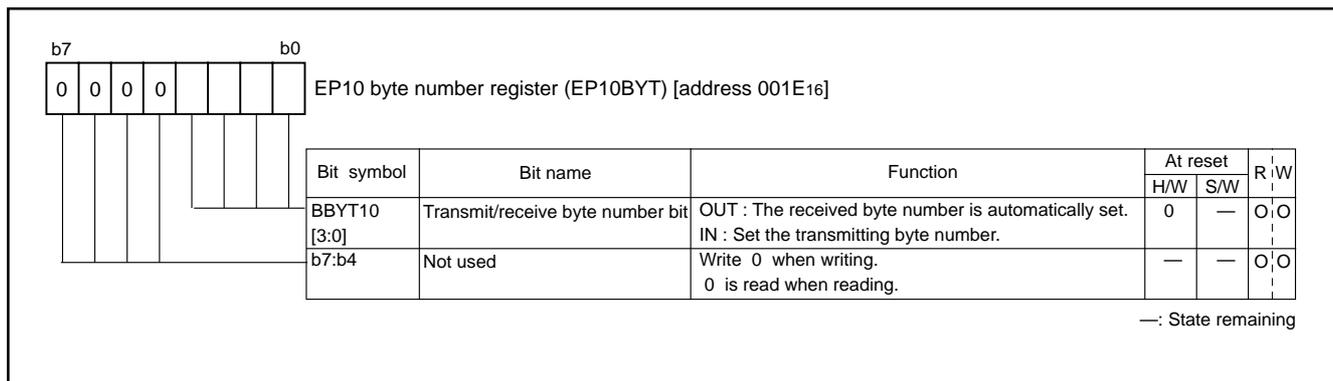


Fig. 79 Structure of EP10 byte number register

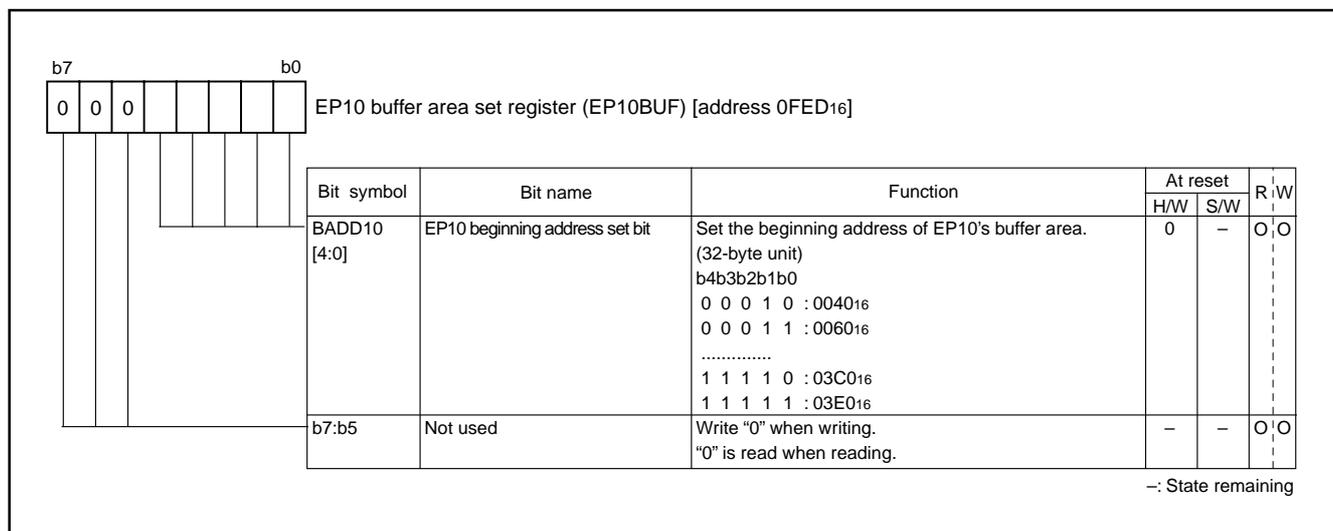


Fig. 80 Structure of EP10 buffer area set register

## (6) Endpoint 11

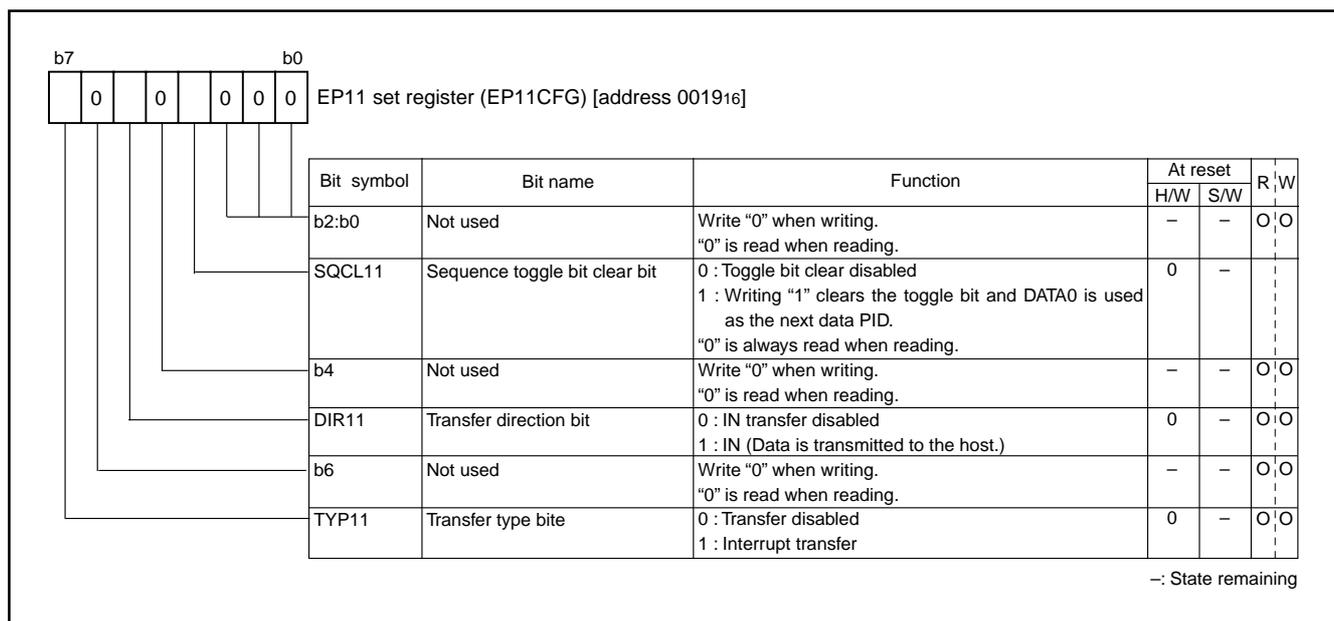


Fig. 81 Structure of EP11 set register

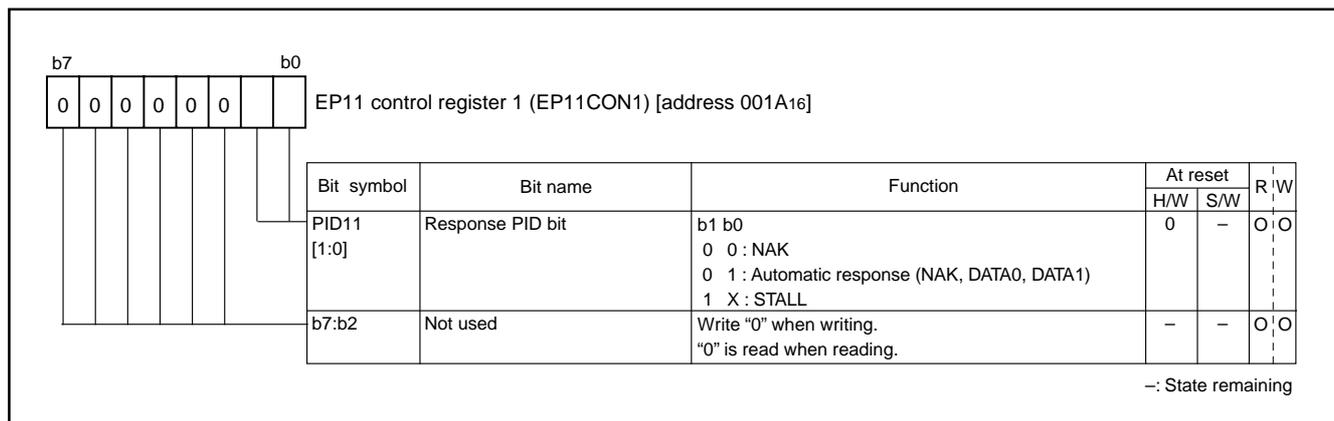


Fig. 82 Structure of EP11 control register 1

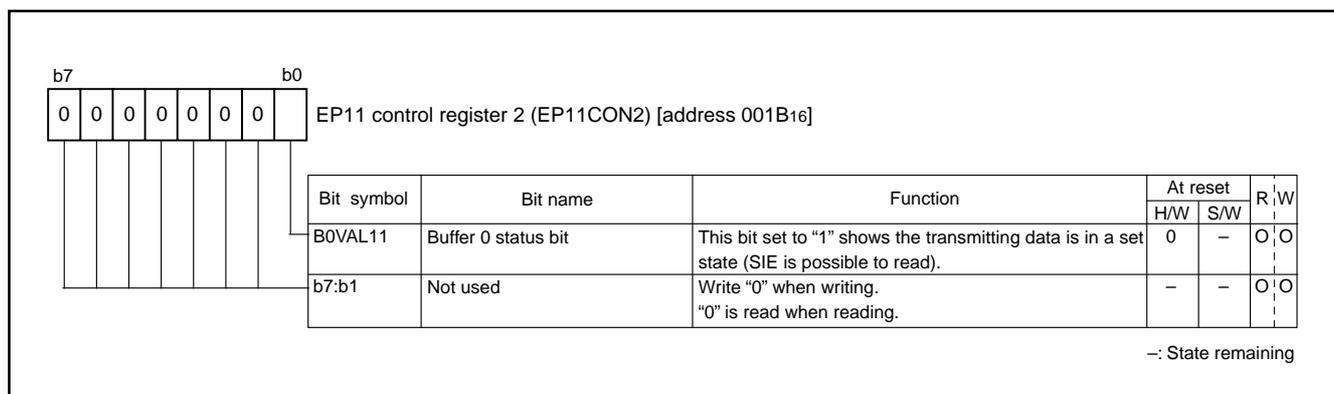


Fig. 83 Structure of EP11 control register 2

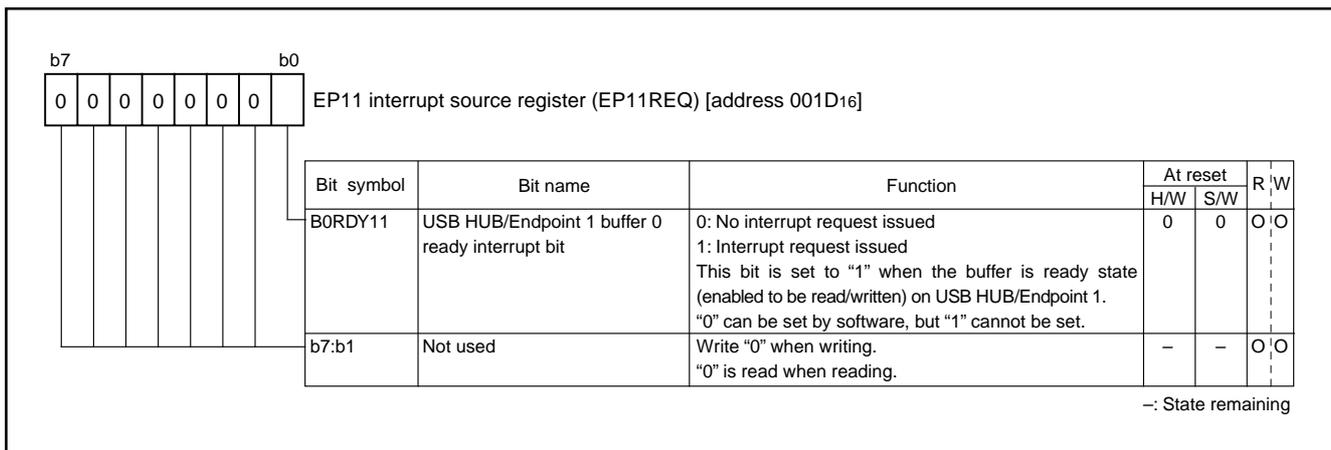


Fig. 84 Structure of EP11 interrupt source register

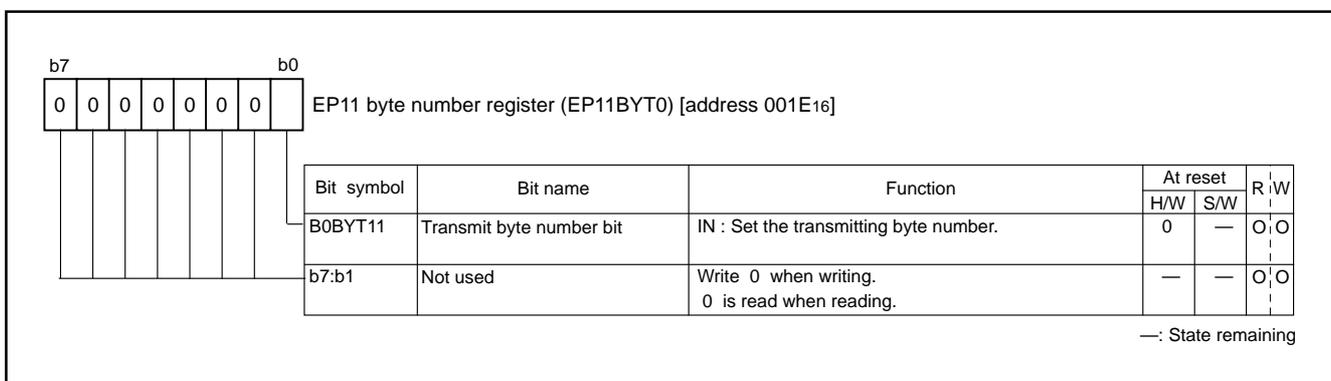


Fig. 85 Structure of EP11 byte number register

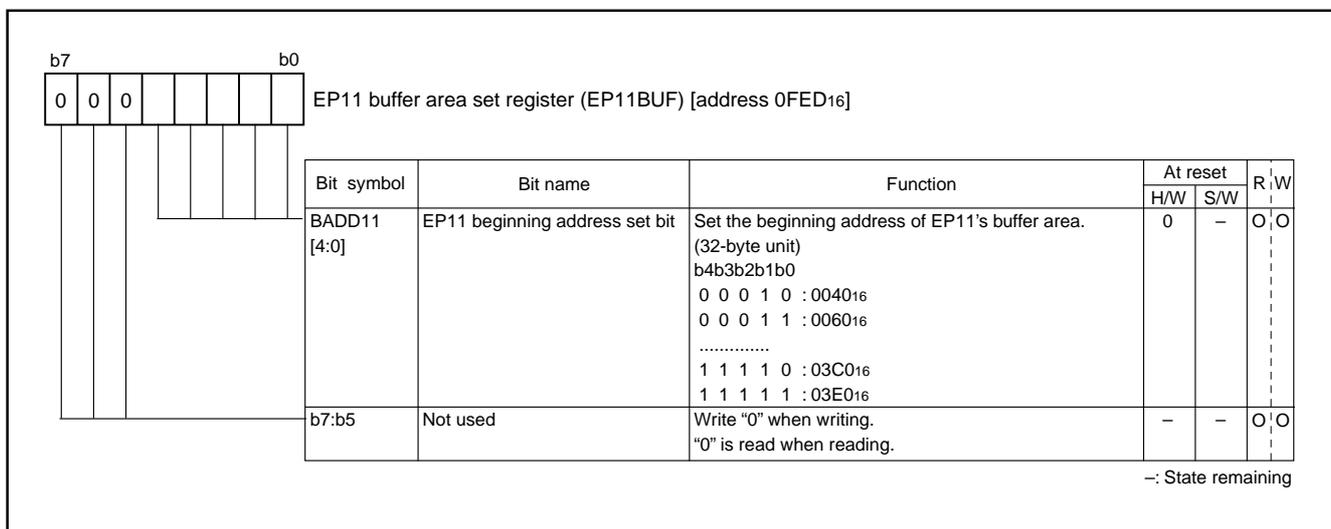


Fig. 86 Structure of EP11 buffer area set register

## HUB FUNCTION

The 38K2 Group has a HUB Function Control Circuit (HUBFCC) that offers easy implementation of USB-hub functions (signal repeat and bus state detection). This circuit is in compliance with USB Specification Version 2.0 Full-Speed/Low-Speed Transfer Modes (12 Mbps/1.5 Mbps, equivalent to Version 1.1).

The HUBFCC operates with two external down-ports and one internal down-port, which is utilized by the USB addresses of the built-in peripherals, enabling management of a total of three down-ports independently.

A dedicated circuit automatically performs the bus state change detection and error detection needed for the sequence management of the hub repeater circuit, data repeat function, and down-port status management. This dedicated control circuit ensures the user easy development of a program or timing design.

Each down-port register can be controlled by USB commands using USB addresses for HUB functions or detecting changes in the bus state of down-ports. The HUBFCC is also equipped with a remote wakeup signal transfer function for use during global resume as other special signals management. The HUBFCC generates an interrupt to the CPU when detecting a down-port state change (1 vector, 10 sources).

The flexibility of the indispensable yet wide-ranging HUBFCC structure and an external interrupt function and I/O ports implemented in the standard features of this MCU enable the power supply management essential for USB-HUB functions and also allow users to easily and effortlessly configure their optimum system.

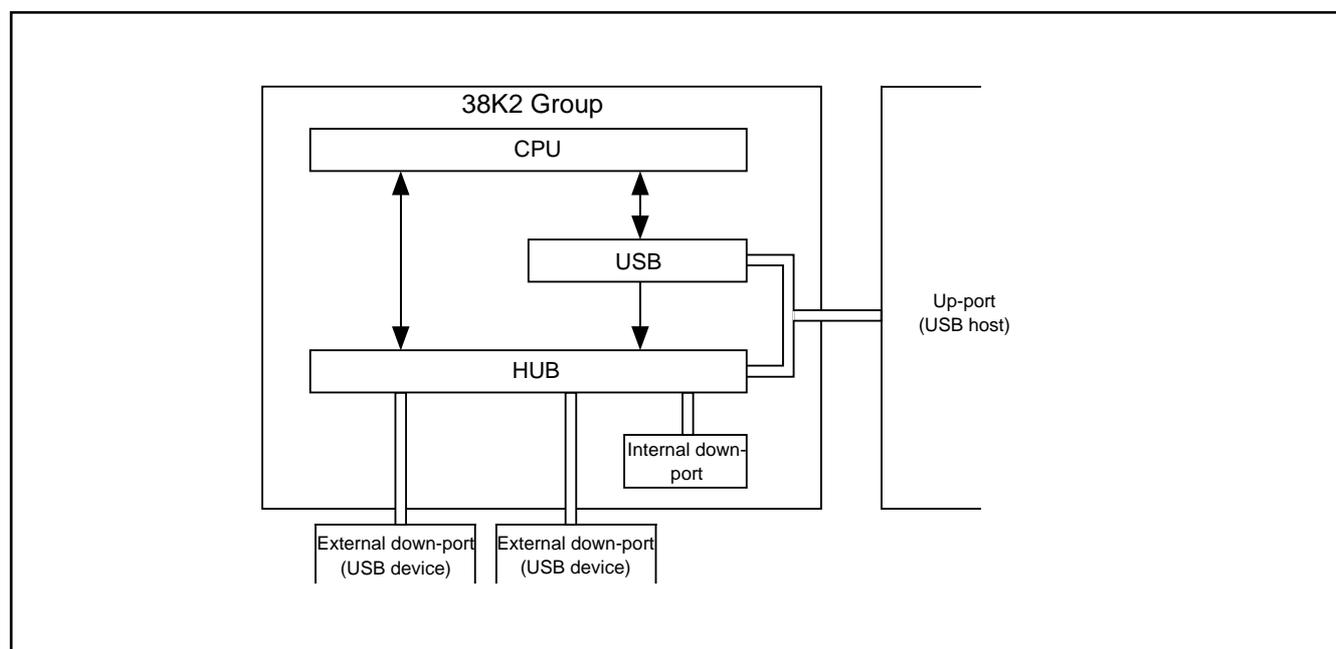


Fig. 87 HUB functions

## HUB Function Control Circuit Block Diagram

The HUB function control circuit, as shown in the diagram below, consists of the following blocks.

- (1) HUB repeater block
- (2) Down-port control block
- (3) CPU interface block (CIF)

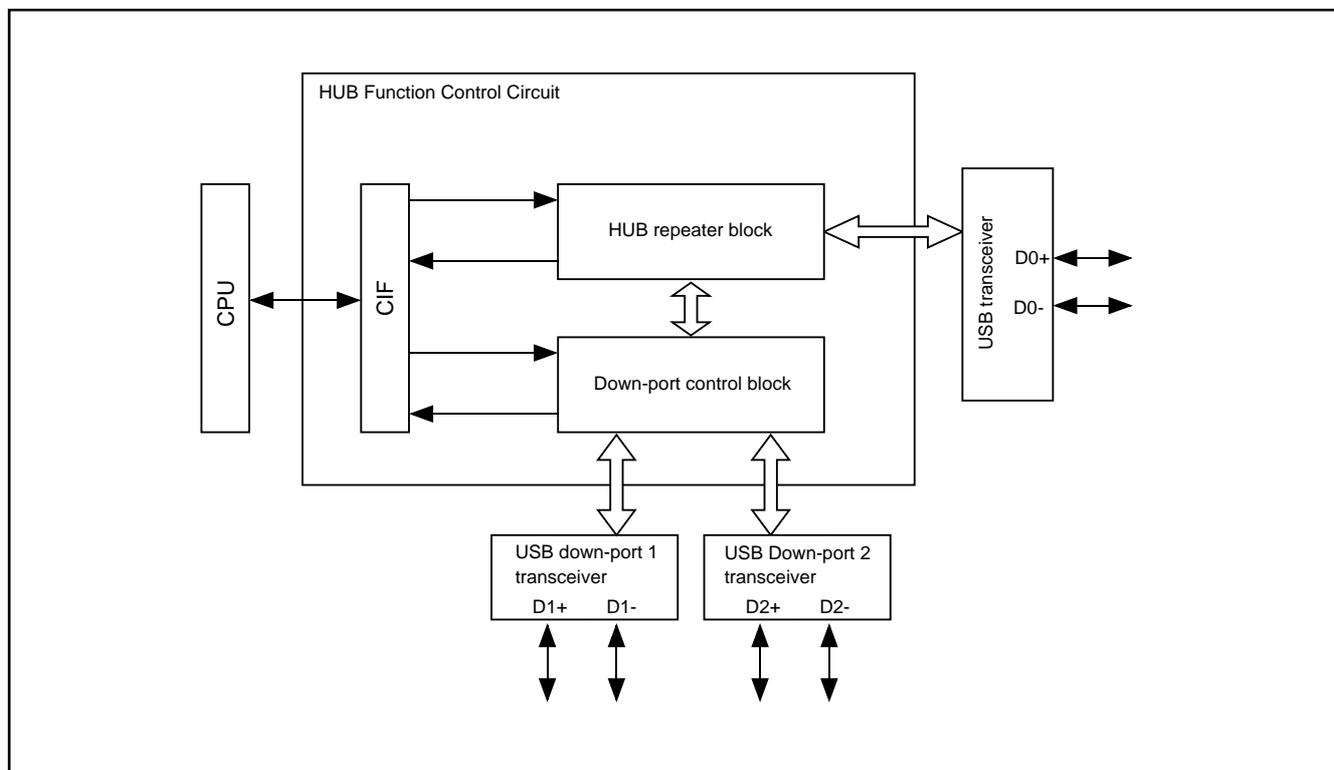


Fig. 88 HUB function control circuit block diagram

### (1) HUB repeater block

The HUB repeater block, consisting of the circuits listed below, processes the HUB repeater function sequence. The HUB repeater is ready for operation after enabling the USB module (USBE = "1").

- Repeater circuit (detects SOP/EOP signal)
- Frame-time circuit (synchronizes to SOF signal and manages frames in 1 ms)
- Receiver circuit (manages up-port states)
- Transmitter circuit (controls up-port outputs)

### (2) Down-port control block

The down-port control block, consisting of the circuits listed below, performs down-port controls under supervision of the HUB repeater state operation.

- Down-port sequencer circuit
- Down-port state change detect circuit

### (3) CPU interface block (CIF)

The CPU interface block performs the following processes.

- Control of repeater/down-port states through registers.
- Generates interrupt signal
- Controls internal bus interface

### USB Down-port Peripheral Circuit Setting

The USB down-port peripheral circuits can be set with the down-stream port control register (address 0FF916). Figures 89 and 90 show the circuit block diagrams.

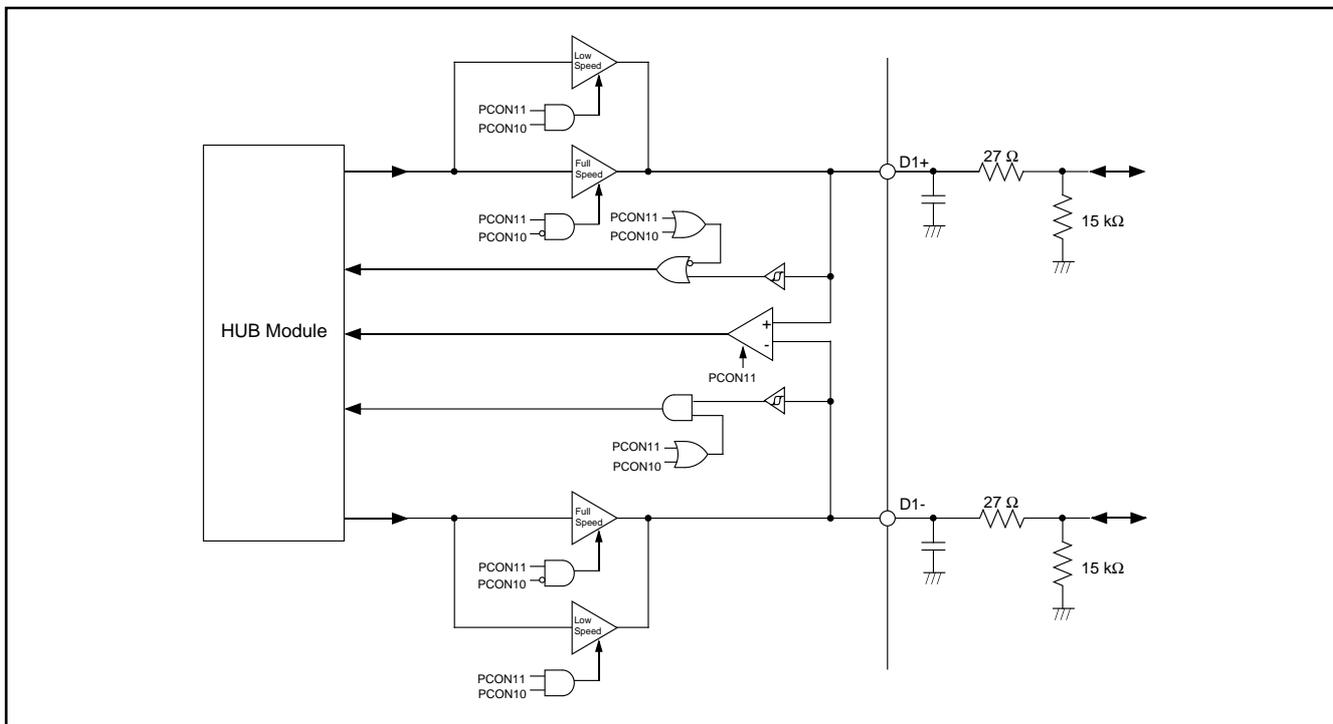


Fig. 89 Block diagram of USB down-port peripheral circuits (D1+, D1-)

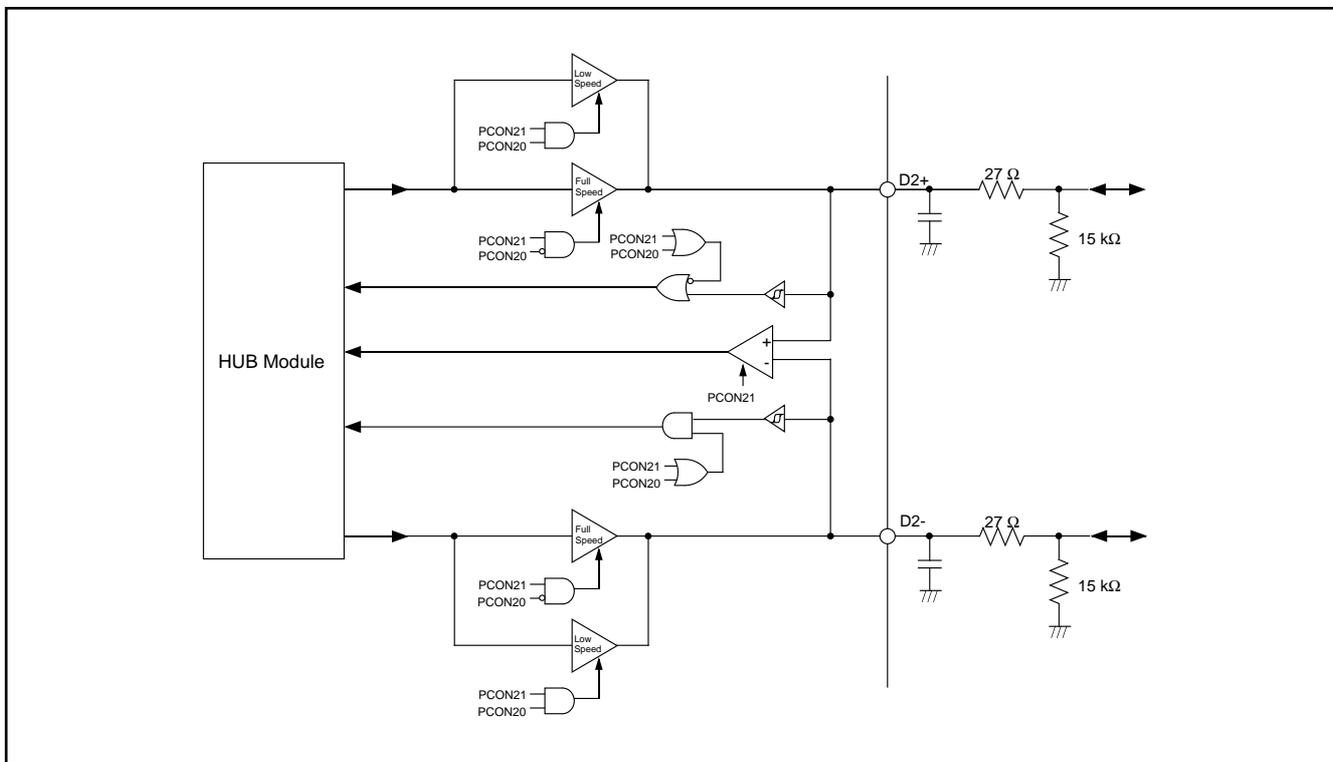


Fig. 90 Block diagram of USB down-port peripheral circuits (D2+, D2-)

### HUB Interrupt Function

The HUB function control circuit has one interrupt request consisting of 10 interrupt sources each of which can be determined through the interrupt source register. Table 8 shows the HUB interrupt sources.

Table 8 HUB interrupt sources

Interrupt request bit (IREQ2: Address 003D16)	HUB interrupt bit (HUBIREQ: Address 002916)	Interrupt source
USB HUB	DP1	At HUB down-port 1 state change detected: <ul style="list-style-type: none"> <li>•Disconnected state detected</li> <li>•Connected state detected</li> <li>•Port error state detected</li> <li>•Resume signal detected</li> <li>•Bus state change detected</li> </ul>
	DP2	At HUB down-port 2 state change detected: <ul style="list-style-type: none"> <li>•Disconnected state detected</li> <li>•Connected state detected</li> <li>•Port error state detected</li> <li>•Resume signal detected</li> <li>•Bus state change detected</li> </ul>

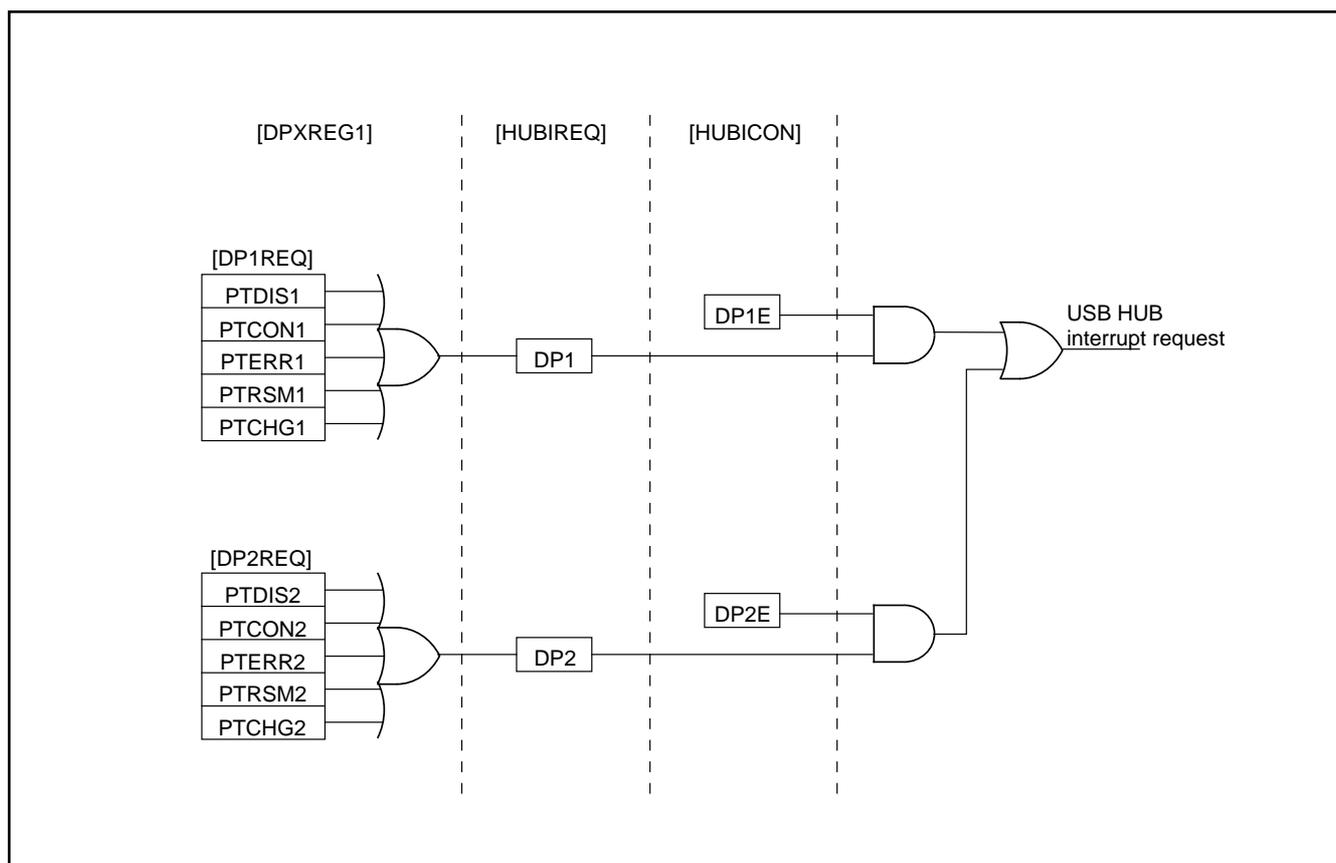


Fig. 91 USB HUB interrupt control

### HUB Register List

The HUB register list is shown below.

Address	Register Name	SYMBOL	USB SFR								
			bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
002816	HUB interrupt source enable register	HUBICON	HRWUE							DP2E	DP1E
002916	HUB interrupt source register	HUBIREQ	HRWU							DP2	DP1
002A16	HUB downstream port index register	HUBINDEX									DPIDX
002B16	HUB port field register 1	DPXREG1									
002C16	HUB port field register 2	DPXREG2									
002D16	HUB port field register 3	DPXREG3									
(1) HUB port 1											
002B16	DP1 interrupt source register	DP1REQ				PTCHG1	PTRSM1	PTERR1	PTCON1	PTDIS1	
002C16	DP1 control register	DP1CON	DSLSPD1	DSRMOD1	DSRSMO1	DSRSTO1	DSDETE1	DSSUSP1	DSPTEN1	DSCONN1	
002D16	DP1 status register	DP1STS							D1PLUS	D1MINUS	
(2) HUB port 2											
002B16	DP2 interrupt source register	DP2REQ				PTCHG2	PTRSM2	PTERR2	PTCON2	PTDIS2	
002C16	DP2 control register	DP2CON	DSLSPD2	DSRMOD2	DSRSMO2	DSRSTO2	DSDETE2	DSSUSP2	DSPTEN2	DSCONN2	
002D16	DP2 status register	DP2STS							D2PLUS	D2MINUS	
0FF916	Downstream port control register	DPCTL						PCON2[1:0]		PCON1[1:0]	

 : Not used

Fig. 92 HUB related registers

## HUB Related Registers

The HUB related registers are shown below.

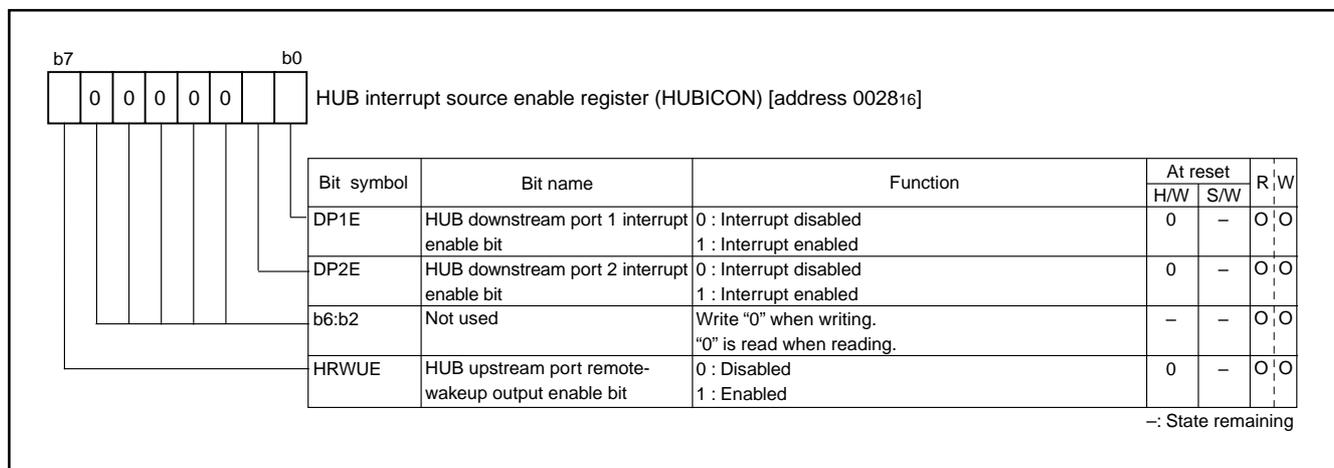


Fig. 93 Structure of HUB interrupt source enable register

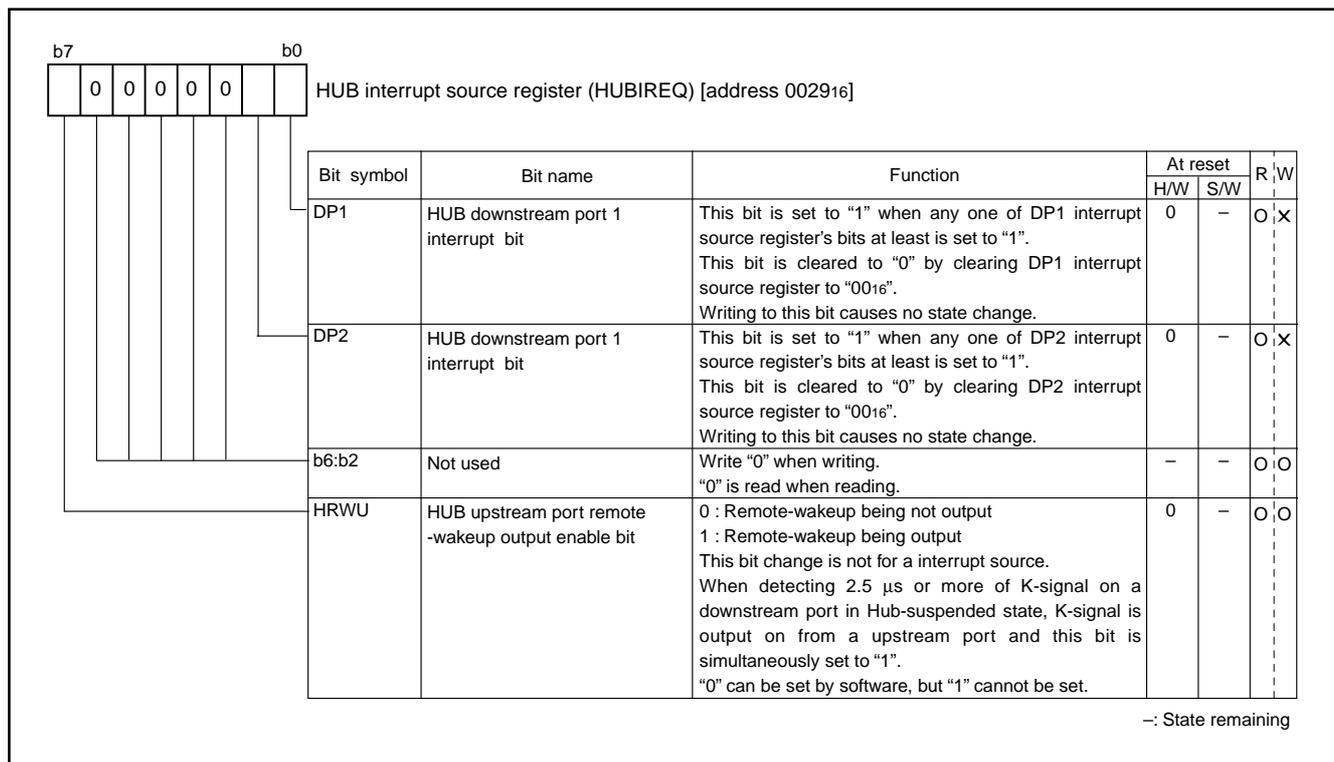


Fig. 94 Structure of HUB interrupt source register

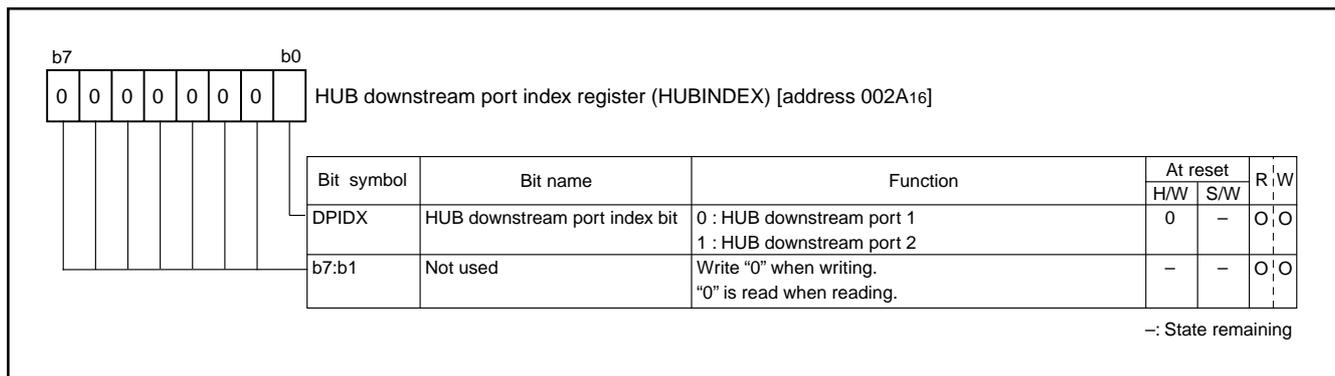


Fig. 95 Structure of HUB downstream port index register

## (1) Downstream port 1

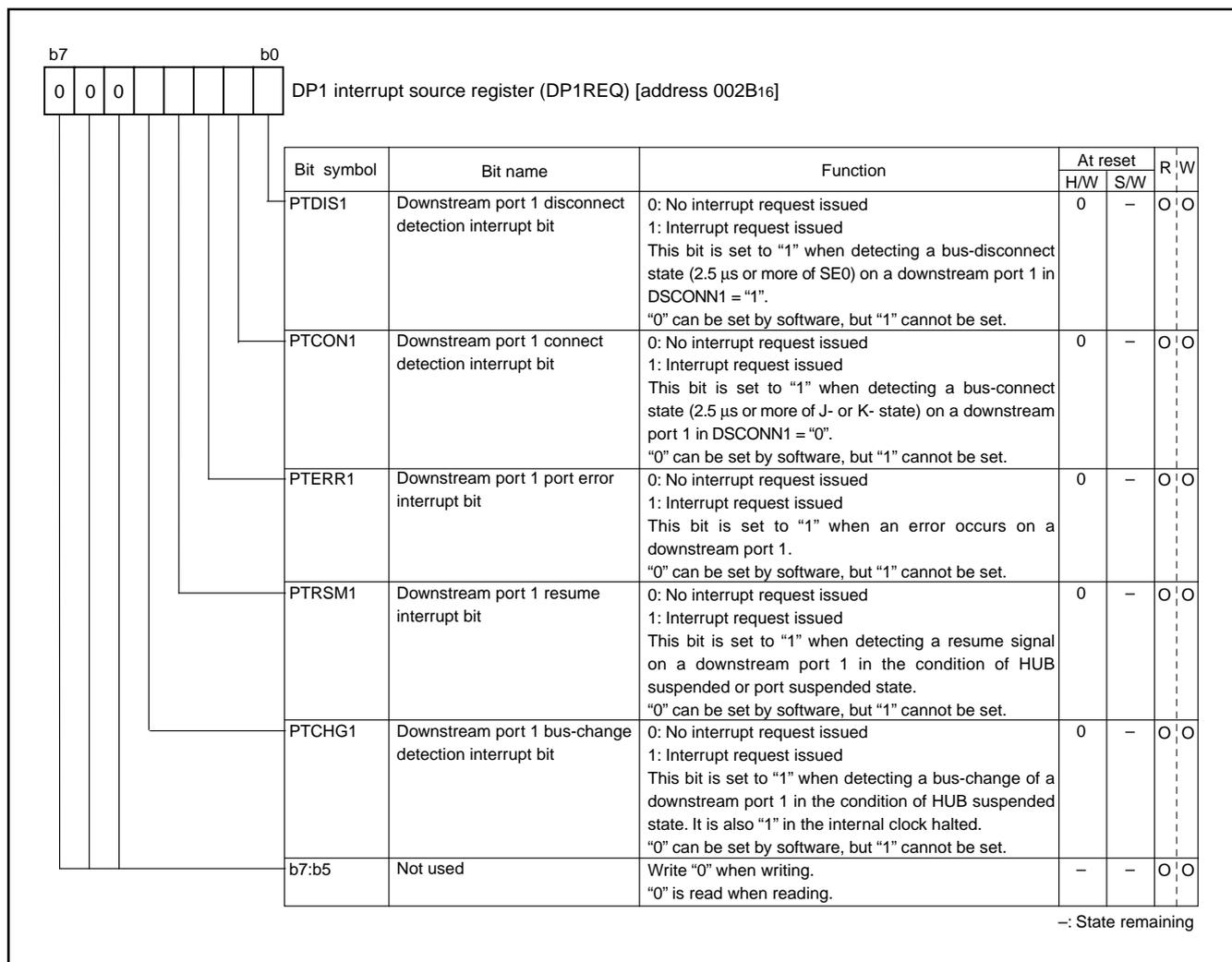


Fig. 96 Structure of DP1 interrupt source register

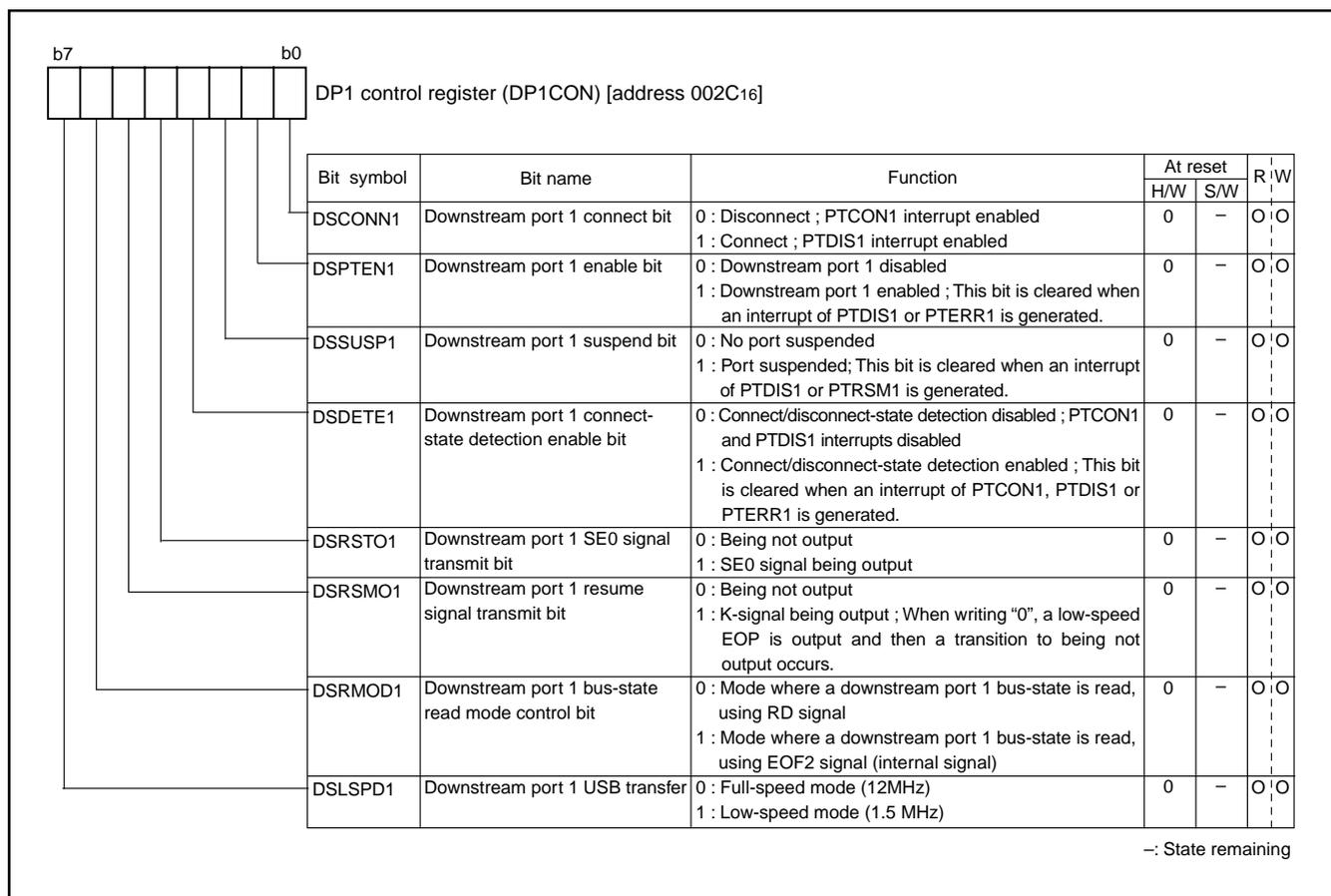


Fig. 97 Structure of DP1 control register

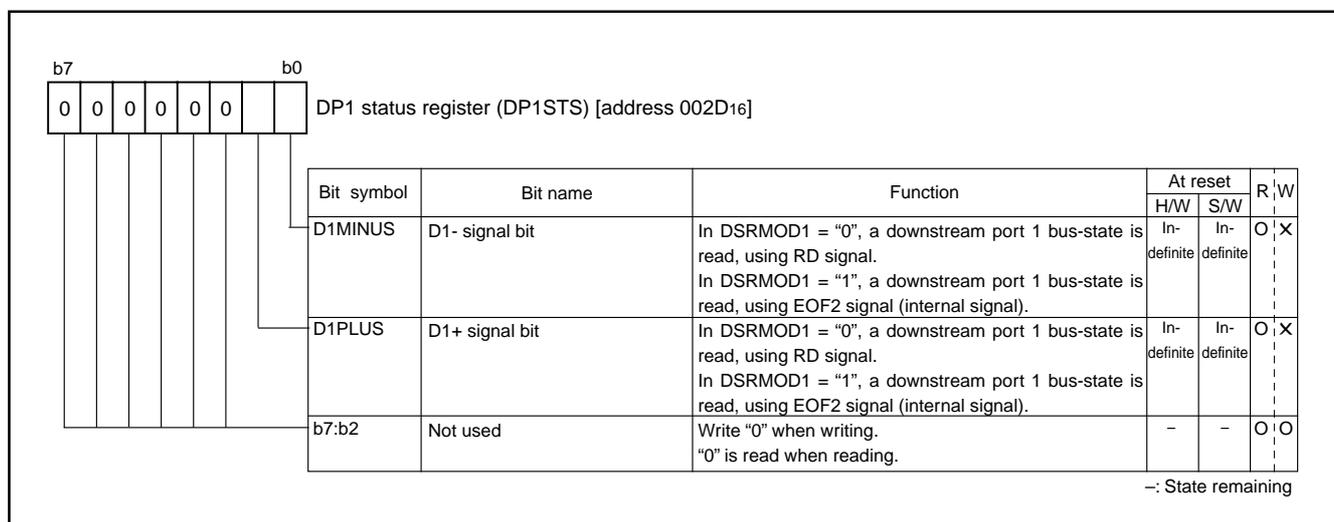


Fig. 98 Structure of DP1 status register

## (2) Downstream port 2

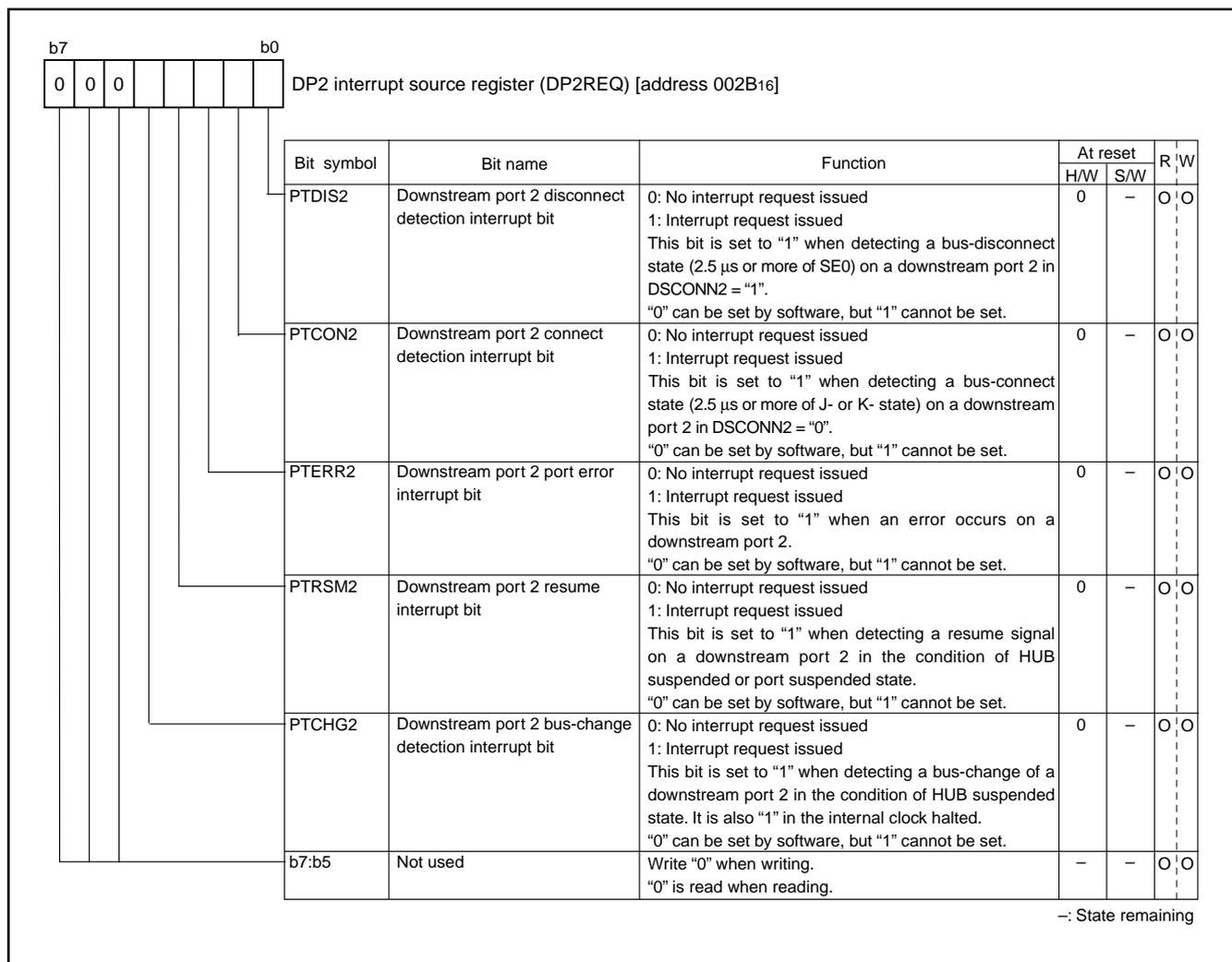


Fig. 99 Structure of DP2 interrupt source register

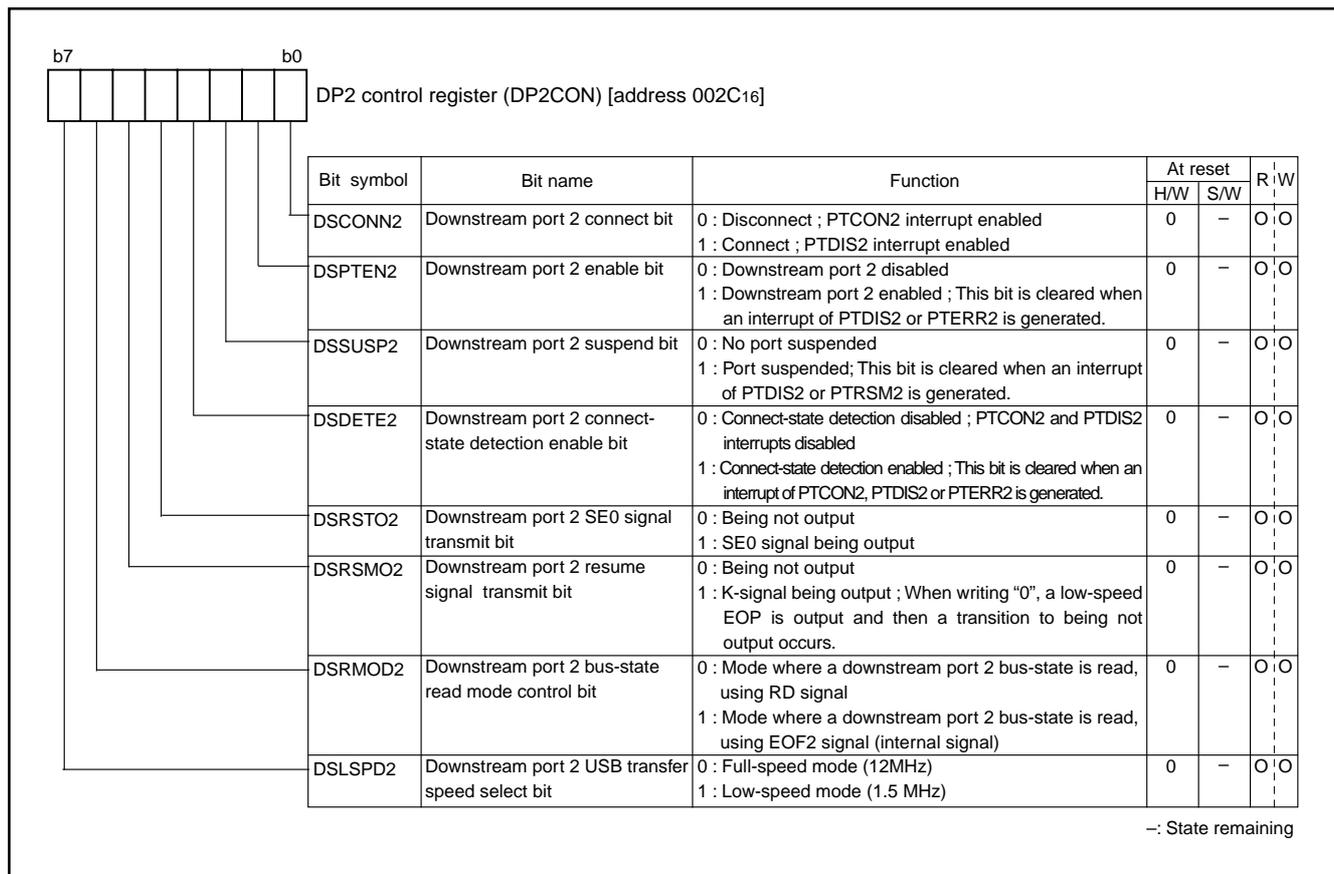


Fig. 100 Structure of DP2 control register

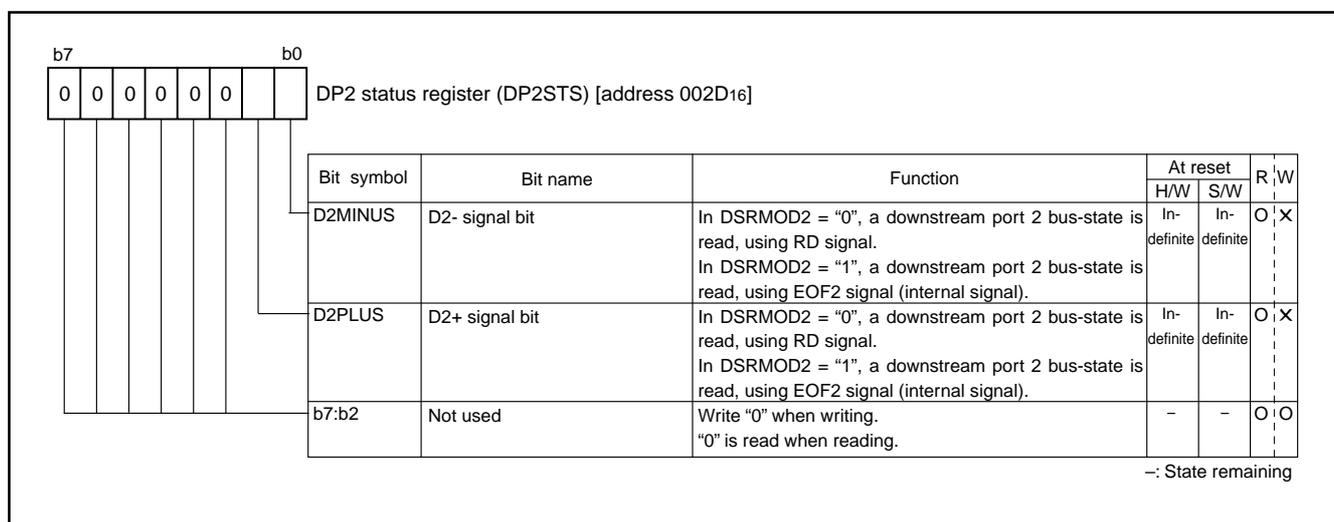


Fig. 101 Structure of DP2 status register

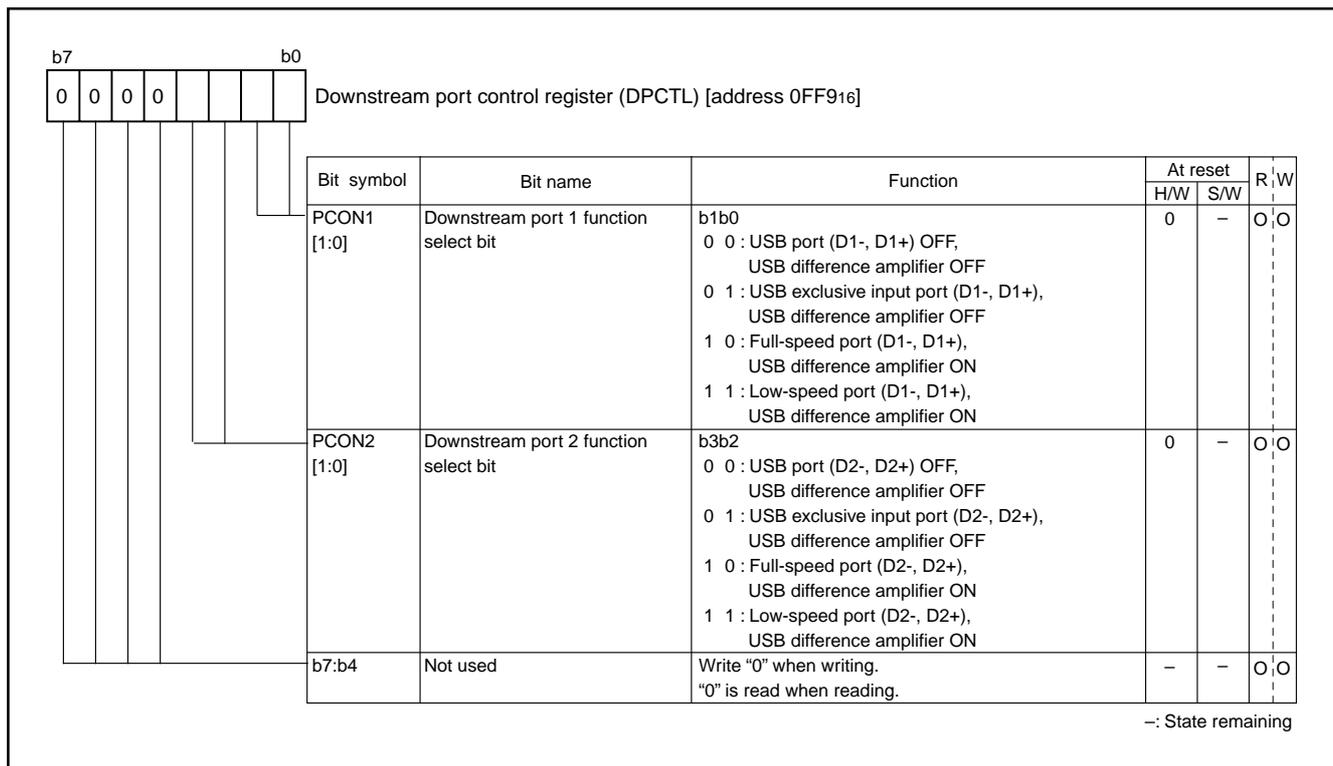


Fig. 102 Structure of Downstream port control register

## EXTERNAL BUS INTERFACE (EXB)

The external bus interface (EXB) controls the data transfer between the external MCU and the 38K2 group's CPU or its

memory (multichannel RAM). The external bus interface is shown below.

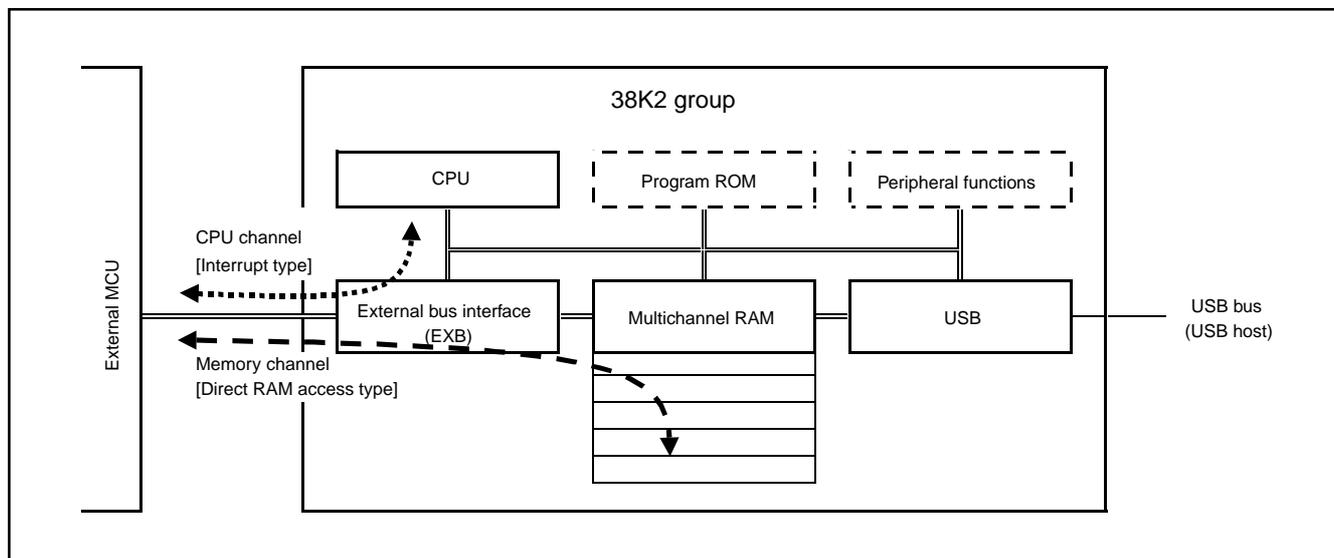


Fig. 103 External bus interface

### ●CPU channel

It is a data transfer course by the interrupt processing between the external MCU and the 38K2 group's CPU.

### ●Memory channel

It is a data transfer course by direct RAM access of the memory channel controller between the external MCU and the 38K2 group's memory (multichannel RAM)

### ●Data transfer of memory channel

When the burst mode is selected with the burst bit of the memory channel operation mode register, data transfer can be carried out at the highest speed. After the external bus interface detects a rise of external read signal/write signal and synchronizes it with the internal clock  $\phi$ , it completes the data transfer between the transmit/receive buffer and the multichannel RAM in two clocks.

However, the waiting time of two clocks at a maximum is generated to access the multichannel RAM in USB being operating because the USB has priority to access.

Therefore, it is necessary to set up the access interval which fills the following timing with the external MCU bus side.

In  $\phi = 8$  MHz, data transfer at about 2 Mbytes/second is possible at a maximum. When there is access simultaneously from the USB, it is about 1.3 Mbytes/second.

In  $\phi = 6$  MHz, data transfer at about 1.5 Mbytes/second is possible at a maximum. When there is access simultaneously from the USB, it is about 1 Mbytes/second.

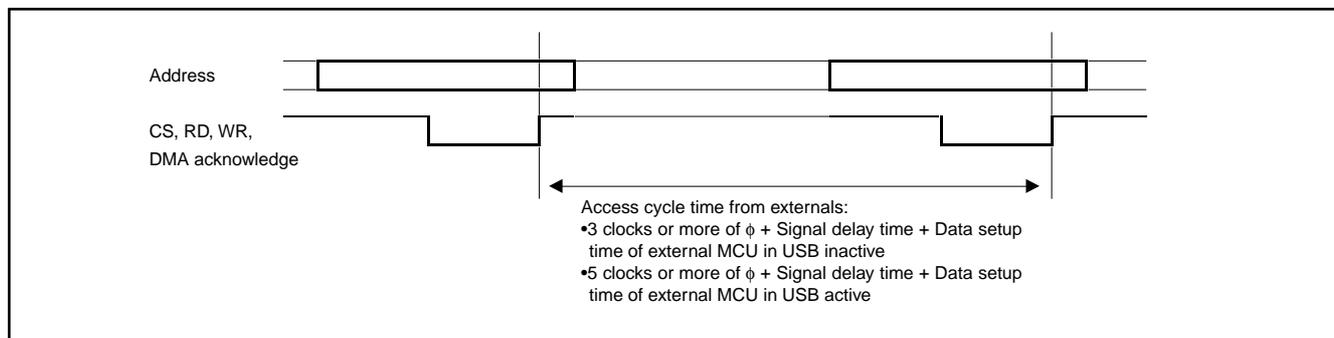


Fig. 104 Data transfer timing of memory channel

## EXB Pin Assignment

The external bus interface (EXB) pins are shown below.

The 38K2 group can transmit/receive a data to/from an external MCU, using the following signals:

- Control input signal ..... 4 (ExCS, ExA0, ExRD, ExWR)
- Data input/output pin ..... 8 (DQ<sub>0</sub> to DQ<sub>7</sub>)
- Interrupt output signal ..... 1 (ExINT)

Additionally, the DMA interface signal and the buffer status read select signal of 38K2 group can be set up per one by the program.

- Control input signal ..... 3 (ExTC, ExDACK, ExRD, ExA1)
- Interrupt output signal ..... 1 (ExDREQ)

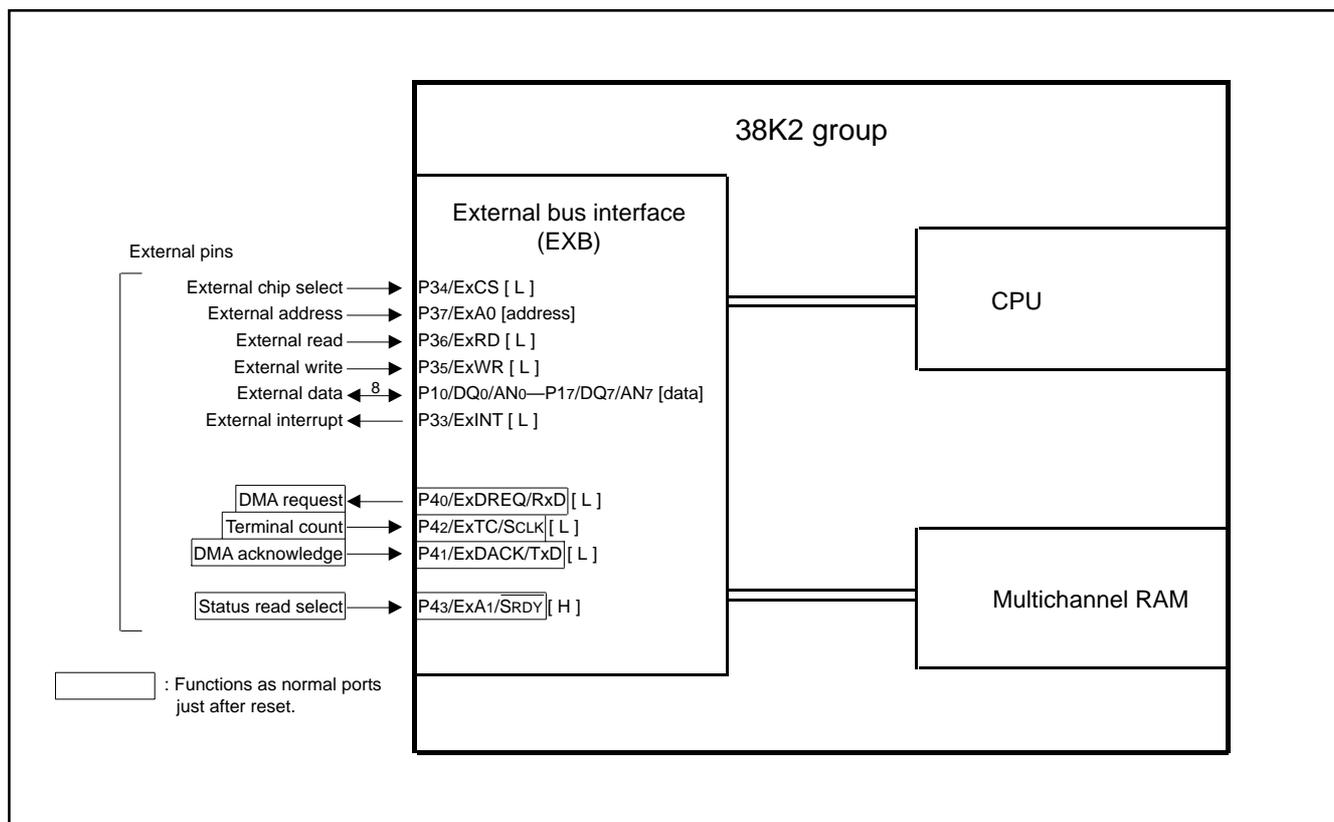


Fig. 105 External bus interface (EXB) pin assignment

### EXB Block Diagram

The block diagram of external bus interface (EXB) is shown below.

The external bus interface (EXB) consists of:

- (1) External I/O interface part
- (2) CPU interface part
- (3) Internal memory interface part
- (4) Transmit/Receive data buffer part

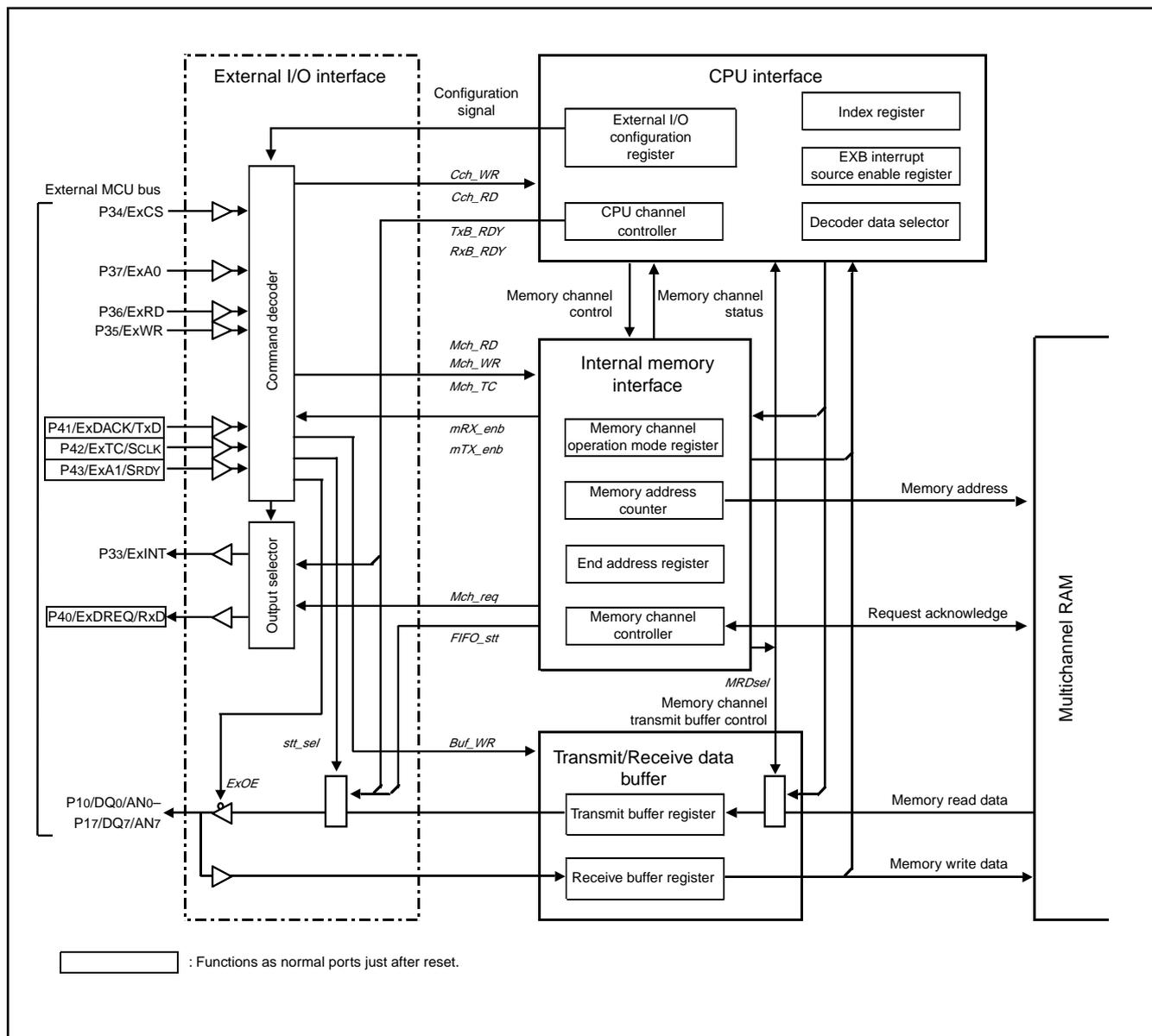


Fig. 106 Block diagram of external bus interface (EXB)

### (1) External I/O Interface Part

The external I/O interface part consists of a command decoder and an output selector. A command decoder generates the following signals to each unit.

#### ●CPU interface part

- CPU channel read (Cch\_RD)
- CPU channel write (Cch\_WR)

#### ●Internal memory interface part

- Memory channel read (Mch\_RD)
- Memory channel write (Mch\_WR)
- Memory channel terminal count (Mch\_TC)

#### ●Transmit/receive data buffer part

- Buffer write (Buf\_WR)

#### ●External I/O interface part

- Status selection (stt\_sel)
- Output enable (ExOE)

Access to the CPU channel can be controlled only by setup of external signals.

Access to the memory channel can be controlled by the value of the external I/O configuration register and the state (mRX\_enb, mTX\_enb signals) of the internal memory interface part.

The output selector has the function which selects from the state of CPU channel (TxB\_RDY and RxD\_RDY) and the state of memory channel (Mch\_req) as the signal assigned to P33/ExINT pin and P40/ExDREQ/RxD pin.

### (2) CPU Interface Part

The CPU interface part consists of the decoder/data selector of the CPU channel, the CPU write register and CPU channel controller

#### ●Decoder/data selector of CPU channel

A write operation to the CPU register is performed by generating a write signal for each register with an address decode signal and a write signal.

A read operation from the CPU register is performed by generating an output enable signal of the internal data bus with an module select signal and a read signal and generating a select signal for each register with an address decode signal.

#### ●CPU write register

There are three CPU write registers as follows:

- EXB interrupt source enable register
- Index register
- External I/O configuration register

The EXB interrupt source register is a read-only register.

A status signal of the CPU channel controller and a status signal of the memory channel controller in the internal memory interface part are generated.

#### ●CPU channel controller

The CPU channel controller generates the following signals, using bits 0 and 1 (RXB\_ENB, TXB\_ENB) of EXB interrupt source enable register.

- Memory channel transmitting buffer control signal (MRD\_sel), generated in the internal memory interface part
- CPU channel command signal (Cch\_RD, Cch\_WR), generated in the external I/O interface part
- Signals RxB\_RDY/RxB\_full and TxB\_RDY/TxB\_empty, generated with read/write signals from the CPU channel

### (3) Internal Memory Interface Part

The internal memory interface part consists of the CPU register and the memory channel controller.

#### ●CPU register

The CPU register consists of the follows:

- Memory channel operation mode register
- Memory address counter
- End address register

The CPU can set the beginning address into the memory address counter when the memory channel operation enable bit (MC\_ENB) of EXB interrupt source enable register is "0". When this bit is "1", the write operation from the CPU is invalid and each access from the external bus causes count-up operation.

#### ●Memory channel controller

The CPU register consists of the follows:

- Main sequencer
- Internal memory request signal generating circuit
- External memory channel request signal generating circuit
- Address end detection circuit
- Terminal end input processing circuit

### (4) Transmit/Receive Data Buffer Part

The transmit/receive data buffer part consists of the 8-bit transmit buffer register (TXBUF) and the 8-bit receive buffer register (RXBUF).

Both CPU channel and memory channel use the same transmit buffer register/receive buffer register to transfer a data to an external MCU bus.

### (5) External Pin

The external bus interface has the following pins to connect with an external MCU bus.

- Chip select ..... P34/ExCS
- Address ..... P37/ExA0
- Data ..... P10/DQ0/AN0 to P17/DQ7/AN7
- Read ..... P36/ExRD
- Write ..... P35/ExWR
- Interrupt request ..... P33/ExINT

It also has the following pins to connect with an external DMAC. Each pin can be programmed for an ordinary port function or a DMA interface pin function.

- DMA request ..... P40/ExDREQ/RxD
- DMA acknowledgment ..... P41/ExDACK/TxD
- Terminal count ..... P42/ExTC/SCLK

It also has the status read select pin (P43/ExA1/ $\overline{\text{SRDY}}$  pin) to confirm a ready status of the data buffer from an external MCU bus. This pin functions as a port just after reset. The status read select function can be set by a program.

- Status read select ..... P43/ExA1/ $\overline{\text{SRDY}}$

#### ●CPU channel: Communication with 38K2 group CPU

When a read/write operation is performed from an external MCU bus in address signal ExA0 = "H", the interrupt is generated and the 38K2 group CPU can confirm its access. The 38K2 group CPU judges the interrupt source and it starts a data transmission/reception with an external MCU bus.

#### ●Memory channel: Communication with 38K2 group memory multichannel RAM

When a read/write operation is performed from an external MCU bus in address signal ExA0 = "L", access to the multichannel RAM is performed. Then an address of the multichannel RAM is made by the external bus interface and it is increased at each access completion. Consequently, FIFO access is performed.

Even if a read/write operation is performed in DACK = "L" instead of ExCS = "L" and ExA0 = "L", FIFO access to the multichannel RAM is performed.

The beginning address and the end address must be set by the CPU in advance.

**●P33/ExINT pin**

Any one of the following signals for this pin can be selected:

- TxB\_RDY (transmit buffer ready) output
- RxB\_RDY (receive buffer ready) output
- Mch\_req (memory channel request) output

Either TxB\_RDY or RxB\_RDY is normally selected. The memory channel request is for an access request signal to the memory channel.

In a small system, a data transfer processing to the internal memory is performed in the interrupt routine. According to that situation, the 38K2 group has the function automatically to switch an interrupt factor attached on the interrupt pin by program.

**●P40/ExDREQ/RxD pin**

This pin is a port at the initial state. Which signal can be set by program.

- RxB\_RDY (receive buffer ready) output
- Mch\_req (memory channel request) output

Mch\_req of DMAC is normally selected. The output method of the memory channel request signal depends on the burst bit (BURST) of memory channel operation mode register. When the burst bit is "0", this signal is periodically output at each 1-byte transfer. (See Figures 124 and 127.)

When the burst bit is "1", this signal is continuously output while the memory address counter is counting from the beginning address to the end address (See Figures 125 and 128.)

**●P41/ExDACK/TxD pin**

This pin is a port at the initial state. The DMA acknowledge signal can be set by program.

The DMA acknowledge signal DACK = "L" is the same state as that of CS = "L" and A0 = "L". Access to multichannel RAM is started by a rise of read signal or write signal which is set during this term.

**Note:** If the DMA acknowledge signal and the chip select signal are simultaneously active (DACK = "L" and CS = "L"), also set the address signal A0 to "L". If A0 is "H", the memory channel and the CPU channel are activated simultaneously and it might cause some error.

**●P42/ExTC/SCLK pin**

This pin is a port at the initial state. The terminal count signal can be set by program.

If the terminal count signal is set at one bus cycle while a memory channel operation write is being performed, the 38K2 group confirms that its bus cycle is the write cycle of the last data and sets the memory channel status bits to "112", and the interrupt is generated and the memory channel operation ends even if the memory address counter has not reached the end address.

The CPU can obtain the last address where the data is written by reading out the value of memory address counter. (See Figure 126.)

## EXB Register List

The EXB register list is shown below.

Address	Register Name	SYMBOL	EXB SFR							
			bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0030 <sub>16</sub>	EXB interrupt source enable register	EXBICON					MC_ENB		TXB_ENB	RXB_ENB
0031 <sub>16</sub>	EXB interrupt source register	EXBIREQ					MC_STS[1:0]		TXB_EMPTY	RXB_FULL
0033 <sub>16</sub>	EXB index register	EXBINDEX	0	0	0	0	0	INDEX[2:0]		
0034 <sub>16</sub>	Register window 1 (low)	EXBREG1	LOW_WIN[7:0]							
0035 <sub>16</sub>	Register window 2 (high)	EXBREG2	HIGH_WIN[7:0]							

: Not used  
0 : "0" fixed

Fig. 107 EXB related registers (1)

### •EXB interrupt source enable register

This register enables/disables access from an external bus and an internal interrupt.

### •EXB interrupt source register

This register indicates the state of CPU channel's transmit/receive buffer register and the memory channel. The same value can be read out from the external MCU bus by using the buffer status read select signal (A1 pin = "H").

### •EXB index register/Register windows 1, 2

The accessible register is switched by treating addresses 0034<sub>16</sub> and 0035<sub>16</sub> as a register window depending on the value of EXB index register at address 0033<sub>16</sub>.

Index	low high	Register Name	SYMBOL	EXB SFR							
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0016	low	External I/O configuration register	EXBCFGL				A1_CTR	INT_CTR[2:0]		EXB_CTR	
	high		EXBCFGH				TC_CTR	DAK_CTR[1:0]	DRQ_CTR[1:0]		
0116	low	Transmit/Receive buffer register	RXBUFF/TXBUF	At CPU read : RXBUF[7:0] At CPU write : TXBUF[7:0]							
	high		—								
0216	low	Memory channel operation mode register	MCHMOD					BURST	MC_DIR[1:0]		
	high		—								
0316	low	Memory address counter	MEMADL	IM_A[7:0]							
	high		MEMADH	0	0	0	0	0	IM_A[10:8]		
0416	low	End address register	ENDADL	END_A[7:0]							
	high		ENDADH	0	0	0	0	0	END_A[10:8]		

: Not used  
0 : "0" fixed

Fig. 108 EXB related registers (2)

### •External I/O configuration register

This register selects the function of each pin.

### •Transmit/Receive buffer register

This register consists of the receive buffer register (RXBUF) and the transmit buffer register (TXBUF)

### •Memory channel operation mode register

This register sets the operation mode of the memory channel.

### •Memory address counter

This is a counter to set the beginning address which FIFO accesses. This register is increased by access from the external MCU bus.

### •End address register

This register is to set the end address which FIFO accesses.

## EXB Related Registers

The EXB related registers are shown below.

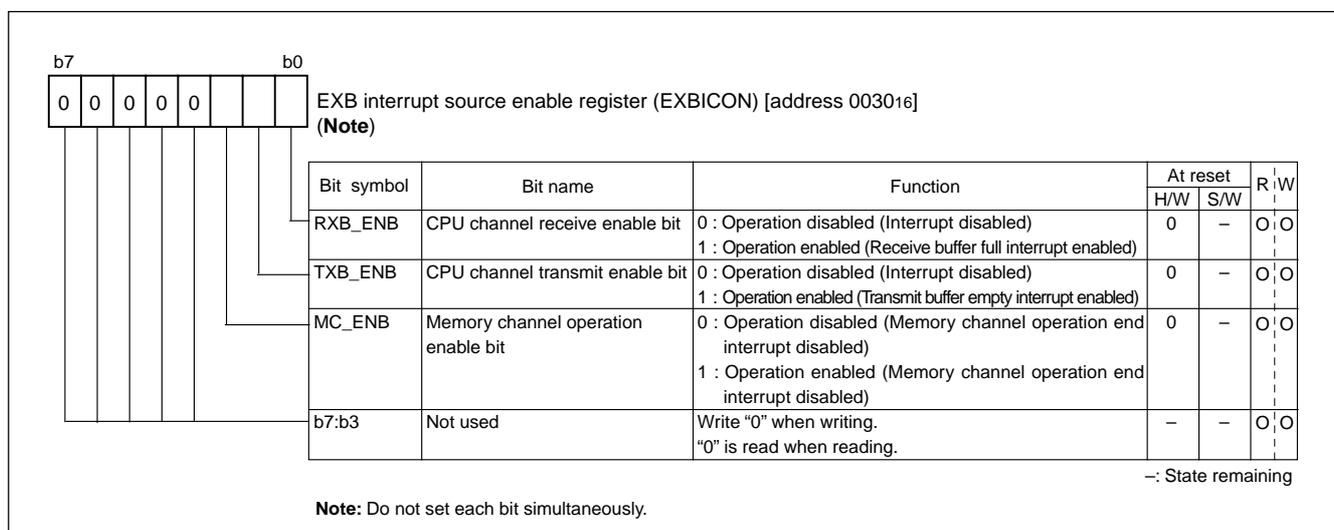


Fig. 109 Structure of EXB interrupt source enable register

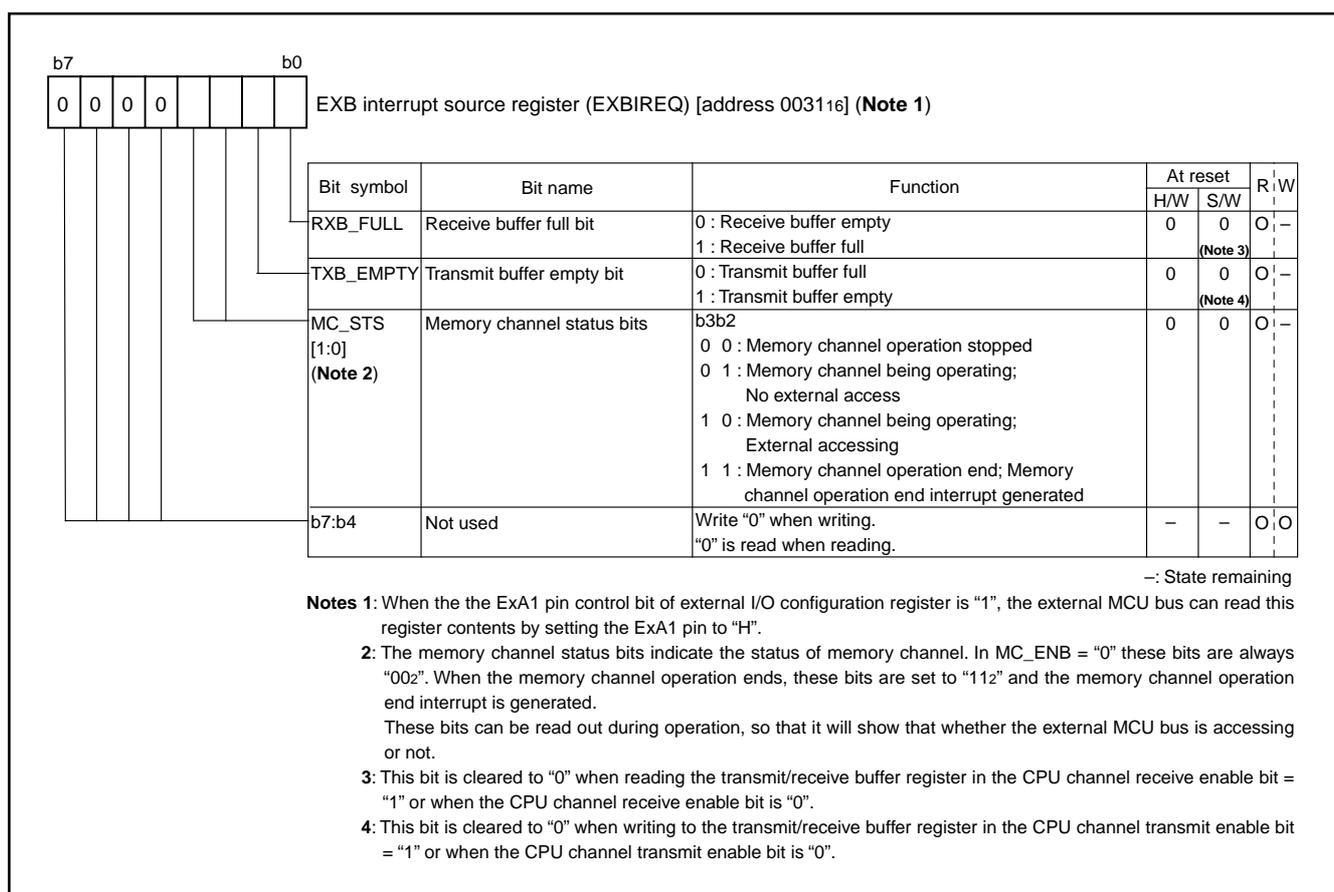


Fig. 110 Structure of EXB interrupt source register

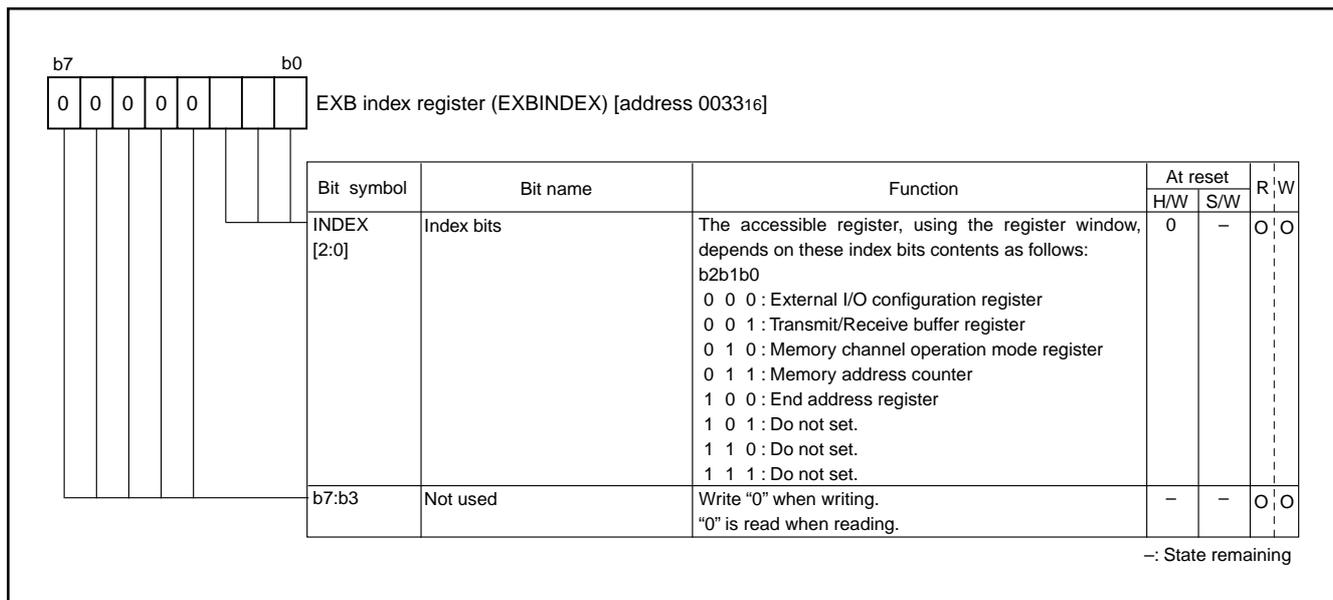


Fig. 111 Structure of EXB index register

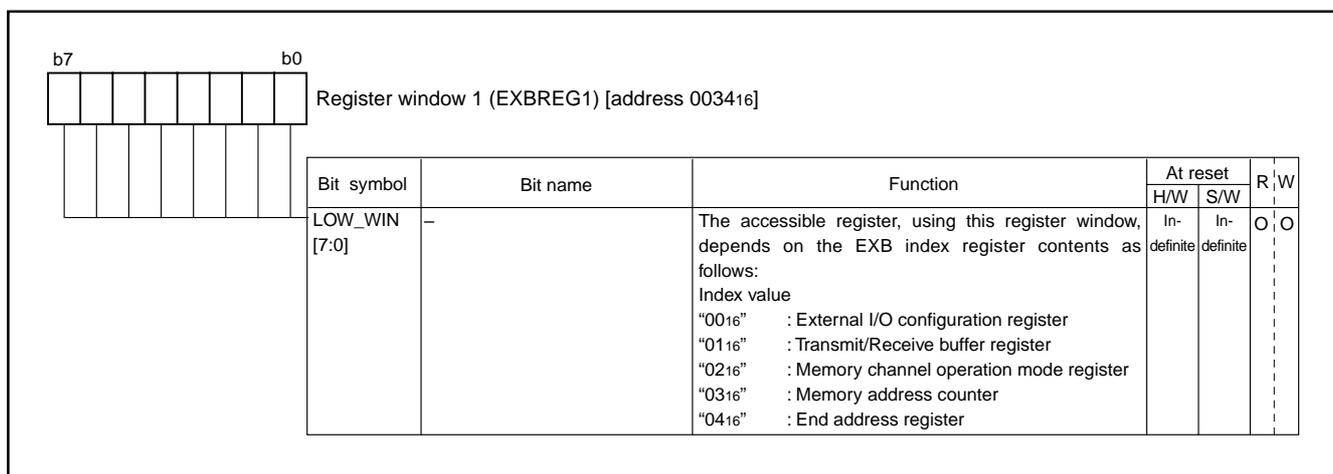


Fig. 112 Structure of Register window 1

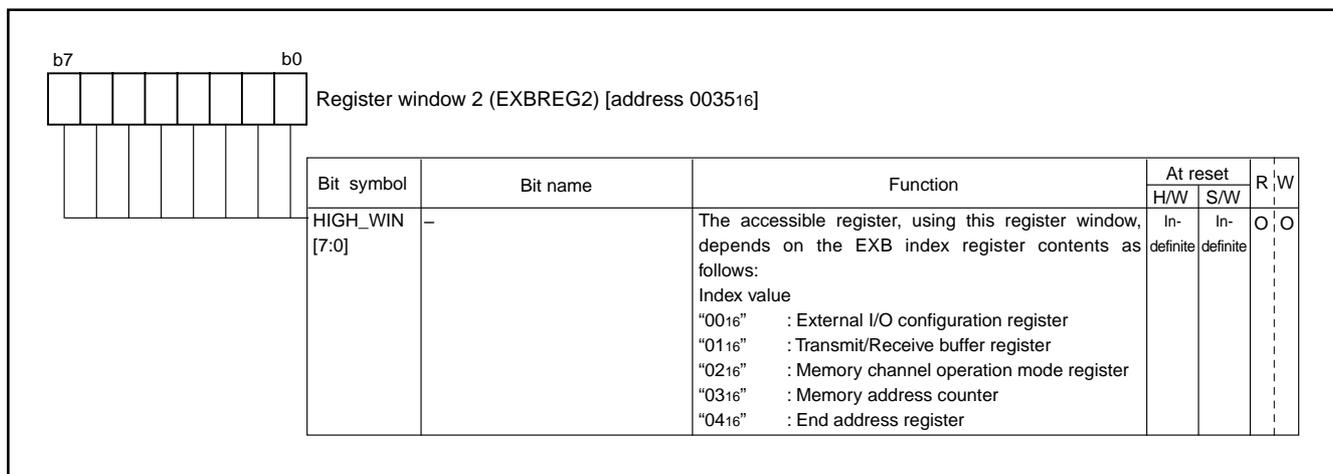


Fig. 113 Structure of Register window 2

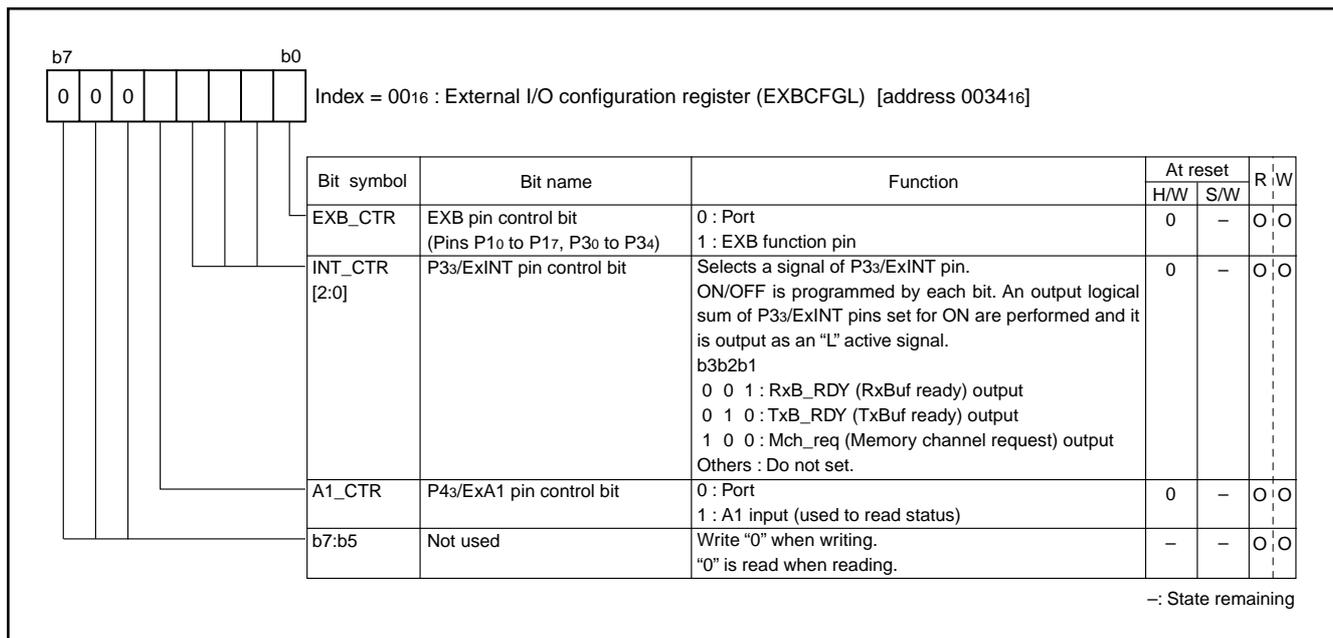


Fig. 114 Index00[low]; Structure of External I/O configuration register

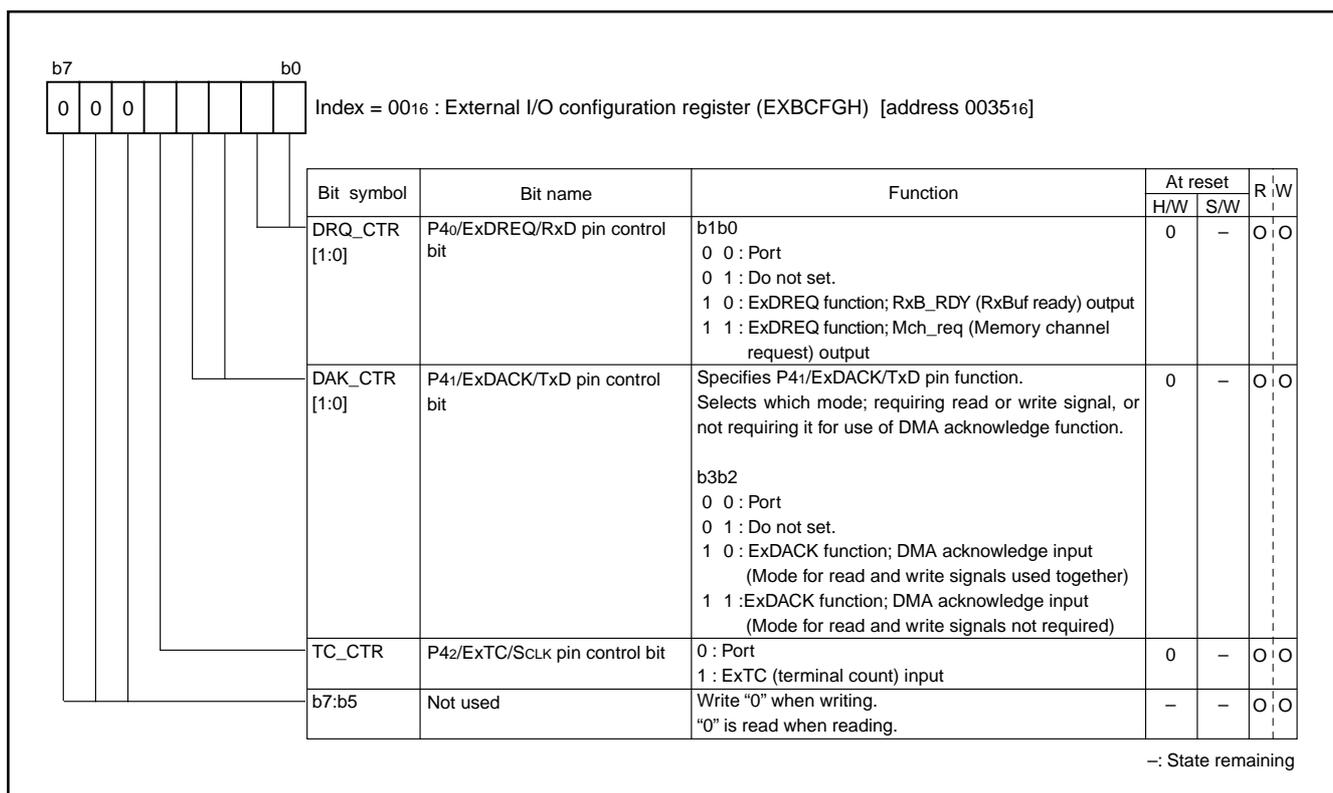


Fig. 115 Index00[high]; Structure of External I/O configuration register

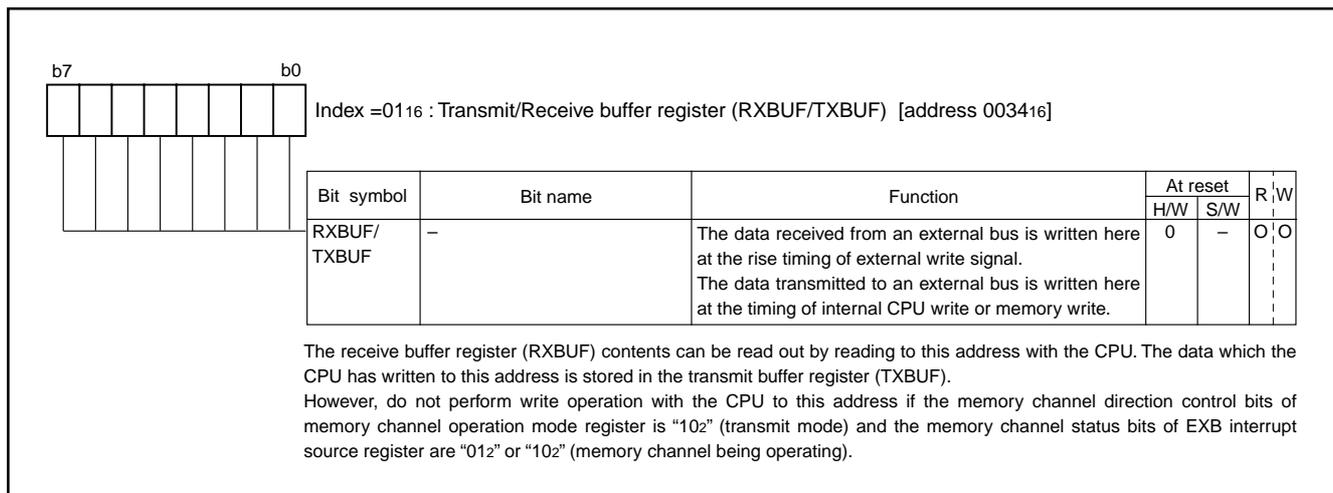


Fig. 116 Index01[low]; Structure of Transmit/Receive buffer register

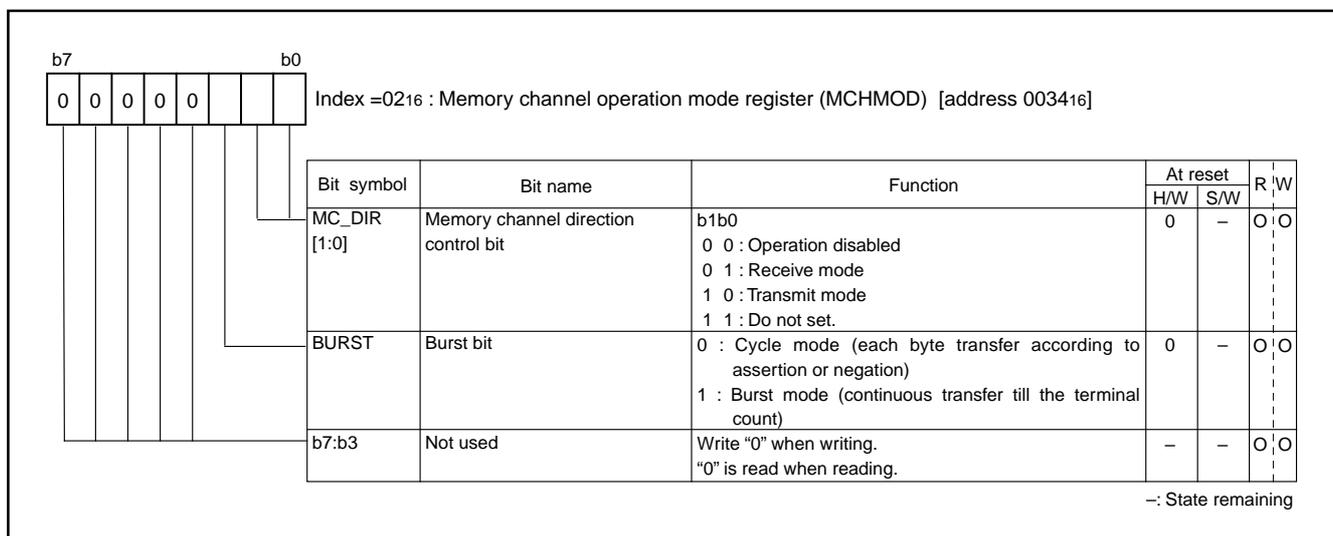


Fig. 117 Index02[low]; Structure of Memory channel operation mode register

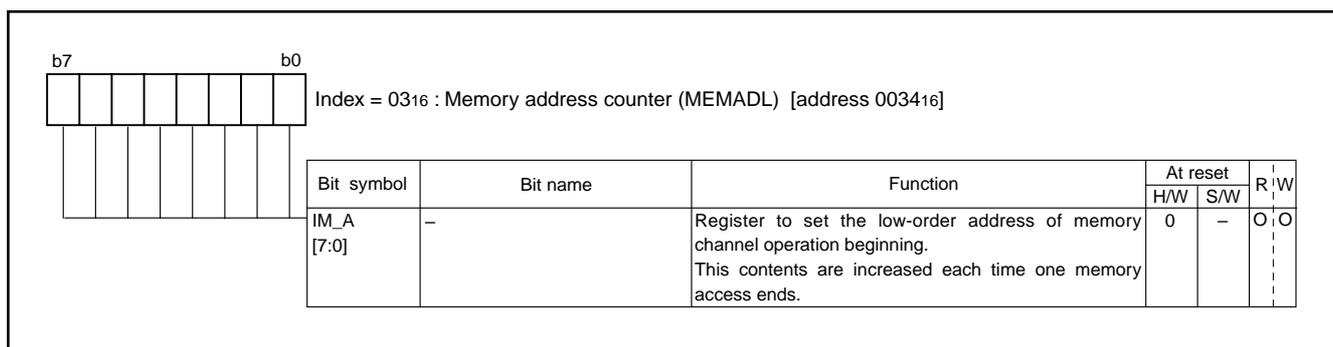


Fig. 118 Index03[low]; Structure of Memory address counter

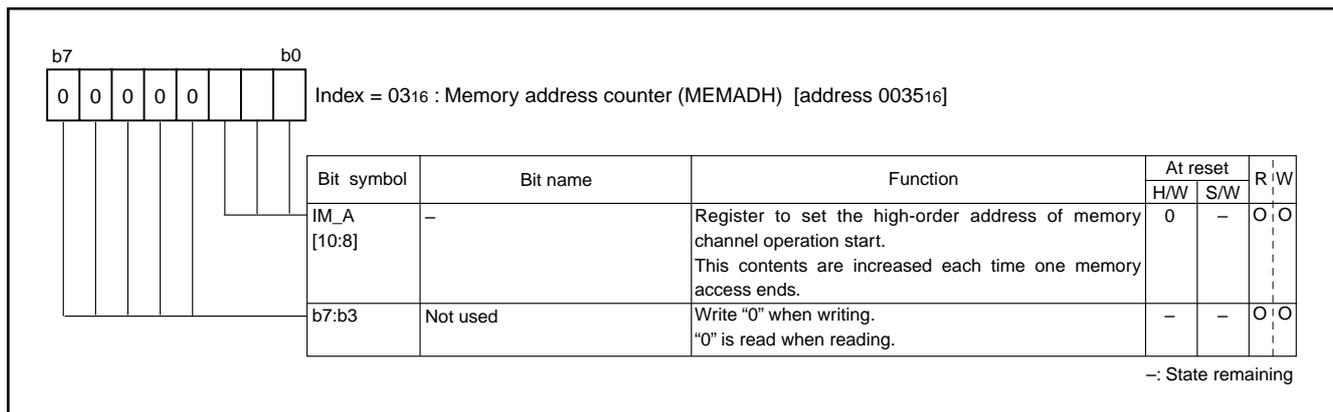


Fig. 119 Index03[high]; Structure of Memory address counter

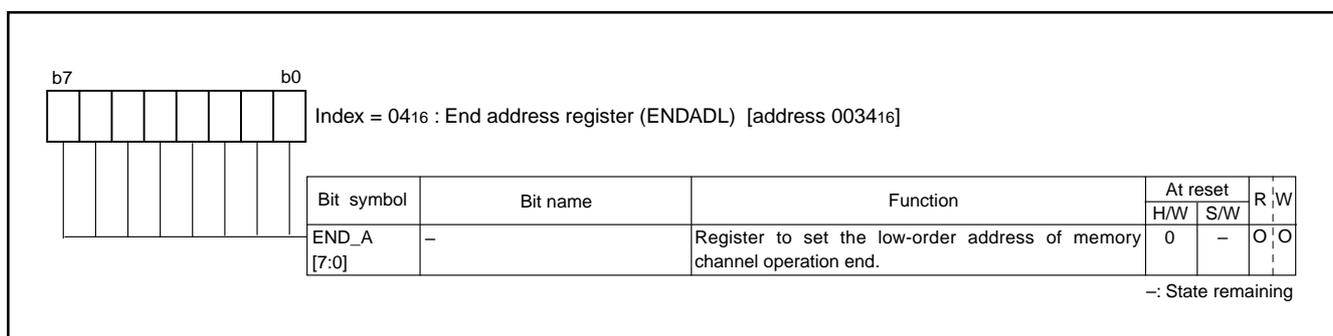


Fig. 120 Index04[low]; Structure of End address register

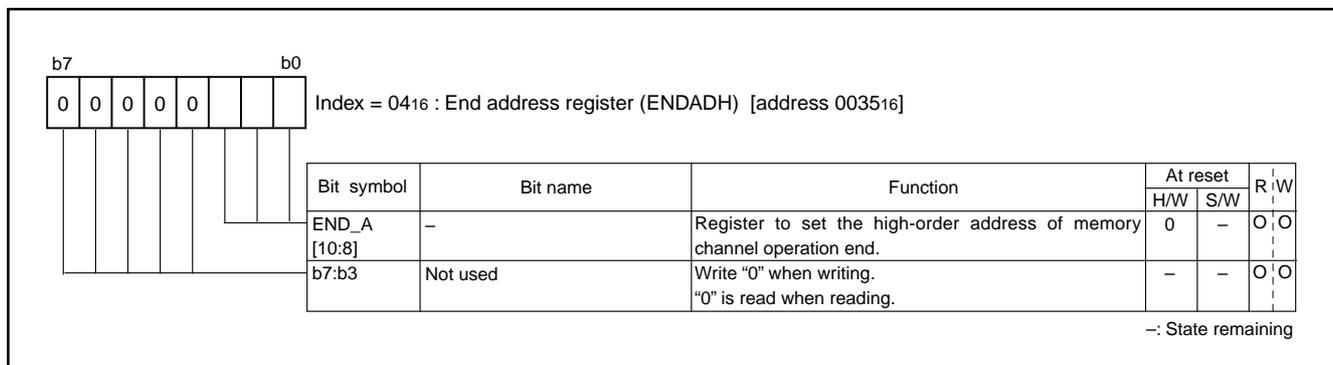


Fig. 121 Index04[high]; Structure of End address register

## EXB Operation Timing Diagram

### (1) CPU Channel Receiving Operation

CPU channel receiving operation is shown below.

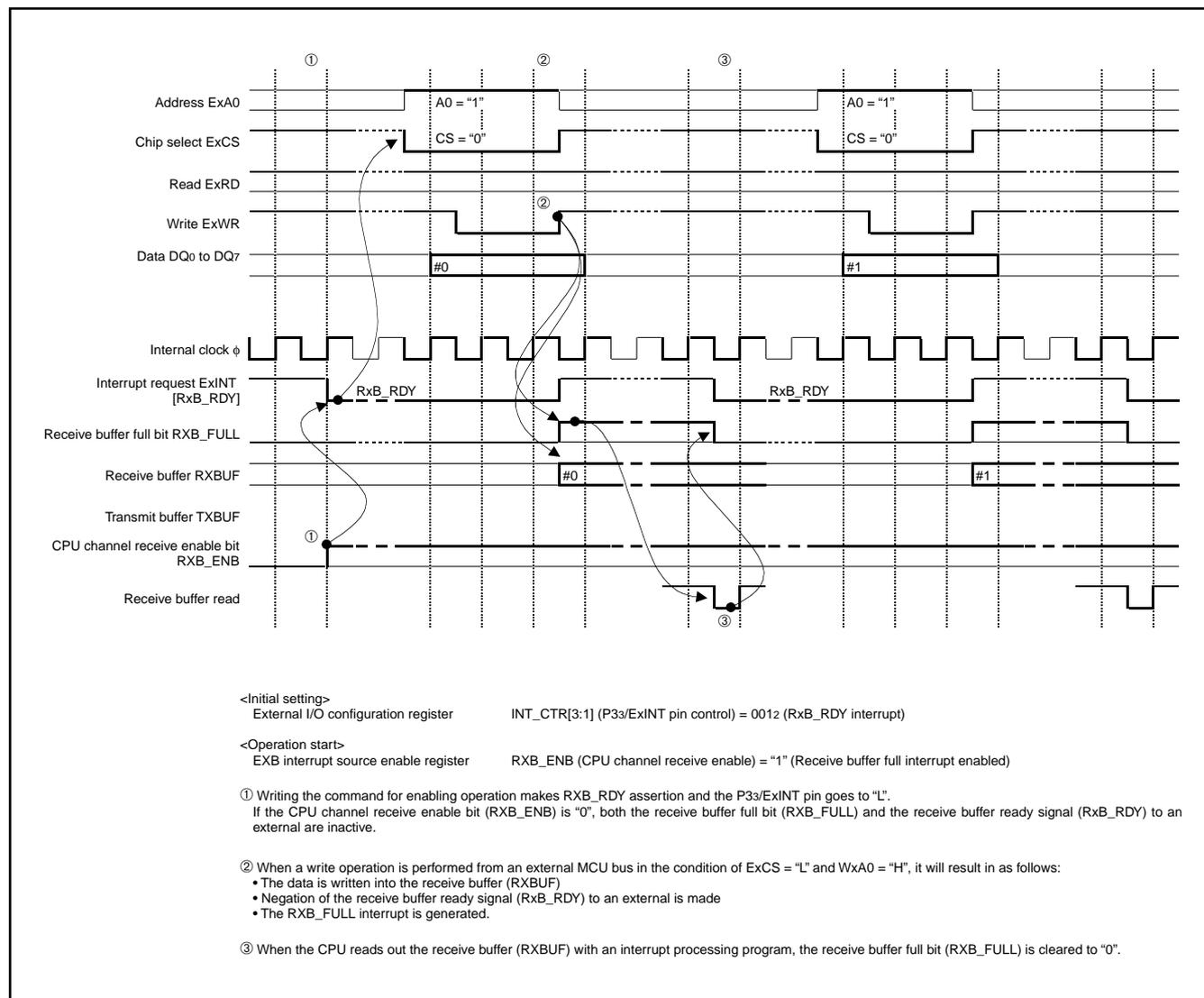


Fig. 122 CPU channel receiving operation

## (2) CPU Channel Transmitting Operation

CPU channel transmitting operation is shown bellow.

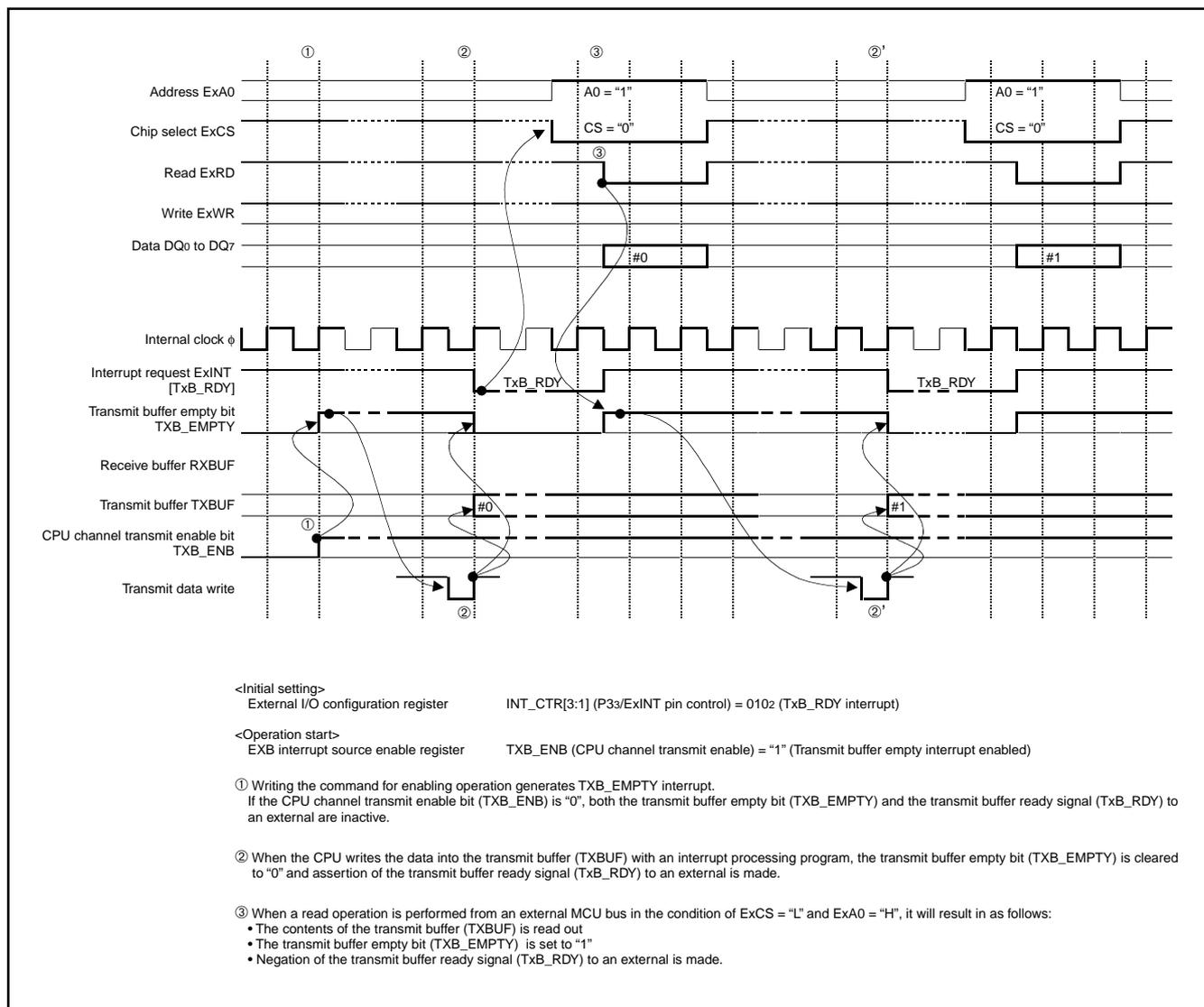


Fig. 123 CPU channel transmitting operation

### (3) Memory Channel Receiving Operation (1)- Cycle Mode

Memory channel receiving operation (1) is shown below.

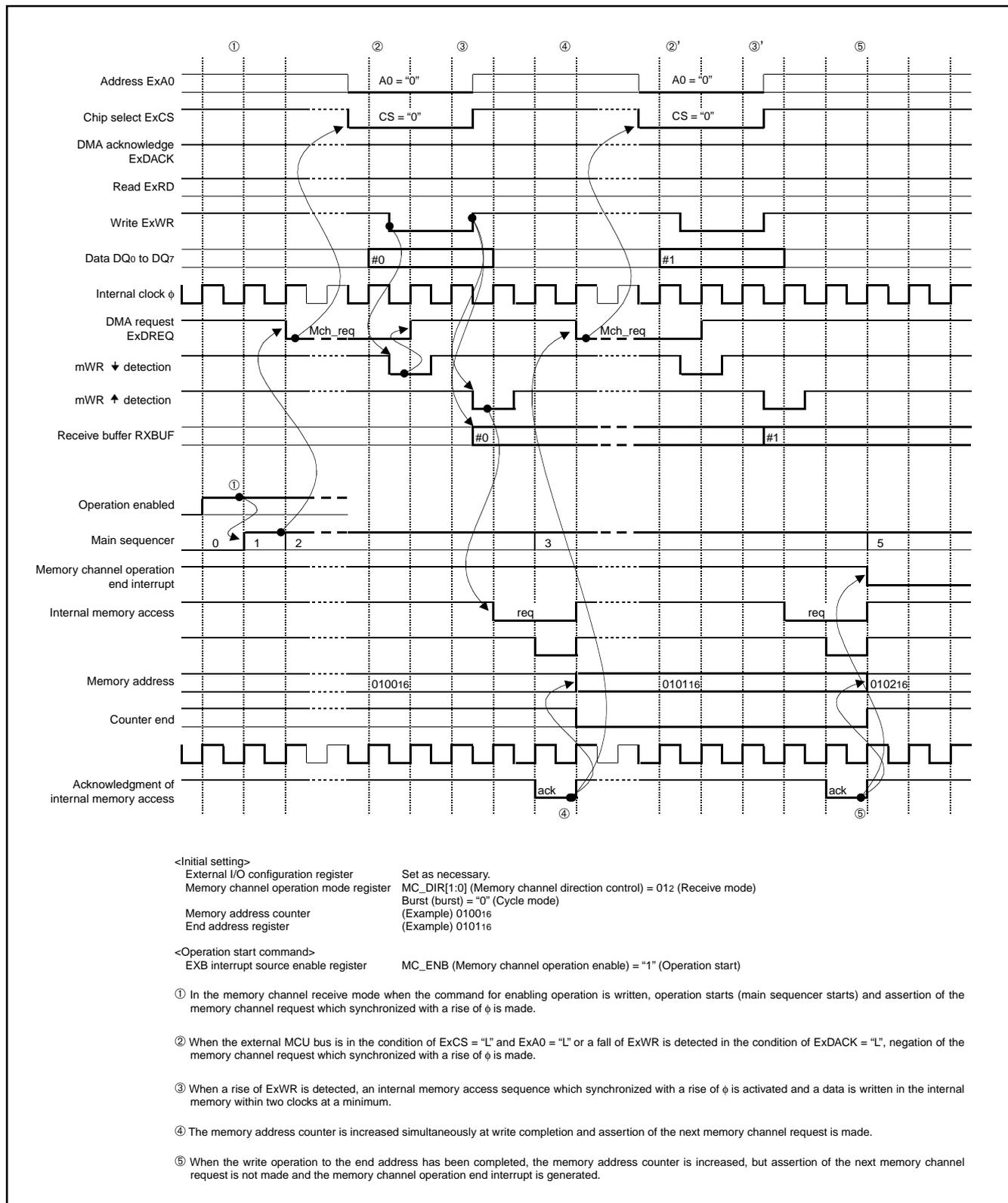


Fig. 124 Memory channel receiving operation (1)









### MULTICHANNEL RAM

The 38K2 group has the built-in multichannel RAM including the small logic circuit (RAM I/F) instead of ordinary RAM.

The multichannel RAM has the USB channel and the EXB channel in addition to the CPU channel.

The multichannel RAM controls access from CPU, USB and EXB, synchronizing control with  $\phi$ . The USB transfer rate is about 1.5 Mbytes/second. Access to the multichannel RAM is performed at every about 5.3 clocks in  $\phi = 8$  MHz, or at every about 4 clocks in  $\phi = 6$  MHz. The USB's access has priority to the EXB's.

The one wait function ( $\overline{ONW}$  function) of 38000 series CPU is used internally to control access with the CPU. When receiving an access request from the USB or the EXB, the multichannel RAM outputs  $\overline{ONW}$  signal to wait the CPU for one clock, and access of the USB or the EXB is performed.

If the multichannel RAM is outputting  $\overline{ONW}$  signal while the CPU is in the state of reading/writing for the RAM area, the CPU read cycle or write cycle is extended by 1 period of  $\phi$ .

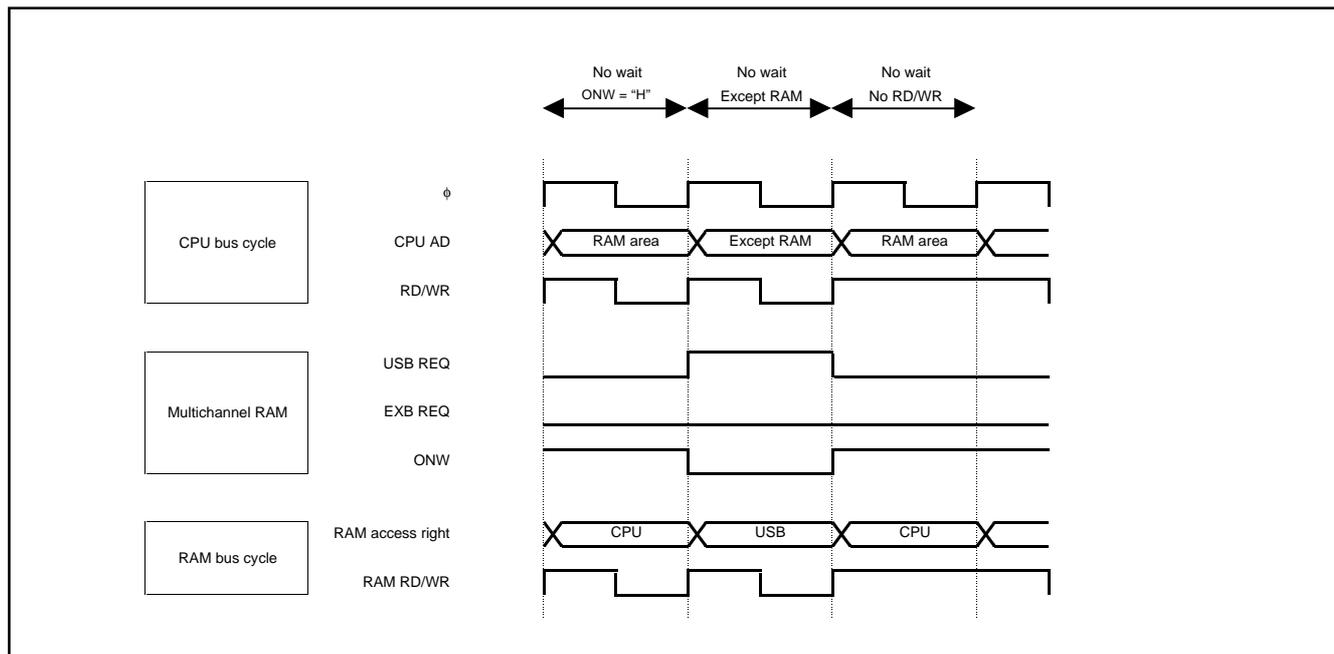


Fig. 129 Multichannel RAM timing diagram (no wait)

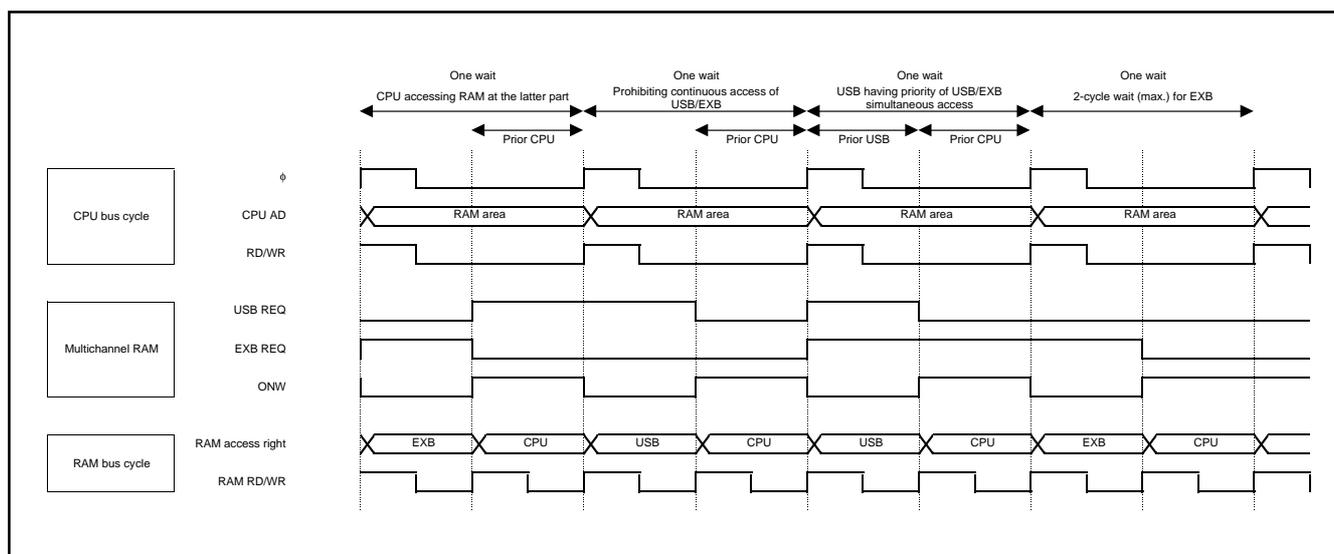


Fig. 130 Multichannel RAM timing diagram (one wait)

### Multichannel RAM Operation Example

The multichannel RAM operation example is shown below.  
This example shows the case that an external MCU uses the 38K2 group as a peripheral LSI (USB controller).

The following explains that the external MCU reads out the data which is received via the USB.

- ① The data which is received via the USB is written into the multichannel RAM.
- ② Receive completion is propagated to the CPU.
- ③ The external bus interface is activated owing to the CPU.
- ④ (1) The external bus interface sets the data which is read from the multichannel RAM into the internal data buffer.  
(2) The external MCU reads out the data bus buffer of the external bus interface.  
(3) The above operation is repeated by the number of the received bytes. After that, the data transfer is completed.

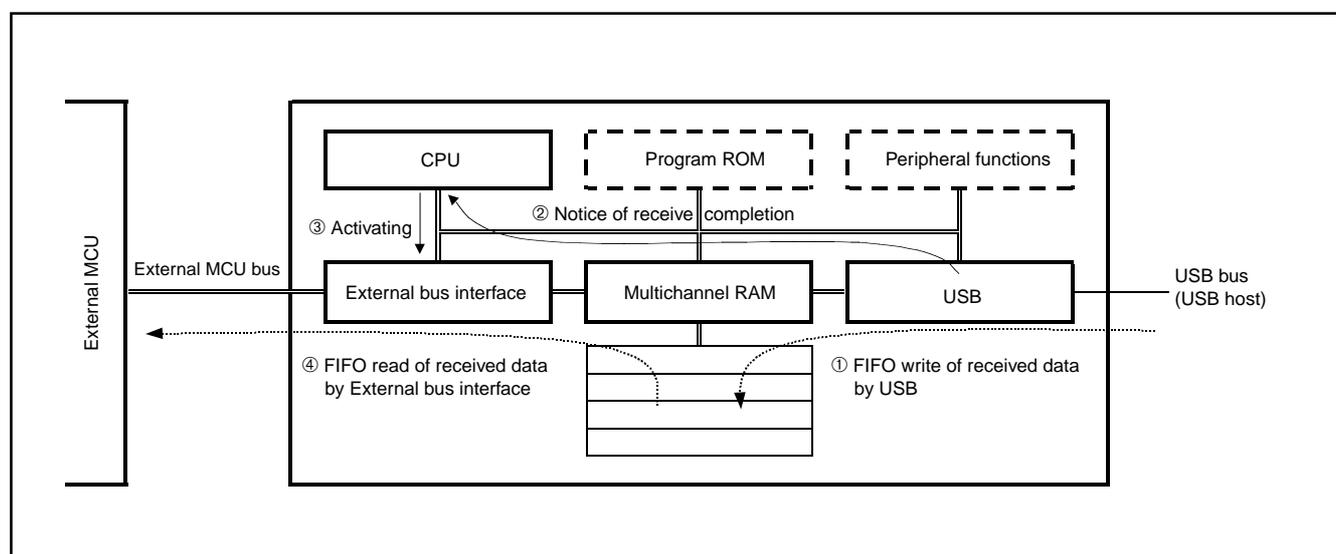


Fig. 131 Multichannel RAM operation example

## A/D CONVERTER

The functional blocks of the A/D converter are described below.

### [AD Conversion Register 1, 2 (AD1, AD2)] 003716, 003816

The AD conversion register is a read-only register that stores the result of an A/D conversion. When reading this register during an A/D conversion, the previous conversion result is read.

Bit 7 of the AD conversion register 2 must be set to "0". Not only 10-bit reading but also only high-order 8-bit reading of conversion result can be performed by selecting the reading procedure of the AD conversion registers 1, 2 after A/D conversion is completed (in Figure 133).

The 8-bit reading inclined to MSB is performed when reading the AD converter register 1 after A/D conversion is started or reset; and when the AD converter register 1 is read after reading the AD converter register 2, the 8-bit reading inclined to LSB is performed.

### [AD Control Register (ADCON)] 003616

The AD control register controls the A/D conversion process. Bits 0 to 2 select a specific analog input pin. Bit 3 signals the completion of an A/D conversion. The value of this bit remains at "0" during an A/D conversion, and changes to "1" when an A/D conversion ends. Writing "0" to this bit starts the A/D conversion.

## Comparison Voltage Generator

The comparison voltage generator divides the voltage between VREF and AVSS into 1024, and that outputs the comparison voltage.

The A/D converter successively compares the comparison voltage Vref in each mode, dividing the VREF voltage (see below), with the input voltage.

#### • 10-bit reading

$$V_{ref} = \frac{V_{REF}}{1024} \times n \quad (n = 0-1023)$$

#### • 8-bit reading

$$V_{ref} = \frac{V_{REF}}{256} \times n \quad (n = 0-255)$$

## Channel Selector

The channel selector selects one of the input ports P17/AN7–P10/AN0.

## Comparator and Control Circuit

The comparator and control circuit compares an analog input voltage with the comparison voltage, and then stores the result in the AD conversion registers 1, 2. When an A/D conversion is completed, the control circuit sets the AD conversion completion bit and the AD interrupt request bit to "1".

Note that because the comparator consists of a capacitor coupling, set f(system clock) to 500 kHz or more during an A/D conversion.

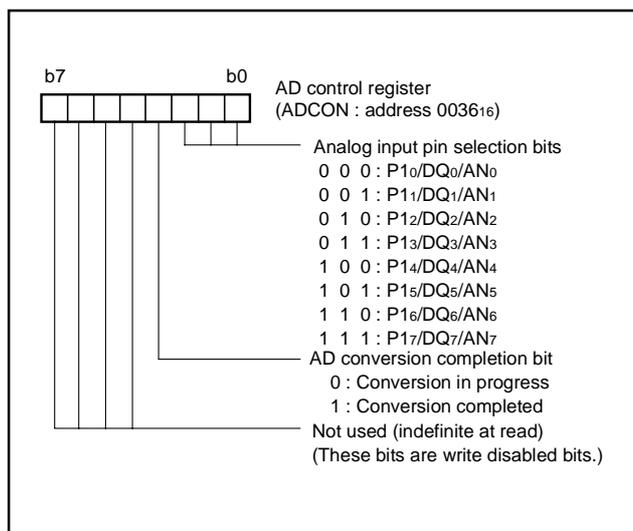


Fig. 132 Structure of AD control register

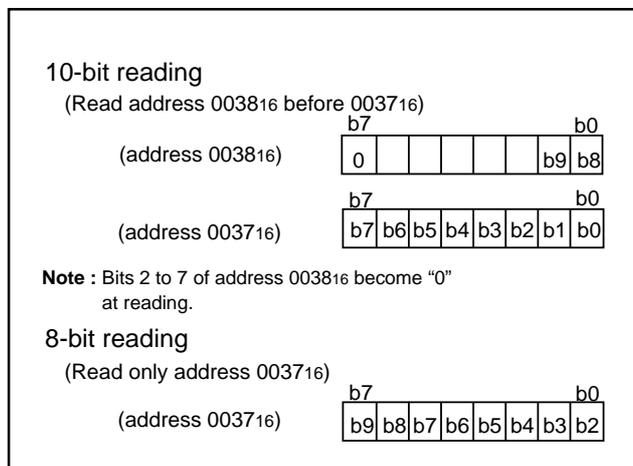


Fig. 133 10-bit/8-bit reading

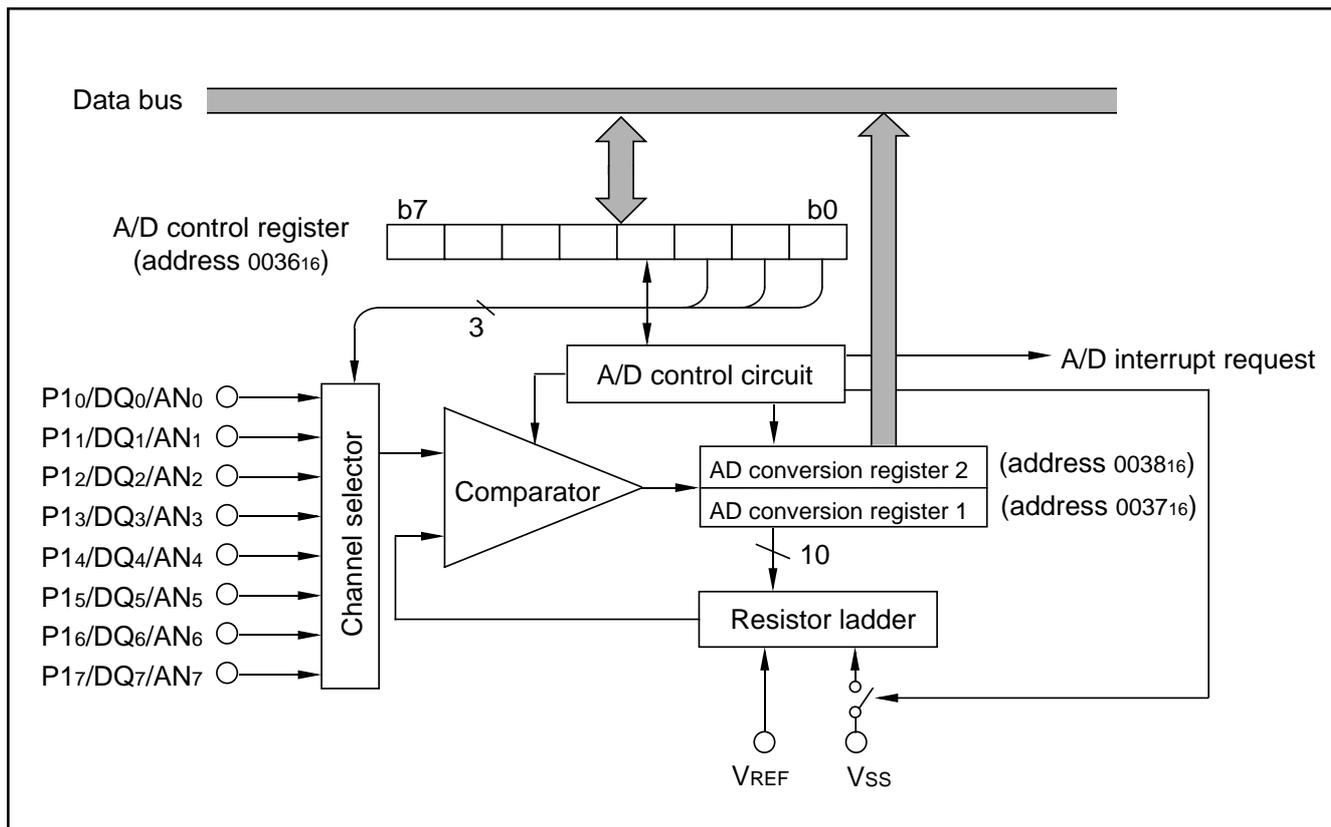


Fig. 134 A/D converter block diagram

## WATCHDOG TIMER

The watchdog timer gives a mean of returning to the reset status when a program cannot run on a normal loop (for example, because of a software run-away). The watchdog timer consists of an 8-bit watchdog timer L and an 8-bit watchdog timer H.

### Standard Operation of Watchdog Timer

When any data is not written into the watchdog timer control register (address 0039<sub>16</sub>) after resetting, the watchdog timer is in the stop state. The watchdog timer starts to count down by writing an optional value into the watchdog timer control register (address 0039<sub>16</sub>) and an internal reset occurs at an underflow of the watchdog timer H.

Accordingly, programming is usually performed so that writing to the watchdog timer control register (address 0039<sub>16</sub>) may be started before an underflow. When the watchdog timer control register (address 0039<sub>16</sub>) is read, the values of the high-order 6 bits of the watchdog timer H, STP instruction disable bit (bit 6), and watchdog timer H count source selection bit (bit 7) are read.

### Initial Value of Watchdog Timer

At reset or writing to the watchdog timer control register (address 0039<sub>16</sub>), each watchdog timer H and L is set to "FF<sub>16</sub>".

#### ● Watchdog timer H count source selection bit operation

Bit 7 of the watchdog timer control register (address 0039<sub>16</sub>) permits selecting a watchdog timer H count source. When this bit is set to "0", the count source becomes the underflow signal of watchdog timer L. The detection time is set to 131.072 ms at system clock 8 MHz frequency.

When this bit is set to "1", the count source becomes the system clock divided by 16. The detection time in this case is set to 512  $\mu$ s at system clock 8 MHz frequency. This bit is cleared to "0" after resetting.

#### ● Operation of STP instruction disable bit

Bit 6 of the watchdog timer control register (address 0039<sub>16</sub>) permits disabling the STP instruction when the watchdog timer is in operation.

When this bit is "0", the STP instruction is enabled.

When this bit is "1", the STP instruction is disabled.

Once the STP instruction is executed, an internal reset occurs.

When this bit is set to "1", it cannot be rewritten to "0" by program. This bit is cleared to "0" after resetting.

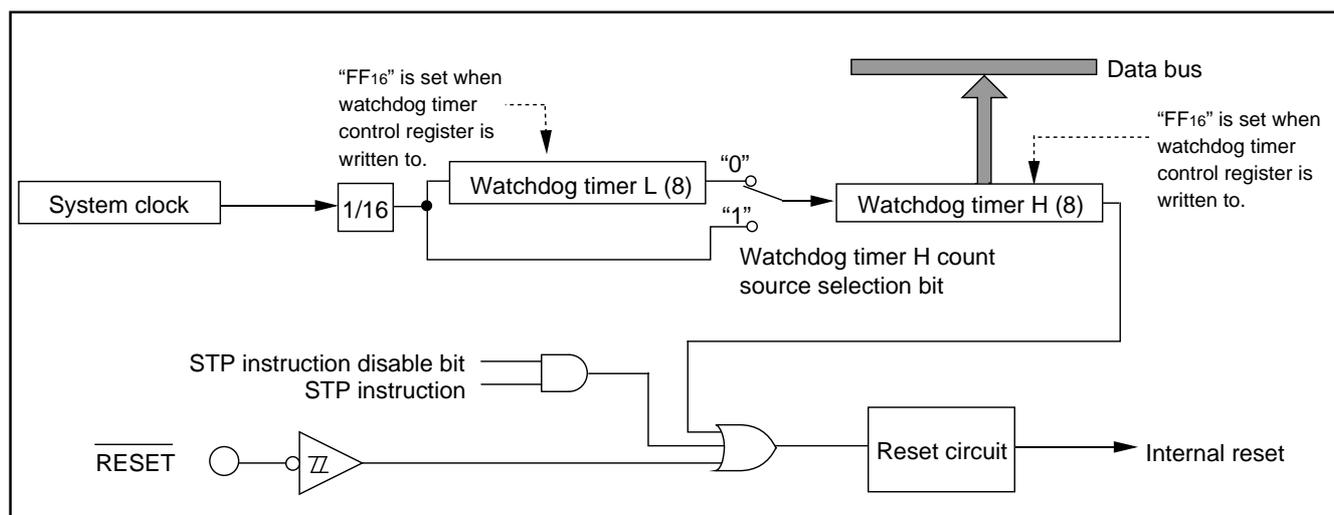


Fig. 135 Block diagram of Watchdog timer

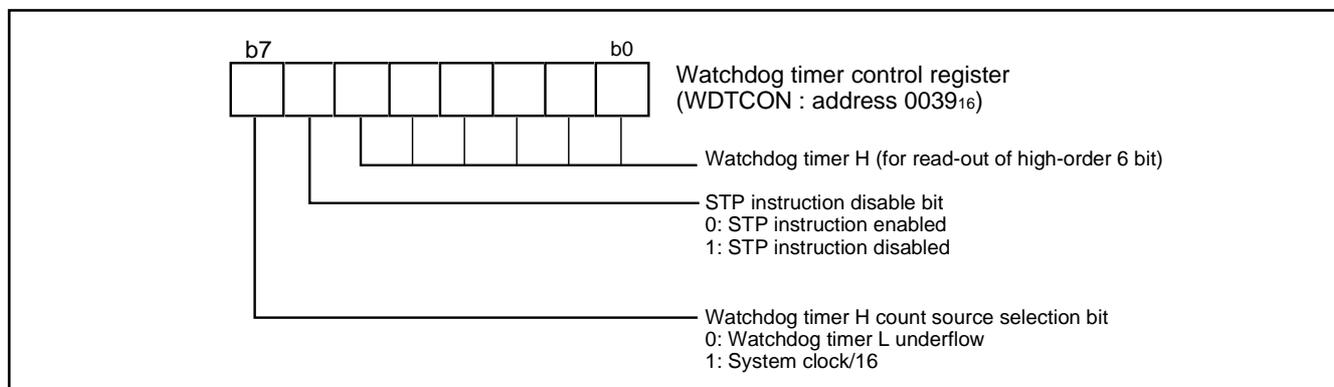


Fig. 136 Structure of Watchdog timer control register

### RESET CIRCUIT

To reset the microcomputer,  $\overline{\text{RESET}}$  pin should be held at an “L” level for 16 cycles or more of  $X_{\text{IN}}$ . Then the  $\overline{\text{RESET}}$  pin is returned to an “H” level (the power source voltage should be between 3.0 V and 5.25 V for L version, and the oscillation should be stable), reset is released. After the reset is completed, the program starts from the address contained in address  $\text{FFFD}_{16}$  (high-order byte) and address  $\text{FFFC}_{16}$  (low-order byte). Make sure that the reset input voltage is under 0.6 V for  $V_{\text{CC}}$  of 3.0 V (L version).

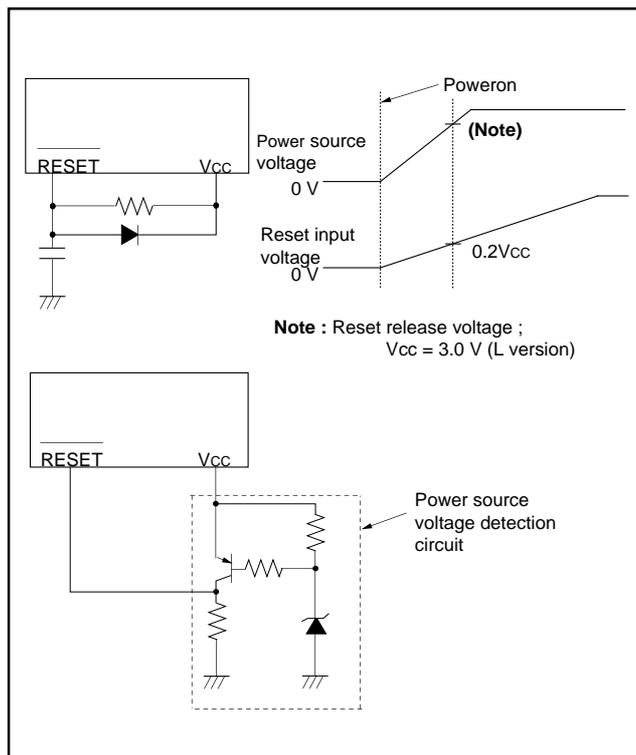


Fig. 137 Example of reset circuit

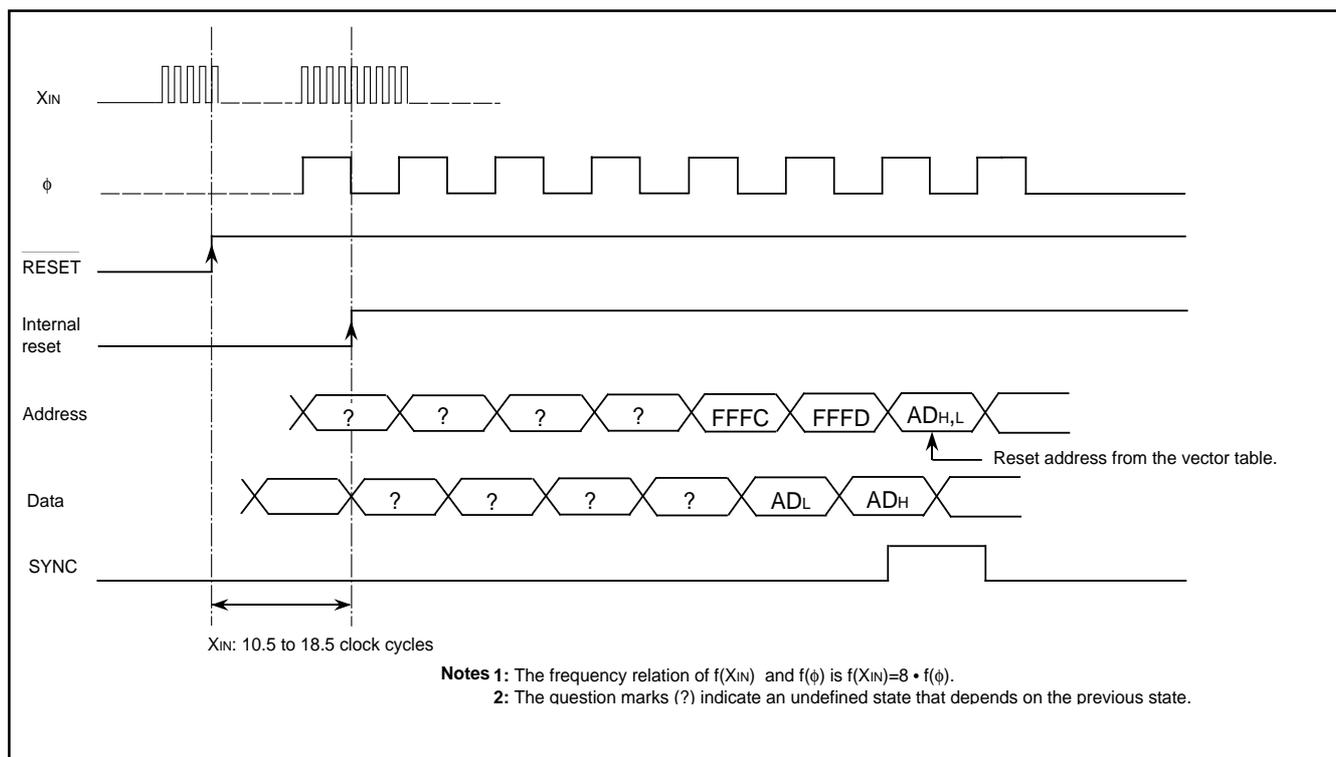


Fig. 138 Reset sequence

### PLL CIRCUIT (FREQUENCY SYNTHESIZER)

The PLL circuit generates  $f_{VCO}$  (PLL output clock), which is required for  $f_{USB}$  (USB clock) and  $f_{SYN}$  (fusb division clock), from  $f(X_{IN})$  (external input reference clock). Figure 139 shows the PLL circuit block diagram.

It is possible to input 6 or 12 MHz clock from the externals as a standard clock input. When using the USB function, set the PLL operation mode selection bit so that  $f_{VCO}$  may be set to 48 MHz.

The PLL circuit operates by setting the PLL operation enable bit to "1". When supplying  $f_{VCO}$  to the USB block, wait for the oscillation stable time (1ms or less) of PLL before selecting  $f_{VCO}$  with the USB clock selection bit.

According to the setting of the USB clock division ratio selection bit, the division clock of  $f_{USB}$  is supplied to  $f_{SYN}$ . When using this clock as system clock, set the USB clock division ratio selection bit so that it may be set to 6 MHz, 8 MHz or 12 MHz. (However, using it only when  $f_{USB}$  is 48MHz is recommended).

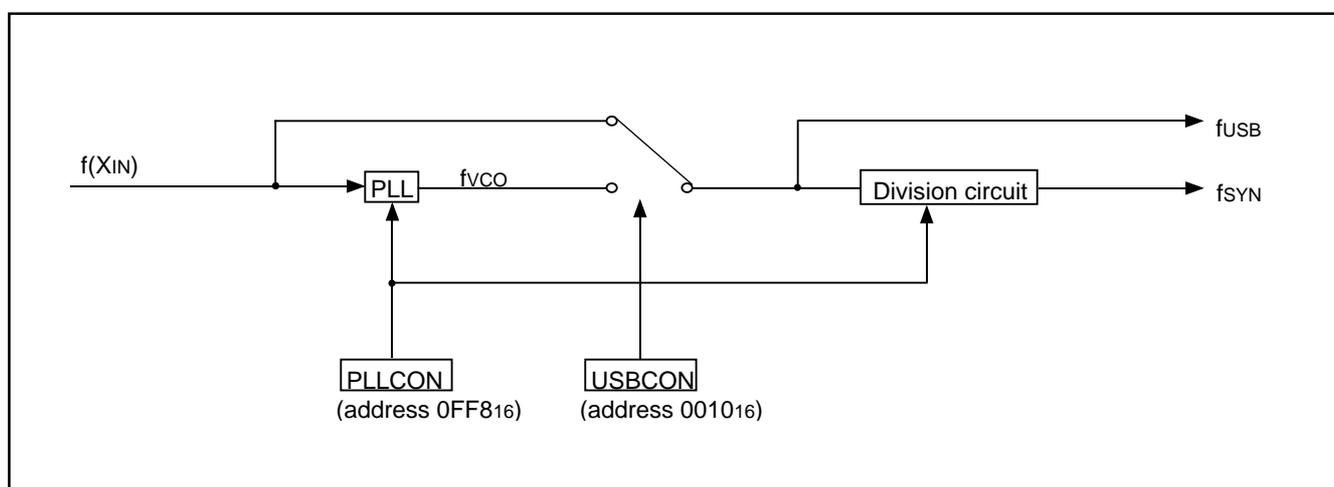


Fig. 139 Block diagram of PLL circuit

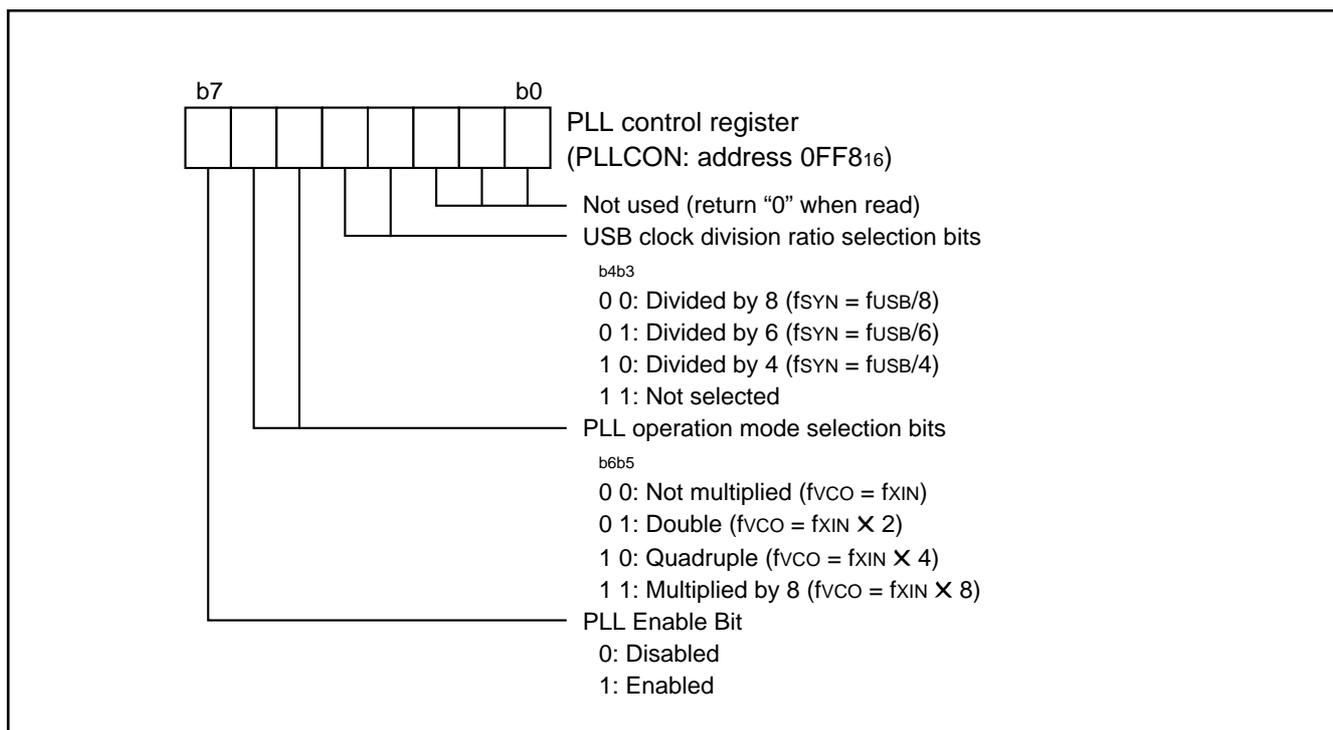


Fig. 140 Structure of PLL control register

## CLOCK GENERATING CIRCUIT

An oscillation circuit can be formed by connecting a resonator between XIN and XOUT. Use the circuit constants in accordance with the resonator manufacturer's recommended values. No external resistor is needed between XIN and XOUT since a feed-back resistor exists on-chip. (An external feed-back resistor may be needed depending on conditions.)

## Frequency Control

Either f<sub>SYN</sub> or f(XIN) can be selected as an internal system clock. Furthermore, the frequency of internal clock  $\phi$  can be selected by the system clock division ratio selection bit.

### (1) f<sub>SYN</sub> clock

f<sub>SYN</sub> clock is generated by the PLL circuit. f(XIN) or f<sub>VCO</sub> can be selected as an input clock. When using as an internal system clock, there is restriction on use. Refer to the clause of "PLL CIRCUIT".

### (2) f(XIN) clock

The frequency applied to the XIN pin is used as an internal system clock frequency.

## Oscillation Control

### (1) Stop mode

If the STP instruction is executed, the internal clock  $\phi$  stops at an "H" level, and the XIN oscillator stops. When the oscillation stabilizing time set after STP instruction released bit is "0," the prescaler 12 is set to "FF<sub>16</sub>" and timer 1 is set to "01<sub>16</sub>." When the oscillation stabilizing time set after STP instruction released bit is "1," set the sufficient time for oscillation of used oscillator to stabilize since nothing is set to the prescaler 12 and timer 1. XIN divided by 16 is compulsorily connected to the input of the prescaler 12. Oscillator restarts when an external interrupt (including USB resume interrupt) is received, but the internal clock  $\phi$  remains at "H" until timer 1 underflows. The internal clock  $\phi$  is not supplied until timer 1 underflows. Because the sufficient time is required for the oscillation to stabilize when a ceramic resonator etc. is used. When the oscillator is restarted by reset, apply "L" level to the  $\overline{\text{RESET}}$  pin until the oscillation is stable since a wait time will not be generated automatically.

### (2) Wait mode

If the WIT instruction is executed, the internal clock  $\phi$  stops at an "H" level, but the oscillator does not stop. The internal clock  $\phi$  restarts at reset or when an interrupt is received. Since the oscillator does not stop, normal operation can be started immediately after the clock is restarted.

To ensure that the interrupts will be received to release the STP or WIT state, their interrupt enable bits must be set to "1" before executing of the STP or WIT instruction.

When releasing the STP state, the prescaler 12 and timer 1 will start counting the clock XIN divided by 16. Accordingly, set the timer 1 interrupt enable bit to "0" before executing the STP instruction.

### ■Note

When using the oscillation stabilizing time set after STP instruction released bit set to "1", evaluate time to stabilize oscillation of the used oscillator and set the value to the timer 1 and prescaler 12.

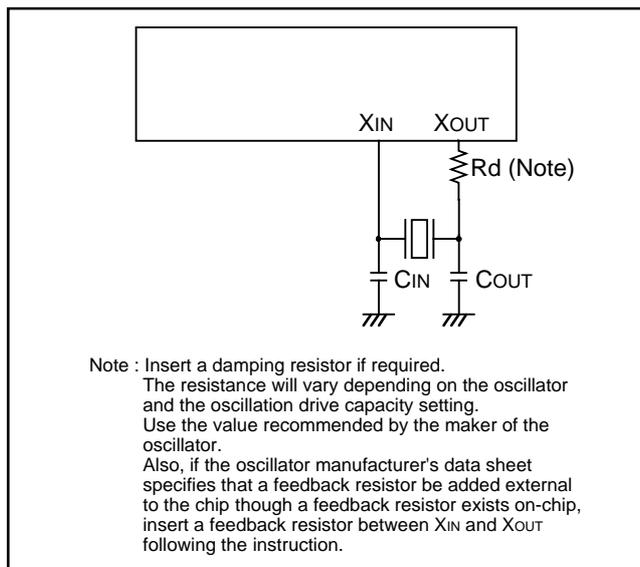


Fig. 141 Ceramic resonator or quartz-crystal oscillator circuit

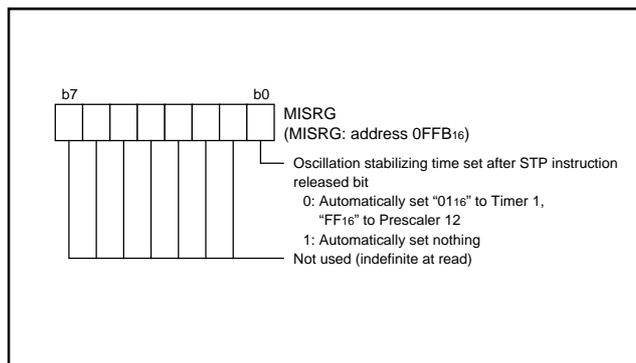


Fig. 143 Structure of MISRG

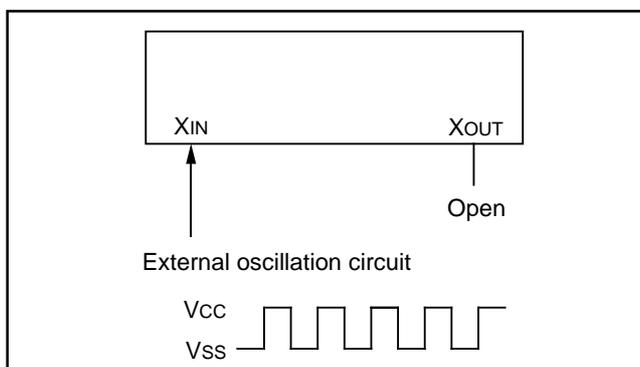


Fig. 142 External clock input circuit

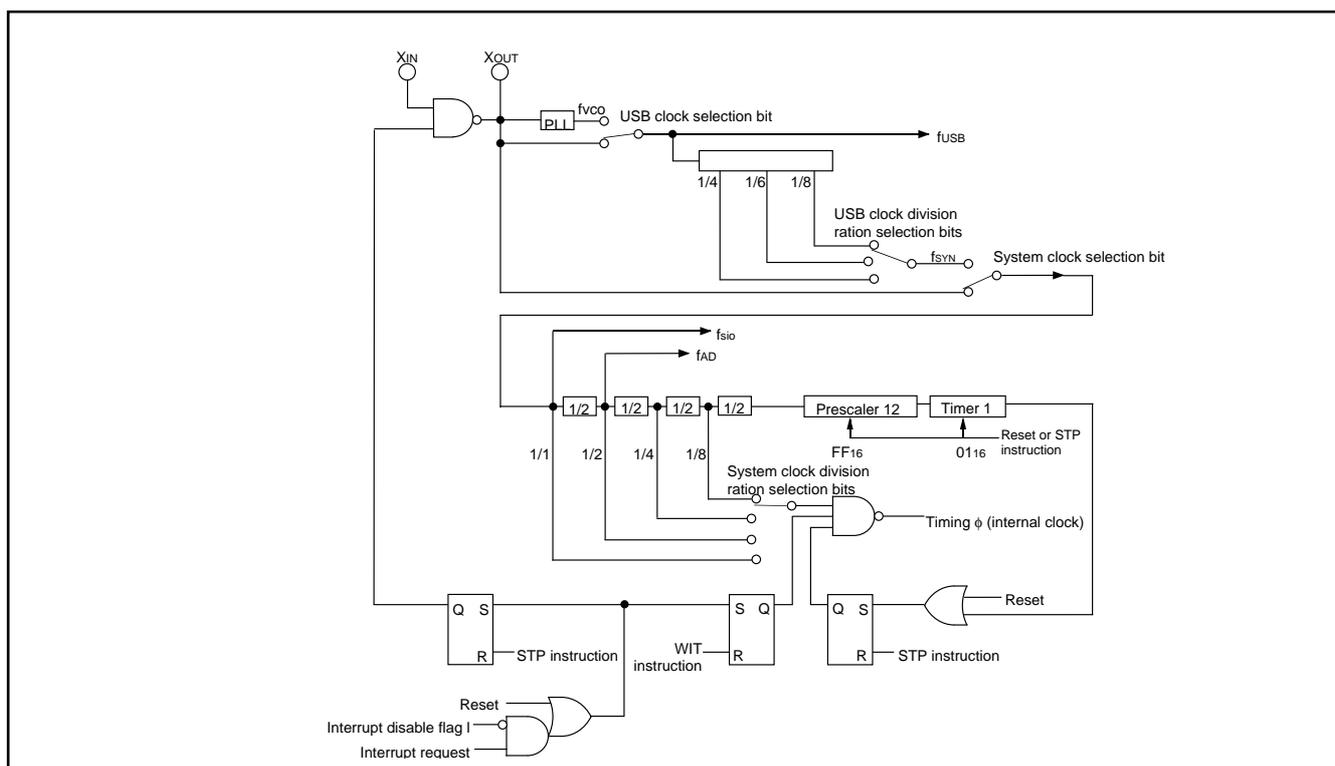


Fig. 144 System clock generating circuit block diagram (single-chip mode)

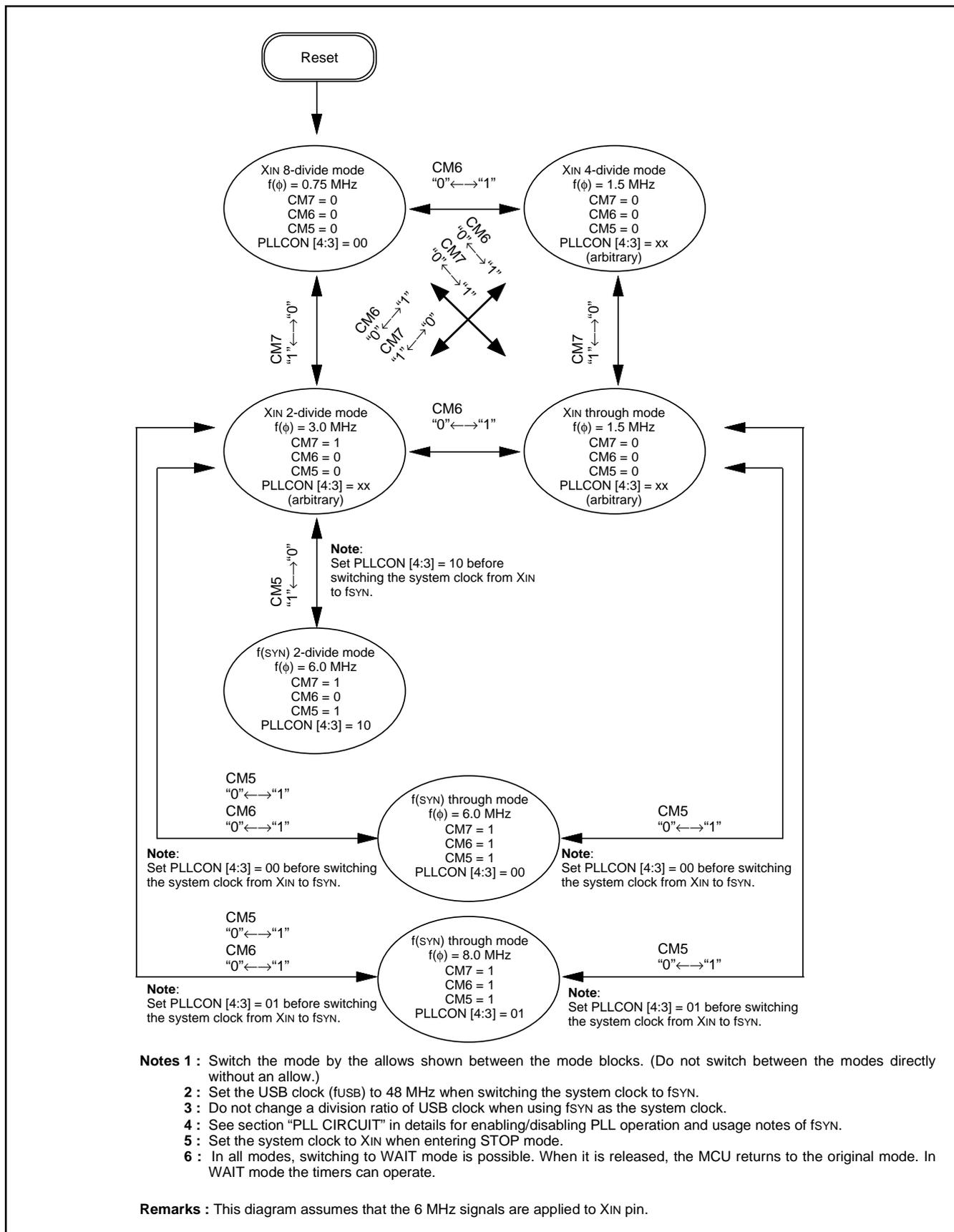


Fig. 145 State transitions of clock

## FLASH MEMORY MODE

The 38K2 group's flash memory version has an internal new DINOR (Divided bit line NOR) flash memory that can be rewritten with a single power source when Vcc is 4.5 to 5.25 V, and 2 power sources when Vcc is 3.0 to 4.5 V.

For this flash memory, three flash memory modes are available in which to read, program, and erase: the parallel I/O and standard serial I/O modes in which the flash memory can be manipulated using a programmer and the CPU rewrite mode in which the flash memory can be manipulated by the Central Processing Unit (CPU).

### Summary

Table 9 lists the summary of the 38K2 group's flash memory version.

This flash memory version has some blocks on the flash memory as shown in Figure 146 and each block can be erased. The flash memory is divided into User ROM area and Boot ROM area.

In addition to the ordinary User ROM area to store the MCU operation control program, the flash memory has a Boot ROM area that is used to store a program to control rewriting in CPU rewrite and standard serial I/O modes. This Boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the factory. However, the user can write a rewrite control program in this area that suits the user's application system. This Boot ROM area can be rewritten in only parallel I/O mode.

Table 9 Summary of 38K2 group's flash memory version

Item		Specifications
Power source voltage (Vcc)		3.00 – 5.25 V (L version) (Program and erase in 4.00 to 5.25 V of Vcc.) 3.00 – 4.00 V (L version) (Program and erase in 3.00 to 5.25 V of Vcc.)
Program/Erase VPP voltage (VPP)		4.50 – 5.25 V
Flash memory mode		3 modes; Flash memory can be manipulated as follows: •CPU rewrite mode: Manipulated by the Central Processing Unit (CPU). •Parallel I/O mode: Manipulated using an external programmer ( <b>Note 1</b> ) •Standard serial I/O mode: Manipulated using an external programmer ( <b>Note 1</b> )
Erase block division	User ROM area	1 block (32 Kbytes)
	Boot ROM area	1 block (4 Kbytes) ( <b>Note 2</b> )
Program method		Byte program
Erase method		Batch erasing
Program/Erase control method		Program/Erase control by software command
Number of commands		6 commands
Number of program/Erase times		100 times
Data retention period		10 years
ROM code protection		Available in parallel I/O mode and standard serial I/O mode

**Notes 1:** In the parallel I/O mode or the standard serial I/O mode, use the exclusive external equipment flash programmer which supports the 38K2 Group (flash memory version).

**2:** The Boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the factory. This Boot ROM area can be rewritten in only parallel I/O mode.

### (1) CPU Rewrite Mode

In CPU rewrite mode, the internal flash memory can be operated on (read, program, or erase) under control of the Central Processing Unit (CPU).

In CPU rewrite mode, only the User ROM area shown in Figure 146 can be rewritten; the Boot ROM area cannot be rewritten. Make sure the program and block erase commands are issued for only the User ROM area and each block area.

The control program for CPU rewrite mode can be stored in either User ROM or Boot ROM area. In the CPU rewrite mode, because the flash memory cannot be read from the CPU, the rewrite control program must be transferred to internal RAM area to be executed before it can be executed.

### Microcomputer Mode and Boot Mode

The control program for CPU rewrite mode must be written into the User ROM or Boot ROM area in parallel I/O mode beforehand. (If the control program is written into the Boot ROM area, the standard serial I/O mode becomes unusable.)

See Figure 146 for details about the Boot ROM area.

Normal microcomputer mode is entered when the microcomputer is reset with pulling CNVss pin low. In this case, the CPU starts operating using the control program in the User ROM area.

When the microcomputer is reset by pulling the P16 (CE) pin high, the CNVss pin high, the CPU starts operating using the control program in the Boot ROM area. This mode is called the "Boot" mode.

### Block Address

Block addresses refer to the maximum address of each block. These addresses are used in the block erase command.

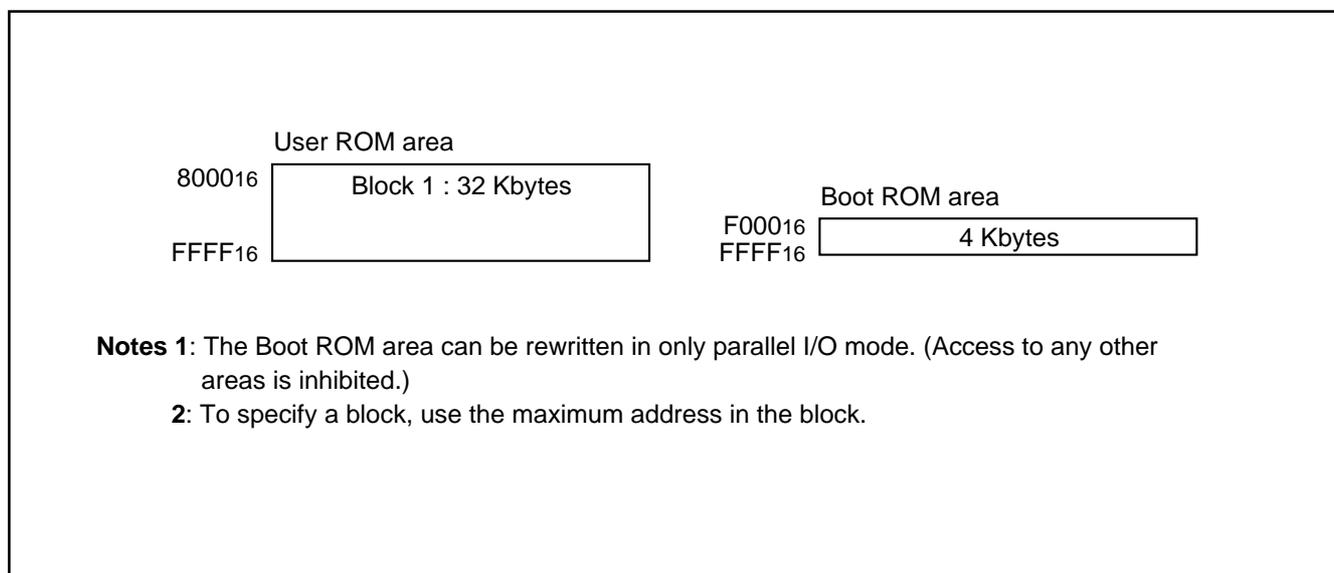


Fig. 146 Block diagram of built-in flash memory

### Outline Performance (CPU Rewrite Mode)

CPU rewrite mode is usable in the single-chip or Boot mode. The only User ROM area can be rewritten in CPU rewrite mode.

In CPU rewrite mode, the CPU erases, programs and reads the internal flash memory as instructed by software commands. This rewrite control program must be transferred to a memory such as the internal RAM before it can be executed.

The MCU enters CPU rewrite mode by applying 4.50 V to 5.25 V to the CNVSS pin and setting "1" to the CPU Rewrite Mode Select Bit (bit 1 of address 0FFE<sub>16</sub>). Software commands are accepted once the mode is entered.

Use software commands to control program and erase operations. Whether a program or erase operation has terminated normally or in error can be verified by reading the status register.

Figure 147 shows the flash memory control register.

Bit 0 is the RY/ $\overline{\text{BY}}$  status flag used exclusively to read the operating status of the flash memory. During programming and erase operations, it is "0" (busy). Otherwise, it is "1" (ready). This is equivalent to the RY/ $\overline{\text{BY}}$  pin function in parallel I/O mode.

Bit 1 is the CPU Rewrite Mode Select Bit. When this bit is set to "1", the MCU enters CPU rewrite mode. Software commands are accepted once the mode is entered. In CPU rewrite mode, the

CPU becomes unable to access the internal flash memory directly. Therefore, use the control program in a memory other than internal flash memory for write to bit 1. To set this bit to "1", it is necessary to write "0" and then write "1" in succession. The bit can be set to "0" by only writing "0".

Bit 2 is the CPU Rewrite Mode Entry Flag. This flag indicates "1" in CPU rewrite mode, so that reading this flag can check whether CPU rewrite mode has been entered or not.

Bit 3 is the flash memory reset bit used to reset the control circuit of internal flash memory. This bit is used when exiting CPU rewrite mode and when flash memory access has failed. When the CPU Rewrite Mode Select Bit is "1", setting "1" for this bit resets the control circuit. To set this bit to "1", it is necessary to write "0" and then write "1" in succession. To release the reset, it is necessary to set this bit to "0".

Bit 4 is the User Area/Boot Area Select Bit. When this bit is set to "1", Boot ROM area is accessed, and CPU rewrite mode in Boot ROM area is available. In Boot mode, this bit is set to "1" automatically. Reprogramming of this bit must be in a memory other than internal flash memory.

Figure 148 shows a flowchart for setting/releasing CPU rewrite mode.

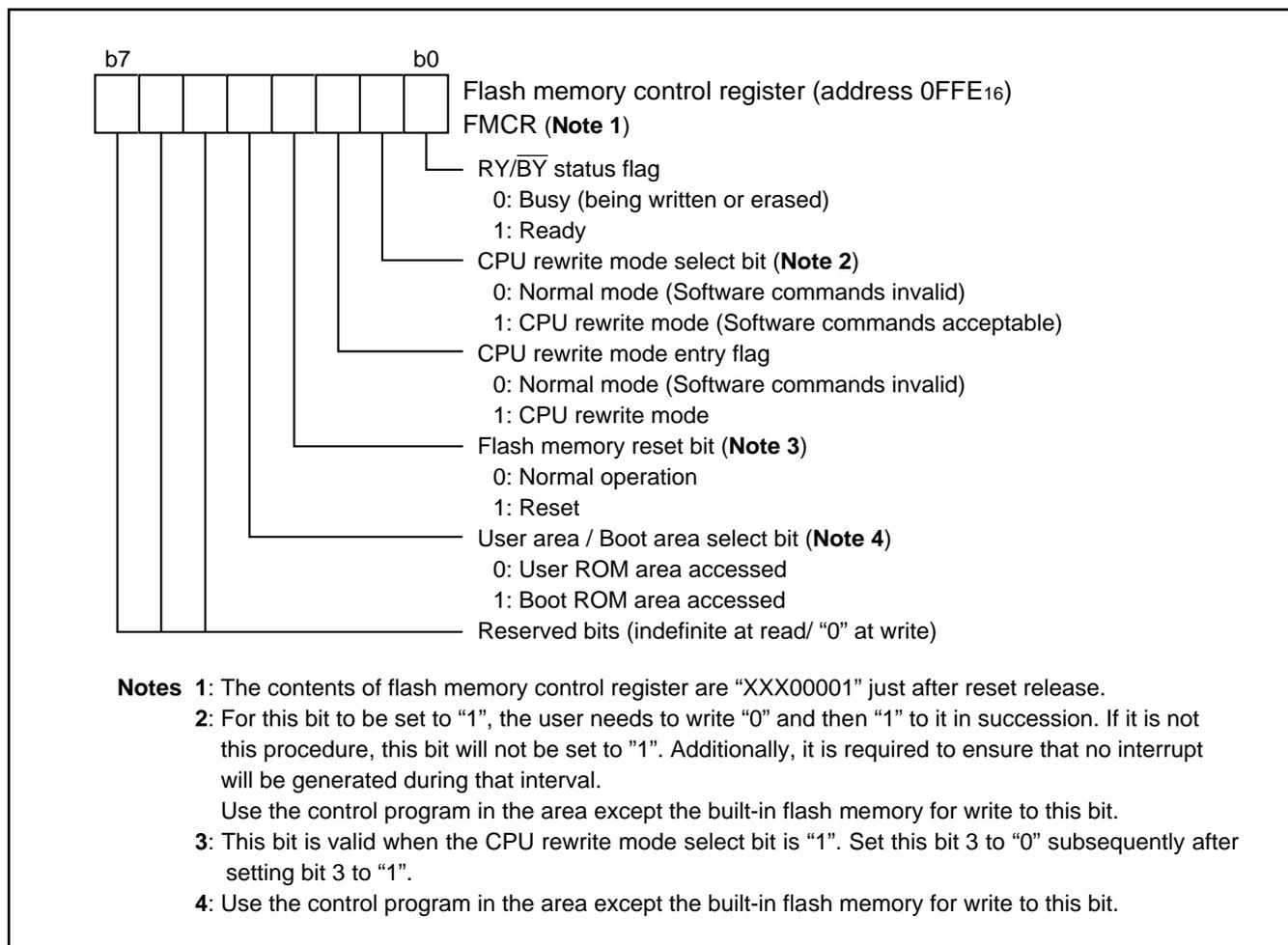


Fig. 147 Structure of flash memory control register

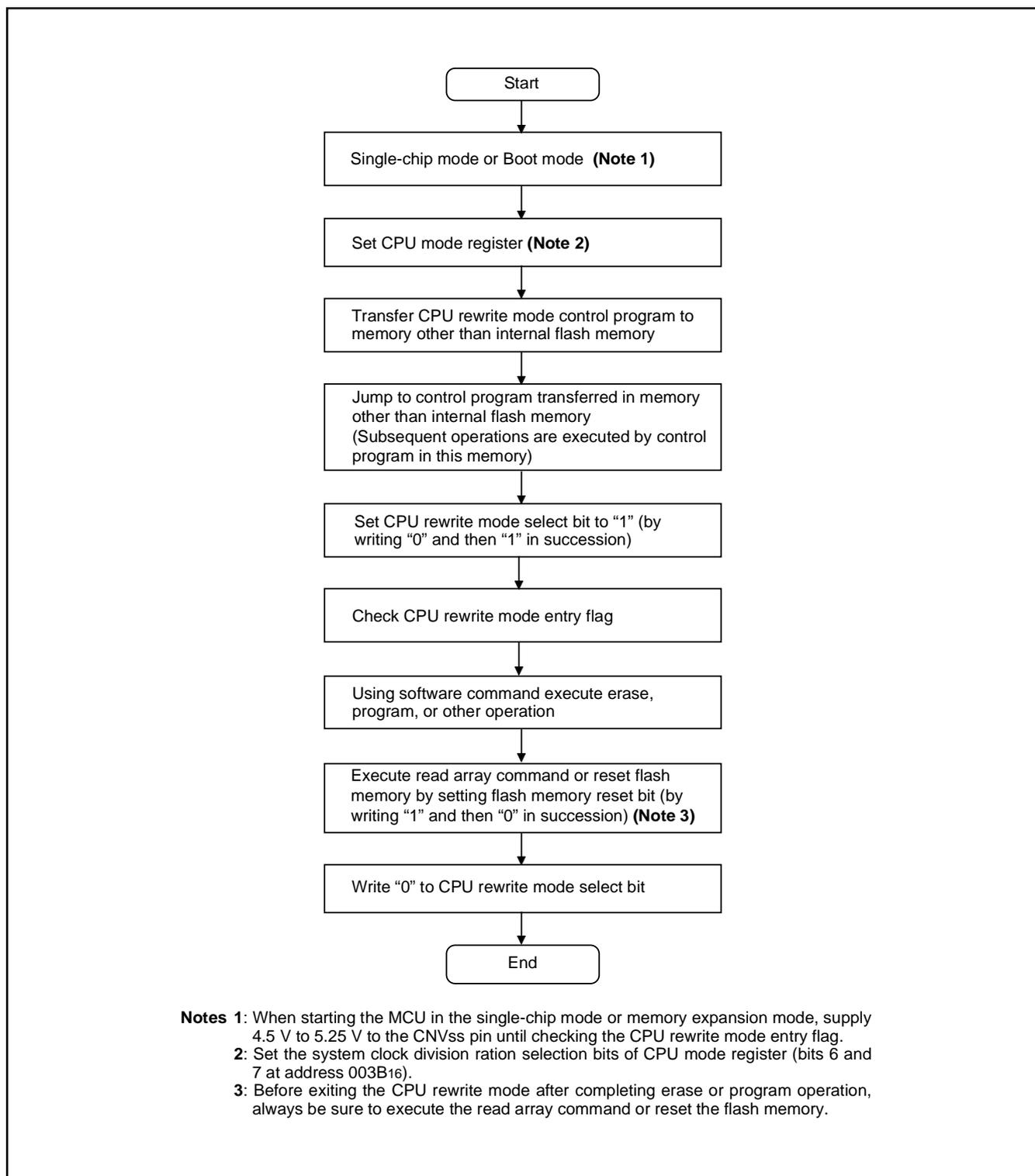


Fig. 148 CPU rewrite mode set/release flowchart

## Notes on CPU Rewrite Mode

Take the notes described below when rewriting the flash memory in CPU rewrite mode.

### ●Operation speed

During CPU rewrite mode, set the internal clock  $\phi$  to 1.5 MHz or less using the system clock division ratio selection bits (bits 6 and 7 of address 003B<sub>16</sub>).

### ●Instructions inhibited against use

The instructions which refer to the internal data of the flash memory cannot be used during CPU rewrite mode .

### ●Interrupts inhibited against use

The interrupts cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory.

### ●Watchdog timer

If the watchdog timer has been already activated, internal reset due to an underflow will not occur because the watchdog timer is surely cleared during program or erase.

### ●Reset

Reset is always valid. The MCU is activated using the boot mode at release of reset in the condition of CNVss = "H", so that the program will begin at the address which is stored in addresses FFFC<sub>16</sub> and FFFD<sub>16</sub> of the boot ROM area.

## Software Commands

Table 10 lists the software commands.

After setting the CPU Rewrite Mode Select Bit to "1", write a software command to specify an erase or program operation.

Each software command is explained below.

### ●Read Array Command (FF<sub>16</sub>)

The read array mode is entered by writing the command code "FF<sub>16</sub>" in the first bus cycle. When an address to be read is input in one of the bus cycles that follow, the contents of the specified address are read out at the data bus (D<sub>0</sub> to D<sub>7</sub>).

The read array mode is retained intact until another command is written.

### ●Read Status Register Command (70<sub>16</sub>)

When the command code "70<sub>16</sub>" is written in the first bus cycle, the contents of the status register are read out at the data bus (D<sub>0</sub> to D<sub>7</sub>) by a read in the second bus cycle.

The status register is explained in the next section.

### ●Clear Status Register Command (50<sub>16</sub>)

This command is used to clear the bits SR4 and SR5 of the status register after they have been set. These bits indicate that operation has ended in an error. To use this command, write the command code "50<sub>16</sub>" in the first bus cycle.

### ●Program Command (40<sub>16</sub>)

Program operation starts when the command code "40<sub>16</sub>" is written in the first bus cycle. Then, if the address and data to program are written in the 2nd bus cycle, the control circuit of flash memory (data programming and verification) will start a program.

Whether the write operation is completed can be confirmed by reading the status register or the RY/BY Status Flag. When the program starts, the read status register mode is entered automatically and the contents of the status register is read at the data bus (D<sub>0</sub> to D<sub>7</sub>). The status register bit 7 (SR7) is set to "0" at the same time the write operation starts and is returned to "1" upon completion of the write operation. In this case, the read status register mode remains active until the read array command (FF<sub>16</sub>) is written.

During the program movement, The RY/BY Status Flag of flash memory control register is set to "0". When the program completes, it becomes "1".

At program end, program results can be checked by reading the status register.

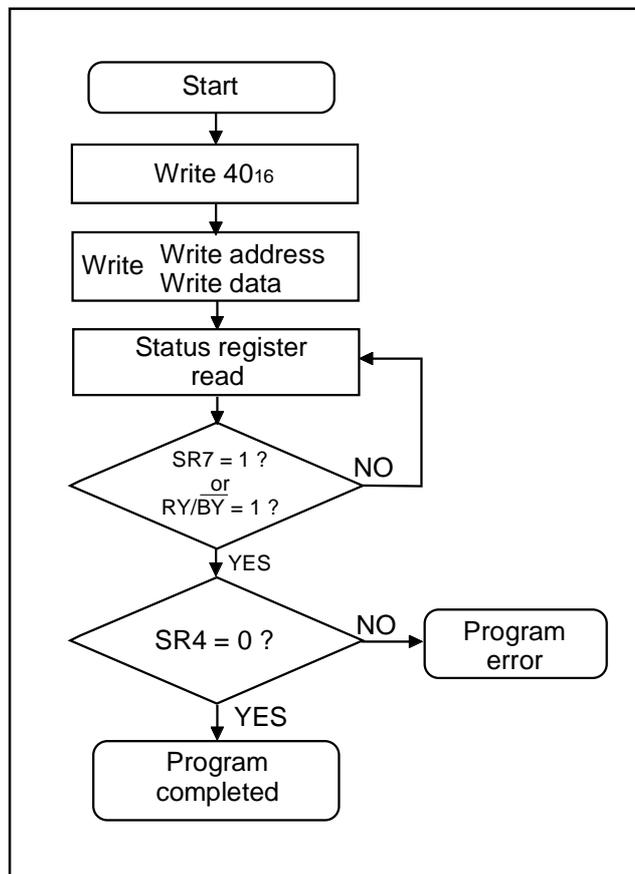


Fig. 149 Program flowchart

Table 10 List of software commands (CPU rewrite mode)

Command	Cycle number	First bus cycle			Second bus cycle		
		Mode	Address	Data (D <sub>0</sub> to D <sub>7</sub> )	Mode	Address	Data (D <sub>0</sub> to D <sub>7</sub> )
Read array	1	Write	X (Note 4)	FF <sub>16</sub>			
Read status register	2	Write	X	70 <sub>16</sub>	Read	X	SRD (Note 1)
Clear status register	1	Write	X	50 <sub>16</sub>			
Program	2	Write	X	40 <sub>16</sub>	Write	WA (Note 2)	WD (Note 2)
Erase all blocks	2	Write	X	20 <sub>16</sub>	Write	X	20 <sub>16</sub>
Block erase	2	Write	X	20 <sub>16</sub>	Write	BA (Note 3)	D0 <sub>16</sub>

**Notes 1:** SRD = Status Register Data

**2:** WA = Write Address, WD = Write Data

**3:** BA = Block Address to be erased (Input the maximum address of each block.)

**4:** X denotes a given address in the User ROM area .

### ●Erase All Blocks Command (20<sub>16</sub>/20<sub>16</sub>)

By writing the command code "20<sub>16</sub>" in the first bus cycle and the confirmation command code "20<sub>16</sub>" in the second bus cycle that follows, the operation of erase all blocks (erase and erase verify) starts.

Whether the erase all blocks command is terminated can be confirmed by reading the status register or the RY/ $\overline{\text{BY}}$  Status Flag of flash memory control register. When the erase all blocks operation starts, the read status register mode is entered automatically and the contents of the status register can be read out at the data bus (D<sub>0</sub> to D<sub>7</sub>). The status register bit 7 (SR7) is set to "0" at the same time the erase operation starts and is returned to "1" upon completion of the erase operation. In this case, the read status register mode remains active until the read array command (FF<sub>16</sub>) is written.

The RY/ $\overline{\text{BY}}$  Status Flag is "0" during erase operation and "1" when the erase operation is completed as is the status register bit 7.

After the erase all blocks end, erase results can be checked by reading the status register. For details, refer to the section where the status register is detailed.

### ●Block Erase Command (20<sub>16</sub>/D0<sub>16</sub>)

By writing the command code "20<sub>16</sub>" in the first bus cycle and the confirmation command code "D0<sub>16</sub>" and the block address in the second bus cycle that follows, the block erase (erase and erase verify) operation starts for the block address of the flash memory to be specified.

Whether the block erase operation is completed can be confirmed by reading the status register or the RY/ $\overline{\text{BY}}$  Status Flag of flash memory control register. At the same time the block erase operation starts, the read status register mode is automatically entered, so that the contents of the status register can be read out. The status register bit 7 (SR7) is set to "0" at the same time the block erase operation starts and is returned to "1" upon completion of the block erase operation. In this case, the read status register mode remains active until the read array command (FF<sub>16</sub>) is written.

The RY/ $\overline{\text{BY}}$  Status Flag is "0" during block erase operation and "1" when the block erase operation is completed as is the status register bit 7.

After the block erase ends, erase results can be checked by reading the status register. For details, refer to the section where the status register is detailed.

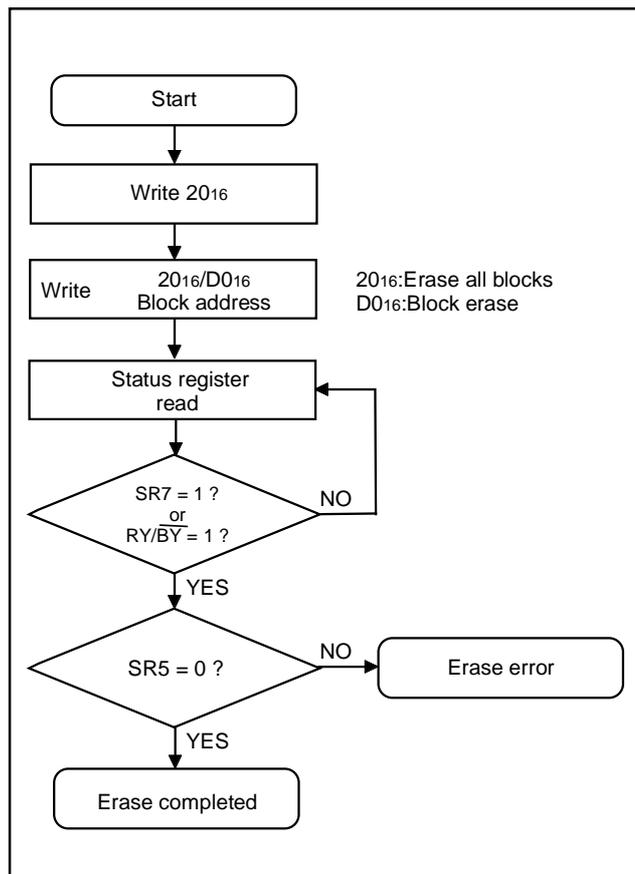


Fig. 150 Erase flowchart

## Status Register (SRD)

The status register shows the operating status of the flash memory and whether erase operations and programs ended successfully or in error. It can be read in the following ways:

- (1) By reading an arbitrary address from the User ROM area after writing the read status register command (70<sub>16</sub>)
- (2) By reading an arbitrary address from the User ROM area in the period from when the program starts or erase operation starts to when the read array command (FF<sub>16</sub>) is input.

Also, the status register can be cleared by writing the clear status register command (50<sub>16</sub>).

After reset, the status register is set to "80<sub>16</sub>".

Table 11 shows the status register. Each bit in this register is explained below.

### •Sequencer status (SR7)

The sequencer status indicates the operating status of the flash memory. This bit is set to "0" (busy) during write or erase operation and is set to "1" when these operations ends.

After power-on, the sequencer status is set to "1" (ready).

### •Erase status (SR5)

The erase status indicates the operating status of erase operation. If an erase error occurs, it is set to "1". When the erase status is cleared, it is set to "0".

### •Program status (SR4)

The program status indicates the operating status of write operation. When a write error occurs, it is set to "1".

The program status is set to "0" when it is cleared.

If "1" is written for any of the SR5 and SR4 bits, the program, erase all blocks, and block erase commands are not accepted. Before executing these commands, execute the clear status register command (50<sub>16</sub>) and clear the status register.

Table 11 Definition of each bit in status register

Each bit of SRD0 bits	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Sequencer status	Ready	Busy
SR6 (bit6)	Reserved	-	-
SR5 (bit5)	Erase status	Terminated in error	Terminated normally
SR4 (bit4)	Program status	Terminated in error	Terminated normally
SR3 (bit3)	Reserved	-	-
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

## Full Status Check

By performing full status check, it is possible to know the execution results of erase and program operations. Figure 151 shows a full status check flowchart and the action to be taken when each error occurs.

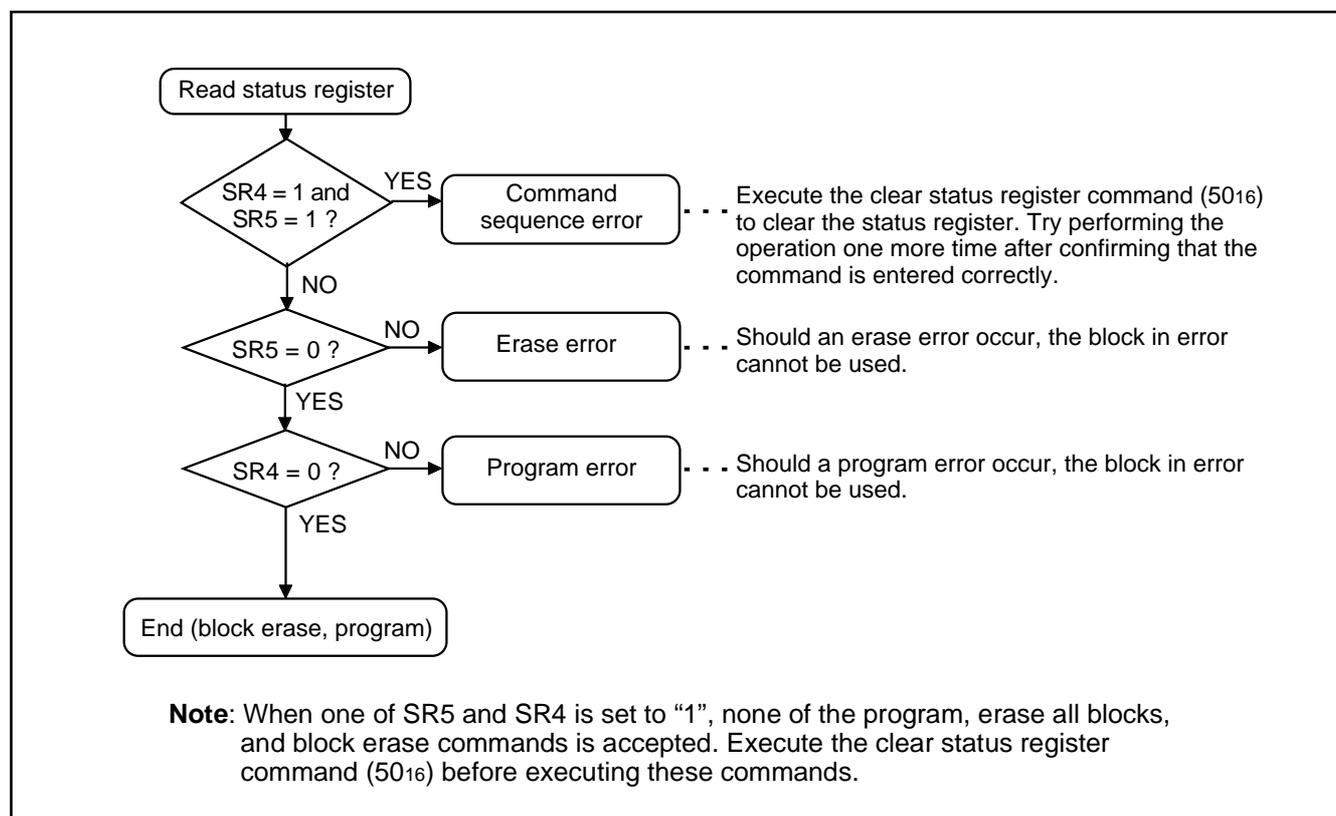


Fig. 151 Full status check flowchart and remedial procedure for errors

## Functions To Inhibit Rewriting Flash Memory Version

To prevent the contents of internal flash memory from being read out or rewritten easily, this MCU incorporates a ROM code protect function for use in parallel I/O mode and an ID code check function for use in standard serial I/O mode.

### ●ROM Code Protect Function

The ROM code protect function is the function to inhibit reading out or modifying the contents of internal flash memory by using the ROM code protect control register (address FFDB<sub>16</sub>) in parallel I/O mode. Figure 152 shows the ROM code protect control register (address FFDB<sub>16</sub>). (This address exists in the User ROM area.)

If one or both of the pair of ROM Code Protect Bits is set to "0", the ROM code protect is turned on, so that the contents of internal flash memory are protected against readout and modification. The ROM code protect is implemented in two levels. If level 2 is selected, the flash memory is protected even against readout by a shipment inspection LSI tester, etc. When an attempt is made to select both level 1 and level 2, level 2 is selected by default.

If both of the two ROM Code Protect Reset Bits are set to "00", the ROM code protect is turned off, so that the contents of internal flash memory can be read out or modified. Once the ROM code protect is turned on, the contents of the ROM Code Protect Reset Bits cannot be modified in parallel I/O mode. Use the serial I/O or CPU rewrite mode to rewrite the contents of the ROM Code Protect Reset Bits.

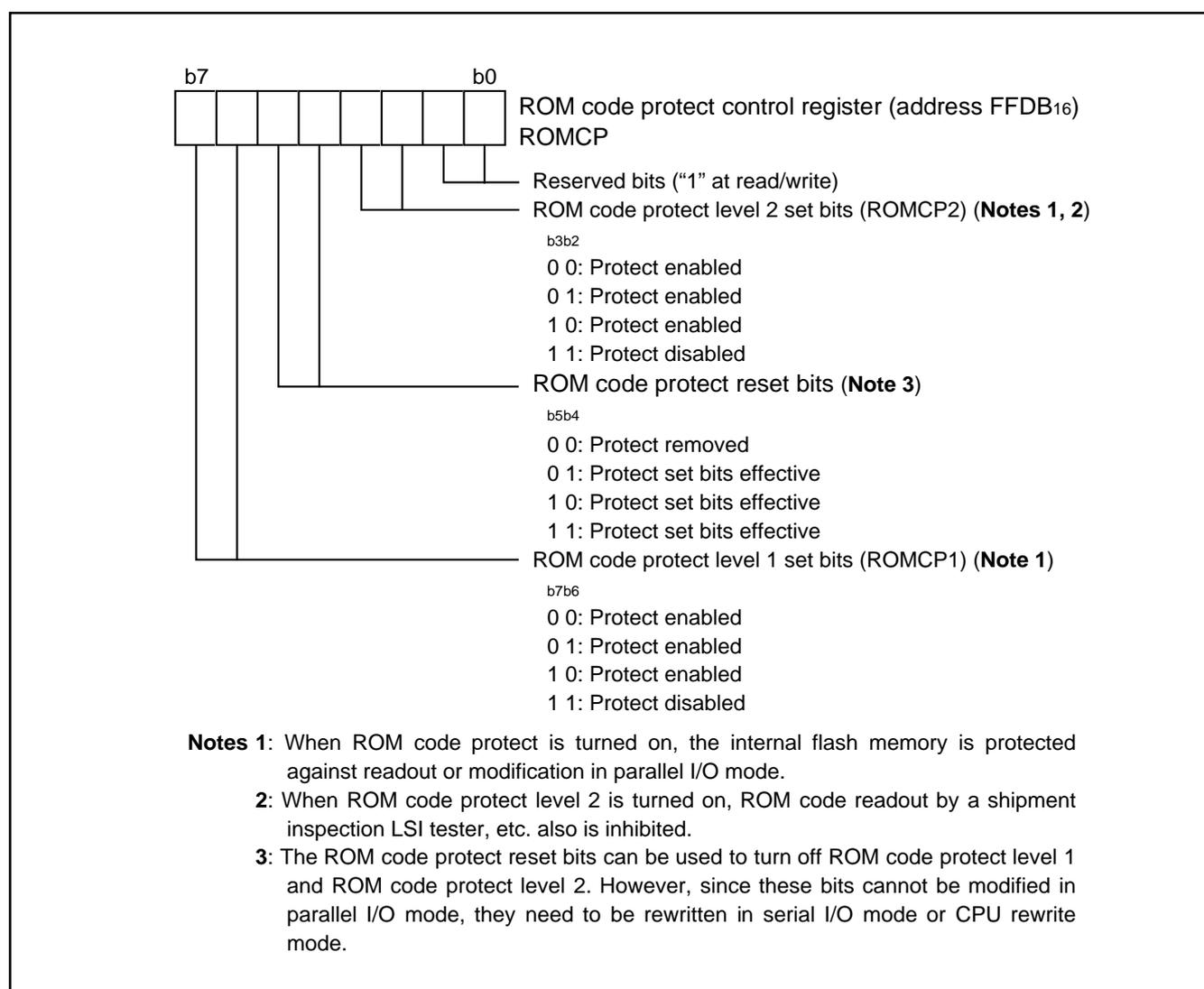


Fig. 152 Structure of ROM code protect control register

### ID Code Check Function

Use this function in standard serial I/O mode. When the contents of the flash memory are not blank, the ID code sent from the programmer is compared with the ID code written in the flash memory to see if they match. If the ID codes do not match, the commands sent from the programmer are not accepted. The ID code consists of 8-bit data, and its areas are FFDA<sub>16</sub> to FFDA<sub>16</sub>. Write a program which has had the ID code preset at these addresses to the flash memory.

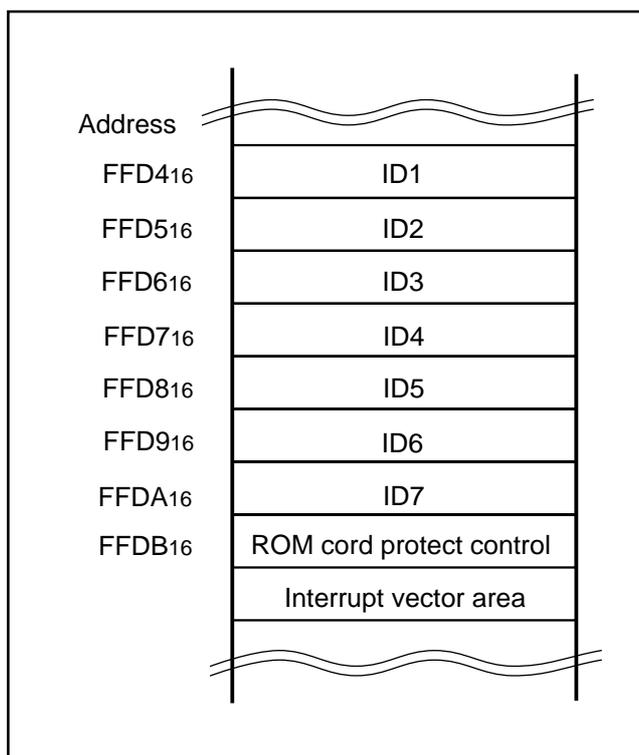


Fig. 153 ID code store addresses

## (2) Parallel I/O Mode

Parallel I/O mode is the mode which parallel output and input software command, address, and data required for the operations (read, program, erase, etc.) to a built-in flash memory. Use the exclusive external equipment flash programmer which supports the 38K2 Group (flash memory version). Refer to each programmer maker's handling manual for the details of the usage.

### User ROM and Boot ROM Areas

In parallel I/O mode, the user ROM and boot ROM areas shown in Figure 146 can be rewritten. Both areas of flash memory can be operated on in the same way.

The boot ROM area is 4 Kbytes in size. It is located at addresses F000<sub>16</sub> through FFFF<sub>16</sub>. Make sure program and block erase operations are always performed within this address range. (Access to any location outside this address range is prohibited.)

In the Boot ROM area, an erase block operation is applied to only one 4 Kbyte block.

The boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the Mitsubishi factory. Therefore, using the device in standard serial I/O mode, you must perform program and block erase in the user ROM area.

### (3) Standard Serial I/O Mode

The standard serial I/O mode inputs and outputs the software commands, addresses and data needed to operate (read, program, erase, etc.) the internal flash memory. This I/O is clock synchronized serial. This mode requires a purpose-specific peripheral unit. The standard serial I/O mode is different from the parallel I/O mode in that the CPU controls flash memory rewrite (uses the CPU rewrite mode), rewrite data input and so forth. The standard serial I/O mode is started by connecting "H" to the P16 (CE) pin and "H" to the P42 (SCLK) pin and "H" to the CNVSS (VPP) pin (apply 4.5 V to 5.25 V to Vpp from an external source), and releasing the reset operation. (In the ordinary microcomputer mode, set CNVSS pin to "L" level.)

This control program is written in the Boot ROM area when the product is shipped from Renesas Technology Corp.. Accordingly, make note of the fact that the standard serial I/O mode cannot be used if the Boot ROM area is rewritten in parallel I/O mode. Figure 154 shows the pin connections for the standard serial I/O mode.

In standard serial I/O mode, serial data I/O uses the four serial I/O pins SCLK, RxD, TxD and SRDY (BUSY). The SCLK pin is the transfer clock input pin through which an external transfer clock is input. The TxD pin is for CMOS output. The SRDY (BUSY) pin outputs "L" level when ready for reception and "H" level when reception starts.

Serial data I/O is transferred serially in 8-bit units.

In standard serial I/O mode, only the User ROM area shown in Figure 146 can be rewritten. The Boot ROM area cannot.

In standard serial I/O mode, a 7-byte ID code is used. When there is data in the flash memory, commands sent from the peripheral unit (programmer) are not accepted unless the ID code matches.

### Outline Performance (Standard Serial I/O Mode)

In standard serial I/O mode, software commands, addresses and data are input and output between the MCU and peripheral units (serial programmer, etc.) using 4-wire clock-synchronized serial I/O. In reception, software commands, addresses and program data are synchronized with the rise of the transfer clock that is input to the SCLK pin, and are then input to the MCU via the RxD pin. In transmission, the read data and status are synchronized with the fall of the transfer clock, and output from the TxD pin.

The TxD pin is for CMOS output. Transfer is in 8-bit units with LSB first.

When busy, such as during transmission, reception, erasing or program execution, the SRDY (BUSY) pin is "H" level. Accordingly, always start the next transfer after the SRDY (BUSY) pin is "L" level.

Also, data and status registers in a memory can be read after inputting software commands. Status, such as the operating state of the flash memory or whether a program or erase operation ended successfully or not, can be checked by reading the status register. Here following explains software commands, status registers, etc.

Table 12 Description of pin function (Standard Serial I/O Mode)

Pin name	Signal name	I/O	Function
VCC,VSS	Power supply		Apply 3.00 to 5.25 V (L version) to the Vcc pin and 0 V to the Vss pin.
VCC_E	Power supply		Connect this pin to Vcc.
CNVSS	VPP	I	Connect this pin to VPP (VPP = 4.50 to 5.25 V).
CNVSS2	CNVSS2	I	Connect this pin to Vss.
VREF	Analog reference voltage	I	Connect this pin to Vcc when not using.
DVCC, PVCC	Analog power supply		Connect this pin to Vcc.
PVSS	Analog power supply		Connect this pin to Vss.
RESET	Reset input	I	To reset, input "L" level for 20 cycles or longer clocks of $\phi$ .
XIN	Clock input	I	Connect a ceramic or crystal resonator between the XIN and XOUT pins. When entering an externally driven clock, enter it from XIN and leave XOUT open.
XOUT	Clock output	O	
USBVREF	USB reference voltage input	I	Connect this pin to Vcc when not using.
TrON	USB reference voltage output	O	Leave this pin open when not using.
D0+,D0-	USB upstream input	I/O	Input "L" level when not using.
D1+,D1-	USB downstream input	I/O	Input "L" level when not using.
D2+,D2-	USB downstream input	I/O	Input "L" level when not using.
P00 to P07	Input port P0	I	Input "L" or "H" level, or keep open.
P10 to P15	Input port P1	I	Input "L" or "H" level, or keep open.
P16	Input port P1	I	Input "L" or "H" level, or keep open. Input "H" level only at release of reset.
P17	Input port P1	I	Input "L" or "H" level, or keep open.
P20 to P24	Input port P2	I	Input "L" or "H" level, or keep open.
P30 to P37	Input port P3	I	Input "L" or "H" level, or keep open.
P40	RxD input	I	This is a serial data input pin.
P41	TxD output	O	This is a serial data output pin.
P42	SCLK input	I	This is a serial clock input pin. Input "H" level only at release of reset.
P43	BUSY output	O	This is a BUSY output pin.
P50 to P57	Input port P5	I	Input "L" or "H" level, or keep open.
P60 to P63	Input port P6	I	Input "L" or "H" level, or keep open.

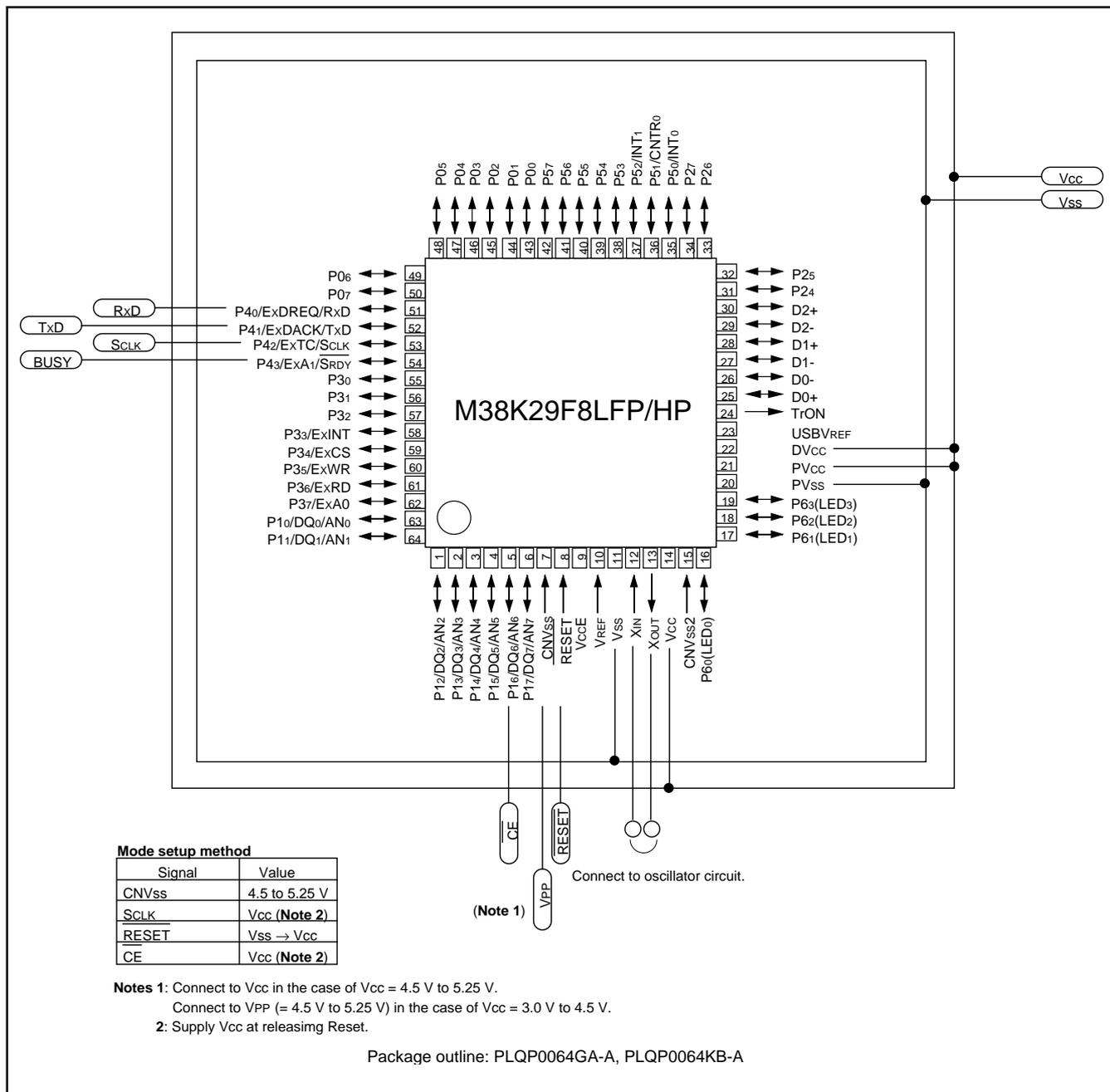


Fig. 154 Pin connection diagram in standard serial I/O mode

## Software Commands

Table 13 lists software commands. In standard serial I/O mode, erase, program and read are controlled by transferring software commands via the RxD pin. Software commands are explained

here below. Basically, the software commands of the standard serial I/O mode are the same as that of the parallel I/O mode, but the block erase function is excluded, and 4 commands are added: ID check, download, version data output and Boot ROM area output functions.

Table 13 Software commands (Standard serial I/O mode)

Control command	1st byte transfer	2nd byte	3rd byte	4th byte	5th byte	6th byte	.....	When ID is not verified
1 Page read	FF <sub>16</sub>	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
2 Page program	41 <sub>16</sub>	Address (middle)	Address (high)	Data input	Data input	Data input	Data input to 259th byte	Not acceptable
3 Erase all blocks	A7 <sub>16</sub>	D0 <sub>16</sub>						Not acceptable
4 Read status register	70 <sub>16</sub>	SRD output	SRD1 output					Acceptable
5 Clear status register	50 <sub>16</sub>							Not acceptable
6 ID check function	F5 <sub>16</sub>	Address (low)	Address (middle)	Address (high)	ID size	ID1	To ID7	Acceptable
7 Download function	FA <sub>16</sub>	Size (low)	Size (high)	Check-sum	Data input	To required number of times		Not acceptable
8 Version data output function	FB <sub>16</sub>	Version data output	Version data output to 9th byte	Acceptable				
9 Boot ROM area output function	FC <sub>16</sub>	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable

**Notes1:** Shading indicates transfer from the internal flash memory microcomputer to a programmer. All other data is transferred from a programmer to the internal flash memory microcomputer.

**2:** SRD refers to status register data. SRD1 refers to status register 1 data.

**3:** All commands can be accepted when the flash memory is totally blank.

**4:** Address low is A0 to A7; Address middle is A8 to A15; Address high is A16 to A23. Address-high A16 to A23 are always "0016".

The contents of software commands are explained as follows.

**●Page Read Command**

This command reads the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page read command as explained here following.

- (1) Transfer the "FF16" command code with the 1st byte.
- (2) Transfer addresses A8 to A15 and A16 to A23 with the 2nd and 3rd bytes respectively.
- (3) From the 4th byte onward, data (D0 to D7) for the page (256 bytes) specified with addresses A8 to A23 will be output sequentially from the smallest address first synchronized with the fall of the clock.

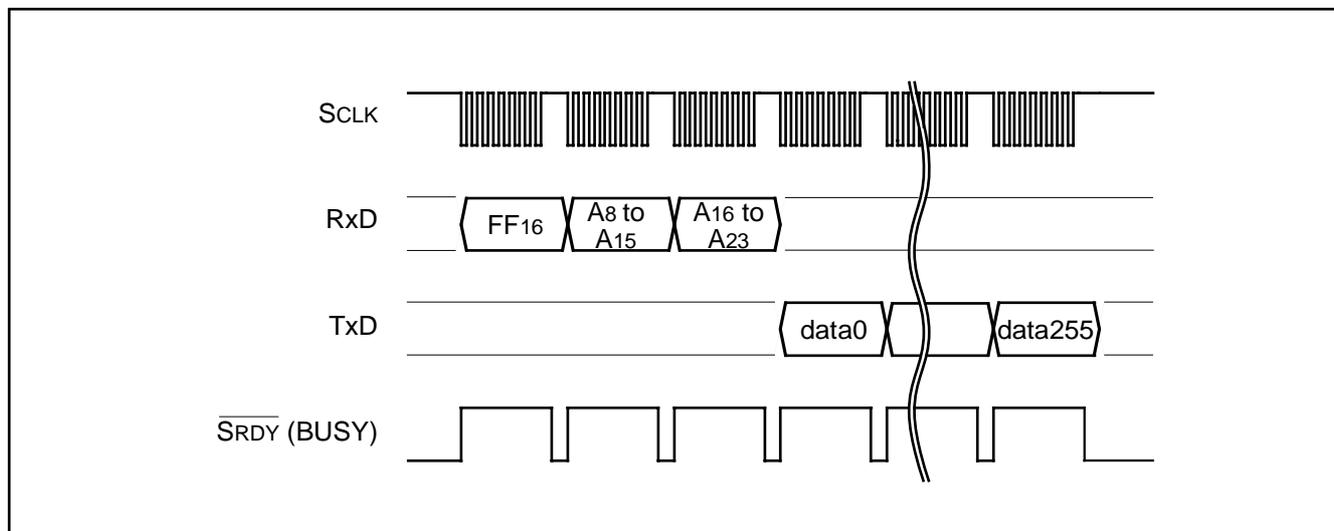


Fig. 155 Timing for page read

**●Read Status Register Command**

This command reads status information. When the "7016" command code is transferred with the 1st byte, the contents of the status register (SRD) with the 2nd byte and the contents of status register 1 (SRD1) with the 3rd byte are read.

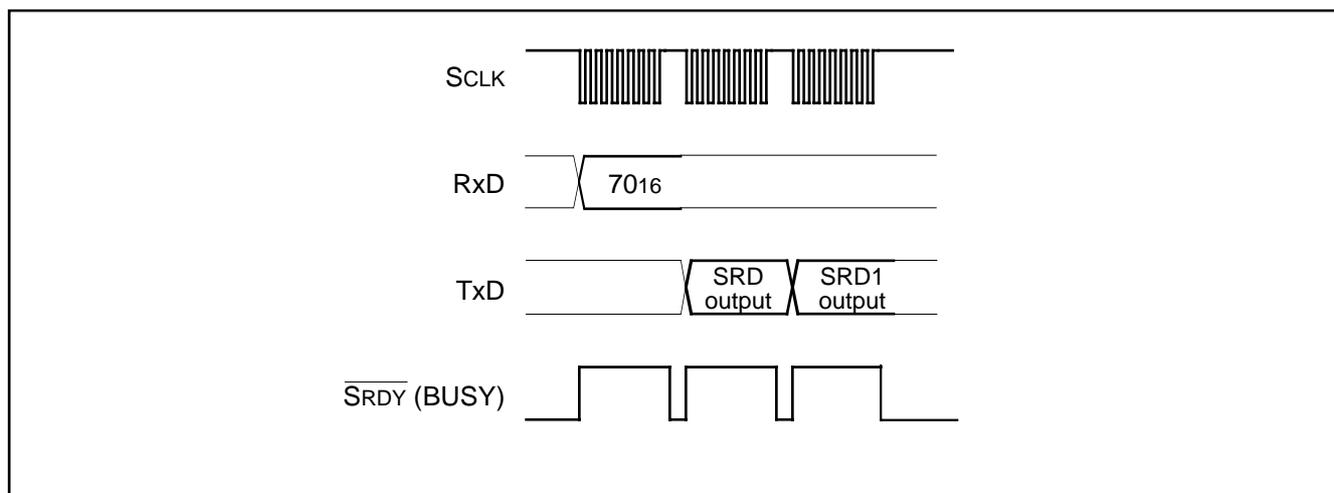


Fig. 156 Timing for reading status register

### ●Clear Status Register Command

This command clears the bits (SR3 to SR5) which are set when the status register operation ends in error. When the "5016" command code is sent with the 1st byte, the aforementioned bits are cleared. When the clear status register operation ends, the  $\overline{\text{SRDY}}$  (BUSY) signal changes from "H" to "L" level.

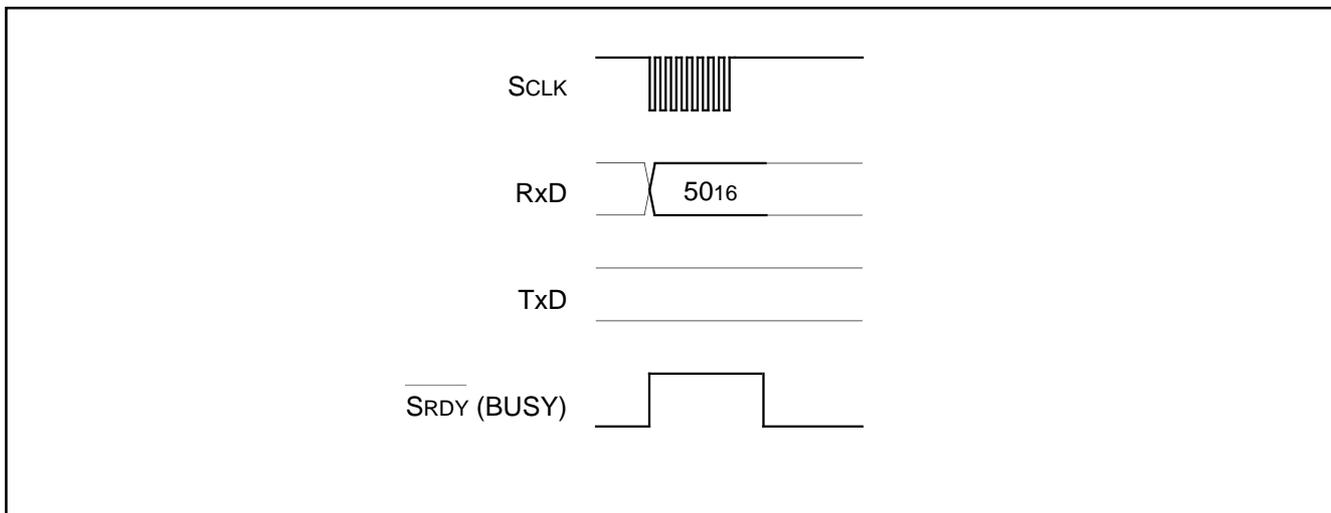


Fig. 157 Timing for clear status register

### ●Page Program Command

This command writes the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page program command as explained here following.

- (1) Transfer the "4116" command code with the 1st byte.
- (2) Transfer addresses A8 to A15 and A16 to A23 with the 2nd and 3rd bytes respectively.

- (3) From the 4th byte onward, as write data (D0 to D7) for the page (256 bytes) specified with addresses A8 to A23 is input sequentially from the smallest address first, that page is automatically written.

When reception setup for the next 256 bytes ends, the  $\overline{\text{SRDY}}$  (BUSY) signal changes from "H" to "L" level. The result of the page program can be known by reading the status register. For more information, see the section on the status register.

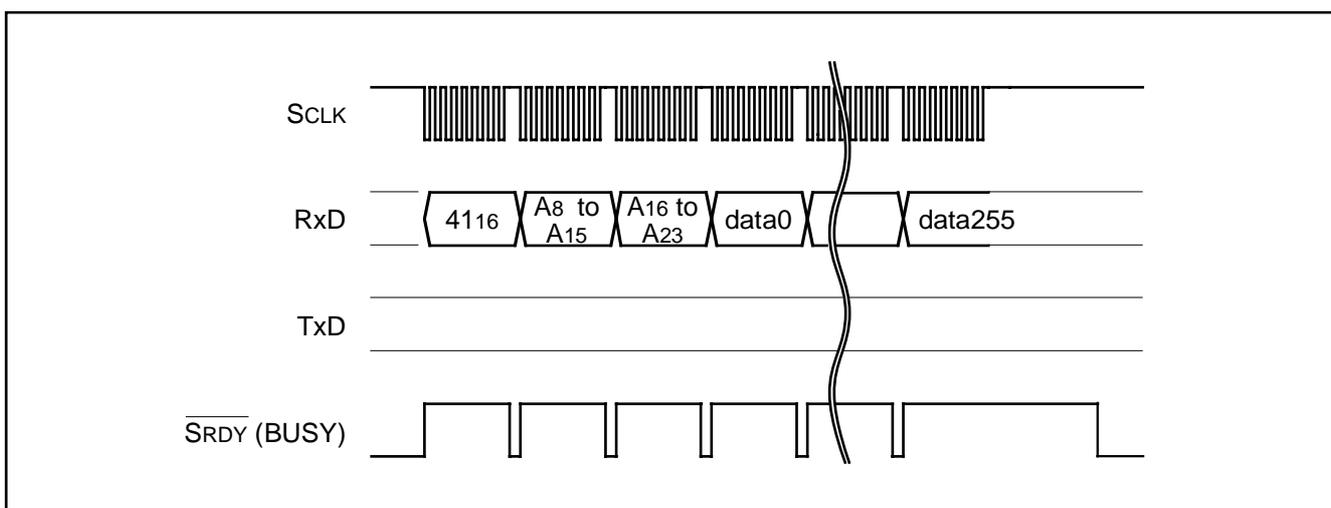


Fig. 158 Timing for page program

### ●Erase All Blocks Command

This command erases the contents of all blocks. Execute the erase all blocks command as explained here following.

- (1) Transfer the "A716" command code with the 1st byte.
- (2) Transfer the verify command code "D016" with the 2nd byte.

With the verify command code, the erase operation will start and continue for all blocks in the flash memory.

When erase all blocks end, the  $\overline{\text{SRDY}}$  (BUSY) signal changes from "H" to "L" level. The result of the erase operation can be known by reading the status register.

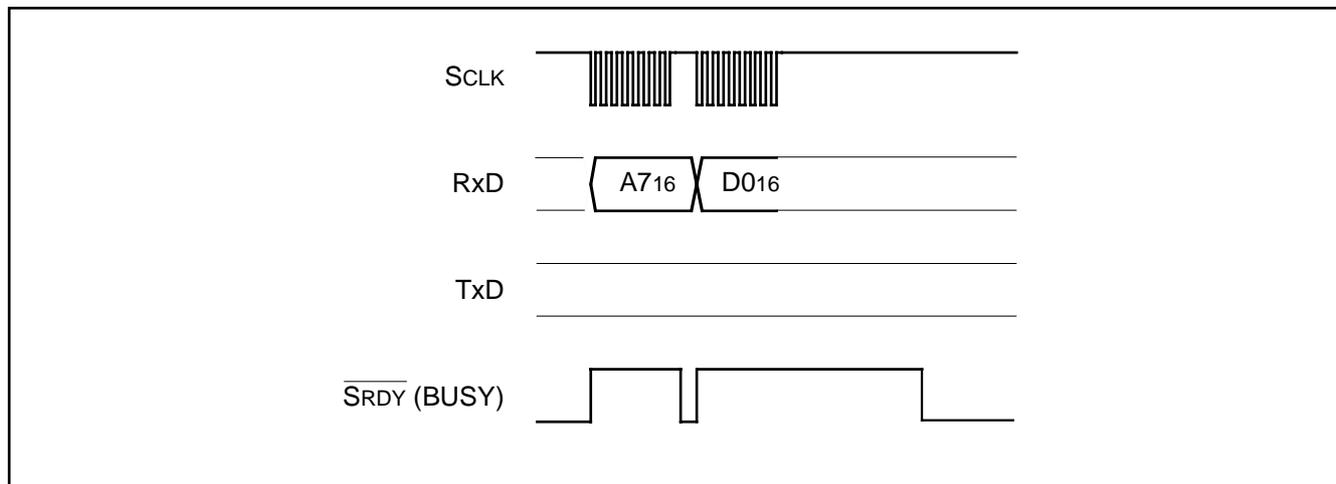


Fig. 159 Timing for erase all blocks

### ●Download Command

This command downloads a program to the RAM for execution.

Execute the download command as explained here following.

- (1) Transfer the "FA16" command code with the 1st byte.
- (2) Transfer the program size with the 2nd and 3rd bytes.
- (3) Transfer the check sum with the 4th byte. The check sum is added to all data sent with the 5th byte onward.
- (4) The program to execute is sent with the 5th byte onward.

When all data has been transmitted, if the check sum matches, the downloaded program is executed. The size of the program will vary according to the internal RAM.

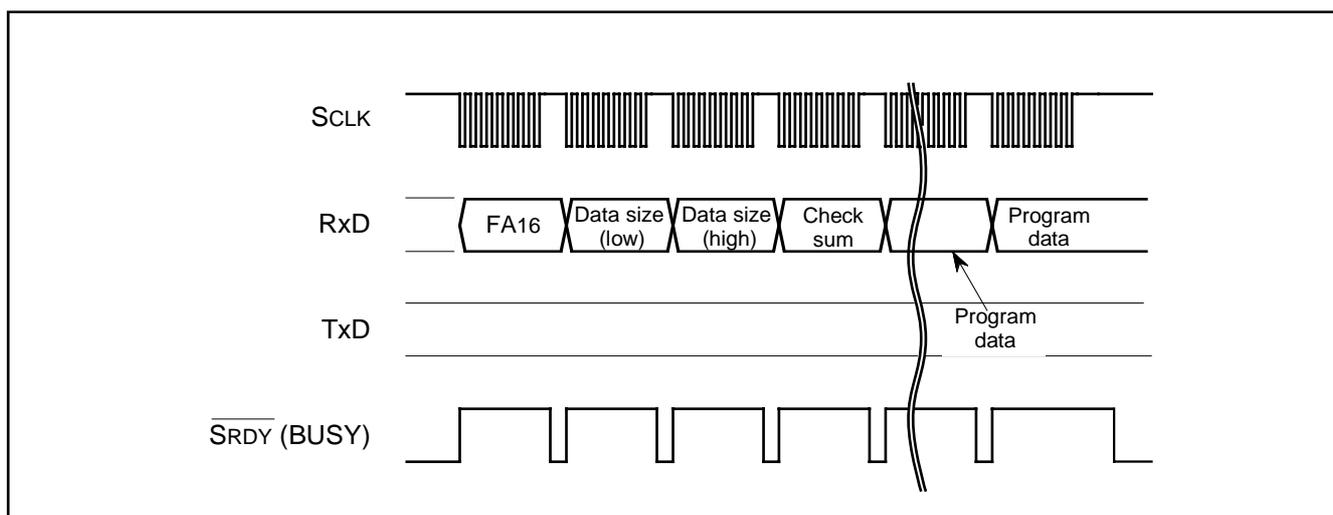


Fig. 160 Timing for download

**●Version Information Output Command**

This command outputs the version information of the control program stored in the Boot ROM area. Execute the version information output command as explained here following.

- (1) Transfer the "FB16" command code with the 1st byte.
  - (2) The version information will be output from the 2nd byte onward.
- This data is composed of 8 ASCII code characters.

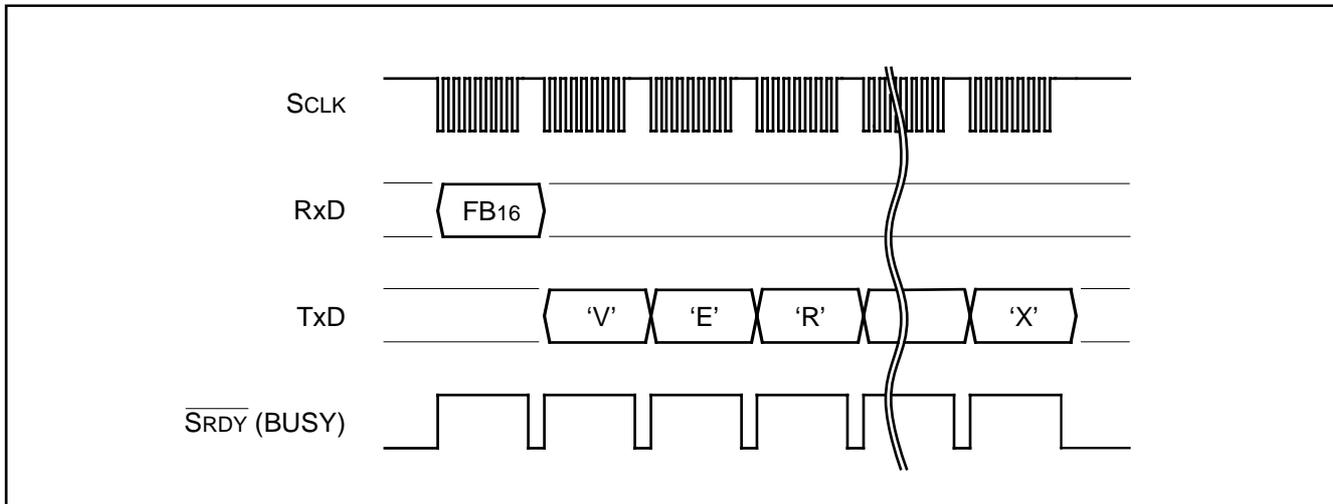


Fig. 161 Timing for version information output

**●Boot ROM Area Output Command**

This command reads the control program stored in the Boot ROM area in page (256 bytes) unit. Execute the Boot ROM area output command as explained here following.

- (1) Transfer the "FC16" command code with the 1st byte.
- (2) Transfer addresses A8 to A15 and A16 to A23 with the 2nd and 3rd bytes respectively.
- (3) From the 4th byte onward, data (D0 to D7) for the page (256 bytes) specified with addresses A8 to A23 will be output sequentially from the smallest address first synchronized with the fall of the clock.

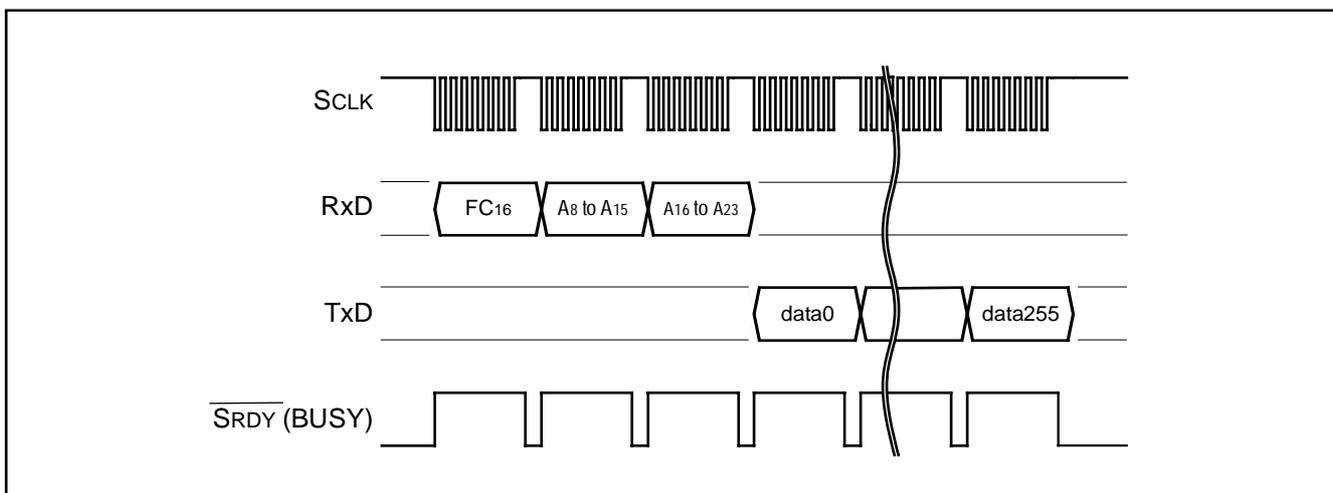


Fig. 162 Timing for Boot ROM area output

●ID Check

This command checks the ID code. Execute the boot ID check command as explained here following.

- (1) Transfer the "F5<sub>16</sub>" command code with the 1st byte.
- (2) Transfer addresses A0 to A7, A8 to A15 and A16 to A23 ("00<sub>16</sub>") of the 1st byte of the ID code with the 2nd, 3rd and 4th respectively.
- (3) Transfer the number of data sets of the ID code with the 5th byte.
- (4) Transfer the ID code with the 6th byte onward, starting with the 1st byte of the code.

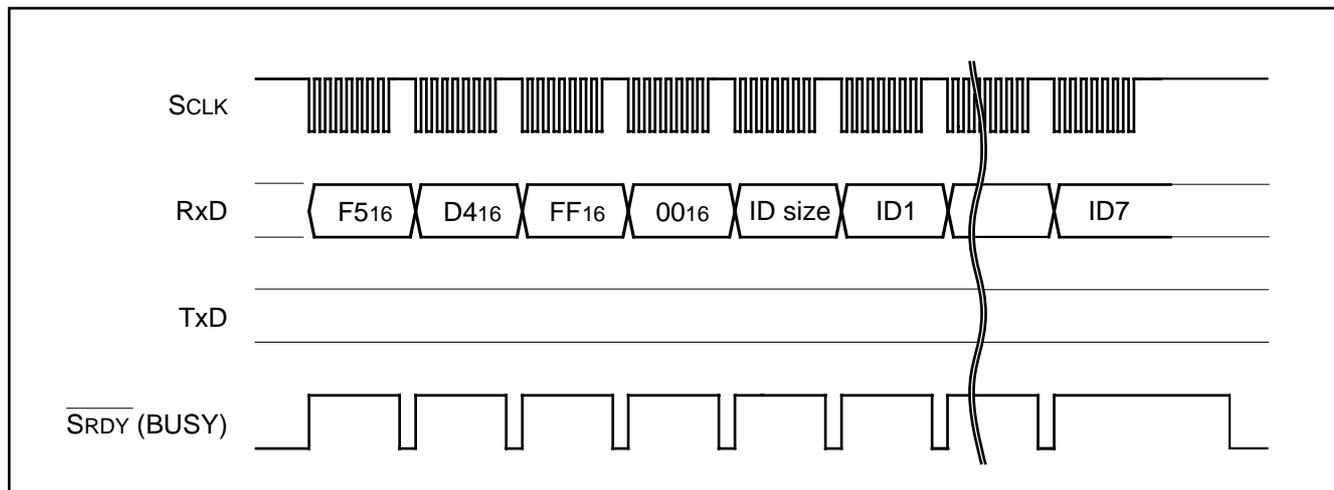


Fig. 163 Timing for ID check

●ID Code

When the flash memory is not blank, the ID code sent from the serial programmer and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the serial programmer is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses FFD4<sub>16</sub> to FFDA<sub>16</sub>. Write a program into the flash memory, which already has the ID code set for these addresses.

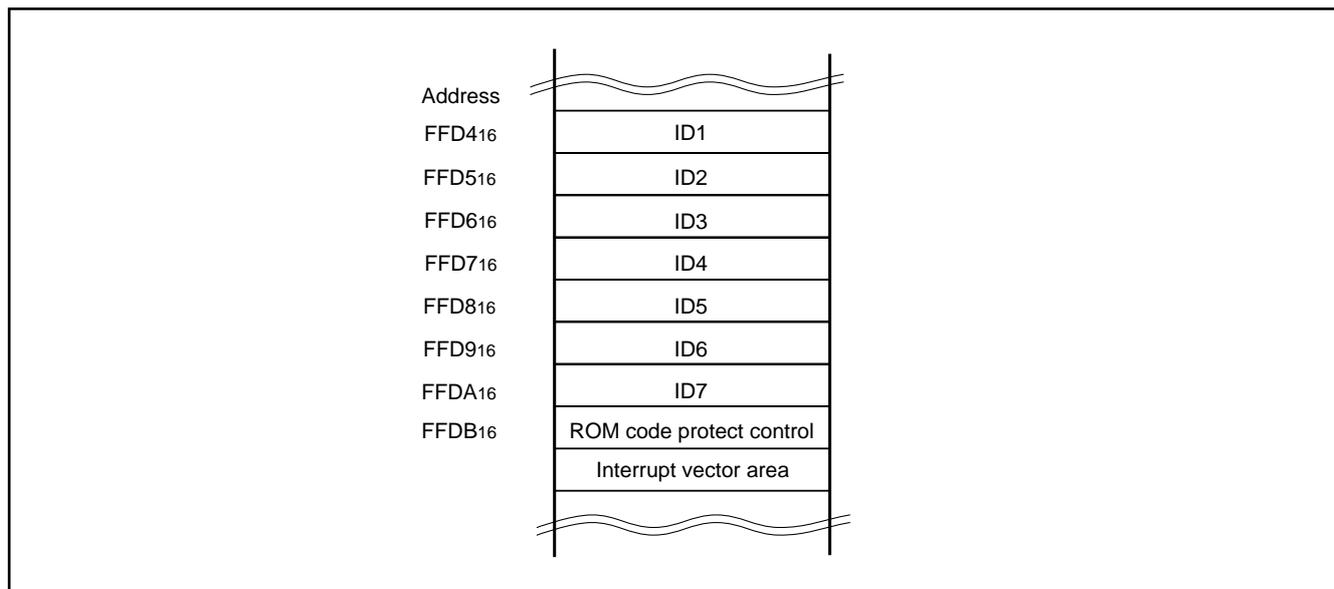


Fig. 164 ID code storage addresses

### ●Status Register (SRD)

The status register indicates operating status of the flash memory and status such as whether an erase operation or a program ended successfully or in error. It can be read by writing the read status register command (70<sub>16</sub>). Also, the status register is cleared by writing the clear status register command (50<sub>16</sub>).

Table 14 lists the definition of each status register bit. After releasing the reset, the status register becomes "80<sub>16</sub>".

### •Sequencer status (SR7)

The sequencer status indicates the operating status of the flash memory.

After power-on and recover from deep power down mode, the sequencer status is set to "1" (ready).

This status bit is set to "0" (busy) during write or erase operation and is set to "1" upon completion of these operations.

### •Erase status (SR5)

The erase status indicates the operating status of erase operation. If an erase error occurs, it is set to "1". When the erase status is cleared, it is set to "0".

### •Program status (SR4)

The program status indicates the operating status of write operation. If a write error occurs, it is set to "1". When the program status is cleared, it is set to "0".

Table 14 Status register (SRD)

SRD0 bits	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Sequencer status	Ready	Busy
SR6 (bit6)	Reserved	-	-
SR5 (bit5)	Erase status	Terminated in error	Terminated normally
SR4 (bit4)	Program status	Terminated in error	Terminated normally
SR3 (bit3)	Reserved	-	-
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

### ●Status Register 1 (SRD1)

The status register 1 indicates the status of serial communications, results from ID checks and results from check sum comparisons. It can be read after the SRD by writing the read status register command (70<sub>16</sub>). Also, status register 1 is cleared by writing the clear status register command (50<sub>16</sub>).

Table 15 lists the definition of each status register 1 bit. This register becomes "00<sub>16</sub>" when power is turned on and the flag status is maintained even after the reset.

### •Boot update completed bit (SR15)

This flag indicates whether the control program was downloaded to the RAM or not, using the download function.

### •Check sum consistency bit (SR12)

This flag indicates whether the check sum matches or not when a program, is downloaded for execution using the download function.

### •ID check completed bits (SR11 and SR10)

These flags indicate the result of ID checks. Some commands cannot be accepted without an ID check.

### •Data reception time out (SR9)

This flag indicates when a time out error is generated during data reception. If this flag is attached during data reception, the received data is discarded and the MCU returns to the command wait state.

**Table 15 Status register 1 (SRD1)**

SRD1 bits	Status name	Definition	
		"1"	"0"
SR15 (bit7)	Boot update completed bit	Update completed	Not Update
SR14 (bit6)	Reserved	-	-
SR13 (bit5)	Reserved	-	-
SR12 (bit4)	Checksum match bit	Match	Mismatch
SR11 (bit3)	ID check completed bits	00	Not verified
SR10 (bit2)		01	Verification mismatch
		10	Reserved
		11	Verified
SR9 (bit1)	Data reception time out	Time out	Normal operation
SR8 (bit0)	Reserved	-	-

## Full Status Check

Results from executed erase and program operations can be known by running a full status check. Figure 165 shows a flowchart of the full status check and explains how to remedy errors which occur.

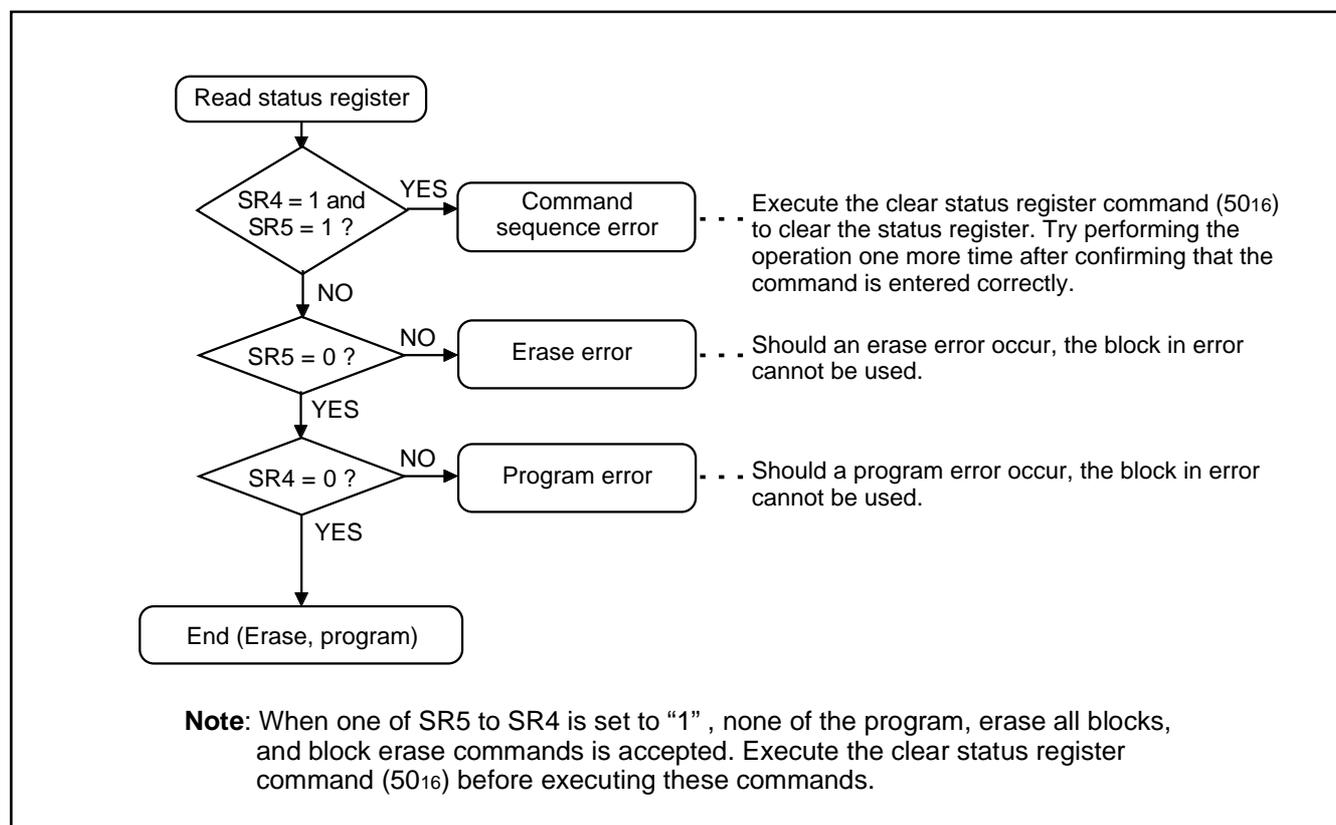


Fig. 165 Full status check flowchart and remedial procedure for errors

### Example Circuit Application for Standard Serial I/O Mode

Figure 166 shows a circuit application for the standard serial I/O mode. Control pins will vary according to a programmer, therefore see a programmer manual for more information.

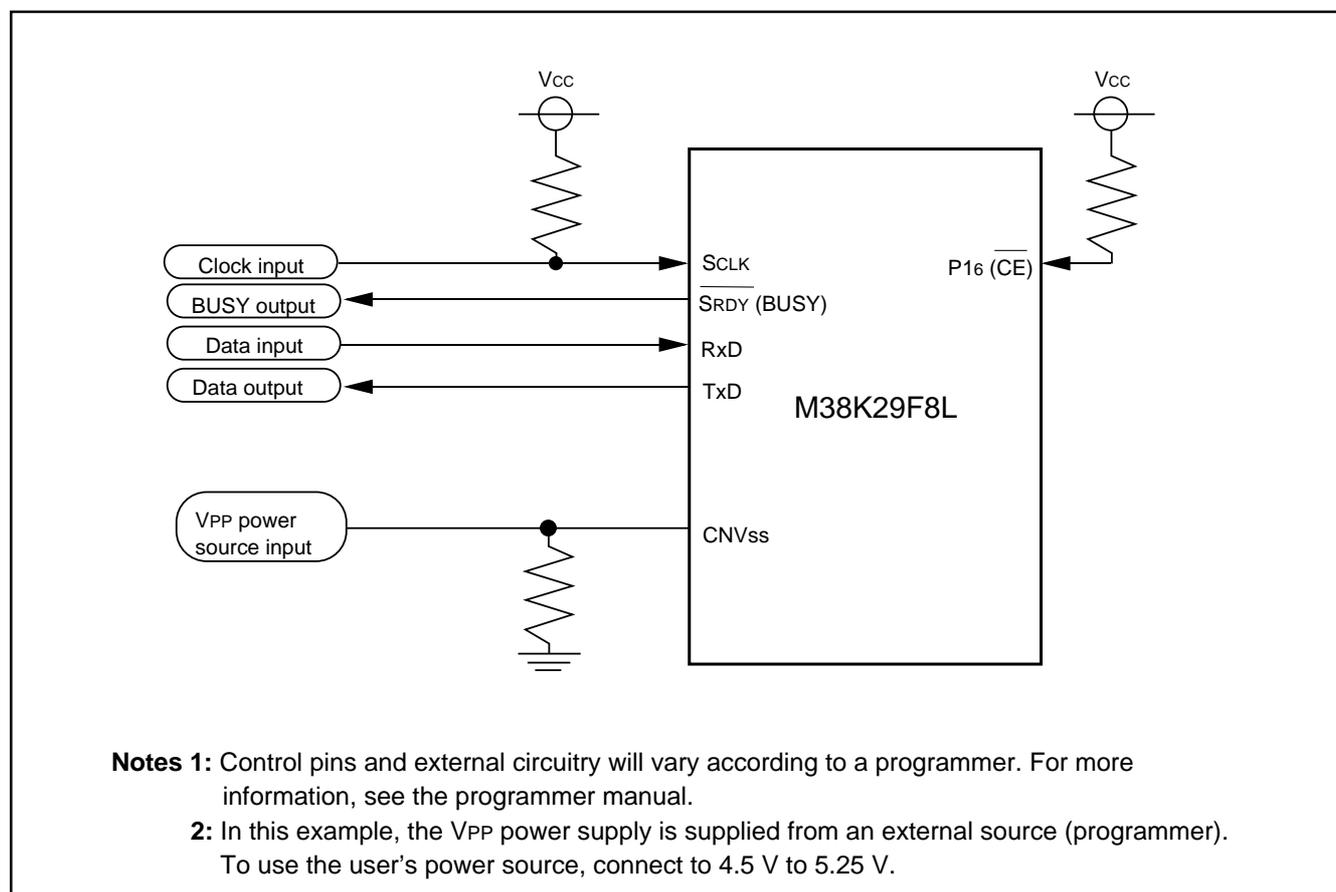


Fig. 166 Example circuit application for standard serial I/O mode

## NOTES ON PROGRAMMING

### Processor Status Register

The contents of the processor status register (PS) after a reset are undefined, except for the interrupt disable flag (I) which is "1." After a reset, initialize flags which affect program execution. In particular, it is essential to initialize the index X mode (T) and the decimal mode (D) flags because of their effect on calculations.

### Interrupts

The contents of the interrupt request bits do not change immediately after they have been written. After writing to an interrupt request register, execute at least one instruction before performing a BBC or BBS instruction.

### Decimal Calculations

- To calculate in decimal notation, set the decimal mode flag (D) to "1", then execute an ADC or SBC instruction. After executing an ADC or SBC instruction, execute at least one instruction before executing a SEC, CLC, or CLD instruction.
- In decimal mode, the values of the negative (N), overflow (V), and zero (Z) flags are invalid.

### Timers

- When n (0 to 255) is written to a timer latch, the frequency division ratio is  $1/(n+1)$ .
- When a count source of timer X is switched, stop a count of timer X.

### Multiplication and Division Instructions

- The index X mode (T) and the decimal mode (D) flags do not affect the MUL and DIV instruction.
- The execution of these instructions does not change the contents of the processor status register.

### Ports

The contents of the port direction registers cannot be read. The following cannot be used:

- The data transfer instruction (LDA, etc.)
- The operation instruction when the index X mode flag (T) is "1"
- The addressing mode which uses the value of a direction register as an index
- The bit-test instruction (BBC or BBS, etc.) to a direction register
- The read-modify-write instructions (ROR, CLB, or SEB, etc.) to a direction register.

Use instructions such as LDM and STA, etc., to set the port direction registers.

### A/D Converter

The comparator uses capacitive coupling amplifier whose charge will be lost if the clock frequency is too low.

Therefore, make sure that  $f(\text{system clock})$  in the middle/high-speed mode is at least on 500 kHz during an A/D conversion.

Do not execute the STP or WIT instruction during an A/D conversion.

### Instruction Execution Time

The instruction execution time is obtained by multiplying the frequency of the internal clock  $\phi$  by the number of cycles needed to execute an instruction. However, When using the USB function or EXB function, an occurrence of one-wait due to the multichannel RAM will double an internal clock  $\phi$  cycle.

### Definition of A/D Conversion Accuracy

The A/D conversion accuracy is defined below (refer to Figure 167).

#### •Relative accuracy

##### ① Zero transition voltage (V<sub>OT</sub>)

This means an analog input voltage when the actual A/D conversion output data changes from “0” to “1.”

##### ② Full-scale transition voltage (V<sub>FST</sub>)

This means an analog input voltage when the actual A/D conversion output data changes from “1023” to “1022.”

##### ③ Non-linearity error

This means a deviation from the line between V<sub>OT</sub> and V<sub>FST</sub> of a converted value between V<sub>OT</sub> and V<sub>FST</sub>.

##### ④ Differential non-linearity error

This means a deviation from the input potential difference required to change a converted value between V<sub>OT</sub> and V<sub>FST</sub> by 1 LSB of the 1 LSB at the relative accuracy.

#### •Absolute accuracy

This means a deviation from the ideal characteristics between 0 to V<sub>REF</sub> of actual A/D conversion characteristics.

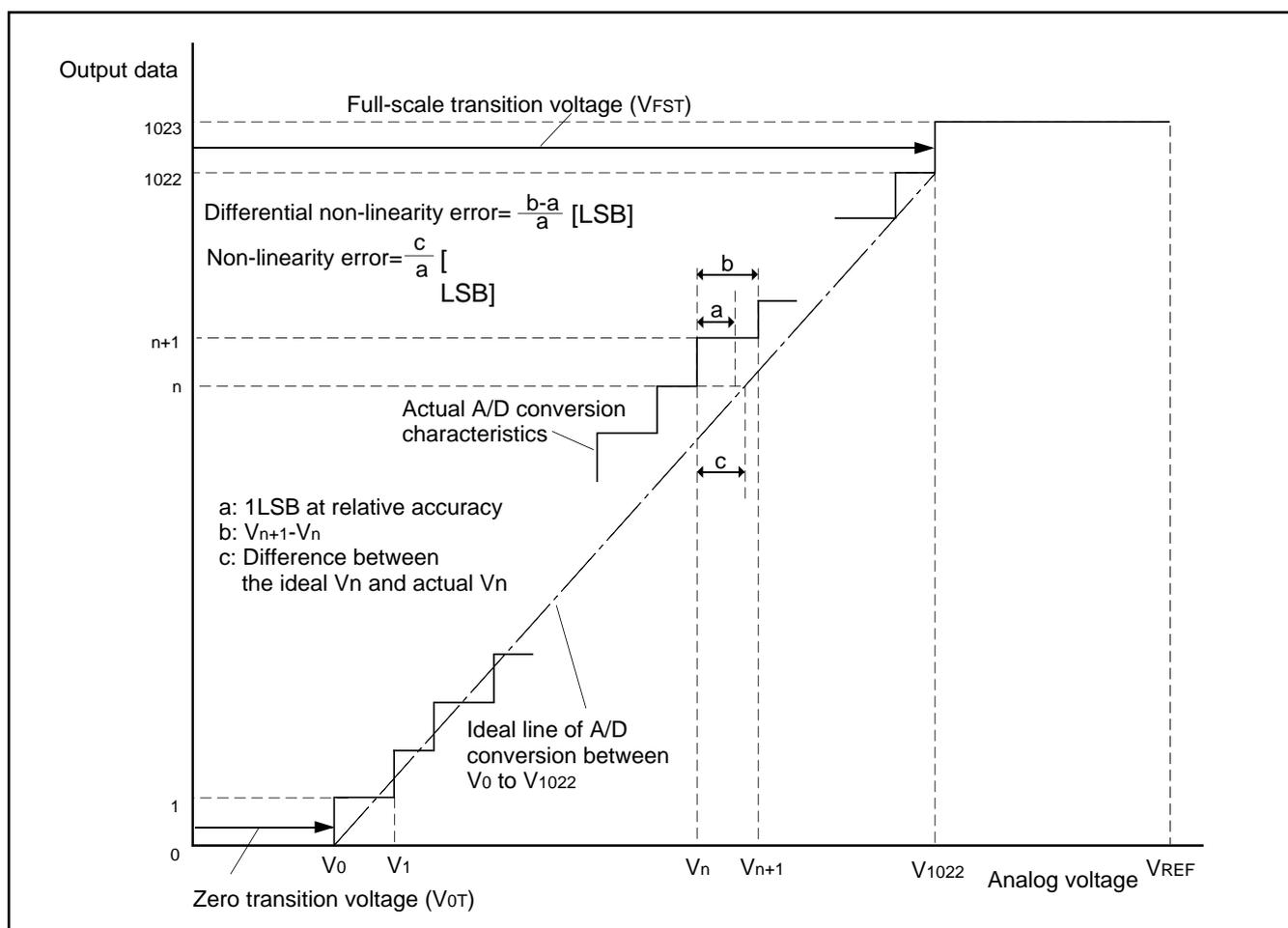


Fig. 167 Definition of A/D conversion accuracy

V<sub>n</sub>: Analog input voltage when the output data changes from “n” to “n + 1” (n = 0 to 1022)

- 1 LSB at relative accuracy  $\rightarrow \frac{V_{FST} - V_{OT}}{1022}$  (V)
- 1 LSB at absolute accuracy  $\rightarrow \frac{V_{REF}}{1024}$  (V)

## NOTES ON USAGE

### Power Source Voltage

When the power source voltage value of a microcomputer is less than the value which is indicated as the recommended operating conditions, the microcomputer does not operate normally and may perform unstable operation.

In a system where the power source voltage drops slowly when the power source voltage drops or the power supply is turned off, reset a microcomputer when the power source voltage is less than the recommended operating conditions and design a system not to cause errors to the system by this unstable operation.

### Handling of Power Source Pin

In order to avoid a latch-up occurrence, connect a capacitor suitable for high frequencies as bypass capacitor between power source pin (Vcc pin) and GND pin (Vss pin). Besides, connect the capacitor to as close as possible. For bypass capacitor which should not be located too far from the pins to be connected, a ceramic or electrolytic capacitor of 1.0  $\mu$ F is recommended.

### USB Port Pins (D0+, D0-, D1+, D1-, D2+, D2-) Treatment

- The USB specification requires a driver-impedance 28 to 44  $\Omega$ . In order to meet the USB specification impedance requirements, connect a resistor (27  $\Omega$  recommended) in series to the USB port pins.

In addition, in order to reduce the ringing and control the falling/rising timing and a crossover point, connect a capacitor between the USB port pins and the Vss pin if necessary.

The values and structure of those peripheral elements depend on the impedance characteristics and the layout of the printed circuit board. Accordingly, evaluate your system and observe waveforms before actual use and decide use of elements and the values of resistors and capacitors.

- Make sure the USB D+/D- lines do not cross any other wires. Keep a large GND area to protect the USB lines. Also, make sure you use a USB specification compliant connector for the connection.

### USBVREF pin Treatment (Noise Elimination)

- Connect a capacitor between the USBVREF pin and the Vss pin. The capacitor should have a 2.2  $\mu$ F capacitor (electrolytic capacitor) and a 0.1  $\mu$ F capacitor (ceramic type capacitor) connected in parallel.

- In Vcc = 3.0 to 3.6 V operation, connect the USBVREF pin directly to the Vcc pin in order to supply power to the USB port circuit. In addition, you will need to disable the built-in USB reference voltage circuit in this operation (set bit 4 of the USB control register to "0".) If you are using the bus powered supply in this condition, the DC-DC converter must be placed outside the MCU.

- In Vcc = 4.00 to 5.25 V operation, do not connect the external DC-DC converter to the USBVREF pin. Use the built-in USB reference voltage circuit.

### USB Communication

In applications requiring high-reliability, we recommend providing the system with protective measures such as USB function initialization by software or USB reset by the host to prevent USB communication from being terminated unexpectedly, for example due to external causes such as noise.

### Flash Memory Version

The CNVss pin is connected to the internal memory circuit block by a low-ohmic resistance, since it has the multiplexed function to be a programmable power source pin (VPP pin) as well.

To improve the noise reduction, connect a track between CNVss pin and Vss pin or Vcc pin with 1 to 10 k $\Omega$  resistance.

The mask ROM version track of CNVss pin has no operational interference even if it is connected to Vss pin or Vcc pin via a resistor.

### Electric Characteristic Differences Between Mask ROM and Flash Memory Version MCUs

There are differences in electric characteristics, operation margin, noise immunity, and noise radiation between Mask ROM and Flash Memory version MCUs due to the difference in the manufacturing processes.

When manufacturing an application system with the Flash Memory version and then switching to use of the Mask ROM version, please perform sufficient evaluations for the commercial samples of the Mask ROM version.

## DATA REQUIRED FOR MASK ORDERS

The following are necessary when ordering a mask ROM production:

1. Mask ROM Order Confirmation Form\*
2. Mark Specification Form\*
3. Data to be written to ROM, in EPROM form (three identical copies) or one floppy disk.

\* For the mask ROM confirmation and the mark specifications, refer to the "Renesas Technology Corp." Homepage (<http://www.renesas.com>).

## FUNCTIONAL DESCRIPTION SUPPLEMENT A/D Converter

A/D conversion is started by setting AD conversion completion bit to "0." During A/D conversion, internal operations are performed as follows.

1. After the start of A/D conversion, AD conversion register goes to "0016."
2. The highest-order bit of AD conversion register is set to "1," and the comparison voltage  $V_{ref}$  is input to the comparator. Then,  $V_{ref}$  is compared with analog input voltage  $V_{IN}$ .
3. As a result of comparison, when  $V_{ref} < V_{IN}$ , the highest-order bit of AD conversion register becomes "1." When  $V_{ref} > V_{IN}$ , the highest-order bit becomes "0."

By repeating the above operations up to the lowest-order bit of the AD conversion register, an analog value converts into a digital value.

A/D conversion completes at 122 clock cycles (15.25  $\mu$ s at system clock = 8 MHz, Through mode) after it is started, and the result of the conversion is stored into the AD conversion register.

Concurrently with the completion of A/D conversion, A/D conversion interrupt request occurs, so that the AD conversion interrupt request bit is set to "1."

**Table 16 Relative formula for a reference voltage  $V_{REF}$  of A/D converter and  $V_{ref}$**

When $n = 0$	$V_{ref} = 0$
When $n = 1$ to 1023	$V_{ref} = \frac{V_{REF}}{1024} \times n$

n: Value of A/D converter (decimal numeral)

**Table 17 Change of AD conversion register during A/D conversion**

	Change of AD conversion register	Value of comparison voltage ( $V_{ref}$ )
<b>At start of conversion</b>	0 0 0 0 0 0 0 0 0 0 0	0
<b>First comparison</b>	1 0 0 0 0 0 0 0 0 0 0	$\frac{V_{REF}}{2}$
<b>Second comparison</b>	*1 1 0 0 0 0 0 0 0 0 0	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4}$
<b>Third comparison</b>	*1 *2 1 0 0 0 0 0 0 0 0	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} \pm \frac{V_{REF}}{8}$
	≈	≈
<b>After completion of tenth comparison</b>	A result of A/D conversion *1 *2 *3 *4 *5 *6 *7 *8 *9 *10	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} \pm \dots \pm \frac{V_{REF}}{1024}$

\*1–\*10: A result of the first comparison to the tenth comparison

Figure 168 shows the A/D conversion equivalent circuit, and Figure 169 shows the A/D conversion timing chart.

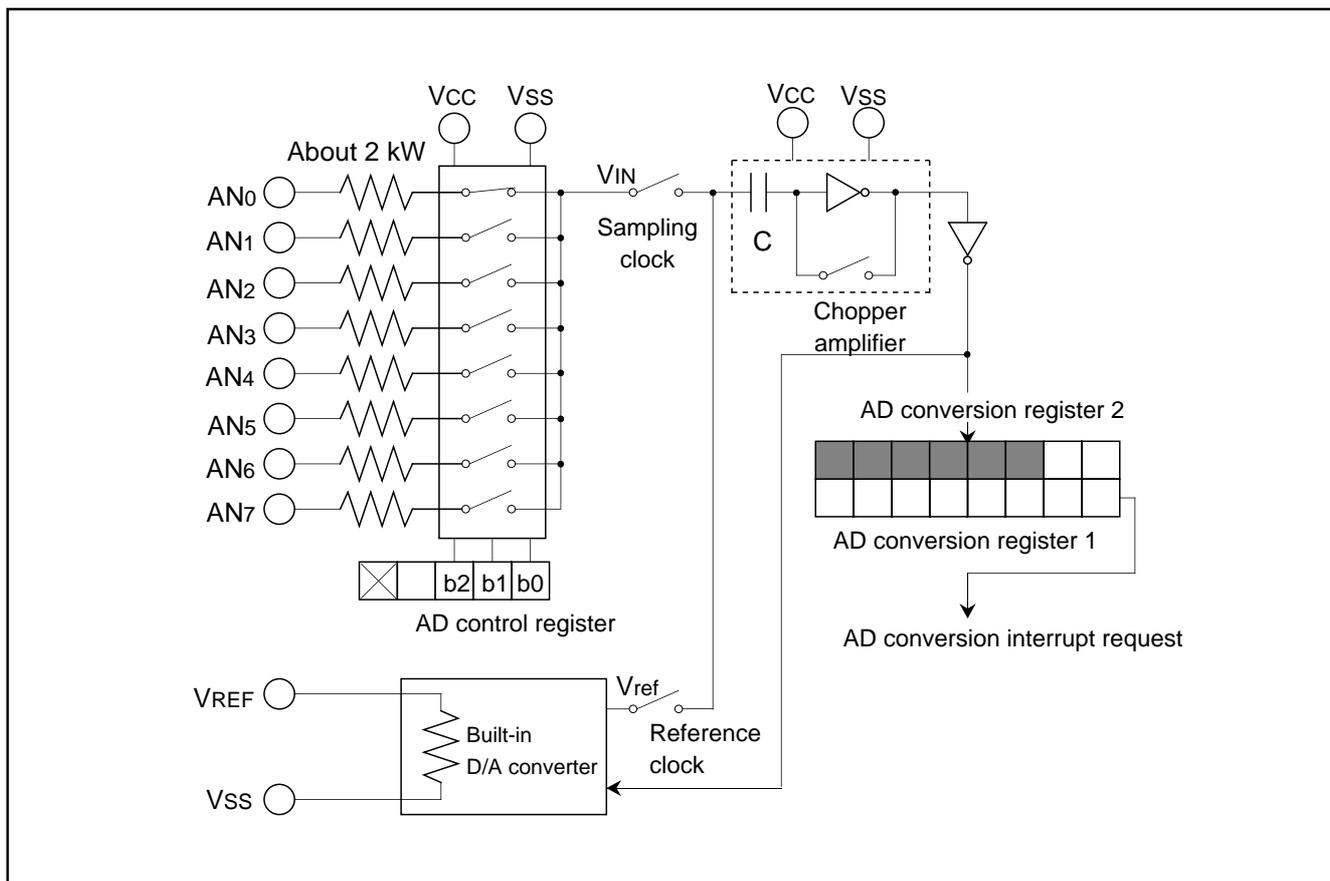


Fig. 168 A/D conversion equivalent circuit

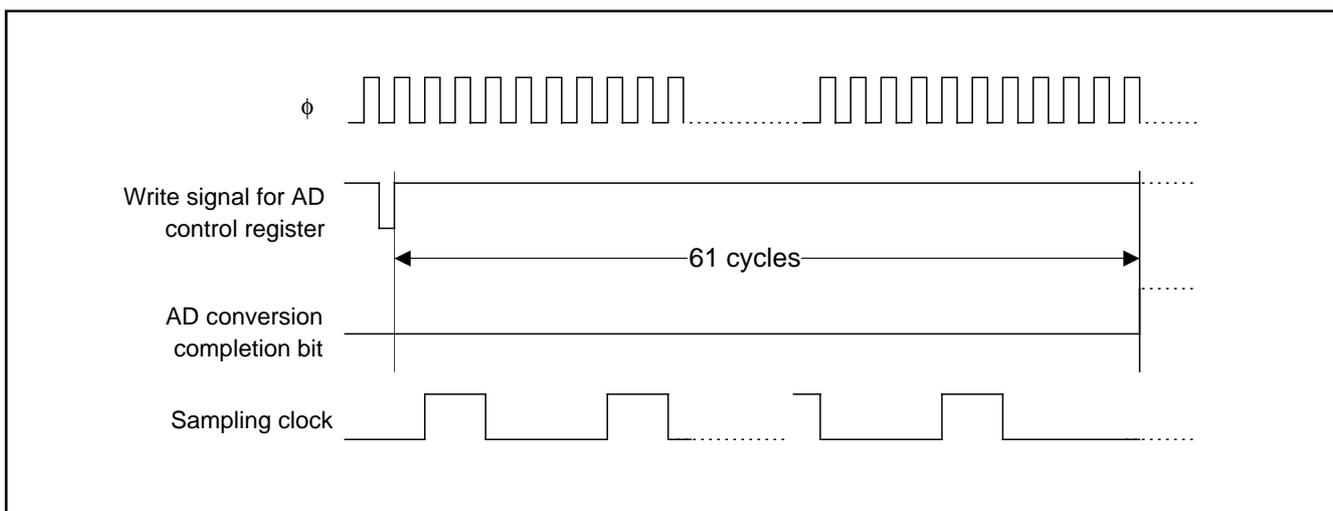


Fig. 169 A/D conversion timing chart

# CHAPTER 2

---

## APPLICATION

- 2.1 I/O port
- 2.2 Interrupt
- 2.3 Timer
- 2.4 Serial I/O
- 2.5 USB function
- 2.6 HUB function
- 2.7 External bus interface (EXB)
- 2.8 A/D converter
- 2.9 Watchdog timer
- 2.10 Reset
- 2.11 Frequency synthesizer (PLL)
- 2.12 Clock generating circuit
- 2.13 Standby function
- 2.14 Flash memory

## 2.1 I/O port

This paragraph explains the registers setting method and the notes related to the I/O ports.

### 2.1.1 Memory map

0000 <sub>16</sub>	Port P0 (P0)
0001 <sub>16</sub>	Port P0 direction register (P0D)
0002 <sub>16</sub>	Port P1 (P1)
0003 <sub>16</sub>	Port P1 direction register (P1D)
0004 <sub>16</sub>	Port P2 (P2)
0005 <sub>16</sub>	Port P2 direction register (P2D)
0006 <sub>16</sub>	Port P3 (P3)
0007 <sub>16</sub>	Port P3 direction register (P3D)
0008 <sub>16</sub>	Port P4 (P4)
0009 <sub>16</sub>	Port P4 direction register (P4D)
000A <sub>16</sub>	Port P5 (P5)
000B <sub>16</sub>	Port P5 direction register (P5D)
000C <sub>16</sub>	Port P6 (P6)
000D <sub>16</sub>	Port P6 direction register (P6D)
⋮	⋮
0FF0 <sub>16</sub>	Port P0 pull-up control register (PULL0)
⋮	⋮
0FF2 <sub>16</sub>	Port P5 pull-up control register (PULL5)

Fig. 2.1.1 Memory map of registers related to I/O port

## 2.1.2 Related registers

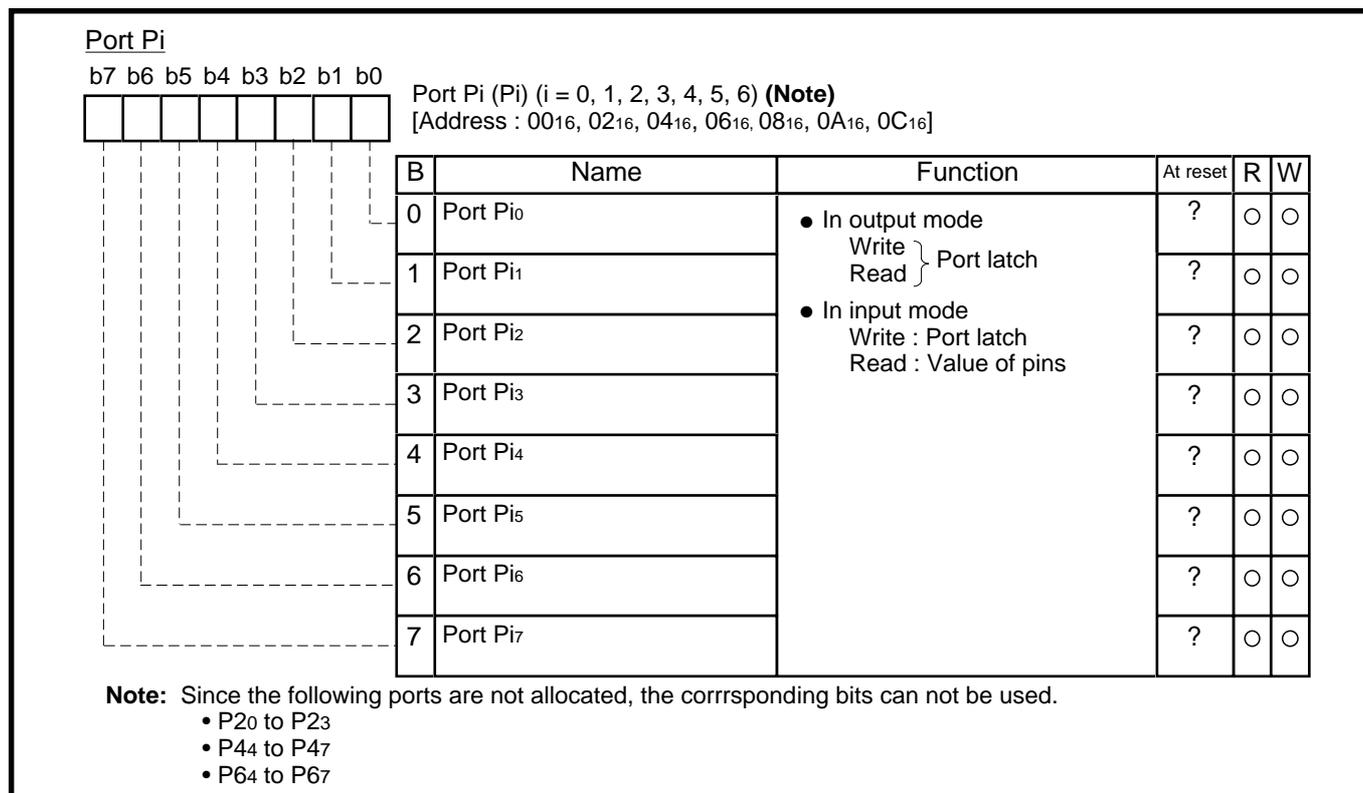


Fig. 2.1.2 Structure of Port Pi (i = 0 to 6)

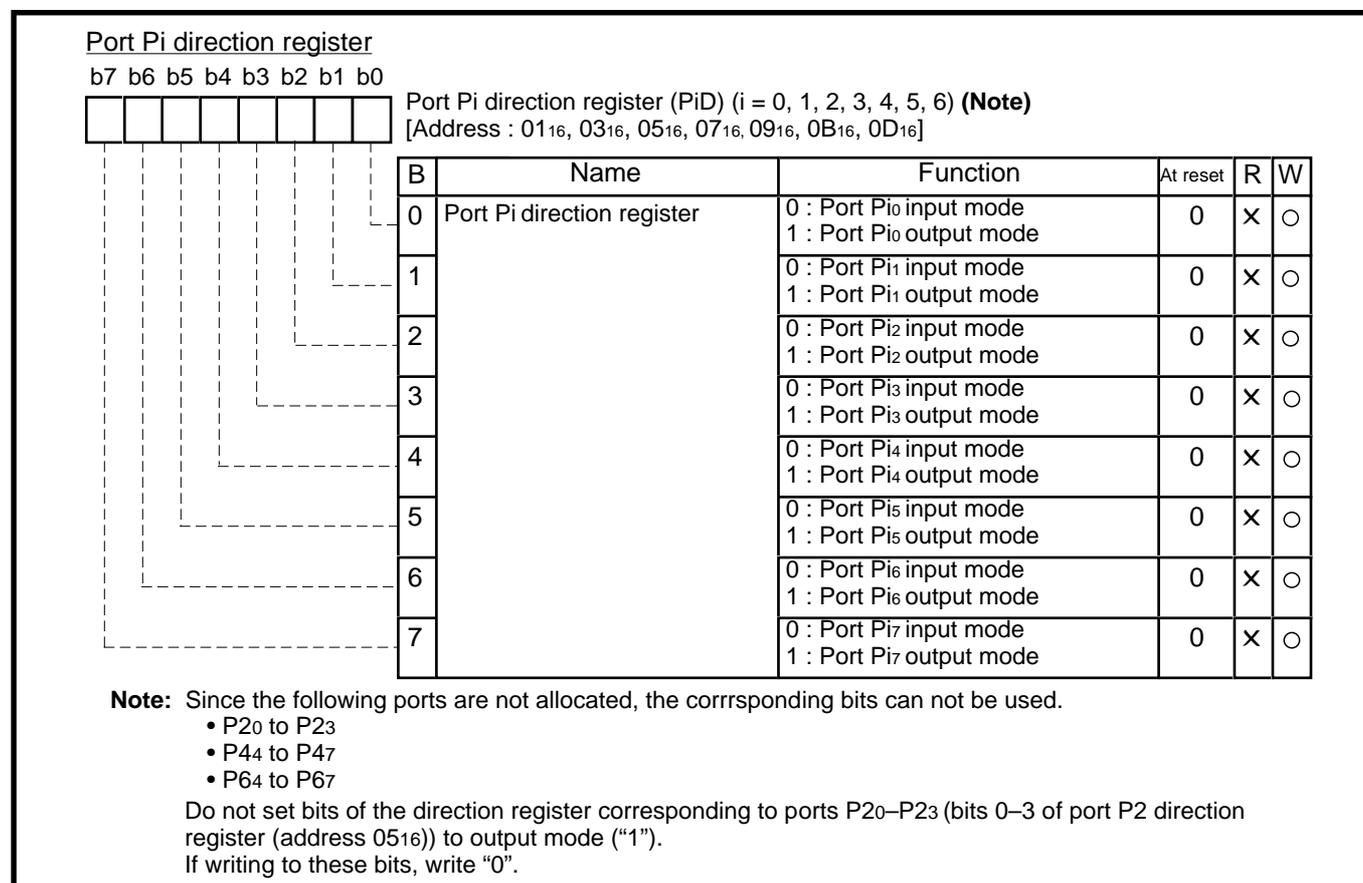


Fig. 2.1.3 Structure of Port Pi direction register (i = 0 to 6)

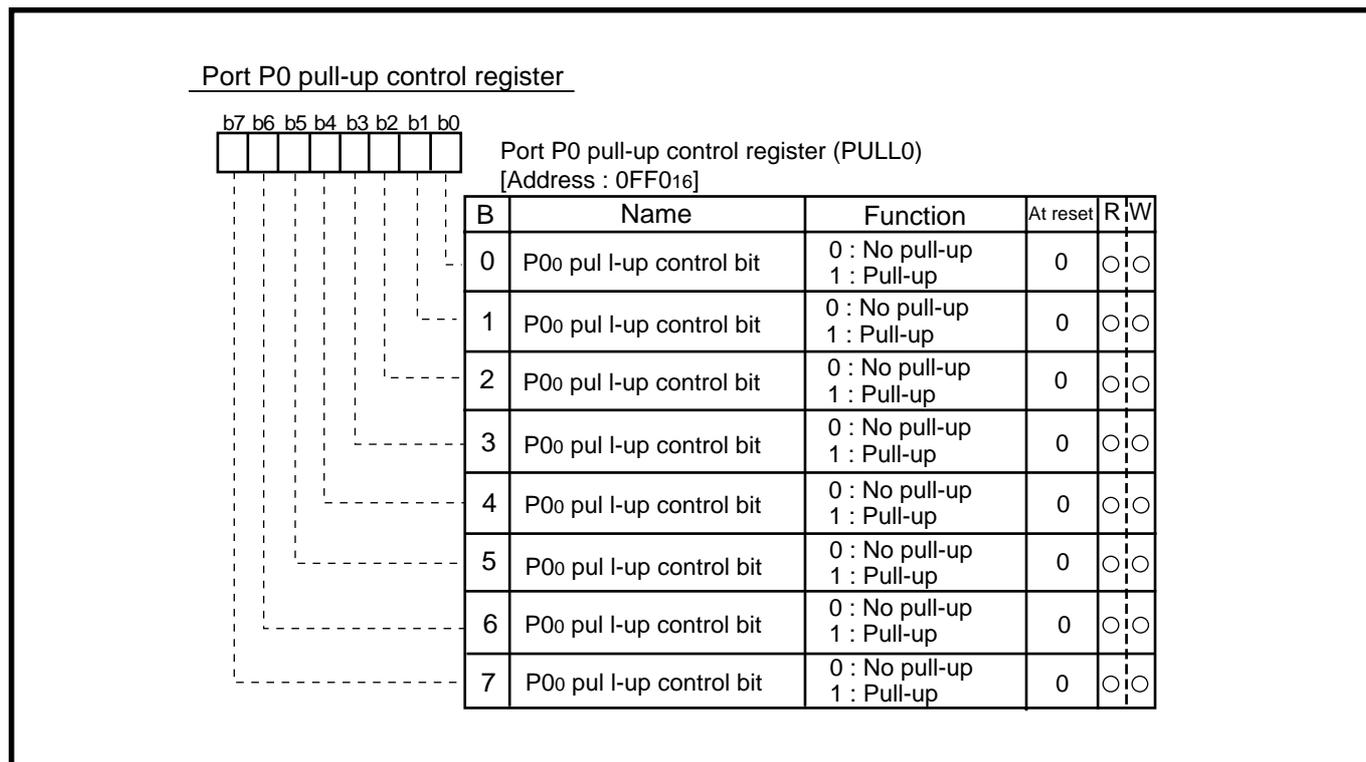


Fig. 2.1.4 Structure of Port P0 pull-up control register

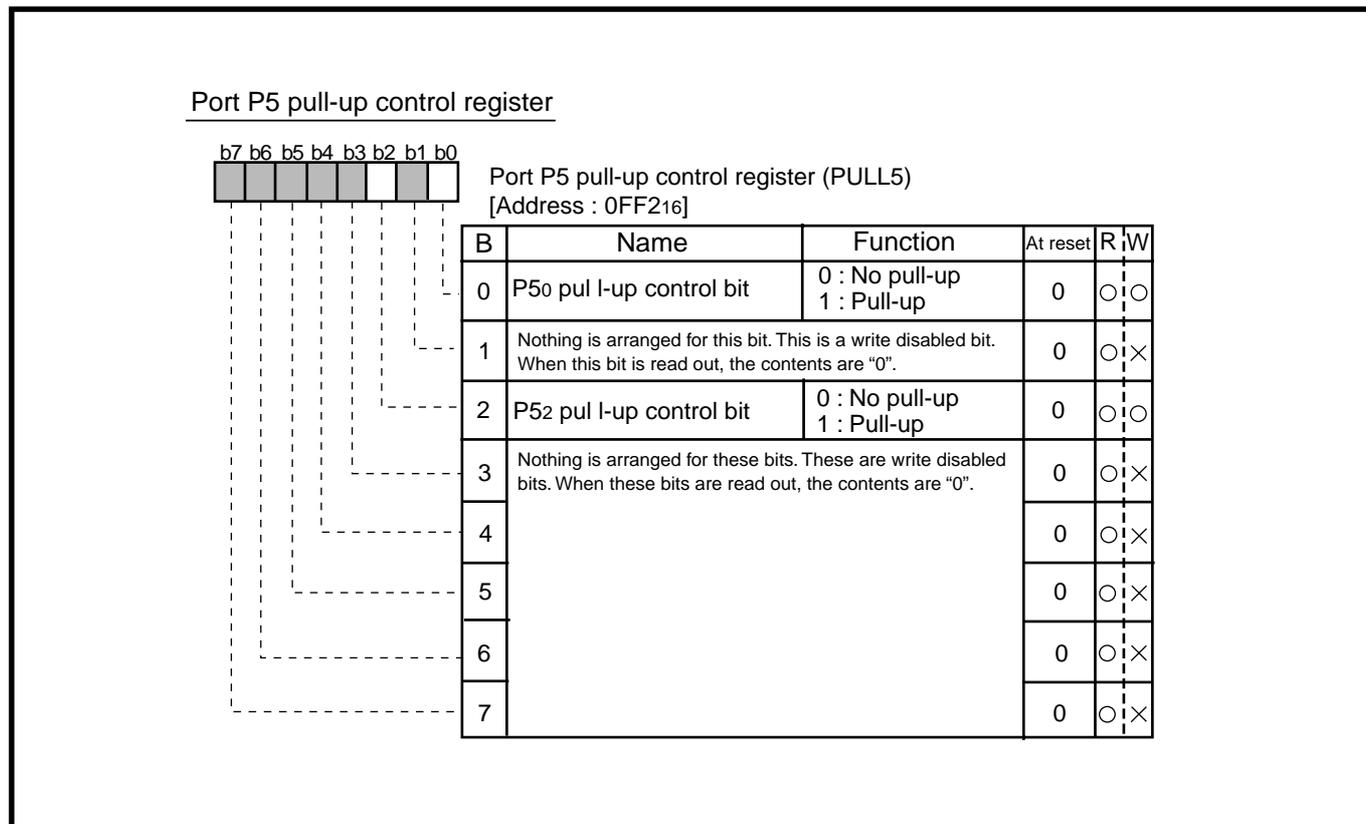


Fig. 2.1.5 Structure of Port P5 pull-up control register

### 2.1.3 Handling of unused pins

**Table 2.1.1 Handling of unused pins**

Pins/Ports name	Handling
P0, P1, P2, P3, P4, P5, P6	<ul style="list-style-type: none"> <li>•Set to the input mode and connect each to V<sub>cc</sub> or V<sub>ss</sub> through a resistor of 1 k<math>\Omega</math> to 10 k<math>\Omega</math>.</li> <li>•Set to the output mode and open at "L" or "H" level.</li> </ul>
V <sub>REF</sub>	•Connect to V <sub>ss</sub> (GND).
X <sub>OUT</sub>	•Open, only when using an external clock.
USBV <sub>REF</sub>	•Connect to V <sub>cc</sub>
TrON	•Open
D0+, D0-, D1+, D1-, D2+, D2-	•Connect each to V <sub>ss</sub> through a resistor of 1 k $\Omega$ to 10 k $\Omega$ .

### 2.1.4 Notes on input and output pins

#### (1) Modifying output data with bit managing instruction

When the port latch of an I/O port is modified with the bit managing instruction\*<sup>1</sup>, the value of the unspecified bit may be changed.

##### ● Reason

The bit managing instructions are read-modify-write form instructions for reading and writing data by a byte unit. Accordingly, when these instructions are executed on a bit of the port latch of an I/O port, the following is executed to all bits of the port latch.

- As for a bit which is set for an input port :  
The pin state is read in the CPU, and is written to this bit after bit managing.
- As for a bit which is set for an output port :  
The bit value of the port latch is read in the CPU, and is written to this bit after bit managing.

Note the following :

- Even when a port which is set as an output port is changed for an input port, its port latch holds the output data.
- As for a bit of the port latch which is set for an input port, its value may be changed even when not specified with a bit managing instruction in case where the pin state differs from its port latch contents.

\*<sup>1</sup> bit managing instructions : **SEB**, and **CLB** instructions

## 2.1.5 Termination of unused pins

### (1) Terminate unused pins

① I/O ports :

- Set the I/O ports for the input mode and connect them to Vcc or Vss through each resistor of 1 k $\Omega$  to 10 k $\Omega$ . With regard to ports which can select the built-in pull-up resistor, the built-in pull-up resistor can be used.

Set the I/O ports for the output mode and open them at “L” or “H”.

- When opening them in the output mode, the input mode of the initial status remains until the mode of the ports is switched over to the output mode by the program after reset. Thus, the potential at these pins is undefined and the power source current may increase in the input mode. With regard to an effects on the system, thoroughly perform system evaluation on the user side.
- Since the direction register setup may be changed because of a program runaway or noise, set direction registers by program periodically to increase the reliability of program.

### (2) Termination remarks

① I/O ports :

Do not open in the input mode.

● Reason

- The power source current may increase depending on the first-stage circuit.
- An effect due to noise may be easily produced as compared with proper termination shown in (1).

② I/O ports :

When setting for the input mode, do not connect to Vcc or Vss directly.

● Reason

If the direction register setup changes for the output mode because of a program runaway or noise, a short circuit may occur between a port and Vcc (or Vss).

③ I/O ports :

When setting for the input mode, do not connect multiple ports in a lump to Vcc or Vss through a resistor.

● Reason

If the direction register setup changes for the output mode because of a program runaway or noise, a short circuit may occur between ports.

- At the termination of unused pins, perform wiring at the shortest possible distance (20 mm or less) from microcomputer pins.

## 2.2 Interrupt

This paragraph explains the registers setting method and the notes related to the interrupt.

### 2.2.1 Memory map

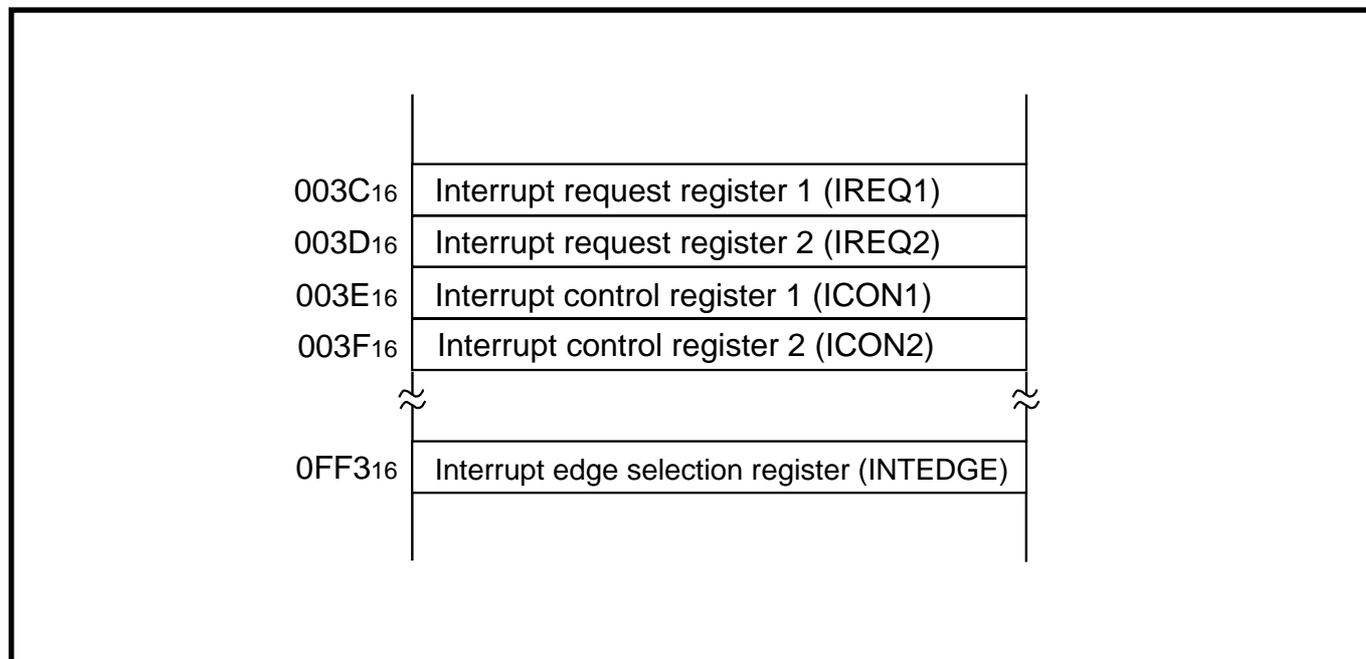


Fig. 2.2.1 Memory map of registers related to interrupt

### 2.2.2 Related registers

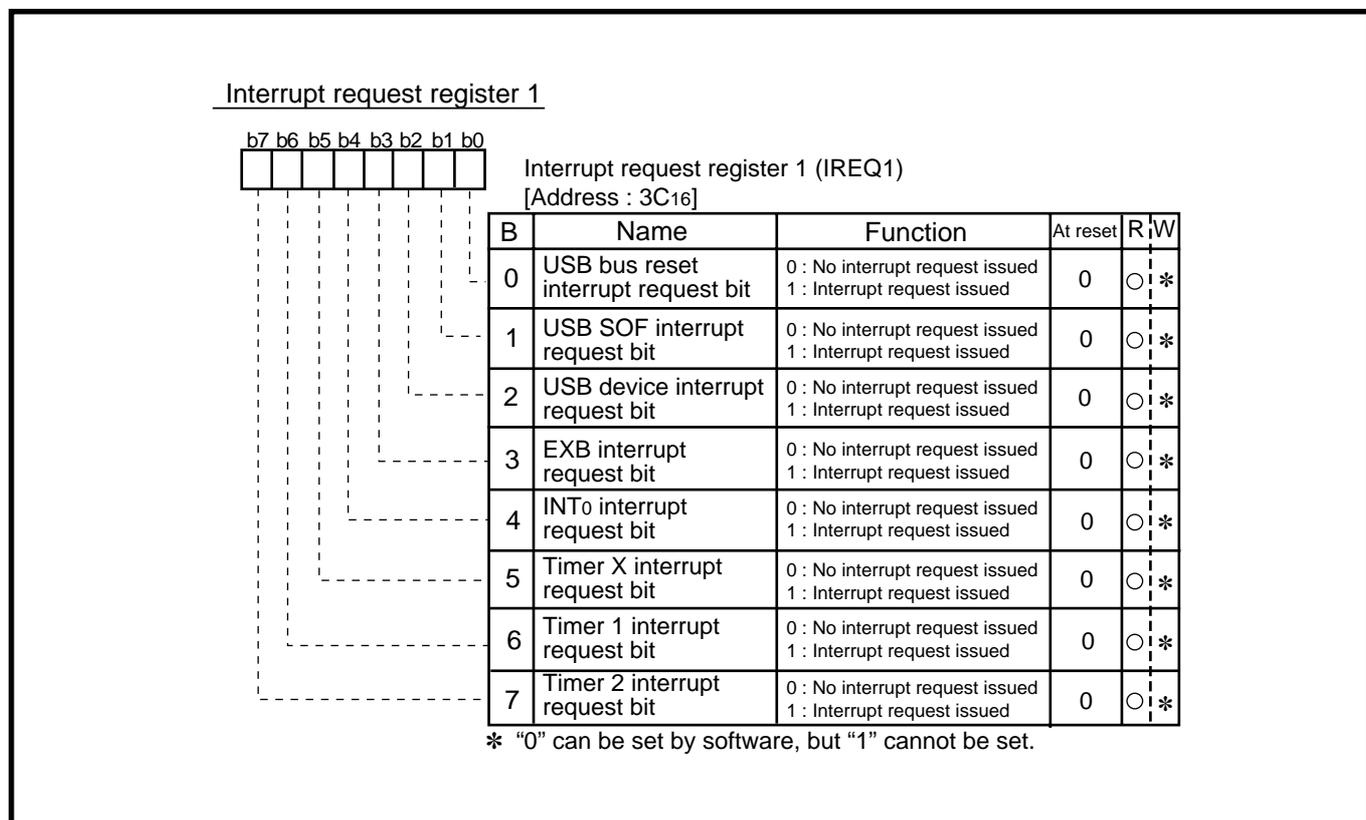


Fig. 2.2.2 Structure of Interrupt request register 1

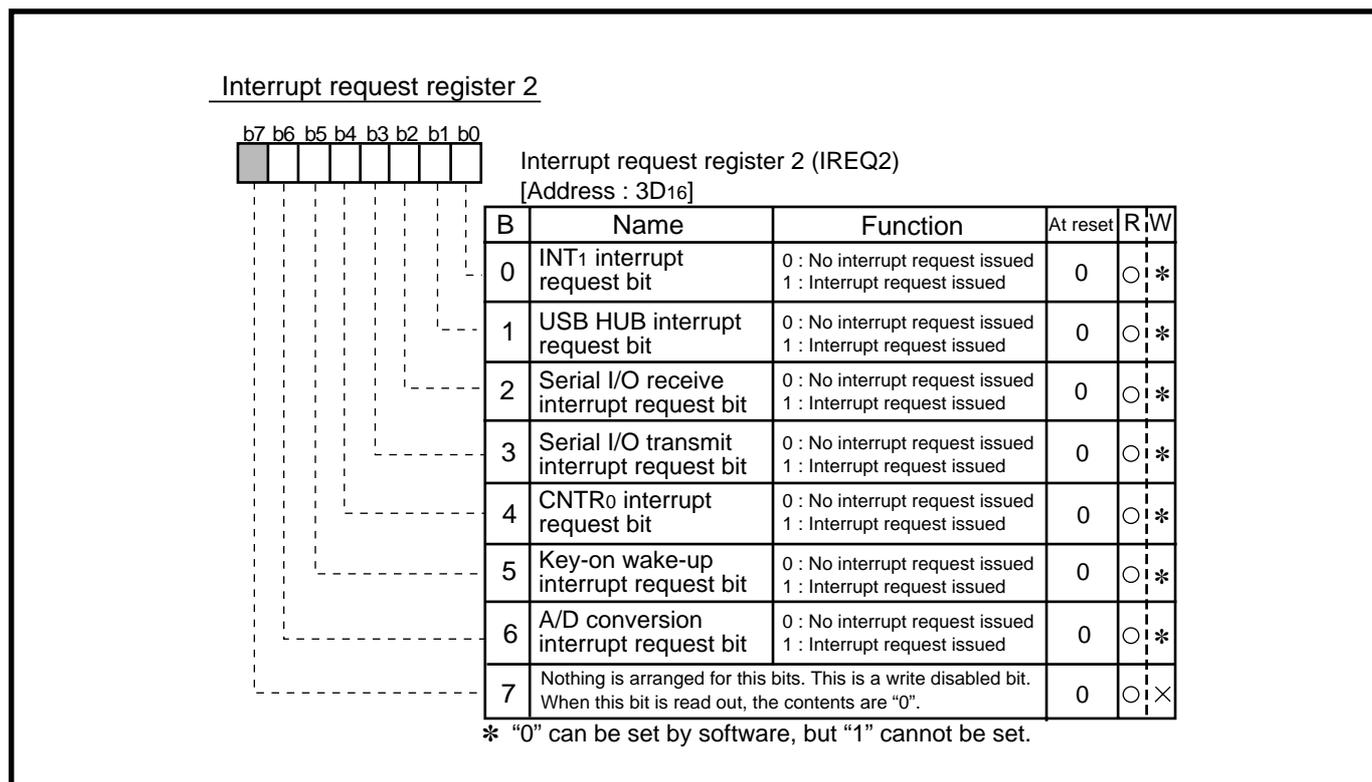


Fig. 2.2.3 Structure of Interrupt request register 2

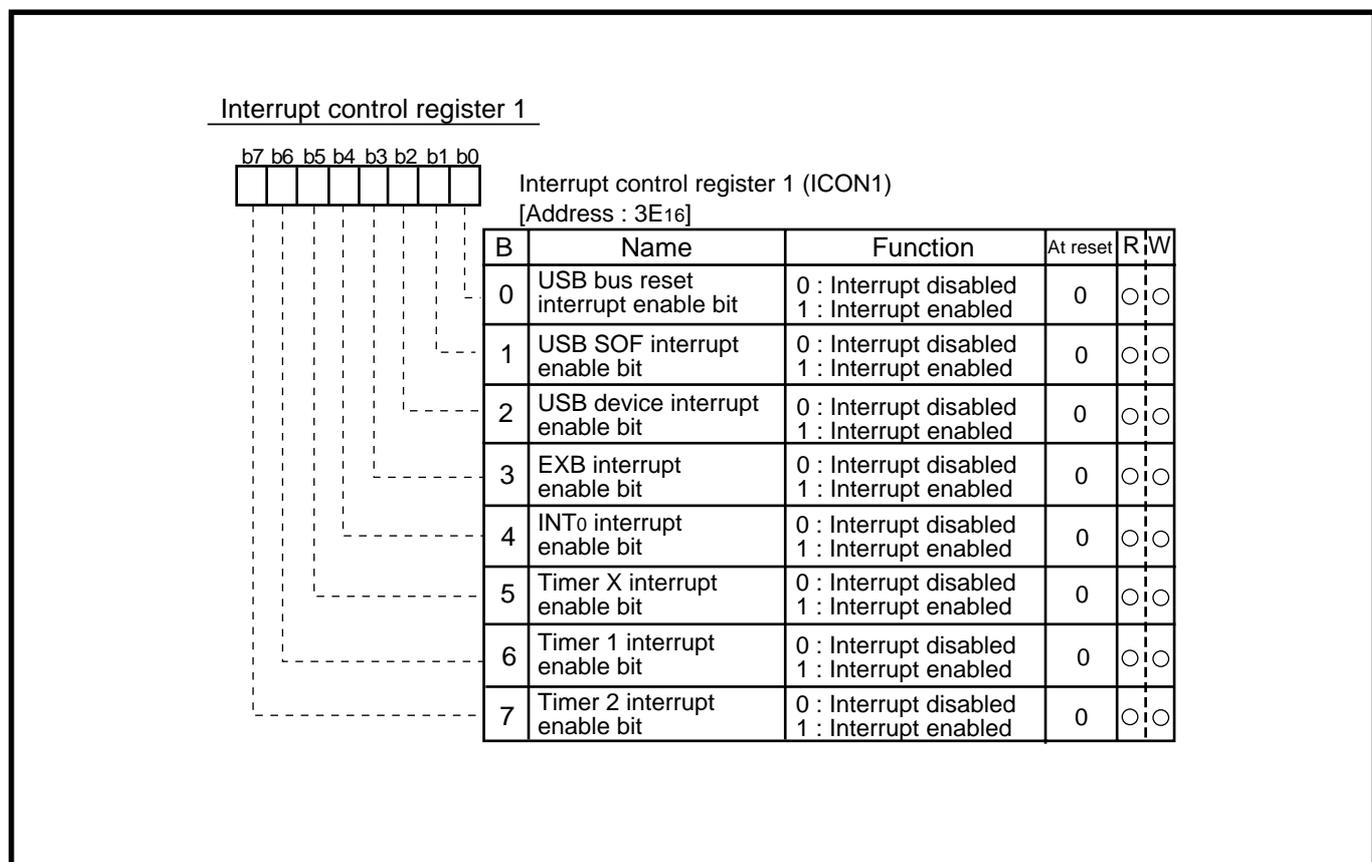


Fig. 2.2.4 Structure of Interrupt control register 1

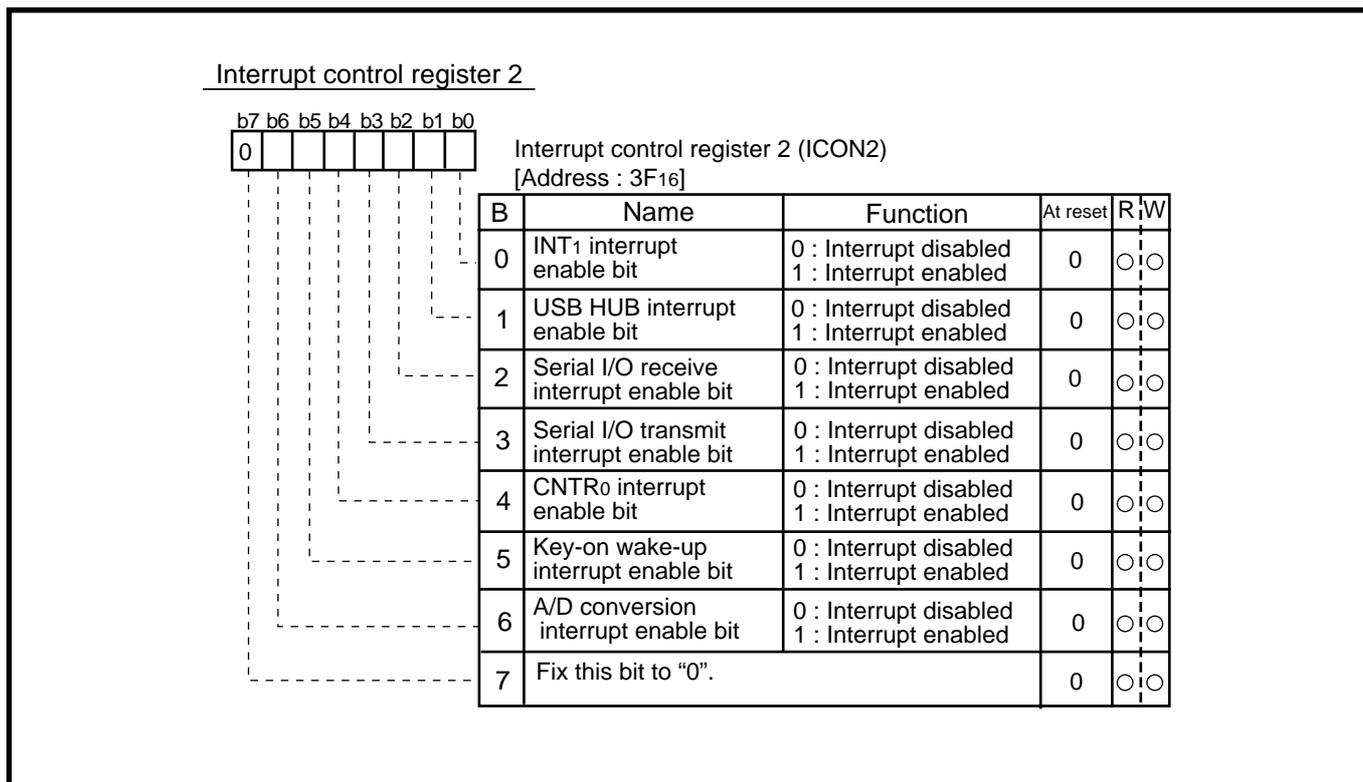


Fig. 2.2.5 Structure of Interrupt control register 2

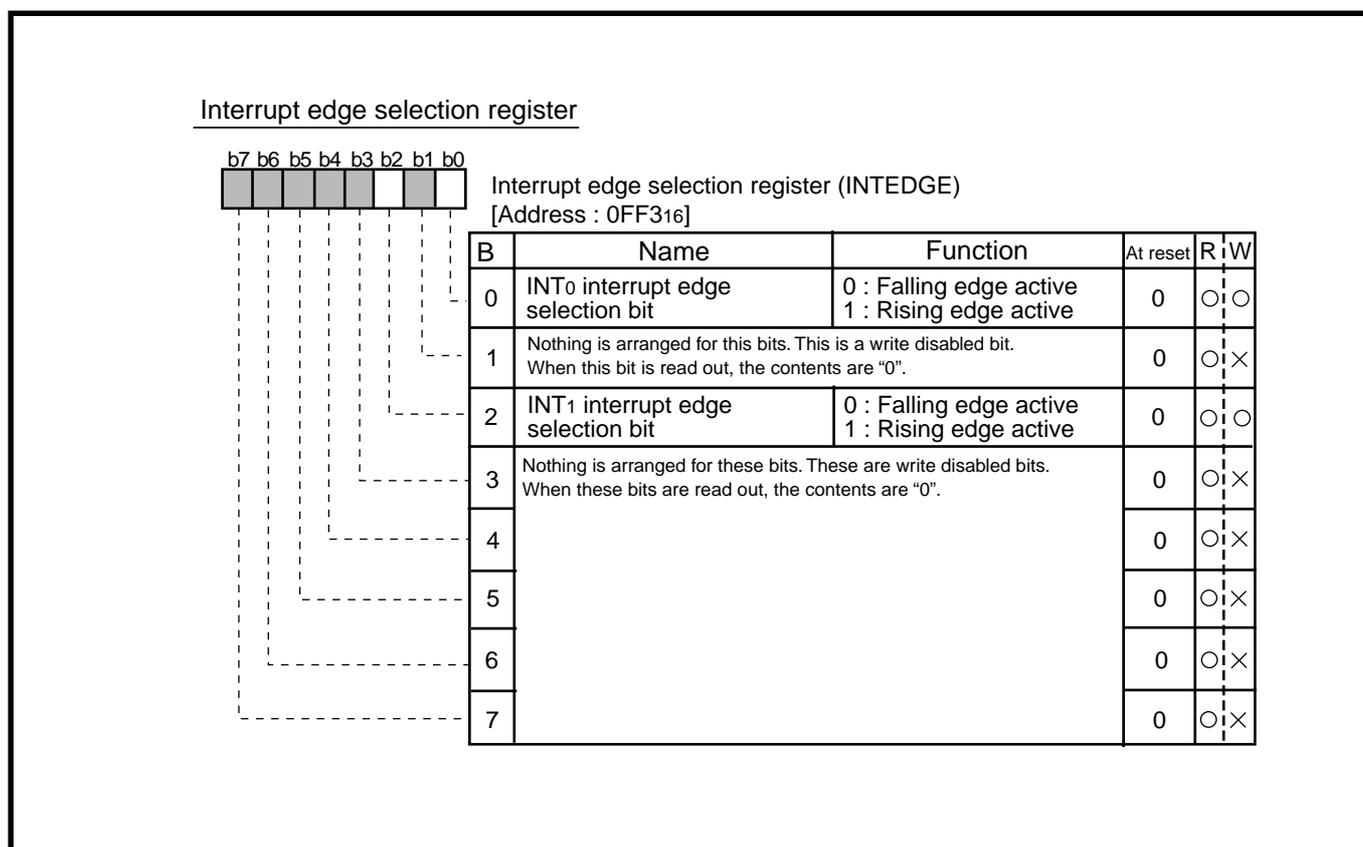


Fig. 2.2.6 Structure of Interrupt edge selection register

### 2.2.3 Interrupt source

The 38K2 group permits interrupts of 16 sources. These are vector interrupts with a fixed priority system. Accordingly, when two or more interrupt requests occur during the same sampling, the higher-priority interrupt is accepted first. This priority is determined by hardware, but a variety of priority processing can be performed by software, using an interrupt enable bit and an interrupt disable flag.

For interrupt sources, vector addresses and interrupt priority, refer to Table 2.2.1.

**Table 2.2.1 Interrupt sources, vector addresses and priority of 38K2 group**

Interrupt Source	Priority	Vector Addresses (Note 1)		Interrupt Request Generating Conditions	Remarks
		High	Low		
Reset (Note 2)	1	FFFD <sub>16</sub>	FFFC <sub>16</sub>	At reset	Non-maskable
USB bus reset	2	FFFB <sub>16</sub>	FFFA <sub>16</sub>	At detection of USB bus reset signal (2.5 $\mu$ s interval SE0)	Valid when USB is selected
USB SOF	3	FFF9 <sub>16</sub>	FFF8 <sub>16</sub>	At detection of USB SOF signal	Valid when USB is selected
USB device	4	FFF7 <sub>16</sub>	FFF6 <sub>16</sub>	At detection of resume signal (K state or SE0) or suspend signal (3 ms interval bus idle), or at completion of transaction	Valid when USB is selected
External bus	5	FFF5 <sub>16</sub>	FFF4 <sub>16</sub>	At completion of reception or transmission or at completion of DMA transmission	Valid when external bus is selected
INT <sub>0</sub>	6	FFF3 <sub>16</sub>	FFF2 <sub>16</sub>	At detection of either rising or falling edge of INT <sub>0</sub> input	External interrupt (active edge selectable)
Timer X	7	FFF1 <sub>16</sub>	FFF0 <sub>16</sub>	At timer X underflow	
Timer 1	8	FFEF <sub>16</sub>	FFEE <sub>16</sub>	At timer 1 underflow	STP release timer underflow
Timer 2	9	FFED <sub>16</sub>	FFEC <sub>16</sub>	At timer 2 underflow	
INT <sub>1</sub>	10	FFEB <sub>16</sub>	FFEA <sub>16</sub>	At detection of either rising or falling edge of INT <sub>1</sub> input	External interrupt (active edge selectable)
USB HUB	11	FFE9 <sub>16</sub>	FFE8 <sub>16</sub>	At detection of status change of USB HUB down ports	Valid when USB HUB is selected
Serial I/O reception	12	FFE7 <sub>16</sub>	FFE6 <sub>16</sub>	At completion of serial I/O data reception	Valid when serial I/O is selected
Serial I/O transmission	13	FFE5 <sub>16</sub>	FFE4 <sub>16</sub>	At completion of serial I/O data transmission	Valid when serial I/O is selected
CNTR <sub>0</sub>	14	FFE3 <sub>16</sub>	FFE2 <sub>16</sub>	At detection of either rising or falling edge of CNTR <sub>0</sub> input	External interrupt (active edge selectable)
Key-on wake up	15	FFE1 <sub>16</sub>	FFE0 <sub>16</sub>	At falling of conjunction of input level for port P0 (at input mode)	External interrupt (active edge selectable)
A/D conversion	16	FFDF <sub>16</sub>	FFDE <sub>16</sub>	At completion of A/D conversion	
BRK instruction	17	FFDD <sub>16</sub>	FFDC <sub>16</sub>	At BRK instruction execution	Non-maskable software interrupt

**Notes 1:** Vector addresses contain interrupt jump destination addresses.

**2:** Reset function in the same way as an interrupt with the highest priority.

### 2.2.4 Interrupt operation

When an interrupt request is accepted, the contents of the following registers just before acceptance of the interrupt requests is automatically pushed onto the stack area in the order of ①, ② and ③.

- ① High-order contents of program counter (PC<sub>H</sub>)
- ② Low-order contents of program counter (PC<sub>L</sub>)
- ③ Contents of processor status register (PS)

After the contents of the above registers are pushed onto the stack area, the accepted interrupt vector address enters the program counter and consequently the interrupt processing routine is executed.

When the RTI instruction is executed at the end of the interrupt processing routine, the contents of the above registers pushed onto the stack area are restored to the respective registers in the order of ③, ② and ①; and the microcomputer resumes the processing executed just before acceptance of the interrupts. Figure 2.2.7 shows an interrupt operation diagram.

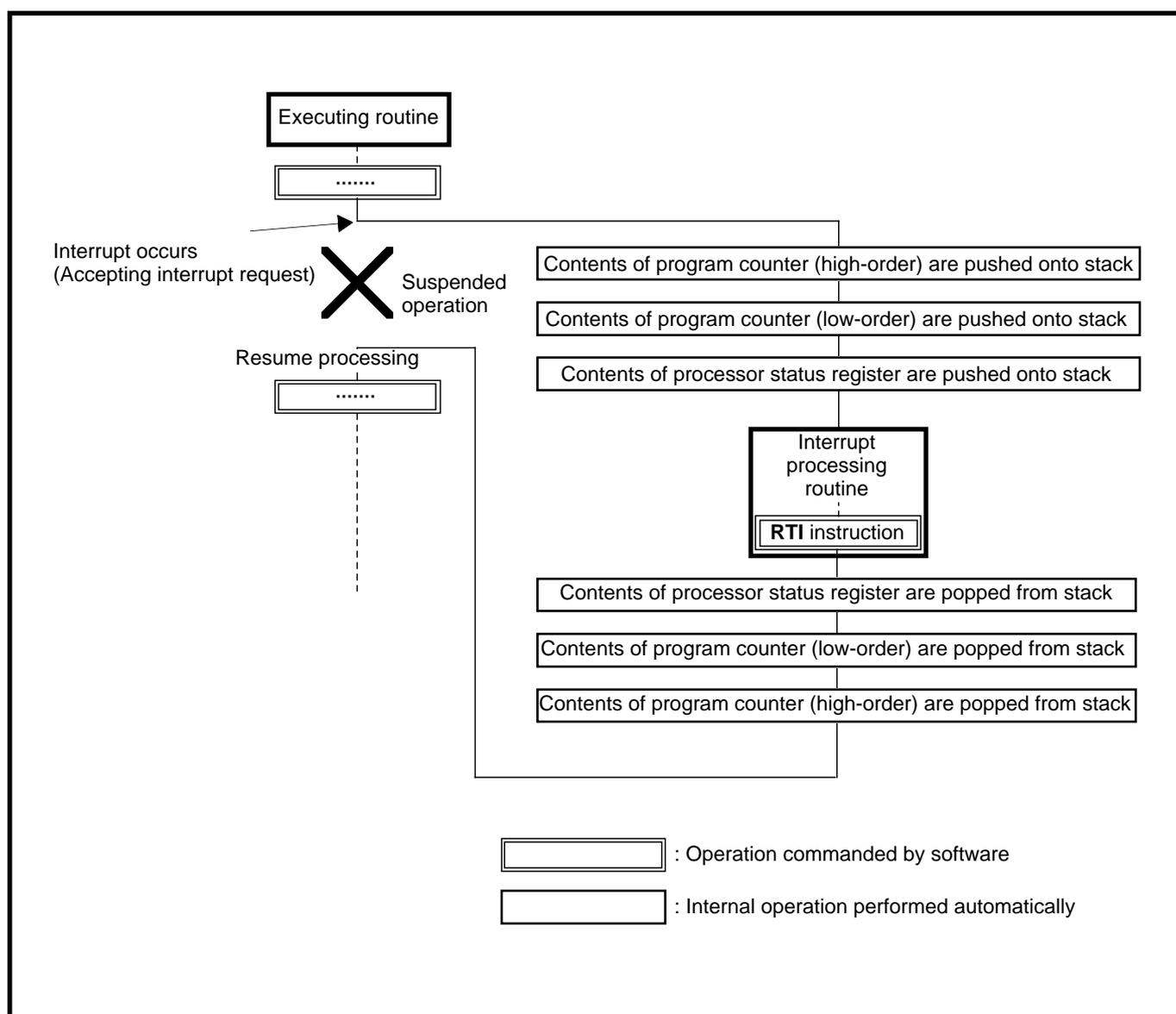


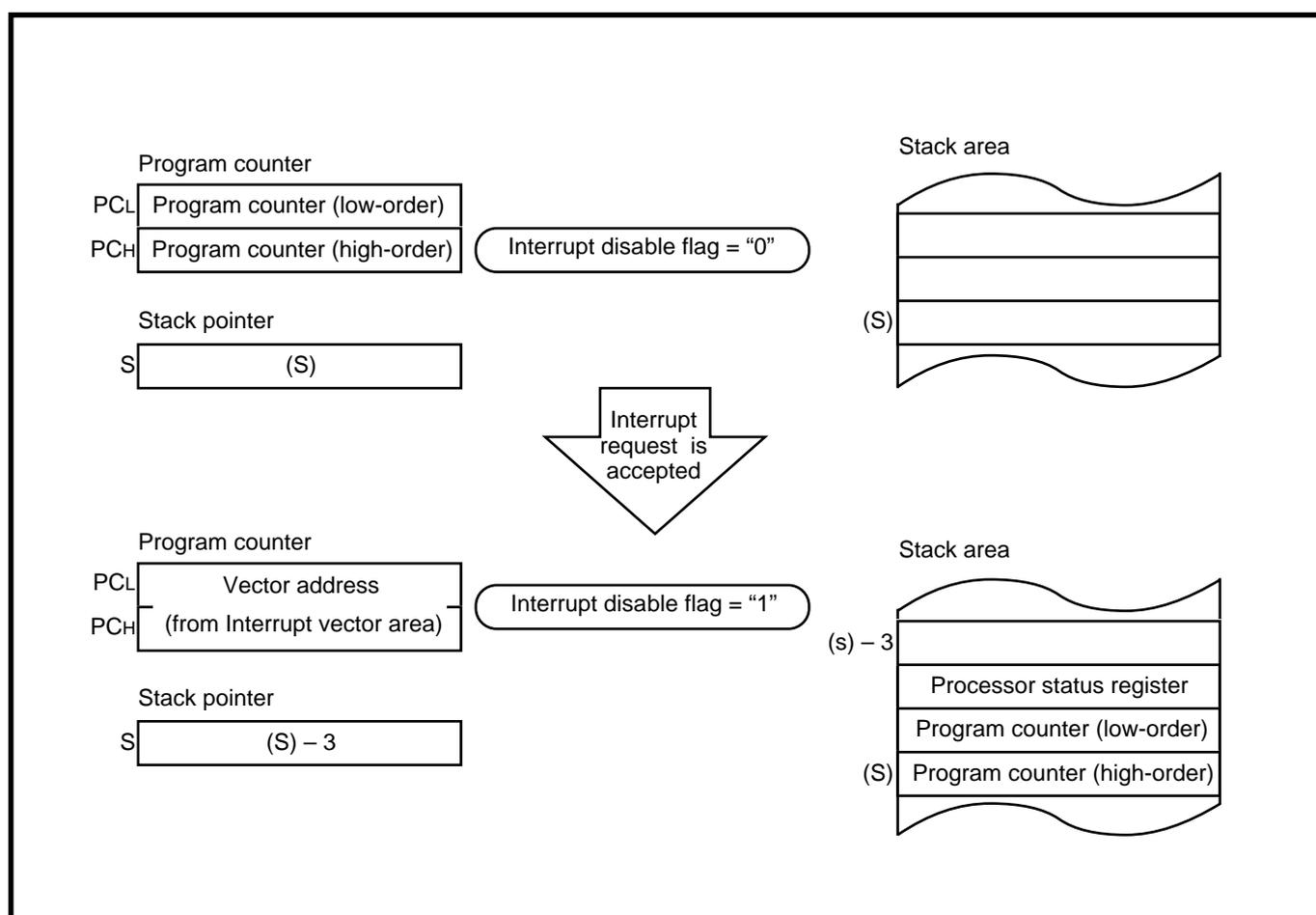
Fig. 2.2.7 Interrupt operation diagram

**(1) Processing upon acceptance of interrupt request**

Upon acceptance of an interrupt request, the following operations are automatically performed.

- ① The processing being executed is stopped.
- ② The contents of the program counter and the processor status register are pushed onto the stack area. Figure 2.2.8 shows the changes of the stack pointer and the program counter upon acceptance of an interrupt request.
- ③ Concurrently with the push operation, the jump destination address (the beginning address of the interrupt processing routine) of the occurring interrupt stored in the vector address is set in the program counter, then the interrupt processing routine is executed.
- ④ After the interrupt processing routine is started, the corresponding interrupt request bit is automatically cleared to "0". The interrupt disable flag is set to "1" so that multiple interrupts are disabled.

Accordingly, for executing the interrupt processing routine, it is necessary to set the jump destination address in the vector area corresponding to each interrupt.

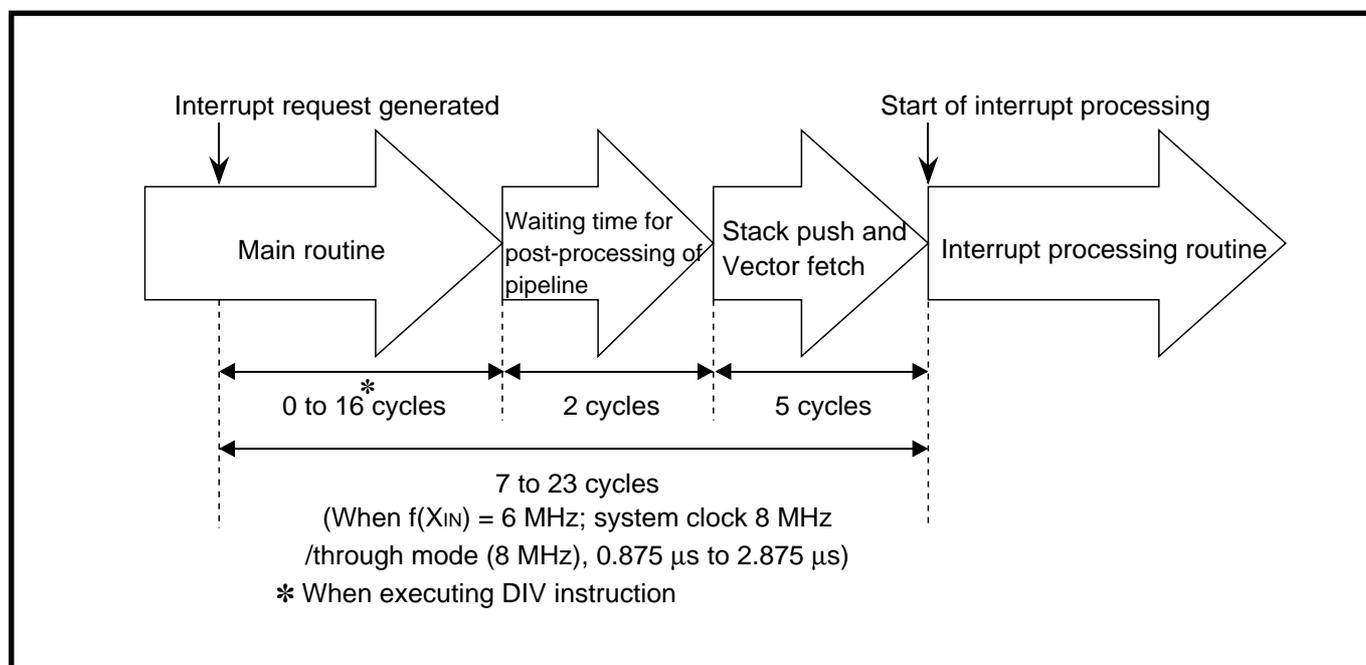


**Fig. 2.2.8 Changes of stack pointer and program counter upon acceptance of interrupt request**

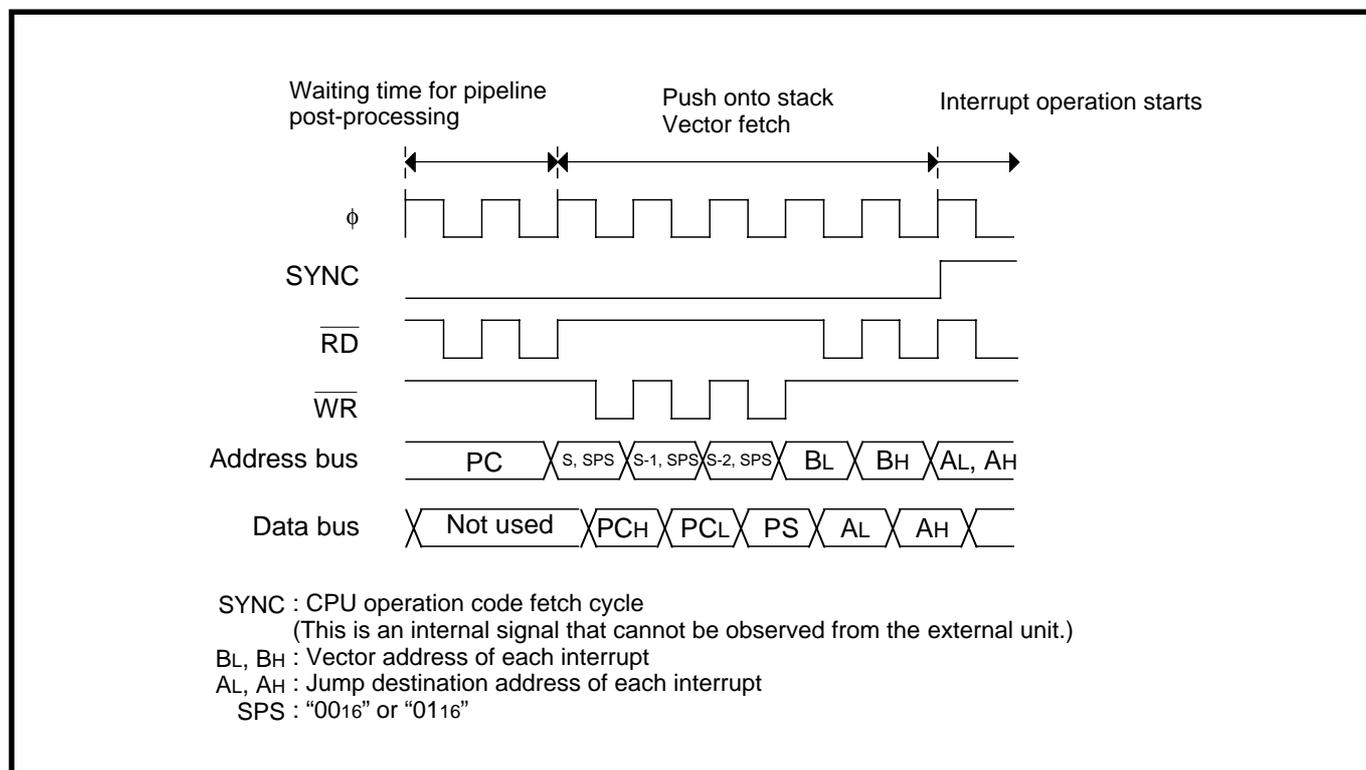
**(2) Timing after acceptance of interrupt request**

The interrupt processing routine begins with the machine cycle following the completion of the instruction that is currently being executed.

Figure 2.2.9 shows the time up to execution of interrupt processing routine and Figure 2.2.10 shows the timing chart after acceptance of interrupt request.



**Fig. 2.2.9 Time up to execution of interrupt processing routine**



**Fig. 2.2.10 Timing chart after acceptance of interrupt request**

### 2.2.5 Interrupt control

The acceptance of all interrupts, excluding the BRK instruction interrupt, can be controlled by the interrupt request bit, interrupt enable bit, and an interrupt disable flag, as described in detail below. Figure 2.2.11 shows an interrupt control diagram.

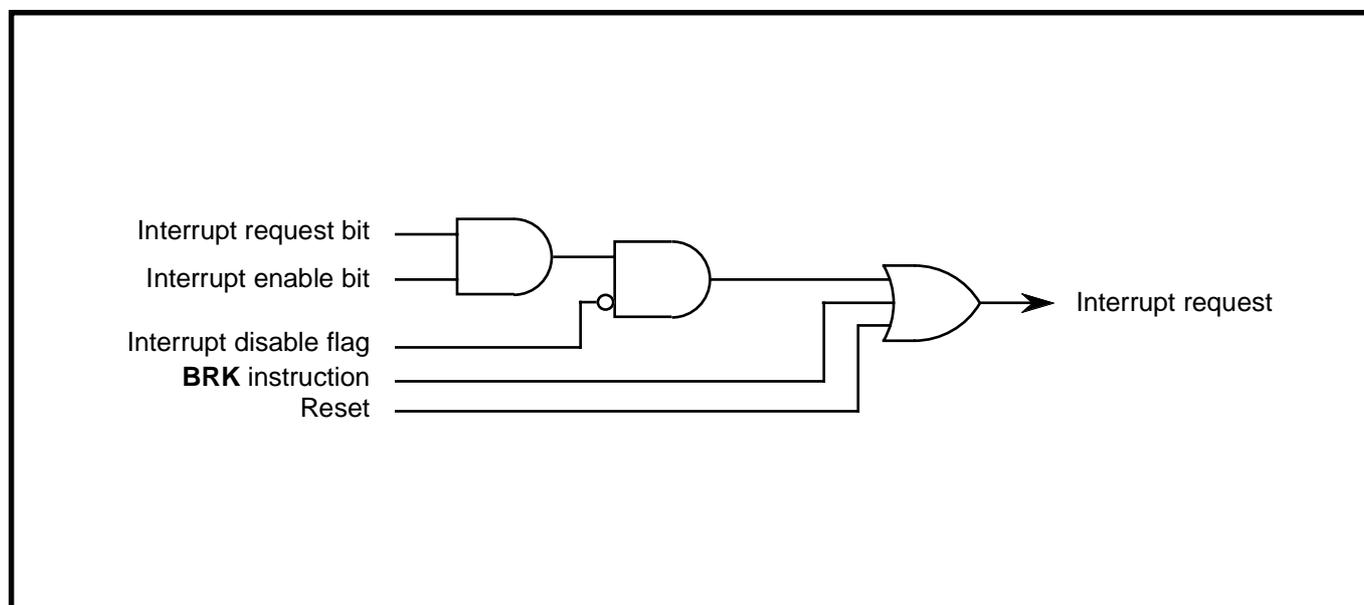


Fig. 2.2.11 Interrupt control diagram

The interrupt request bit, interrupt enable bit and interrupt disable flag function independently and do not affect each other. An interrupt is accepted when all the following conditions are satisfied.

- Interrupt request bit ..... "1"
- Interrupt enable bit ..... "1"
- Interrupt disable flag ..... "0"

Though the interrupt priority is determined by hardware, a variety of priority processing can be performed by software using the above bits and flag. Table 2.2.2 shows a list of interrupt control bits according to the interrupt source.

#### (1) Interrupt request bits

The interrupt request bits are allocated to the interrupt request register 1 (address  $3C_{16}$ ) and interrupt request register 2 (address  $3D_{16}$ ).

The occurrence of an interrupt request causes the corresponding interrupt request bit to be set to "1". The interrupt request bit is held in the "1" state until the interrupt is accepted. When the interrupt is accepted, this bit is automatically cleared to "0".

Each interrupt request bit can be set to "0", but cannot be set to "1", by software.

#### (2) Interrupt enable bits

The interrupt enable bits are allocated to the interrupt control register 1 (address  $003E_{16}$ ) and the interrupt control register 2 (address  $3F_{16}$ ).

The interrupt enable bits control the acceptance of the corresponding interrupt request.

When an interrupt enable bit is "0", the corresponding interrupt request is disabled. If an interrupt request occurs when this bit is "0", the corresponding interrupt request bit is set to "1" but the interrupt is not accepted. In this case, unless the interrupt request bit is set to "0" by software, the interrupt request bit remains in the "1" state.

When an interrupt enable bit is "1", the corresponding interrupt is enabled. If an interrupt request occurs when this bit is "1", the interrupt is accepted (when interrupt disable flag = "0").

Each interrupt enable bit can be set to "0" or "1" by software.

**(3) Interrupt disable flag**

The interrupt disable flag is allocated to bit 2 of the processor status register. The interrupt disable flag controls the acceptance of interrupt request except BRK instruction.

When this flag is "1", the acceptance of interrupt requests is disabled. When the flag is "0", the acceptance of interrupt requests is enabled. This flag is set to "1" with the SEI instruction and is set to "0" with the CLI instruction.

When a main routine branches to an interrupt processing routine, this flag is automatically set to "1", so that multiple interrupts are disabled. To use multiple interrupts, set this flag to "0" with the CLI instruction within the interrupt processing routine. Figure 2.2.12 shows an example of multiple interrupts.

**Table 2.2.2 List of interrupt bits according to interrupt source**

Interrupt source	Interrupt enable bit		Interrupt request bit	
	Address	Bit	Address	Bit
USB bus reset	003E <sub>16</sub>	b0	003C <sub>16</sub>	b0
USB SOF	003E <sub>16</sub>	b1	003C <sub>16</sub>	b1
USB device	003E <sub>16</sub>	b2	003C <sub>16</sub>	b2
External bus	003E <sub>16</sub>	b3	003C <sub>16</sub>	b3
INT <sub>0</sub>	003E <sub>16</sub>	b4	003C <sub>16</sub>	b4
Timer X	003E <sub>16</sub>	b5	003C <sub>16</sub>	b5
Timer 1	003E <sub>16</sub>	b6	003C <sub>16</sub>	b6
Timer 2	003E <sub>16</sub>	b7	003C <sub>16</sub>	b7
INT <sub>1</sub>	003F <sub>16</sub>	b0	003D <sub>16</sub>	b0
USB HUB	003F <sub>16</sub>	b1	003D <sub>16</sub>	b1
Serial I/O receive	003F <sub>16</sub>	b2	003D <sub>16</sub>	b2
Serial I/O transmit	003F <sub>16</sub>	b3	003D <sub>16</sub>	b3
CNTR <sub>0</sub>	003F <sub>16</sub>	b4	003D <sub>16</sub>	b4
Key-on wake-up	003F <sub>16</sub>	b5	003D <sub>16</sub>	b5
A/D converter	003F <sub>16</sub>	b6	003D <sub>16</sub>	b6

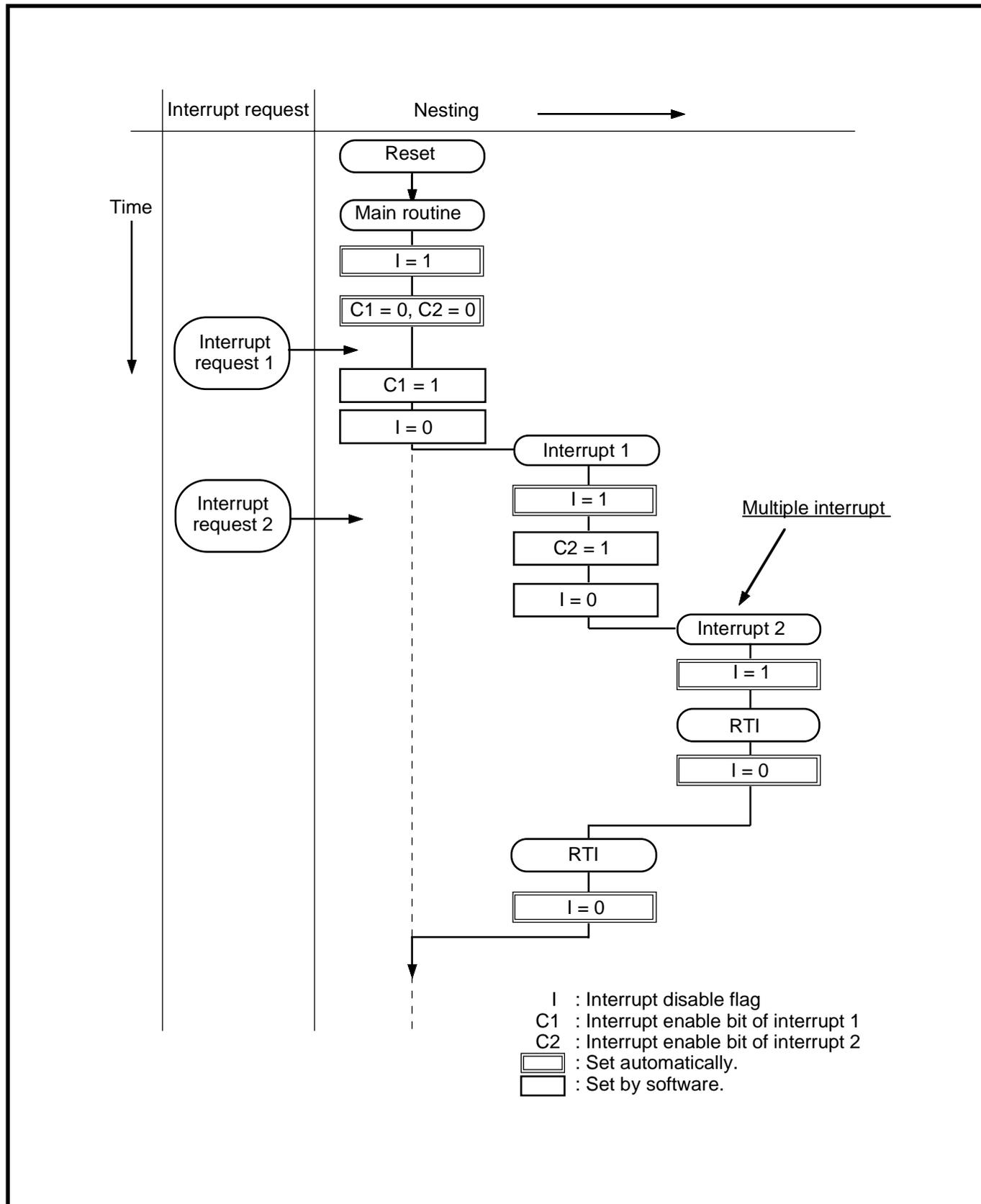


Fig. 2.2.12 Example of multiple interrupts

### 2.2.6 INT interrupt

The INT interrupt requests is generated when the microcomputer detects a level change of each INT pin (INT<sub>0</sub>, INT<sub>1</sub>).

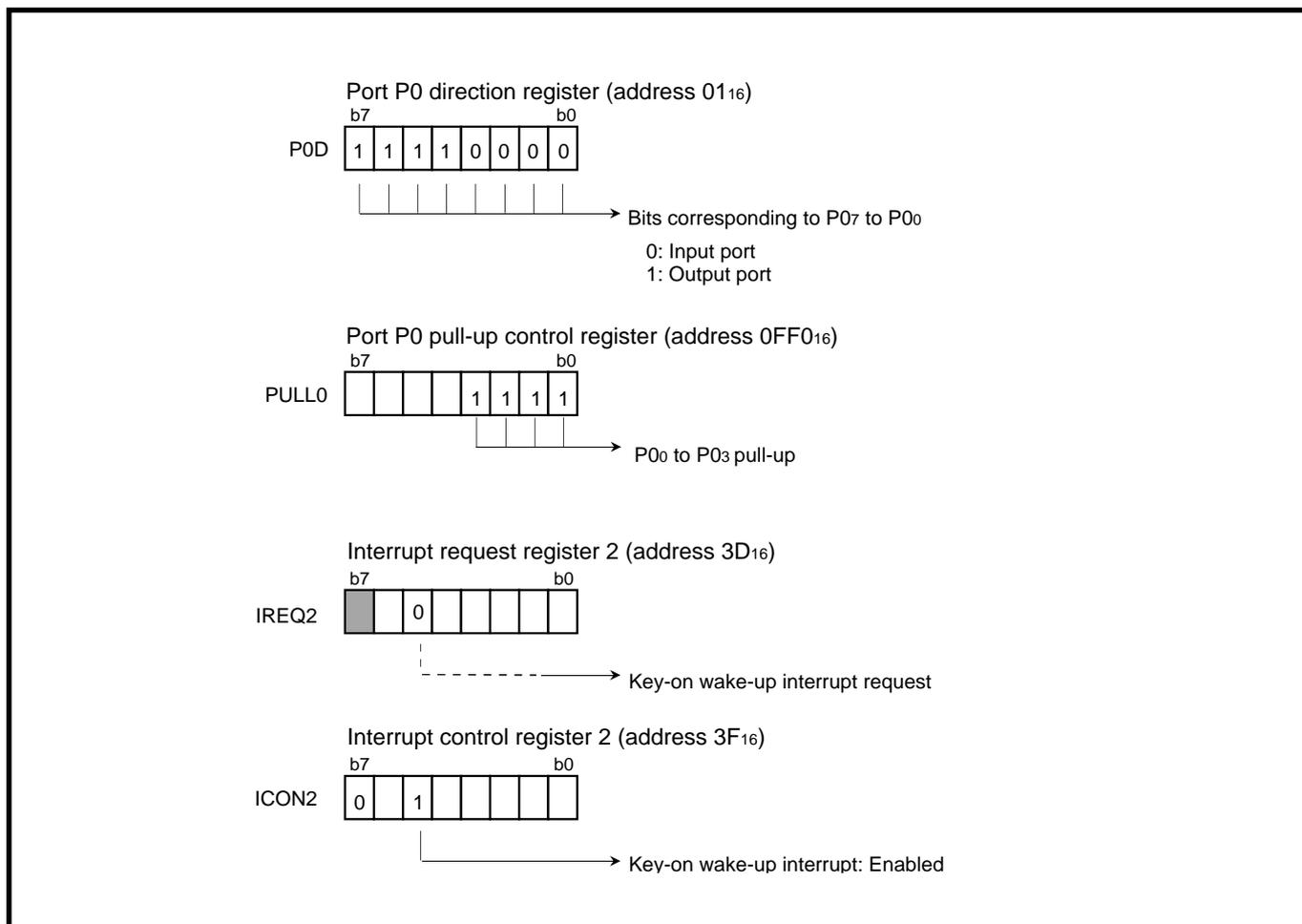
#### (1) Active edge selection

INT<sub>0</sub> and INT<sub>1</sub> can be selected from either a falling edge or rising edge detection as an active edge by the interrupt edge selection register. In the "0" state, the falling edge of the corresponding pin is detected. In the "1" state, the rising edge of the corresponding pin is detected.



**(2) Related registers setting**

Figure 2.2.14 shows the related registers setting (corresponding to Figure 2.2.13).



**Fig. 2.2.14 Registers setting related to key input interrupt (corresponding to Figure 2.2.13)**

### 2.2.8 Notes on interrupts

#### (1) Change of relevant register settings

When the setting of the following registers or bits is changed, the interrupt request bit may be set to "1". When not requiring the interrupt occurrence synchronized with these setting, take the following sequence.

- Interrupt edge selection register (address 0FF3<sub>16</sub>)
- Timer X mode register (address 23<sub>16</sub>)

Set the above listed registers or bits as the following sequence.

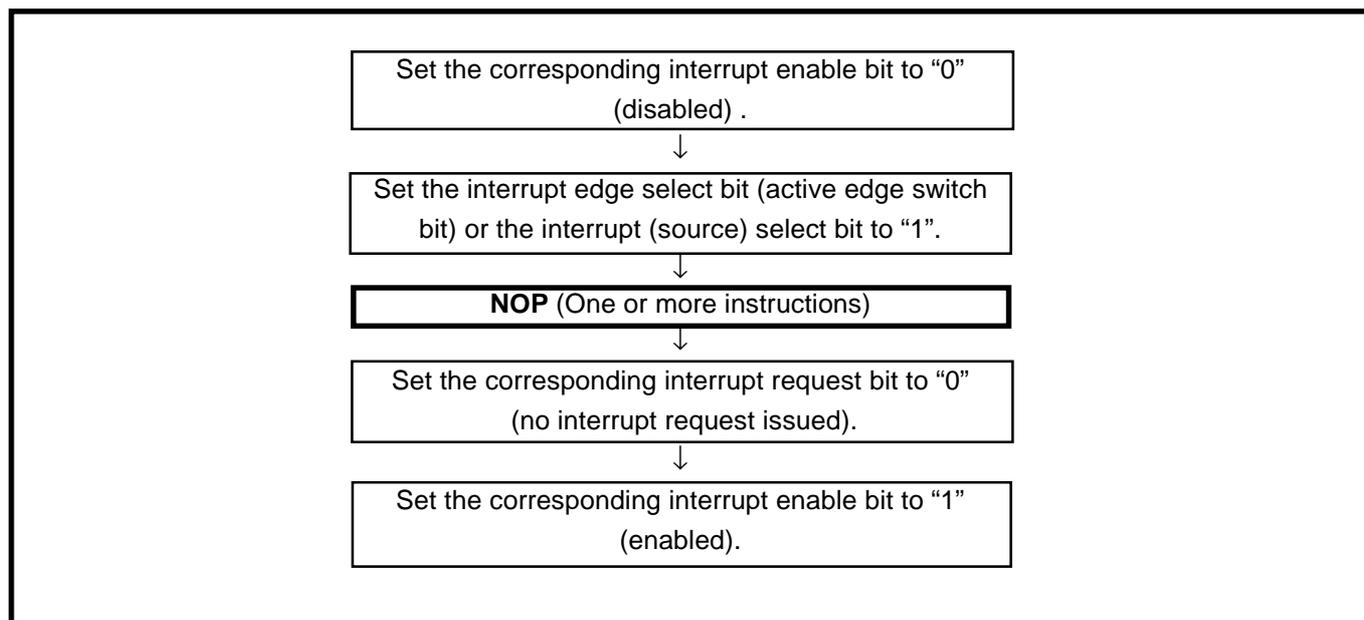


Fig. 2.2.15 Sequence of changing relevant register

#### ■ Reason

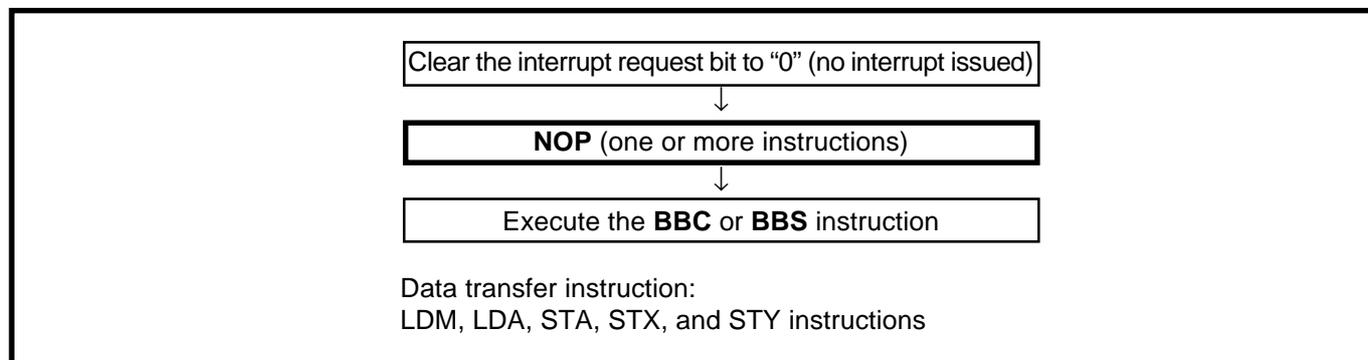
When setting the following, the interrupt request bit may be set to "1".

- When setting external interrupt active edge

Concerned register: Interrupt edge selection register (address 0FF3<sub>16</sub>)  
Timer X mode register (address 23<sub>16</sub>)

**(2) Check of interrupt request bit**

- When executing the **BBC** or **BBS** instruction to an interrupt request bit of an interrupt request register immediately after this bit is set to "0" by using a data transfer instruction, execute one or more instructions before executing the **BBC** or **BBS** instruction.

**Fig. 2.2.16 Sequence of check of interrupt request bit****■ Reason**

If the BBC or BBS instruction is executed immediately after an interrupt request bit of an interrupt request register is cleared to "0", the value of the interrupt request bit before being cleared to "0" is read.

## 2.3 Timer

This paragraph explains the registers setting method and the notes related to the timers.

### 2.3.1 Memory map

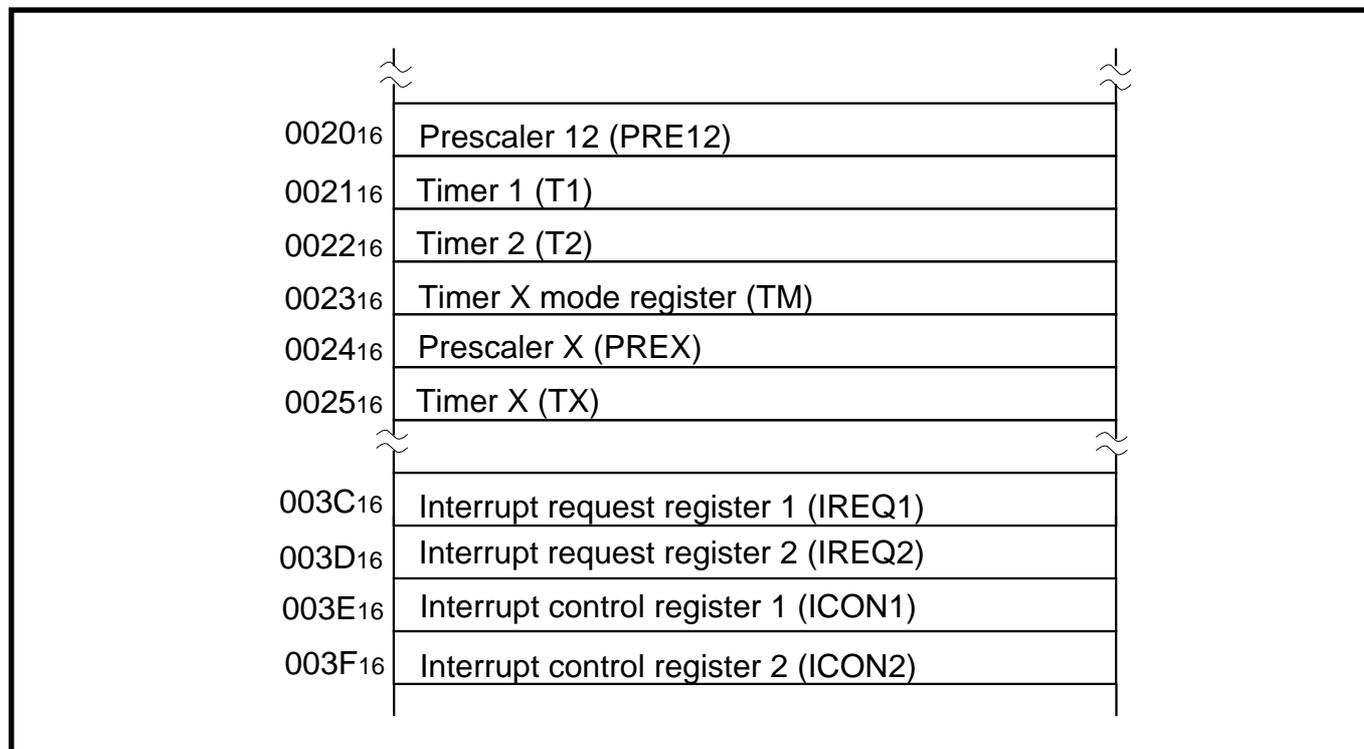


Fig. 2.3.1 Memory map of registers related to timers

### 2.3.2 Related registers

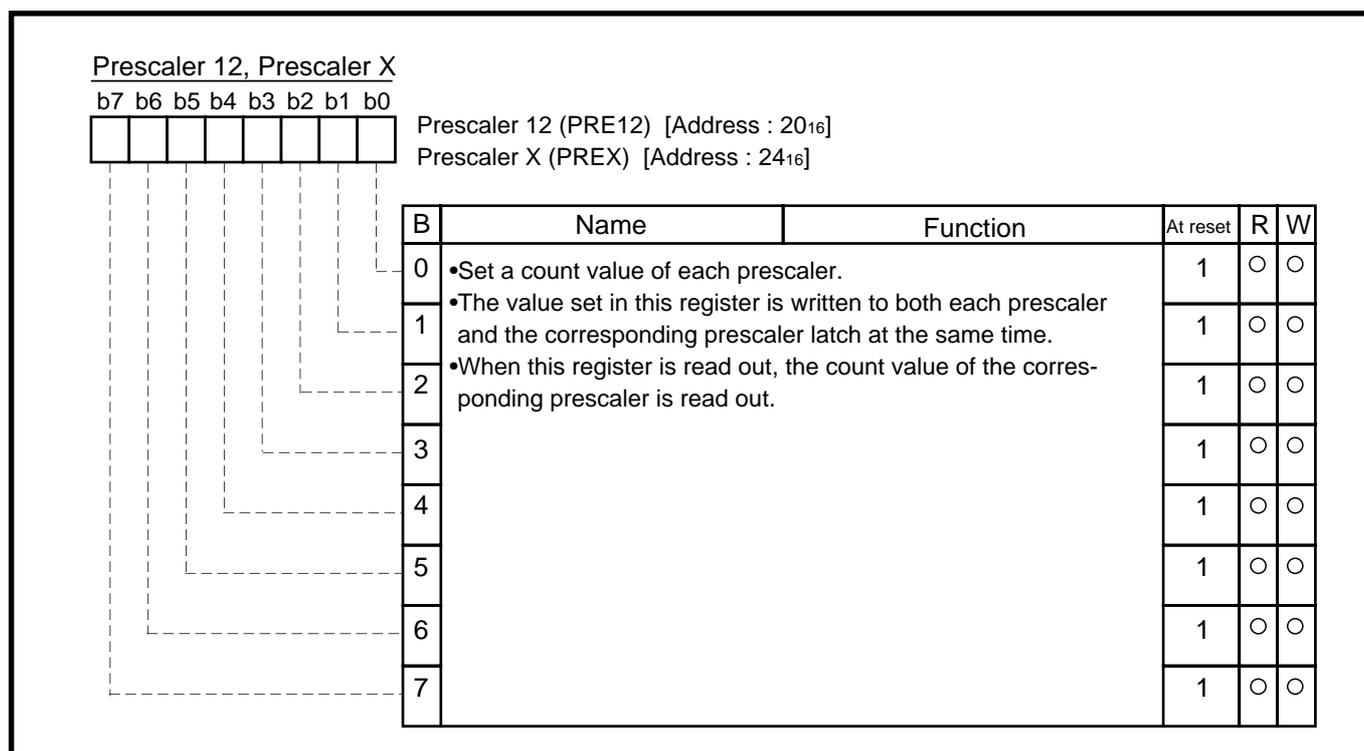


Fig. 2.3.2 Structure of Prescaler 12, Prescaler X

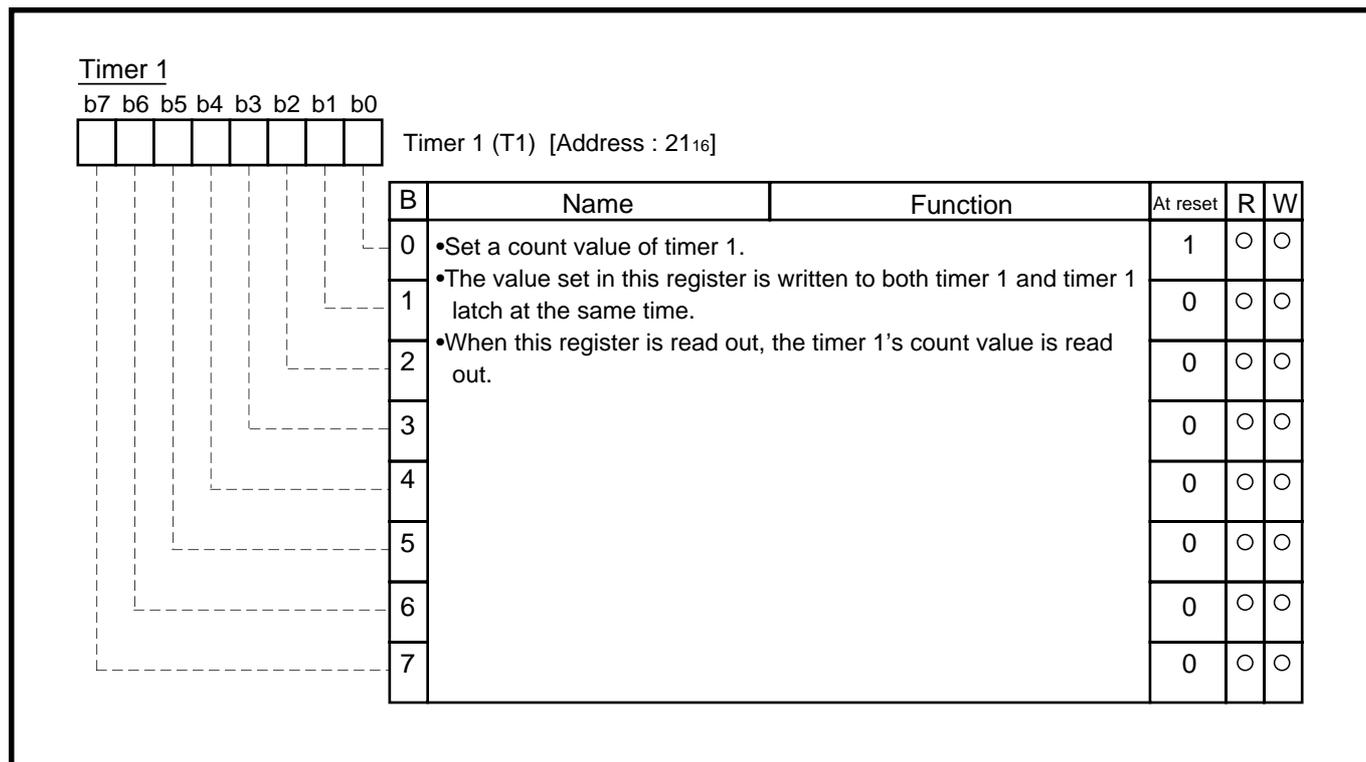


Fig. 2.3.3 Structure of Timer 1

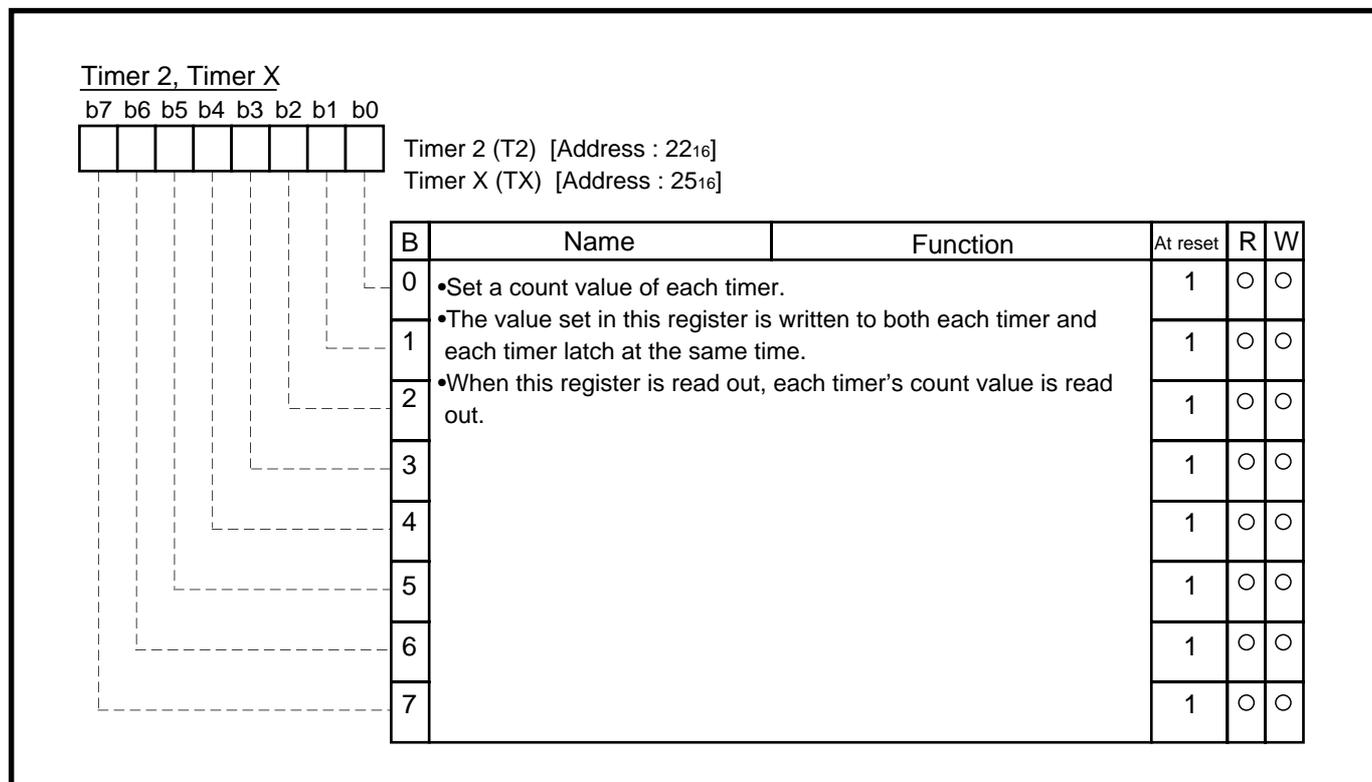


Fig. 2.3.4 Structure of Timer 2, Timer X

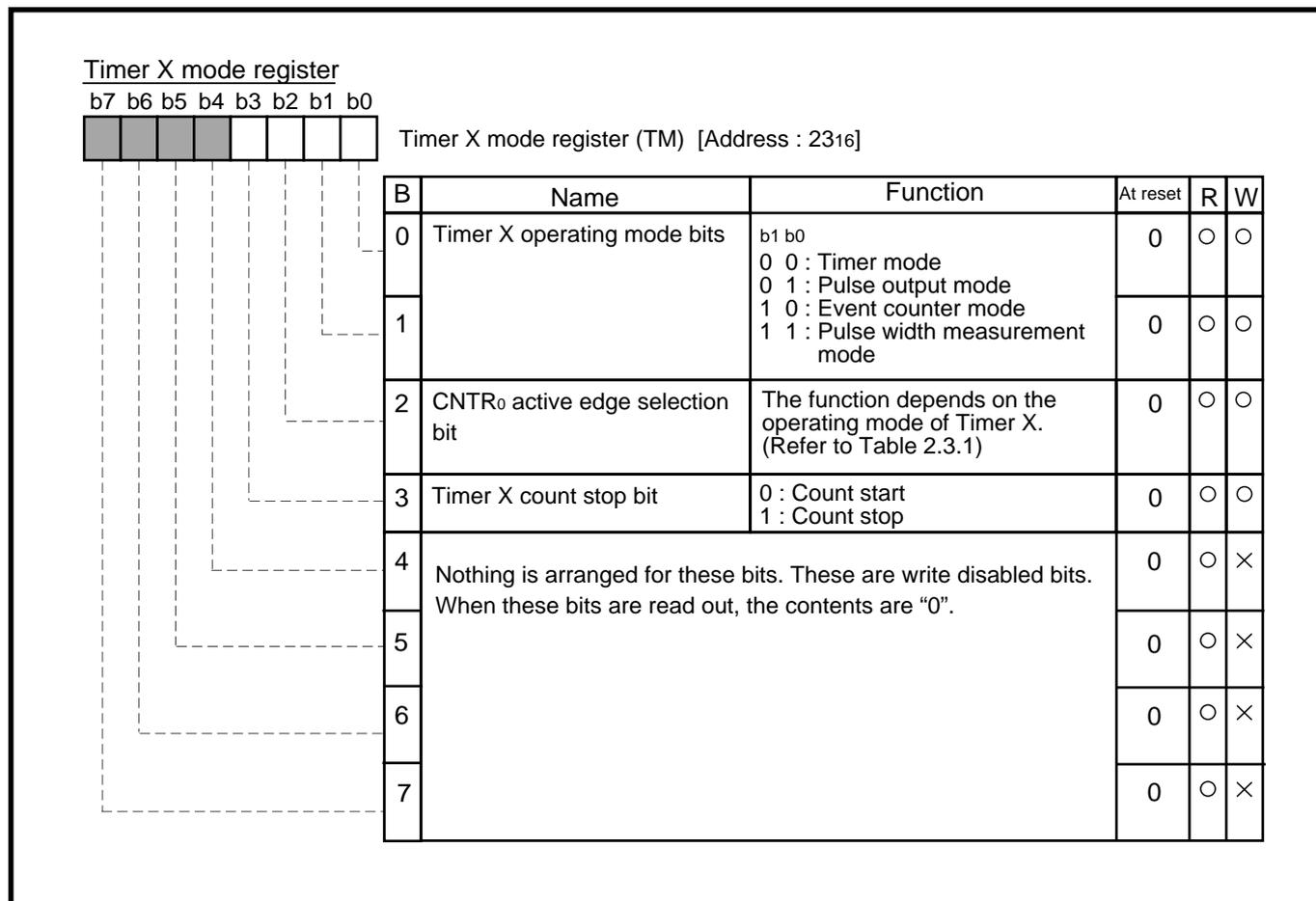


Fig. 2.3.5 Structure of Timer X mode register

Table 2.3.1 CNTR<sub>0</sub> active edge selection bit function

Timer X operation modes	CNTR <sub>0</sub> active edge selection bit (bits 2 of address 23 <sub>16</sub> ) contents	
Timer mode	"0"	CNTR <sub>0</sub> interrupt request occurrence: Falling edge ; No influence to timer count
	"1"	CNTR <sub>0</sub> interrupt request occurrence: Rising edge ; No influence to timer count
Pulse output mode	"0"	Pulse output start: Beginning at "H" level CNTR <sub>0</sub> interrupt request occurrence: Falling edge
	"1"	Pulse output start: Beginning at "L" level CNTR <sub>0</sub> interrupt request occurrence: Rising edge
Event counter mode	"0"	Timer X: Rising edge count CNTR <sub>0</sub> interrupt request occurrence: Falling edge
	"1"	Timer X: Falling edge count CNTR <sub>0</sub> interrupt request occurrence: Rising edge
Pulse width measurement mode	"0"	Timer X: "H" level width measurement CNTR <sub>0</sub> interrupt request occurrence: Falling edge
	"1"	Timer X: "L" level width measurement CNTR <sub>0</sub> interrupt request occurrence: Rising edge

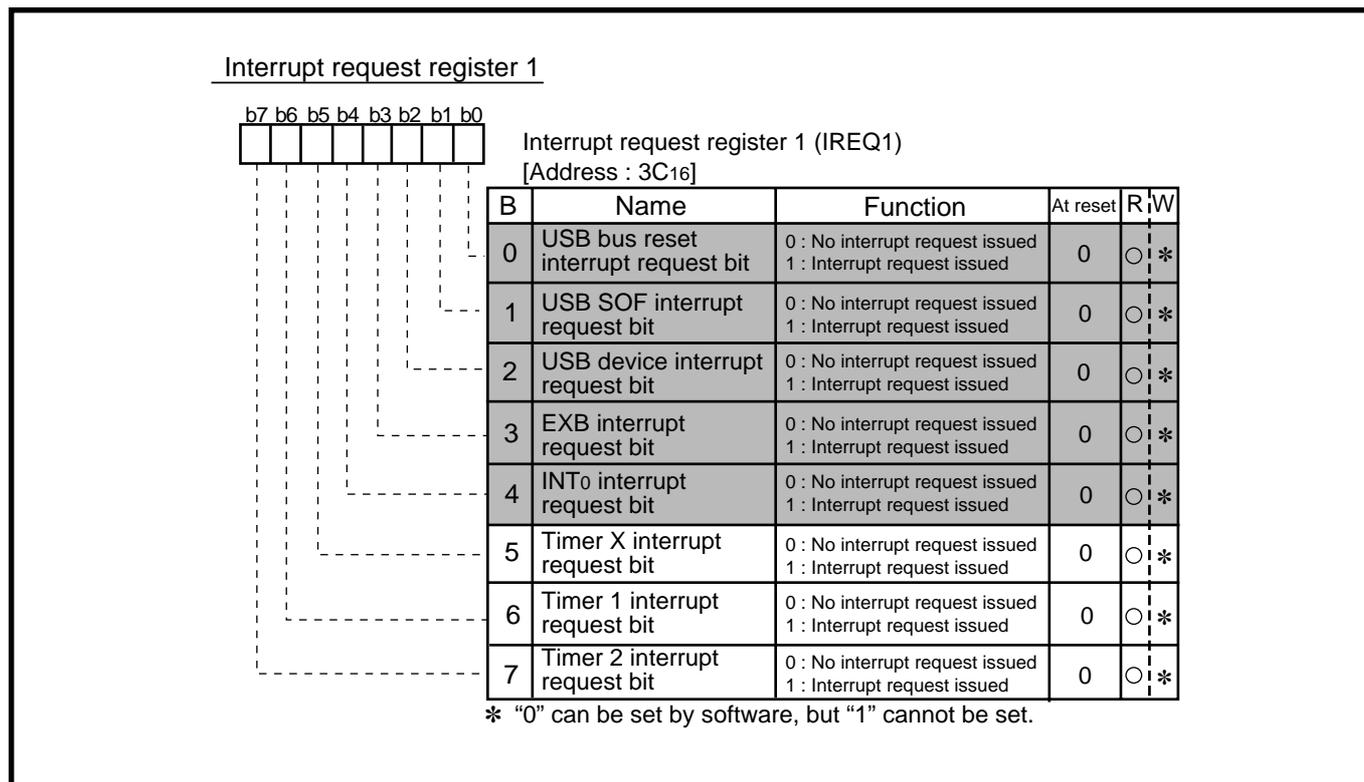


Fig. 2.3.6 Structure of Interrupt request register 1

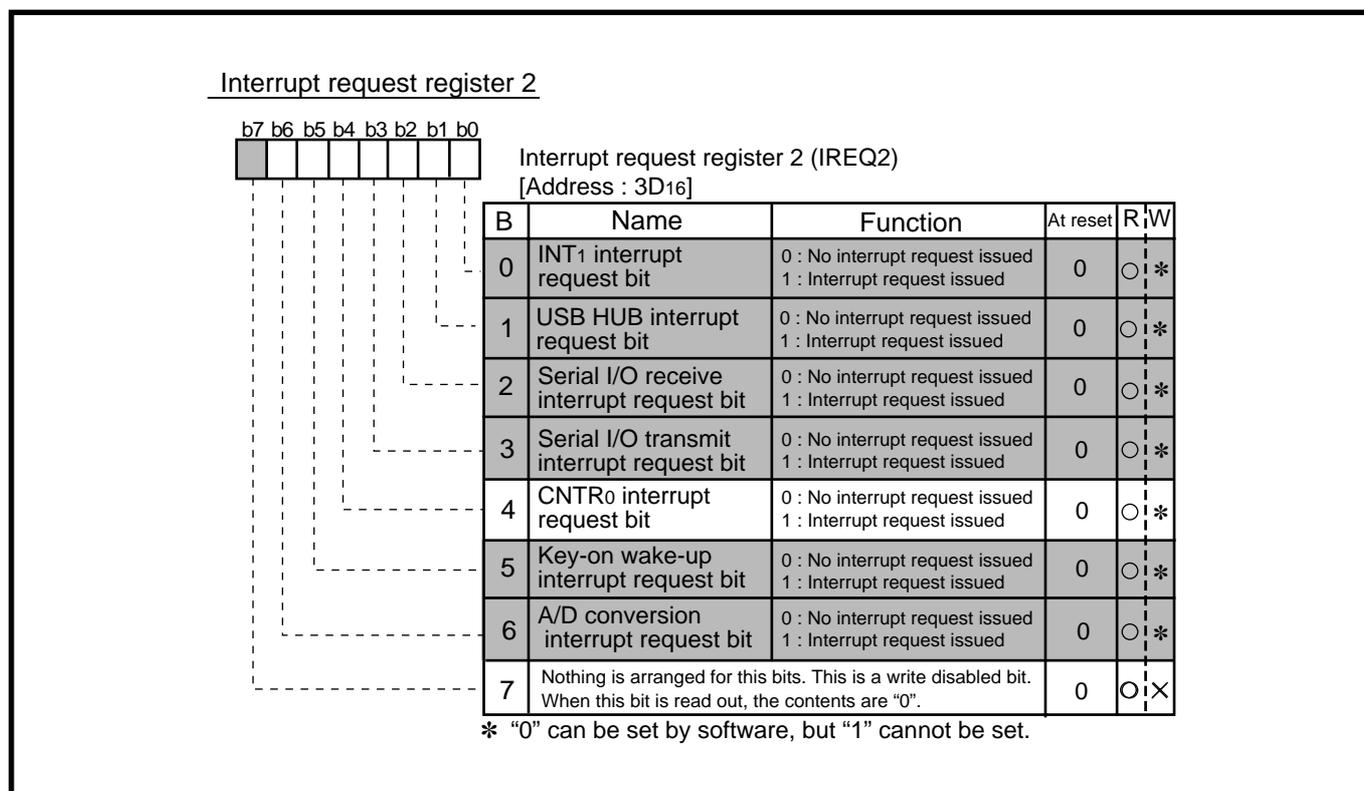


Fig. 2.3.7 Structure of Interrupt request register 2

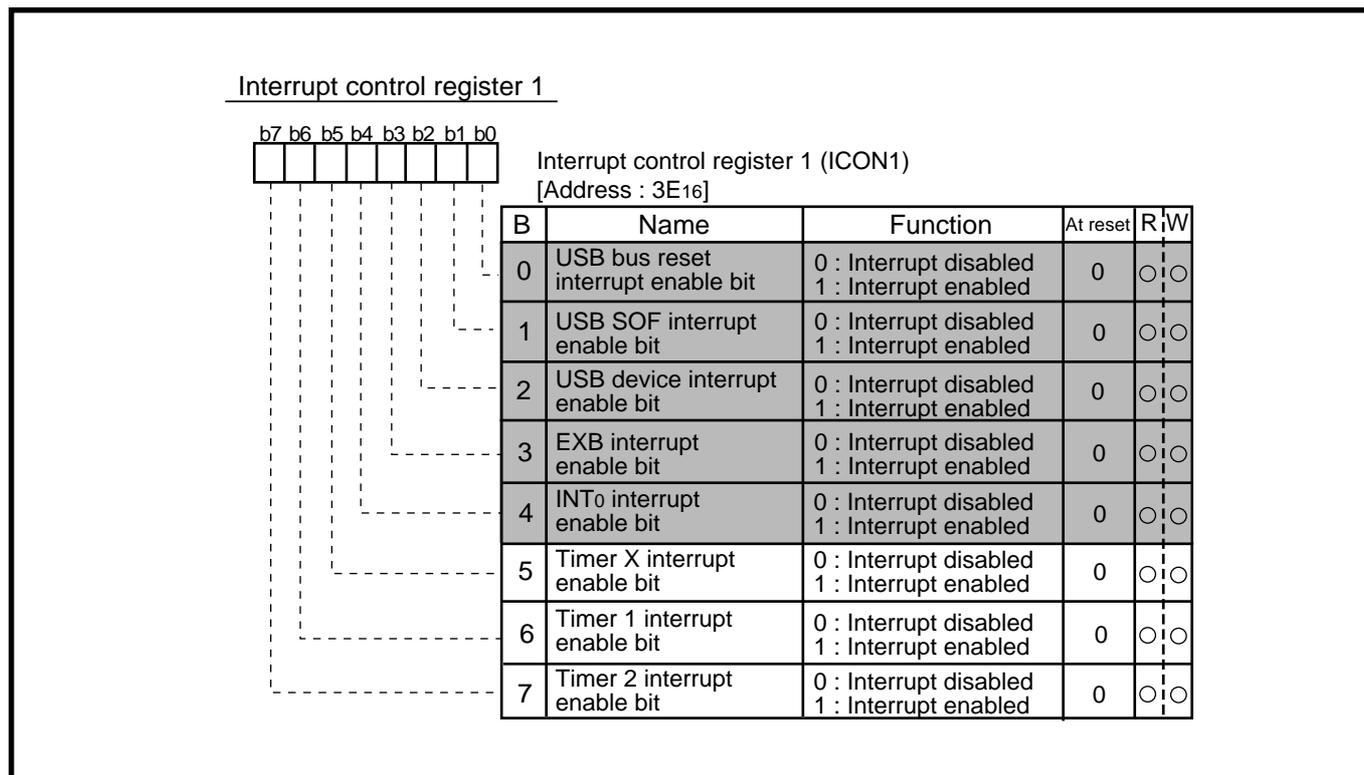


Fig. 2.3.8 Structure of Interrupt control register 1

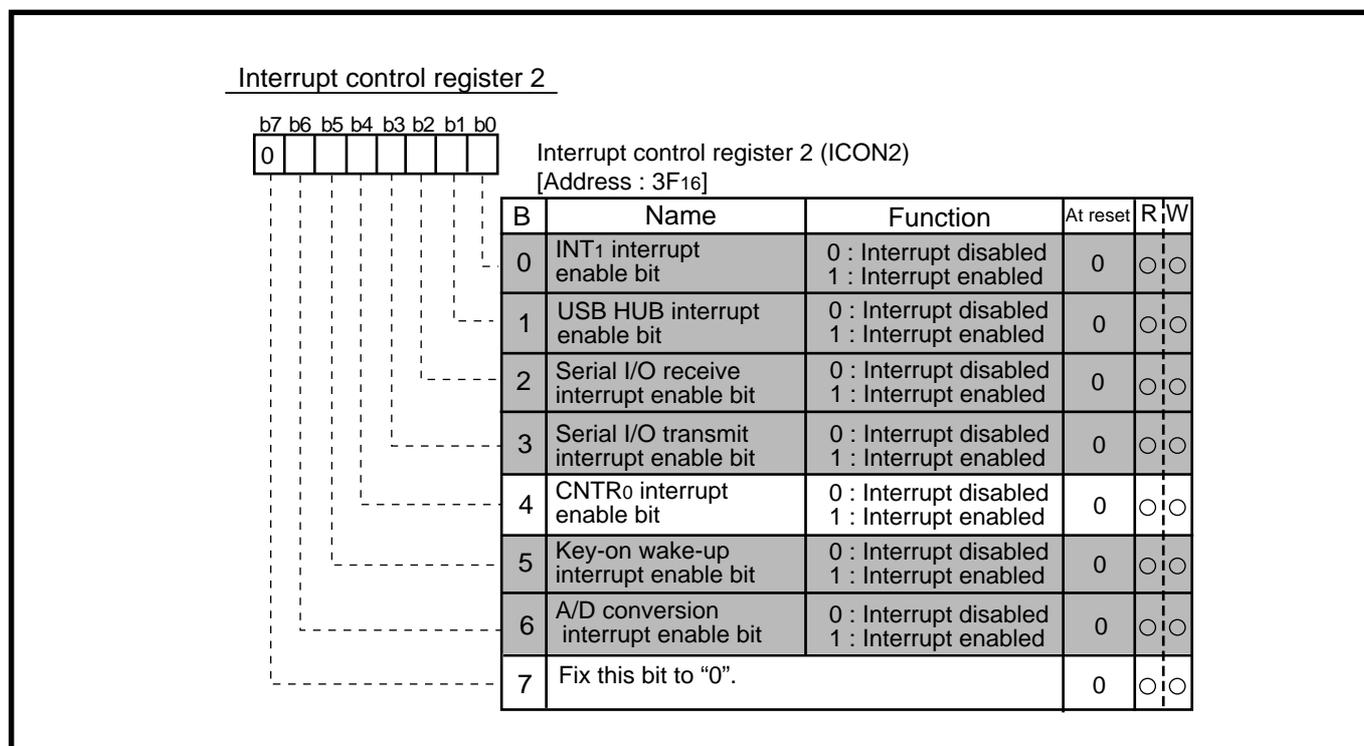


Fig. 2.3.9 Structure of Interrupt control register 2

### 2.3.3 Timer application examples

#### (1) Basic functions and uses

##### [Function 1] Control of Event interval (Timer X, Timer 1, Timer 2)

When a certain time, by setting a count value to each timer, has passed, the timer interrupt request occurs.

<Use>

- Generation of an output signal timing
- Generation of a wait time

##### [Function 2] Control of Cyclic operation (Timer X, Timer 1, Timer 2)

The value of the timer latch is automatically written to the corresponding timer each time the timer underflows, and each timer interrupt request occurs in cycles.

<Use>

- Generation of cyclic interrupts
- Clock function (measurement of 10 ms); see Application example 1
- Control of a main routine cycle

##### [Function 3] Output of Rectangular waveform (Timer X)

The output level of the CNTR<sub>0</sub> pin is inverted each time the timer underflows (in the pulse output mode).

<Use>

- Piezoelectric buzzer output; see Application example 2
- Generation of the remote control carrier waveforms

##### [Function 4] Count of External pulses (Timer X)

External pulses input to the CNTR<sub>0</sub> pin are counted as the timer count source (in the event counter mode).

<Use>

- Frequency measurement; see Application example 3
- Division of external pulses
- Generation of interrupts due to a cycle using external pulses as the count source; count of a reel pulse

##### [Function 5] Measurement of External pulse width (Timer X)

The "H" or "L" level width of external pulses input to CNTR<sub>0</sub> pin is measured (in the pulse width measurement mode).

<Use>

- Measurement of external pulse frequency (measurement of pulse width of FG pulse\* for a motor); see Application example 4
- Measurement of external pulse duty (when the frequency is fixed)

FG pulse\*: Pulse used for detecting the motor speed to control the motor speed.

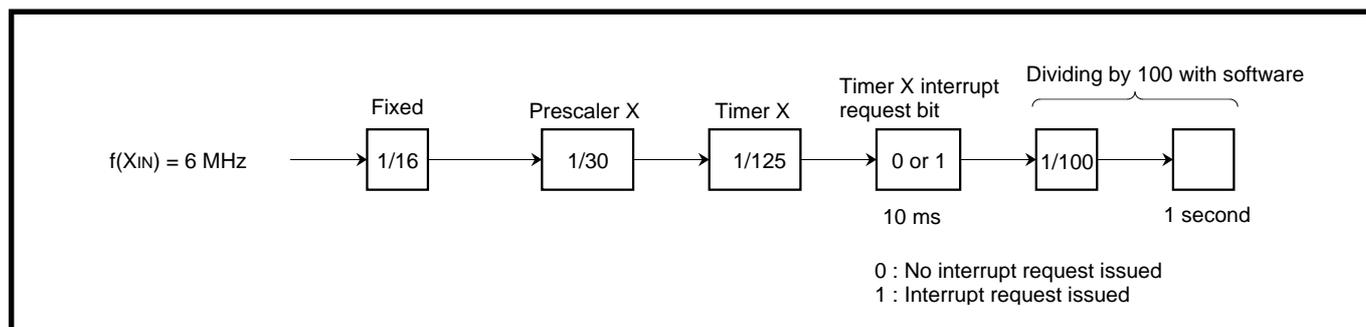
**(2) Timer application example 1: Clock function (measurement of 10 ms)**

**Outline:** The input clock is divided by the timer so that the clock can count up at 10 ms intervals.

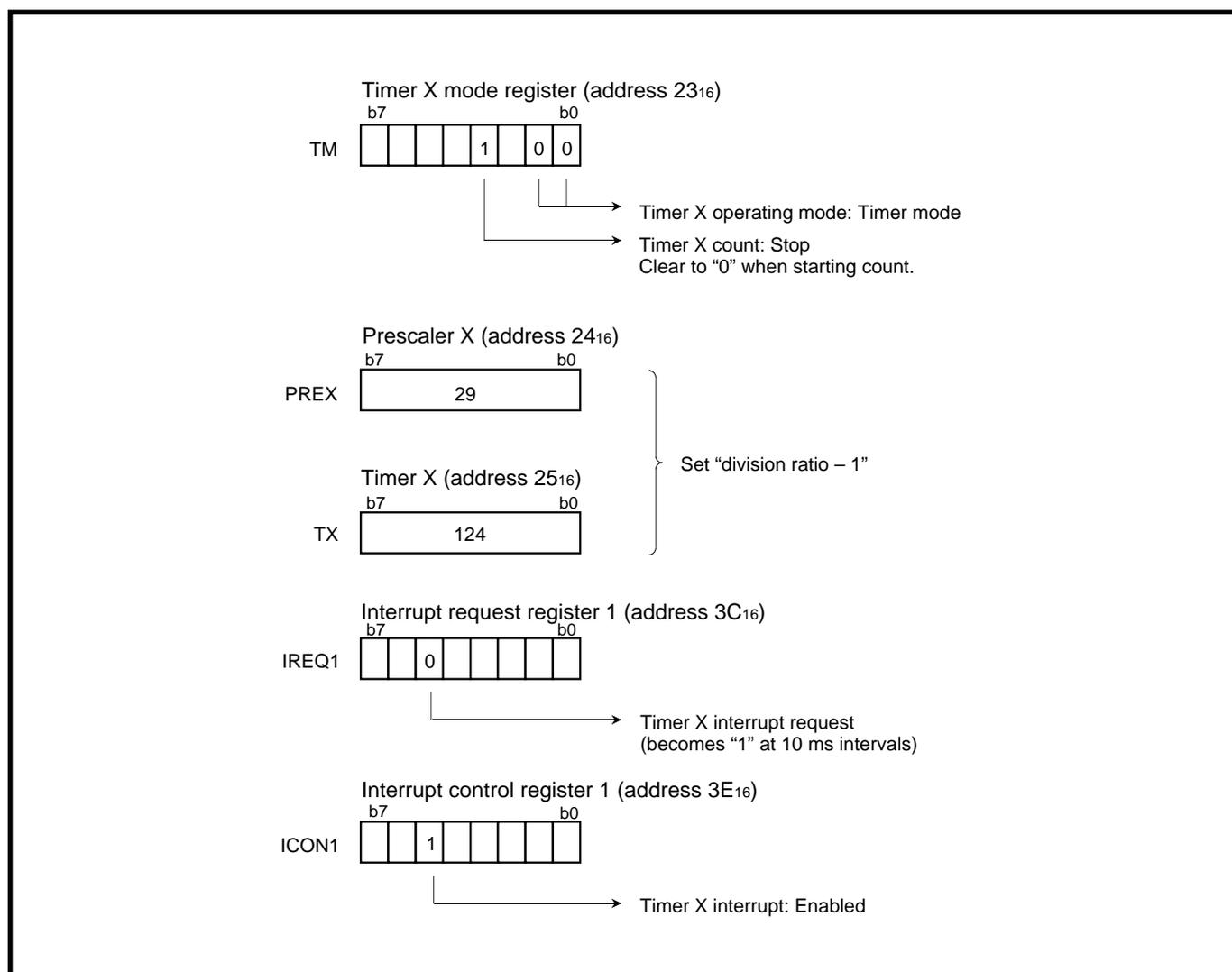
**Specifications:** •The clock  $f(X_{IN}) = 6$  MHz is divided by the timer.

- The clock is counted up in the process routine of the timer X interrupt which occurs at 10 ms intervals.

Figure 2.3.10 shows the timers connection and setting of division ratios; Figure 2.3.11 shows the related registers setting; Figure 2.3.12 shows the control procedure.



**Fig. 2.3.10 Timers connection and setting of division ratios**



**Fig. 2.3.11 Related registers setting**

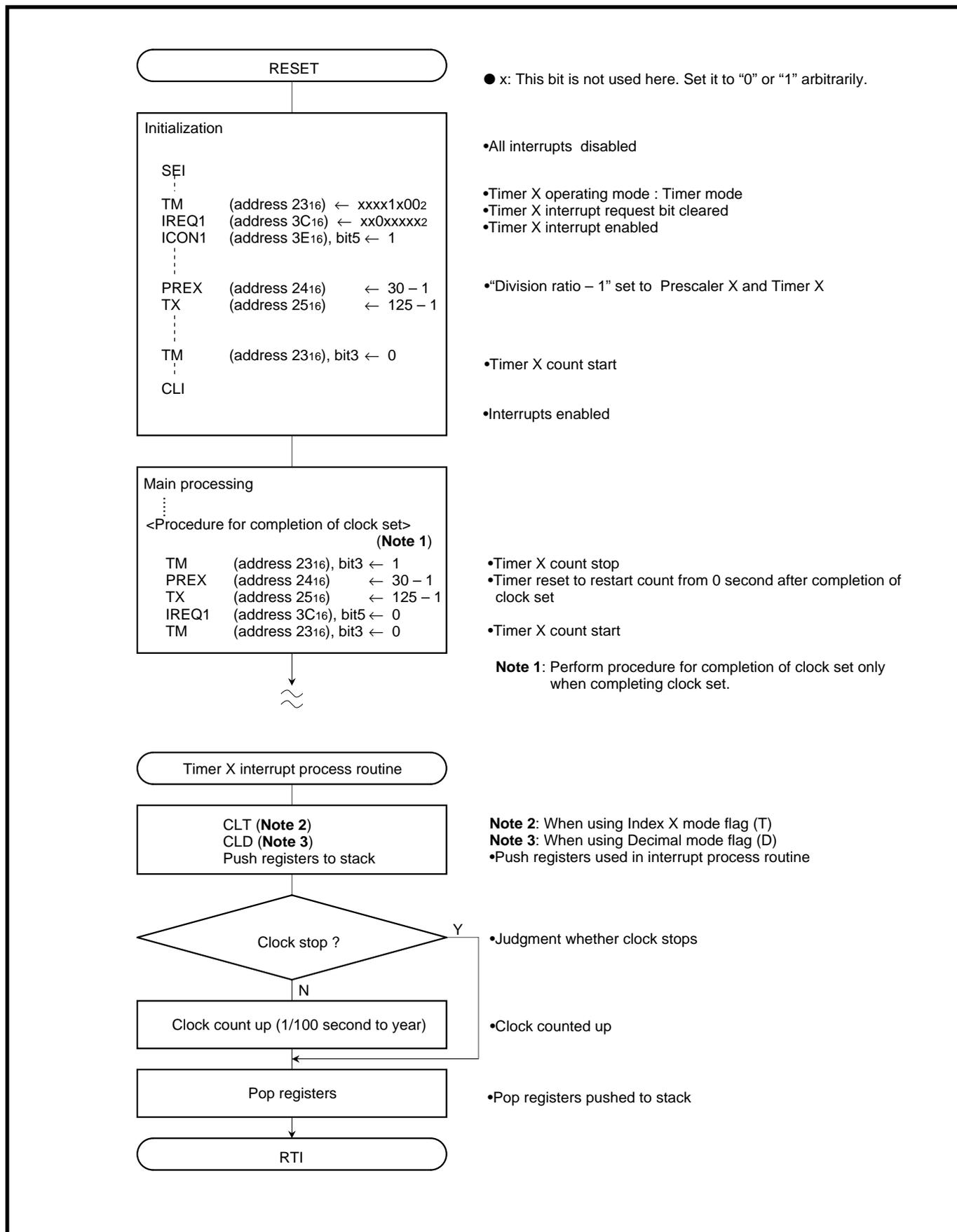


Fig. 2.3.12 Control procedure

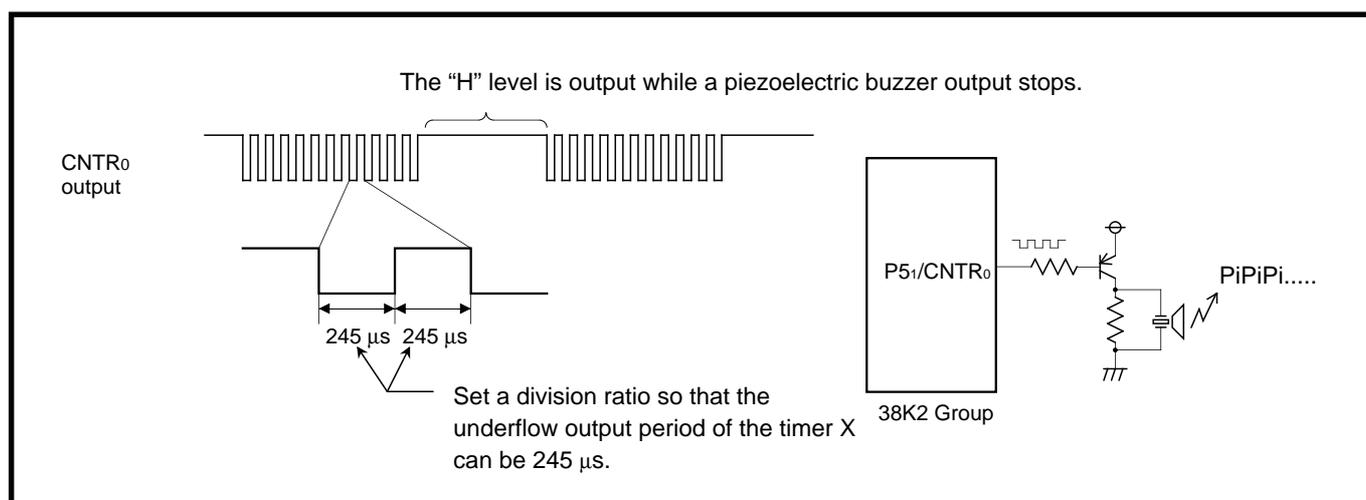
**(3) Timer application example 2: Piezoelectric buzzer output**

**Outline:** The rectangular waveform output function of the timer is applied for a piezoelectric buzzer output.

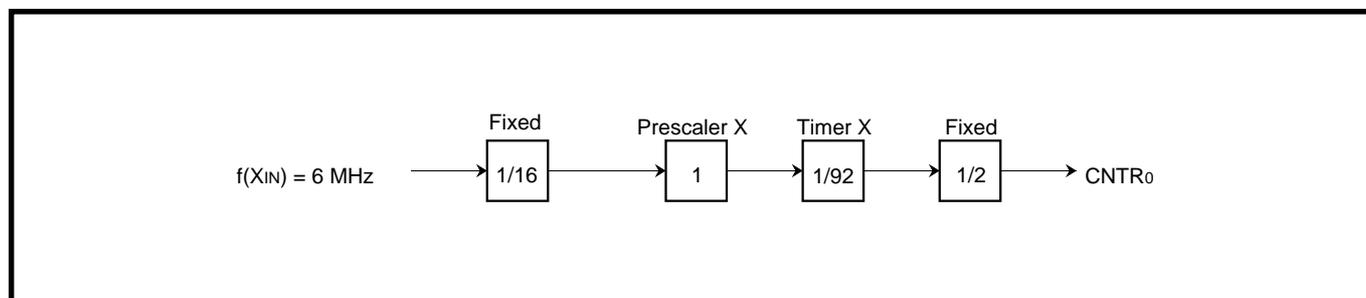
**Specifications:** •The rectangular waveform, dividing the clock  $f(X_{IN}) = 6 \text{ MHz}$  into about 2 kHz (2038 Hz), is output from the P5<sub>1</sub>/CNTR<sub>0</sub> pin.

•The level of the P5<sub>1</sub>/CNTR<sub>0</sub> pin is fixed to "H" while a piezoelectric buzzer output stops.

Figure 2.3.13 shows a peripheral circuit example, and Figure 2.3.14 shows the timers connection and setting of division ratios. Figures 2.3.15 shows the related registers setting, and Figure 2.3.16 shows the control procedure.



**Fig. 2.3.13 Peripheral circuit example**



**Fig. 2.3.14 Timers connection and setting of division ratios**

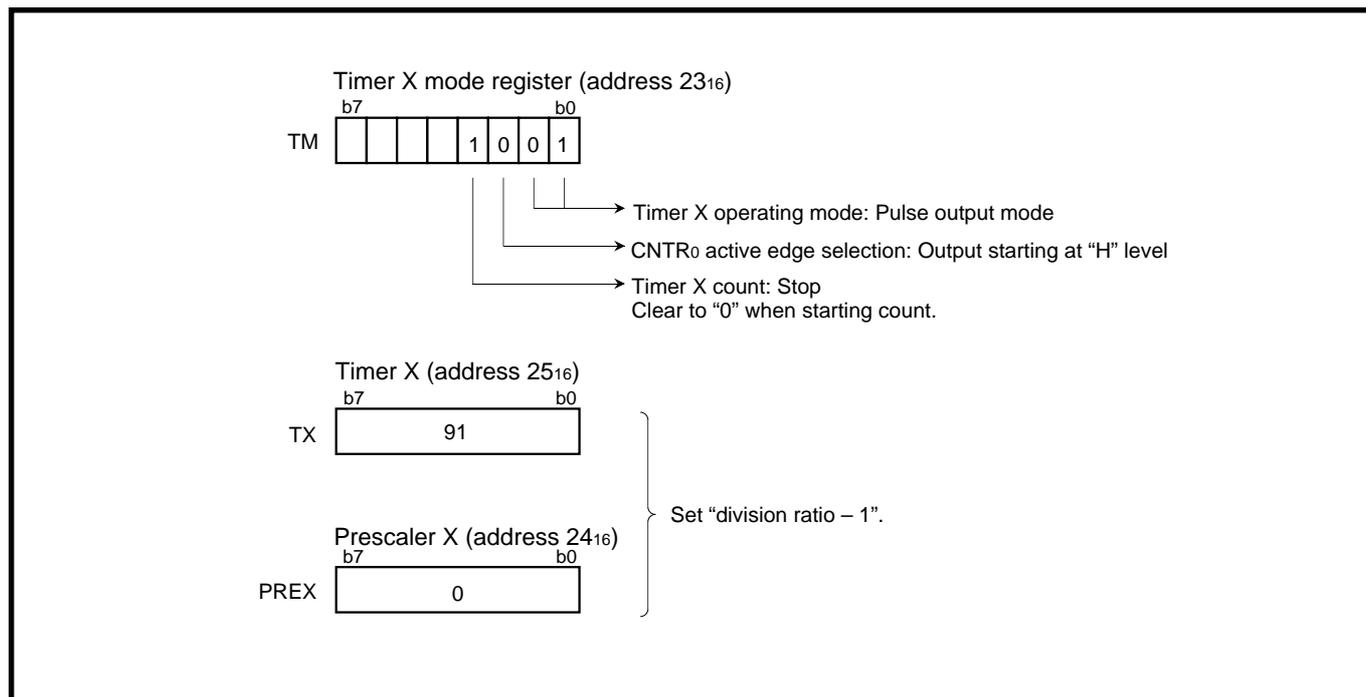


Fig. 2.3.15 Related registers setting

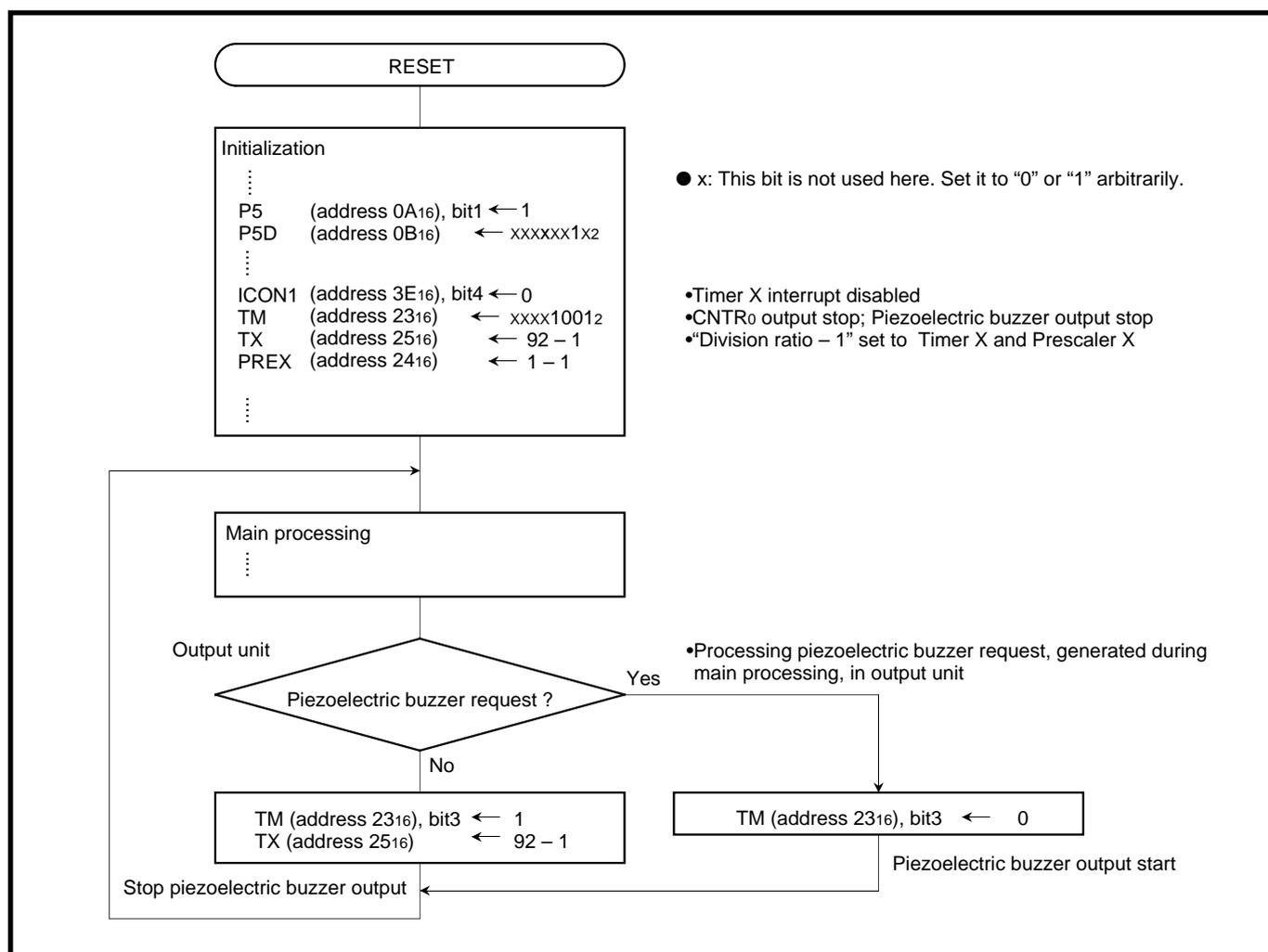


Fig. 2.3.16 Control procedure

**(4) Timer application example 3: Frequency measurement**

**Outline:** The following two values are compared to judge whether the frequency is within a valid range.

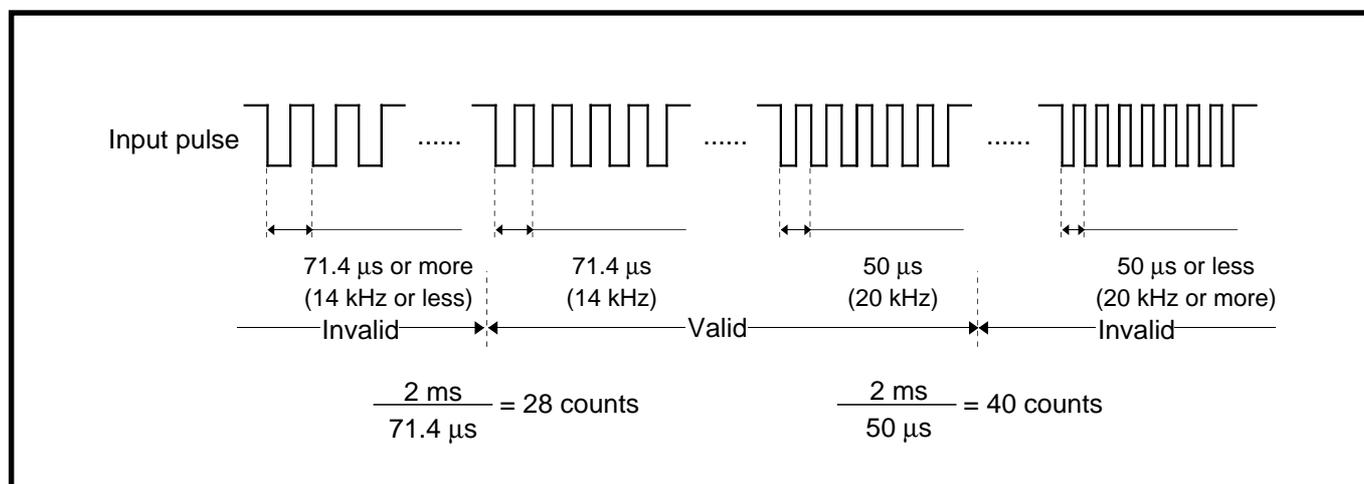
- A value by counting pulses input to P5<sub>1</sub>/CNTR<sub>0</sub> pin with the timer.
- A reference value

**Specifications:**

- The pulse is input to the P5<sub>1</sub>/CNTR<sub>0</sub> pin and counted by the timer X.
- A count value is read out at about 2 ms intervals, the timer 1 interrupt interval. When the count value is 28 to 40, it is judged that the input pulse is valid.
- Because the timer is a down-counter, the count value is compared with 227 to 215 (Note).

**Note:** 227 to 215 = {255 (initial value of counter) – 28} to {255 – 40}; 28 to 40 means the number of valid value.

Figure 2.3.17 shows the judgment method of valid/invalid of input pulses; Figure 2.3.18 shows the related registers setting; Figure 2.3.19 shows the control procedure.



**Fig. 2.3.17 Judgment method of valid/invalid of input pulses**

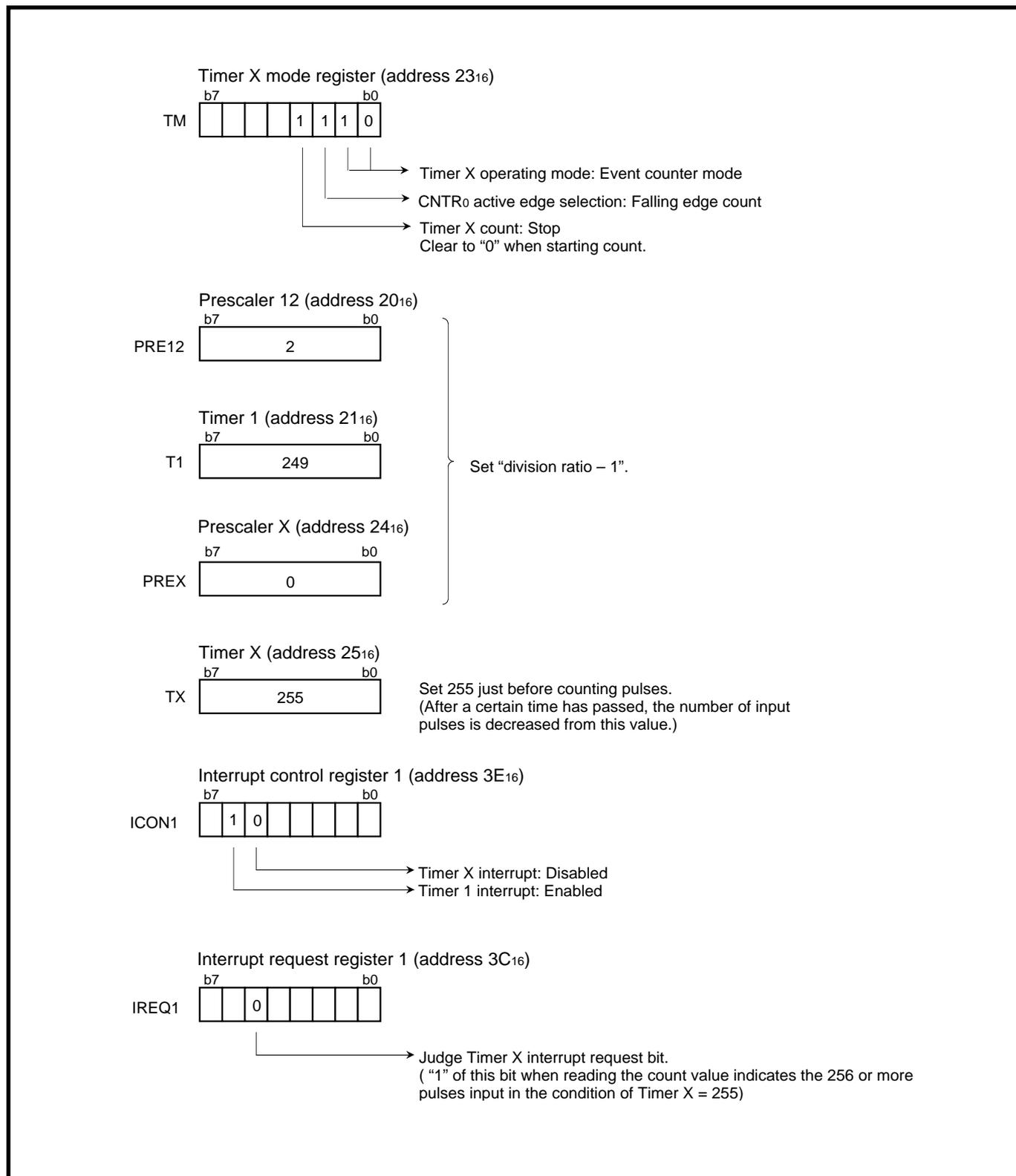


Fig. 2.3.18 Related registers setting

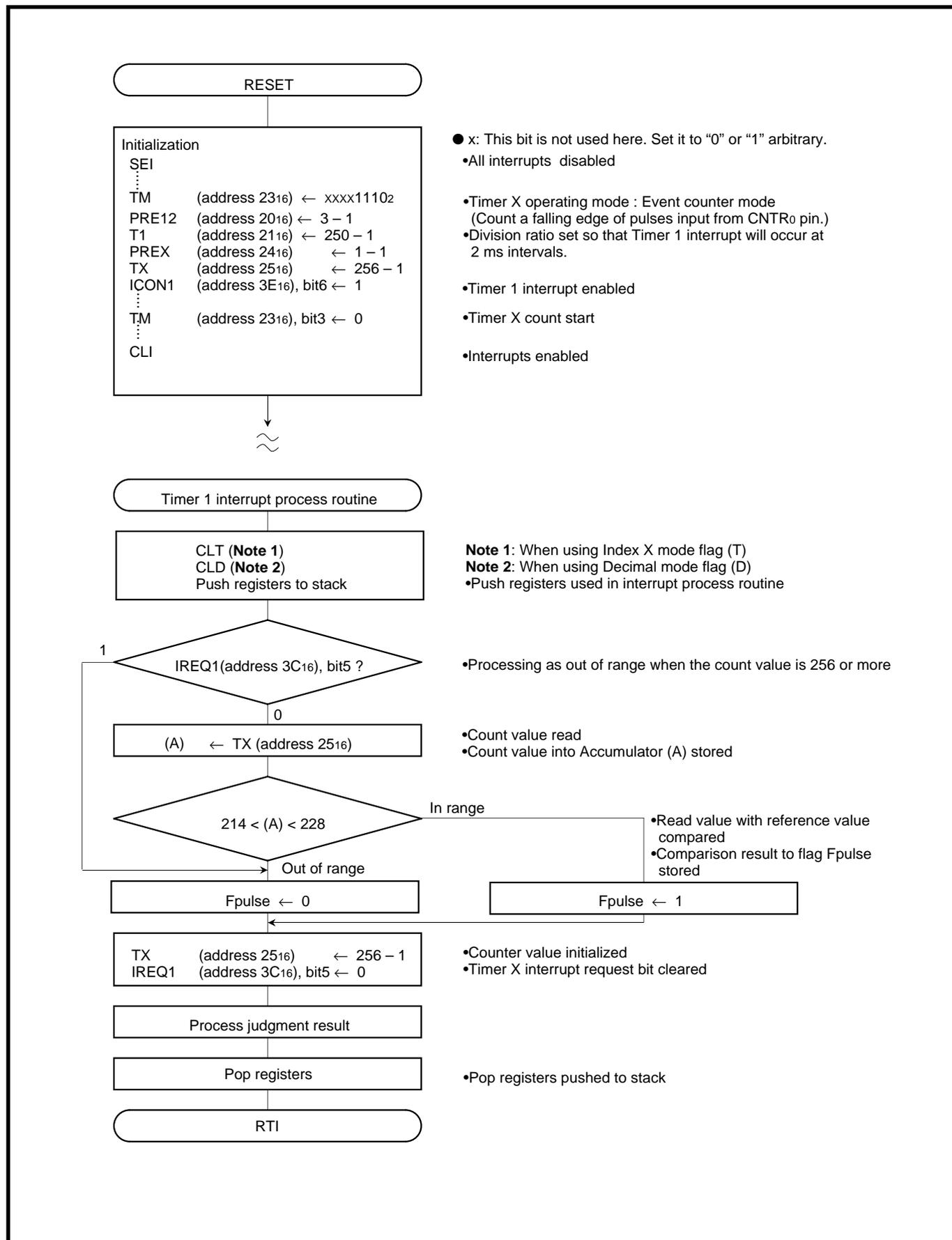


Fig. 2.3.19 Control procedure

**(5) Timer application example 4: Measurement of FG pulse width for motor**

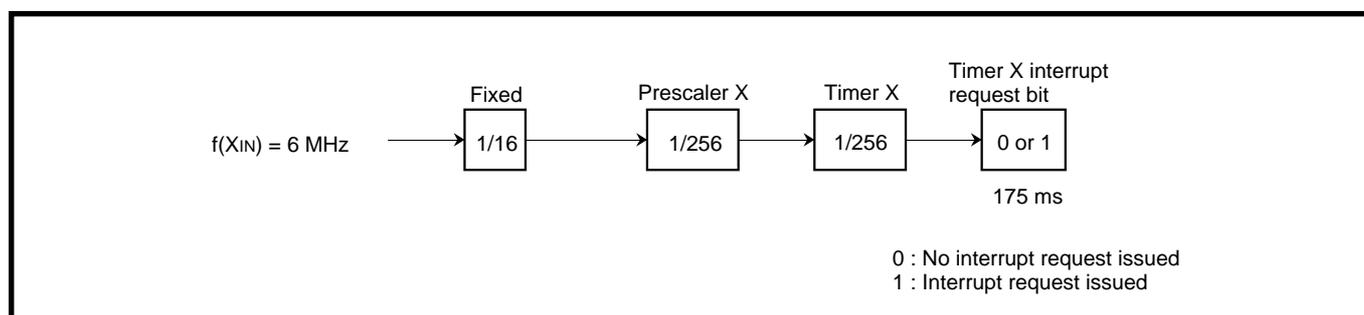
**Outline:** The timer X counts the “H” level width of the pulses input to the P5<sub>1</sub>/CNTR<sub>0</sub> pin. An underflow is detected by the timer X interrupt and an end of the input pulse “H” level is detected by the CNTR<sub>0</sub> interrupt.

**Specifications:** •The timer X counts the “H” level width of the FG pulse input to the P5<sub>1</sub>/CNTR<sub>0</sub> pin.

<Example>

When the clock frequency is 6 MHz, the count source is 2.67  $\mu$ s, which is obtained by dividing the clock frequency by 16. Measurement can be performed to 175 ms in the range of FFFF<sub>16</sub> to 0000<sub>16</sub>.

Figure 2.3.20 shows the timers connection and setting of division ratio; Figure 2.3.21 shows the related registers setting; Figure 2.3.22 shows the control procedure.



**Fig. 2.3.20 Timers connection and setting of division ratios**

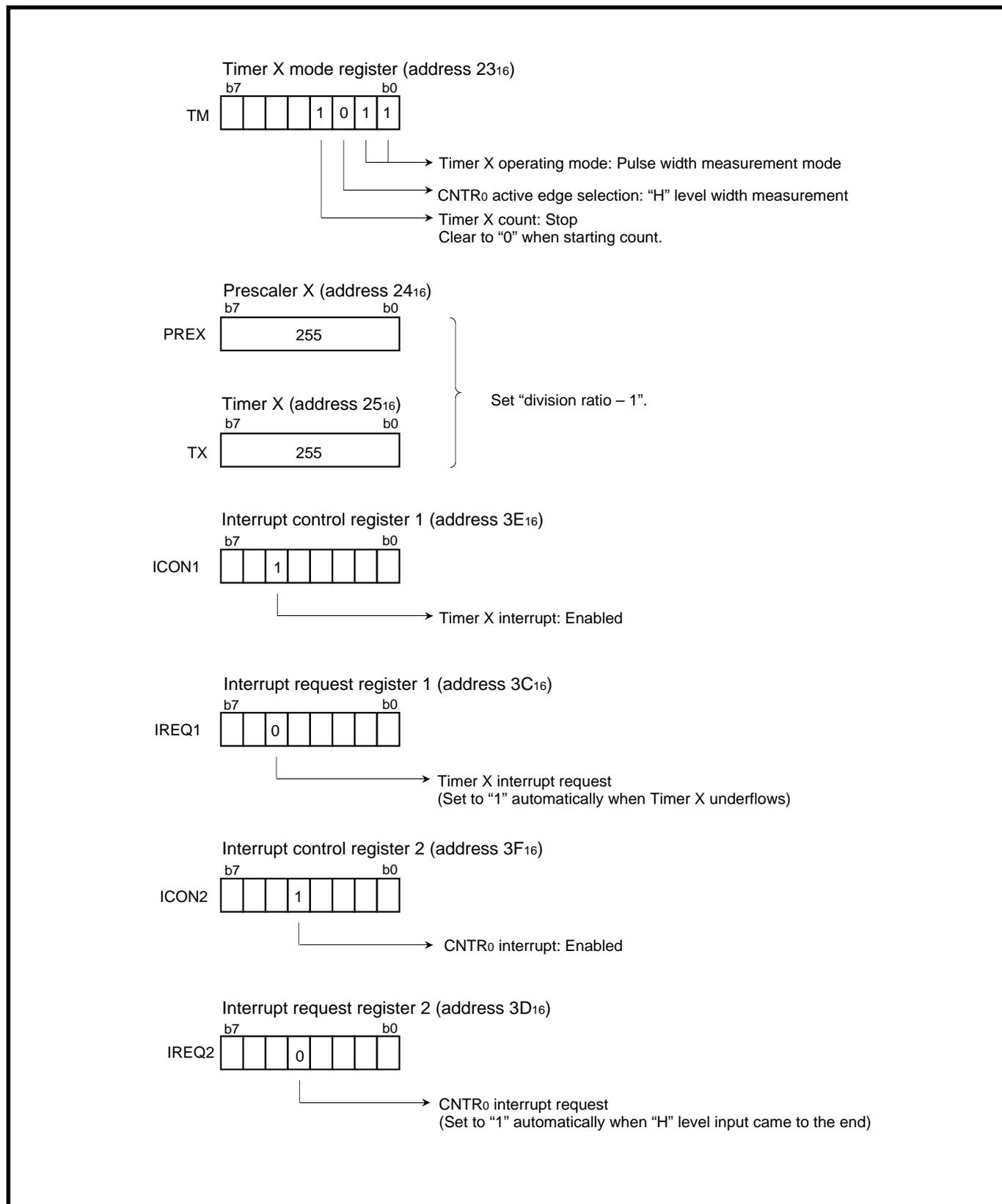


Fig. 2.3.21 Related registers setting

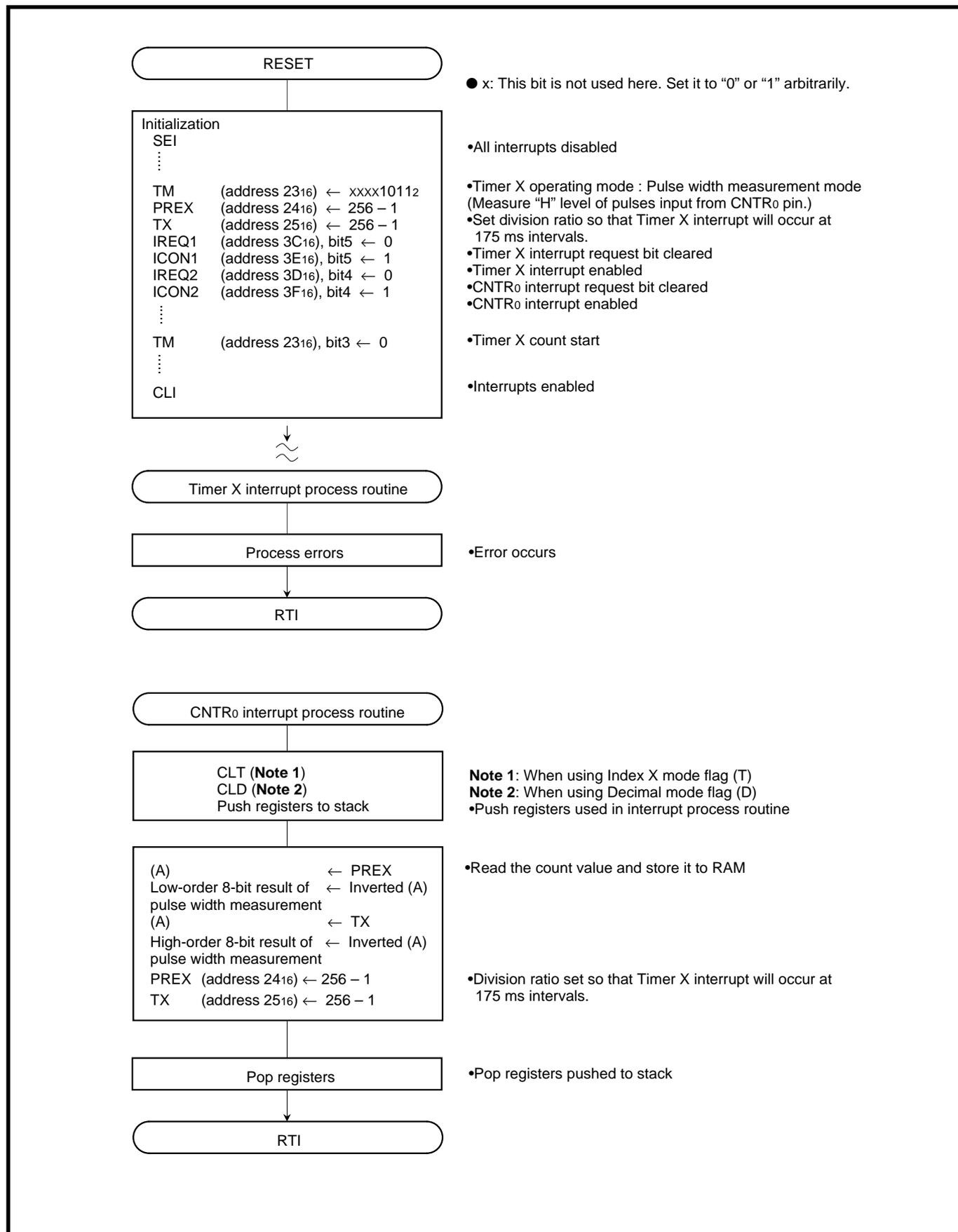


Fig. 2.3.22 Control procedure

#### 2.3.4 Notes on timer

- If a value  $n$  (between 0 and 255) is written to a timer latch, the frequency division ratio is  $1/(n+1)$ .
- When switching the count source by the timer X count source selection bit, the value of timer count is altered in unconsiderable amount owing to generating of a thin pulses in the count input signals. Therefore, select the timer count source before set the value to the prescaler and the timer.

## 2.4 Serial I/O

This paragraph explains the registers setting method and the notes related to the Serial I/O.

### 2.4.1 Memory map

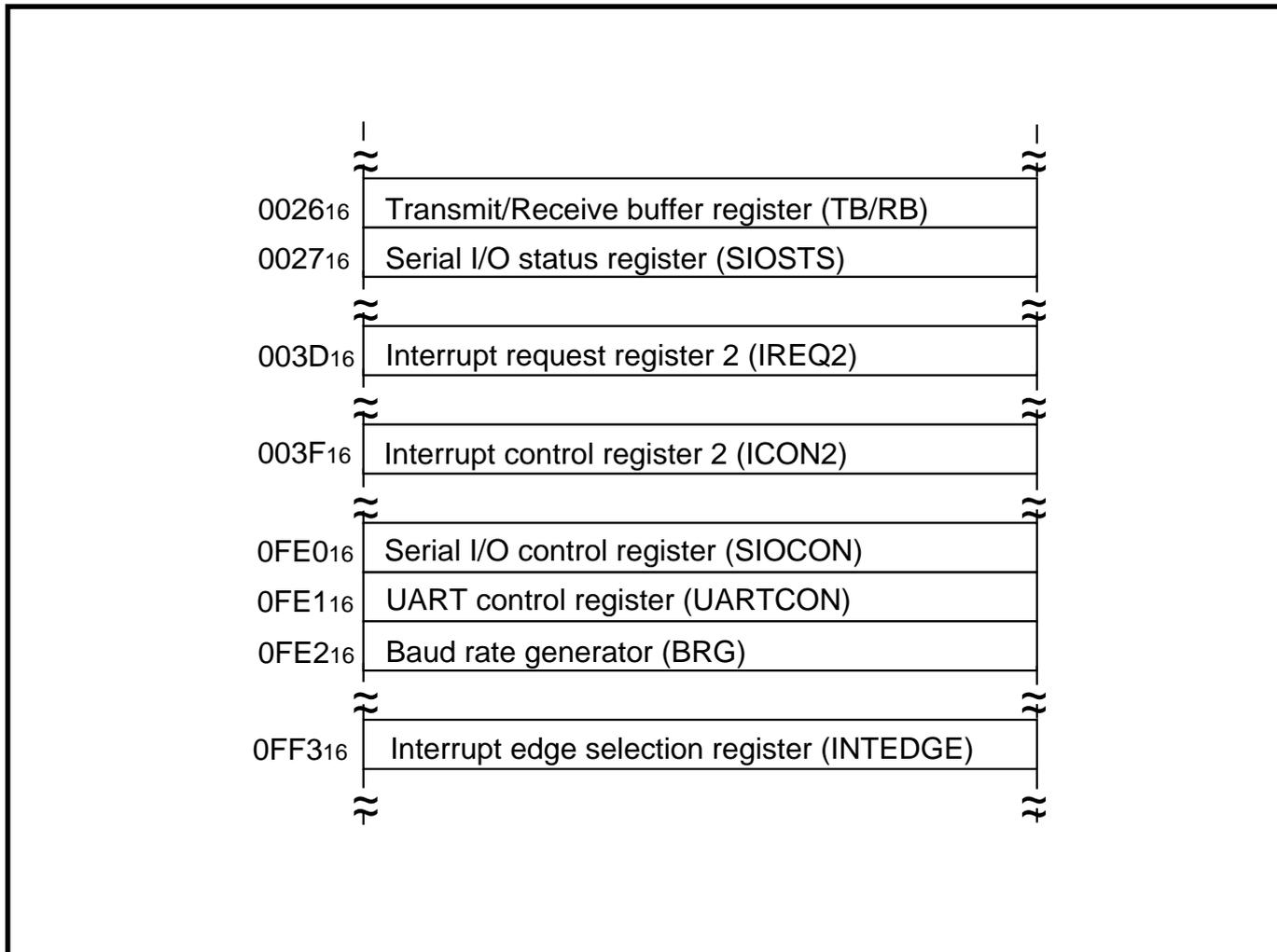


Fig. 2.4.1 Memory map of registers related to Serial I/O

## 2.4.2 Related registers

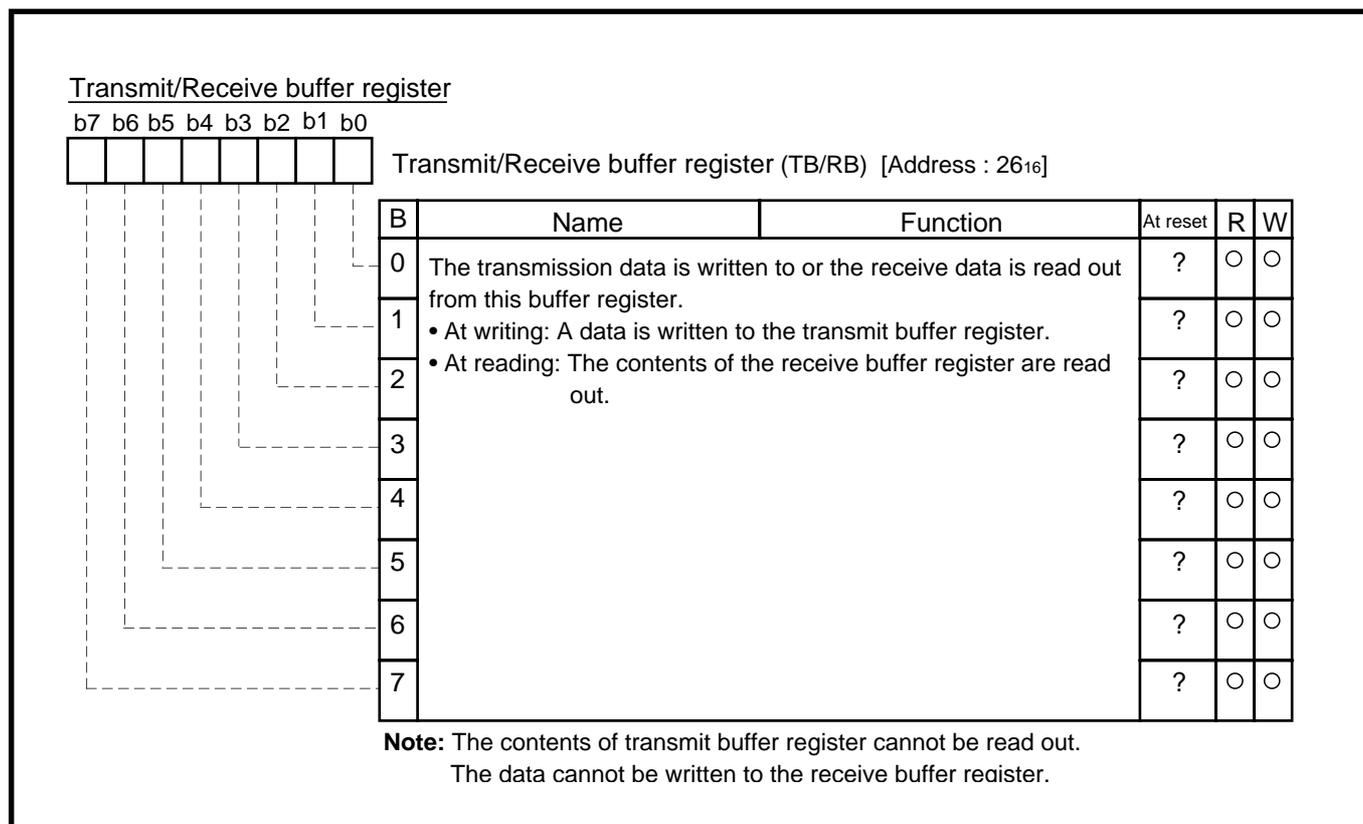


Fig. 2.4.2 Structure of Transmit/Receive buffer register

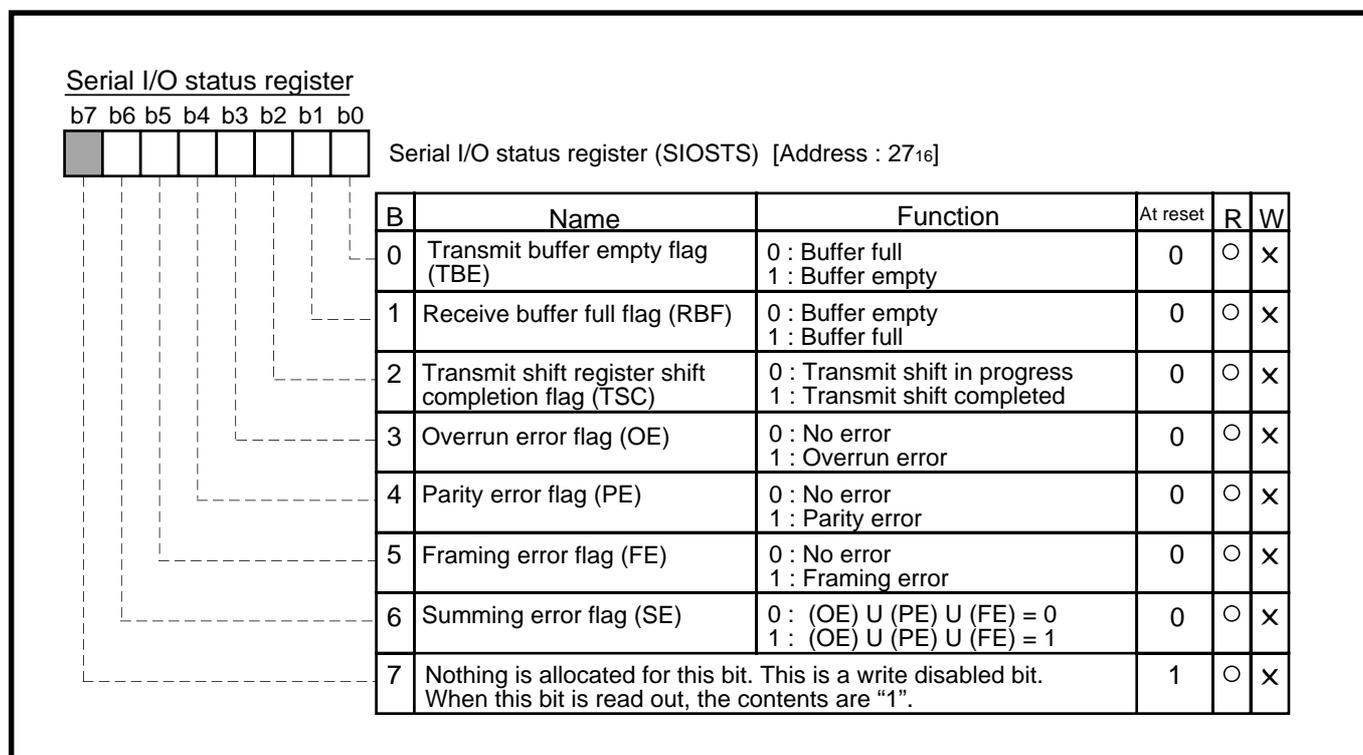


Fig. 2.4.3 Structure of Serial I/O status register

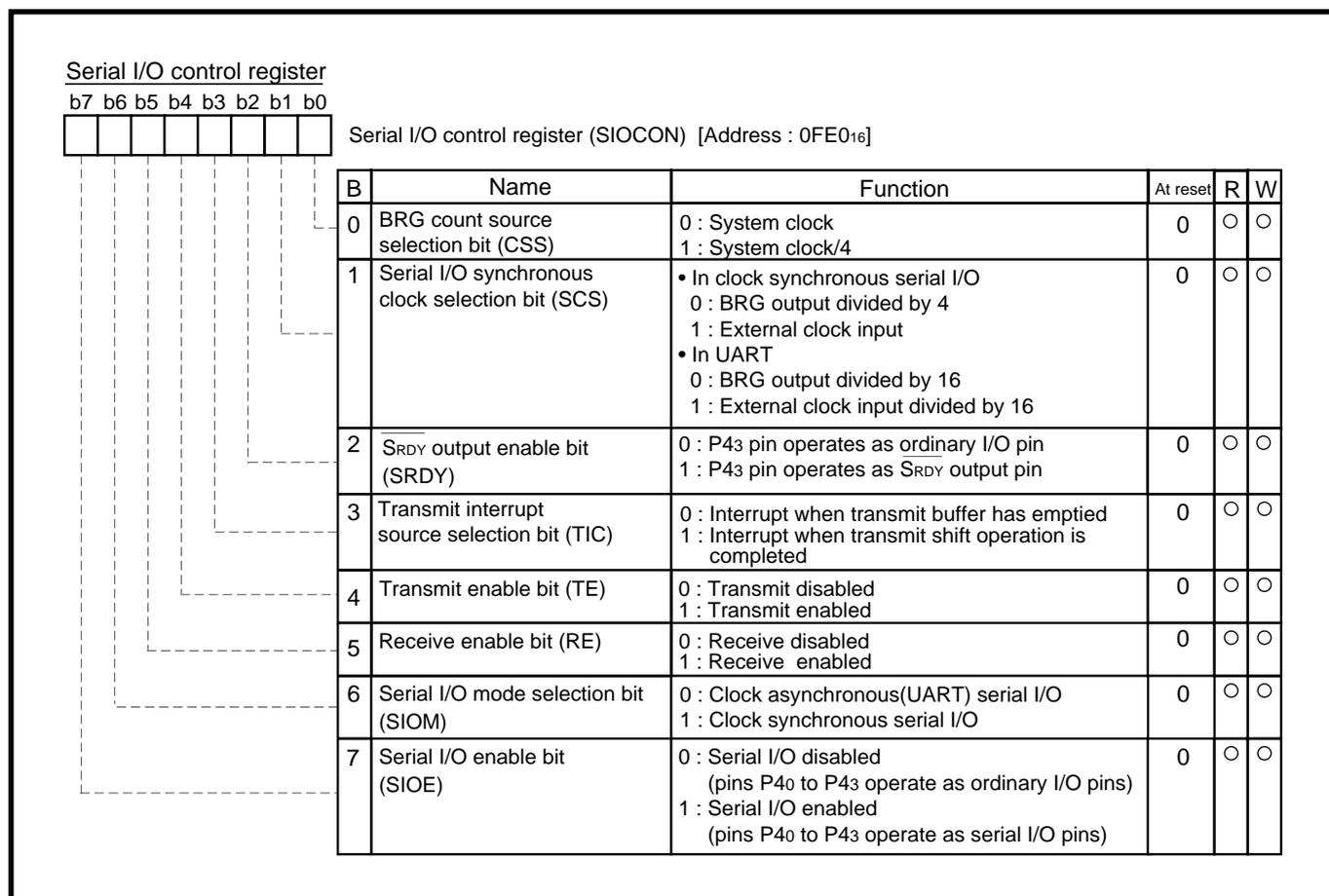


Fig. 2.4.4 Structure of Serial I/O control register

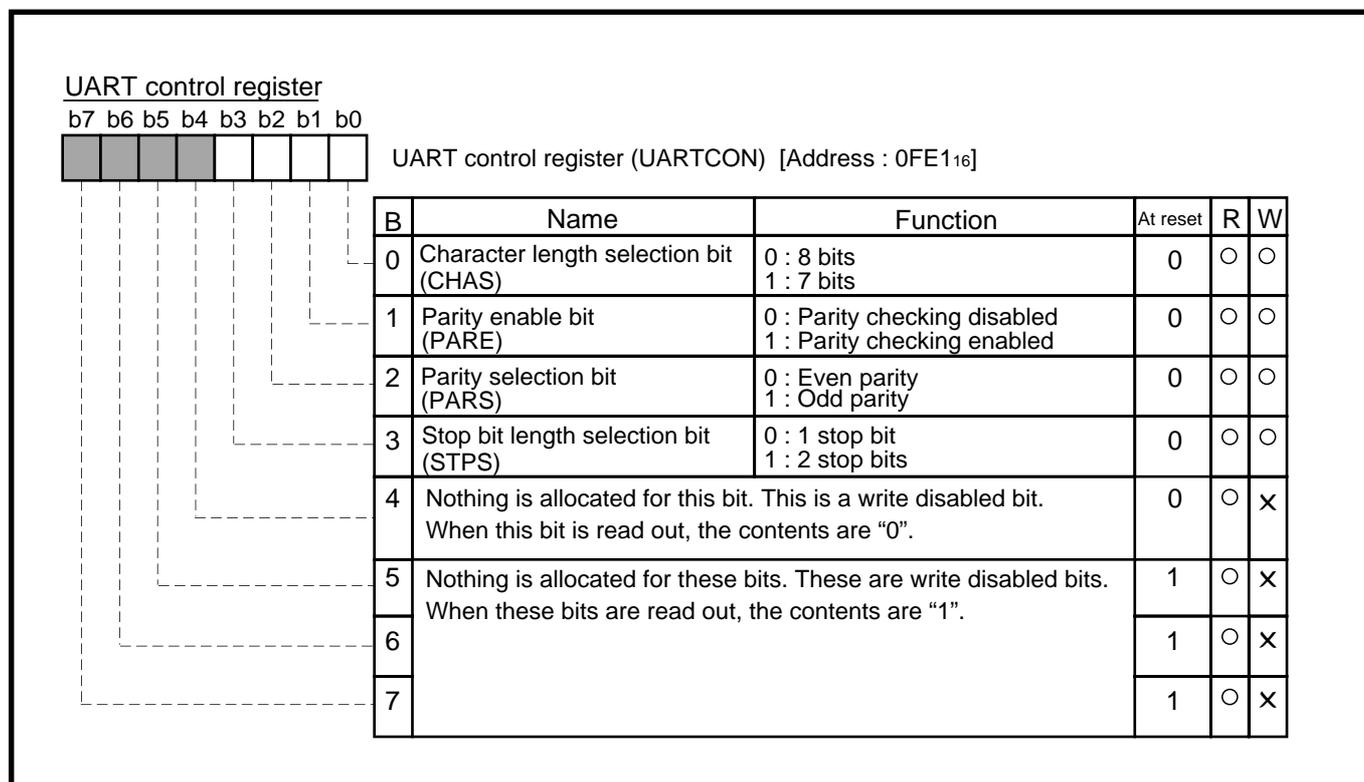


Fig. 2.4.5 Structure of UART control register

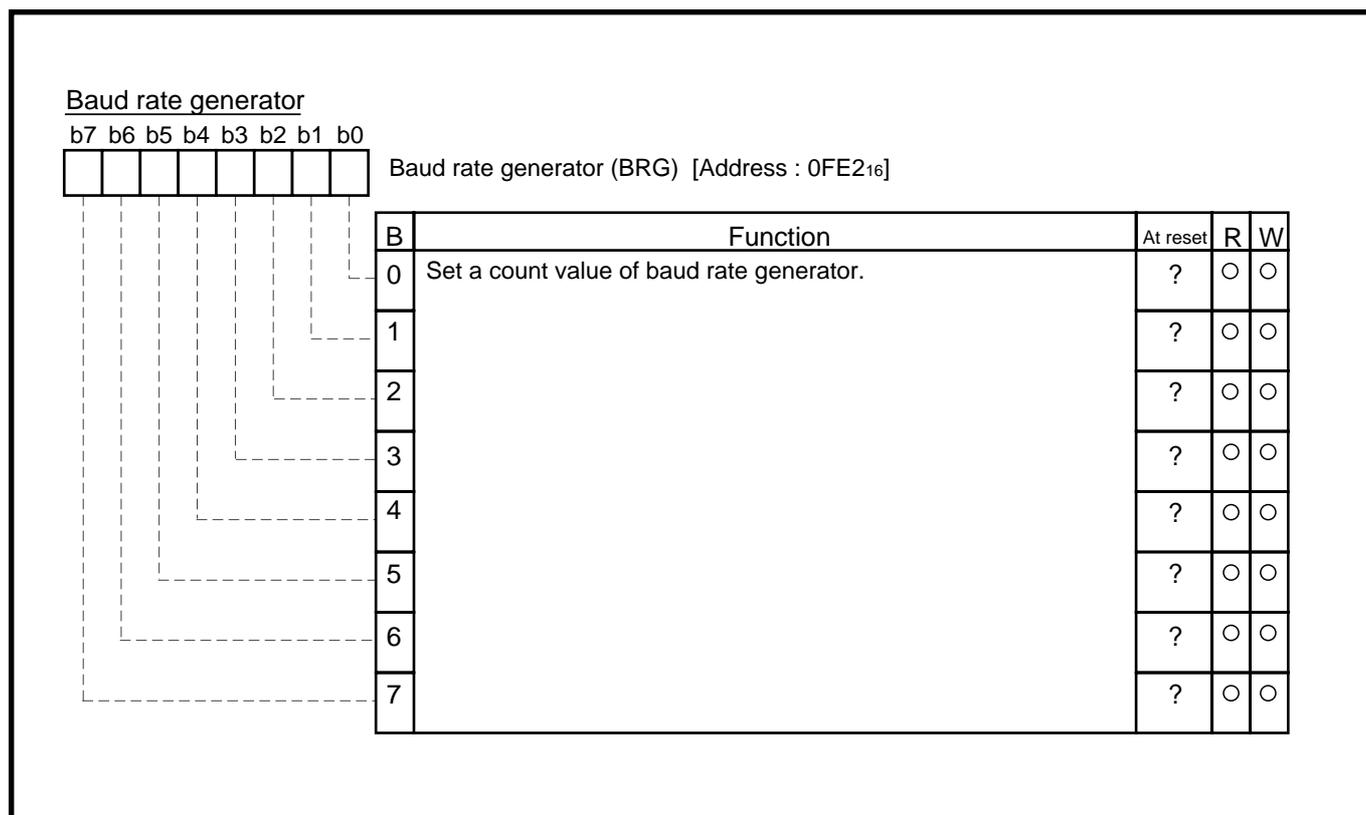


Fig. 2.4.6 Structure of Baud rate generator

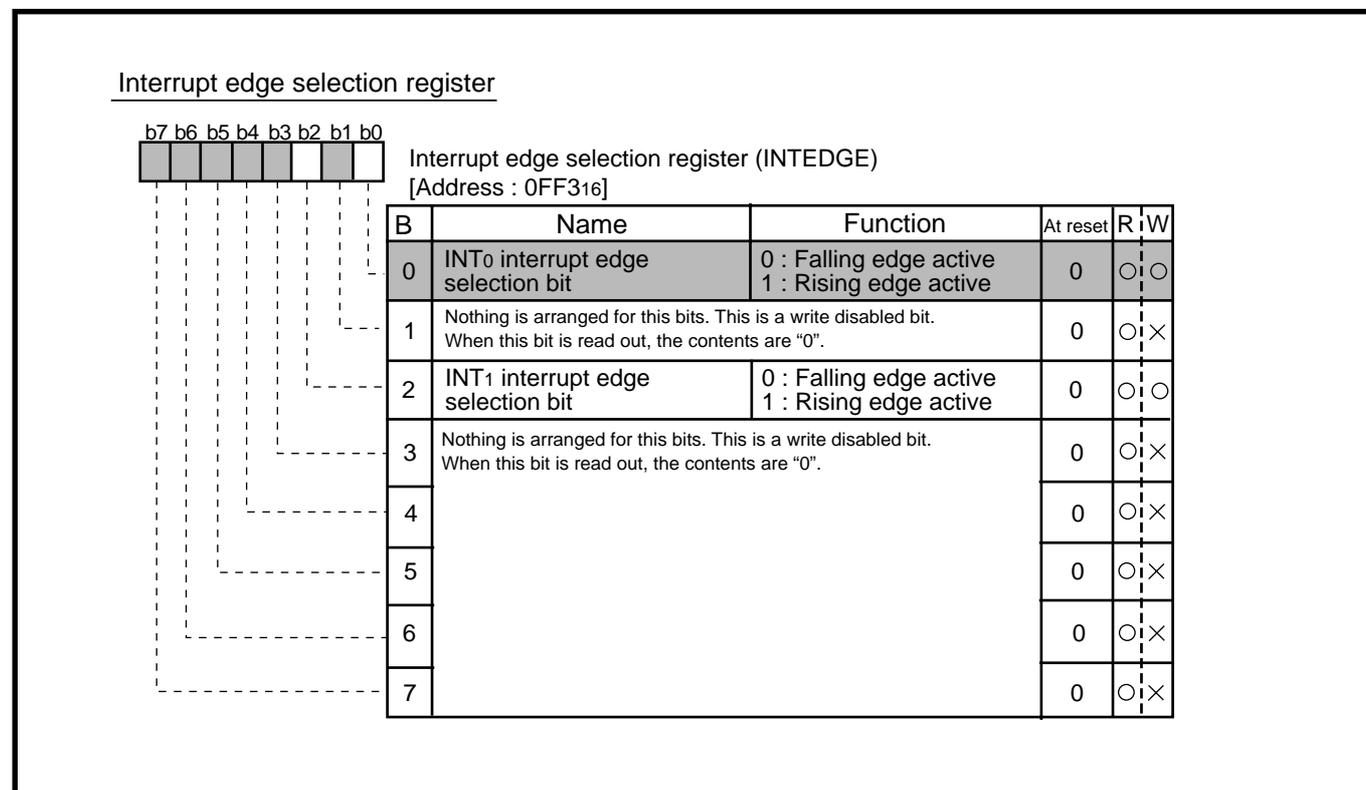


Fig. 2.4.7 Structure of Interrupt edge selection register

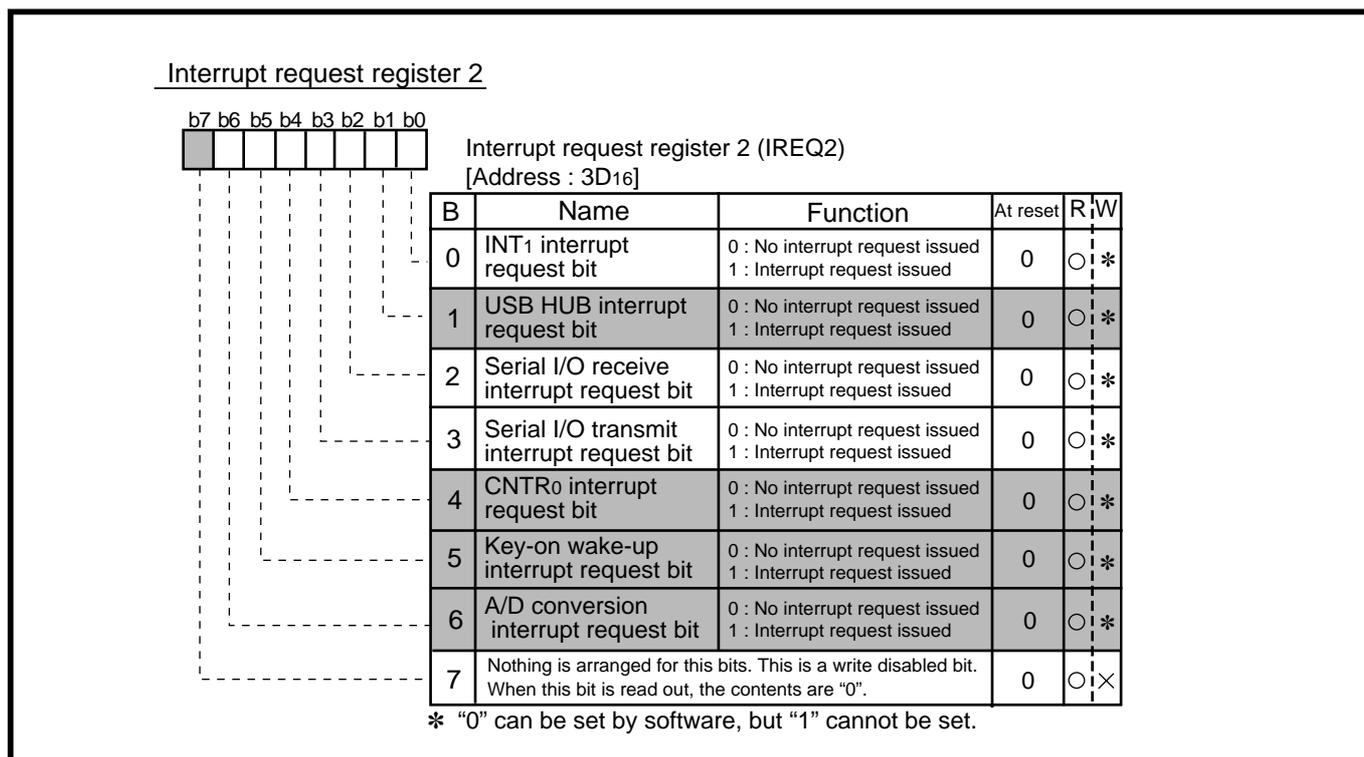


Fig. 2.4.8 Structure of Interrupt request register 2

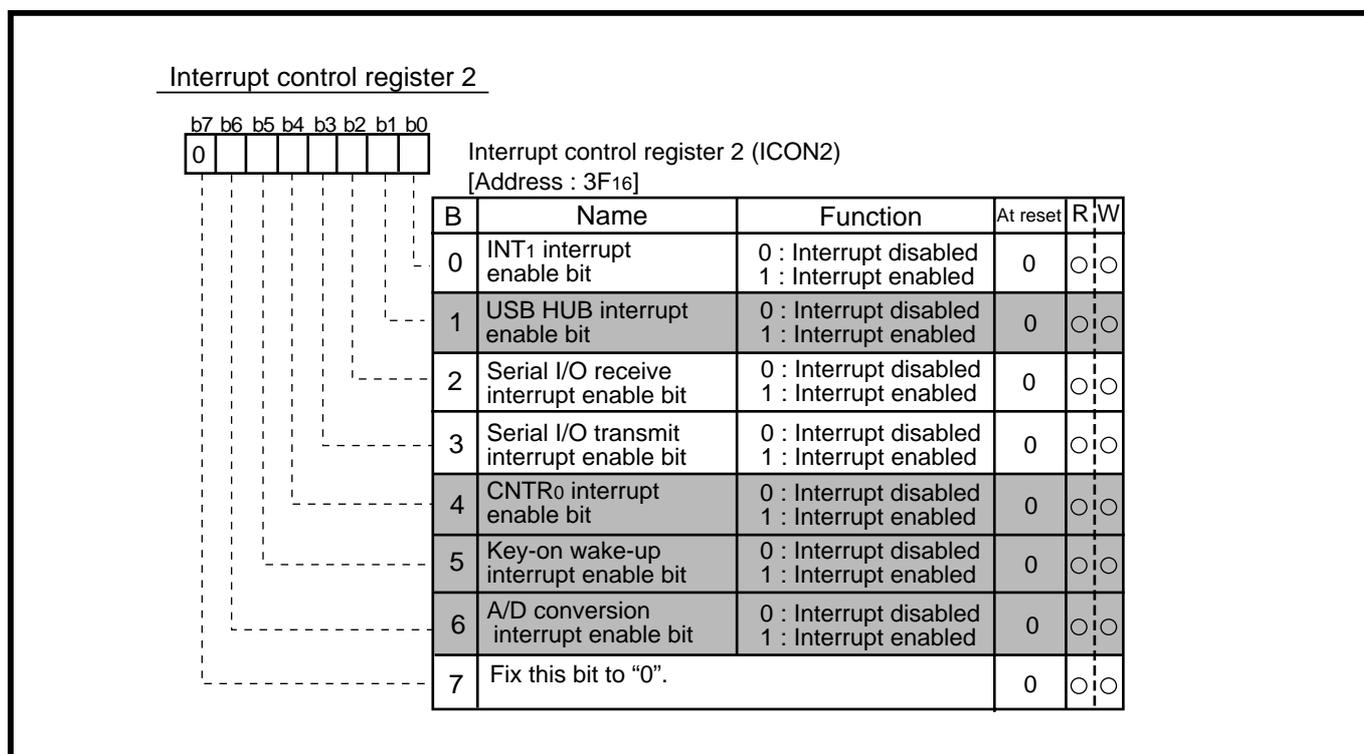


Fig. 2.4.9 Structure of Interrupt control register 2

### 2.4.3 Serial I/O connection examples

#### (1) Control of peripheral IC equipped with CS pin

Figure 2.4.10 shows connection examples of a peripheral IC equipped with the CS pin. There are connection examples using a clock synchronous serial I/O mode.

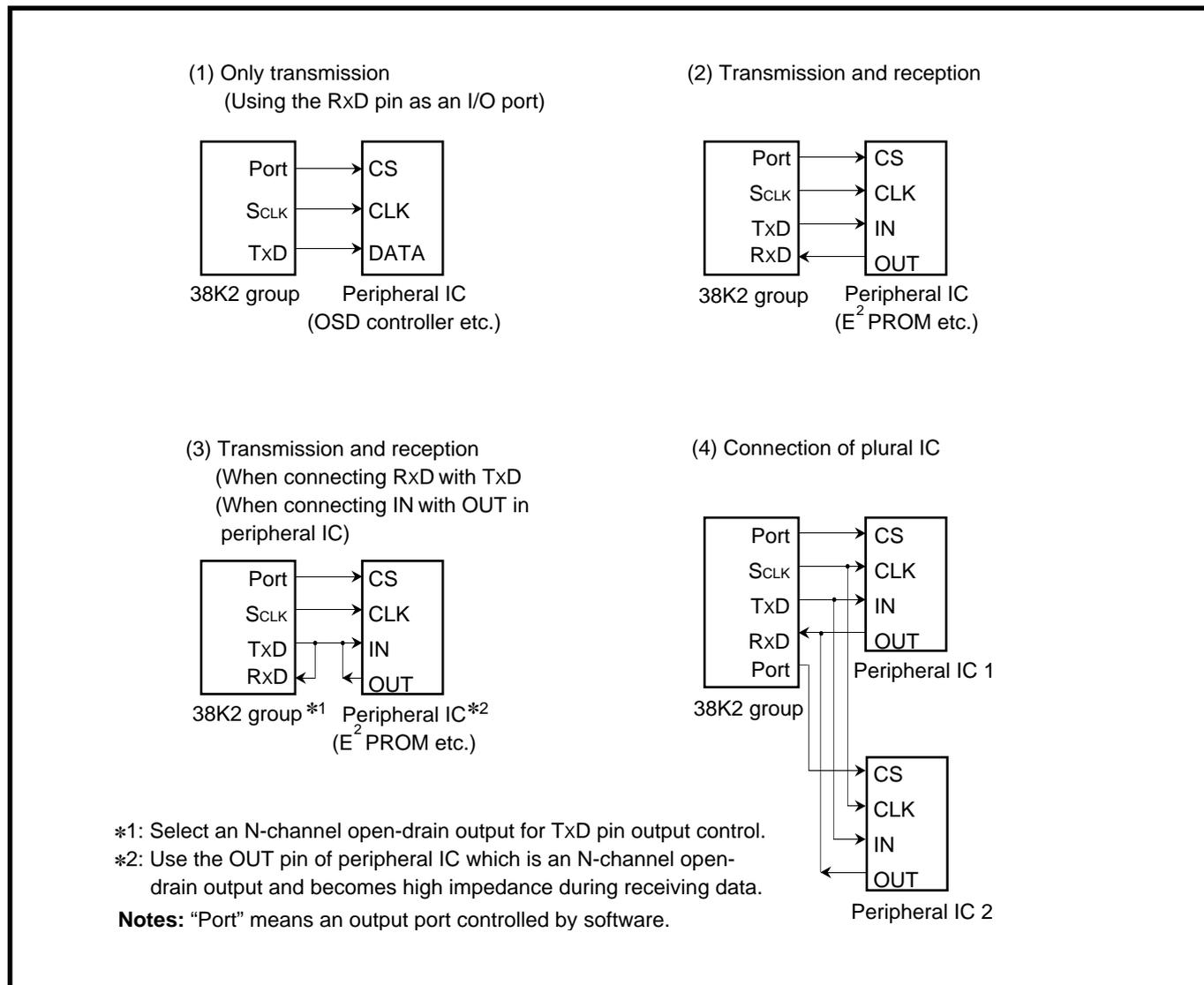


Fig. 2.4.10 Serial I/O connection examples (1)

**(2) Connection with microcomputer**

Figure 2.4.11 shows connection examples with another microcomputer.

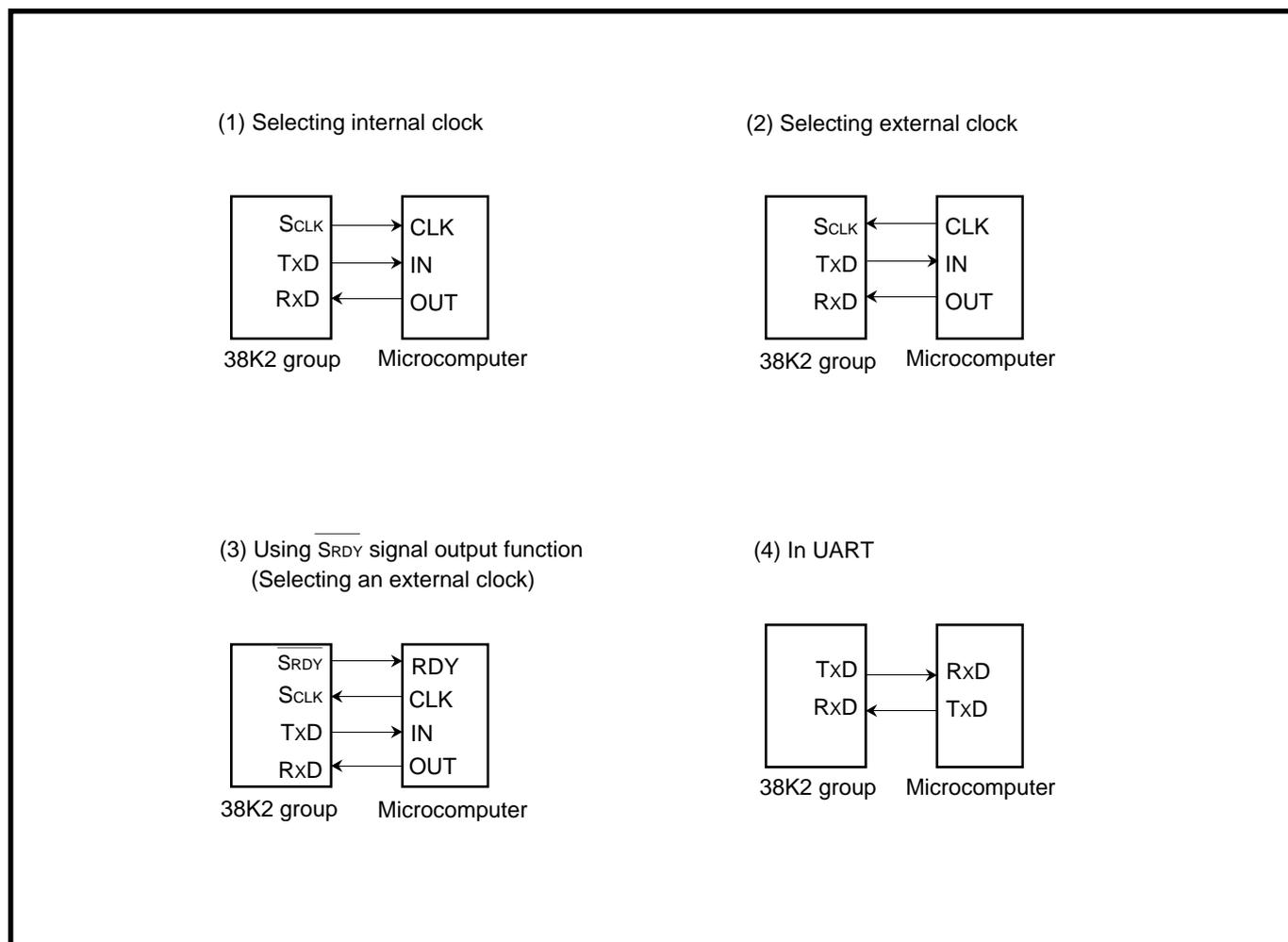


Fig. 2.4.11 Serial I/O connection examples (2)

### 2.4.4 Setting of serial I/O transfer data format

A clock synchronous or clock asynchronous (UART) can be selected as a data format of Serial I/O. Figure 2.4.12 shows the serial I/O transfer data format.

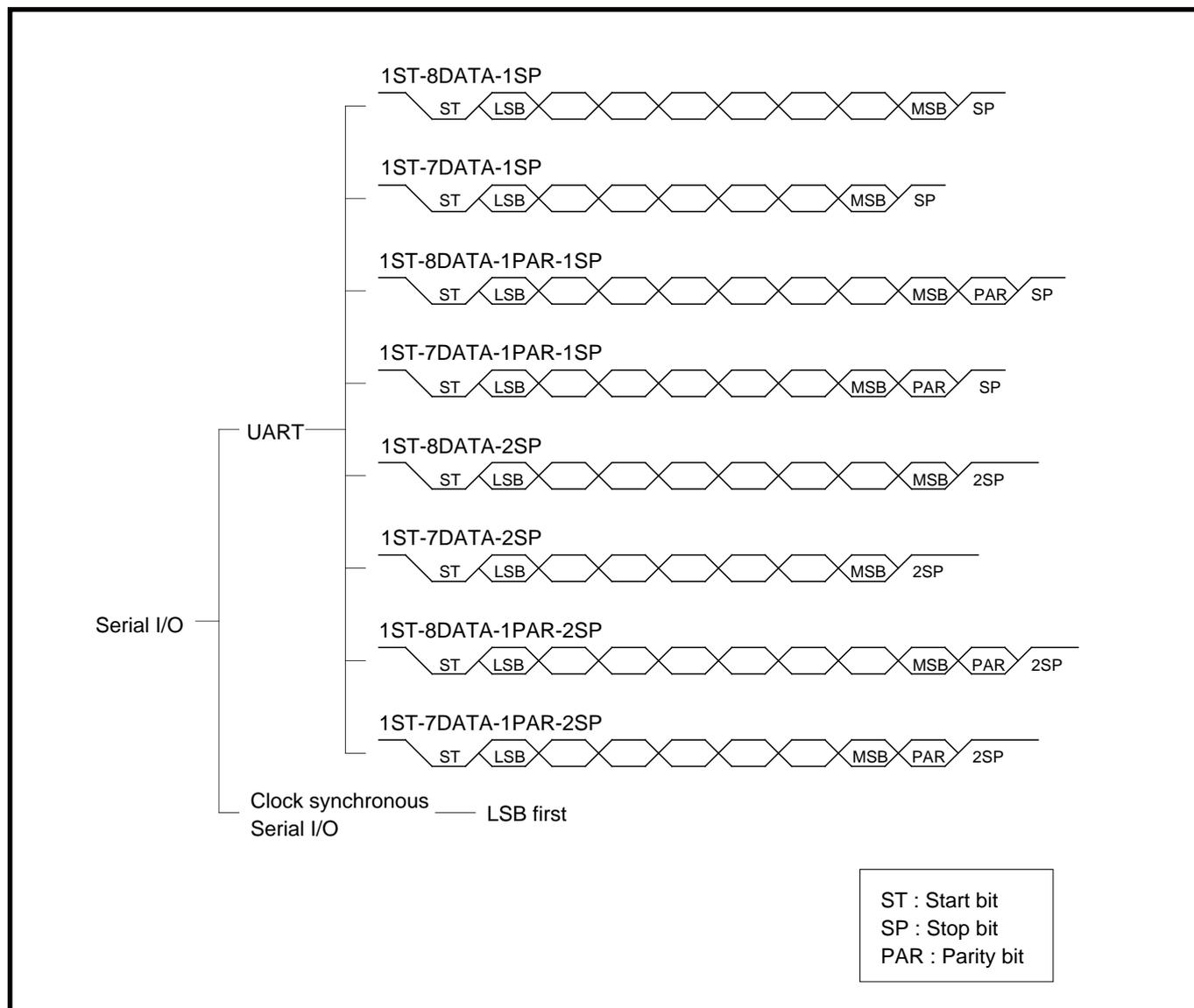


Fig. 2.4.12 Serial I/O transfer data format

### 2.4.5 Serial I/O application examples

#### (1) Communication using clock synchronous serial I/O (transmit/receive)

**Outline :** 2-byte data is transmitted and received, using the clock synchronous serial I/O.  
The  $\overline{\text{SRDY}}$  signal is used for communication control.

Figure 2.4.13 shows a connection diagram, and Figure 2.4.14 shows a timing chart.

Figure 2.4.15 shows a registers setting related to the transmitting side, and Figure 2.4.16 shows registers setting related to the receiving side.

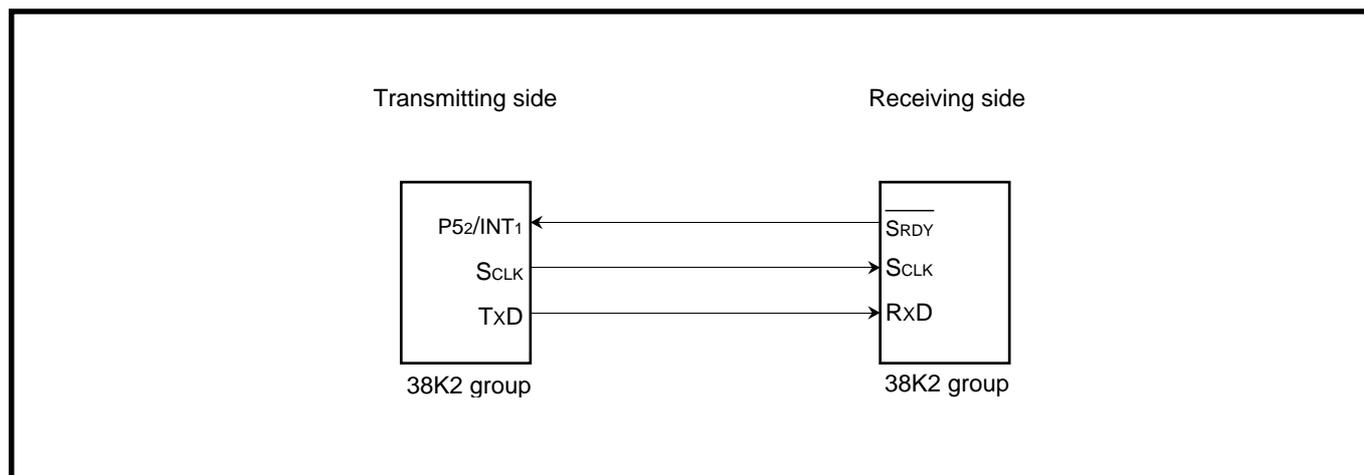


Fig. 2.4.13 Connection diagram

- Specifications :**
- The Serial I/O is used (clock synchronous serial I/O is selected.)
  - Synchronous clock frequency : 125 kHz ( $f(\text{X}_{\text{IN}}) = 6 \text{ MHz}$  is divided by 48)
  - The  $\overline{\text{SRDY}}$  (receivable signal) is used.
  - The receiving side outputs the  $\overline{\text{SRDY}}$  signal at intervals of 2 ms (generated by timer), and 2-byte data is transferred from the transmitting side to the receiving side.

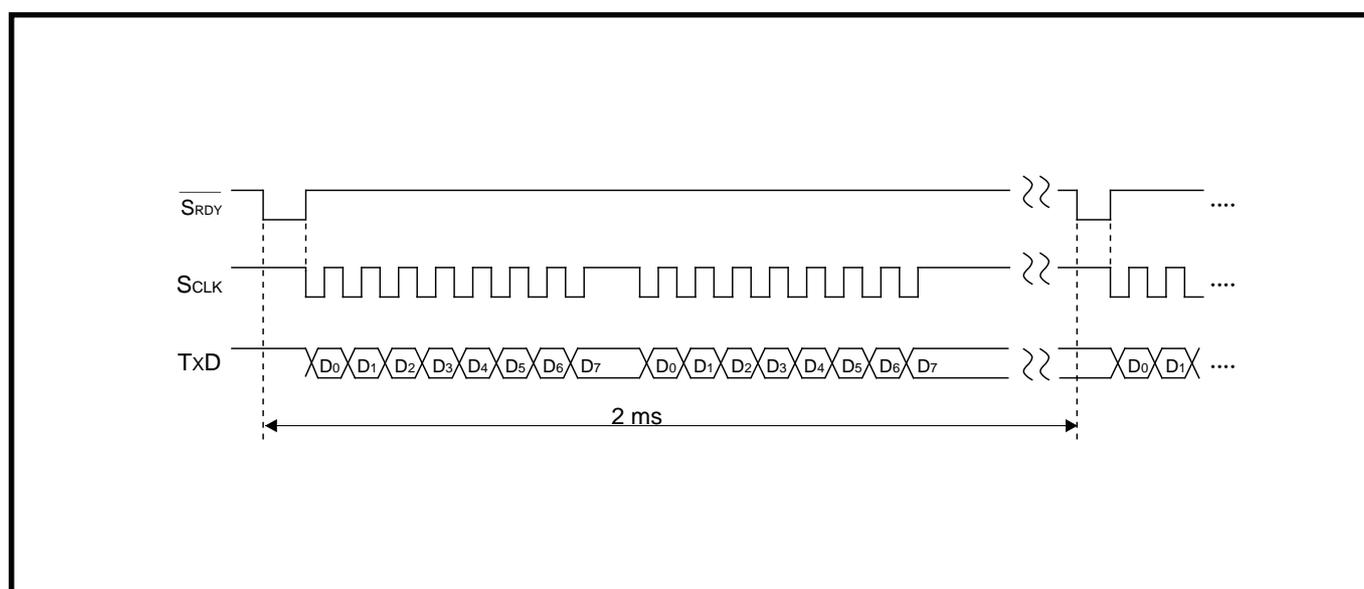


Fig. 2.4.14 Timing chart

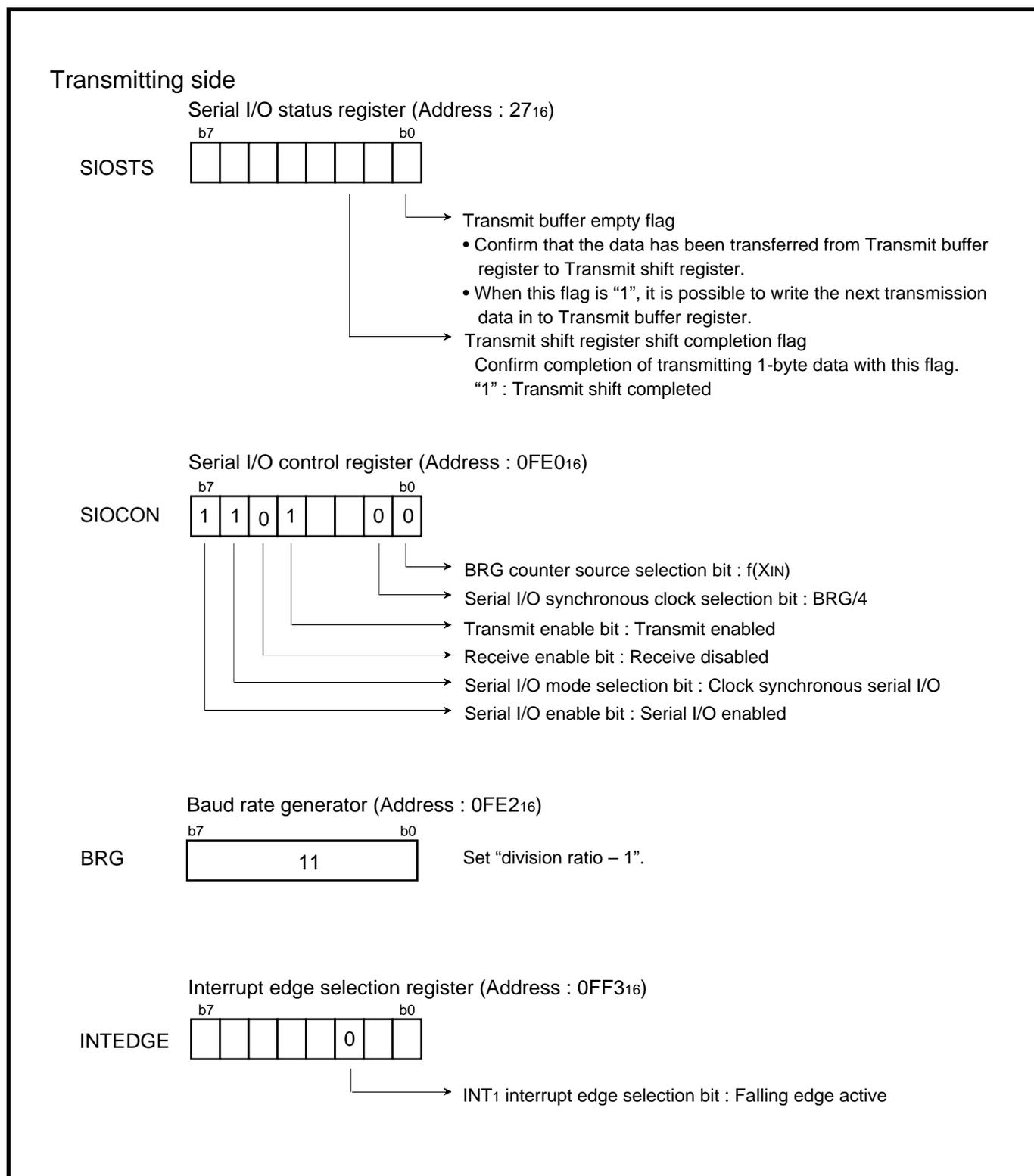


Fig. 2.4.15 Registers setting related to transmitting side

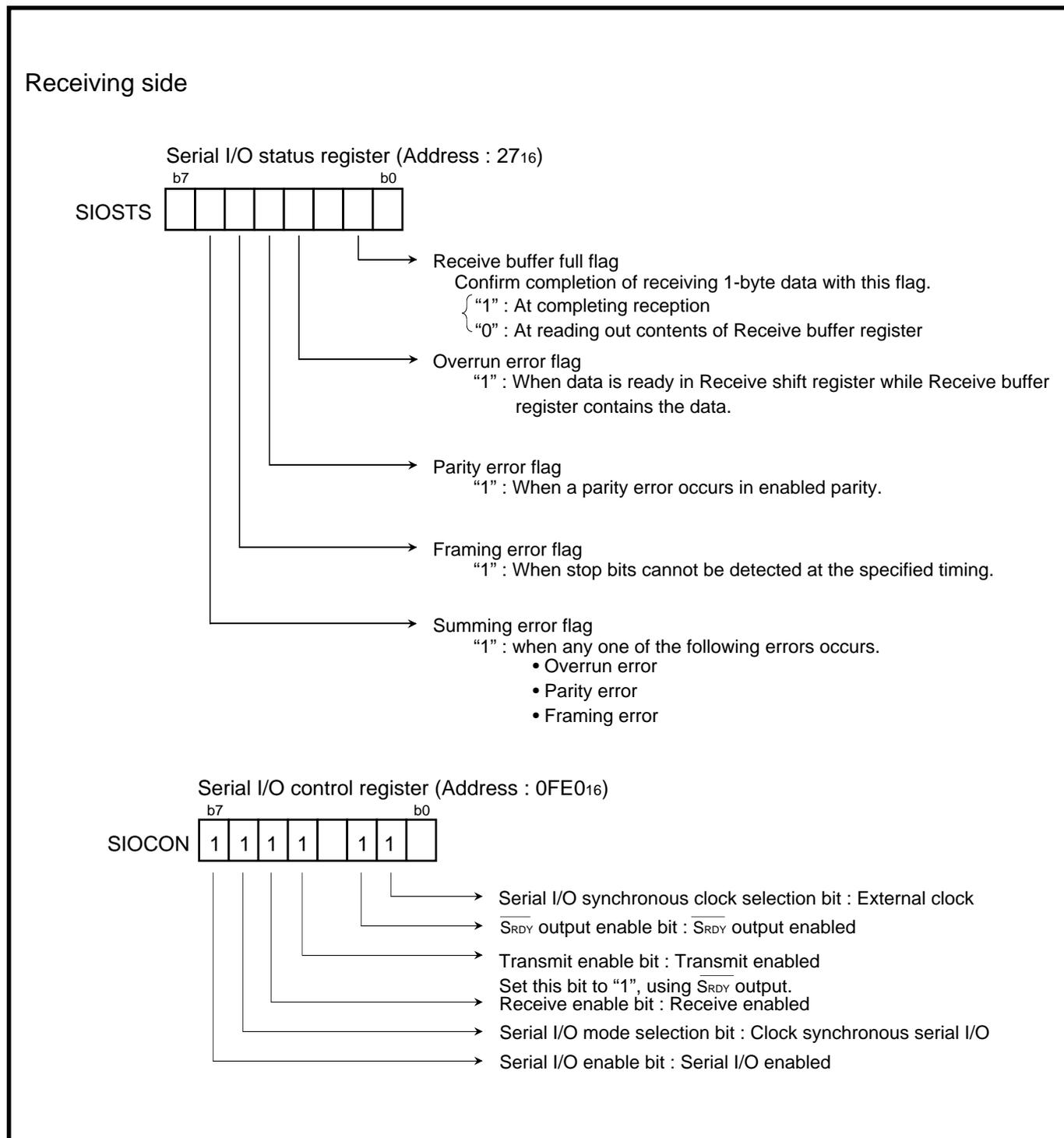


Fig. 2.4.16 Registers setting related to receiving side

Figure 2.4.17 shows a control procedure of the transmitting side, and Figure 2.4.18 shows a control procedure of the receiving side.

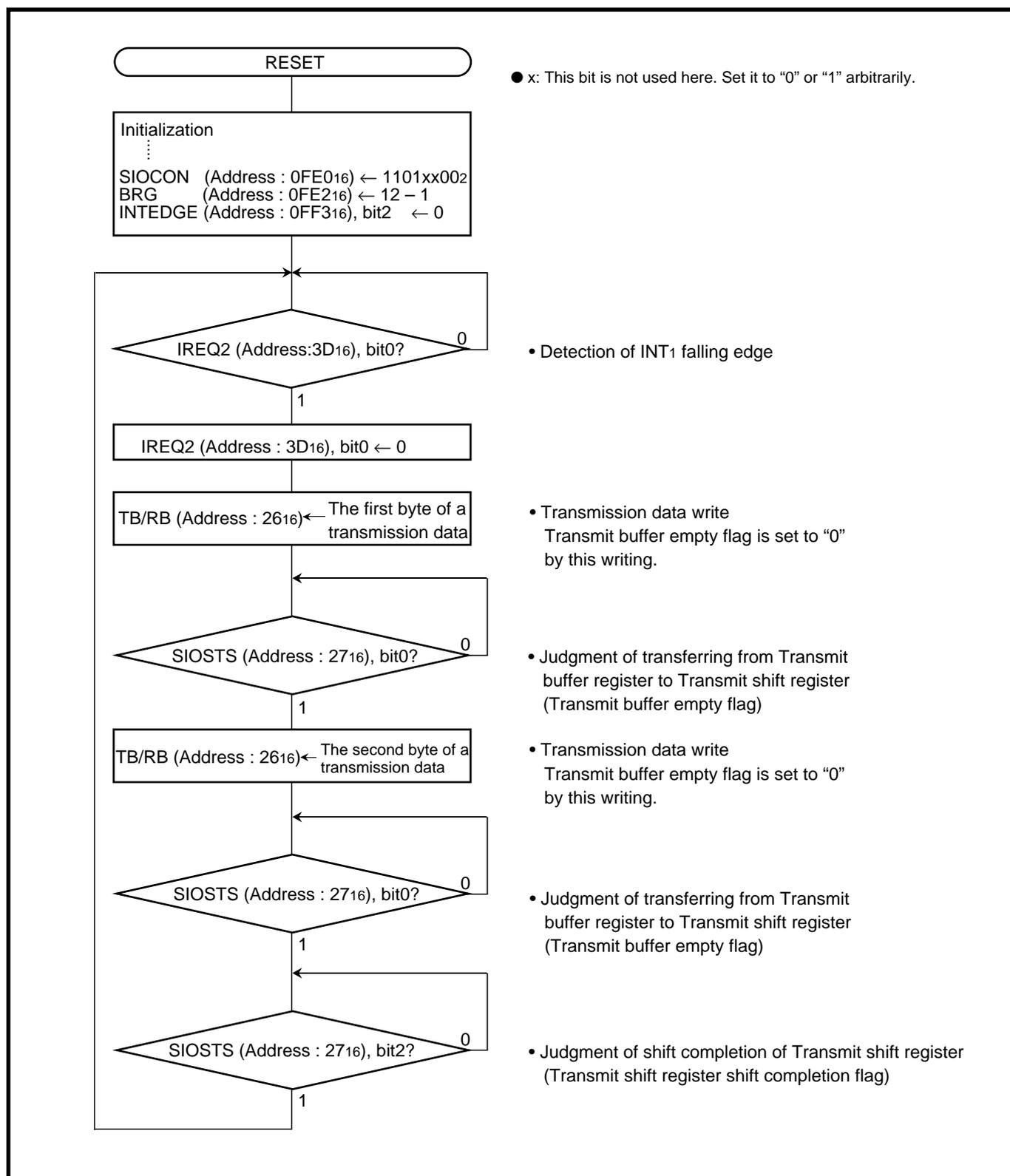


Fig. 2.4.17 Control procedure of transmitting side

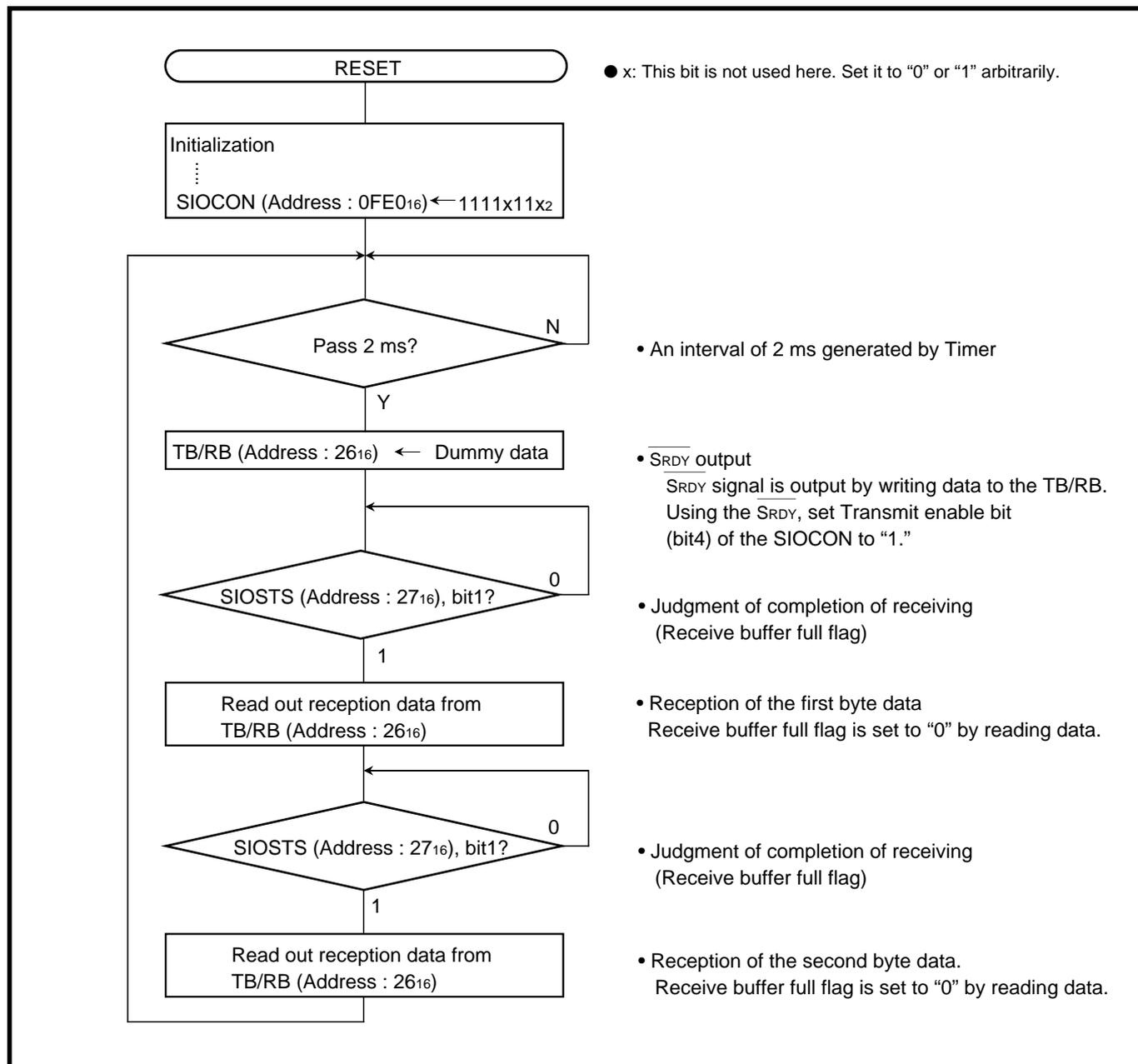


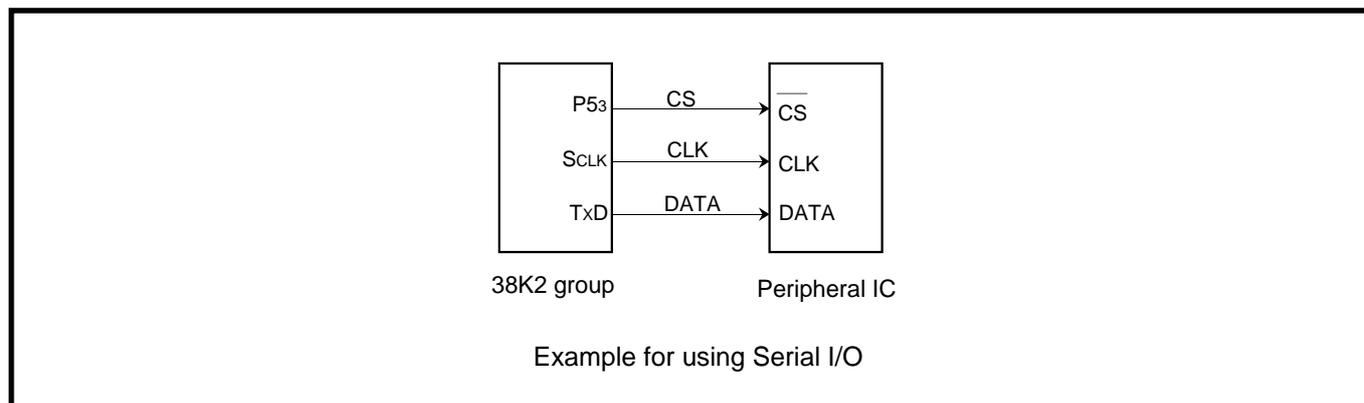
Fig. 2.4.18 Control procedure of receiving side

**(2) Output of serial data (control of peripheral IC)**

**Outline :** 4-byte data is transmitted and received, using the clock synchronous serial I/O.  
The CS signal is output to a peripheral IC through port P5<sub>3</sub>.

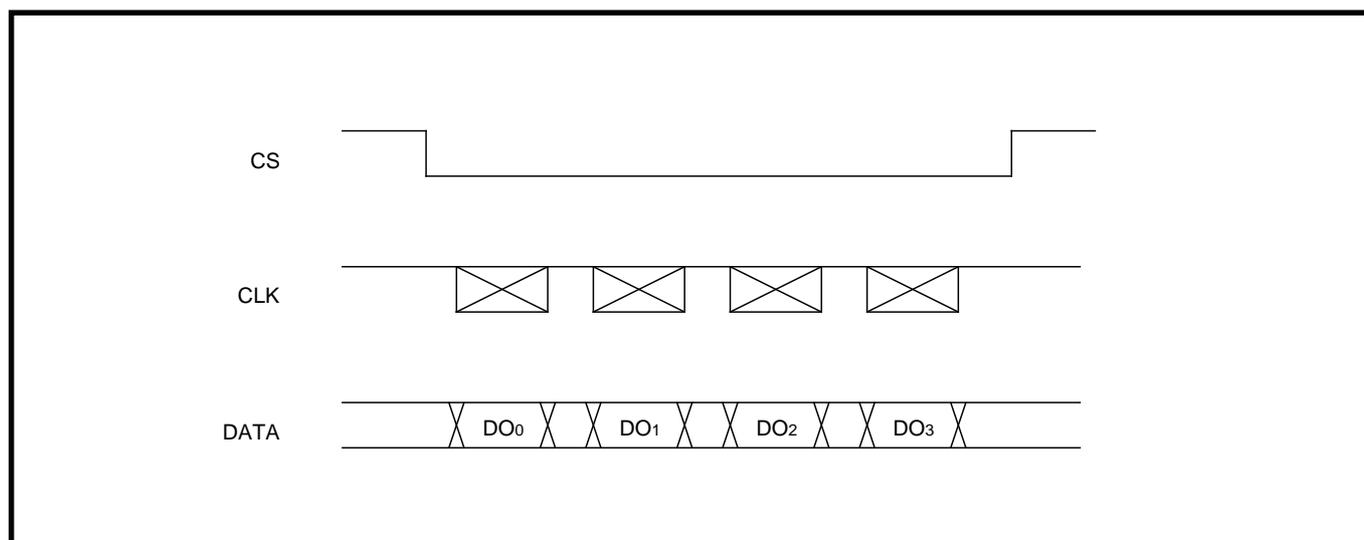
The example for using Serial I/O is shown.

Figure 2.4.19 shows a connection diagram, and Figure 2.4.20 shows a timing chart.



**Fig. 2.4.19 Connection diagram**

- Specifications :**
- The Serial I/O is used (clock synchronous serial I/O is selected.)
  - Synchronous clock frequency : 125 kHz ( $f(X_{IN}) = 6 \text{ MHz}$  is divided by 48)
  - Transfer direction : LSB first
  - The Serial I/O interrupt is not used.
  - Port P5<sub>3</sub> is connected to the  $\overline{CS}$  pin ("L" active) of the peripheral IC for transmission control; the output level of port P5<sub>3</sub> is controlled by software.



**Fig. 2.4.20 Timing chart**

Figure 2.4.21 shows registers setting related to Serial I/O, and Figure 2.4.22 shows a setting of serial I/O transmission data.

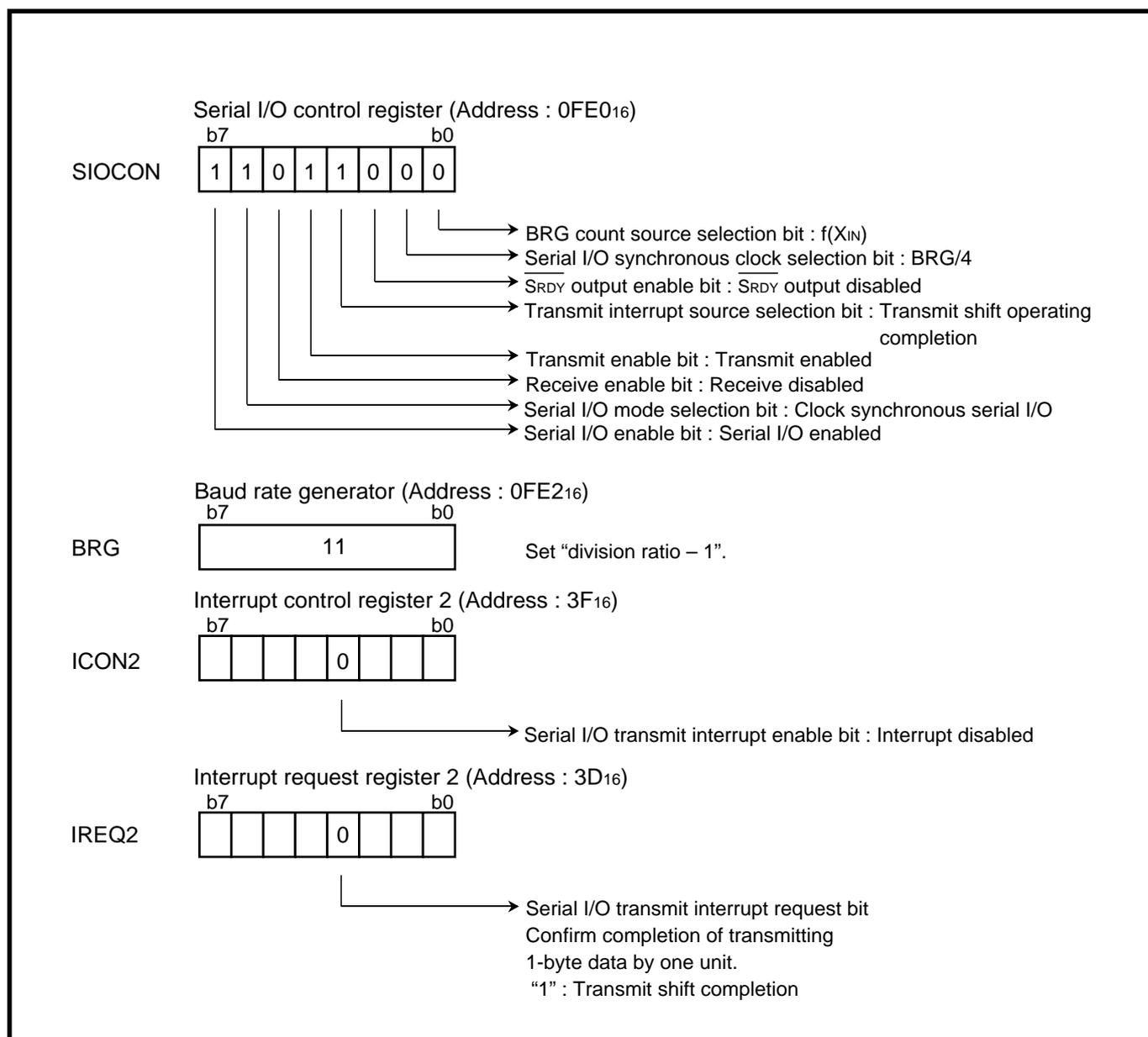


Fig. 2.4.21 Registers setting related to Serial I/O

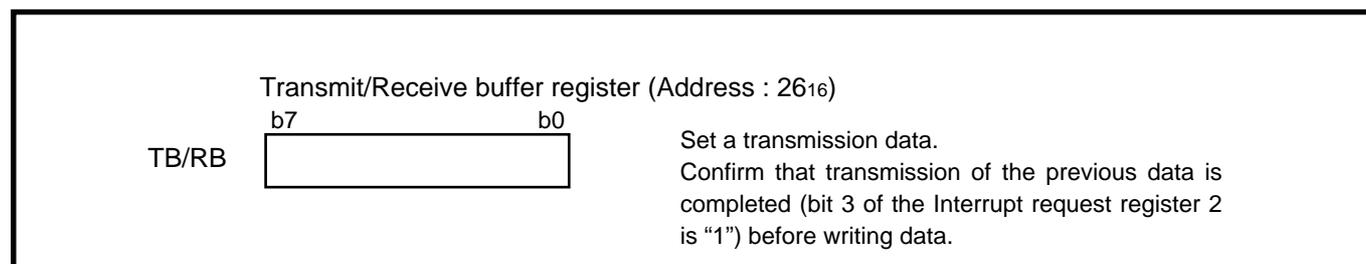


Fig. 2.4.22 Setting of serial I/O transmission data

When the registers are set as shown in Fig. 2.4.21, the Serial I/O can transmit 1-byte data by writing data to the transmit buffer register.

Thus, after setting the CS signal to "L", write the transmission data to the transmit buffer register by each 1 byte, and return the CS signal to "H" when the target number of bytes has been transmitted. Figure 2.4.23 shows a control procedure of Serial I/O.

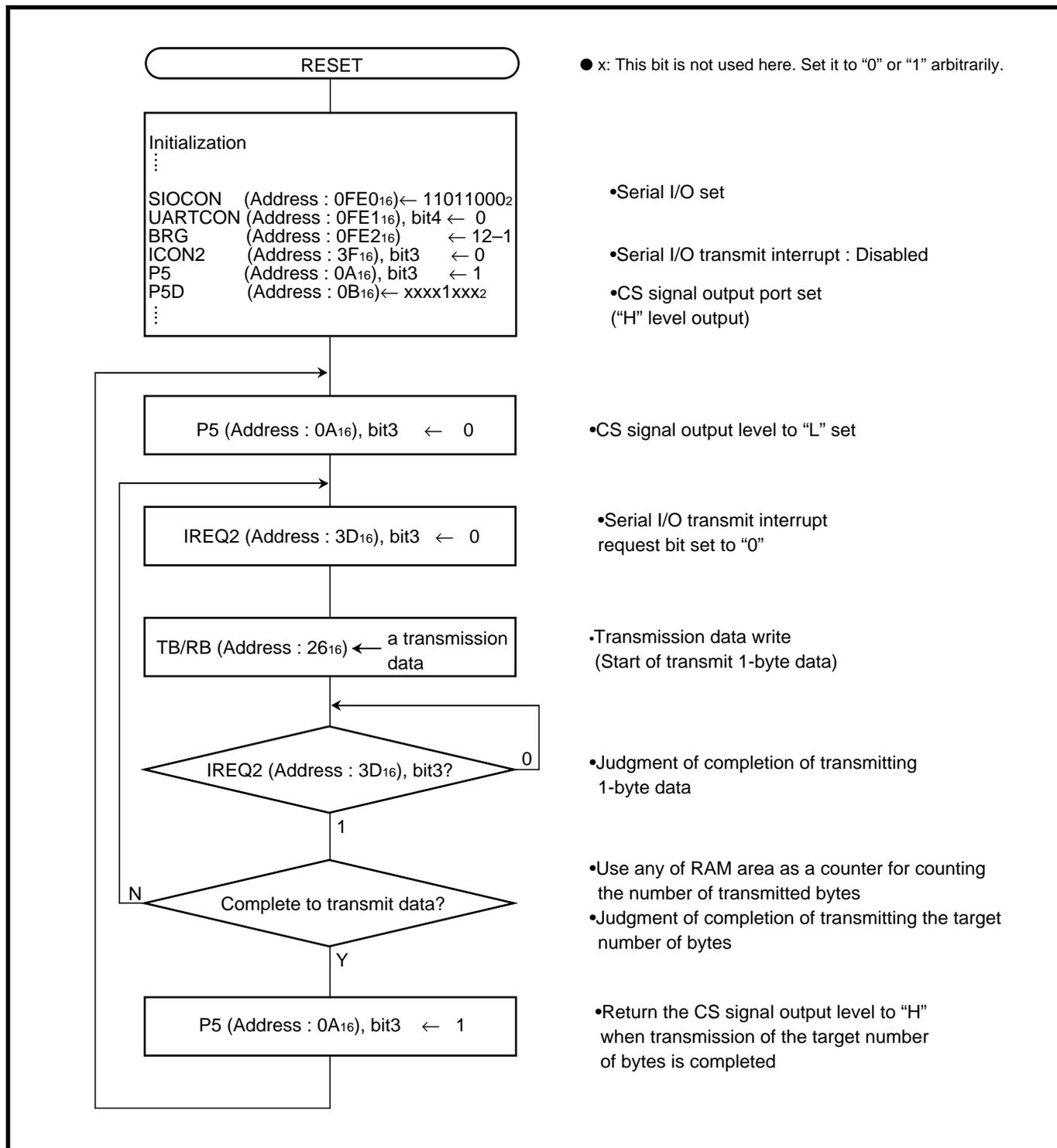


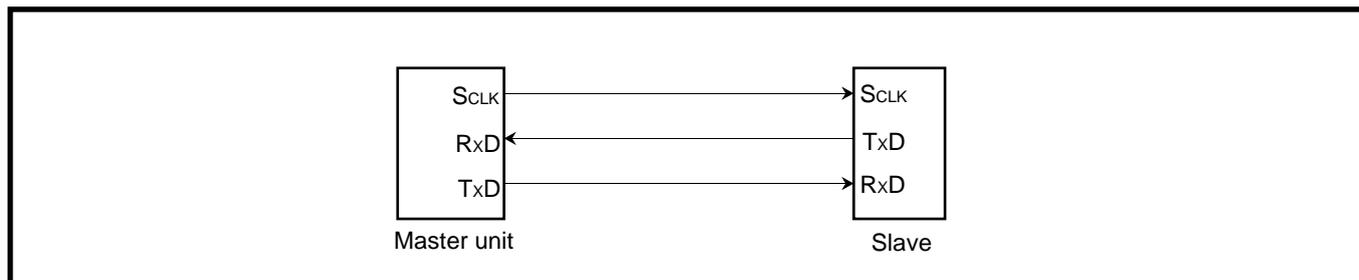
Fig. 2.4.23 Control procedure of Serial I/O

### (3) Cyclic transmission or reception of block data (data of specified number of bytes) between two microcomputers

**Outline :** When the clock synchronous serial I/O is used for communication, synchronization of the clock and the data between the transmitting and receiving sides may be lost because of noise included in the synchronous clock. It is necessary to correct that constantly, using "heading adjustment".

This "heading adjustment" is carried out by using the interval between blocks in this example.

Figure 2.4.24 shows a connection diagram.



**Fig. 2.4.24 Connection diagram**

#### Specifications :

- The serial I/O is used (clock synchronous serial I/O is selected).
- Synchronous clock frequency : 125 kHz ( $f(X_{IN}) = 6 \text{ MHz}$  is divided by 48)
- Byte cycle: 488  $\mu\text{s}$
- Number of bytes for transmission or reception : 8 byte/block
- Block transfer cycle : 16 ms
- Block transfer term : 3.5 ms
- Interval between blocks : 12.5 ms
- Heading adjustment time : 8 ms

#### Limitations of specifications :

- Reading of the reception data and setting of the next transmission data must be completed within the time obtained from "byte cycle – time for transferring 1-byte data" (in this example, the time taken from generating of the serial I/O receive interrupt request to input of the next synchronous clock is 431  $\mu\text{s}$ ).
- "Heading adjustment time < interval between blocks" must be satisfied.



**Control procedure :**

## ● Control in the master unit

After setting the related registers shown in Figure 2.4.26, the master unit starts transmission or reception of 1-byte data by writing transmission data to the transmit buffer register.

To perform the communication in the timing shown in Figure 2.4.25, take the timing into account and write transmission data. Additionally, read out the reception data when the serial I/O transmit interrupt request bit is set to "1," or before the next transmission data is written to the transmit buffer register.

Figure 2.4.27 shows a control procedure of the master unit using timer interrupts.

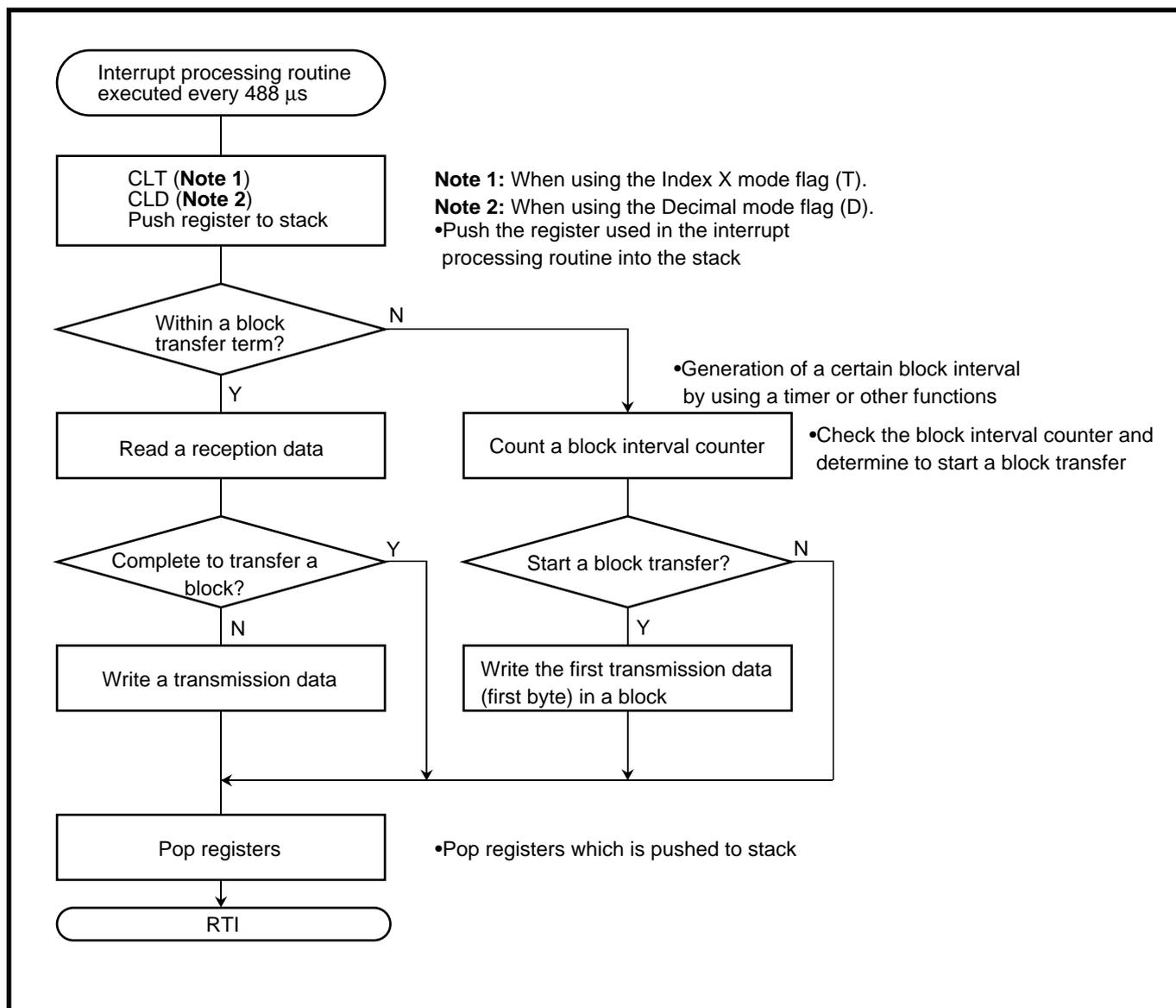


Fig. 2.4.27 Control procedure of master unit

● Control in the slave unit

After setting the related registers as shown in Figure 2.4.26, the slave unit becomes the state where a synchronous clock can be received at any time, and the serial I/O receive interrupt request bit is set to "1" each time an 8-bit synchronous clock is received.

In the serial I/O receive interrupt processing routine, the data to be transmitted next is written to the transmit buffer register after the received data is read out.

However, if no serial I/O receive interrupt occurs for a certain time (heading adjustment time or more), the following processing will be performed.

1. The first 1-byte data of the transmission data in the block is written into the transmit buffer register.
  2. The data to be received next is processed as the first 1 byte of the received data in the block.
- Figure 2.4.28 shows a control procedure of the slave unit using the serial I/O receive interrupt and any timer interrupt (for heading adjustment).

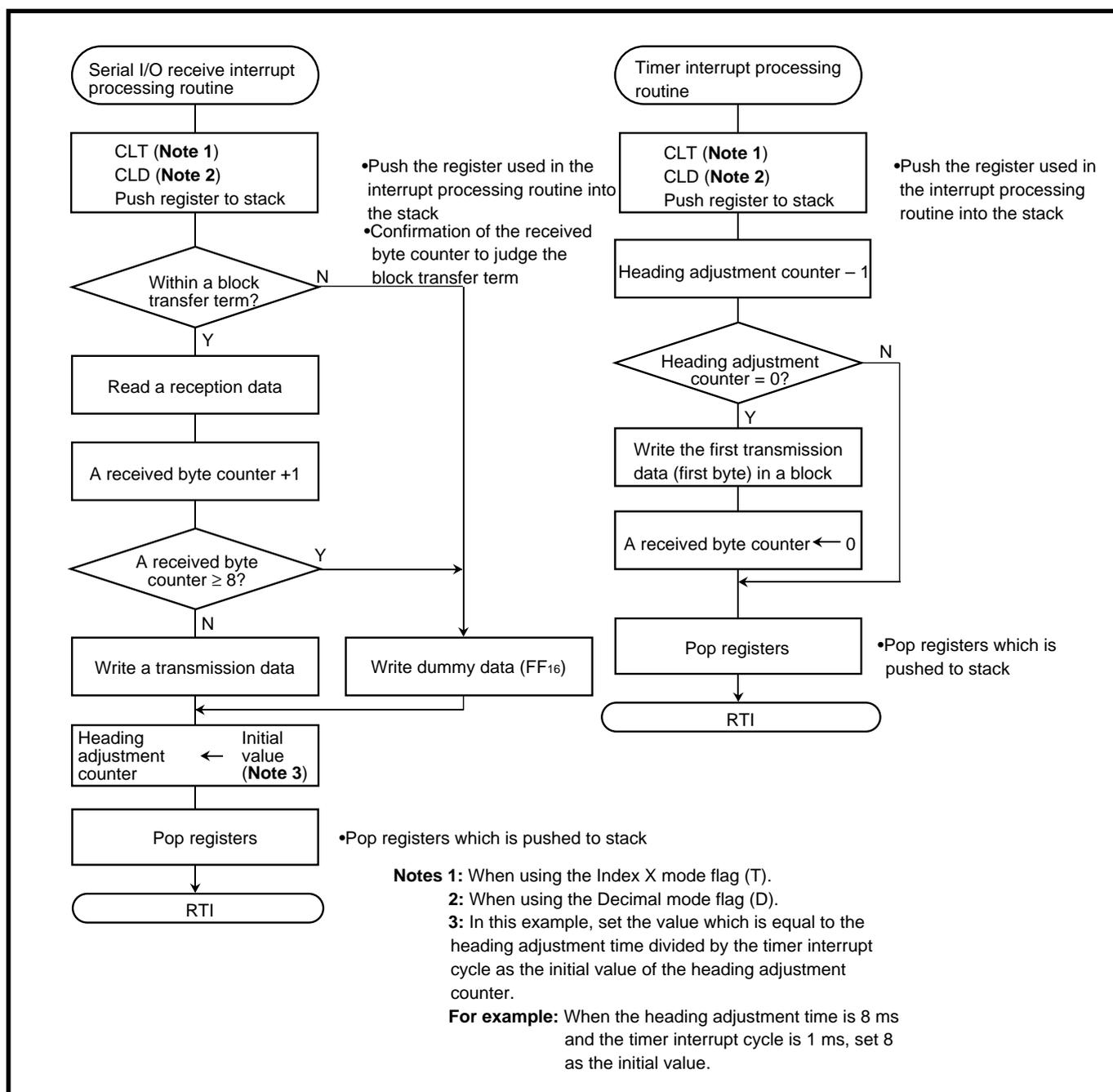
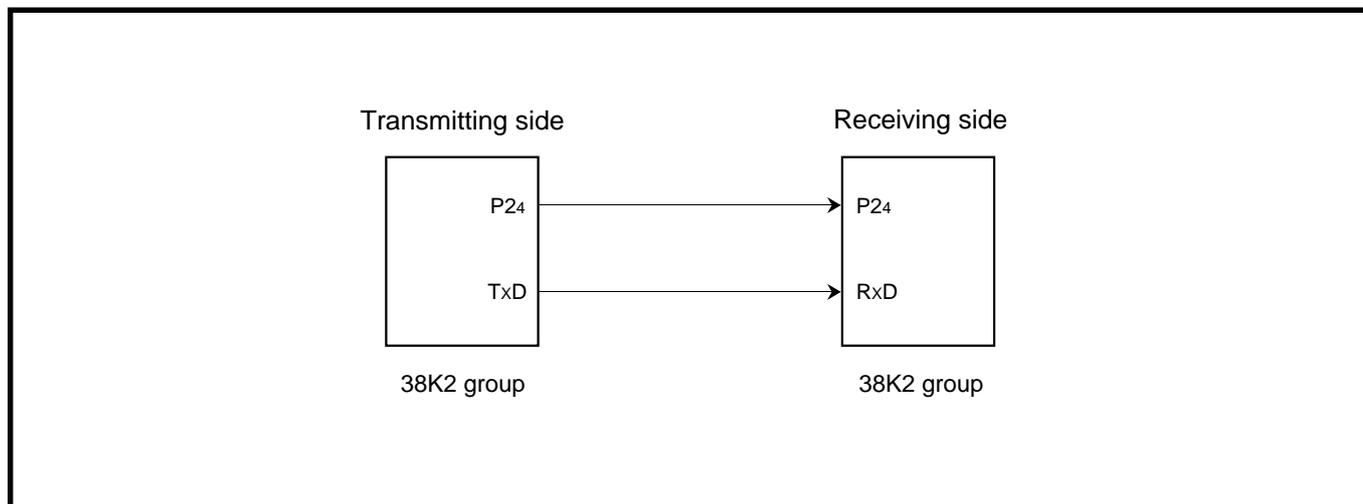


Fig. 2.4.28 Control procedure of slave unit

**(4) Communication (transmit/receive) using asynchronous serial I/O (UART)**

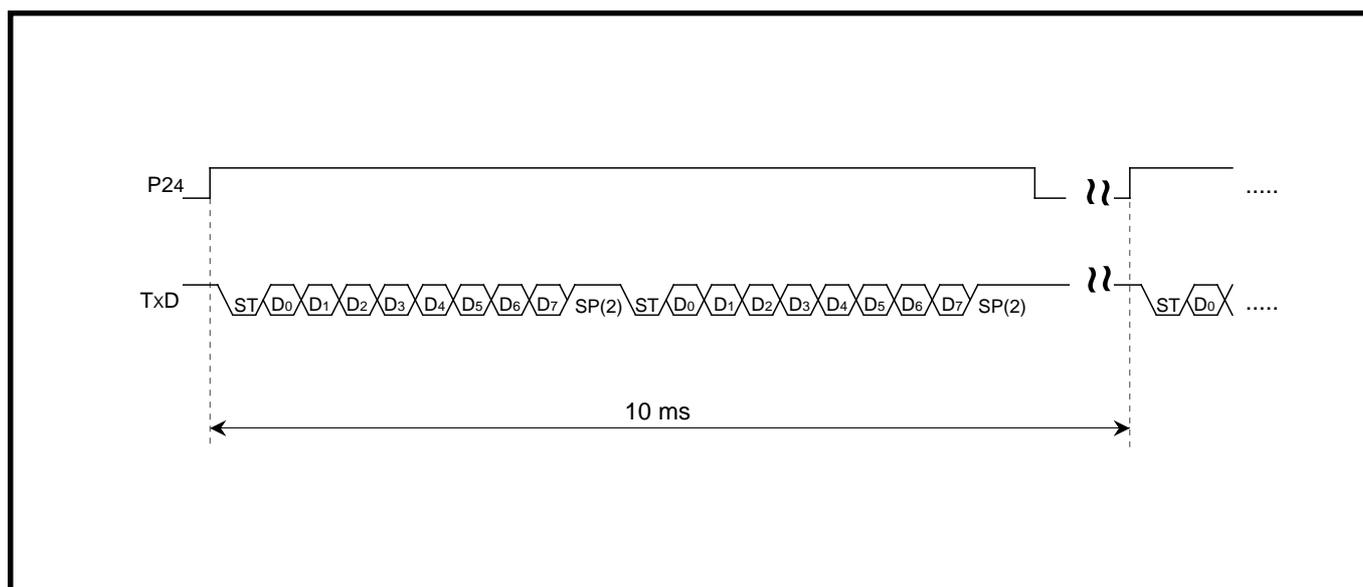
**Outline :** 2-byte data is transmitted and received, using the asynchronous serial I/O.  
Port P2<sub>4</sub> is used for communication control.

Figure 2.4.29 shows a connection diagram, and Figure 2.4.30 shows a timing chart.



**Fig. 2.4.29 Connection diagram (Communication using UART)**

- Specifications :**
- The Serial I/O is used (UART is selected).
  - Transfer bit rate : 9600 bps ( $f(X_{IN}) = 6 \text{ MHz}$  is divided by 624)
  - Communication control using port P2<sub>4</sub>  
(The output level of port P2<sub>4</sub> is controlled by software.)
  - 2-byte data is transferred from the transmitting side to the receiving side at intervals of 10 ms generated by the timer.



**Fig. 2.4.30 Timing chart (using UART)**

Table 2.4.1 shows setting examples of the baud rate generator (BRG) values and transfer bit rate values; Figure 2.4.31 shows registers setting related to the transmitting side; Figure 2.4.32 shows registers setting related to the receiving side.

**Table 2.4.1 Setting examples of Baud rate generator values and transfer bit rate values**

Transfer bit rate (bps) (Note 3)	BRG count source (Note 1)	At f(X <sub>IN</sub> ) = 6 MHz BRG setting value (Note 2)	At f(X <sub>IN</sub> ) = 8 MHz BRG setting value (Note 2)
600	f(X <sub>IN</sub> )/4	155	207
1200	f(X <sub>IN</sub> )/4	77	103
2400	f(X <sub>IN</sub> )	155	207
4800	f(X <sub>IN</sub> )	77	103
9600	f(X <sub>IN</sub> )	38	51
14400	f(X <sub>IN</sub> )	25	34
19200	f(X <sub>IN</sub> )	19	25
38400	f(X <sub>IN</sub> )	9	12
57600	f(X <sub>IN</sub> )	–	8

**Notes 1:** Select the BRG count source with bit 0 of the serial I/O control register (Address : 0FE016).

**2:** These are setting values with small errors.

**3:** Equation of transfer bit rate:

$$\text{Transfer bit rate (bps)} = \frac{f(X_{IN})}{(\text{BRG setting value} + 1) \times 16 \times m^*}$$

\*m: When bit 0 of the serial I/O control register (Address : 0FE016) is set to "0", a value of m is 1.

When bit 0 of the serial I/O control register (Address : 0FE016) is set to "1", a value of m is 4.

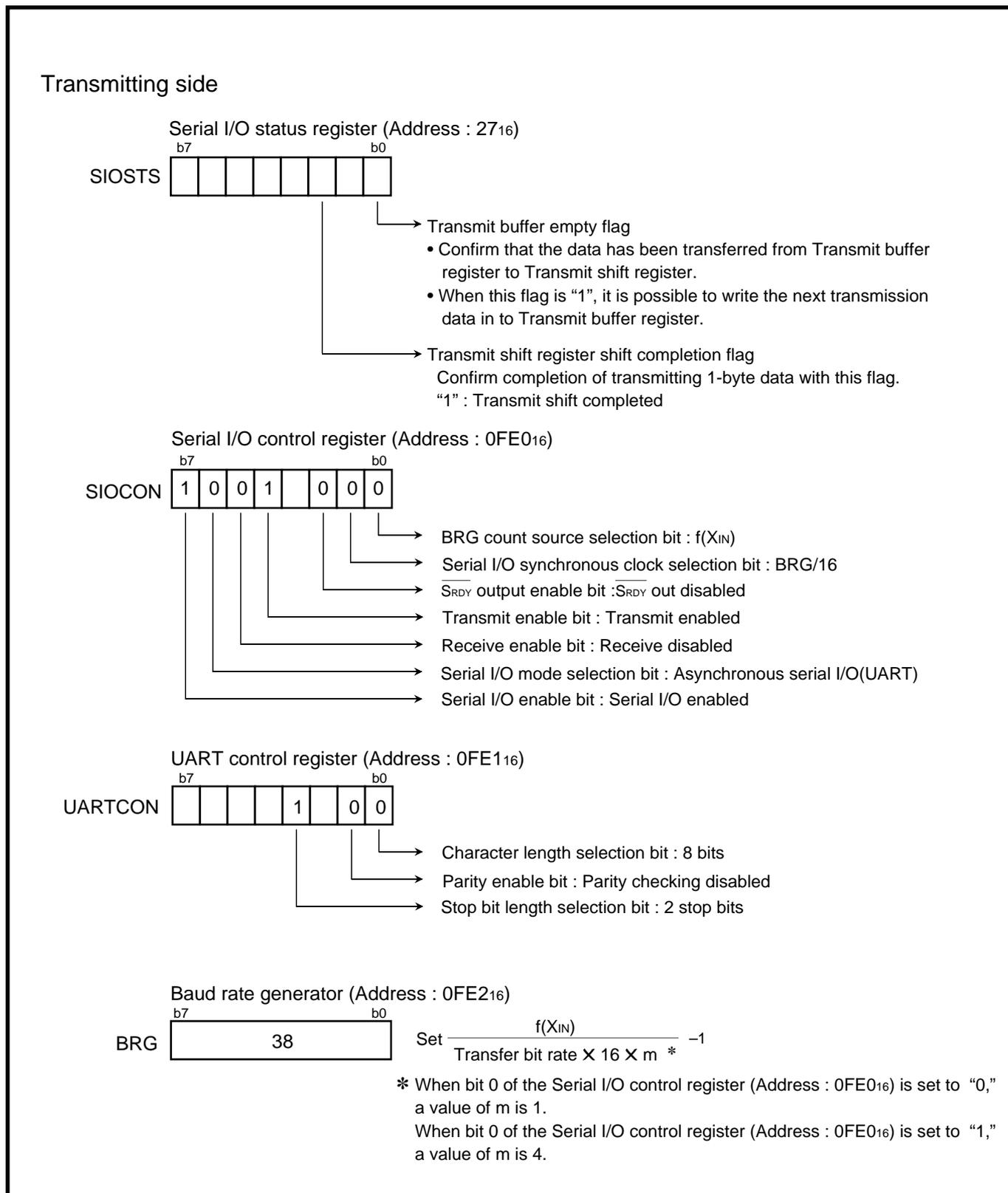


Fig. 2.4.31 Registers setting related to transmitting side

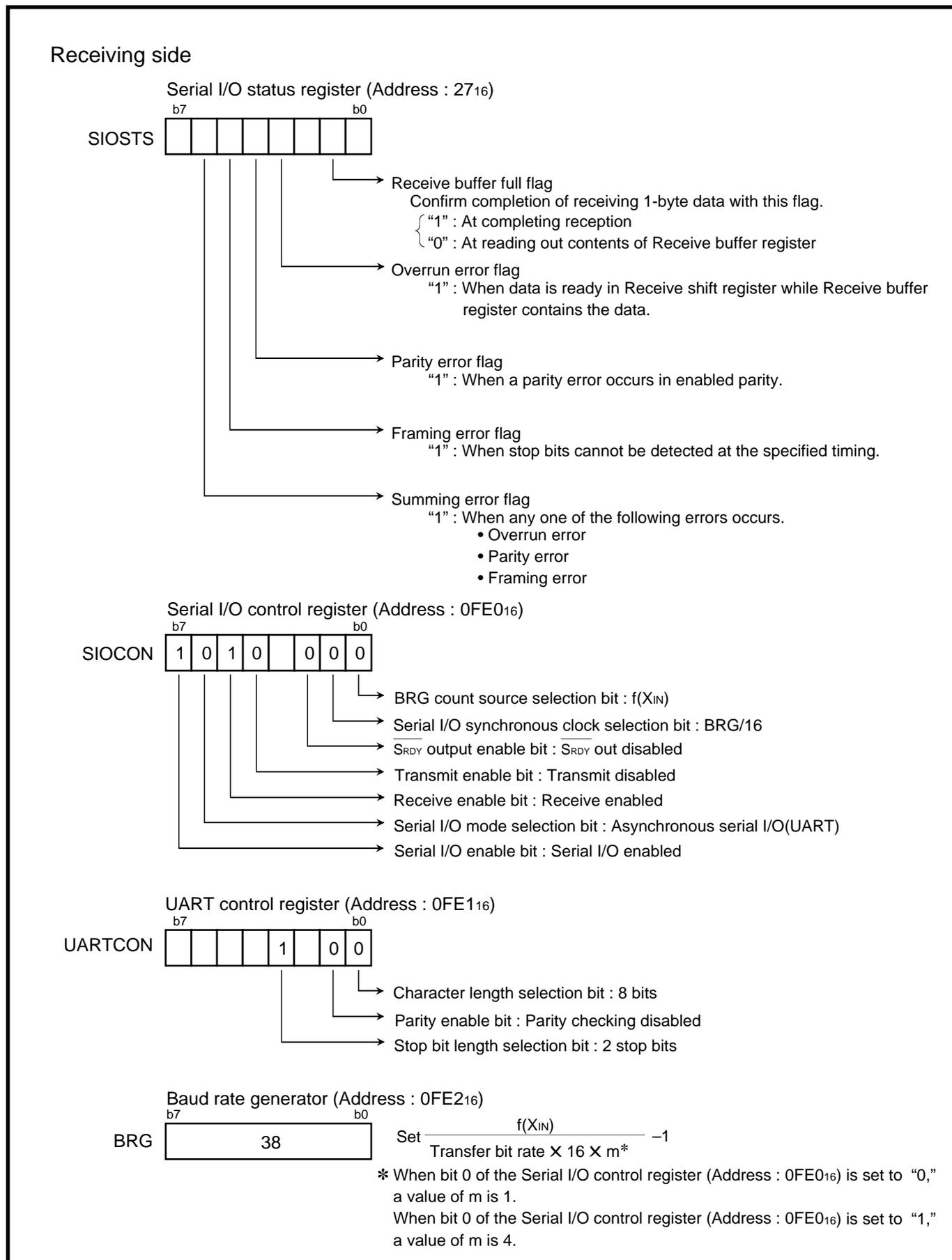


Fig. 2.4.32 Registers setting related to receiving side

Figure 2.4.33 shows a control procedure of the transmitting side, and Figure 2.4.34 shows a control procedure of the receiving side.

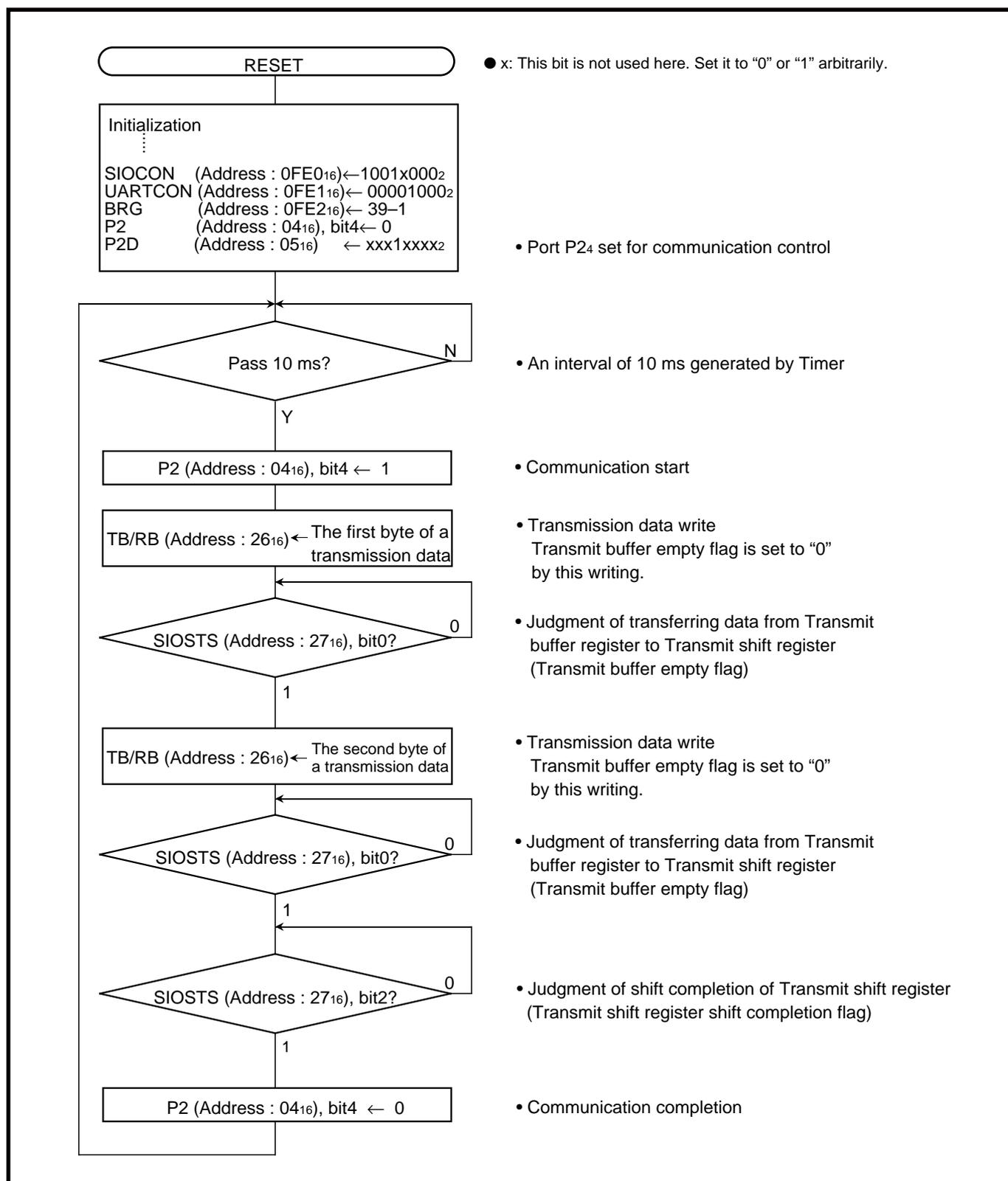


Fig. 2.4.33 Control procedure of transmitting side

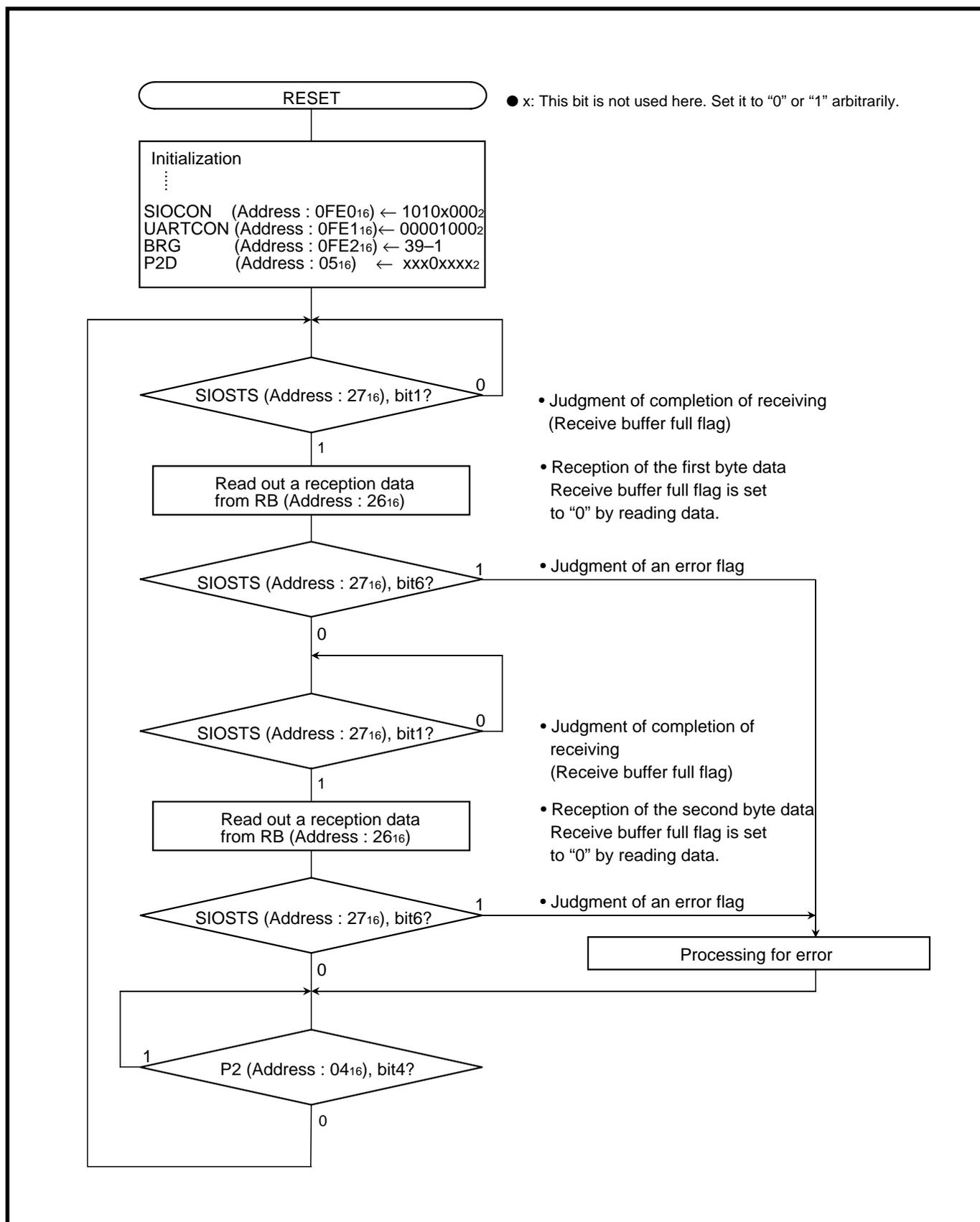


Fig. 2.4.34 Control procedure of receiving side

## 2.4.6 Notes on serial I/O

### (1) Notes when selecting clock synchronous serial I/O (Serial I/O)

#### ① Stop of transmission operation

Clear the serial I/O enable bit and the transmit enable bit to "0" (Serial I/O and transmit disabled).

#### ● Reason

Since transmission is not stopped and the transmission circuit is not initialized even if only the serial I/O enable bit is cleared to "0" (Serial I/O disabled), the internal transmission is running (in this case, since pins TxD, RxD, SCLK, and SRDY function as I/O ports, the transmission data is not output). When data is written to the transmit buffer register in this state, data starts to be shifted to the transmit shift register. When the serial I/O enable bit is set to "1" at this time, the data during internally shifting is output to the TxD pin and an operation failure occurs.

#### ② Stop of receive operation

Clear the receive enable bit to "0" (receive disabled), or clear the serial I/O enable bit to "0" (Serial I/O disabled).

#### ③ Stop of transmit/receive operation

Clear the transmit enable bit and receive enable bit to "0" simultaneously (transmit and receive disabled).

(when data is transmitted and received in the clock synchronous serial I/O mode, any one of data transmission and reception cannot be stopped.)

#### ● Reason

In the clock synchronous serial I/O mode, the same clock is used for transmission and reception. If any one of transmission and reception is disabled, a bit error occurs because transmission and reception cannot be synchronized.

In this mode, the clock circuit of the transmission circuit also operates for data reception. Accordingly, the transmission circuit does not stop by clearing only the transmit enable bit to "0" (transmit disabled). Also, the transmission circuit is not initialized by clearing the serial I/O enable bit to "0" (Serial I/O disabled) (refer to (1) ①).

**(2) Notes when selecting clock asynchronous serial I/O (Serial I/O)****① Stop of transmission operation**

Clear the transmit enable bit to "0" (transmit disabled).

**● Reason**

Since transmission is not stopped and the transmission circuit is not initialized even if only the serial I/O enable bit is cleared to "0" (Serial I/O disabled), the internal transmission is running (in this case, since pins TxD, RxD, SCLK, and SRDY function as I/O ports, the transmission data is not output). When data is written to the transmit buffer register in this state, data starts to be shifted to the transmit shift register. When the serial I/O enable bit is set to "1" at this time, the data during internally shifting is output to the TxD pin and an operation failure occurs.

**② Stop of receive operation**

Clear the receive enable bit to "0" (receive disabled).

**③ Stop of transmit/receive operation****Only transmission operation is stopped.**

Clear the transmit enable bit to "0" (transmit disabled).

**● Reason**

Since transmission is not stopped and the transmission circuit is not initialized even if only the serial I/O enable bit is cleared to "0" (Serial I/O disabled), the internal transmission is running (in this case, since pins TxD, RxD, SCLK, and SRDY function as I/O ports, the transmission data is not output). When data is written to the transmit buffer register in this state, data starts to be shifted to the transmit shift register. When the serial I/O enable bit is set to "1" at this time, the data during internally shifting is output to the TxD pin and an operation failure occurs.

**Only receive operation is stopped.**

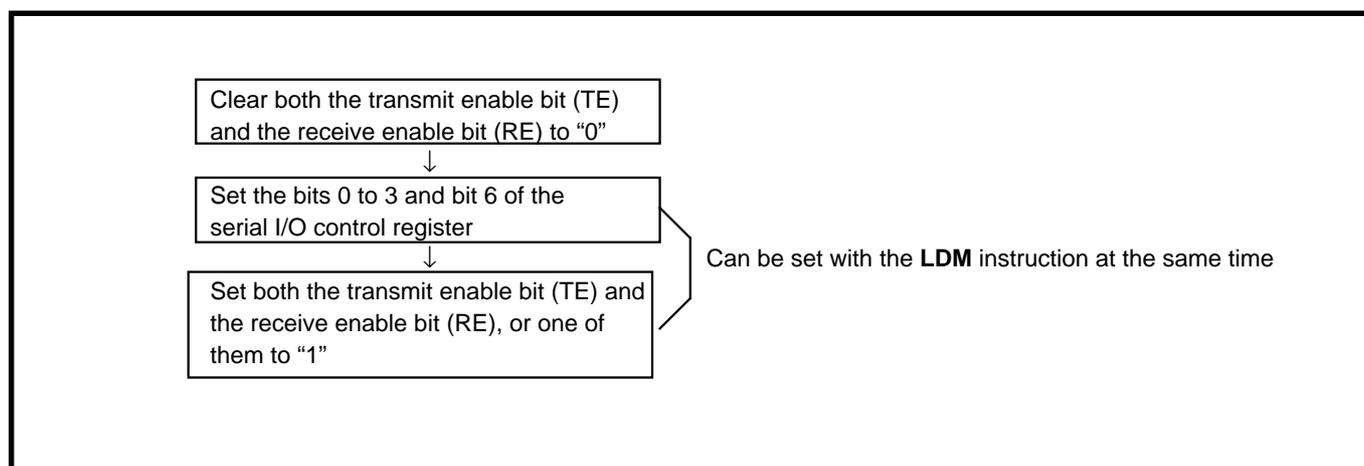
Clear the receive enable bit to "0" (receive disabled).

**(3)  $\overline{\text{SRDY}}$  output of reception side (Serial I/O)**

When signals are output from the  $\overline{\text{SRDY}}$  pin on the reception side by using an external clock in the clock synchronous serial I/O mode, set all of the receive enable bit, the  $\overline{\text{SRDY}}$  output enable bit, and the transmit enable bit to "1" (transmit enabled).

**(4) Setting serial I/O control register again (Serial I/O)**

Set the serial I/O control register again after the transmission and the reception circuits are reset by clearing both the transmit enable bit and the receive enable bit to "0."



**Fig. 2.4.35 Sequence of setting serial I/O control register again**

**(5) Data transmission control with referring to transmit shift register completion flag (Serial I/O)**

The transmit shift register completion flag changes from “1” to “0” with a delay of 0.5 to 1.5 shift clocks. When data transmission is controlled with referring to the flag after writing the data to the transmit buffer register, note the delay.

**(6) Transmission control when external clock is selected (Serial I/O)**

When an external clock is used as the synchronous clock for data transmission, set the transmit enable bit to “1” at “H” of the SCLK input level. Also, write the transmit data to the transmit buffer register (serial I/O shift register) at “H” of the SCLK input level.

**(7) Transmit interrupt request when transmit enable bit is set (Serial I/O)**

When the transmit interrupt is used, set the transmit interrupt enable bit to transmit enabled as shown in the following sequence.

- ① Set the interrupt enable bit to “0” (disabled) with CLB instruction.
- ② Prepare serial I/O for transmission/reception.
- ③ Set the interrupt request bit to “0” with CLB instruction after 1 or more instruction has been executed.
- ④ Set the interrupt enable bit to “1” (enabled).

**● Reason**

When the transmission enable bit is set to “1”, the transmit buffer empty flag and transmit shift register completion flag are set to “1”. The interrupt request is generated and the transmission interrupt bit is set regardless of which of the two timings listed below is selected as the timing for the transmission interrupt to be generated.

- Transmit buffer empty flag is set to “1”
- Transmit shift register completion flag is set to “1”

## 2.5 USB function

Some application notes are available on the Web site: "Renesas Technology Corp." Homepage  
USB Device

(<http://www.renesas.com/en/usb>)

Please refer to them for explanation and application of USB function.

## 2.6 HUB function

Some application notes are available on the Web site: "Renesas Technology Corp." Homepage  
USB Device

(<http://www.renesas.com/en/usb>)

Please refer to them for explanation and application of HUB function.

## 2.7 External bus interface(EXB)

Some application notes are available on the Web site: "Renesas Technology Corp." Homepage  
USB Device

(<http://www.renesas.com/en/usb>)

Please refer to them for explanation and application of external bus interface.

## 2.8 A/D converter

This paragraph explains the registers setting method and the notes related to the A/D converter.

### 2.8.1 Memory map

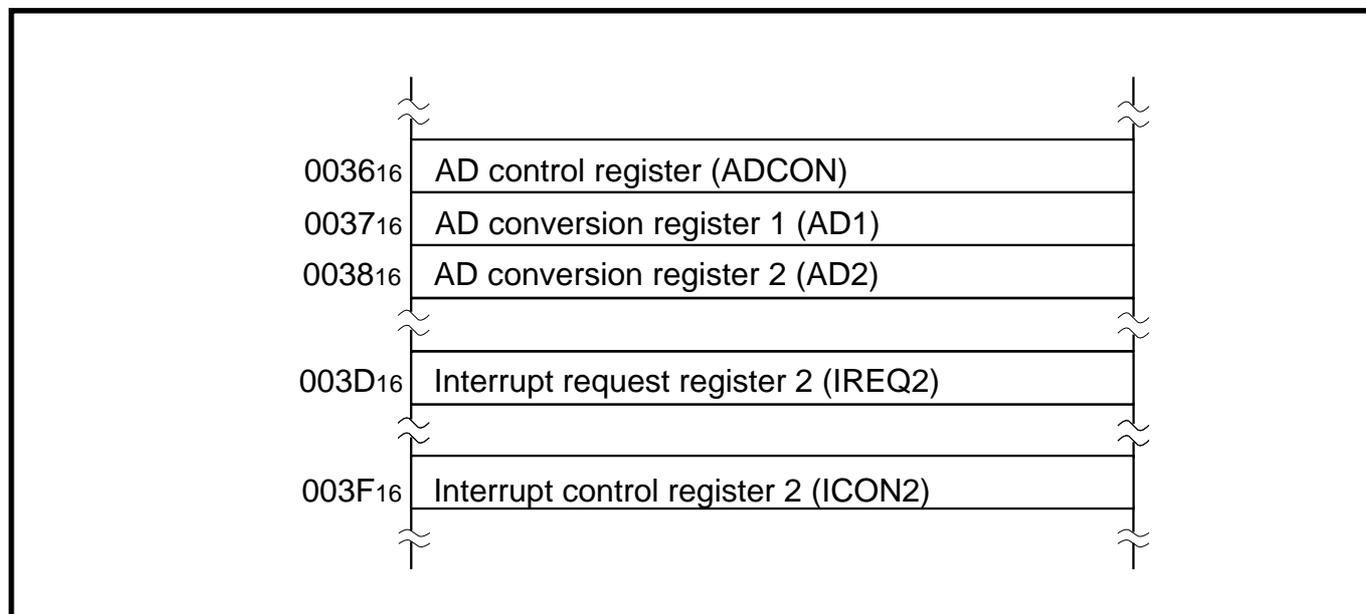


Fig. 2.8.1 Memory map of registers related to A/D converter

### 2.8.2 Related registers

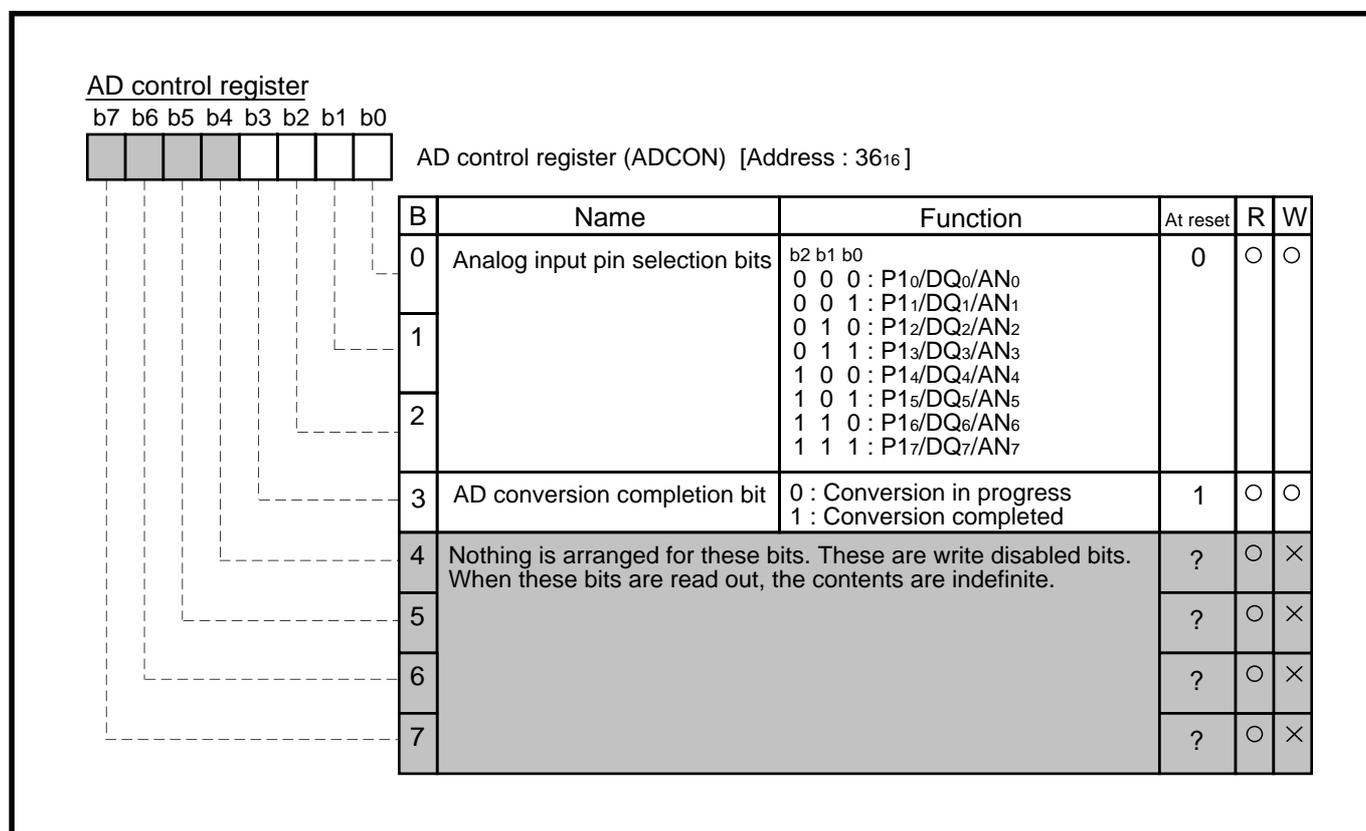


Fig. 2.8.2 Structure of AD control register

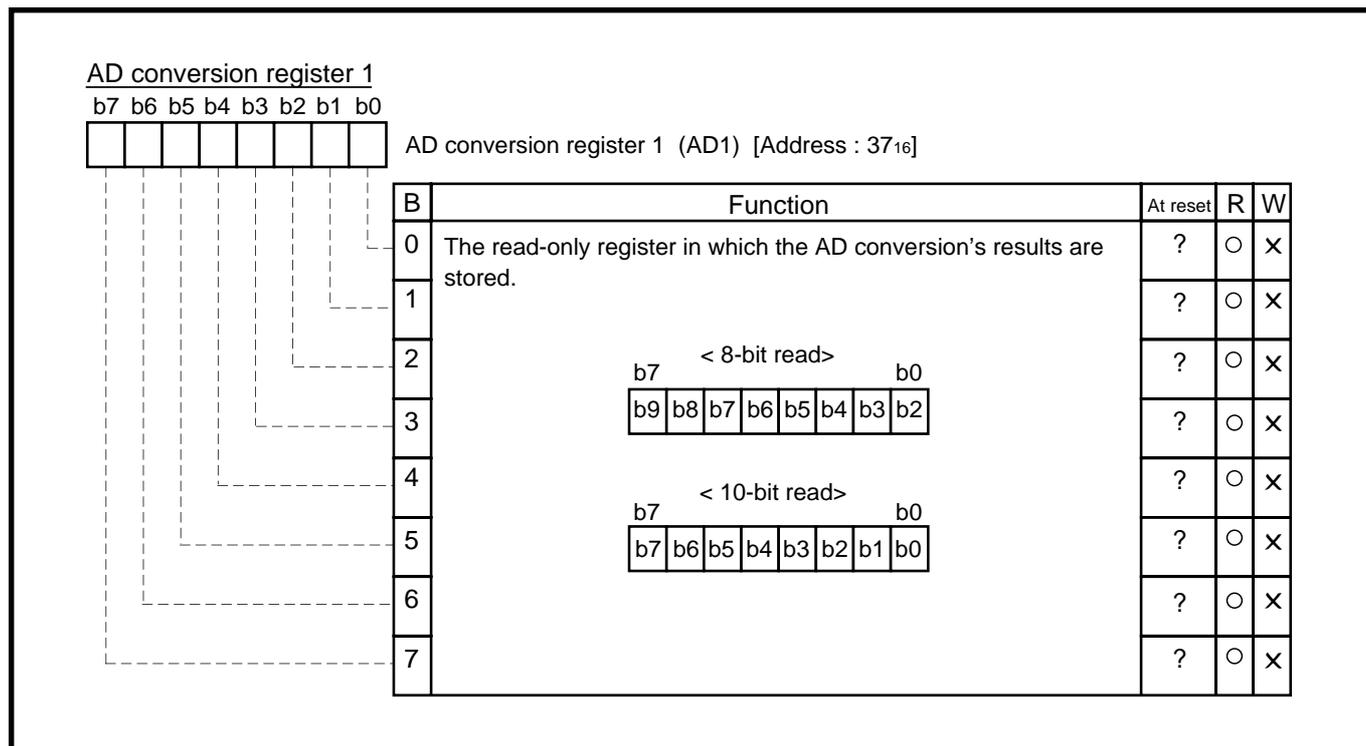


Fig. 2.8.3 Structure of AD conversion register 1

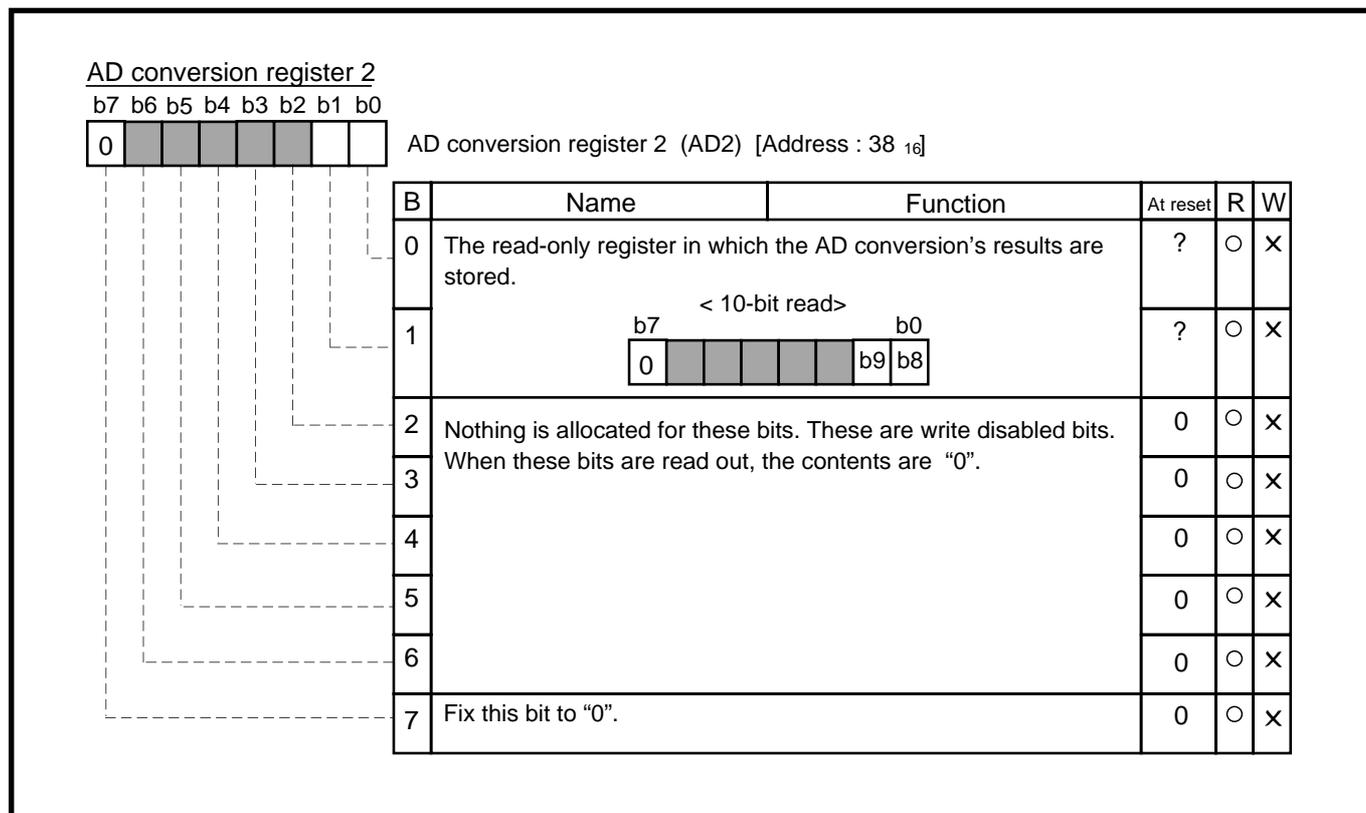


Fig. 2.8.4 Structure of AD conversion register 2

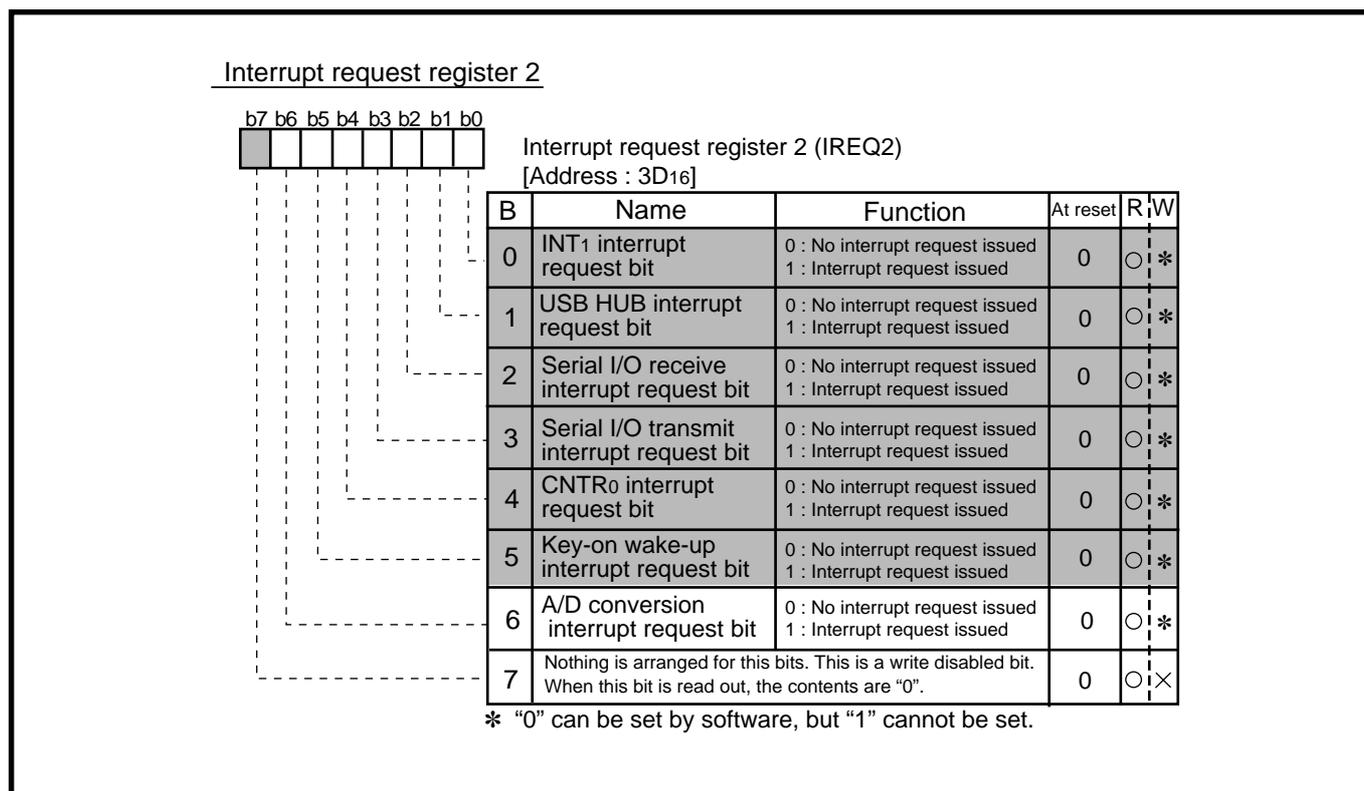


Fig. 2.8.5 Structure of Interrupt request register 2

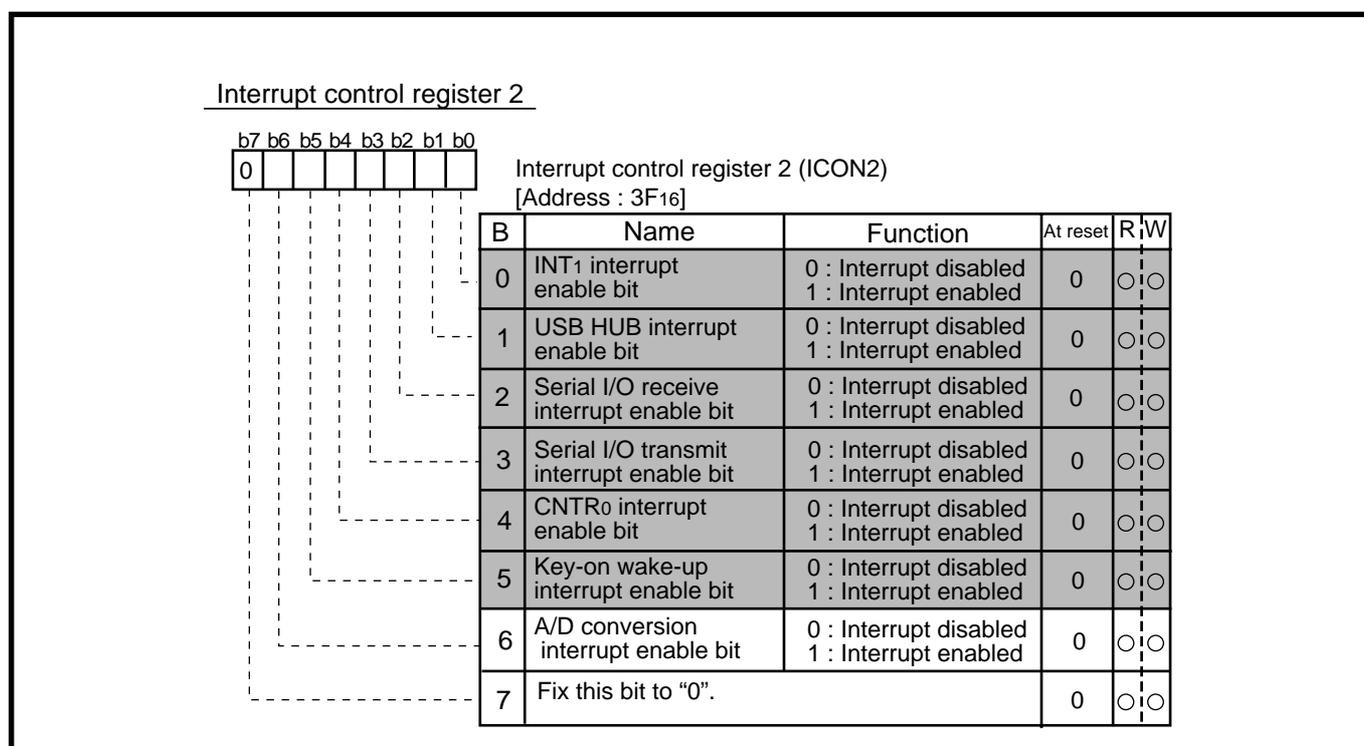


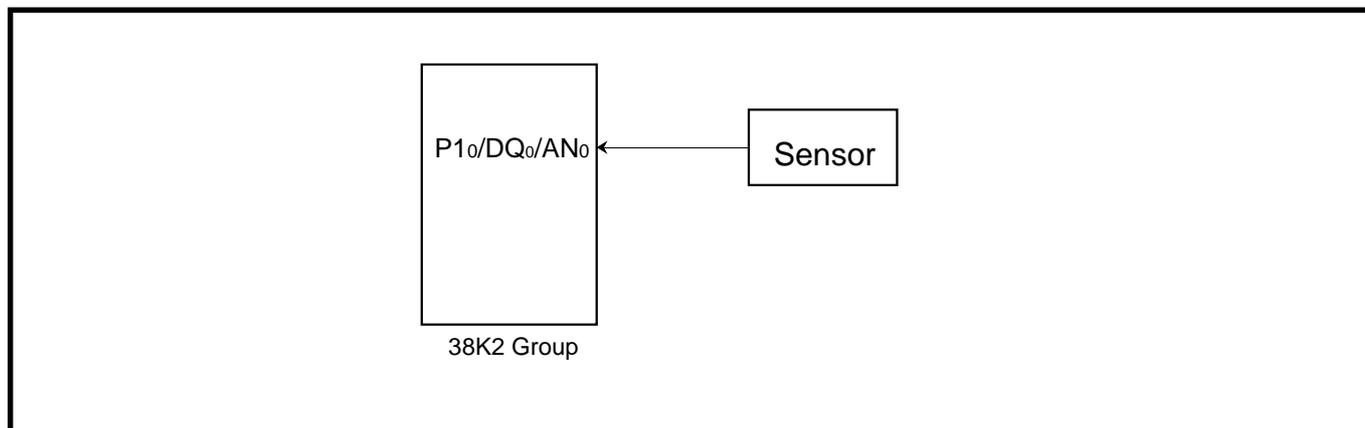
Fig. 2.8.6 Structure of Interrupt control register 2

### 2.8.3 A/D converter application examples

#### (1) Conversion of analog input voltage

**Outline :** The analog input voltage input from a sensor is converted to digital values.

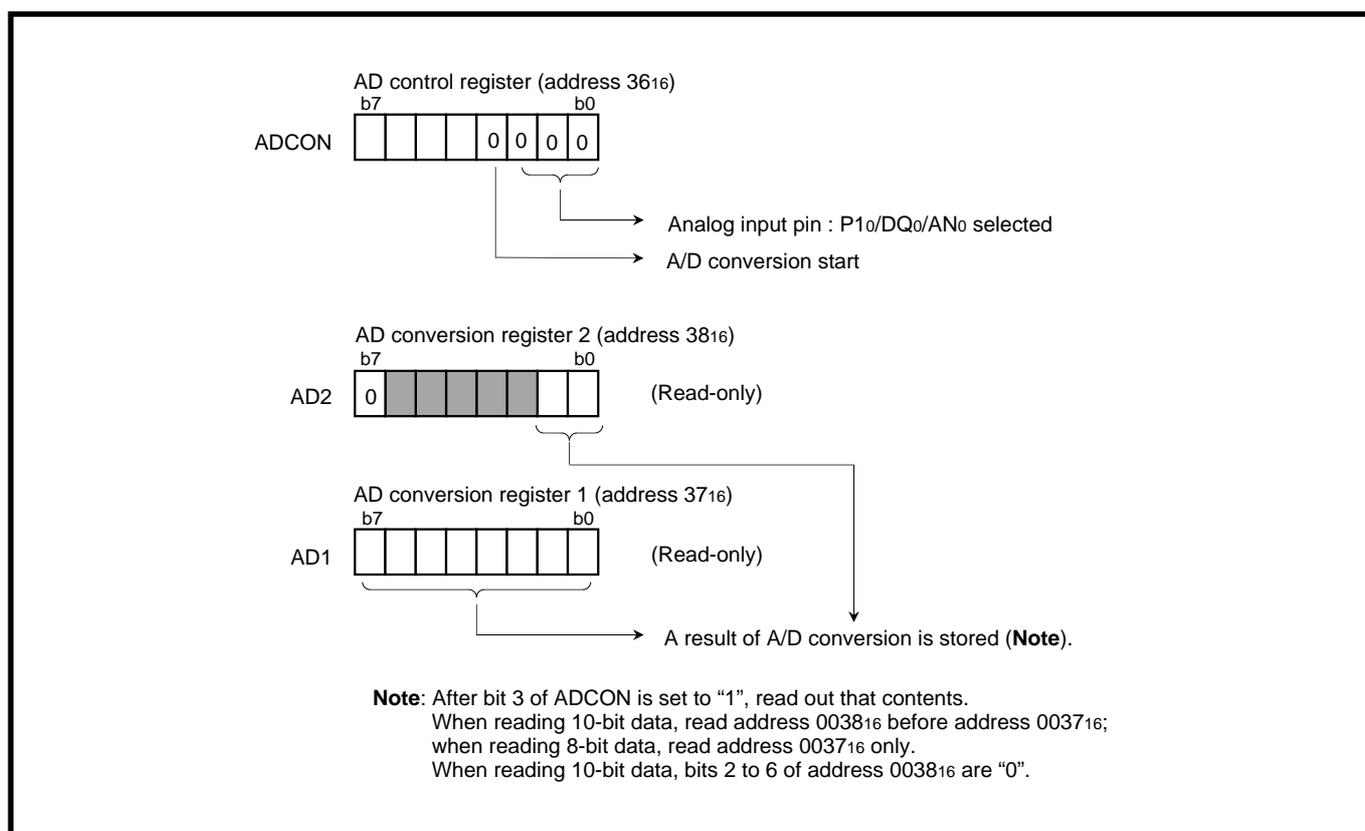
Figure 2.8.7 shows a connection diagram, and Figure 2.8.8 shows the related registers setting.



**Fig. 2.8.7 Connection diagram**

**Specifications :**

- The analog input voltage input from a sensor is converted to digital values.
- P10/DQ0/AN0 pin is used as an analog input pin.



**Fig. 2.8.8 Related registers setting**

An analog input signal from a sensor is converted to the digital value according to the related registers setting shown by Figure 2.8.8. Figure 2.8.9 shows the control procedure for 8-bit read, and Figure 2.8.10 shows the control procedure for 10-bit read.

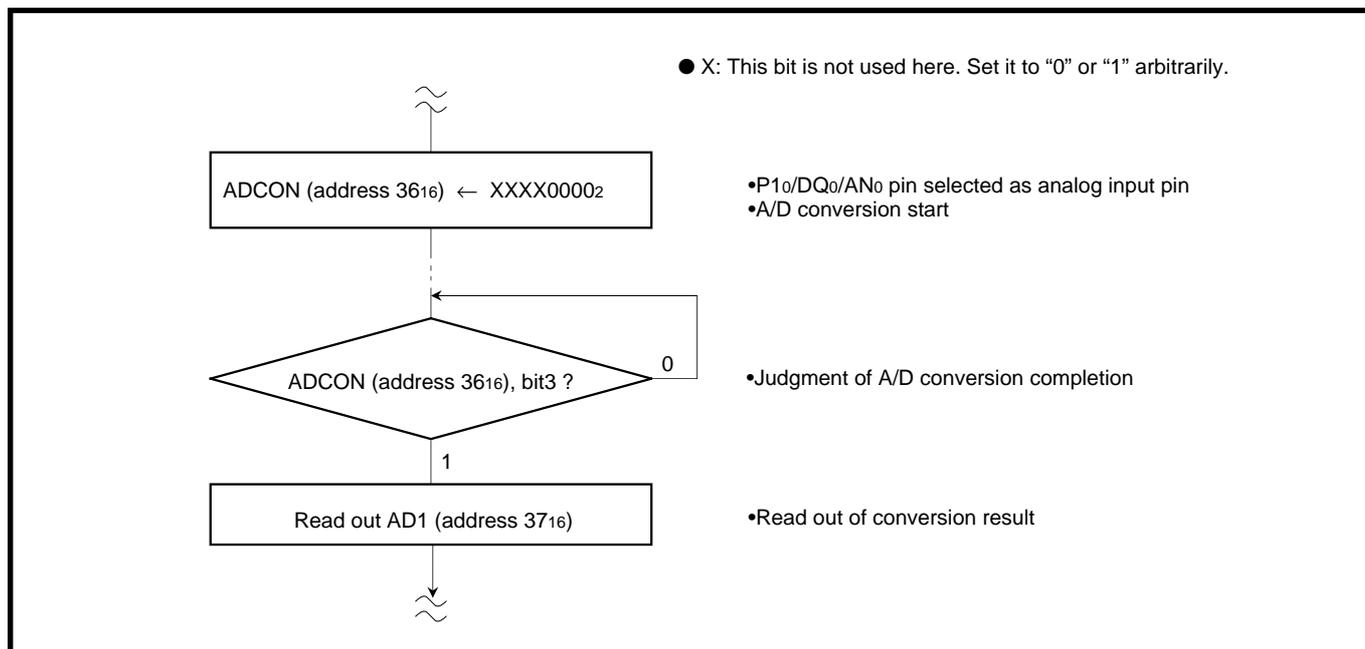


Fig. 2.8.9 Control procedure for 8-bit read

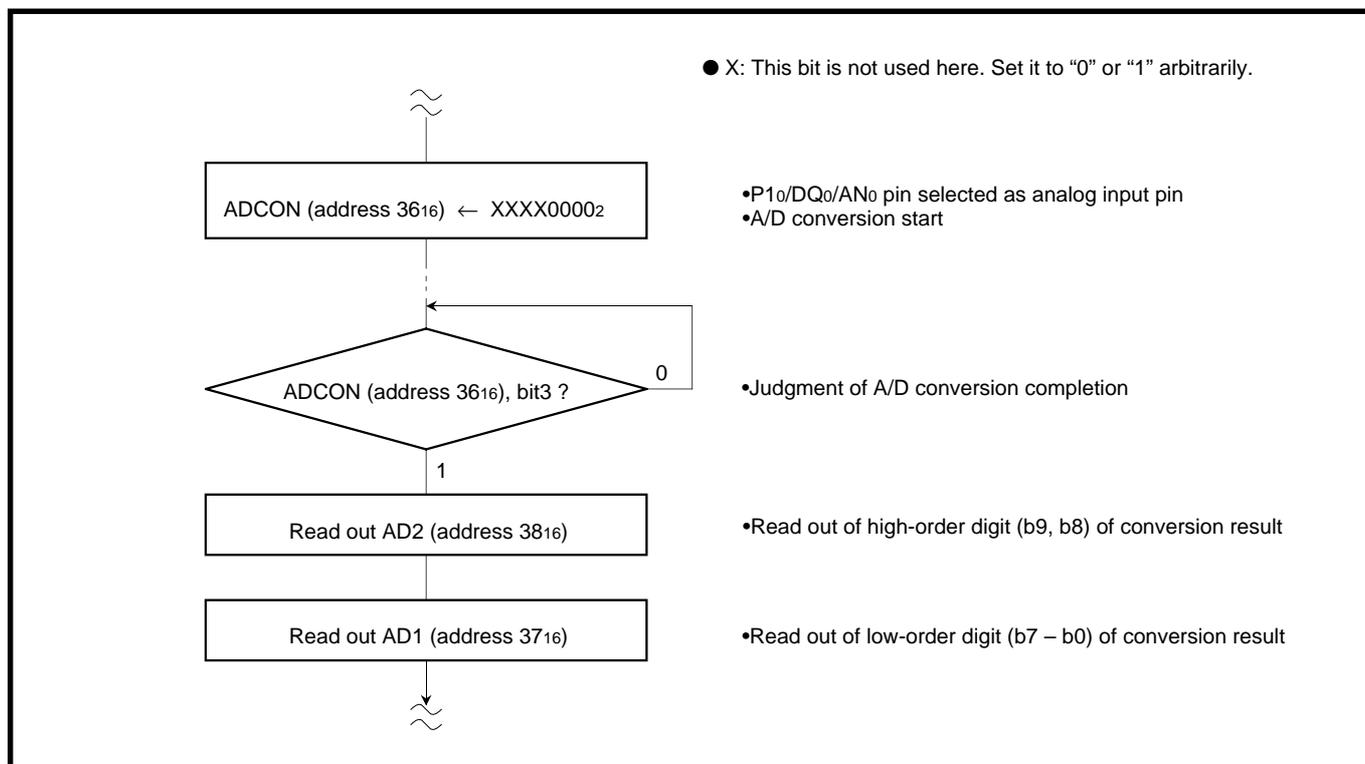


Fig. 2.8.10 Control procedure for 10-bit read

### 2.8.4 Notes on A/D converter

#### (1) Analog input pin

Make the signal source impedance for analog input low, or equip an analog input pin with an external capacitor of 0.01  $\mu$ F to 1  $\mu$ F. Further, be sure to verify the operation of application products on the user side.

##### ● Reason

An analog input pin includes the capacitor for analog voltage comparison. Accordingly, when signals from signal source with high impedance are input to an analog input pin, charge and discharge noise generates. This may cause the A/D conversion precision to be worse.

#### (2) Clock frequency during A/D conversion

The comparator consists of a capacity coupling, and a charge of the capacity will be lost if the clock frequency is too low. Thus, make sure the following during an A/D conversion.

- $f(X_{IN})$  is 500 kHz or more
- Do not execute the **STP** instruction

## 2.9 Watchdog timer

This paragraph explains the registers setting method and the notes related to the watchdog timer.

### 2.9.1 Memory map

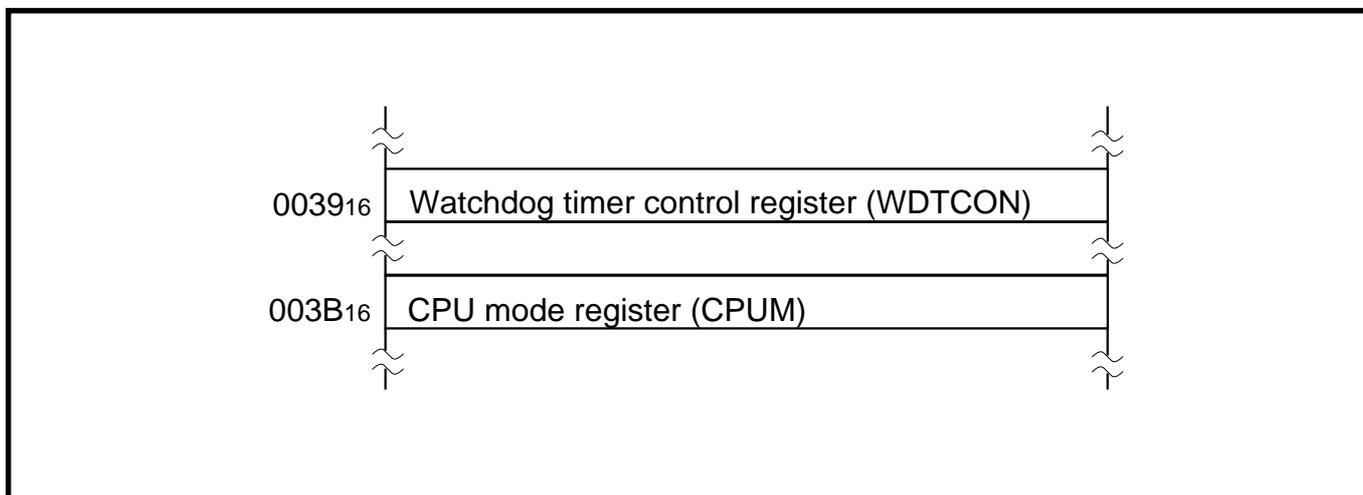


Fig. 2.9.1 Memory map of registers related to watchdog timer

### 2.9.2 Related registers

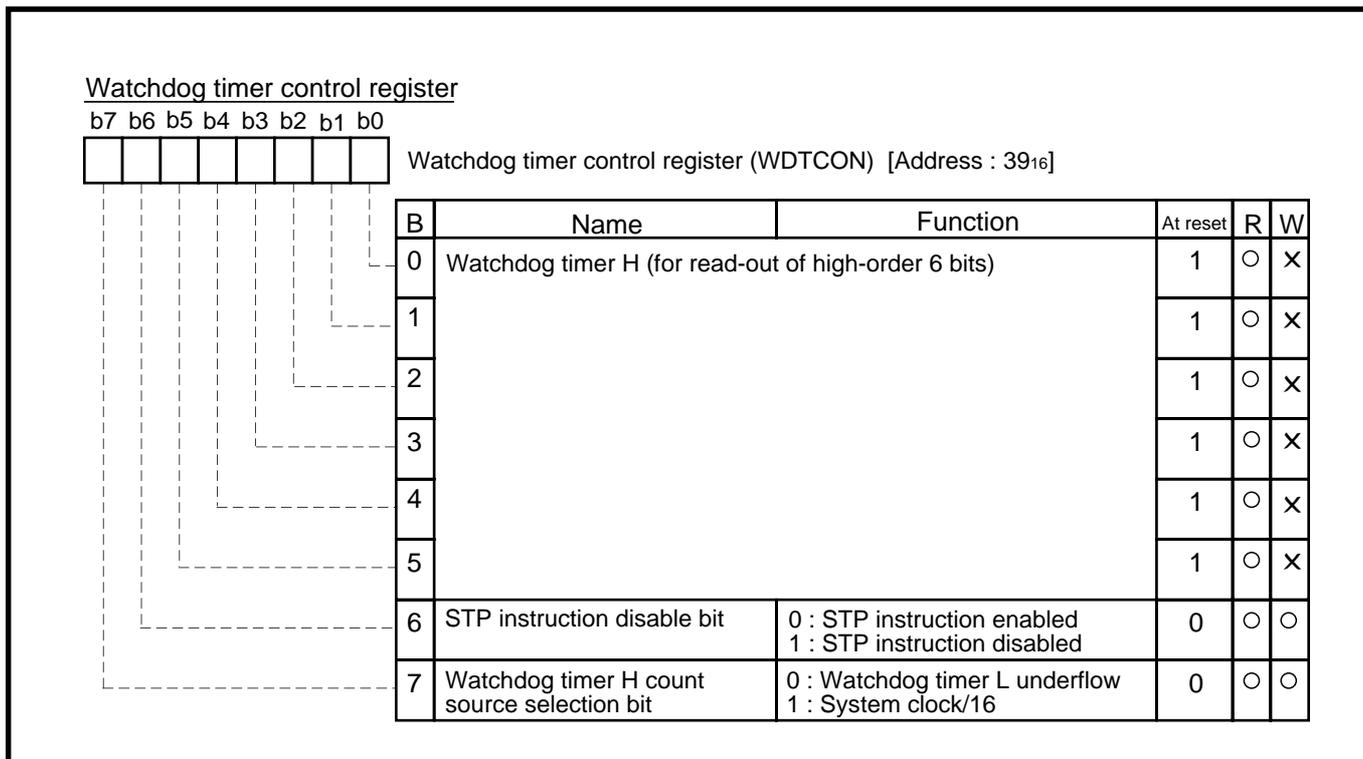


Fig. 2.9.2 Structure of Watchdog timer control register

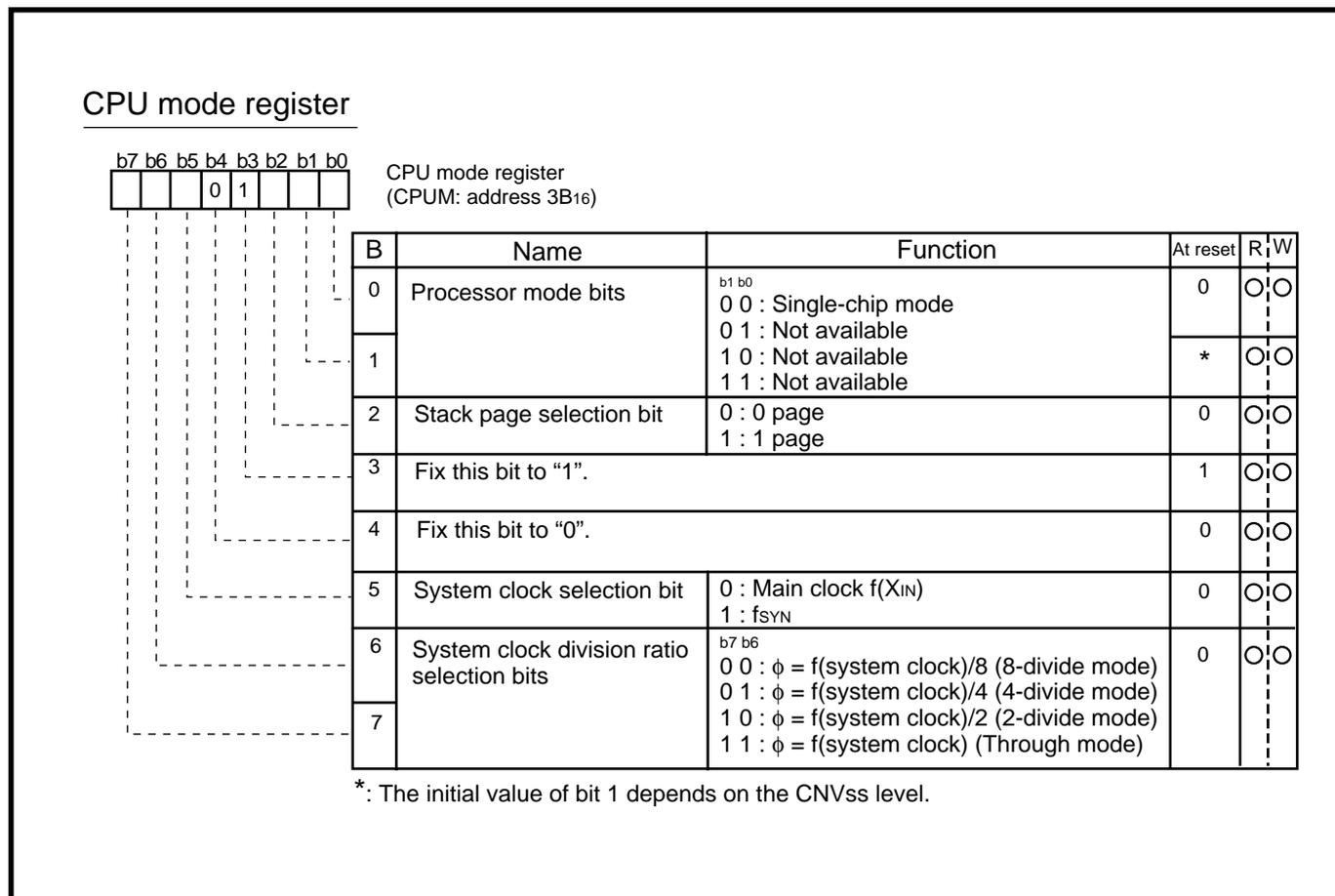


Fig. 2.9.3 Structure of CPU mode register

### 2.9.3 Watchdog timer application examples

#### (1) Detection of program runaway

**Outline:** If program runaway occurs, let the microcomputer reset, using the internal timer for detection of program runaway.

- Specifications:**
- An underflow of watchdog timer H is judged to be program runaway, and the microcomputer is returned to the reset status.
  - Before the watchdog timer H underflows, "0" is set into bit 7 of the watchdog timer control register at every cycle in a main routine.
  - Through mode is used as a system clock division ratio.
  - An underflow signal of the watchdog timer L is supplied as the count source of watchdog timer H.

Figure 2.9.4 shows a watchdog timer connection and division ratio setting; Figure 2.9.5 shows the related registers setting; Figure 2.9.6 shows the control procedure.

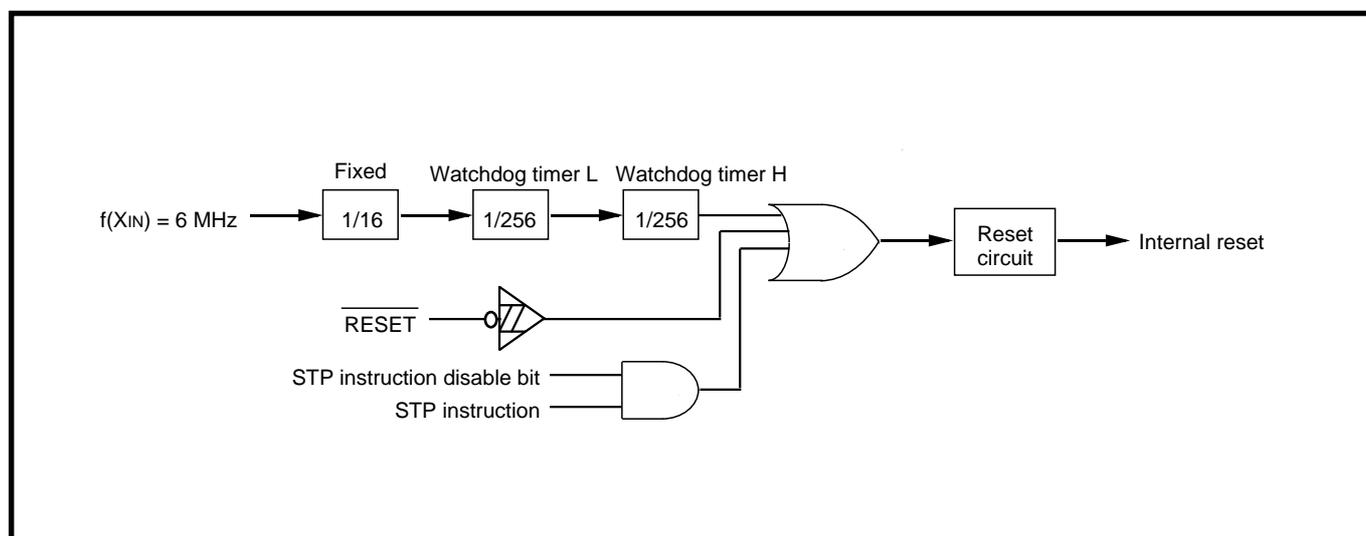


Fig. 2.9.4 Watchdog timer connection and division ratio setting

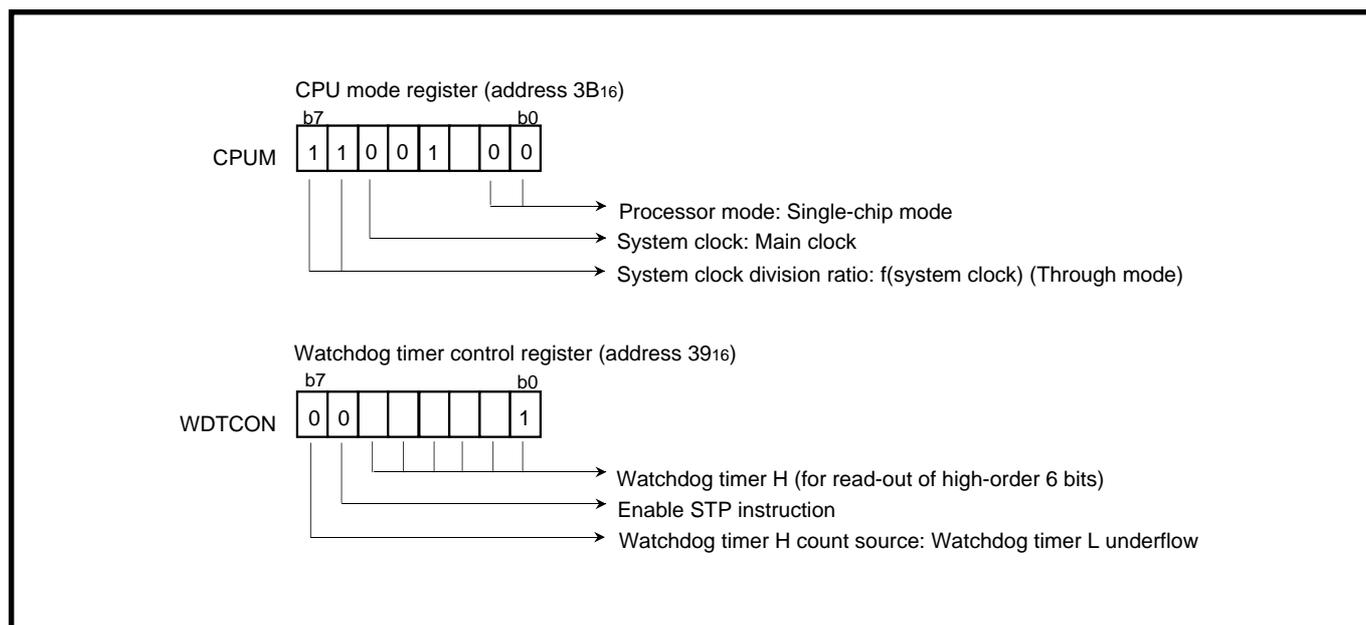


Fig. 2.9.5 Related registers setting

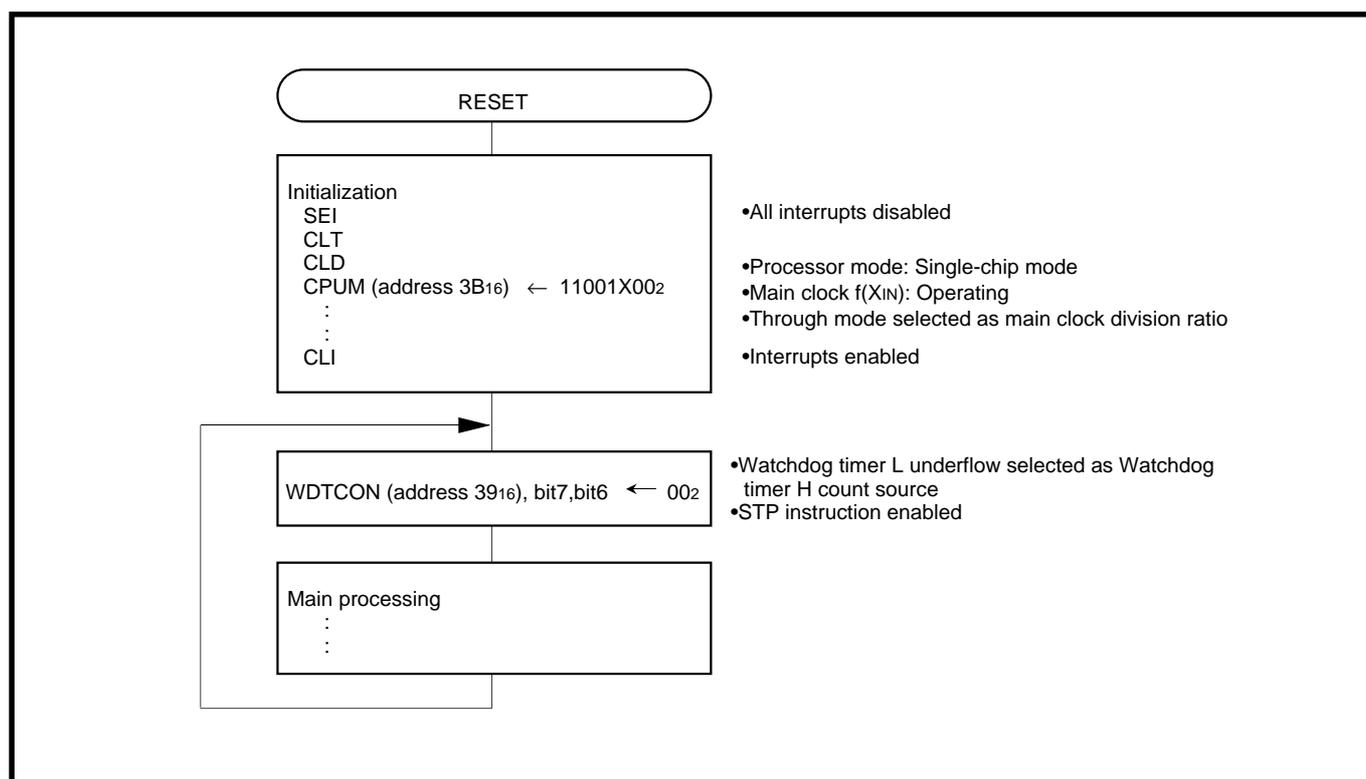


Fig. 2.9.6 Control procedure

### 2.9.4 Notes on watchdog timer

- Make sure that the watchdog timer does not underflow while waiting Stop release, because the watchdog timer keeps counting during that term.
- When the STP instruction disable bit has been set to “1”, it is impossible to switch it to “0” by a program.

## 2.10 Reset

### 2.10.1 Connection example of reset IC

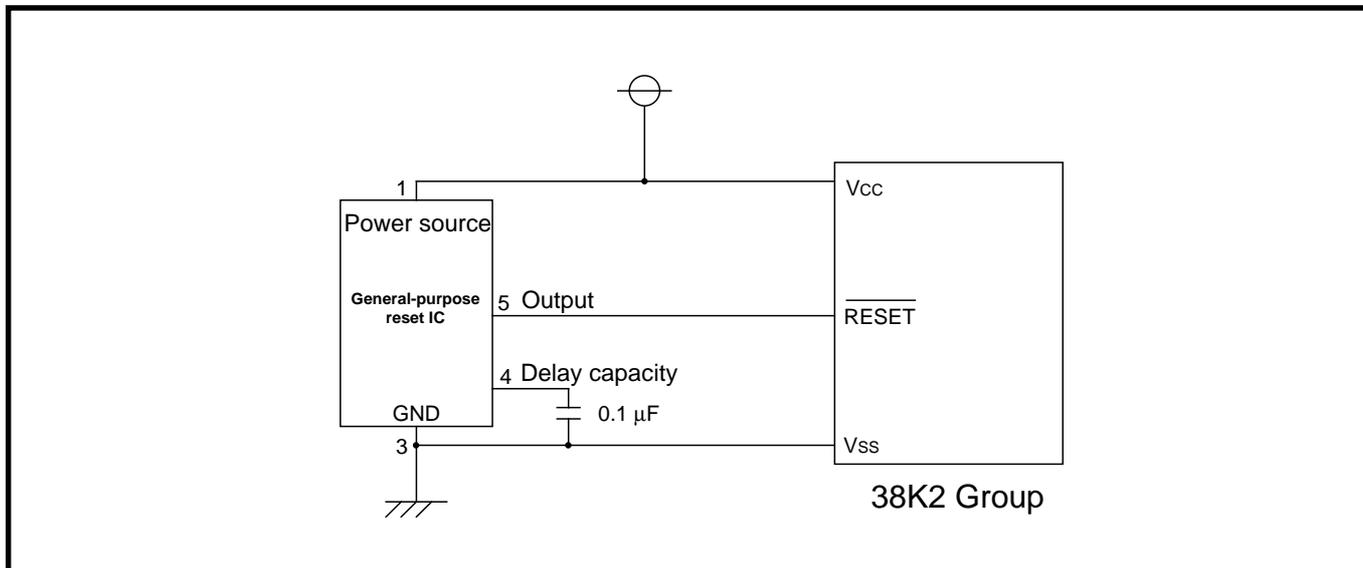


Fig. 2.10.1 Example of power on reset circuit

Figure 2.10.2 shows the system example which switches to the RAM backup mode by detecting a drop of the system power source voltage with the INT interrupt.

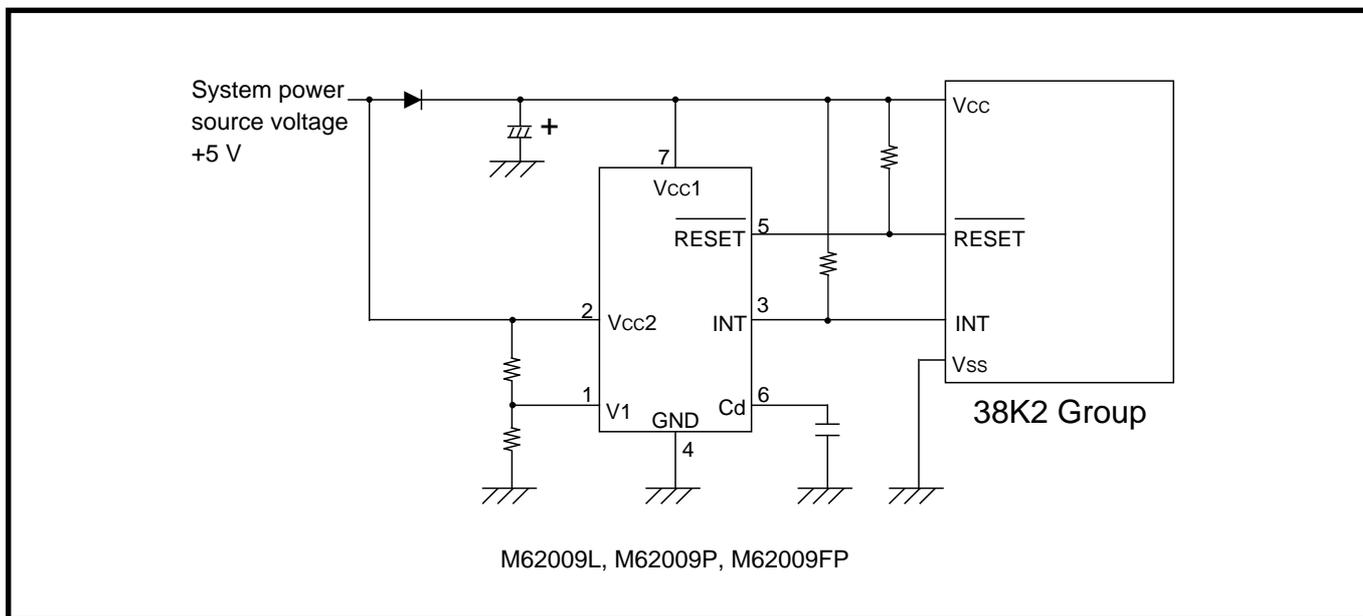


Fig. 2.10.2 RAM backup system

### 2.10.2 Notes on $\overline{\text{RESET}}$ pin

#### Connecting capacitor

In case where the  $\overline{\text{RESET}}$  signal rise time is long, connect a ceramic capacitor or others across the  $\overline{\text{RESET}}$  pin and the VSS pin. Use a 1000 pF or more capacitor for high frequency use. When connecting the capacitor, note the following :

- Make the length of the wiring which is connected to a capacitor as short as possible.
- Be sure to verify the operation of application products on the user side.

#### ● Reason

If the several nanosecond or several ten nanosecond impulse noise enters the  $\overline{\text{RESET}}$  pin, it may cause a microcomputer failure.

## 2.11 Frequency synthesizer (PLL)

This paragraph explains the registers setting method and the notes related to the frequency synthesizer (PLL circuit).

### 2.11.1 Memory map

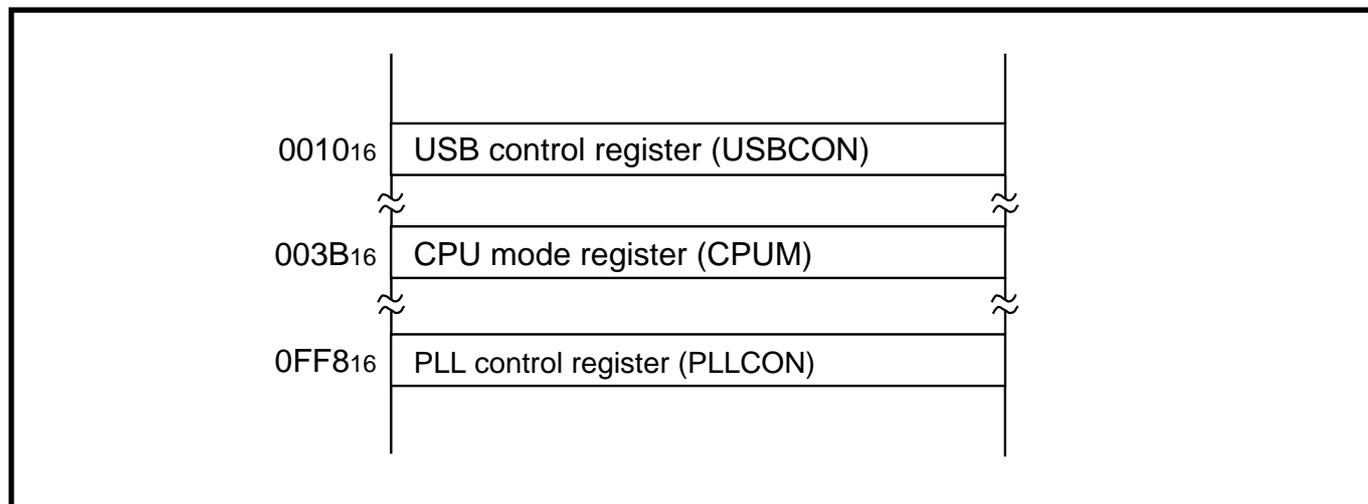


Fig. 2.11.1 Memory map of registers related to PLL

### 2.11.2 Related registers

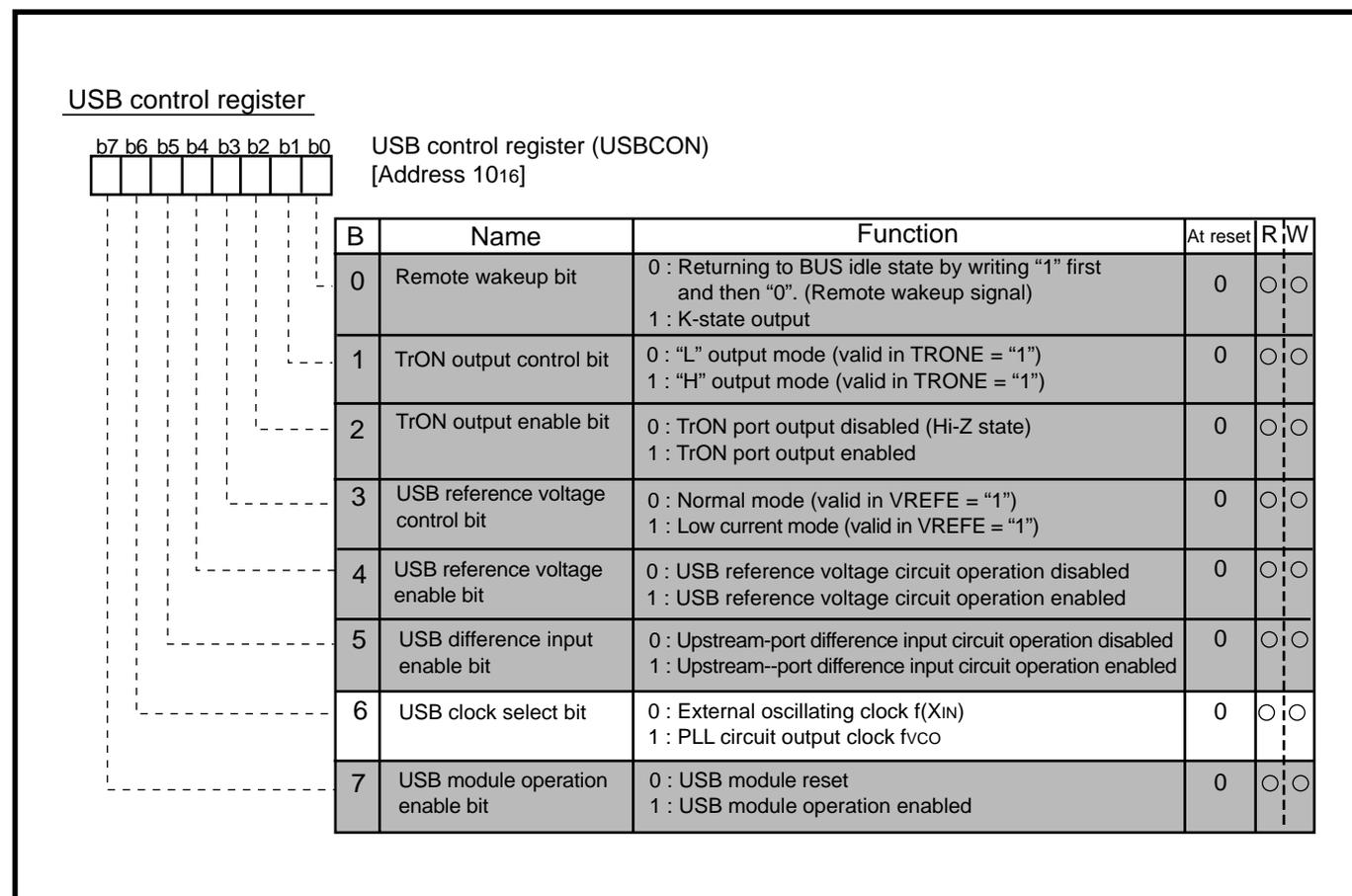


Fig. 2.11.2 Structure of USB control register

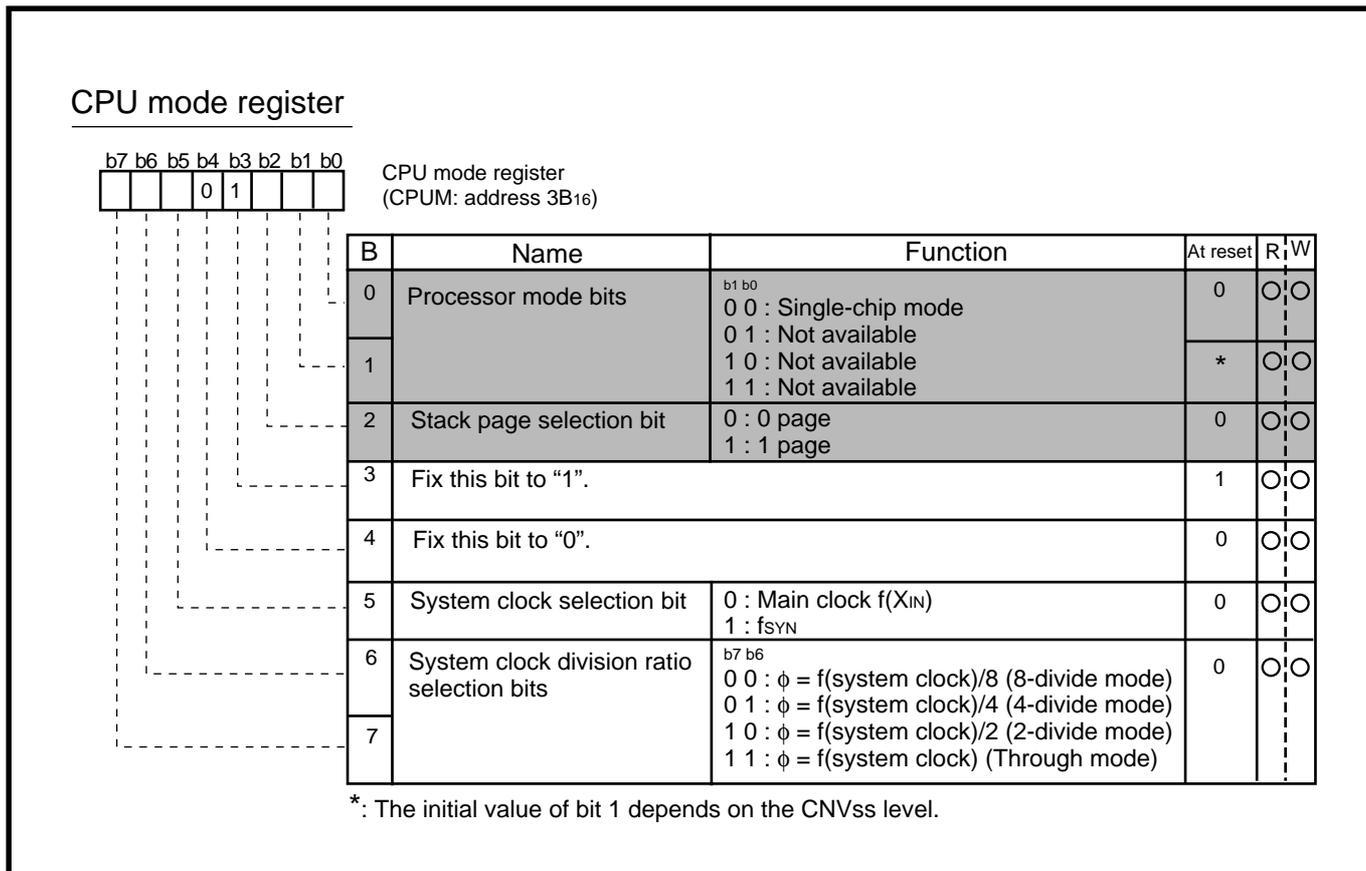


Fig. 2.11.3 Structure of CPU mode register

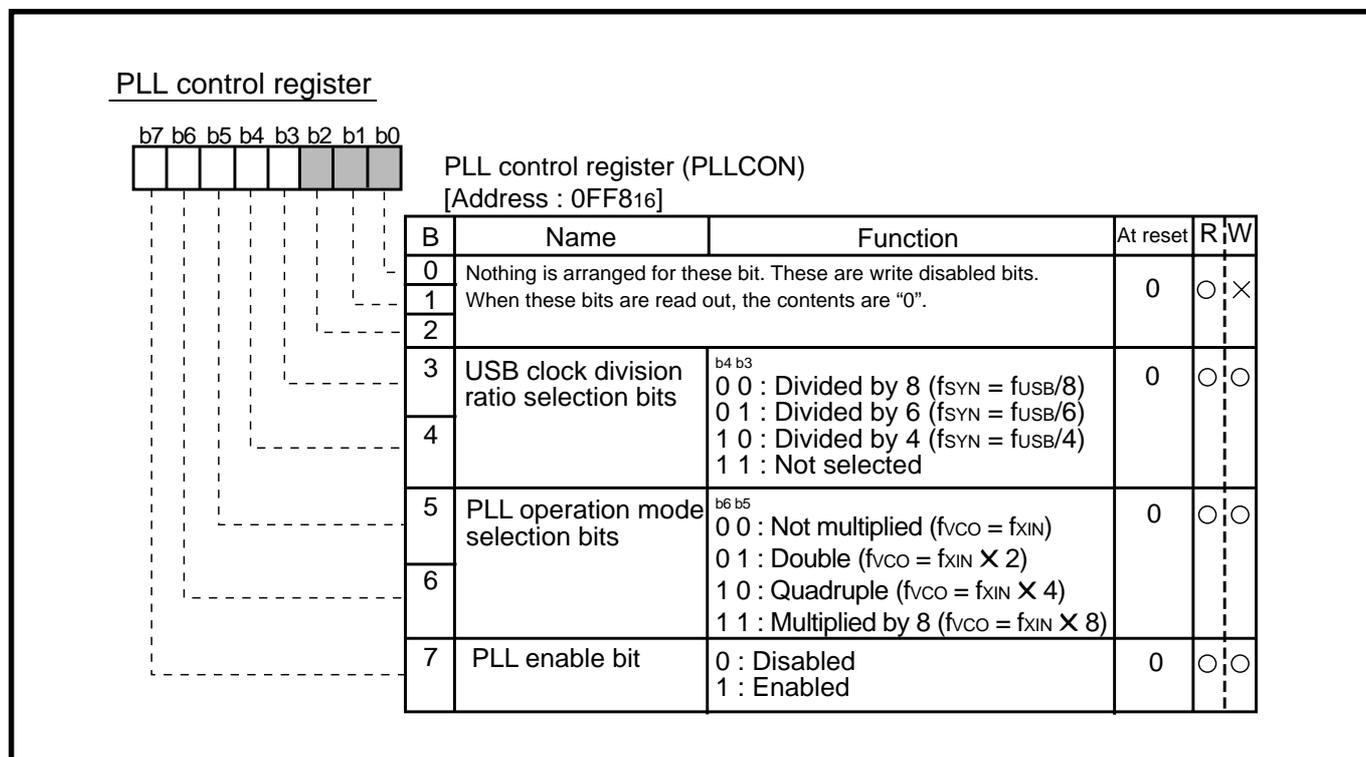


Fig. 2.11.4 Structure of PLL control register

### 2.11.3 Functional description

The frequency synthesizer generates the 48 MHz clock which is multiples of the external input reference  $f(X_{IN})$  and is needed for operating USB function. When using the USB function, set PLL enable bit of PLL control register (PLLCON: address 0FF8<sub>16</sub>) to "1" (enabled) to send the 48 MHz PLL output clock ( $f_{VCO}$ ) into USB function control unit. Figure 2.11.5 shows the block diagram for the frequency synthesizer circuit.

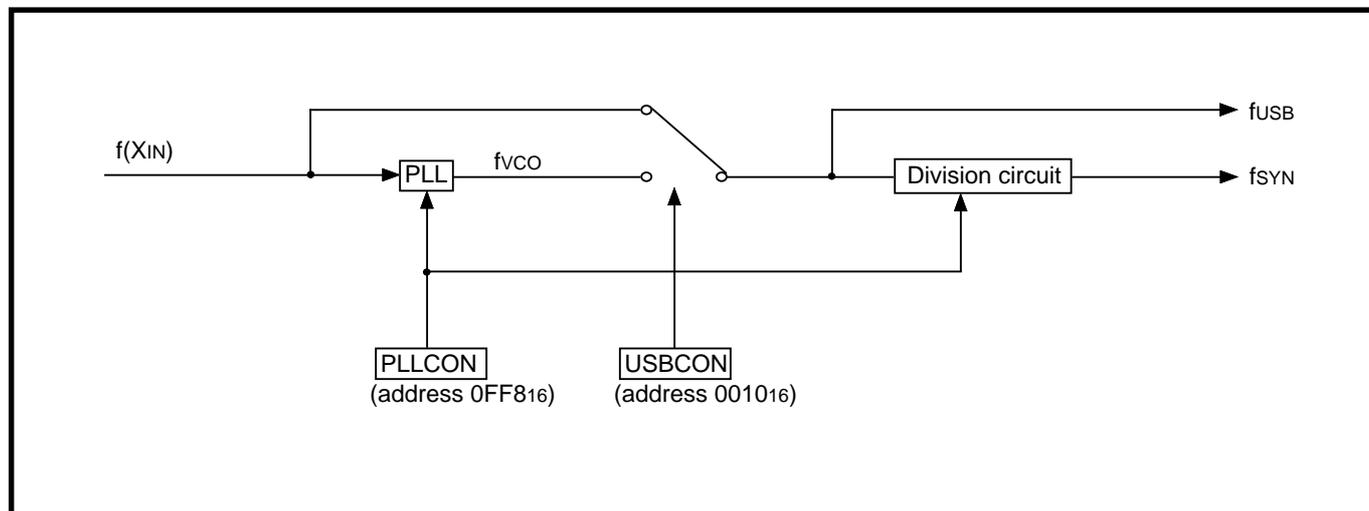


Fig. 2.11.5 Block diagram for frequency synthesizer circuit

#### ● $f_{VCO}$ (PLL output clock)

$f_{VCO}$  is generated by multiplying PLL input clock according to the contents of PLL operation mode selection bits (bits 6, 5 of PLLCON), where

$f_{VCO} = f(X_{IN}) \times n$ ,  $n$ : value selected by PLL operation mode selection bits

Set PLL operation mode selection bits so that  $f_{VCO}$  may be set to 48 MHz.

While the PLL enable bit is "0" (disabled),  $f_{VCO}$  retains "L" level (except when PLL operation mode selection bits are set to "00<sub>2</sub>").

Table 2.11.1 shows the example of PLL operation mode selection bits setting.

Table 2.11.1 PLL operation mode selection bits setting example

$f(X_{IN})$	PLL operation mode selection bits *	$f_{VCO}$
6 MHz	11	48 MHz
12 MHz	10	48 MHz

\*: PLL control register (bits 6,5)

Furthermore, when PLL operation mode selection bits are set to "00<sub>2</sub>", the clock input into PLL is used as  $f_{VCO}$ , which is not multiplied, regardless of PLL operation enabled or disabled.

#### ● $f_{USB}$ (USB clock)

Either  $f(X_{IN})$  (main clock) or  $f_{VCO}$  (PLL output clock) can be selected for  $f_{USB}$  by USB clock select bit of USB control register (bit6 of USBCON: address 0010<sub>16</sub>), and it is supplied to the USB function control circuit. When supplying  $f_{VCO}$  to the USB function control circuit, after setting PLL enable bit to "1" (enabled) and then set USB clock select bit to "1" (USB clock).

### ● $f_{\text{SYN}}$ ( $f_{\text{USB}}$ division clock)

According to the setting of the USB clock division ratio selection bits (bits 4, 3 of PLLCON), the division clock of  $f_{\text{USB}}$  is supplied to  $f_{\text{SYN}}$ .

$f_{\text{SYN}} = f_{\text{USB}} / m$ ,  $m$ : value selected by USB clock division ratio selection bits

Set the USB clock division ratio selection bits so that  $f_{\text{SYN}}$  may be set to 6 MHz, 8 MHz or 12 MHz.

When using  $f_{\text{SYN}}$  as internal system clock, set the system clock selection bit of CPU mode register (bit 5 of CPUM: address 003B<sub>16</sub>) to "1" ( $f_{\text{SYN}}$ ).

Table 2.11.2 shows the example of USB clock division ratio selection bits setting.

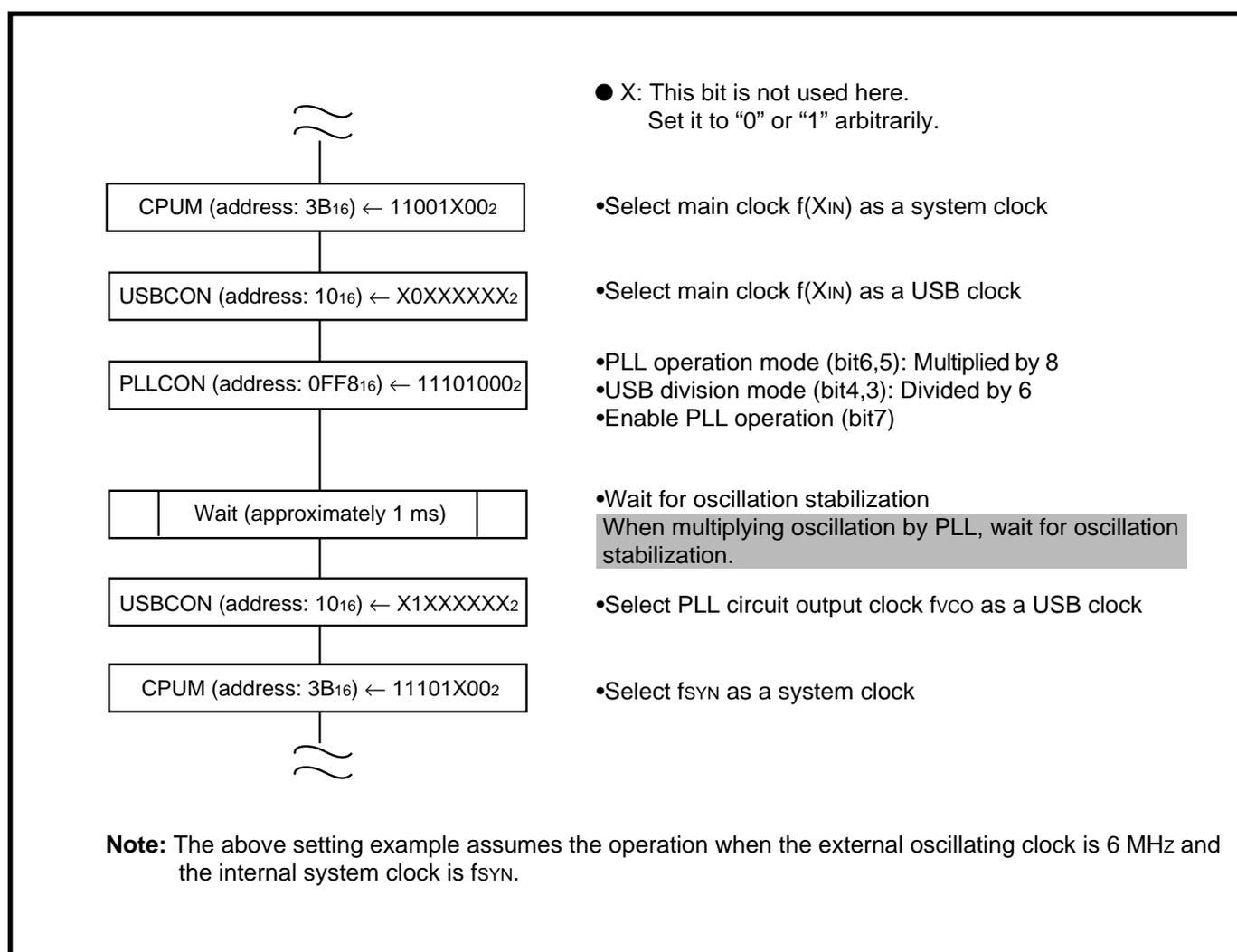
**Table 2.11.2 USB clock division ratio selection bits setting example**

$f_{\text{USB}}$	USB clock division ratio selection bits *	$f_{\text{SYN}}$
48 MHz	00	6 MHz
	01	8 MHz
	10	12 MHz

\*: PLL control register (bit4,3)

### ● Setting for starting up PLL circuit when hardware reset

Figure 2.11.6 shows the example of related registers setting.



**Fig. 2.11.6 Related registers setting when hardware reset**

● **Procedure for stop and return of PLL circuit when stop mode**

Figure 2.11.7 shows the stop procedure of PLL circuit, and figure 2.11.8 shows the return procedure of PLL circuit.

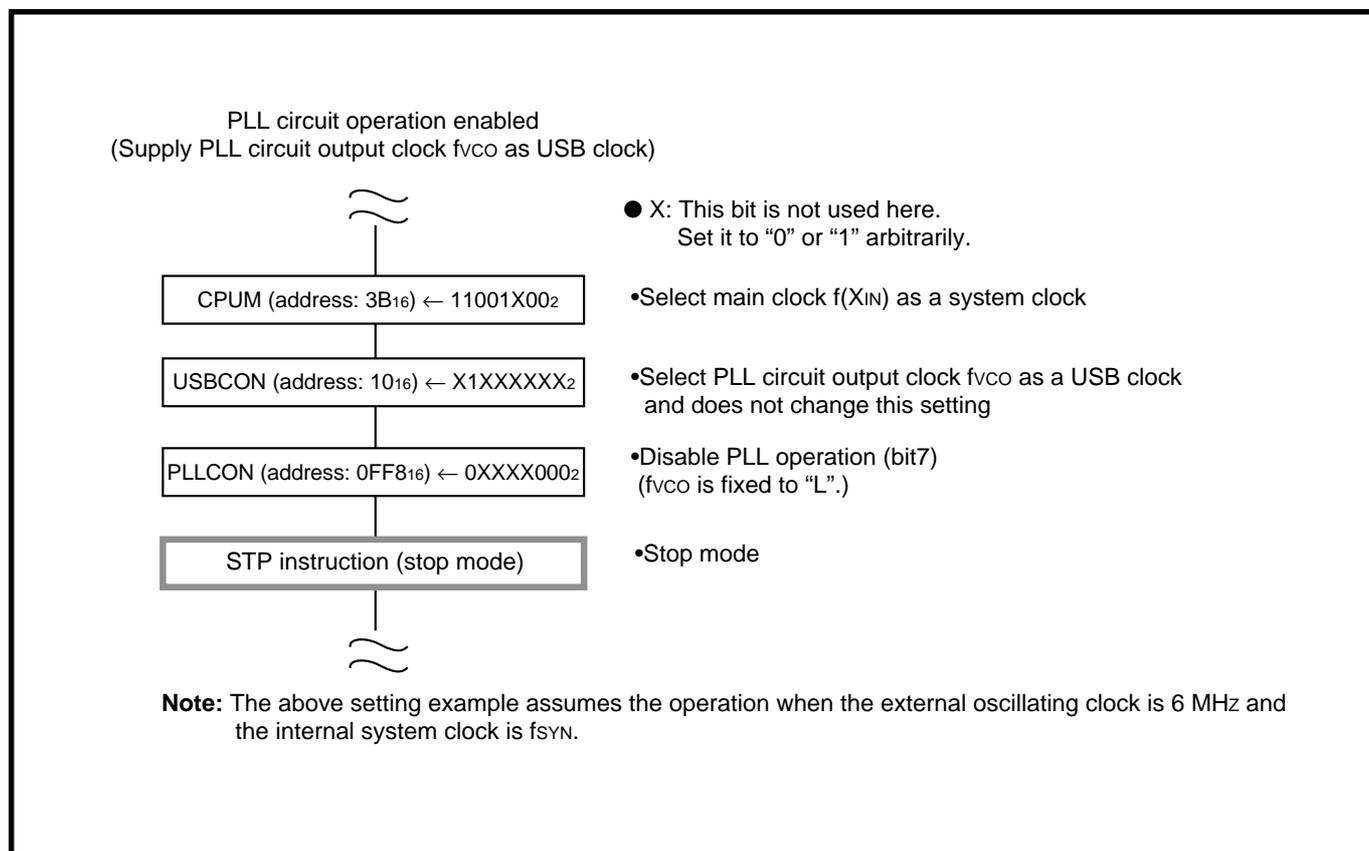


Fig. 2.11.7 Related registers setting when stop mode

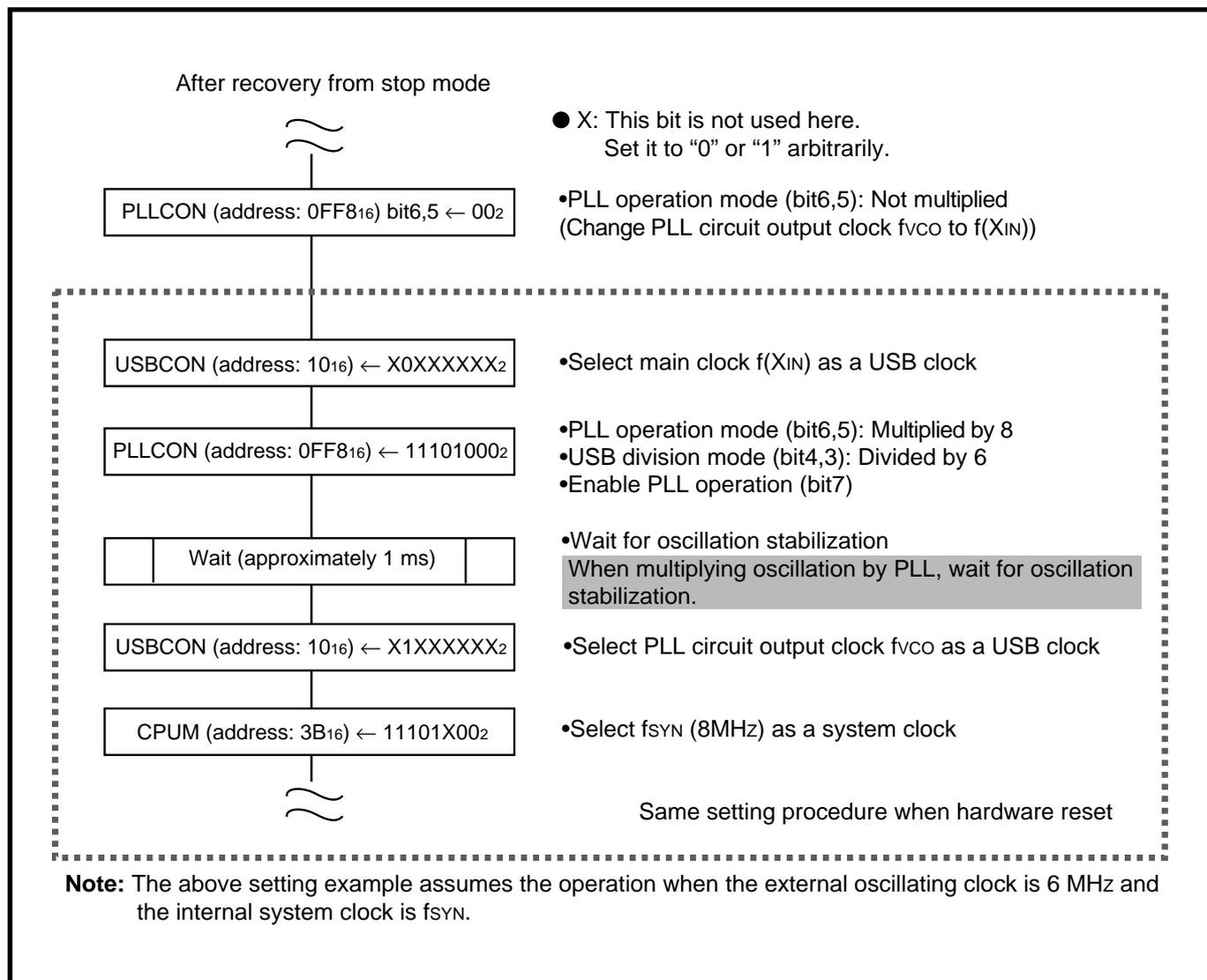


Fig. 2.11.8 Related registers setting when recovery from stop mode

#### 2.11.4 Notes on PLL

- 6 MHz or 12 MHz external oscillator can be connected as an input reference clock ( $f(X_{IN})$ ). When using the frequency synthesized clock function, we recommend using the fastest frequency possible of  $f(X_{IN})$  as an input clock reference for the PLL.
- When enabling PLL operation from PLL disabled status (disabled when reset), set the USB clock select bit of USBCON to "0" ( $f(X_{IN})$ ) to operate with the main clock ( $f(X_{IN})$ ).
- When supplying  $f_{VCO}$  to the USB block after setting PLL operation enable bit to "1" (PLL enabled), wait for the oscillation stable time (1 ms or less) of PLL to avoid any instability caused by the clock, then set USB clock select bit to "1" (USB clock).
- When selecting  $f_{SYN}$  as an internal system clock,  $f_{USB}$  must be 48 MHz.
- When selecting  $f_{SYN}$  as an internal system clock, change the system clock selection bit to main clock ( $f(X_{IN})$ ) before executing STP instruction. It is because the following are needed for the low-power consumption:
  - $f_{USB}$  must be stopped by disabling PLL operation in Stop mode.
  - The timer 1 for waiting oscillation stabilization when returning from Stop mode will require the input count source.

## 2.12 Clock generating circuit

This paragraph explains the registers setting method and the notes related to the clock generating circuit.

### 2.12.1 Memory map

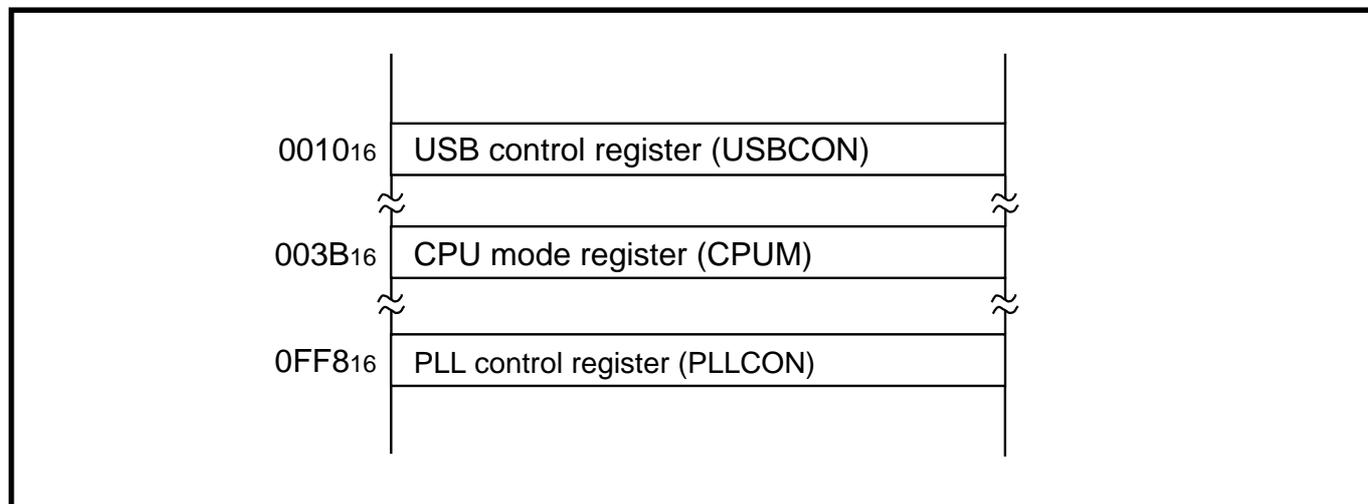


Fig. 2.12.1 Memory map of registers related to clock generating circuit

### 2.12.2 Related registers

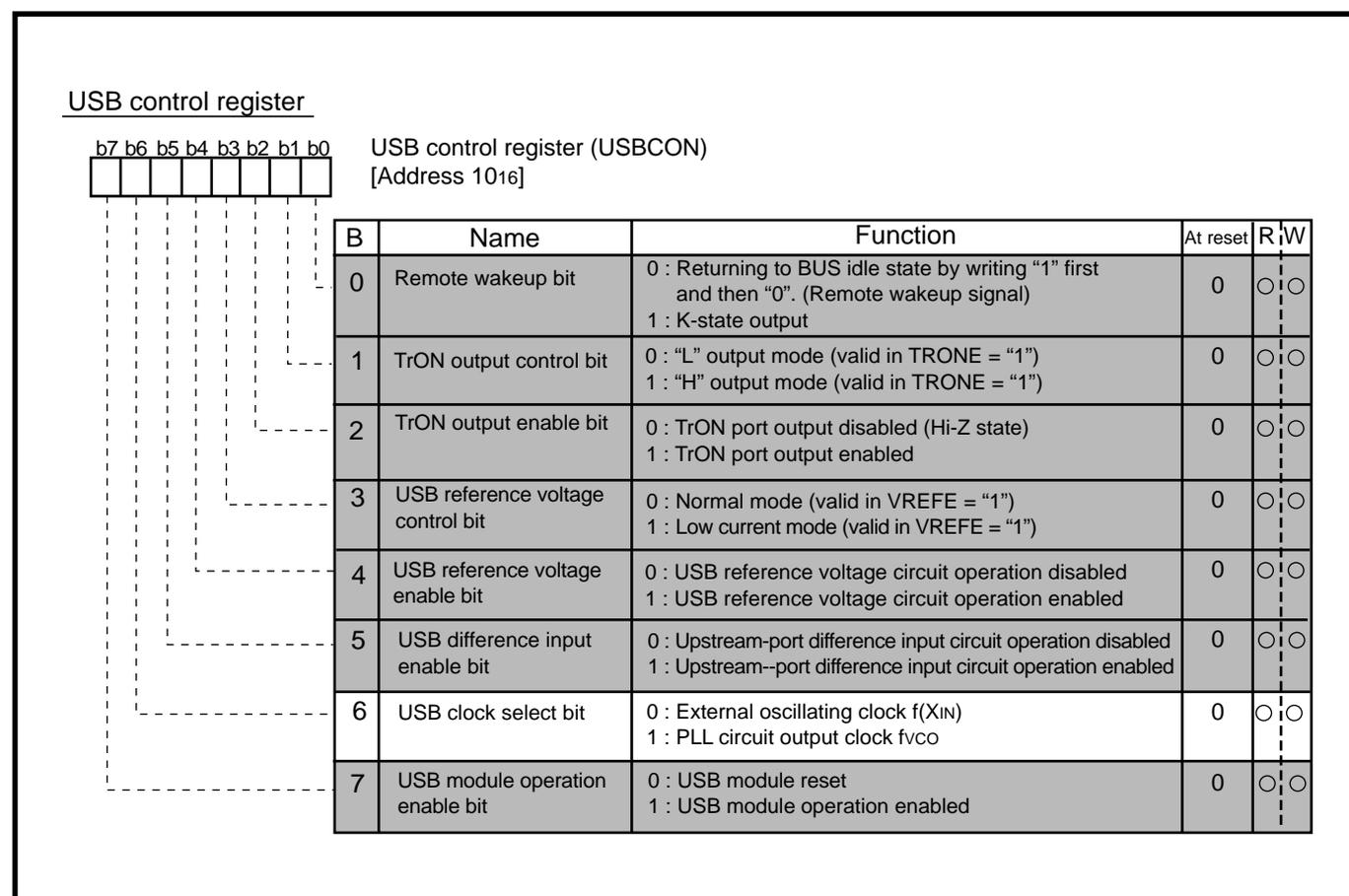


Fig. 2.12.2 Structure of USB control register

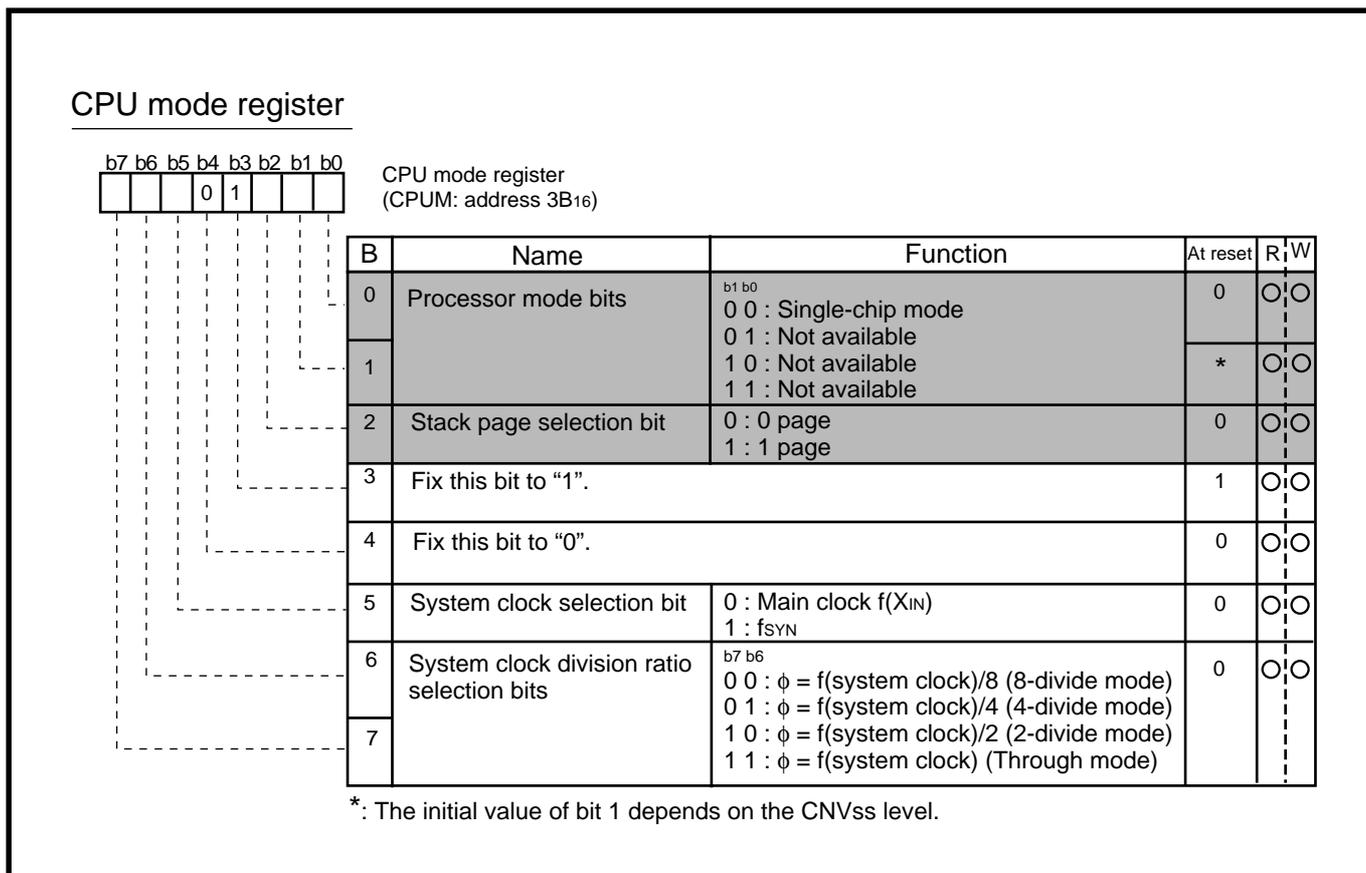


Fig. 2.12.3 Structure of CPU mode register

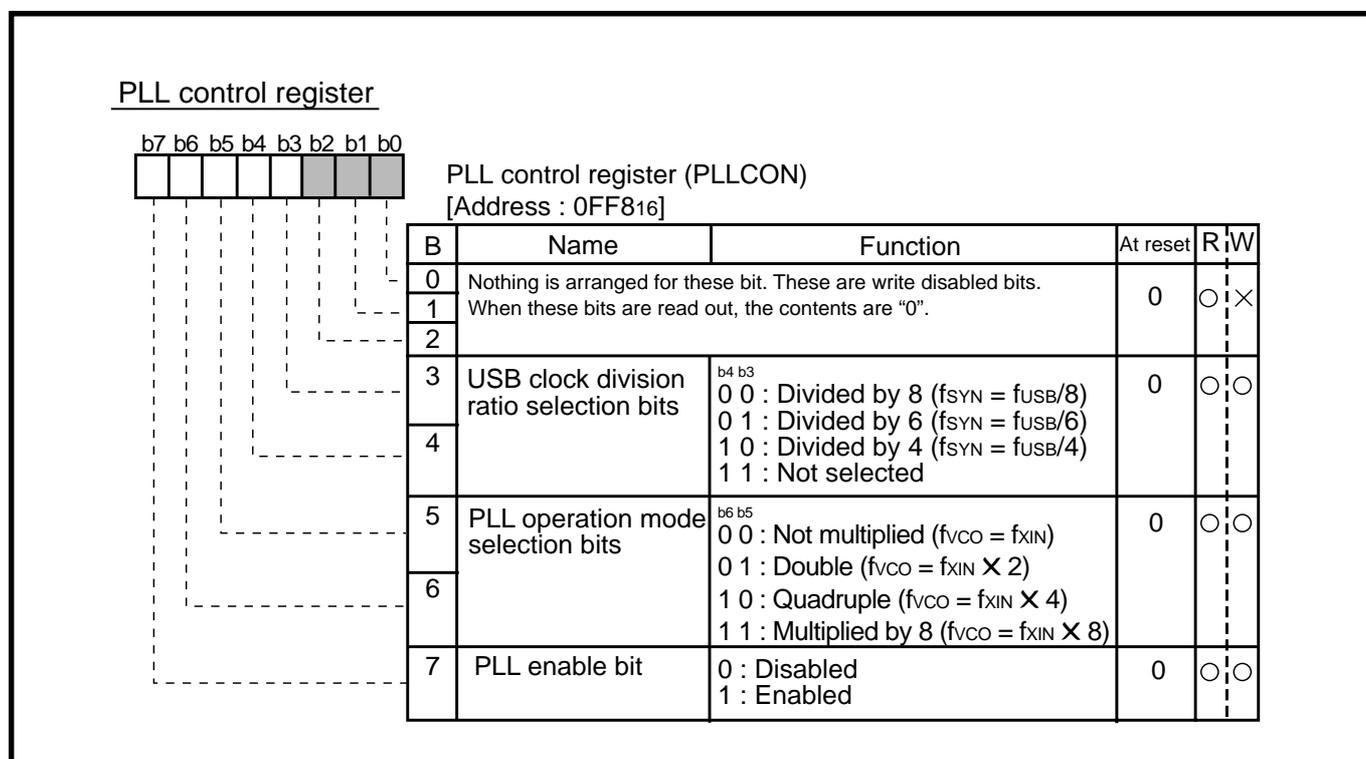


Fig. 2.12.4 Structure of PLL control register

### 2.12.3 Oscillation control

Either can be selected as an internal system clock between the following two by system clock selection bit.

- Main clock  $f(X_{IN})$
- $f_{SYN}$  ( $f_{USB}$  division clock)

Any one can be selected as an internal clock  $\phi$  among the following four by system clock division ratio selection bits.

- $f(X_{IN})$  or  $f_{SYN}/8$  (8-divide mode)
- $f(X_{IN})$  or  $f_{SYN}/4$  (4-divide mode)
- $f(X_{IN})$  or  $f_{SYN}/2$  (2-divide mode)
- $f(X_{IN})$  or  $f_{SYN}$  (Through mode)

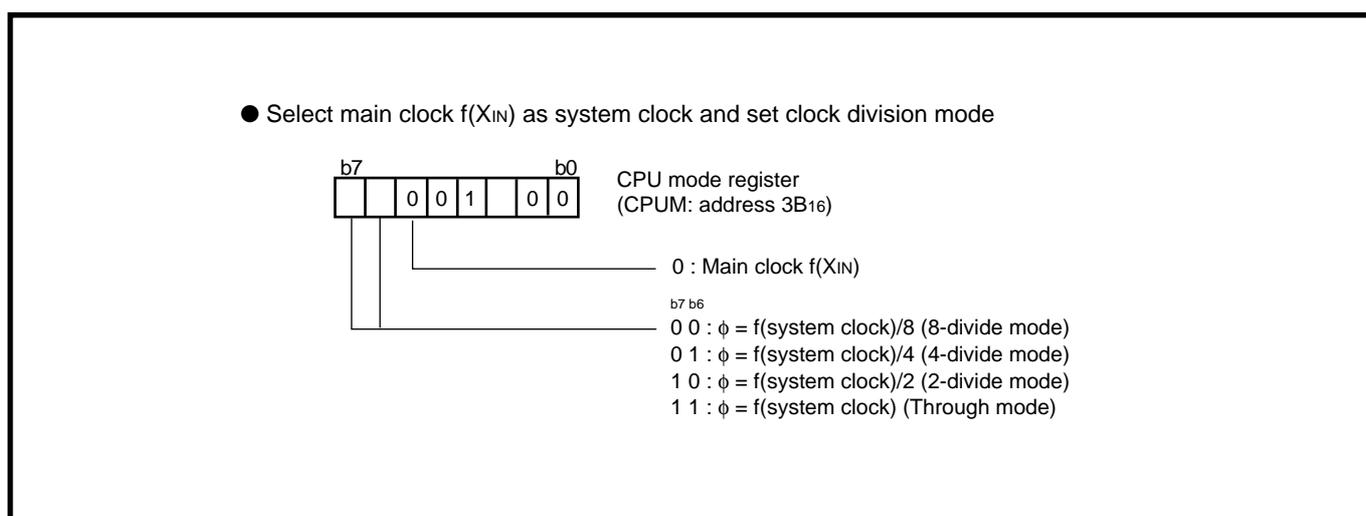
#### (1) Generation of internal clock $f(\phi)$ using main clock $f(X_{IN})$

Table 2.12.1 shows the example of internal clock  $f(\phi)$  generation using main clock  $f(X_{IN})$ ; Figure 2.12.5 shows the related registers setting.

**Table 2.12.1 Example of internal clock  $f(\phi)$  generation using main clock  $f(X_{IN})$**

System clock	System clock division ratio selection bits *	$f(\phi)$	Power source voltage $V_{CC}$ [V]
6 MHz	0 0	0.75 MHz	3.00 to 5.25
	0 1	1.5 MHz	
	1 0	3 MHz	
	1 1	6 MHz	
8 MHz	0 0	1 MHz	4.00 to 5.25
	0 1	2 MHz	
	1 0	4 MHz	
	1 1	8 MHz	
12 MHz	0 0	1.5 MHz	4.00 to 5.25
	0 1	3 MHz	
	1 0	6 MHz	

\*: CPU mode register (bits 7,6)



**Fig. 2.12.5 Related registers setting**

**(2) Generation of internal clock  $f(\phi)$  using  $f_{\text{SYN}}$  ( $f_{\text{USB}}$  division clock)**

Table 2.12.2 shows the example of internal clock  $f(\phi)$  generation using  $f_{\text{SYN}}$ ; Figure 2.12.6 shows the related registers setting.

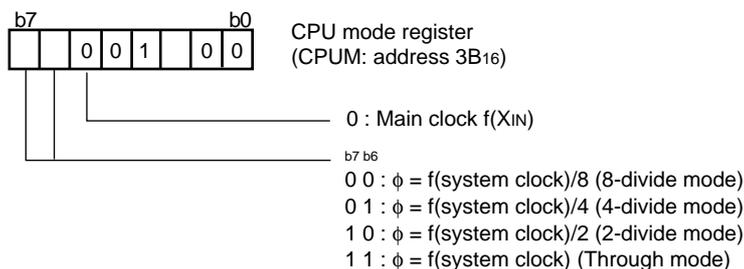
**Table 2.12.2 Example of internal clock  $f(\phi)$  generation using  $f_{\text{SYN}}$** 

$f_{\text{USB}}$	USB clock division ratio selection bits *1	$f_{\text{SYN}}$	System clock division ratio selection bits *2	$f(\phi)$	Power source voltage $V_{\text{CC}}$ [V]
48 MHz	0 0	6 MHz	0 0	0.75 MHz	3.00 to 5.25
			0 1	1.5 MHz	
			1 0	3 MHz	
			1 1	6 MHz	
	0 1	8 MHz	0 0	1 MHz	4.00 to 5.25
			0 1	2 MHz	
			1 0	4 MHz	
			1 1	8 MHz	
	1 1	12 MHz	0 0	1.5 MHz	
			0 1	3 MHz	
			1 0	6 MHz	

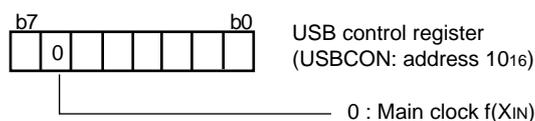
\*1: PLL control register (bits 4,3)

\*2: CPU mode register (bits 7,6)

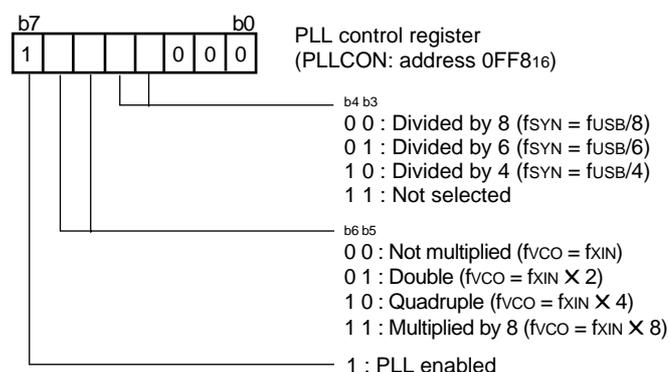
1. Select main clock  $f(X_{IN})$  as system clock and set clock division mode.



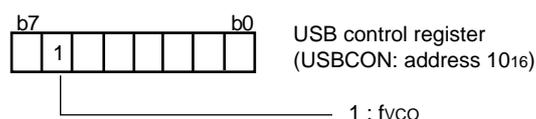
2. Select main clock  $f(X_{IN})$  as USB clock.



3. Enable PLL circuit, and generating PLL output clock ( $f_{VCO}$ ) 48 MHz and  $f_{SYN}$ .



4. Select PLL output clock ( $f_{VCO}$ ) as USB clock.



5. Select  $f_{SYN}$  as system clock.

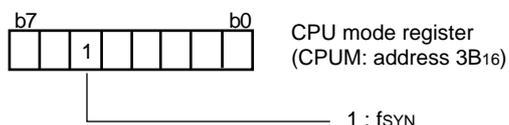


Fig. 2.12.6 Related registers setting

**Note:** When selecting  $f_{SYN}$  as an internal system clock, refer to "2.11 Frequency synthesizer (PLL)" for details concerning how to generate  $f_{USB}$  (USB clock) from  $f(X_{IN})$  and the notes on PLL circuit.

## 2.13 Standby function

The 38K2 group is provided with standby functions to stop the CPU by software and put the CPU into the low-power operation.

The following two types of standby functions are available.

- Stop mode using STP instruction
- Wait mode using WIT instruction

### 2.13.1 Memory map

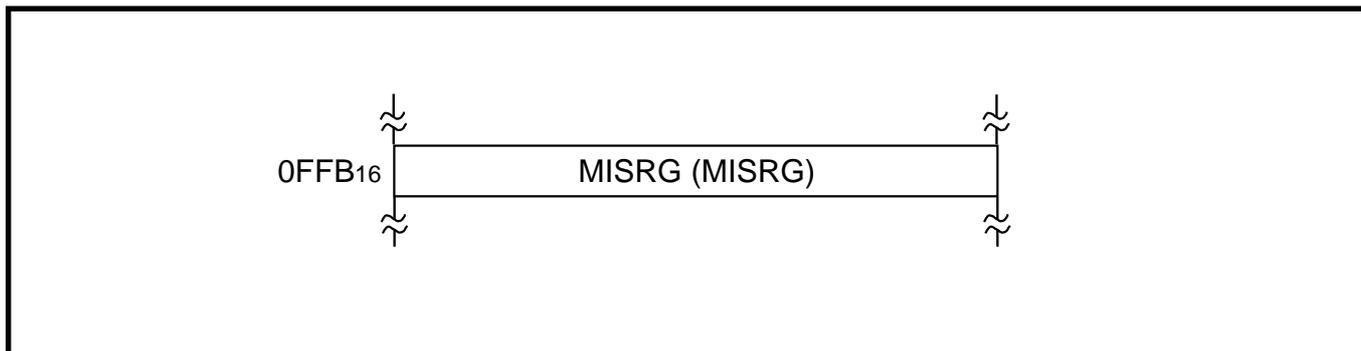


Fig. 2.13.1 Memory map of registers related to standby function

### 2.13.2 Related registers

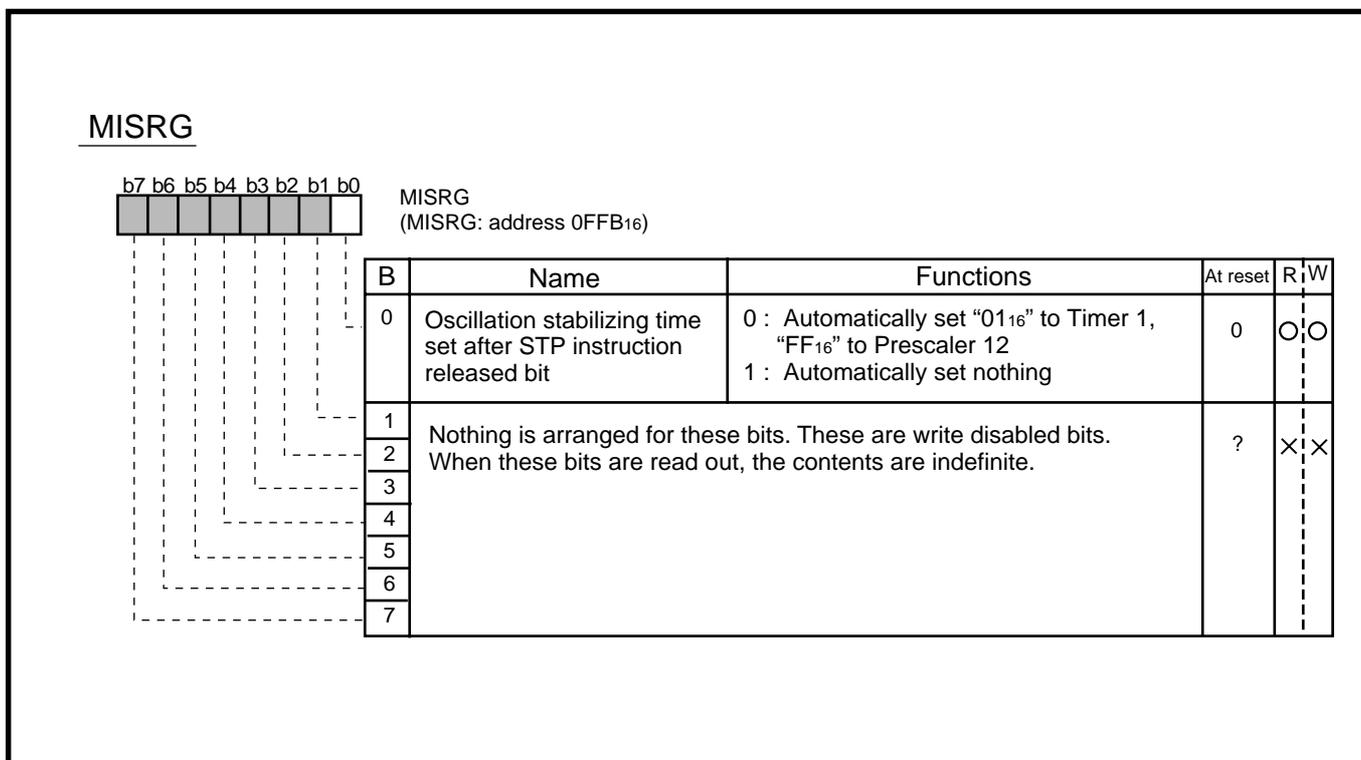


Fig. 2.13.2 Structure of MISRG

### 2.13.3 Stop mode

The stop mode is set by executing the STP instruction. In the stop mode, the oscillation of clock ( $X_{IN}$ – $X_{OUT}$ ) stops and the internal clock  $\phi$  stops at the “H” level. The CPU stops and peripheral units stop operating. As a result, power dissipation is reduced.

#### (1) State in stop mode

Table 2.13.1 shows the state in the stop mode.

**Table 2.13.1 State in stop mode**

Item	State in stop mode
Oscillation	Stopped.
CPU	Stopped.
Internal clock $\phi$	Stopped at “H” level.
I/O ports P0–P6	Retains the state at the STP instruction execution.
Timer	Stopped. (Timers 1, 2, X) However, Timers X can be operated in the event counter mode.
Watchdog timer	Stopped.
Serial I/O	Stopped. However, these can be operated only when an external clock is selected.
USB function	Stopped.
HUB function	Stopped.
External BUS interface	Stopped.
A/D converter	Stopped.
Comparator	Stopped.

**(2) Release of stop mode**

The stop mode is released by a reset input or by the occurrence of an interrupt request. Note the differences in the restoration process according to reset input or interrupt request, as described below.

**■Restoration by reset input**

The stop mode is released by holding the  $\overline{\text{RESET}}$  pin to the “L” input level during the stop mode. Oscillation is started when all ports are in the input state and the stop mode of the main clock ( $X_{\text{IN}}-X_{\text{OUT}}$ ) is released.

Oscillation is unstable when restarted. For this reason, time for stabilizing of oscillation (oscillation stabilizing time) is required. The input of the  $\overline{\text{RESET}}$  pin should be held at the “L” level until oscillation stabilizes.

When the  $\overline{\text{RESET}}$  pin is held at the “L” level for 16 cycles or more of  $X_{\text{IN}}$  after the oscillation has stabilized, the microcomputer will go to the reset state. After the input level of the  $\overline{\text{RESET}}$  pin is returned to “H”, the reset state is released in approximately 10.5 to 18.5 cycles of the  $X_{\text{IN}}$  input. Figure 2.13.3 shows the oscillation stabilizing time at restoration by reset input.

At release of the stop mode by reset input, the internal RAM retains its contents previous to the reset. However, the previous contents of the CPU register and SFR are not retained. For more details concerning reset, refer to “2.10 Reset”.

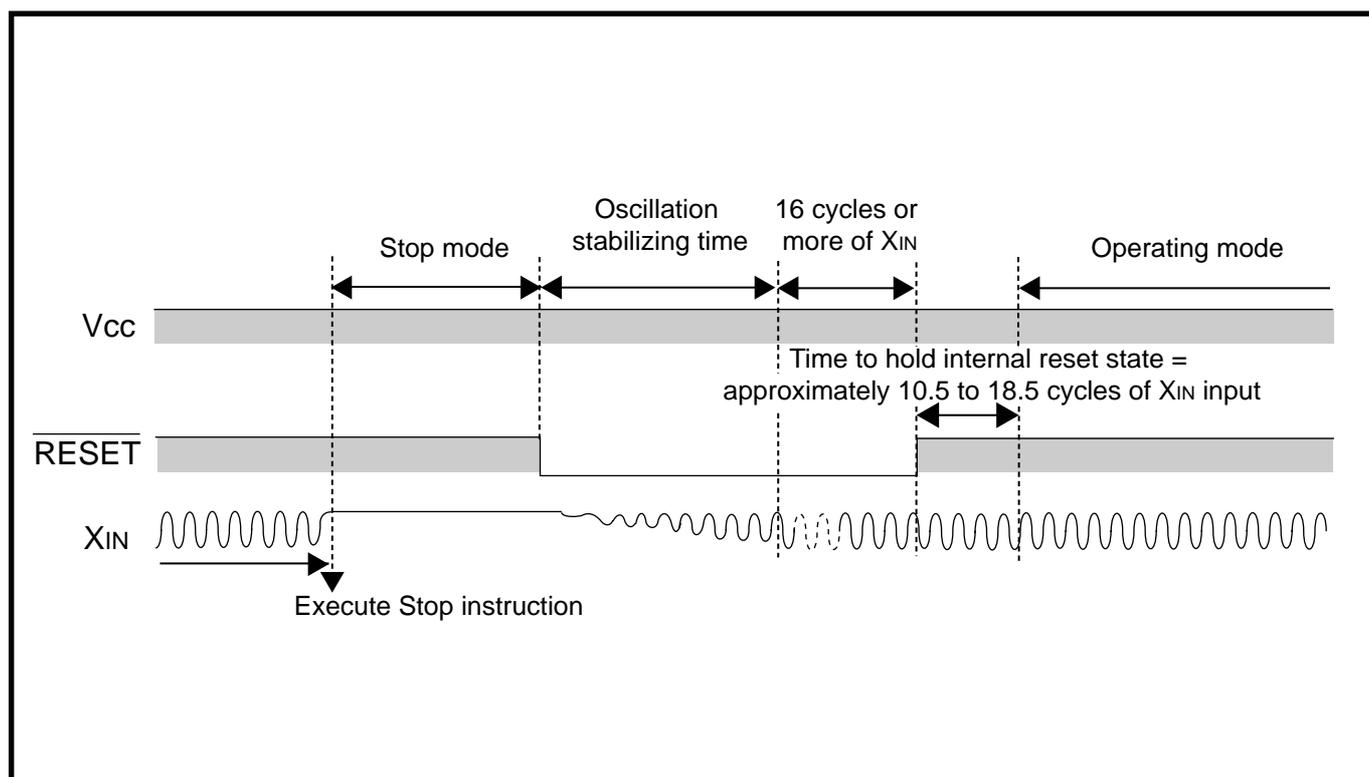


Fig. 2.13.3 Oscillation stabilizing time at restoration by reset input

### ■Restoration by interrupt request

The occurrence of an interrupt request in the stop mode releases the stop mode. As a result, oscillation is resumed. The interrupts available for restoration are:

- INT<sub>0</sub>, INT<sub>1</sub>
- CNTR<sub>0</sub>
- Serial I/O using an external clock
- Timer X using an external event count
- Key input (key-on wake-up)
- USB function (resume)

However, when using any of these interrupt requests for restoration from the stop mode, in order to enable the selected interrupt, you must execute the STP instruction after setting the following conditions.

[Necessary register setting]

- ① Interrupt disable flag I = "0" (interrupt enabled)
- ② Timer 1 interrupt enable bit = "0" (interrupt disabled)
- ③ Interrupt request bit of interrupt source to be used for restoration = "0" (no interrupt request issued)
- ④ Interrupt enable bit of interrupt source to be used for restoration = "1" (interrupts enabled)

For more details concerning interrupts, refer to "2.2 Interrupts".

Oscillation is unstable when restarted. For this reason, time for stabilizing of oscillation (oscillation stabilizing time) is required. For restoration by an interrupt request, waiting time prior to supplying internal clock  $\phi$  to the CPU is automatically generated\*2 by Prescaler 12 and Timer 1\*1. This waiting time is reserved as the oscillation stabilizing time on the system clock side. The supply of internal clock  $\phi$  to the CPU is started at the Timer 1 underflow.

Figure 2.13.4 shows an execution sequence example at restoration by the occurrence of an INT<sub>0</sub> interrupt request.

- \*1: If the STP instruction is executed when the oscillation stabilizing time set after STP instruction released bit is "0", "FF<sub>16</sub>" and "01<sub>16</sub>" are automatically set in the Prescaler 12 counter/latch and Timer 1 counter/latch, respectively. When the oscillation stabilizing time set after STP instruction released bit is "1", nothing is automatically set to either Prescaler 12 or Timer 1. For this reason, any suitable value can be set to Prescaler 12 and Timer 1 for the oscillation stabilizing time.
- \*2: Immediately after the oscillation is started, the count source is supplied to the prescaler 12 so that a count operation is started.

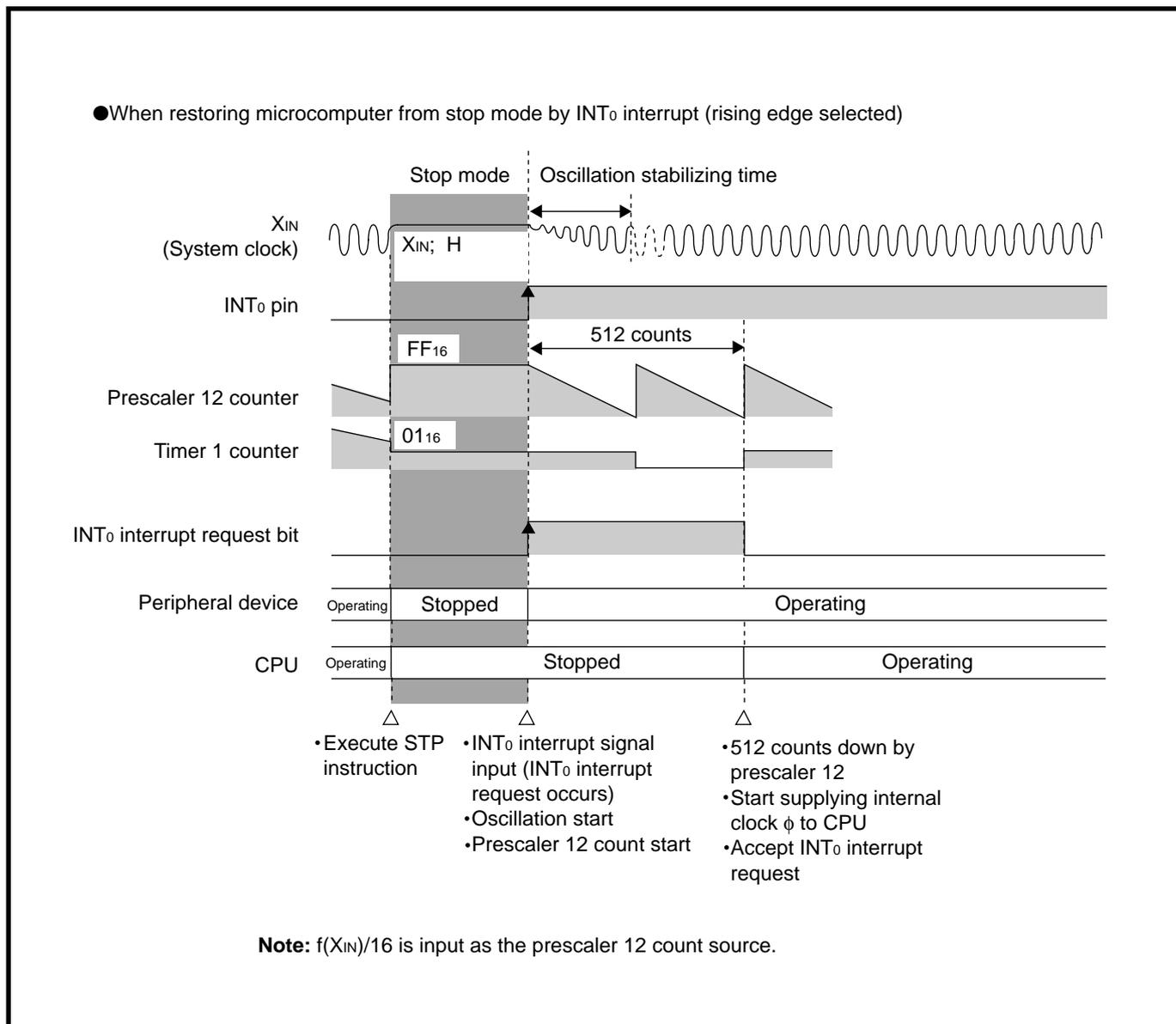


Fig. 2.13.4 Execution sequence example at restoration by occurrence of INT<sub>0</sub> interrupt request

### (3) Notes on using stop mode

#### ■Register setting

Since values of the prescaler 12 and Timer 1 are automatically reloaded when returning from the stop mode, set them again, respectively. (When the oscillation stabilizing time set after STP instruction released bit is "0")

#### ■Clock restoration

When the main clock side is set as a system clock, the oscillation stabilizing time for approximately 8,000 cycles of the X<sub>IN</sub> input is reserved at restoration from the stop mode.

### 2.13.4 Wait mode

The wait mode is set by execution of the WIT instruction. In the wait mode, oscillation continues, but the internal clock  $\phi$  stops at the “H” level.

The CPU stops, but most of the peripheral units continue operating.

#### (1) State in wait mode

The continuation of oscillation permits clock supply to the peripheral units. Table 2.13.2 shows the state in the wait mode.

**Table 2.13.2 State in wait mode**

Item	State in wait mode
Oscillation	Operating.
CPU	Stopped.
Internal clock $\phi$	Stopped at “H” level.
I/O ports P0–P6	Retains the state at the WIT instruction execution.
Timer	Operating.
Watchdog timer	Operating.
Serial I/O	Operating.
USB function	Operating.
HUB function	Operating.
External BUS interface	Stopped.
A/D converter	Operating.
Comparator	Operating.

**(2) Release of wait mode**

The wait mode is released by reset input or by the occurrence of an interrupt request. Note the differences in the restoration process according to reset input or interrupt request, as described below.

In the wait mode, oscillation is continued, so an instruction can be executed immediately after the wait mode is released.

**■Restoration by reset input**

The wait mode is released by holding the input level of the  $\overline{\text{RESET}}$  pin at "L" in the wait mode. Upon release of the wait mode, all ports are in the input state, and supply of the internal clock  $\phi$  to the CPU is started. To reset the microcomputer, the  $\overline{\text{RESET}}$  pin should be held at an "L" level for 16 cycles or more of  $X_{\text{IN}}$ . The reset state is released in approximately 10.5 cycles to 18.5 cycles of the  $X_{\text{IN}}$  input after the input of the  $\overline{\text{RESET}}$  pin is returned to the "H" level.

At release of wait mode, the internal RAM retains its contents previous to the reset. However, the previous contents of the CPU register and SFR are not retained.

Figure 2.13.5 shows the reset input time.

For more details concerning reset, refer to "2.10 Reset".

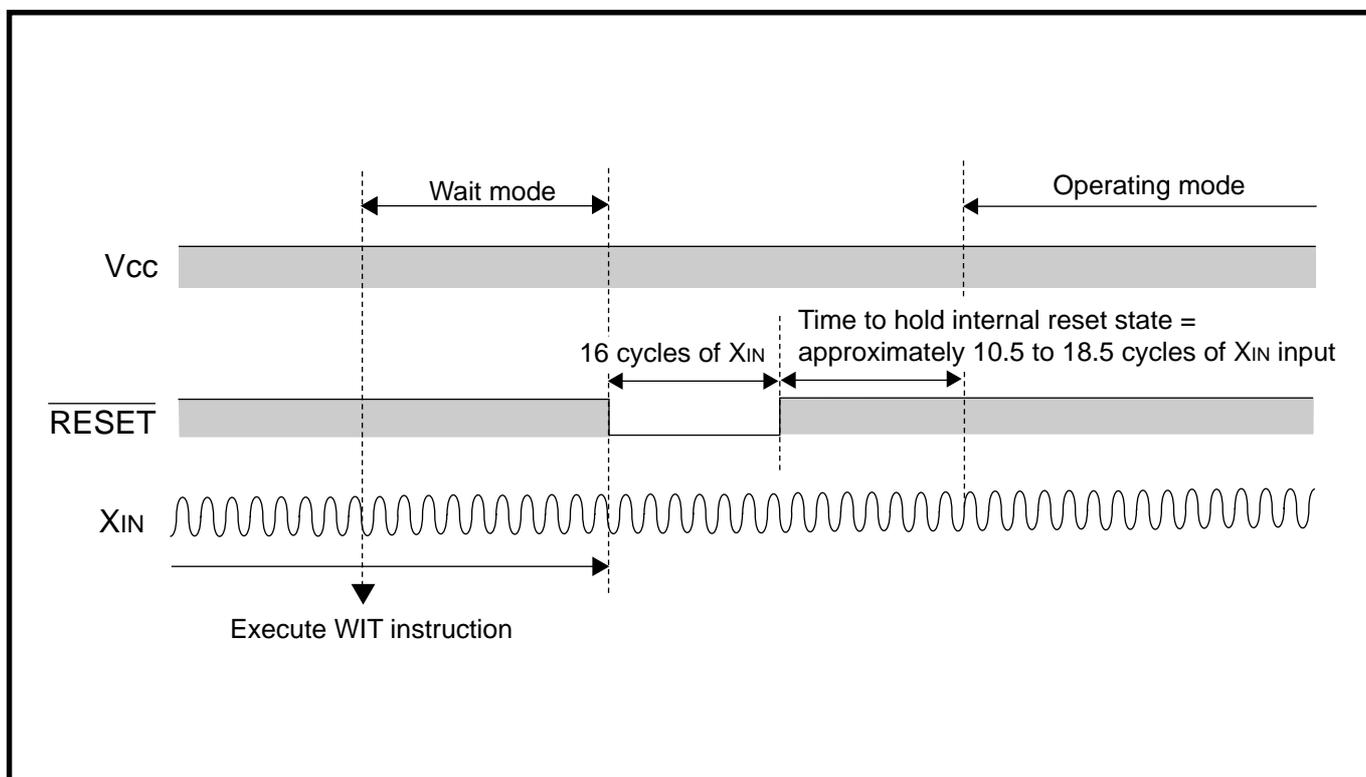


Fig. 2.13.5 Reset input time

### ■ Restoration by interrupt request

In the wait mode, the occurrence of an interrupt request releases the wait mode and supply of the internal clock  $\phi$  to the CPU is started. At the same time, the interrupt request used for restoration is accepted, so the interrupt processing routine is executed.

However, when using an interrupt request for restoration from the wait mode, in order to enable the selected interrupt, you must execute the STP instruction after setting the following conditions.

[Necessary register setting]

- ① Interrupt disable flag I = "0" (interrupt enabled)
- ② Interrupt request bit of interrupt source to be used for restoration = "0" (no interrupt request issued)
- ③ Interrupt enable bit of interrupt source to be used for restoration = "1" (interrupts enabled)

For more details concerning interrupts, refer to "2.2 Interrupts".

### 2.13.5 Notes on stand-by function

In stand-by state\*1 for low-power dissipation, do not make input levels of an input port and an I/O port "undefined".

Pull-up (connect the port to  $V_{CC}$ ) these ports through a resistor.

When determining a resistance value, note the following points:

- External circuit
- Variation of output levels during the ordinary operation

When using built-in pull-up resistor, note on varied current values.

- When setting as an input port: Fix its input level
- When setting as an output port: Prevent current from flowing out to external

#### ● Reason

The potential which is input to the input buffer in a microcomputer is unstable in the state that input levels of an input port and an I/O port are "undefined". This may cause power source current.

\*1 stand-by state :     the stop mode by executing the **STP** instruction  
                               the wait mode by executing the **WIT** instruction

## 2.14 Flash memory

This paragraph explains the registers setting method and the notes related to the flash memory version.

### 2.14.1 Overview

The functions of the flash memory version are similar to those of the mask ROM version except that the flash memory is built-in and some of the SFR area differ from that of the mask ROM version (refer to “2.14.2 Memory map”).

In the flash memory version, the built-in flash memory can be programmed or erased by using the following three modes.

- CPU rewrite mode
- Parallel I/O mode
- Standard serial I/O mode

### 2.14.2 Memory map

38K2 group flash memory version has 32 Kbytes of built-in flash memory.

Figure 2.14.1 shows the memory map of the flash memory version.

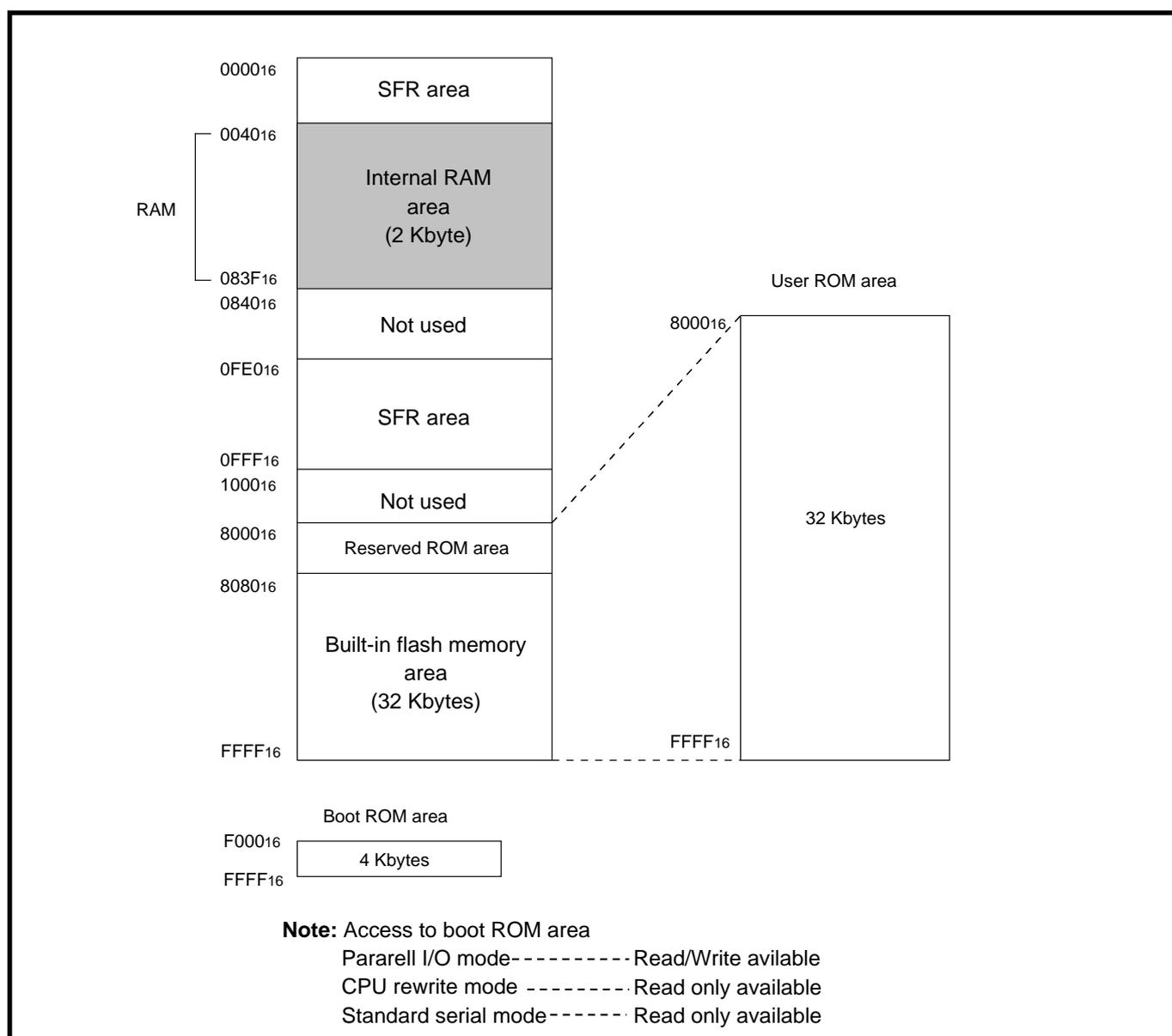


Fig. 2.14.1 Memory map of flash memory version for 38K2 Group

## 2.14.3 Related registers

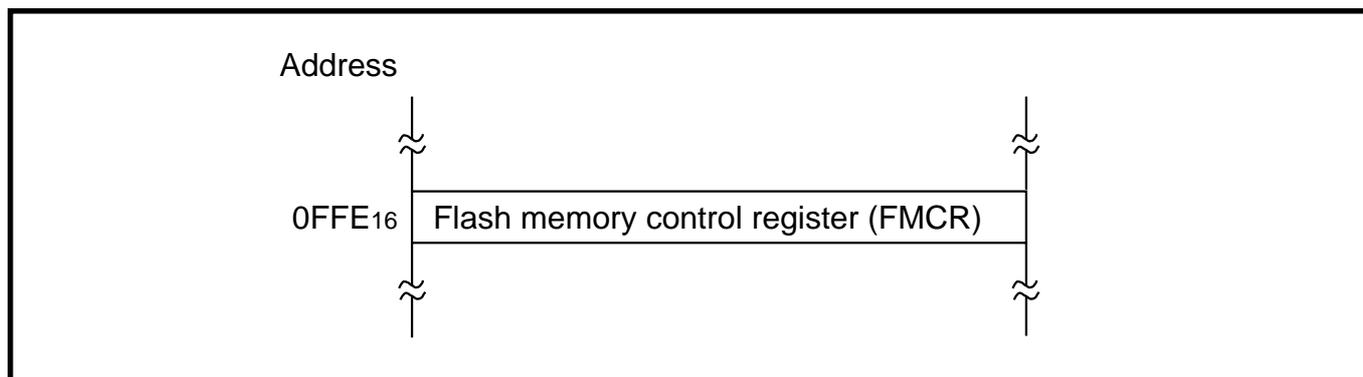


Fig. 2.14.2 Memory map of registers related to flash memory

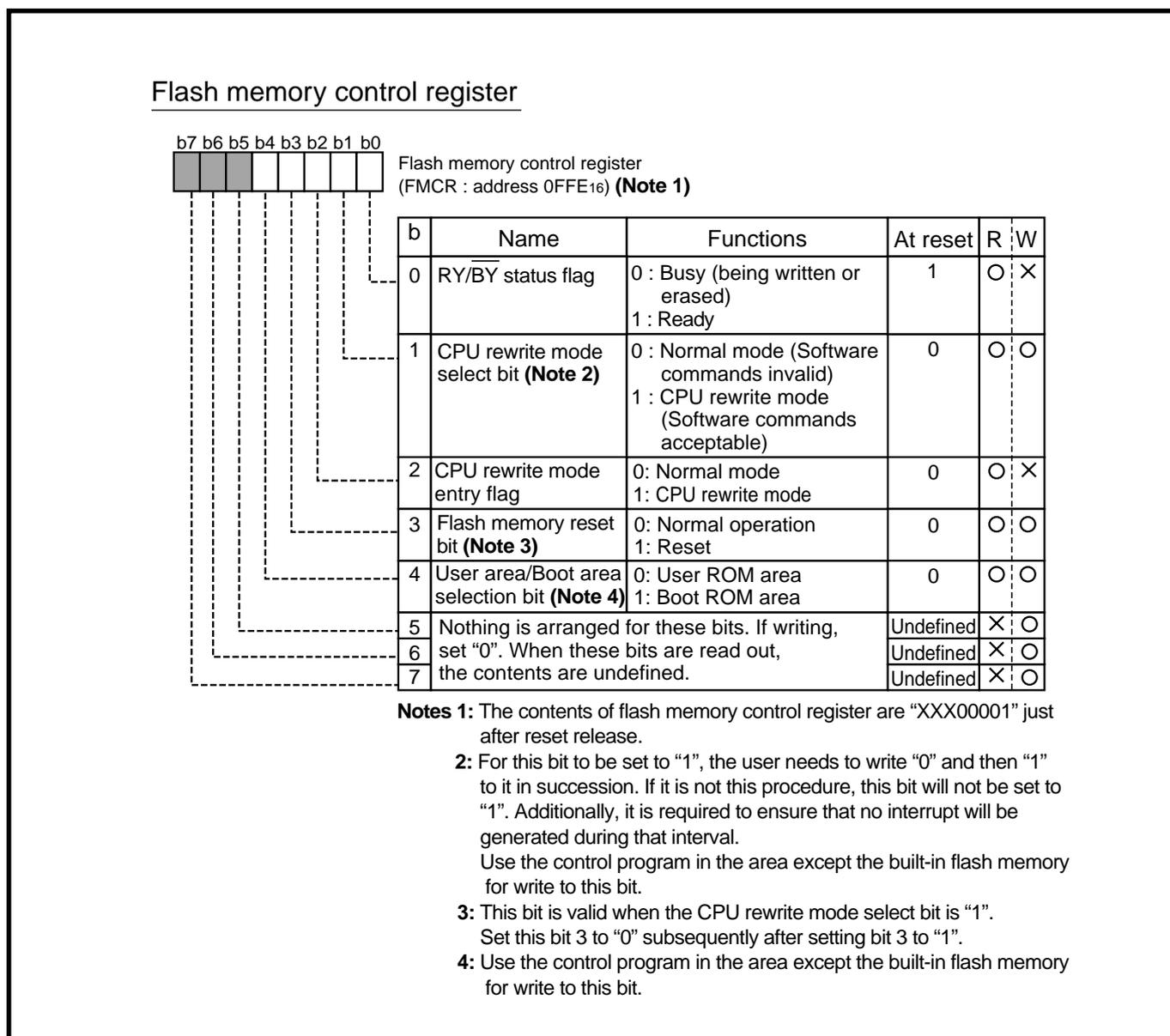


Fig. 2.14.3 Structure of Flash memory control register

#### 2.14.4 Parallel I/O mode

In the parallel I/O mode, program/erase to the built-in flash memory can be performed by a flash programmer (MFW-1).

The memory area of program/erase is from 0F000<sub>16</sub> to 0FFFF<sub>16</sub> (boot ROM area) or from 08000<sub>16</sub> to 0FFFF<sub>16</sub> (user ROM area). Be especially careful when erasing; if the memory area is not set correctly, the products will be damaged eternally.

Table 2.14.1 shows the setting of programmers when programming in the parallel I/O mode.

•MFW-1 provided by Sunny Giken Inc. (<http://www.sunnygiken.co.jp/english/index.html>)

**Table 2.14.1 Setting of programmers when parallel programming**

Products	Parallel adapter	Boot ROM area	User ROM area
M38K29F8HP/LHP	MFW-S18	0F000 <sub>16</sub> to 0FFFF <sub>16</sub>	08000 <sub>16</sub> to 0FFFF <sub>16</sub>
M38K29F8FP/LFP	MFW-S19		

#### 2.14.5 Standard serial I/O mode

Table 2.14.2 shows a pin connection example (4 wires) between the programmer (MFW-1) and the microcomputer when programming in the standard serial I/O mode.

•MFW-1 provided by Sunny Giken Inc. (<http://www.sunnygiken.co.jp/english/index.html>)

**Table 2.14.2 Connection example to flash programmer when serial programming (4 wires)**

Function	MFW-1		38K2 Group flash memory version	
	Signal name	MFW-1 side connector Line number	Pin name	Pin number
Transfer clock input	CLK	3	P4 <sub>2</sub> /E <sub>x</sub> TC/S <sub>CLK</sub>	53
Serial data input	RxD	10	P4 <sub>0</sub> /E <sub>x</sub> DREQ/RxD	51
Serial data output	TxD	4	P4 <sub>1</sub> /E <sub>x</sub> DACK/TxD	52
Transmit/Receive enable output	BUSY	2	P4 <sub>3</sub> /E <sub>x</sub> A1/S <sub>RDY</sub>	54
V <sub>PP</sub> input	CNV <sub>SS</sub>	1	CNV <sub>SS</sub>	7
Reset input	RESET	8	RESET	8
Target board power source monitor input	V <sub>CC</sub> ( <b>Note 2</b> )	1	V <sub>CC</sub> , PV <sub>CC</sub> , DV <sub>CC</sub> ( <b>Note 2</b> )	14, 21, 22
GND	GND ( <b>Note 1</b> )	7	V <sub>SS</sub> , PV <sub>SS</sub> ( <b>Note 1</b> )	11, 20

**Notes 1:** When connecting a serial programmer, first connect both GNDs to the same GND level.

**2:** V<sub>CC</sub> power of MFW-1 is supplied from a target board. Power consumption of MFW-1 is Max. 200 mA when serial programming. Therefore, when the current capacity of target board is short, connect AC adapter and supply power source to MFW-1.

### 2.14.6 CPU rewrite mode

In the CPU rewrite mode, issuing software commands through the Central Processing Unit (CPU) can rewrite the built-in flash memory. Accordingly, the contents of the built-in flash memory can be rewritten with the microcomputer itself mounted on board, without using the programmer.

Store the rewrite control program to the built-in flash memory in advance. The built-in flash memory cannot be read in the CPU rewrite mode. Accordingly, after transferring the rewrite control program to the internal RAM, execute it on the RAM.

The following commands can be used in the CPU rewrite mode: read array, read status register, clear status register, program, erase all block, and block erase. For details concerning each command, refer to "CHAPTER 1 Flash memory mode (CPU rewrite mode)".

#### (1) CPU rewrite mode beginning/release procedures

Operation procedure in the CPU rewrite mode for the built-in flash memory is described below. As for the control example, refer to "2.14.7 (2) Control example in the CPU rewrite mode."

##### [Beginning procedure]

- ① Apply 4.50 to 5.25 V to the CNV<sub>SS</sub>/V<sub>PP</sub> pin (at selecting boot ROM area).
- ② Release reset.
- ③ Set bits 6 and 7 (main clock division ratio selection bits) of the CPU mode register.
- ④ After CPU rewrite mode control program is transferred to internal RAM, jump to this control program on RAM. (The following operations are controlled by this control program).
- ⑤ Apply 4.50 to 5.25 to the CNV<sub>SS</sub>/V<sub>PP</sub> pin (in single-chip mode).
- ⑥ Set "1" to the CPU rewrite mode select bit (bit 1 of address 0FFE<sub>16</sub>).
- ⑦ Read the CPU rewrite mode entry flag (bit 2 of address 0FFE<sub>16</sub>) to confirm that the CPU rewrite mode is set to "1".
- ⑧ Flash memory operations are executed by using software commands.

**Note:** The following procedures are also necessary.

- Control for data which is input from the external (serial I/O etc.) and to be programmed to the flash memory.
- Initial setting for ports, etc.
- Writing to the watchdog timer

##### [Release procedure]

- ① Execute the read command or set the flash memory reset bit (bit 3 of address 0FFE<sub>16</sub>).
- ② Set the CPU rewrite mode select bit (bit 0 of address 0FFE<sub>16</sub>) to "0".

### 2.14.7 Flash memory mode application examples

The control pin processing example on the system board in the standard serial I/O mode and the control example in the CPU rewrite mode are described below.

#### (1) Control pin connection example on the system board in standard serial I/O mode

As shown in Figure 2.14.4, in the standard serial I/O mode, the built-in flash memory can be rewritten with the microcomputer mounted on board. Connection examples of control pins (P4<sub>0</sub>/E<sub>x</sub>DREQ/R<sub>x</sub>D, P4<sub>1</sub>/E<sub>x</sub>DACK/T<sub>x</sub>D, P4<sub>2</sub>/E<sub>x</sub>TC/S<sub>CLK</sub>, P4<sub>3</sub>/E<sub>x</sub>A1/ $\overline{S}$ <sub>RDY</sub>, P1<sub>6</sub>, CNV<sub>SS</sub>, and RESE $\overline{T}$  pin) in the standard serial I/O mode are described below.

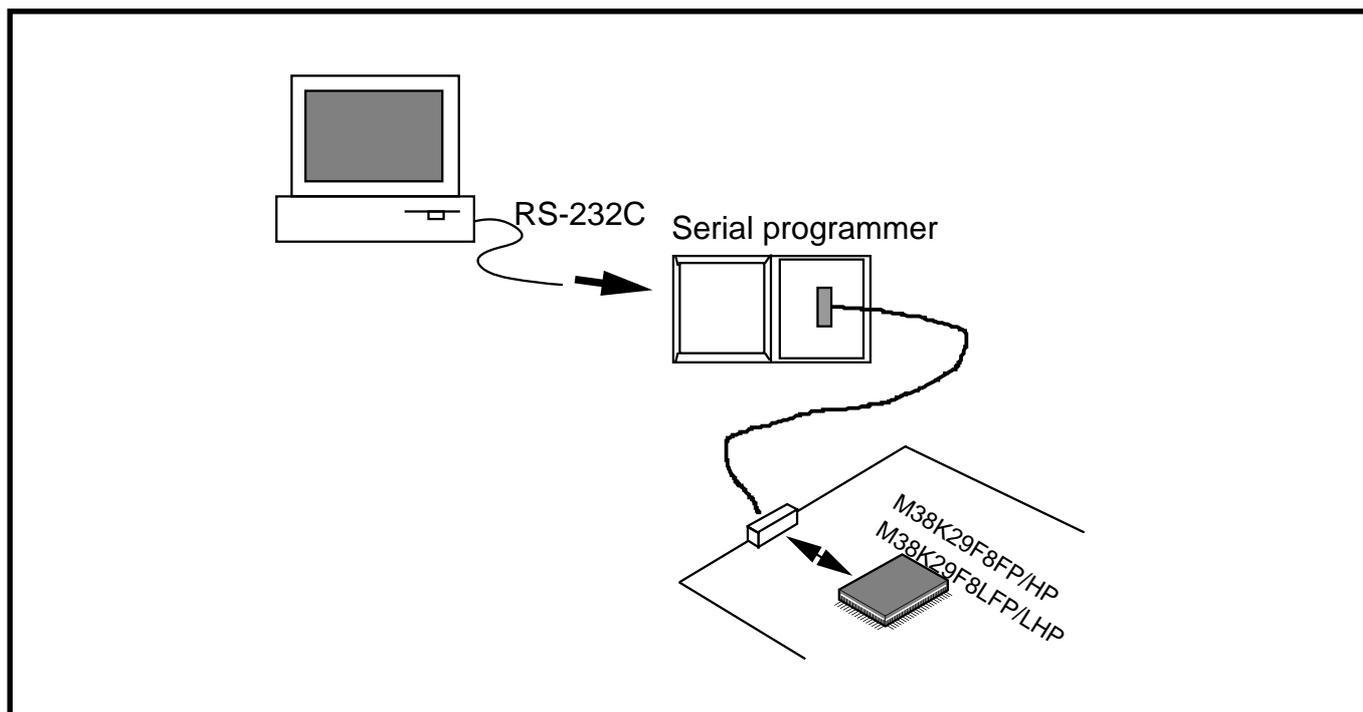


Fig. 2.14.4 Rewrite example of built-in flash memory in standard serial I/O mode

Table 2.14.3 shows the setting condition in the standard serial I/O mode.

Table 2.14.3 Setting condition in serial I/O mode

38K2 Group flash memory version		Value
Pin name	Pin number	
CNV <sub>SS</sub> /V <sub>PP</sub> ( <b>Note</b> )	7	4.50 to 5.25 V
P1 <sub>6</sub>	5	V <sub>CC</sub>
P4 <sub>2</sub> /E <sub>x</sub> TC/S <sub>CLK</sub>	53	V <sub>CC</sub>
RESE $\overline{T}$	8	Edge from V <sub>SS</sub> to V <sub>CC</sub>

**Note:** CNV<sub>SS</sub>/V<sub>PP</sub> is not V<sub>CC</sub> but a voltage when programming.

① When control signals are not affected to user system circuit

When the control signals in the standard serial I/O mode are not used or not affected to the user system circuit, they can be connected as shown in Figure 2.14.5.

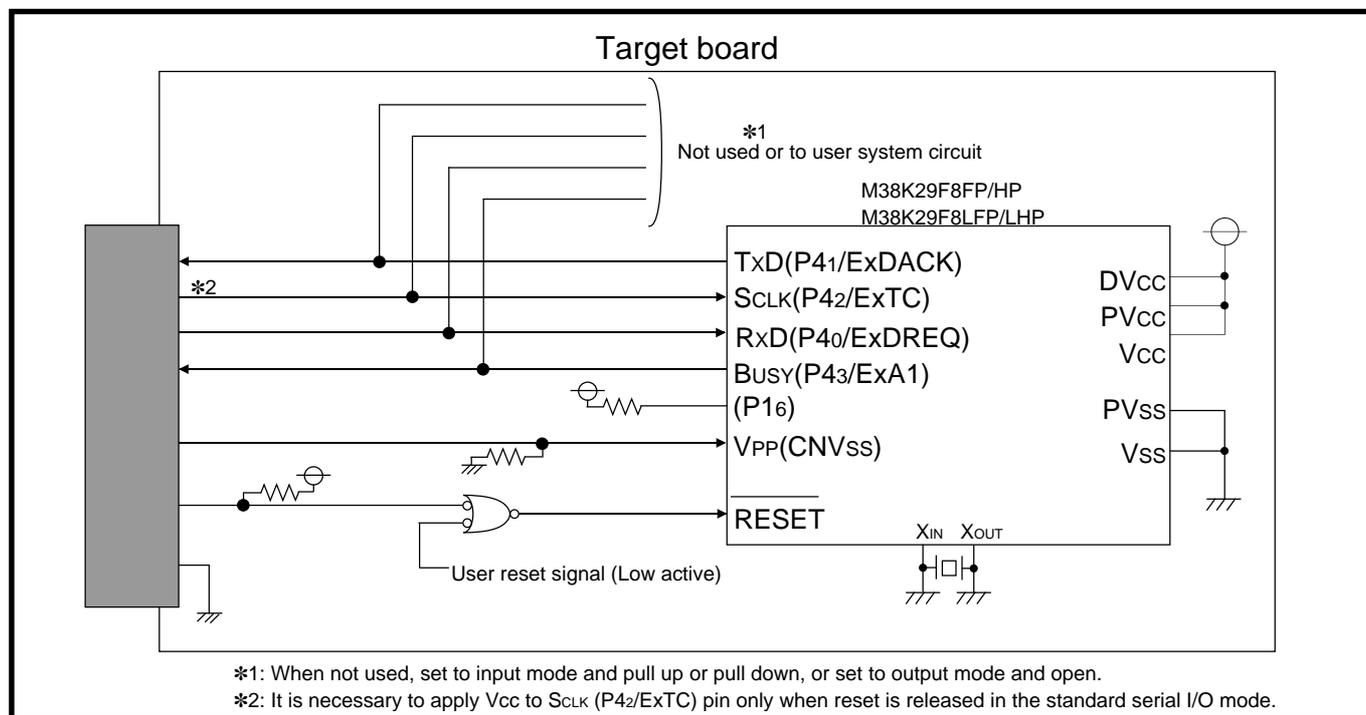


Fig. 2.14.5 Connection example in standard serial I/O mode (1)

② When control signals are affected to user system circuit-1

Figure 2.14.6 shows an example that the jumper switch cut-off the control signals not to supply to the user system circuit in the standard serial I/O mode.

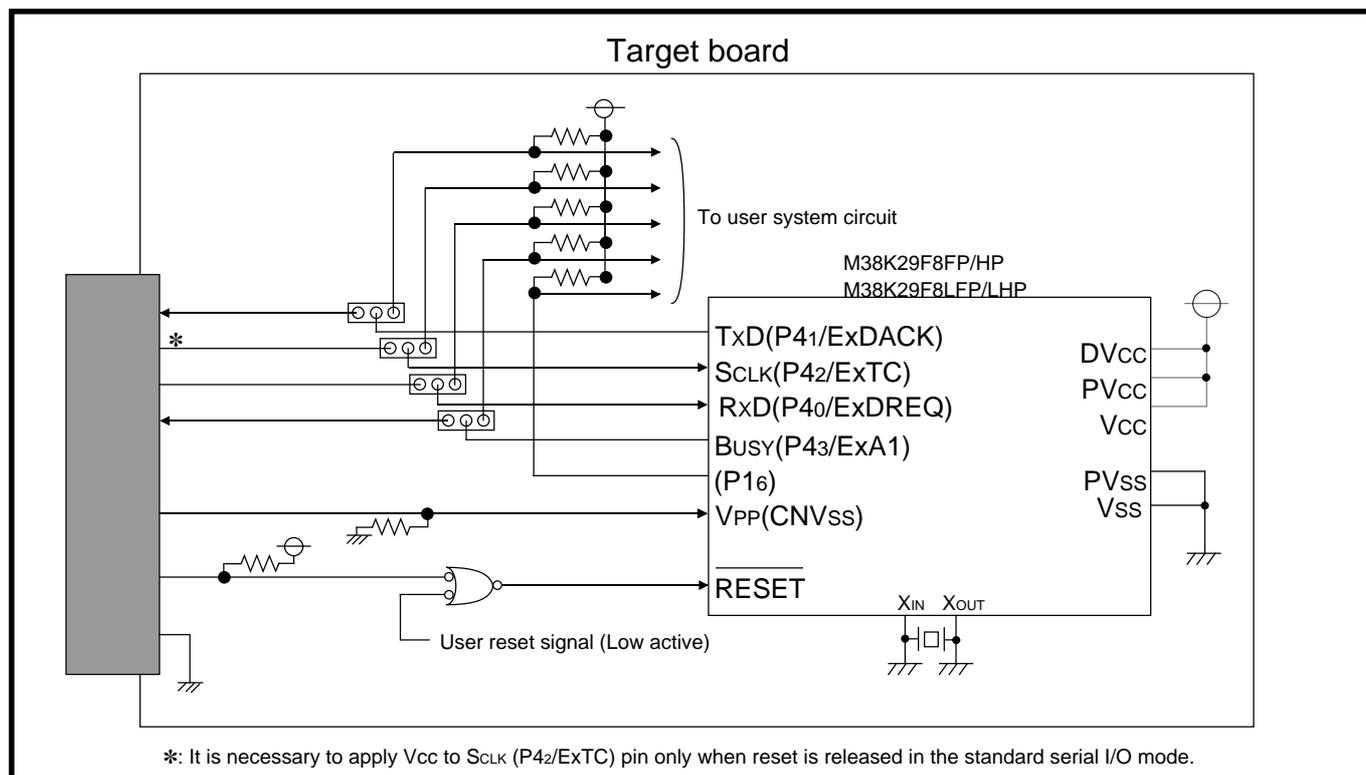


Fig. 2.14.6 Connection example in standard serial I/O mode (2)

③ When control signals are affected to user system circuit-2

Figure 2.14.7 shows an example that the analog switch (74HC4066) cut-off the control signals not to supply to the user system circuit in the standard serial I/O mode.

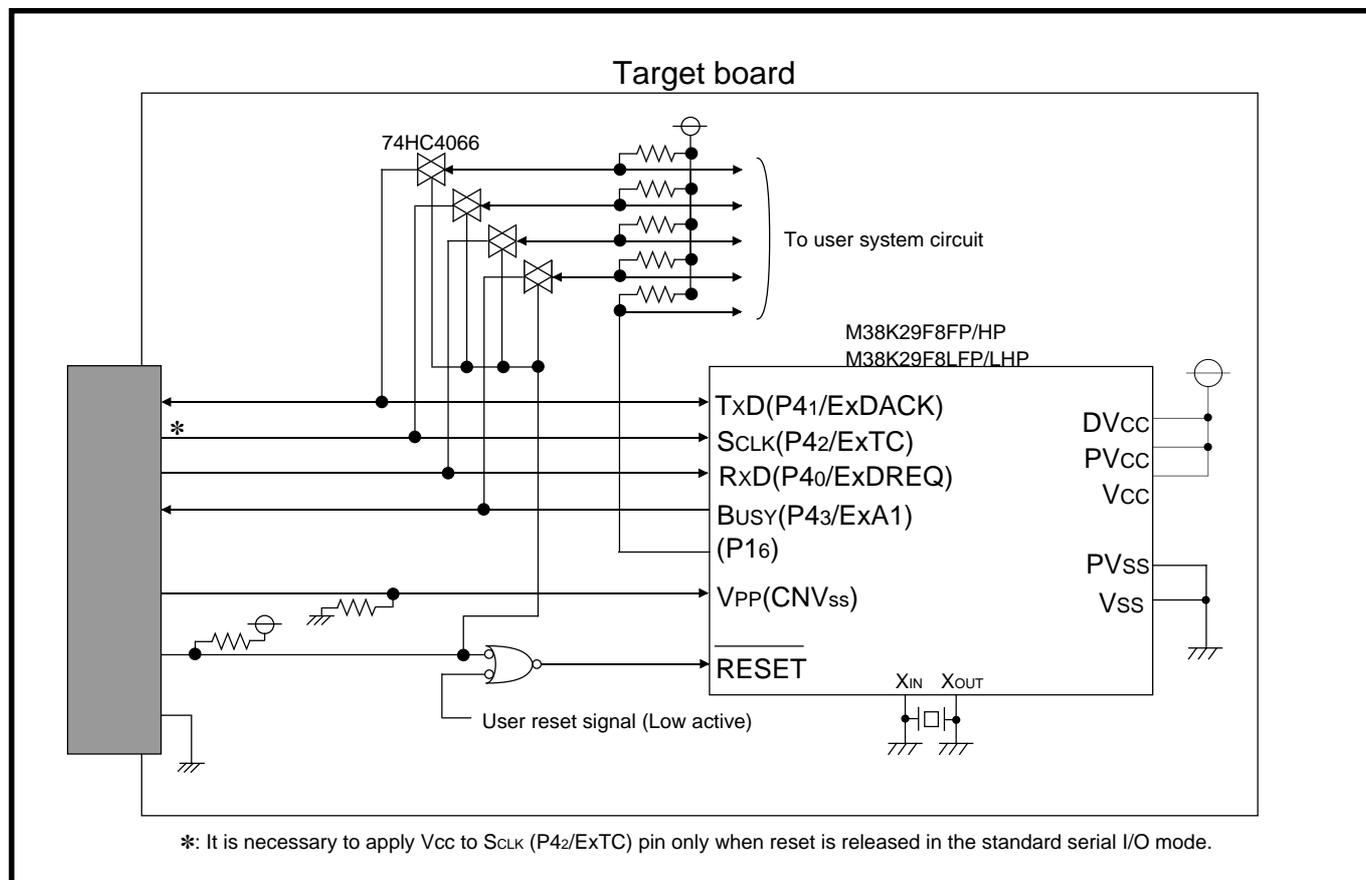


Fig. 2.14.7 Connection example in standard serial I/O mode (3)

**(2) Control example in CPU rewrite mode**

In this example, data is received by using serial I/O, and the data is programmed to the built-in flash memory in the CPU rewrite mode.

Figure 2.14.8 shows an example of the reprogramming system for the built-in flash memory in the CPU rewrite mode. Figure 2.14.9 shows the CPU rewrite mode beginning/release flowchart.

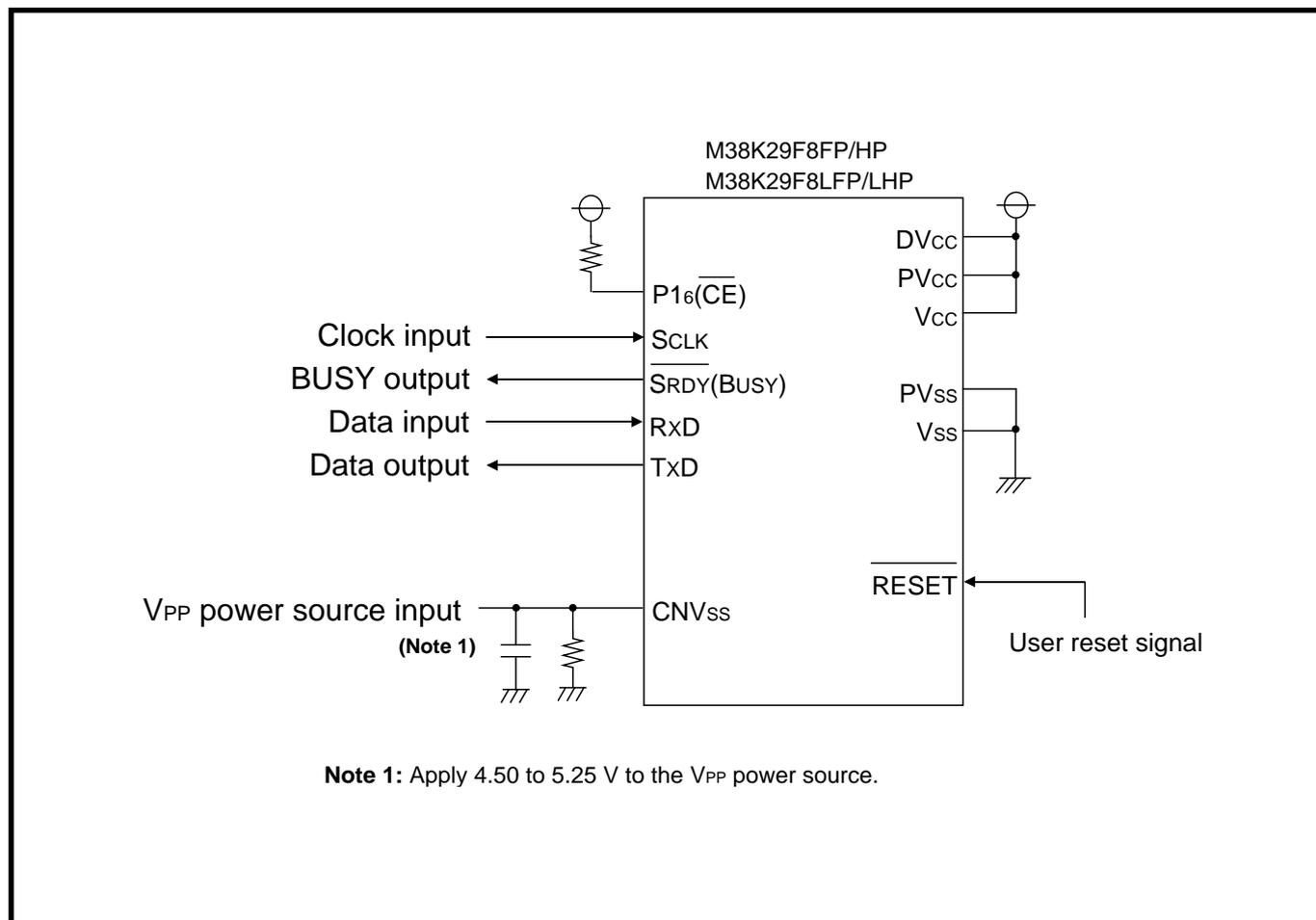


Fig. 2.14.8 Example of rewrite system for built-in flash memory in CPU rewrite mode

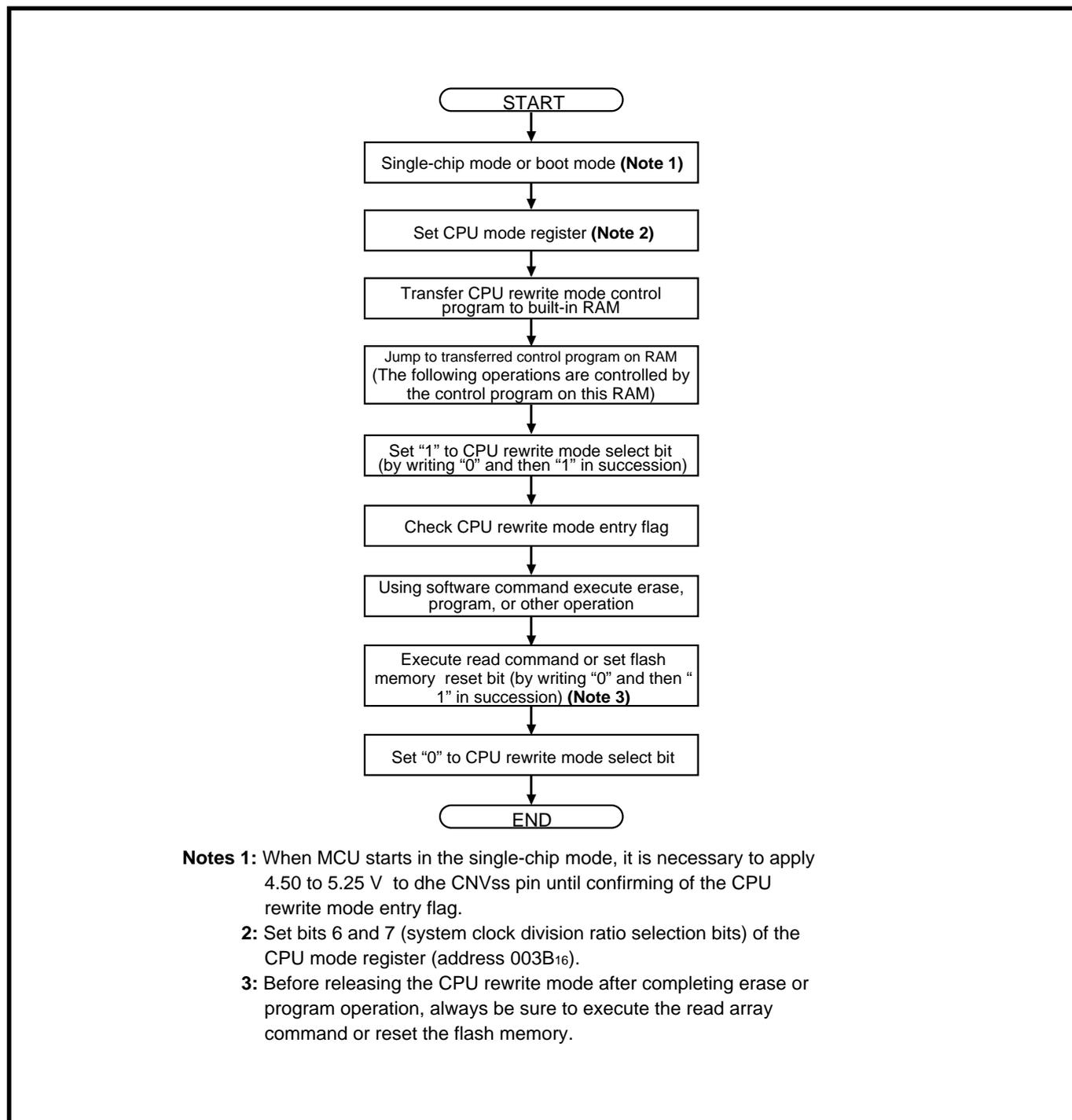


Fig. 2.14.9 CPU rewrite mode beginning/release flowchart

### 2.14.8 Notes on CPU rewrite mode

**(1) Operation speed**

During CPU rewrite mode, set the internal clock  $\phi$  1.5 MHz or less using the system clock division ratio selection bits (bits 6 and 7 of address 003B<sub>16</sub>).

**(2) Instructions inhibited against use**

The instructions which refer to the internal data of the flash memory cannot be used during the CPU rewrite mode .

**(3) Interrupts inhibited against use**

The interrupts cannot be used during the CPU rewrite mode because they refer to the internal data of the flash memory.

**(4) Watchdog timer**

In case of the watchdog timer has been running already, the internal reset generated by watchdog timer underflow does not happen, because of watchdog timer is always clearing during program or erase operation.

**(5) Reset**

Reset is always valid. In case of CNV<sub>SS</sub> = "H" when reset is released, boot mode is active. So the program starts from the address contained in address FFFC<sub>16</sub> and FFFD<sub>16</sub> in boot ROM area.

# CHAPTER 3

---

## APPENDIX

- 3.1 Electrical characteristics
- 3.2 Notes on use
- 3.3 Countermeasures against noise
- 3.4 List of registers
- 3.5 Package outline
- 3.6 List of instruction code
- 3.7 Machine instructions
- 3.8 SFR memory map
- 3.9 Pin configurations

## 3.1 Electrical characteristics

### 3.1.1 Absolute maximum ratings

Table 3.1.1 Absolute maximum ratings

Symbol	Parameter	Conditions	Ratings	Unit	
VCC	Power source voltage	All voltages are based on VSS. Output transistors are cut off.	-0.3 to 6.5	V	
AVCC	Analog power source voltage V <sub>CC</sub> E, V <sub>REF</sub> , PV <sub>CC</sub> , DV <sub>CC</sub> , USBV <sub>REF</sub>		-0.3 to V <sub>CC</sub> + 0.3	V	
V <sub>I</sub>	Input voltage P00-P07, P10-P17, P24-P27, P30-P37, P40-P43, P50-P57, P60-P63		-0.3 to V <sub>CC</sub> + 0.3	V	
V <sub>I</sub>	Input voltage RESET, X <sub>IN</sub> , CNV <sub>SS</sub> 2		-0.3 to V <sub>CC</sub> + 0.3	V	
V <sub>I</sub>	Input voltage CNV <sub>SS</sub>		Mask ROM version	-0.3 to V <sub>CC</sub> + 0.3	V
			Flash memory version	-0.3 to 6.5	V
V <sub>I</sub>	Input voltage D0+, D0-, D1+, D1-, D2+, D2-		-0.5 to 3.8	V	
V <sub>O</sub>	Output voltage P00-P07, P10-P17, P24-P27, P30-P37, P40-P43, P50-P57, P60-P63, X <sub>OUT</sub>	-0.3 to V <sub>CC</sub> + 0.3	V		
V <sub>O</sub>	Output voltage D0+, D0-, D1+, D1-, D2+, D2-, TrON	-0.5 to 3.8	V		
P <sub>d</sub>	Power dissipation (Note)	T <sub>a</sub> = 25°C	500	mW	
T <sub>opr</sub>	Operating temperature	MCU operating	-20 to 85	°C	
		In flash memory mode (For flash memory version)	25±5	°C	
T <sub>stg</sub>	Storage temperature		-40 to 125	°C	

**Note:** The maximum rating value depends on not only the MCU's power dissipation but the heat consumption characteristics of the package.

### 3.1.2 Recommended operating conditions (L.Ver)

**Table 3.1.2 Recommended operating conditions** ( $V_{CC} = 3.00$  to  $5.25$  V,  $V_{SS} = 0$  V,  $T_a = -20$  to  $85^\circ\text{C}$ , unless otherwise noted)

Symbol	Parameter		Limits			Unit	
			Min.	Typ.	Max.		
VCC	Power source voltage	VCC	System clock 12 MHz (2-/4-/8-divide mode)	4.00	5.00	5.25	V
			System clock 8 MHz	4.00	5.00	5.25	V
			System clock 6 MHz	3.00	5.00	5.25	V
AVCC	Analog power source voltage	PVCC, DVCC		VCC		V	
AVCC	Analog power source voltage	VCCE		VCC		V	
VREF	Analog reference voltage	VREF	2.0		VCC	V	
VREF	Analog reference voltage	USBVREF	$V_{CC} = 3.6$ to $4.0$ V	3.0		3.6	V
			$V_{CC} = 3.0$ to $3.6$ V	3.0		VCC	V
VSS	Power source voltage	VSS		0		V	
AVSS	Analog power source voltage	PVSS		0		V	
VIH	"H" input voltage	P00–P07, P24–P27, P50–P57, P60–P63	$0.8V_{CC}$		VCC	V V	
VIH	"H" input voltage	P10–P17, P30–P37, P40–P43	$0.8V_{CC_E}$		VCC_E	V	
VIH	"H" input voltage	RESET, XIN, CNVSS, CNVSS2	$0.8V_{CC}$		VCC	V	
VIH	"H" input voltage	D0+, D0-, D1+, D1-, D2+, D2-	2.0		3.6	V	
VIL	"L" input voltage	P00–P07, P24–P27, P50–P57, P60–P63	0		$0.2V_{CC}$	V	
VIL	"L" input voltage	P10–P17, P30–P37, P40–P43	0		$0.2V_{CC_E}$	V	
VIL	"L" input voltage	RESET, XIN, CNVSS, CNVSS2	0		$0.2V_{CC}$	V	
VIL	"L" input voltage	D0+, D0-, D1+, D1-, D2+, D2-	0		0.8	V	

**Table 3.1.3 Recommended operating conditions** ( $V_{CC} = 3.00$  to  $5.25$  V,  $V_{SS} = 0$  V,  $T_a = -20$  to  $85^\circ\text{C}$ , unless otherwise noted)

Symbol	Parameter		Limits			Unit
			Min.	Typ.	Max.	
$\Sigma I_{OH(peak)}$	"H" total peak output current ( <b>Note 1</b> )	P00–P07, P24–P27, P50–P57, P60–P63			–80	mA
$\Sigma I_{OH(peak)}$	"H" total peak output current ( <b>Note 1</b> )	P10–P17, P30–P37, P40–P43			–80	mA
$\Sigma I_{OL(peak)}$	"L" total peak output current ( <b>Note 1</b> )	P00–P07, P24–P27, P50–P57			80	mA
$\Sigma I_{OL(peak)}$	"L" total peak output current ( <b>Note 1</b> )	P60–P63			80	mA
$\Sigma I_{OL(peak)}$	"L" total peak output current ( <b>Note 1</b> )	P10–P17, P30–P37, P40–P43			80	mA
$\Sigma I_{OH(avg)}$	"H" total average output current ( <b>Note 1</b> )	P00–P07, P24–P27, P50–P57, P60–P63			–40	mA
$\Sigma I_{OH(avg)}$	"H" total average output current ( <b>Note 1</b> )	P10–P17, P30–P37, P40–P43			–40	mA
$\Sigma I_{OL(avg)}$	"L" total average output current ( <b>Note 1</b> )	P00–P07, P24–P27, P50–P57			40	mA
$\Sigma I_{OL(avg)}$	"L" total average output current ( <b>Note 1</b> )	P60–P63			40	mA
$\Sigma I_{OL(avg)}$	"L" total average output current ( <b>Note 1</b> )	P10–P17, P30–P37, P40–P43			40	mA
$I_{OH(peak)}$	"H" peak output current ( <b>Note 2</b> )	P00–P07, P24–P27, P50–P57, P60–P63			–10	mA
$I_{OH(peak)}$	"H" peak output current ( <b>Note 2</b> )	P10–P17, P30–P37, P40–P43			–10	mA
$I_{OL(peak)}$	"L" peak output current ( <b>Note 2</b> )	P00–P07, P24–P27, P50–P57			10	mA
$I_{OL(peak)}$	"L" peak output current ( <b>Note 2</b> )	P60–P63			20	mA
$I_{OL(peak)}$	"L" peak output current ( <b>Note 2</b> )	P10–P17, P30–P37, P40–P43			10	mA
$I_{OH(avg)}$	"H" average output current ( <b>Note 3</b> )	P00–P07, P24–P27, P50–P57, P60–P63			–5	mA
$I_{OH(avg)}$	"H" average output current ( <b>Note 3</b> )	P10–P17, P30–P37, P40–P43			–5	mA
$I_{OL(avg)}$	"L" average output current ( <b>Note 3</b> )	P00–P07, P24–P27, P50–P57			5	mA
$I_{OL(avg)}$	"L" average output current ( <b>Note 3</b> )	P60–P63			10	mA
$I_{OL(avg)}$	"L" average output current ( <b>Note 3</b> )	P10–P17, P30–P37, P40–P43			5	mA
$f(XIN)$	Main clock input oscillation frequency ( <b>Note 4</b> )	$V_{CC} = 4.00$ to $5.25$ V	6		12	MHz
		$V_{CC} = 3.00$ to $4.00$ V	6		6	MHz
$f(XIN)$ or $f(SYN)$	System clock frequency	$V_{CC} = 4.00$ to $5.25$ V	6		12	MHz
		$V_{CC} = 3.00$ to $4.00$ V	6		6	MHz
$f(\phi)$	$\phi$ frequency	$V_{CC} = 4.00$ to $5.25$ V			8	MHz
		$V_{CC} = 3.00$ to $4.00$ V			6	MHz

**Notes 1:** The total peak output current is the absolute value of the peak currents flowing through all the applicable ports. The total average output current is the average value of the absolute value of the currents measured over 100 ms flowing through all the applicable ports.

**2:** The peak output current is the absolute value of the peak current flowing in each port.

**3:** The average output current is the average value of the absolute value of the currents measured over 100 ms.

**4:** The duty of oscillation frequency is 50 %. 6 MHz or 12 MHz is usable.

## 3.1.3 Electrical characteristics (L.Ver)

Table 3.1.4 Electrical characteristics (1) (V<sub>CC</sub> = 3.00 to 5.25 V, V<sub>SS</sub> = 0 V, T<sub>a</sub> = -20 to 85°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
V <sub>OH</sub>	"H" output voltage P00-P07, P24-P27, P50-P57, P60-P63	I <sub>OH</sub> = -10 mA (V <sub>CC</sub> = 4.00 to 5.25 V)	V <sub>CC</sub> -2.0			V
		I <sub>OH</sub> = -1 mA	V <sub>CC</sub> -1.0			V
V <sub>OH</sub>	"H" output voltage P10-P17, P30-P37, P40-P43	I <sub>OH</sub> = -10 mA (V <sub>CC</sub> E = 4.00 to 5.25 V)	V <sub>CC</sub> E-2.0			V
		I <sub>OH</sub> = -1 mA	V <sub>CC</sub> E-1.0			V
V <sub>OH</sub>	"H" output voltage D0+, D0-, D1+, D1-, D2+, D2-	D+ and D- pins pull- down with 0 V via a resistor of 15 kΩ ± 5 %	2.8		3.6	V
V <sub>OL</sub>	"L" output voltage P00-P07, P24-P27, P50-P57	I <sub>OL</sub> = 10 mA (V <sub>CC</sub> = 4.00 to 5.25 V)			2.0	V
		I <sub>OL</sub> = 1 mA			1.0	V
V <sub>OL</sub>	"L" output voltage P60-P63	I <sub>OL</sub> = 20 mA (V <sub>CC</sub> = 4.00 to 5.25 V)			2.0	V
		I <sub>OL</sub> = 1 mA			1.0	V
V <sub>OL</sub>	"L" output voltage P10-P17, P30-P37, P40-P43	I <sub>OL</sub> = 10 mA (V <sub>CC</sub> E = 4.00 to 5.25 V)			2.0	V
		I <sub>OL</sub> = 1 mA (V <sub>CC</sub> E = 3.00 to 5.25 V)			1.0	V
V <sub>OL</sub>	"L" output voltage D0+, D0-, D1+, D1-, D2+, D2-	D+ and D- pins pull-up with 3.6 V via a resistor of 1.5 kΩ ± 5 %	0		0.3	V
V <sub>T+</sub> -V <sub>T-</sub>	Hysteresis CNTR0, INT0, INT1			0.6		V
V <sub>T+</sub> -V <sub>T-</sub>	Hysteresis P10/DQ0-P17/DQ7, P30-P32, P33/ExINT, P34/ExCS, P35/ExWR, P36/ExRD, P37/ ExA0, P40/ExDREQ/RxD, P41/ExDACK/ TxD, P42/ExTC/SCLK, P43/ExA1/SRDY			0.6		V
V <sub>T+</sub> -V <sub>T</sub>	Hysteresis D0+, D0-, D1+, D1-, D2+, D2-			0.25		V
V <sub>T+</sub> -V <sub>T-</sub>	Hysteresis $\overline{\text{RESET}}$			0.5		V
I <sub>IH</sub>	"H" input current P00-P07, P24-P27, P50-P57, P60-P63	V <sub>I</sub> = V <sub>CC</sub> (Pull-ups "off")			5.0	μA
I <sub>IH</sub>	"H" input current P10-P17, P30-P37, P40-P43	V <sub>I</sub> = V <sub>CC</sub> E			5.0	μA
I <sub>IH</sub>	"H" input current $\overline{\text{RESET}}$ , CNV <sub>SS</sub>	V <sub>I</sub> = V <sub>CC</sub>			5.0	μA
I <sub>IH</sub>	"H" input current X <sub>IN</sub>	V <sub>I</sub> = V <sub>CC</sub>		4.0		μA
I <sub>IL</sub>	"L" input current P00-P07, P24-P27, P50-P57, P60-P63	V <sub>I</sub> = V <sub>SS</sub> (Pull-ups "off")			-5.0	μA
I <sub>IL</sub>	"L" input current P10-P17, P30-P37, P40-P43	V <sub>I</sub> = V <sub>SS</sub>			-5.0	μA
I <sub>IL</sub>	"L" input current $\overline{\text{RESET}}$ , CNV <sub>SS</sub> , CNV <sub>SS</sub> 2	V <sub>I</sub> = V <sub>SS</sub>			-5.0	μA
I <sub>IL</sub>	"L" input current X <sub>IN</sub>	V <sub>I</sub> = V <sub>SS</sub>		-4.0		μA
I <sub>IL</sub>	"L" input current P00-P07, P50, P52 (Pull-ups "on")	V <sub>I</sub> = V <sub>SS</sub> (V <sub>CC</sub> = 4.00 to 5.25 V)	-20.0	-60.0	-120.0	μA
		V <sub>I</sub> = V <sub>SS</sub>	-10.0			μA
V <sub>RAM</sub>	RAM hold voltage	When clock is stopped	2.00		5.25	V

**Table 3.1.5 Electrical characteristics (2)** (V<sub>CC</sub> = 3.00 to 5.25 V, V<sub>SS</sub> = 0 V, T<sub>a</sub> = -20 to 85°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit			
			Min.	Typ.	Max.				
I <sub>CC</sub>	Power source current (Output transistor is isolated.)	Normal mode (Note 1)	V <sub>CC</sub> = 4.00 to 5.25 V	f(X <sub>IN</sub> ) = system clock = 12 MHz, φ = 6 MHz, USB reference voltage circuit enabled		21.0	60	mA	
				f(X <sub>IN</sub> ) = 12 MHz, System clock = φ = 8 MHz, USB reference voltage circuit enabled		22.5	60	mA	
				f(X <sub>IN</sub> ) = 6 MHz, System clock = φ = 8 MHz, USB reference voltage circuit enabled		22.0	60	mA	
				f(X <sub>IN</sub> ) = system clock = φ = 6 MHz, USB reference voltage circuit enabled		21.0	60	mA	
				V <sub>CC</sub> = 3.00 to 4.00 V	f(X <sub>IN</sub> ) = system clock = φ = 6 MHz, USB reference voltage circuit disabled			35	mA
				V <sub>CC</sub> = 3.00 to 3.60 V	f(X <sub>IN</sub> ) = system clock = φ = 6 MHz, USB reference voltage circuit disabled		9.0	30	mA
		Wait mode (Note 2)	V <sub>CC</sub> = 4.00 to 5.25 V	f(X <sub>IN</sub> ) = 12 MHz, System clock = φ = 8 MHz, USB reference voltage circuit enabled		6.0		mA	
				V <sub>CC</sub> = 3.00 to 4.00 V	f(X <sub>IN</sub> ) = system clock = φ = 6 MHz, USB reference voltage circuit disabled		2.0		mA
		Stop mode (Note 3)	V <sub>CC</sub> = 4.00 to 5.25 V	USB reference voltage circuit enabled Low current mode		125.0	250	μA	
				V <sub>CC</sub> = 3.00 to 5.25 V	USB reference voltage circuit disabled T <sub>a</sub> = 25 °C		0.1		μA
USB reference voltage circuit disabled T <sub>a</sub> = 85 °C						10	μA		

**<Test conditions>****Notes 1:** Operating in single-chip modeClock input from X<sub>IN</sub> pin (X<sub>OUT</sub> oscillator stopped)f<sub>USB</sub> = 48 MHz

All USB difference-input circuits enabled

Leaving I/O pins open

Operating functions: PLL circuit, CPU, Timers

**2:** Operating in single-chip mode with Wait modeClock input from X<sub>IN</sub> pin (X<sub>OUT</sub> oscillator stopped)f<sub>USB</sub> = 48 MHz

All USB difference-input circuits enabled

Leaving I/O pins open

Operating functions: PLL circuit, Timers, USB receiving

Disabled functions: CPU

**3:** Operating in single-chip mode with Stop mode

Oscillation stopped

All USB difference-input circuits disabled

Leaving I/O pins open

### 3.1.4 A/D converter characteristics (L.Ver)

**Table 3.1.6 A/D Converter characteristics** ( $V_{CC} = 3.00$  to  $5.25$  V,  $V_{SS} = 0$  V,  $T_a = -20$  to  $85^\circ\text{C}$ , unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
—	Resolution				10	Bits
—	Linearity error	$T_a = 25^\circ\text{C}$			$\pm 3$	LSB
—	Differential nonlinear error	$T_a = 25^\circ\text{C}$			$\pm 1.5$	LSB
VOT	Zero transition voltage	$V_{CC} = V_{REF} = 5.12$ V	0	15	35	mV
VFST	Full scale transition voltage	$V_{CC} = V_{REF} = 5.12$ V	5105	5125	5150	mV
tCONV	Conversion time				122	tc(XIN) or tc(fsYN)
RLADDER	Ladder resistor			35		k $\Omega$
I <sub>VREF</sub>	Reference power source input current	A/D converter operating; $V_{REF} = 5.0$ V	50	150	200	$\mu\text{A}$
		A/D converter not operating; $V_{REF} = 5.0$ V			5	
I <sub>I(AD)</sub>	A/D port input current				5.0	$\mu\text{A}$

### 3.1.5 Timing Requirements (L.Ver)

**Table 3.1.7 Timing requirements (1)** ( $V_{CC} = 4.00$  to  $5.25$  V,  $V_{SS} = 0$  V,  $T_a = -20$  to  $85$  °C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_w(\overline{\text{RESET}})$	Reset input "L" pulse width	2			$\mu\text{s}$
$t_c(\text{XIN})$	Main clock input cycle time	83			ns
$t_{WH}(\text{XIN})$	Main clock input "H" pulse width	35			ns
$t_{WL}(\text{XIN})$	Main clock input "L" pulse width	35			ns
$t_c(\text{CNTR})$	CNTR0 input cycle time	200			ns
$t_{WH}(\text{CNTR})$	CNTR0 input "H" pulse width	80			ns
$t_{WL}(\text{CNTR})$	CNTR0 input "L" pulse width	80			ns
$t_{WH}(\text{INT})$	INT0, INT1 input "H" pulse width	80			ns
$t_{WL}(\text{INT})$	INT0, INT1 input "L" pulse width	80			ns
$t_c(\text{SCLK})$	Serial I/O clock input cycle time ( <b>Note</b> )	800			ns
$t_{WH}(\text{SCLK})$	Serial I/O clock input "H" pulse width ( <b>Note</b> )	370			ns
$t_{WL}(\text{SCLK})$	Serial I/O clock input "L" pulse width ( <b>Note</b> )	370			ns
$t_{su}(\text{Rx}D\text{--SCLK})$	Serial I/O input set up time	220			ns
$t_h(\text{SCLK--Rx}D)$	Serial I/O input hold time	100			ns

**Note:** These limits are the rating values in the clock synchronous mode, bit 6 of address 0FE016 = "1". In the UART mode, bit 6 of address 0FE016 = "0"; the rating values are set to one fourth.

**Table 3.1.8 Timing requirements (2)** ( $V_{CC} = 3.00$  to  $4.00$  V,  $V_{SS} = 0$  V,  $T_a = -20$  to  $85$  °C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_w(\overline{\text{RESET}})$	Reset input "L" pulse width	2			$\mu\text{s}$
$t_c(\text{XIN})$	Main clock input cycle time	166			ns
$t_{WH}(\text{XIN})$	Main clock input "H" pulse width	70			ns
$t_{WL}(\text{XIN})$	Main clock input "L" pulse width	70			ns
$t_c(\text{CNTR})$	CNTR0 input cycle time	500			ns
$t_{WH}(\text{CNTR})$	CNTR0 input "H" pulse width	230			ns
$t_{WL}(\text{CNTR})$	CNTR0 input "L" pulse width	230			ns
$t_{WH}(\text{INT})$	INT0, INT1 input "H" pulse width	230			ns
$t_{WL}(\text{INT})$	INT0, INT1 input "L" pulse width	230			ns
$t_c(\text{SCLK})$	Serial I/O clock input cycle time ( <b>Note</b> )	2000			ns
$t_{WH}(\text{SCLK})$	Serial I/O clock input "H" pulse width ( <b>Note</b> )	950			ns
$t_{WL}(\text{SCLK})$	Serial I/O clock input "L" pulse width ( <b>Note</b> )	950			ns
$t_{su}(\text{Rx}D\text{--SCLK})$	Serial I/O input set up time	400			ns
$t_h(\text{SCLK--Rx}D)$	Serial I/O input hold time	200			ns

**Note:** These limits are the rating values in the clock synchronous mode, bit 6 of address 0FE016 = "1". In the UART mode, bit 6 of address 0FE016 = "0"; the rating values are set to one fourth.

**Table 3.1.9 Timing requirements of external bus interface (EXB) (1)**(V<sub>CC</sub> = 4.00 to 5.25 V, V<sub>SS</sub> = 0 V, T<sub>a</sub> = -20 to 85 °C, unless otherwise noted)

Symbol	Parameter		Limits			Unit
			Min.	Typ.	Max.	
t <sub>su</sub> (S-R)	ExCS setup time for read		0			ns
t <sub>su</sub> (S-W)	ExCS setup time for write		0			ns
t <sub>h</sub> (R-S)	ExCS hold time for read		0			ns
t <sub>h</sub> (W-S)	ExCS hold time for write		0			ns
t <sub>su</sub> (A-R)	ExA0, ExA1 setup time for read		10			ns
t <sub>su</sub> (A-W)	ExA0, ExA1 setup time for write		10			ns
t <sub>h</sub> (R-A)	ExA0, ExA1 hold time for read		0			ns
t <sub>h</sub> (W-A)	ExA0, ExA1 hold time for write		0			ns
t <sub>su</sub> (ACK-R)	ExDACK setup time for read		10			ns
t <sub>su</sub> (ACK-W)	ExDACK setup time for write		10			ns
t <sub>h</sub> (R-ACK)	ExDACK hold time for read		0			ns
t <sub>h</sub> (W-ACK)	ExDACK hold time for write		0			ns
t <sub>WH</sub> (R)	Read "H" pulse width		80			ns
t <sub>WL</sub> (R)	Read "L" pulse width		80			ns
t <sub>WH</sub> (W)	Write "H" pulse width		80			ns
t <sub>WL</sub> (W)	Write "L" pulse width		80			ns
t <sub>WH</sub> (ACK)	ExDACK "H" pulse width		120			ns
t <sub>WL</sub> (ACK)	ExDACK "L" pulse width		120			ns
t <sub>su</sub> (D-W)	Data input setup time before write		40			ns
t <sub>h</sub> (W-D)	Data input hold time after write		0			ns
t <sub>su</sub> (D-ACK)	Data input setup time before ExDACK		60			ns
t <sub>h</sub> (ACK-W)	Data input hold time after ExDACK		5			ns
t <sub>C</sub> ( $\phi$ )	CPU clock cycle time		125			ns
t <sub>w</sub> (cycle)	Burst mode access cycle time	USB function not operating	t <sub>C</sub> ( $\phi$ )*3+10			ns
		USB function operating	t <sub>C</sub> ( $\phi$ )*5+10			ns

**Table 3.1.10 Timing requirements of external bus interface (EXB) (2)**(V<sub>CC</sub> = 3.00 to 4.00 V, V<sub>SS</sub> = 0 V, T<sub>a</sub> = -20 to 85 °C, unless otherwise noted)

Symbol	Parameter		Limits			Unit
			Min.	Typ.	Max.	
t <sub>su</sub> (S-R)	ExCS setup time for read		0			ns
t <sub>su</sub> (S-W)	ExCS setup time for write		0			ns
t <sub>h</sub> (R-S)	ExCS hold time for read		0			ns
t <sub>h</sub> (W-S)	ExCS hold time for write		0			ns
t <sub>su</sub> (A-R)	ExA0, ExA1 setup time for read		30			ns
t <sub>su</sub> (A-W)	ExA0, ExA1 setup time for write		30			ns
t <sub>h</sub> (R-A)	ExA0, ExA1 hold time for read		0			ns
t <sub>h</sub> (W-A)	ExA0, ExA1 hold time for write		0			ns
t <sub>su</sub> (ACK-R)	ExDACK setup time for read		30			ns
t <sub>su</sub> (ACK-W)	ExDACK setup time for write		30			ns
t <sub>h</sub> (R-ACK)	ExDACK hold time for read		0			ns
t <sub>h</sub> (W-ACK)	ExDACK hold time for write		0			ns
t <sub>WH</sub> (R)	Read "H" pulse width		120			ns
t <sub>WL</sub> (R)	Read "L" pulse width		120			ns
t <sub>WH</sub> (W)	Write "H" pulse width		120			ns
t <sub>WL</sub> (W)	Write "L" pulse width		120			ns
t <sub>WH</sub> (ACK)	ExDACK "H" pulse width		160			ns
t <sub>WL</sub> (ACK)	ExDACK "L" pulse width		160			ns
t <sub>su</sub> (D-W)	Data input setup time before write		60			ns
t <sub>h</sub> (W-D)	Data input hold time after write		0			ns
t <sub>su</sub> (D-ACK)	Data input setup time before ExDACK		80			ns
t <sub>h</sub> (ACK-W)	Data input hold time after ExDACK		10			ns
t <sub>c</sub> ( $\phi$ )	CPU clock cycle time		166			ns
t <sub>w</sub> (cycle)	Burst mode access cycle time	USB function not operating	t <sub>c</sub> ( $\phi$ )*3+30			ns
		USB function operating	t <sub>c</sub> ( $\phi$ )*5+30			ns

### 3.1.6 Switching Characteristics (L.Ver)

**Table 3.1.11 Switching characteristics (1)** ( $V_{CC} = 4.00$  to  $5.25$  V,  $V_{SS} = 0$  V,  $T_a = -20$  to  $85$  °C, unless otherwise noted)

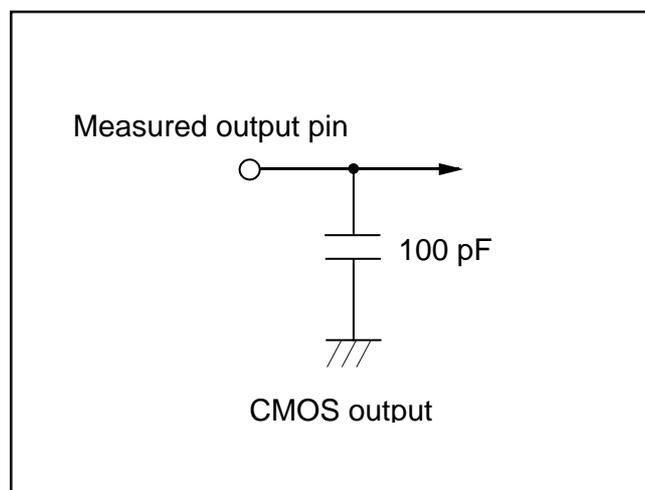
Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t <sub>WH</sub> (SCLK)	Serial I/O clock output "H" pulse width	t <sub>c</sub> (SCLK)/2–30			ns
t <sub>WL</sub> (SCLK)	Serial I/O clock output "L" pulse width	t <sub>c</sub> (SCLK)/2–30			ns
t <sub>d</sub> (SCLK–TxD)	Serial I/O output delay time			140	ns
t <sub>v</sub> (SCLK–TxD)	Serial I/O output valid time	–30			ns
t <sub>r</sub> (SCLK)	Serial I/O clock output rising time			30	ns
t <sub>f</sub> (SCLK)	Serial I/O clock output falling time			30	ns
t <sub>r</sub> (CMOS)	CMOS output rising time ( <b>Note</b> )			30	ns
t <sub>f</sub> (CMOS)	CMOS output falling time ( <b>Note</b> )			30	ns

**Notes:** Pins XOUT, D0+, D0-, D1+, D2-, D2+, D2- are excluded.

**Table 3.1.12 Switching characteristics (2)** ( $V_{CC} = 3.00$  to  $4.00$  V,  $V_{SS} = 0$  V,  $T_a = -20$  to  $85$  °C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t <sub>WH</sub> (SCLK)	Serial I/O clock output "H" pulse width	t <sub>c</sub> (SCLK)/2–50			ns
t <sub>WL</sub> (SCLK)	Serial I/O clock output "L" pulse width	t <sub>c</sub> (SCLK)/2–50			ns
t <sub>d</sub> (SCLK–TxD)	Serial I/O output delay time			350	ns
t <sub>v</sub> (SCLK–TxD)	Serial I/O output valid time	–30			ns
t <sub>r</sub> (SCLK)	Serial I/O clock output rising time			50	ns
t <sub>f</sub> (SCLK)	Serial I/O clock output falling time			50	ns
t <sub>r</sub> (CMOS)	CMOS output rising time ( <b>Note</b> )			50	ns
t <sub>f</sub> (CMOS)	CMOS output falling time ( <b>Note</b> )			50	ns

**Notes:** Pins XOUT, D0+, D0-, D1+, D2-, D2+, D2- are excluded.



**Fig. 3.1.1 Output switching characteristics measurement circuit**

**Table 3.1.13 Switching characteristics of external bus interface (EXB) (1)**

(VCC = 4.00 to 5.25 V, VSS = 0 V, Ta = -20 to 85 °C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
ta(R-D)	Data output enable time after read			60	ns
tv(R-D)	Data output disable time after read	0			ns
ta(ACK-D)	Data output enable time after ExDACK			80	ns
tv(ACK-D)	Data output disable time after ExDACK	0			ns
td(R-Mdis)	In cycle mode Mch_req disable output delay time after read			tC(φ)+10	ns
td(W-Mdis)	In cycle mode Mch_req disable output delay time after write			tC(φ)+10	ns
td(R-Men)	In cycle mode Mch_req enable output delay time after read	USB function not operating		tC(φ)*3+10	ns
		USB function operating		tC(φ)*5+10	ns
td(W-Men)	In cycle mode Mch_req enable output delay time after write	USB function not operating		tC(φ)*3+10	ns
		USB function operating		tC(φ)*5+10	ns

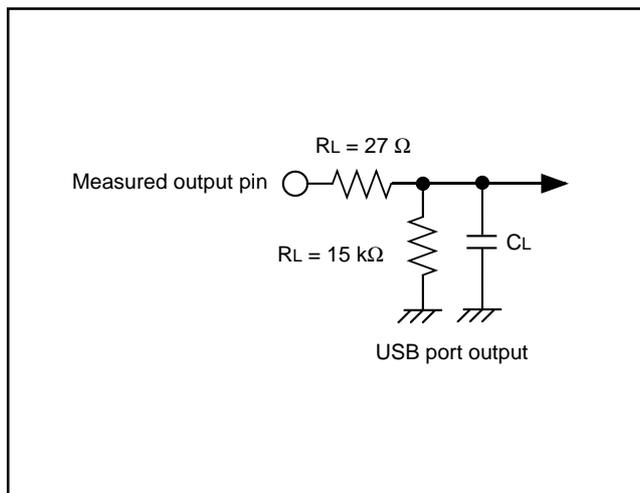
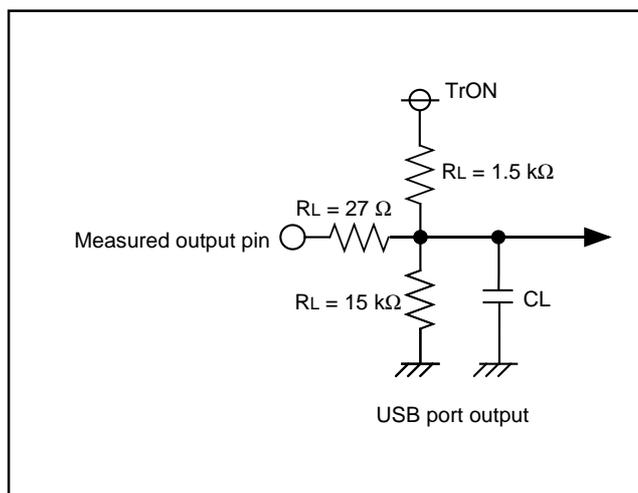
**Table 3.1.14 Switching characteristics of external bus interface (EXB) (2)**

(VCC = 3.00 to 4.00 V, VSS = 0 V, Ta = -20 to 85 °C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
ta(R-D)	Data output enable time after read			80	ns
tv(R-D)	Data output disable time after read	0			ns
ta(ACK-D)	Data output enable time after ExDACK			120	ns
tv(ACK-D)	Data output disable time after ExDACK	0			ns
td(R-Mdis)	In cycle mode Mch_req disable output delay time after read			tC(φ)+30	ns
td(W-Mdis)	In cycle mode Mch_req disable output delay time after write			tC(φ)+30	ns
td(R-Men)	In cycle mode Mch_req enable output delay time after read	USB function not operating		tC(φ)*3+30	ns
		USB function operating		tC(φ)*5+30	ns
td(W-Men)	In cycle mode Mch_req enable output delay time after write	USB function not operating		tC(φ)*3+30	ns
		USB function operating		tC(φ)*5+30	ns

**Table 3.1.15 Switching characteristics (USB ports)** ( $V_{CC} = 3.00$  to  $5.25$  V,  $V_{SS} = 0$  V,  $T_a = -20$  to  $85$  °C, unless otherwise noted)

Symbol	Parameter		Limits			Unit
			Min.	Typ.	Max.	
$t_{tr}(D+/D-)$	USB full-speed output rising time	$CL = 50$ pF	4		20	ns
$t_{ff}(D+/D-)$	USB full-speed output falling time	$CL = 50$ pF	4		20	ns
$t_{lr}(D+/D-)$	USB low-speed output rising time	$CL = 200$ to $600$ pF $T_a = 0$ to $85$ °C	75		300	ns
		$CL = 250$ to $600$ pF $T_a = -20$ to $85$ °C	75		300	ns
		$CL = 200$ to $600$ pF $T_a = -20$ to $85$ °C	65		300	ns
$t_{fr}(D+/D-)$	USB low-speed output falling time	$CL = 200$ to $600$ pF $T_a = 0$ to $85$ °C	75		300	ns
		$CL = 250$ to $600$ pF $T_a = -20$ to $85$ °C	75		300	ns
		$CL = 200$ to $600$ pF $T_a = -20$ to $85$ °C	65		300	ns
$t_{frm}(D+/D-)$	USB full-speed ports rising/falling ratio	$t_{tr}(D+/D-)/t_{ff}(D+/D-)$	90		111.11	%
$t_{frm}(D+/D-)$	USB low-speed ports rising/falling ratio	$t_{tr}(D+/D-)/t_{ff}(D+/D-)$	80		125	%
$V_{crs}(D+/D-)$	USB output signal cross-over voltage		1.3		2.0	V

**Fig. 3.1.2** USB output switching characteristics measurement circuit (1) for D0-, D1+/D2+ (low-speed), D1-/D2- (full-speed)**Fig. 3.1.3** USB output switching characteristics measurement circuit (2) for D0+, D1+/D2+ (full-speed), D1-/D2- (low-speed)

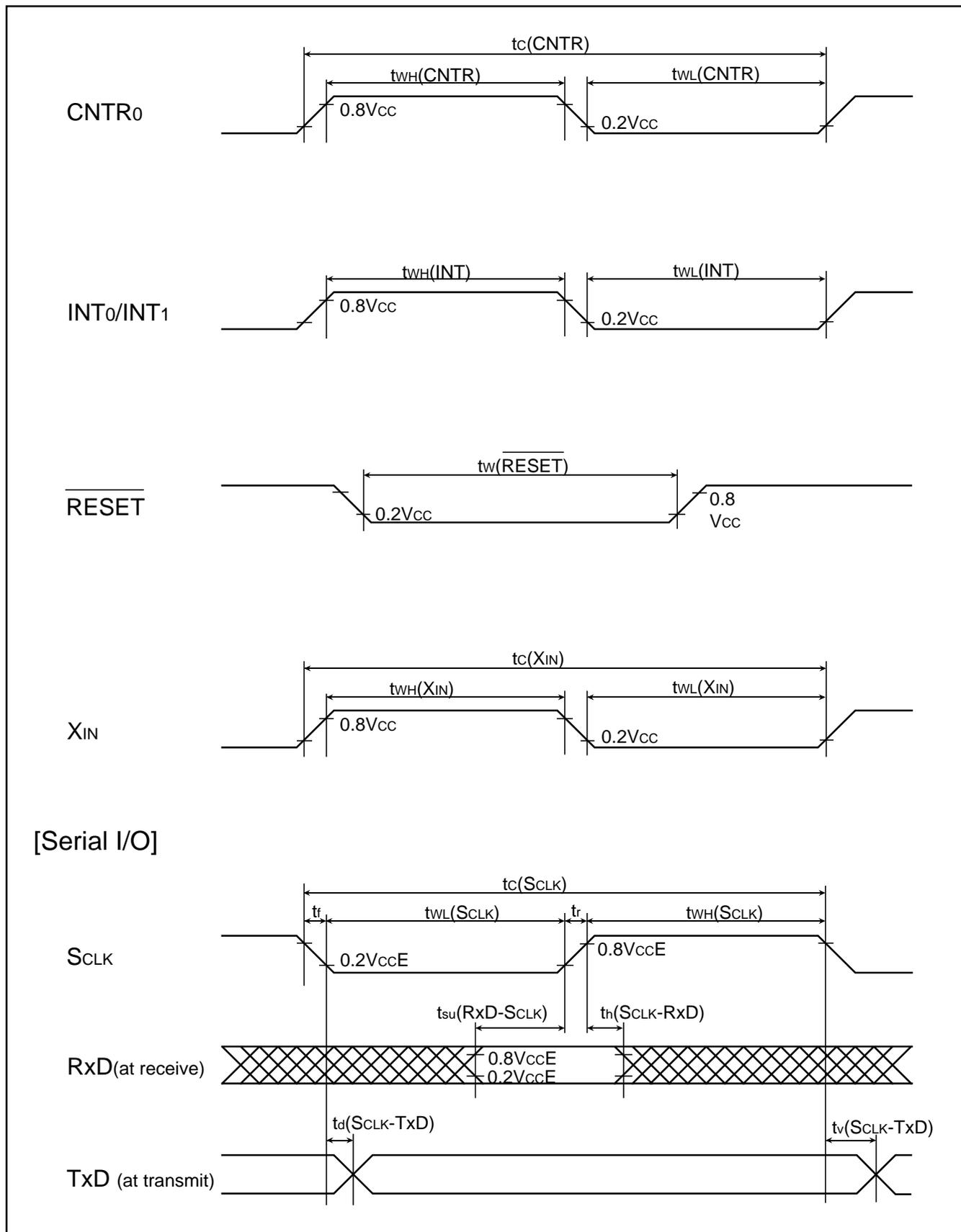
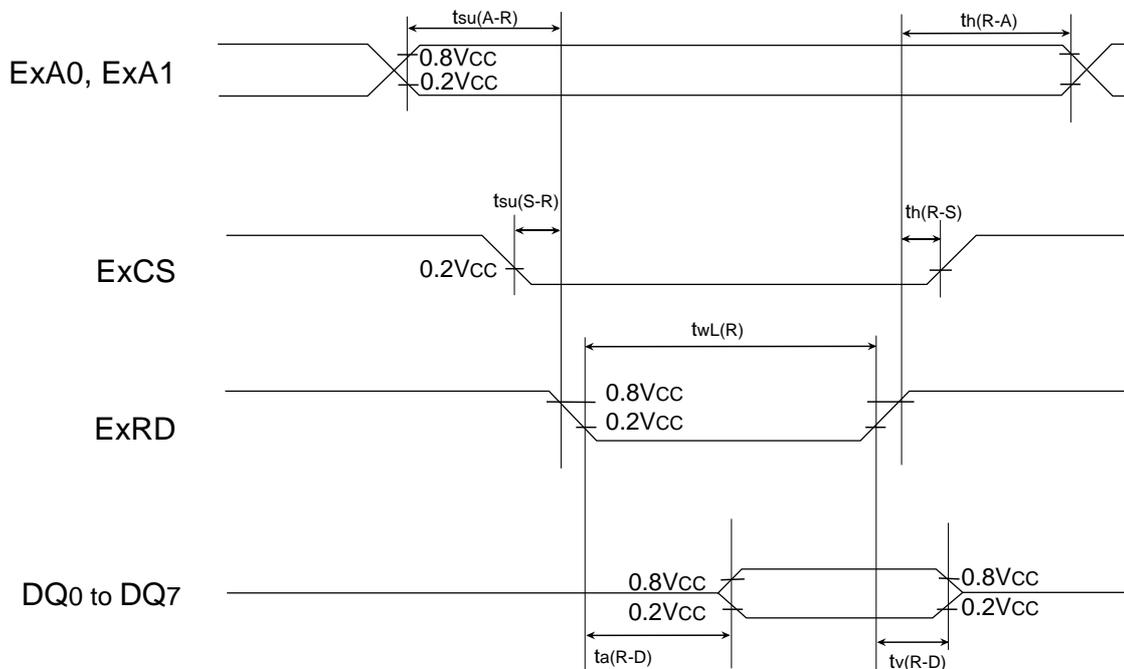


Fig. 3.1.4 Timing chart (1)

● Timing chart

[ EXB <CPU channel mode> ]

< Read >



< Write >

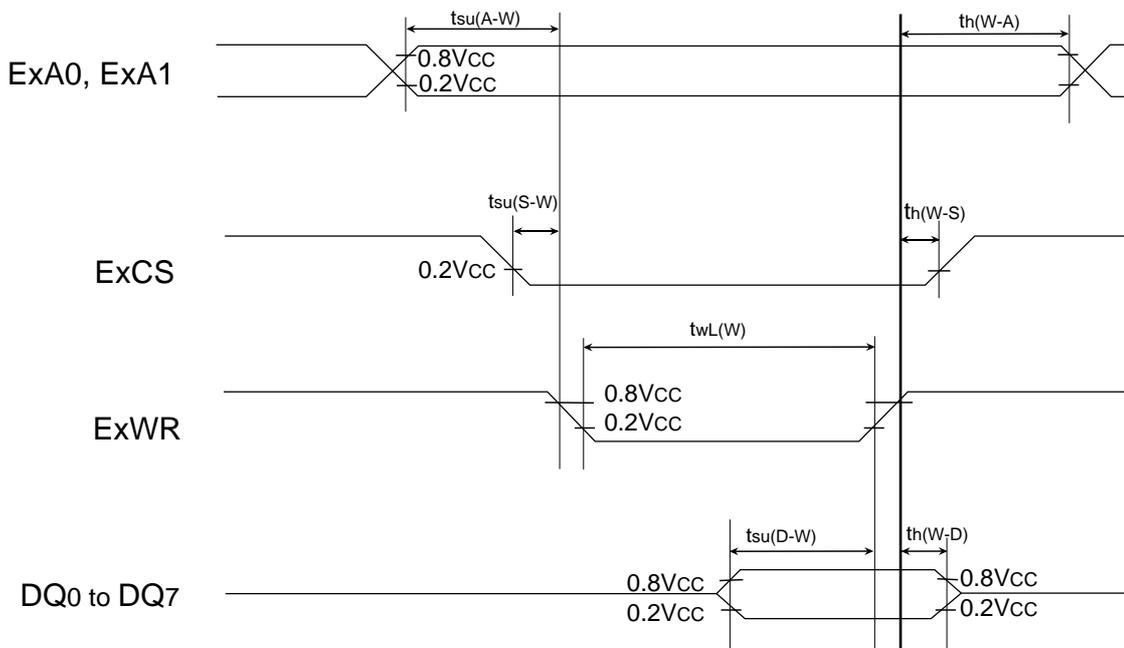
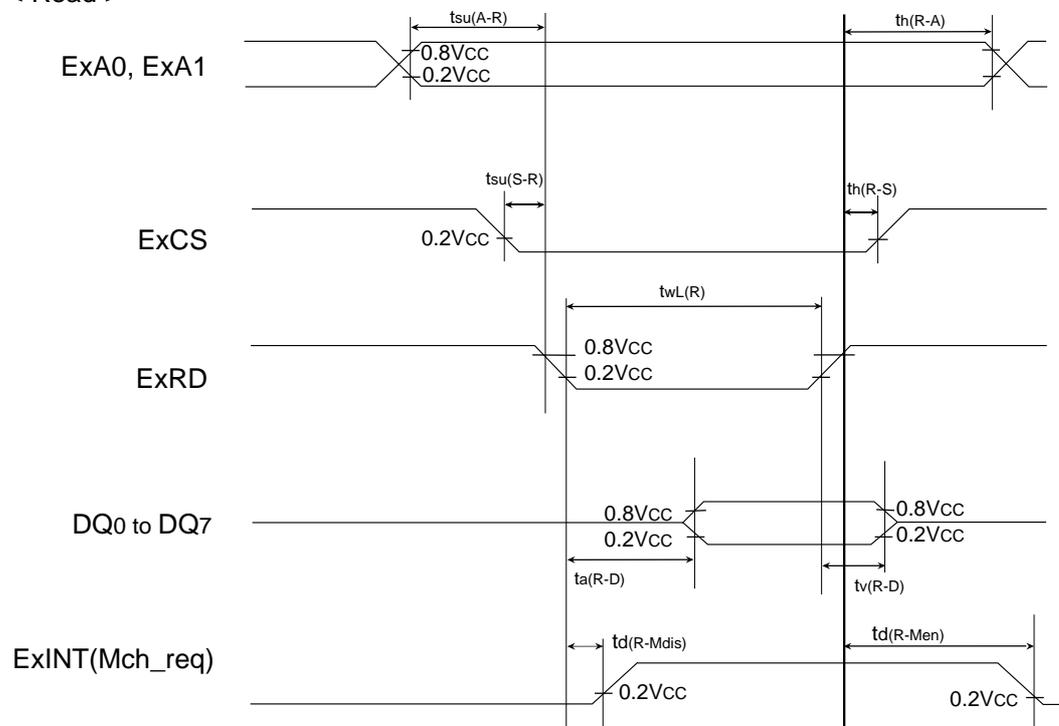


Fig. 3.1.5 Timing chart (2)

● Timing chart

[ EXB <Memory channel mode, Normal port function> ]

< Read >



< Write >

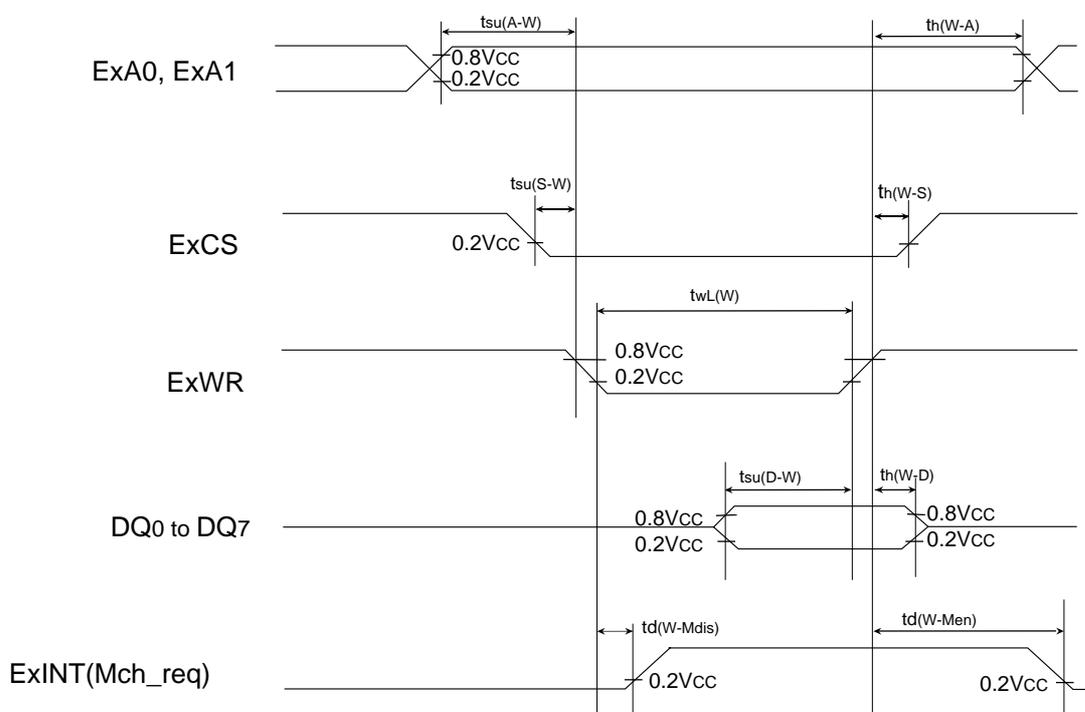
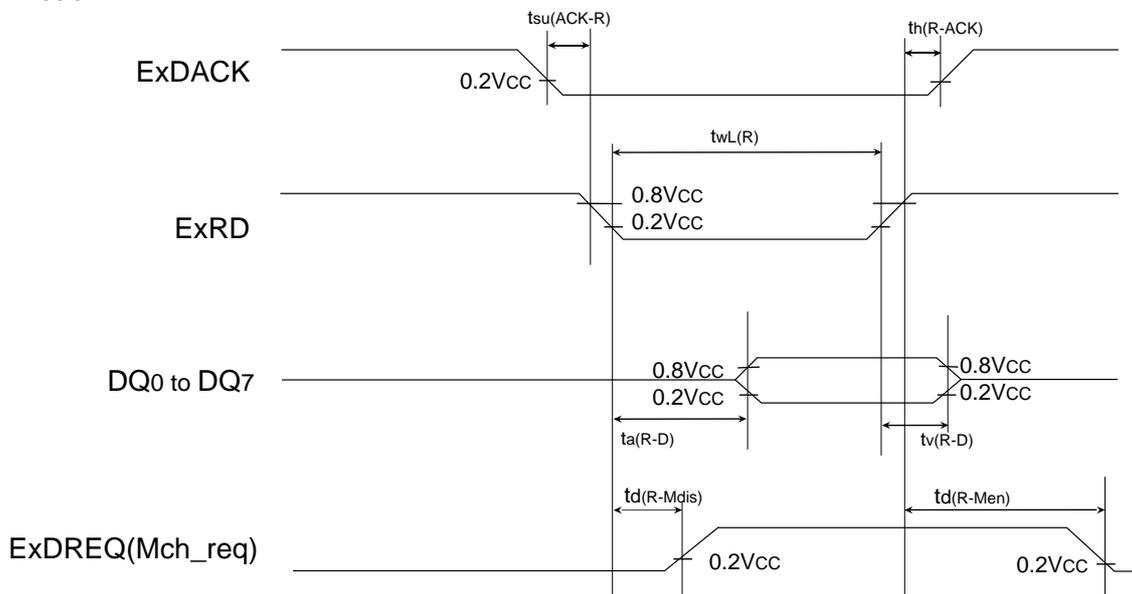


Fig. 3.1.6 Timing chart (3)

● Timing chart

[ EXB <Memory channel mode, DMA interface pin function, Read and write signals used together mode> ]

< Read >



< Write >

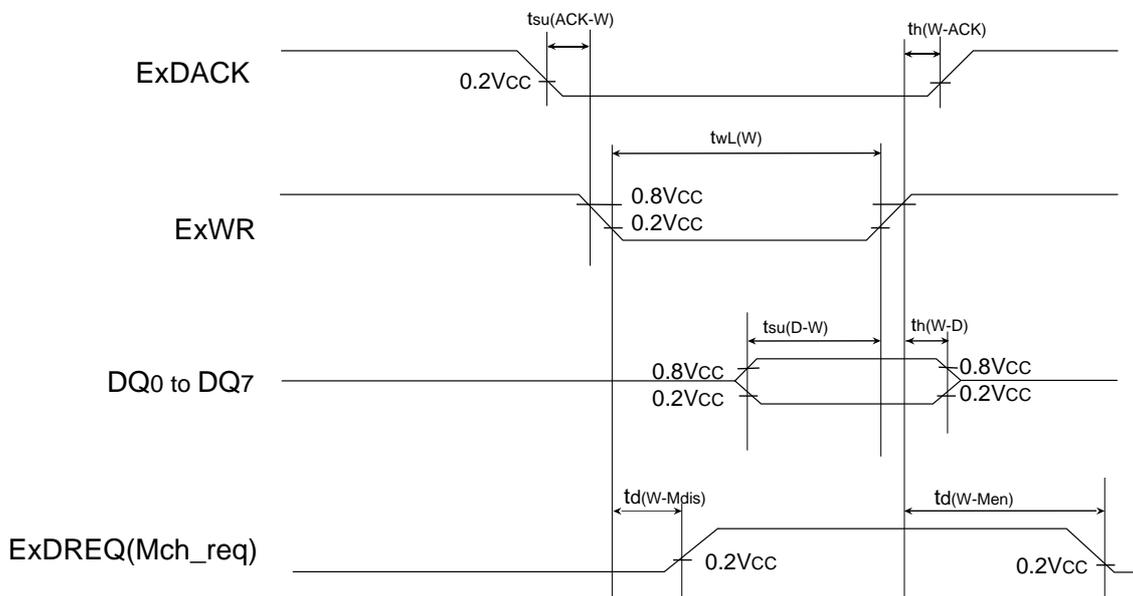
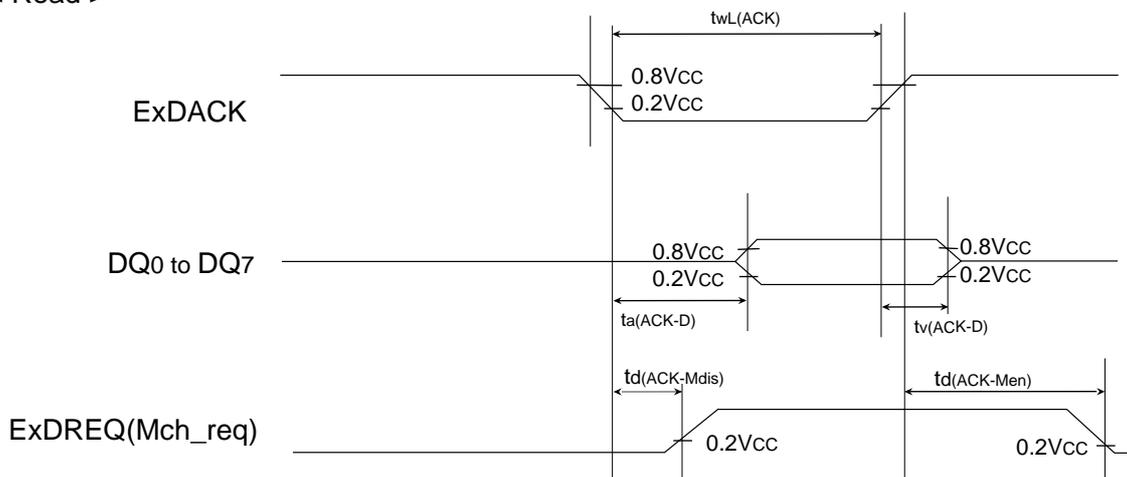


Fig. 3.1.7 Timing chart (4)

● Timing chart

[ EXB <Memory channel mode, DMA interface pin function, Read and write signals not required mode> ]

< Read >



< Write >

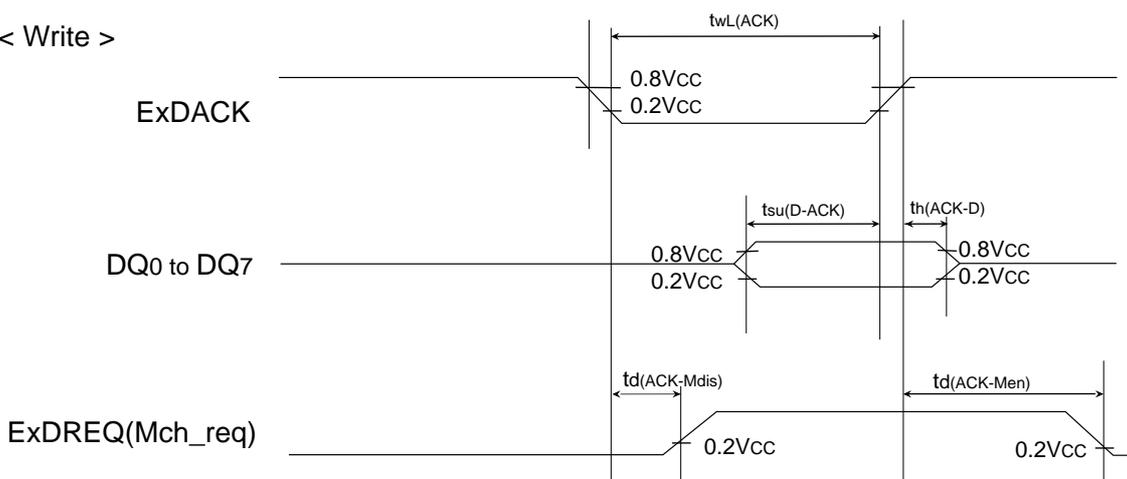
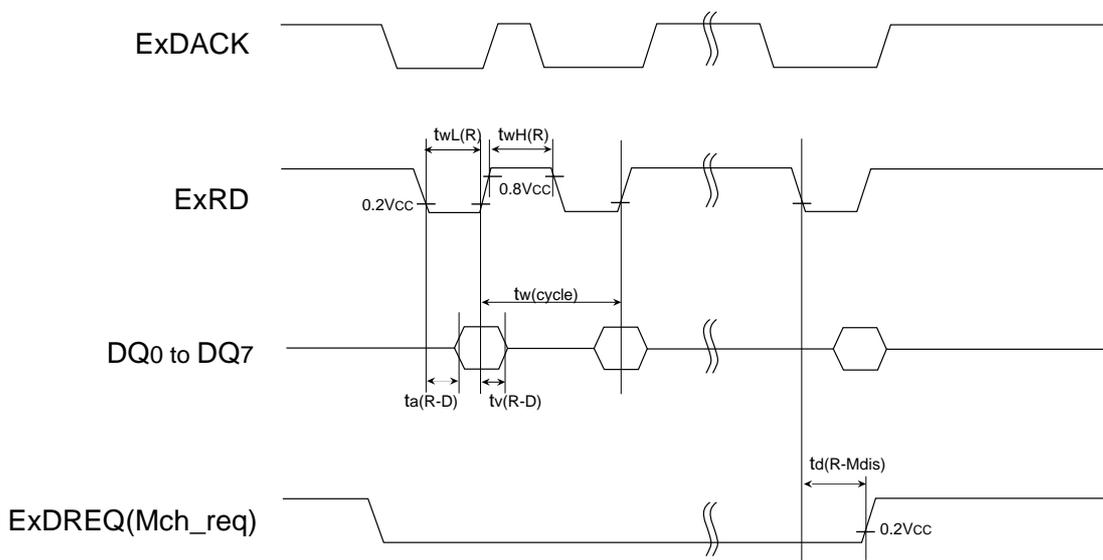


Fig. 3.1.8 Timing chart (5)

● Timing chart

[ EXB <Memory channel mode, Burst transfer> ]

< Read >



< Write >

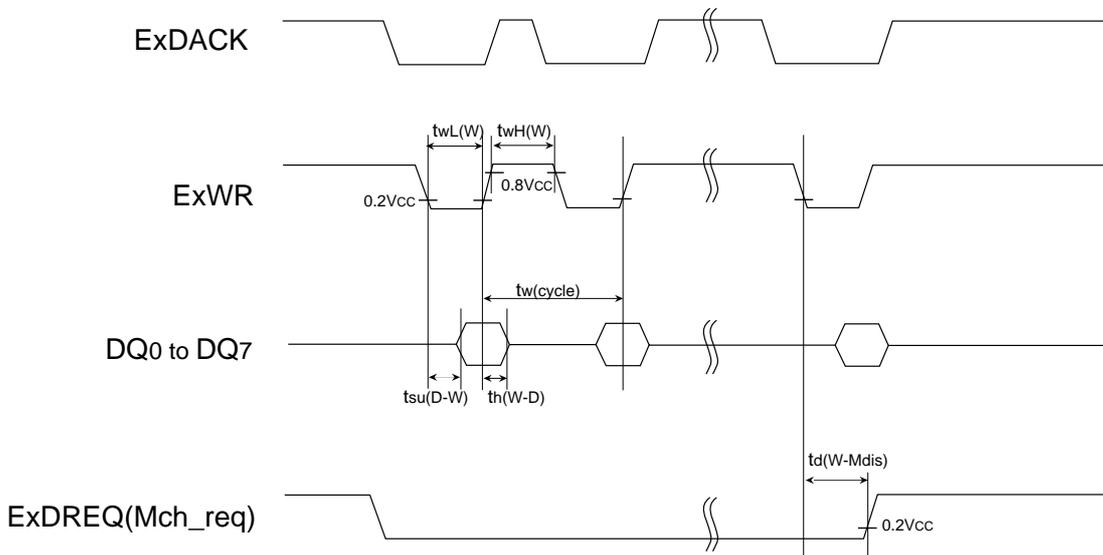


Fig. 3.1.9 Timing chart (6)

## 3.2 Notes on use

### 3.2.1 Notes on input and output ports

#### (1) Modifying output data with bit managing instruction

When the port latch of an I/O port is modified with the bit managing instruction\*<sup>1</sup>, the value of the unspecified bit may be changed.

##### ● Reason

The bit managing instructions are read-modify-write form instructions for reading and writing data by a byte unit. Accordingly, when these instructions are executed on a bit of the port latch of an I/O port, the following is executed to all bits of the port latch.

- As for bit which is set for input port:

The pin state is read in the CPU, and is written to this bit after bit managing.

- As for bit which is set for output port:

The bit value is read in the CPU, and is written to this bit after bit managing.

Note the following:

- Even when a port which is set as an output port is changed for an input port, its port latch holds the output data.
- As for a bit of which is set for an input port, its value may be changed even when not specified with a bit managing instruction in case where the pin state differs from its port latch contents.

\*<sup>1</sup> Bit managing instructions: **SEB** and **CLB** instructions

### 3.2.2 Termination of unused pins

#### (1) Terminate unused pins

① I/O ports :

- Set the I/O ports for the input mode and connect them to VCC or VSS through each resistor of 1 k $\Omega$  to 10 k $\Omega$ .  
Set the I/O ports for the output mode and open them at "L" or "H".
- When opening them in the output mode, the input mode of the initial status remains until the mode of the ports is switched over to the output mode by the program after reset. Thus, the potential at these pins is undefined and the power source current may increase in the input mode. With regard to an effects on the system, thoroughly perform system evaluation on the user side.
- Since the direction register setup may be changed because of a program runaway or noise, set direction registers by program periodically to increase the reliability of program.

#### (2) Termination remarks

① I/O ports :

Do not open in the input mode.

● Reason

- The power source current may increase depending on the first-stage circuit.
- An effect due to noise may be easily produced as compared with proper termination shown on the above.

② I/O ports :

When setting for the input mode, do not connect to VCC or VSS directly.

● Reason

If the direction register setup changes for the output mode because of a program runaway or noise, a short circuit may occur between a port and VCC (or VSS).

③ I/O ports :

When setting for the input mode, do not connect multiple ports in a lump to VCC or VSS through a resistor.

● Reason

If the direction register setup changes for the output mode because of a program runaway or noise, a short circuit may occur between ports.

- At the termination of unused pins, perform wiring at the shortest possible distance (20 mm or less) from microcomputer pins.

### 3.2.3 Notes on interrupts

#### (1) Change of relevant register settings

When the setting of the following registers or bits is changed, the interrupt request bit may be set to "1". When not requiring the interrupt occurrence synchronized with these setting, take the following sequence.

- Interrupt edge selection register (address  $0FF3_{16}$ )
- Timer X mode register (address  $23_{16}$ )

Set the above listed registers or bits as the following sequence.

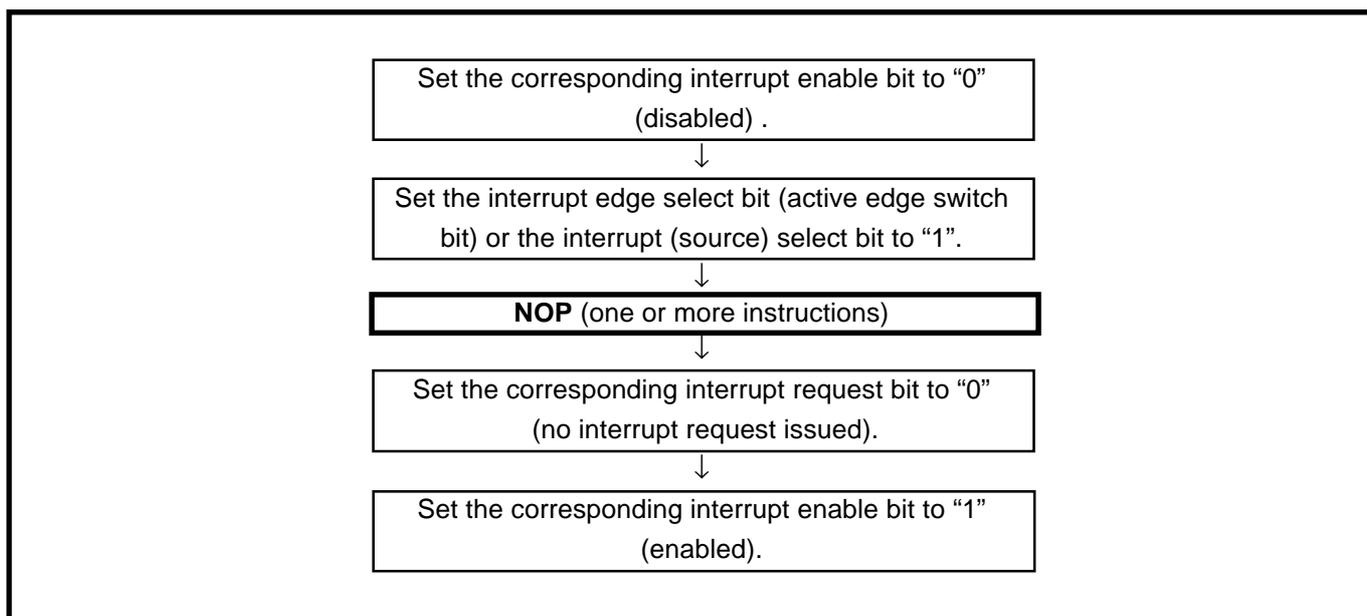


Fig. 3.2.1 Sequence of changing relevant register

#### ■ Reason

When setting the following, the interrupt request bit may be set to "1".

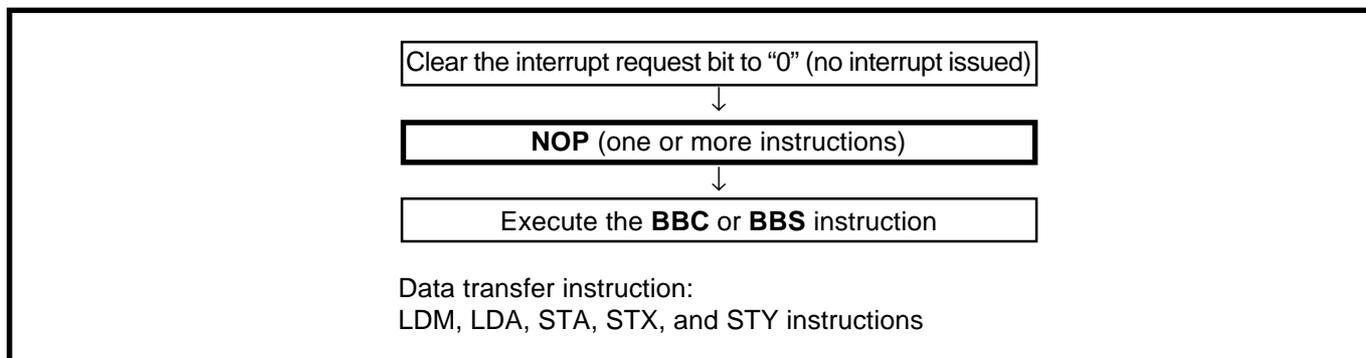
- When setting external interrupt active edge

Concerned register: Interrupt edge selection register (address  $0FF3_{16}$ )

Timer X mode register (address  $23_{16}$ )

**(2) Check of interrupt request bit**

- When executing the **BBC** or **BBS** instruction to an interrupt request bit of an interrupt request register immediately after this bit is set to "0" by using a data transfer instruction, execute one or more instructions before executing the **BBC** or **BBS** instruction.



**Fig. 3.2.2 Sequence of check of interrupt request bit**

■ **Reason**

If the **BBC** or **BBS** instruction is executed immediately after an interrupt request bit of an interrupt request register is cleared to "0", the value of the interrupt request bit before being cleared to "0" is read.

### 3.2.4 Notes on timer

- If a value  $n$  (between 0 and 255) is written to a timer latch, the frequency division ratio is  $1/(n+1)$ .
- When switching the count source by the timer 12 and X count source selection bits, the value of timer count is altered in unconsiderable amount owing to generating of thin pulses in the count input signals.  
Therefore, select the timer count source before set the value to the prescaler and the timer.

### 3.2.5 Notes on serial I/O

#### (1) Notes when selecting clock synchronous serial I/O (Serial I/O)

##### ① Stop of transmission operation

Clear the serial I/O enable bit and the transmit enable bit to "0" (Serial I/O and transmit disabled).

##### ● Reason

Since transmission is not stopped and the transmission circuit is not initialized even if only the serial I/O enable bit is cleared to "0" (Serial I/O disabled), the internal transmission is running (in this case, since pins TxD, RxD, SCLK, and SRDY function as I/O ports, the transmission data is not output). When data is written to the transmit buffer register in this state, data starts to be shifted to the transmit shift register. When the serial I/O enable bit is set to "1" at this time, the data during internally shifting is output to the TxD pin and an operation failure occurs.

##### ② Stop of receive operation

Clear the receive enable bit to "0" (receive disabled), or clear the serial I/O enable bit to "0" (Serial I/O disabled).

##### ③ Stop of transmit/receive operation

Clear the transmit enable bit and receive enable bit to "0" simultaneously (transmit and receive disabled).

(when data is transmitted and received in the clock synchronous serial I/O mode, any one of data transmission and reception cannot be stopped.)

##### ● Reason

In the clock synchronous serial I/O mode, the same clock is used for transmission and reception. If any one of transmission and reception is disabled, a bit error occurs because transmission and reception cannot be synchronized.

In this mode, the clock circuit of the transmission circuit also operates for data reception. Accordingly, the transmission circuit does not stop by clearing only the transmit enable bit to "0" (transmit disabled). Also, the transmission circuit is not initialized by clearing the serial I/O enable bit to "0" (Serial I/O disabled) (refer to (1) ①).

**(2) Notes when selecting clock asynchronous serial I/O (Serial I/O)****① Stop of transmission operation**

Clear the transmit enable bit to “0” (transmit disabled).

**● Reason**

Since transmission is not stopped and the transmission circuit is not initialized even if only the serial I/O enable bit is cleared to “0” (Serial I/O disabled), the internal transmission is running (in this case, since pins Tx<sub>D</sub>, Rx<sub>D</sub>, SCLK, and SRDY function as I/O ports, the transmission data is not output). When data is written to the transmit buffer register in this state, data starts to be shifted to the transmit shift register. When the serial I/O enable bit is set to “1” at this time, the data during internally shifting is output to the Tx<sub>D</sub> pin and an operation failure occurs.

**② Stop of receive operation**

Clear the receive enable bit to “0” (receive disabled).

**③ Stop of transmit/receive operation****Only transmission operation is stopped.**

Clear the transmit enable bit to “0” (transmit disabled).

**● Reason**

Since transmission is not stopped and the transmission circuit is not initialized even if only the serial I/O enable bit is cleared to “0” (Serial I/O disabled), the internal transmission is running (in this case, since pins Tx<sub>D</sub>, Rx<sub>D</sub>, SCLK, and SRDY function as I/O ports, the transmission data is not output). When data is written to the transmit buffer register in this state, data starts to be shifted to the transmit shift register. When the serial I/O enable bit is set to “1” at this time, the data during internally shifting is output to the Tx<sub>D</sub> pin and an operation failure occurs.

**Only receive operation is stopped.**

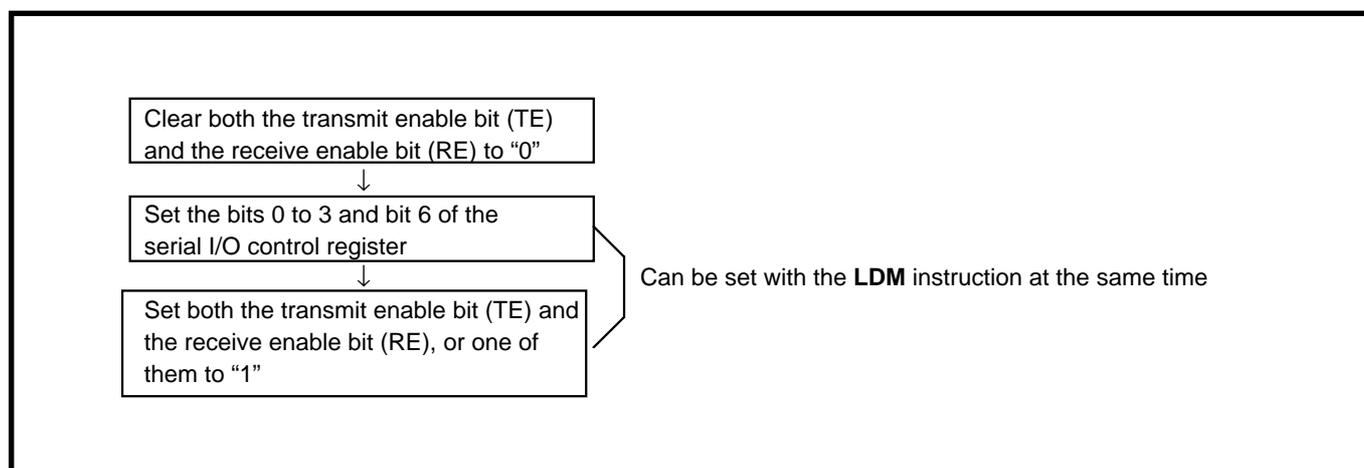
Clear the receive enable bit to “0” (receive disabled).

**(3)  $\overline{\text{SRDY}}$  output of reception side (Serial I/O)**

When signals are output from the  $\overline{\text{SRDY}}$  pin on the reception side by using an external clock in the clock synchronous serial I/O mode, set all of the receive enable bit, the  $\overline{\text{SRDY}}$  output enable bit, and the transmit enable bit to “1” (transmit enabled).

**(4) Setting serial I/O control register again (Serial I/O)**

Set the serial I/O control register again after the transmission and the reception circuits are reset by clearing both the transmit enable bit and the receive enable bit to “0.”



**Fig. 3.2.3 Sequence of setting serial I/O control register again**

**(5) Data transmission control with referring to transmit shift register completion flag (Serial I/O)**

The transmit shift register completion flag changes from “1” to “0” with a delay of 0.5 to 1.5 shift clocks. When data transmission is controlled with referring to the flag after writing the data to the transmit buffer register, note the delay.

**(6) Transmission control when external clock is selected (Serial I/O)**

When an external clock is used as the synchronous clock for data transmission, set the transmit enable bit to “1” at “H” of the SCLK input level. Also, write the transmit data to the transmit buffer register (serial I/O shift register) at “H” of the SCLK input level.

**(7) Transmit interrupt request when transmit enable bit is set (Serial I/O)**

When the transmit interrupt is used, set the transmit interrupt enable bit to transmit enabled as shown in the following sequence.

- ① Set the interrupt enable bit to “0” (disabled) with CLB instruction.
- ② Prepare serial I/O for transmission/reception.
- ③ Set the interrupt request bit to “0” with CLB instruction after 1 or more instruction has been executed.
- ④ Set the interrupt enable bit to “1” (enabled).

● Reason

When the transmission enable bit is set to “1”, the transmit buffer empty flag and transmit shift register completion flag are set to “1”. The interrupt request is generated and the transmission interrupt bit is set regardless of which of the two timings listed below is selected as the timing for the transmission interrupt to be generated.

- Transmit buffer empty flag is set to “1”
- Transmit shift register completion flag is set to “1”

**3.2.6 Notes on USB function****(1) Port pins (D0+, D0-, D1+, D1-, D2+, D2-) treatment**

- The USB specification requires a driver-impedance 28 to 44  $\Omega$ . In order to meet the USB specification impedance requirements, connect a resistor (27  $\Omega$  recommended) in series to the USB port pins. In addition, in order to reduce the ringing and control the falling/rising timing and a crossover point, connect a capacitor between the USB port pins and the Vss pin if necessary. The values and structure of those peripheral elements depend on the impedance characteristics and the layout of the printed circuit board. Accordingly, evaluate your system and observe waveforms before actual use and decide use of elements and the values of resistors and capacitors.
- Make sure the USB D+/D- lines do not cross any other wires. Keep a large GND area to protect the USB lines. Also, make sure you use a USB specification compliant connector for the connection.

**(2) USB<sub>REF</sub> pin treatment (Noise Elimination)**

- Connect a capacitor between the USB<sub>REF</sub> pin and the Vss pin. The capacitor should have a 2.2  $\mu$ F capacitor (electrolytic capacitor) and a 0.1  $\mu$ F capacitor (ceramic type capacitor) connected in parallel.
- In Vcc = 3.0 to 3.6 V operation, connect the USB<sub>REF</sub> pin directly to the Vcc pin in order to supply power to the USB port circuit. In addition, you will need to disable the built-in USB reference voltage circuit in this operation (set bit 4 of the USB control register to “0”.) If you are using the bus powered supply in this condition, the DC-DC converter must be placed outside the MCU.
- In Vcc = 4.00 to 5.25 V operation, do not connect the external DC-DC converter to the USB<sub>REF</sub> pin. Use the built-in USB reference voltage circuit.

**(3) USB Communication**

- In applications requiring high-reliability, we recommend providing the system with protective measures such as USB function initialization by software or USB reset by the host to prevent USB communication from being terminated unexpectedly, for example due to external causes such as noise.

### 3.2.7 Notes on A/D converter

#### (1) Analog input pin

Make the signal source impedance for analog input low, or equip an analog input pin with an external capacitor of 0.01  $\mu\text{F}$  to 1  $\mu\text{F}$ . Further, be sure to verify the operation of application products on the user side.

##### ● Reason

An analog input pin includes the capacitor for analog voltage comparison. Accordingly, when signals from signal source with high impedance are input to an analog input pin, charge and discharge noise generates. This may cause the A/D conversion precision to be worse.

#### (2) Clock frequency during A/D conversion

The comparator consists of a capacity coupling, and a charge of the capacity will be lost if the clock frequency is too low. Thus, make sure the following during an A/D conversion.

- $f(X_{\text{IN}})$  is 500 kHz or more
- Do not execute the **STP** instruction

### 3.2.8 Notes on watchdog timer

● Make sure that the watchdog timer does not underflow while waiting Stop release, because the watchdog timer keeps counting during that term.

● When the STP instruction disable bit has been set to "1", it is impossible to switch it to "0" by a program

### 3.2.9 Notes on $\overline{\text{RESET}}$ pin

#### Connecting capacitor

In case where the  $\overline{\text{RESET}}$  signal rise time is long, connect a ceramic capacitor or others across the  $\overline{\text{RESET}}$  pin and the VSS pin. Use a 1000 pF or more capacitor for high frequency use. When connecting the capacitor, note the following :

- Make the length of the wiring which is connected to a capacitor as short as possible.
- Be sure to verify the operation of application products on the user side.

##### ● Reason

If the several nanosecond or several ten nanosecond impulse noise enters the  $\overline{\text{RESET}}$  pin, it may cause a microcomputer failure.

### 3.2.10 Notes on PLL

● 6 MHz or 12 MHz external oscillator can be connected as an input reference clock ( $f(X_{\text{IN}})$ ). When using the frequency synthesized clock function, we recommend using the fastest frequency possible of  $f(X_{\text{IN}})$  as an input clock reference for the PLL.

● When enabling PLL operation from PLL disabled status (disabled when reset), set the USB clock select bit of USBCON to "0" ( $f(X_{\text{IN}})$ ) to operate with the main clock ( $f(X_{\text{IN}})$ ).

● When supplying  $f_{\text{VCO}}$  to the USB block after setting PLL operation enable bit to "1" (PLL enabled), wait for the oscillation stable time (1 ms or less) of PLL to avoid any instability caused by the clock, then set USB clock select bit to "1" (USB clock).

● When selecting  $f_{\text{SYN}}$  as an internal system clock,  $f_{\text{USB}}$  must be 48 MHz.

● When selecting  $f_{\text{SYN}}$  as an internal system clock, change the system clock selection bit to main clock ( $f(X_{\text{IN}})$ ) before executing STP instruction. It is because the following are needed for the low-power consumption:

- $f_{\text{USB}}$  must be stopped by disabling PLL operation in Stop mode.
- The timer 1 for waiting oscillation stabilization when returning from Stop mode will require the input count source.

### 3.2.11 Notes on stand-by function

#### (1) Notes on using stop mode

##### ■ Register setting

Since values of the prescaler 12 and Timer 1 are automatically reloaded when returning from the stop mode, set them again, respectively. (When the oscillation stabilizing time set after STP instruction released bit is "0")

##### ■ Clock restoration

When the main clock side is set as a system clock, the oscillation stabilizing time for approximately 8,000 cycles of the X<sub>IN</sub> input is reserved at restoration from the stop mode.

#### (2) Notes on stand-by function

In stand-by state\*<sup>1</sup> for low-power dissipation, do not make input levels of an input port and an I/O port "undefined".

Pull-up (connect the port to V<sub>CC</sub>) these ports through a resistor.

When determining a resistance value, note the following points:

- External circuit
- Variation of output levels during the ordinary operation

When using built-in pull-up resistor, note on varied current values.

- When setting as an input port: Fix its input level
- When setting as an output port: Prevent current from flowing out to external

##### ● Reason

The potential which is input to the input buffer in a microcomputer is unstable in the state that input levels of an input port and an I/O port are "undefined". This may cause power source current.

\*<sup>1</sup> stand-by state : the stop mode by executing the **STP** instruction  
the wait mode by executing the **WIT** instruction

### 3.2.12 Notes on CPU rewrite mode

#### (1) Operation speed

During CPU rewrite mode, set the internal clock  $\phi$  1.5 MHz or less using the system clock division ratio selection bits (bits 6 and 7 of address 003B<sub>16</sub>).

#### (2) Instructions inhibited against use

The instructions which refer to the internal data of the flash memory cannot be used during the CPU rewrite mode .

#### (3) Interrupts inhibited against use

The interrupts cannot be used during the CPU rewrite mode because they refer to the internal data of the flash memory.

#### (4) Watchdog timer

In case of the watchdog timer has been running already, the internal reset generated by watchdog timer underflow does not happen, because of watchdog timer is always clearing during program or erase operation.

#### (5) Reset

Reset is always valid. In case of CNV<sub>SS</sub> = "H" when reset is released, boot mode is active. So the program starts from the address contained in address FFFC<sub>16</sub> and FFFD<sub>16</sub> in boot ROM area.

### 3.2.13 Notes on programming

#### (1) Processor status register

##### ① Initializing of processor status register

Flags which affect program execution must be initialized after a reset.

In particular, it is essential to initialize the T and D flags because they have an important effect on calculations.

##### ● Reason

After a reset, the contents of the processor status register (PS) are undefined except for the I flag which is "1".

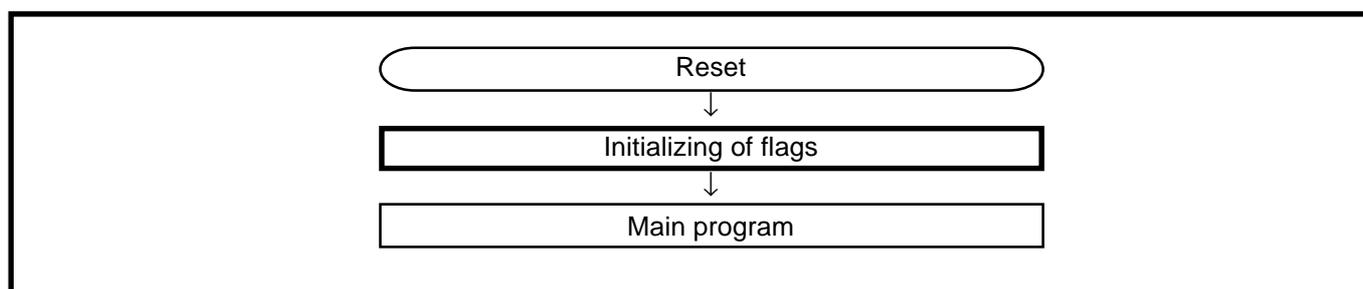


Fig. 3.2.4 Initialization of processor status register

##### ② How to reference the processor status register

To reference the contents of the processor status register (PS), execute the **PHP** instruction once then read the contents of (S+1). If necessary, execute the **PLP** instruction to return the PS to its original status.

A **NOP** instruction should be executed after every **PLP** instruction.

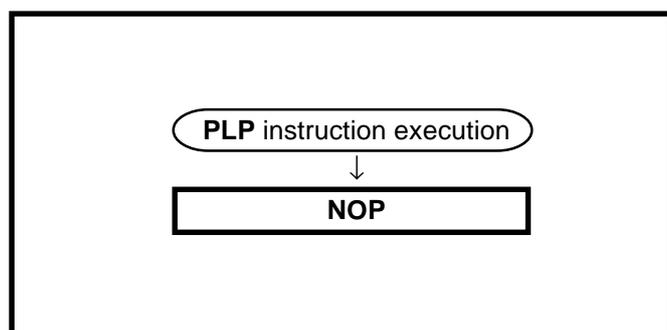


Fig. 3.2.5 Sequence of PLP instruction execution

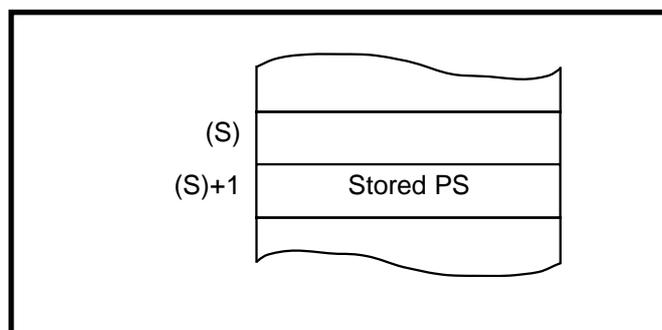


Fig. 3.2.6 Stack memory contents after PHP instruction execution

#### (2) BRK instruction

##### ① Interrupt priority level

When the BRK instruction is executed with the following conditions satisfied, the interrupt execution is started from the address of interrupt vector which has the highest priority.

- Interrupt request bit and interrupt enable bit are set to "1".
- Interrupt disable flag (I) is set to "1" to disable interrupt.

### (3) Decimal calculations

#### ① Execution of decimal calculations

The **ADC** and **SBC** are the only instructions which will yield proper decimal notation, set the decimal mode flag (D) to “1” with the **SED** instruction. After executing the **ADC** or **SBC** instruction, execute another instruction before executing the **SEC**, **CLC**, or **CLD** instruction.

#### ② Notes on status flag in decimal mode

When decimal mode is selected, the values of three of the flags in the status register (the N, V, and Z flags) are invalid after a **ADC** or **SBC** instruction is executed.

The carry flag (C) is set to “1” if a carry is generated as a result of the calculation, or is cleared to “0” if a borrow is generated. To determine whether a calculation has generated a carry, the C flag must be initialized to “0” before each calculation. To check for a borrow, the C flag must be initialized to “1” before each calculation.

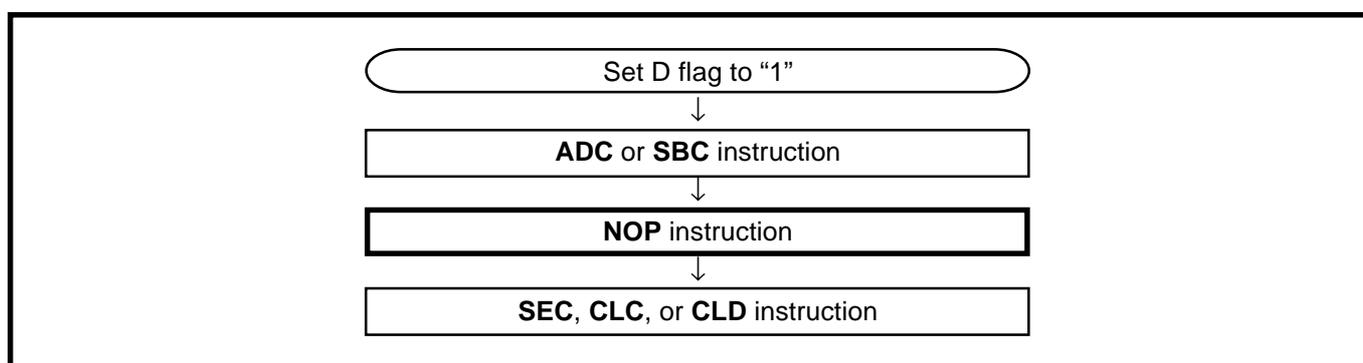


Fig. 3.2.7 Status flag at decimal calculations

#### (4) JMP instruction

When using the **JMP** instruction in indirect addressing mode, do not specify the last address on a page as an indirect address.

#### (5) Multiplication and Division Instructions

- The index X mode (T) and the decimal mode (D) flags do not affect the **MUL** and **DIV** instruction.
- The execution of these instructions does not change the contents of the processor status register.

#### (6) Ports

The contents of the port direction registers cannot be read. The following cannot be used:

- The data transfer instruction (**LDA**, etc.)
- The operation instruction when the index X mode flag (T) is “1”
- The addressing mode which uses the value of a direction register as an index
- The bit-test instruction (**BBC** or **BBS**, etc.) to a direction register
- The read-modify-write instructions (**ROR**, **CLB**, or **SEB**, etc.) to a direction register.

Use instructions such as **LDM** and **STA**, etc., to set the port direction registers.

#### (7) Instruction Execution Time

The instruction execution time is obtained by multiplying the frequency of the internal clock  $f$  by the number of cycles needed to execute an instruction.

The number of cycles required to execute an instruction is shown in the list of machine instructions.

The frequency of the internal clock  $f$  is half of the  $X_{IN}$  frequency in high-speed mode.

**3.2.14 Notes on flash memory version**

The CNVss pin is connected to the internal memory circuit block by a low-ohmic resistance, since it has the multiplexed function to be a programmable power source pin ( $V_{PP}$  pin) as well.

To improve the noise reduction, connect a track between CNVss pin and Vss pin or Vcc pin with 1 to 10 k $\Omega$  resistance.

The mask ROM version track of CNVss pin has no operational interference even if it is connected to Vss pin or Vcc pin via a resistor.

**3.2.15 Electric Characteristic Differences Between Mask ROM and Flash Memory Version MCUs**

There are differences in electric characteristics, operation margin, noise immunity, and noise radiation between Mask ROM and Flash Memory version MCUs due to the difference in the manufacturing processes. When manufacturing an application system with the Flash Memory version and then switching to use of the Mask ROM version, please perform sufficient evaluations for the commercial samples of the Mask ROM version.

### 3.3 Countermeasures against noise

Countermeasures against noise are described below. The following countermeasures are effective against noise in theory, however, it is necessary not only to take measures as follows but to evaluate before actual use.

#### 3.3.1 Shortest wiring length

The wiring on a printed circuit board can function as an antenna which feeds noise into the microcomputer. The shorter the total wiring length (by mm unit), the less the possibility of noise insertion into a microcomputer.

##### (1) Package

Select the smallest possible package to make the total wiring length short.

##### ● Reason

The wiring length depends on a microcomputer package. Use of a small package, for example QFP and not DIP, makes the total wiring length short to reduce influence of noise.

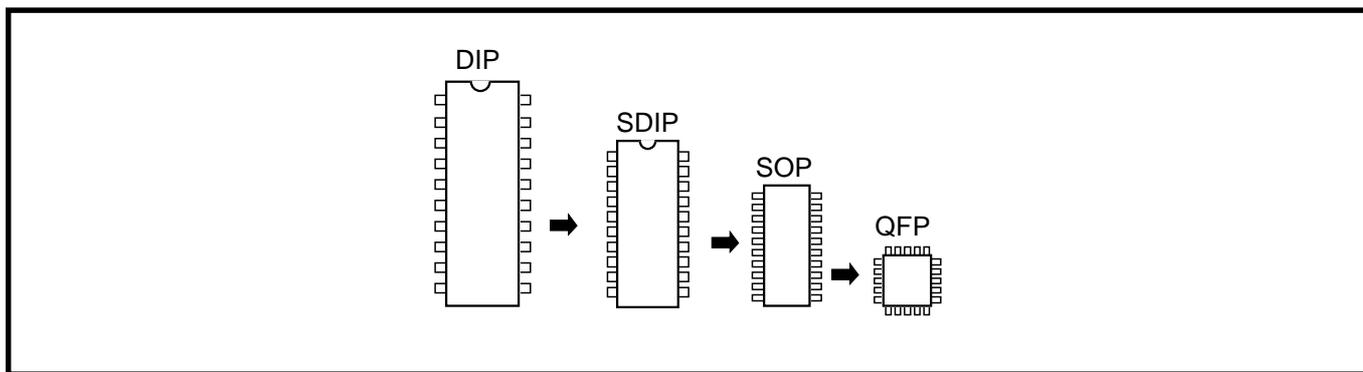


Fig. 3.3.1 Selection of packages

##### (2) Wiring for $\overline{\text{RESET}}$ pin

Make the length of wiring which is connected to the  $\overline{\text{RESET}}$  pin as short as possible. Especially, connect a capacitor across the  $\overline{\text{RESET}}$  pin and the  $V_{\text{ss}}$  pin with the shortest possible wiring (within 20mm).

##### ● Reason

The width of a pulse input into the  $\overline{\text{RESET}}$  pin is determined by the timing necessary conditions. If noise having a shorter pulse width than the standard is input to the  $\overline{\text{RESET}}$  pin, the reset is released before the internal state of the microcomputer is completely initialized. This may cause a program runaway.

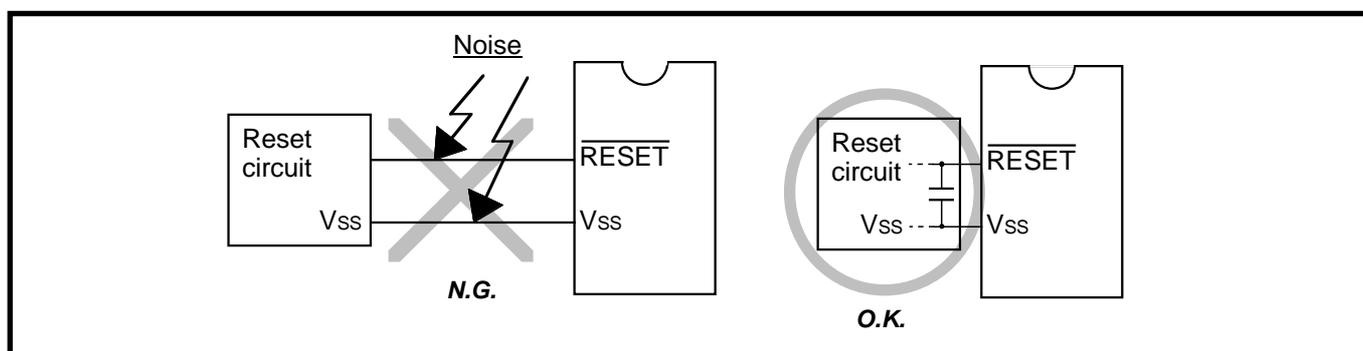


Fig. 3.3.2 Wiring for the  $\overline{\text{RESET}}$  pin

**(3) Wiring for clock input/output pins**

- Make the length of wiring which is connected to clock I/O pins as short as possible.
- Make the length of wiring (within 20 mm) across the grounding lead of a capacitor which is connected to an oscillator and the Vss pin of a microcomputer as short as possible.
- Separate the Vss pattern only for oscillation from other Vss patterns.

**● Reason**

If noise enters clock I/O pins, clock waveforms may be deformed. This may cause a program failure or program runaway. Also, if a potential difference is caused by the noise between the Vss level of a microcomputer and the Vss level of an oscillator, the correct clock will not be input in the microcomputer.

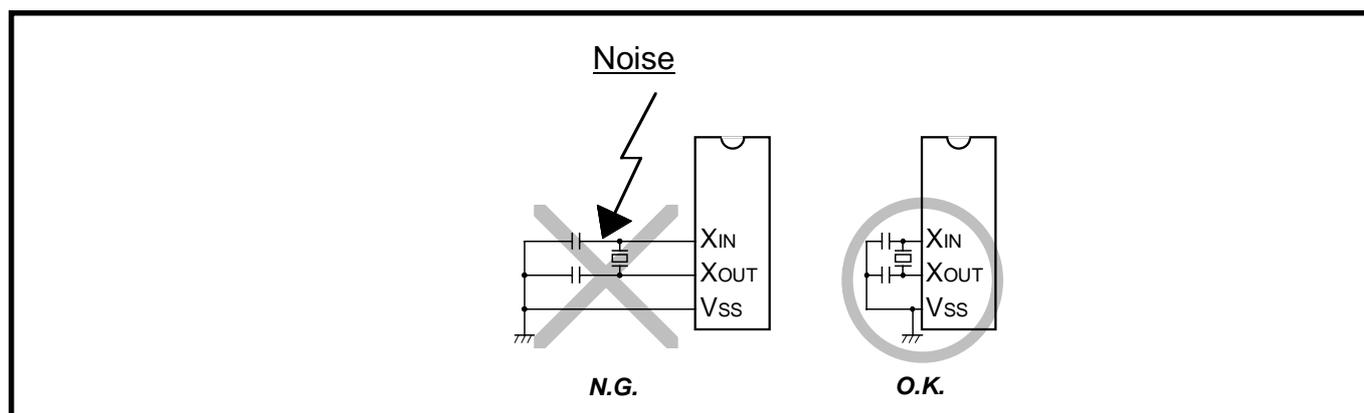


Fig. 3.3.3 Wiring for clock I/O pins

**(4) Wiring to CNVss pin**

Connect the CNVss pin to the Vss pin with the shortest possible wiring.

**● Reason**

The processor mode of a microcomputer is influenced by a potential at the CNVss pin. If a potential difference is caused by the noise between pins CNVss and Vss, the processor mode may become unstable. This may cause a microcomputer malfunction or a program runaway.

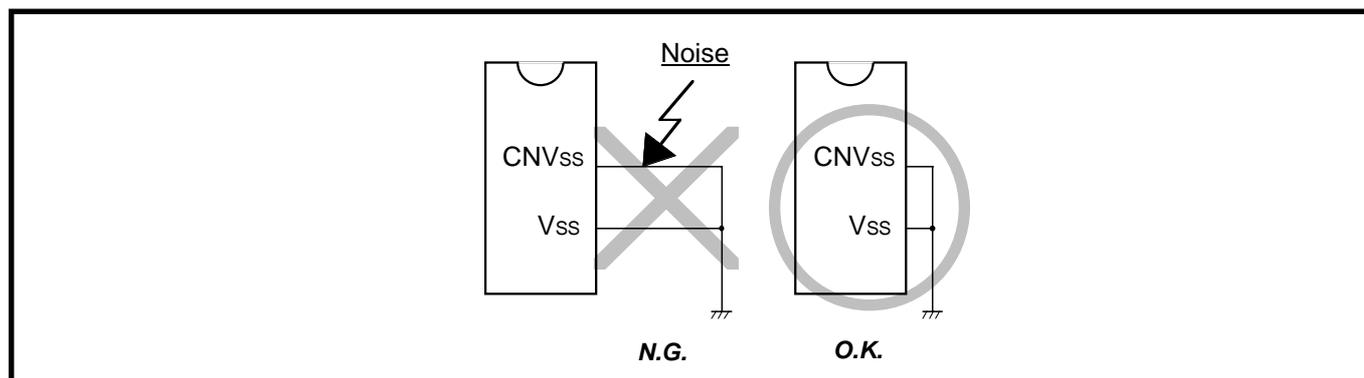


Fig. 3.3.4 Wiring for CNVss pin

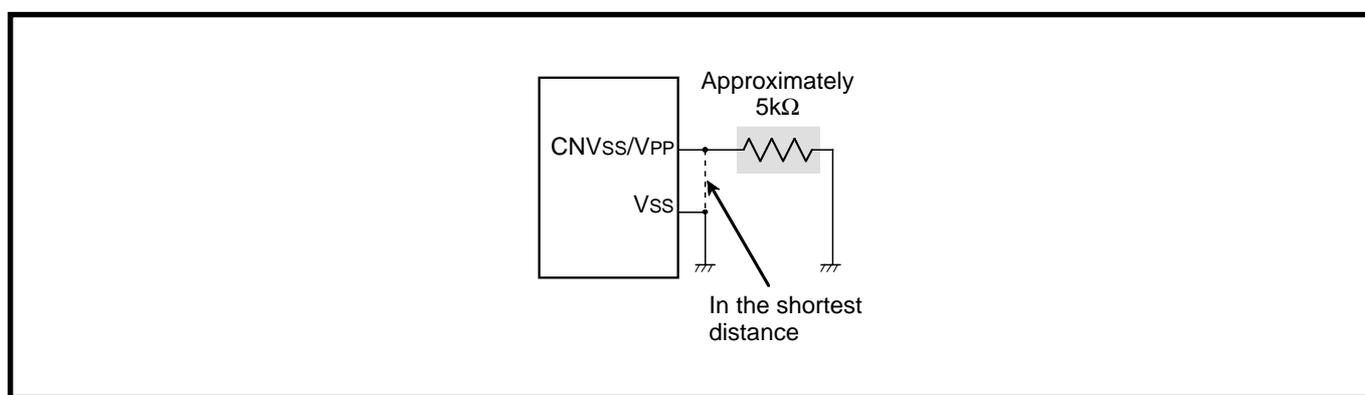
**(5) Wiring to VPP pin of Flash memory version**

Connect an approximately 5 k $\Omega$  resistor to the VPP pin the shortest possible in series and also to the VSS pin. When not connecting the resistor, make the length of wiring between the VPP pin and the VSS pin the shortest possible.

**Note:** Even when a circuit which included an approximately 5 k $\Omega$  resistor is used in the Mask ROM version, the microcomputer operates correctly.

**● Reason**

The VPP pin of the flash memory version is the power source input pin for the built-in flash memory. When programming in the built-in flash memory, the impedance of the VPP pin is low to allow the electric current for writing flow into the flash memory. Because of this, noise can enter easily. If noise enters the VPP pin, abnormal instruction codes or data are read from the built-in flash memory, which may cause a program runaway.

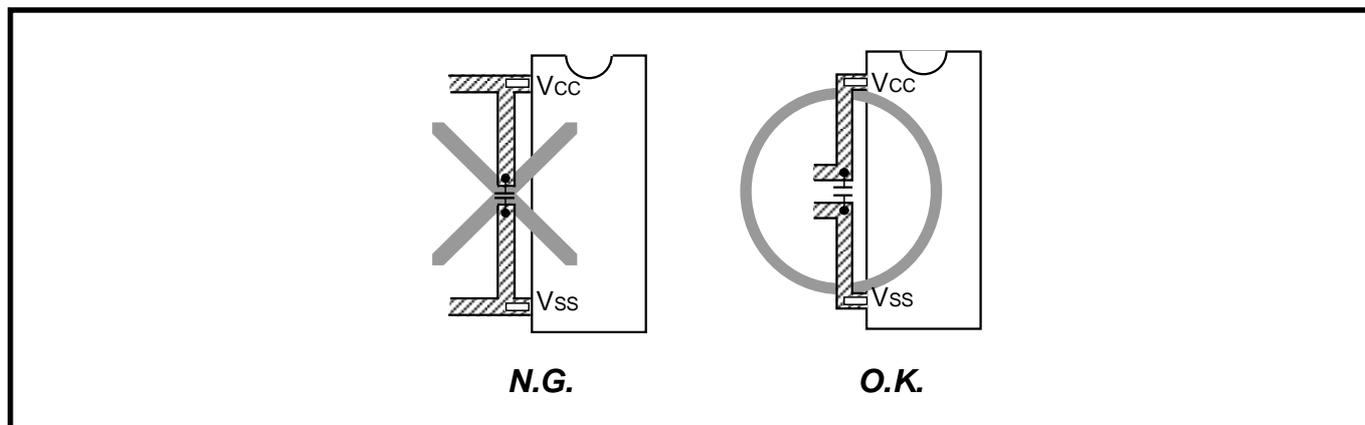


**Fig. 3.3.5 Wiring for the VPP pin of the flash memory version**

**3.3.2 Connection of bypass capacitor across Vss line and Vcc line**

Connect an approximately 0.1  $\mu$ F bypass capacitor across the VSS line and the VCC line as follows:

- Connect a bypass capacitor across the VSS pin and the VCC pin at equal length.
- Connect a bypass capacitor across the VSS pin and the VCC pin with the shortest possible wiring.
- Use lines with a larger diameter than other signal lines for VSS line and VCC line.
- Connect the power source wiring via a bypass capacitor to the VSS pin and the VCC pin.



**Fig. 3.3.6 Bypass capacitor across the Vss line and the Vcc line**

### 3.3.3 Wiring to analog input pins

- Connect an approximately  $100\ \Omega$  to  $1\ \text{k}\Omega$  resistor to an analog signal line which is connected to an analog input pin in series. Besides, connect the resistor to the microcomputer as close as possible.
- Connect an approximately  $1000\ \text{pF}$  capacitor across the Vss pin and the analog input pin. Besides, connect the capacitor to the Vss pin as close as possible. Also, connect the capacitor across the analog input pin and the Vss pin at equal length.

#### ● Reason

Signals which is input in an analog input pin (such as an A/D converter/comparator input pin) are usually output signals from sensor. The sensor which detects a change of event is installed far from the printed circuit board with a microcomputer, the wiring to an analog input pin is longer necessarily. This long wiring functions as an antenna which feeds noise into the microcomputer, which causes noise to an analog input pin.

If a capacitor between an analog input pin and the V<sub>SS</sub> pin is grounded at a position far away from the V<sub>SS</sub> pin, noise on the GND line may enter a microcomputer through the capacitor.

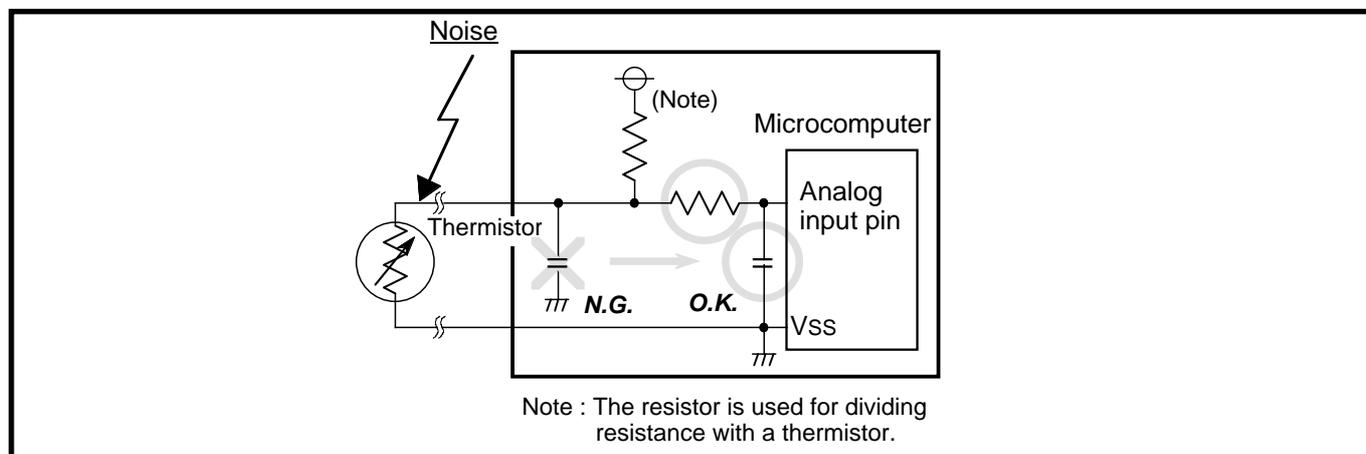


Fig. 3.3.7 Analog signal line and a resistor and a capacitor

### 3.3.4 Oscillator concerns

Take care to prevent an oscillator that generates clocks for a microcomputer operation from being affected by other signals.

#### (1) Keeping oscillator away from large current signal lines

Install a microcomputer (and especially an oscillator) as far as possible from signal lines where a current larger than the tolerance of current value flows.

##### ● Reason

In the system using a microcomputer, there are signal lines for controlling motors, LEDs, and thermal heads or others. When a large current flows through those signal lines, strong noise occurs because of mutual inductance.

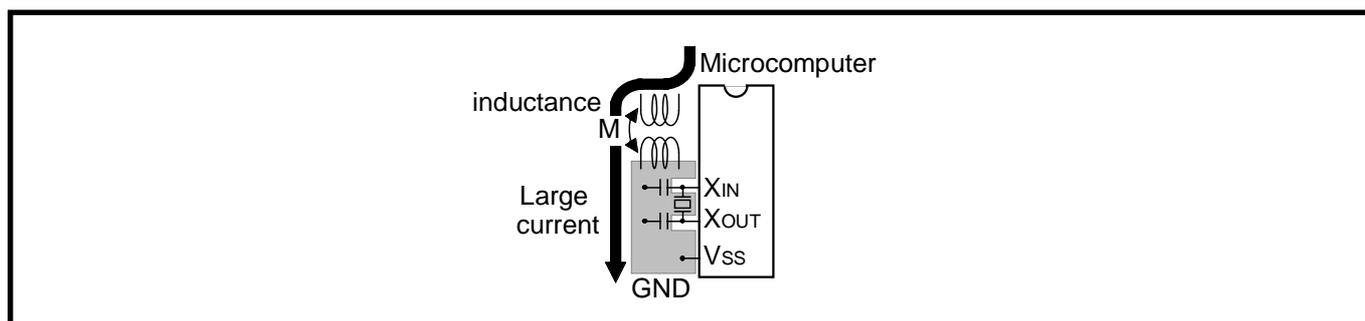


Fig. 3.3.8 Wiring for a large current signal line

#### (2) Installing oscillator away from signal lines where potential levels change frequently

Install an oscillator and a connecting pattern of an oscillator away from signal lines where potential levels change frequently. Also, do not cross such signal lines over the clock lines or the signal lines which are sensitive to noise.

##### ● Reason

Signal lines where potential levels change frequently (such as the CNTR pin signal line) may affect other lines at signal rising edge or falling edge. If such lines cross over a clock line, clock waveforms may be deformed, which causes a microcomputer failure or a program runaway.

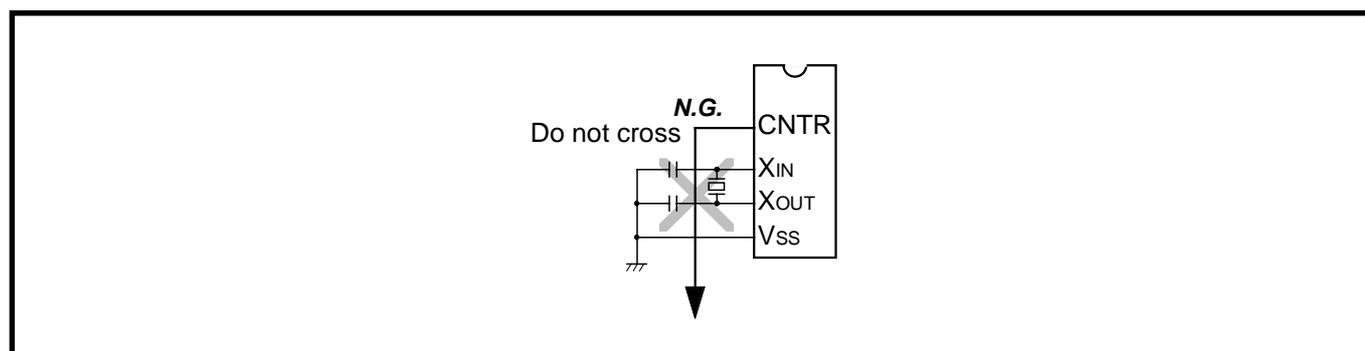


Fig. 3.3.9 Wiring of  $\overline{\text{RESET}}$  pin

### (3) Oscillator protection using Vss pattern

As for a two-sided printed circuit board, print a Vss pattern on the underside (soldering side) of the position (on the component side) where an oscillator is mounted.

Connect the Vss pattern to the microcomputer Vss pin with the shortest possible wiring. Besides, separate this Vss pattern from other Vss patterns.

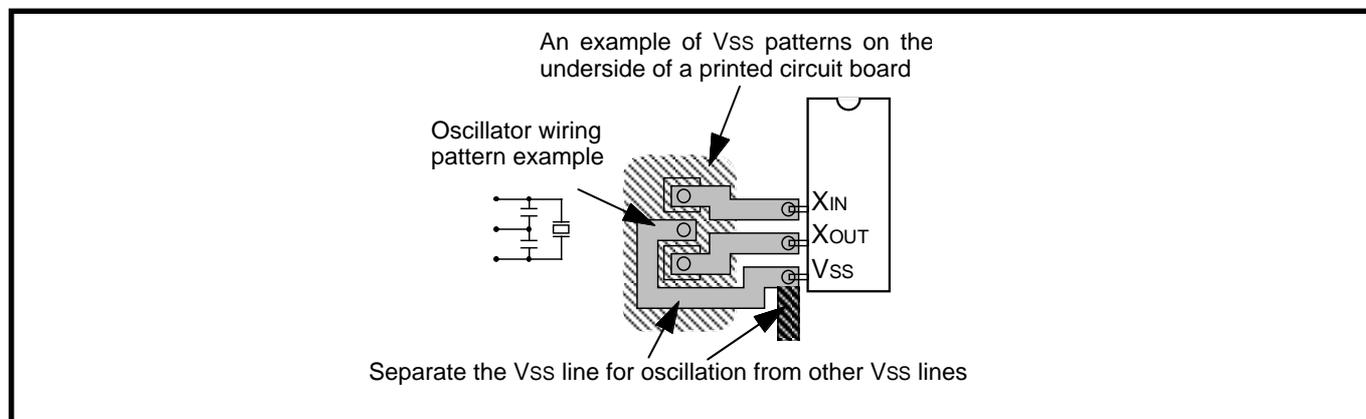


Fig. 3.3.10 Vss pattern on the underside of an oscillator

### 3.3.5 Setup for I/O ports

Setup I/O ports using hardware and software as follows:

#### <Hardware>

- Connect a resistor of 100  $\Omega$  or more to an I/O port in series.

#### <Software>

- As for an input port, read data several times by a program for checking whether input levels are equal or not.
- As for an output port, since the output data may reverse because of noise, rewrite data to its port latch at fixed periods.
- Rewrite data to direction registers at fixed periods.

**Note:** When a direction register is set for input port again at fixed periods, a several-nanosecond short pulse may be output from this port. If this is undesirable, connect a capacitor to this port to remove the noise pulse.

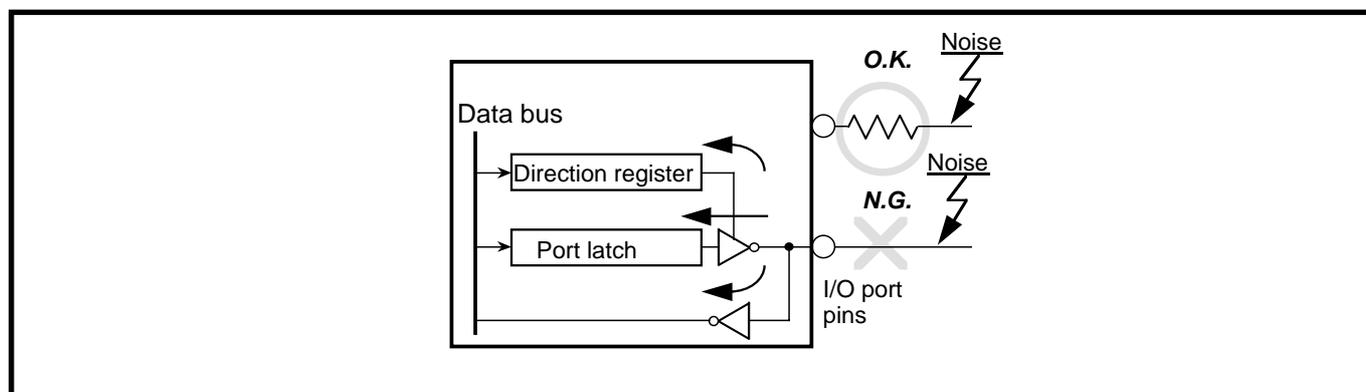


Fig. 3.3.11 Setup for I/O ports

### 3.3.6 Providing of watchdog timer function by software

If a microcomputer runs away because of noise or others, it can be detected by a software watchdog timer and the microcomputer can be reset to normal operation. This is equal to or more effective than program runaway detection by a hardware watchdog timer. The following shows an example of a watchdog timer provided by software.

In the following example, to reset a microcomputer to normal operation, the main routine detects errors of the interrupt processing routine and the interrupt processing routine detects errors of the main routine. This example assumes that interrupt processing is repeated multiple times in a single main routine processing.

#### <The main routine>

- Assigns a single byte of RAM to a software watchdog timer (SWDT) and writes the initial value N in the SWDT once at each execution of the main routine. The initial value N should satisfy the following condition:  

$$N+1 \geq (\text{Counts of interrupt processing executed in each main routine})$$
 As the main routine execution cycle may change because of an interrupt processing or others, the initial value N should have a margin.
- Watches the operation of the interrupt processing routine by comparing the SWDT contents with counts of interrupt processing after the initial value N has been set.
- Detects that the interrupt processing routine has failed and determines to branch to the program initialization routine for recovery processing in the following case:  
 If the SWDT contents do not change after interrupt processing.

#### <The interrupt processing routine>

- Decrements the SWDT contents by 1 at each interrupt processing.
- Determines that the main routine operates normally when the SWDT contents are reset to the initial value N at almost fixed cycles (at the fixed interrupt processing count).
- Detects that the main routine has failed and determines to branch to the program initialization routine for recovery processing in the following case:  
 If the SWDT contents are not initialized to the initial value N but continued to decrement and if they reach 0 or less.

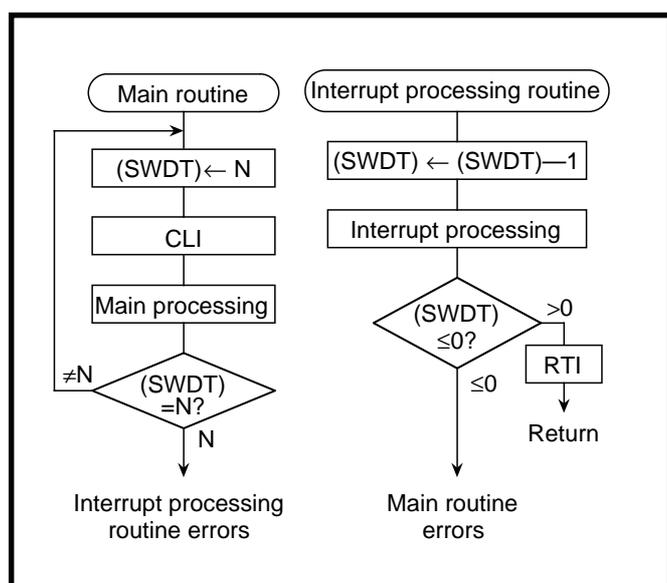


Fig. 3.3.12 Watchdog timer by software

### 3.4 List of registers

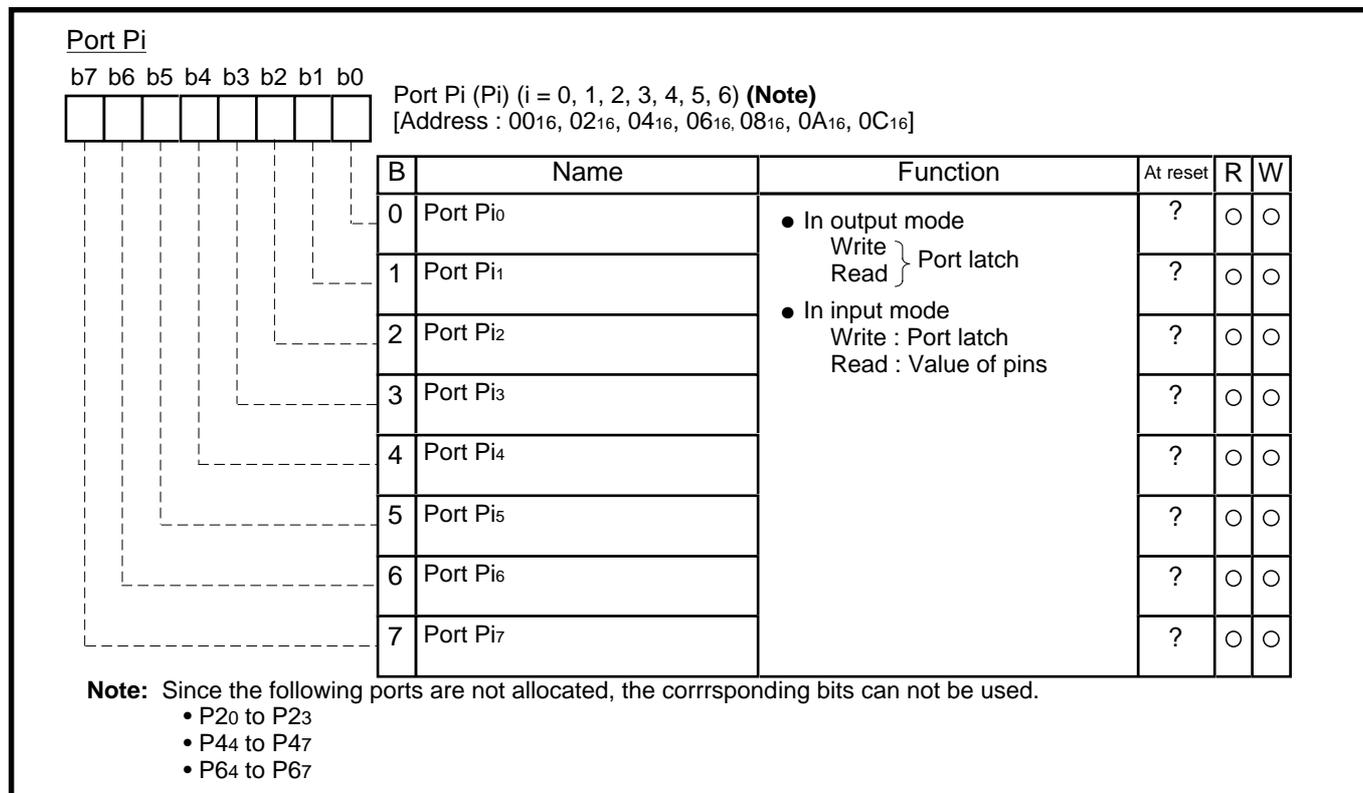


Fig. 3.4.1 Structure of Port Pi

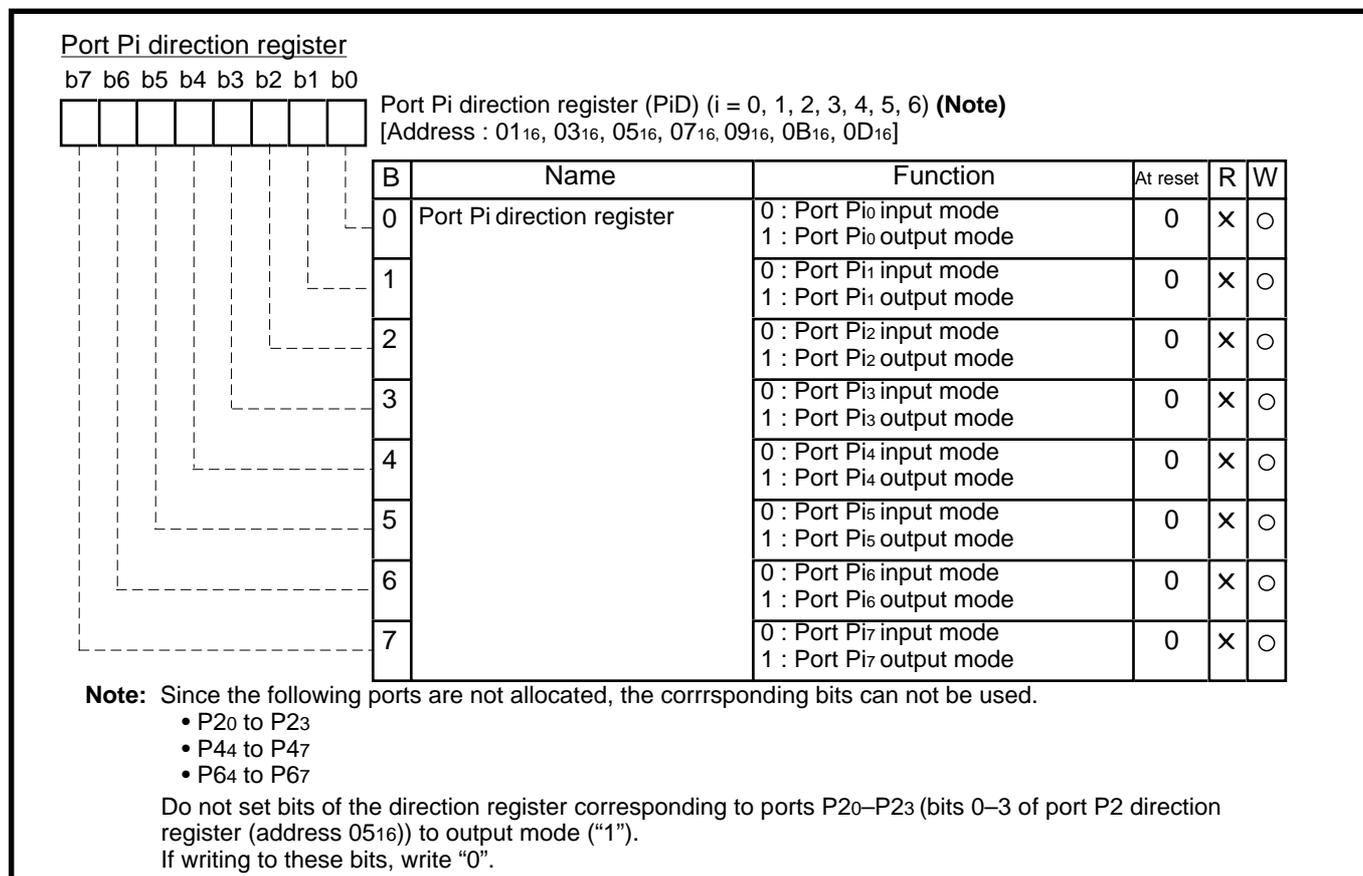


Fig. 3.4.2 Structure of Port Pi direction register

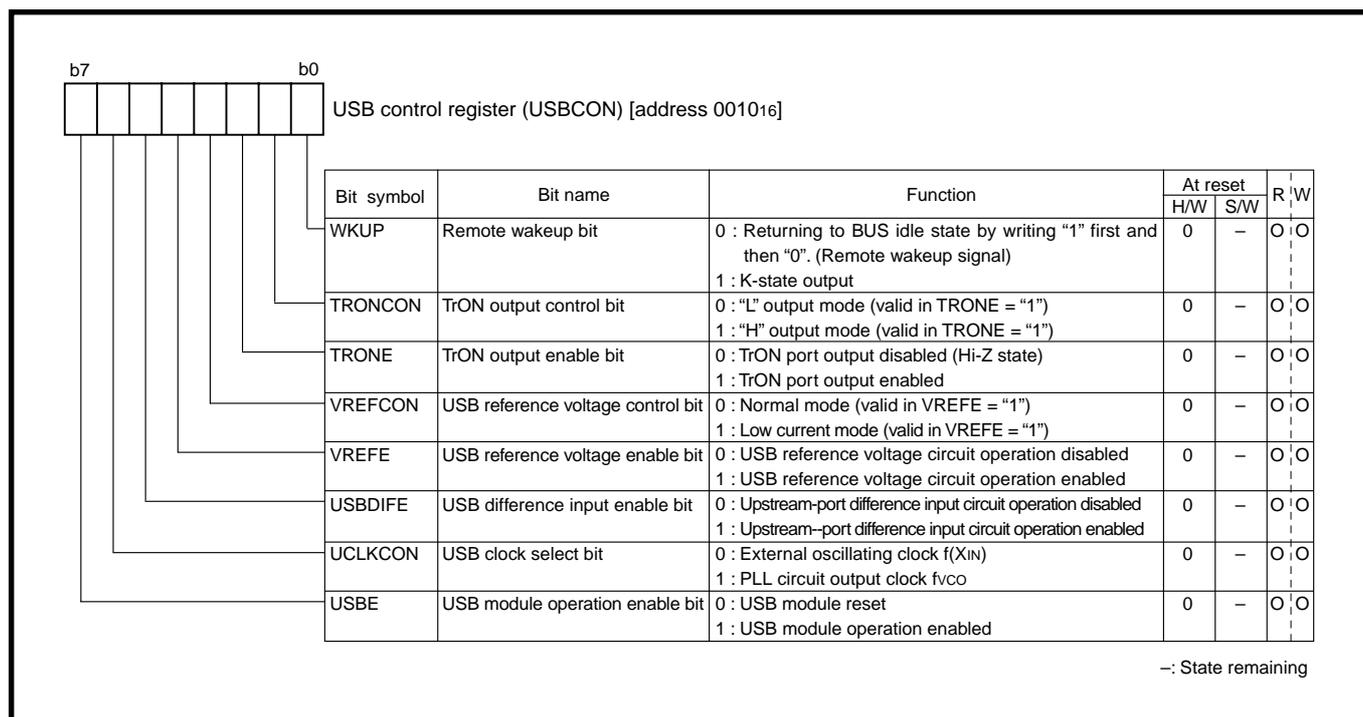


Fig. 3.4.3 Structure of USB control register

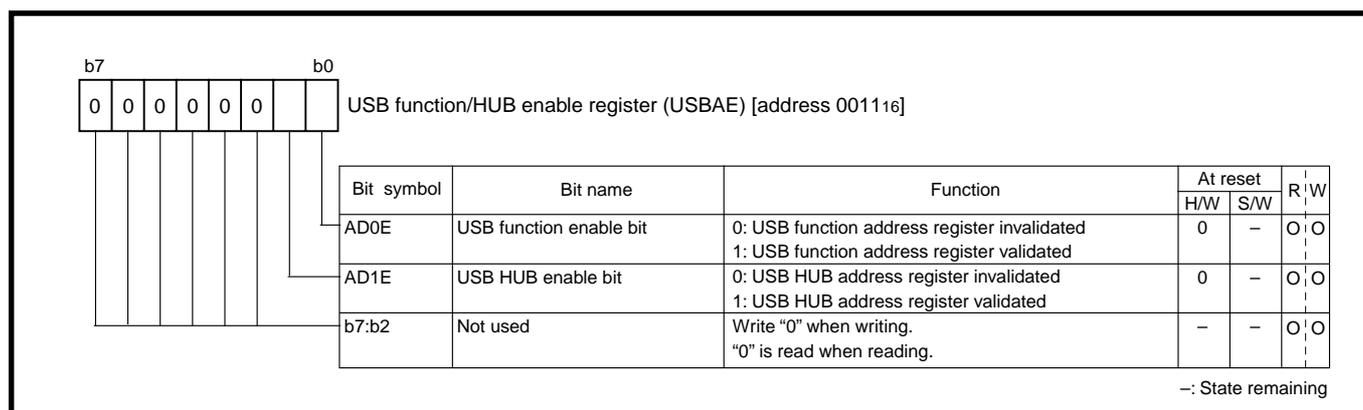


Fig. 3.4.4 Structure of USB function/HUB enable register

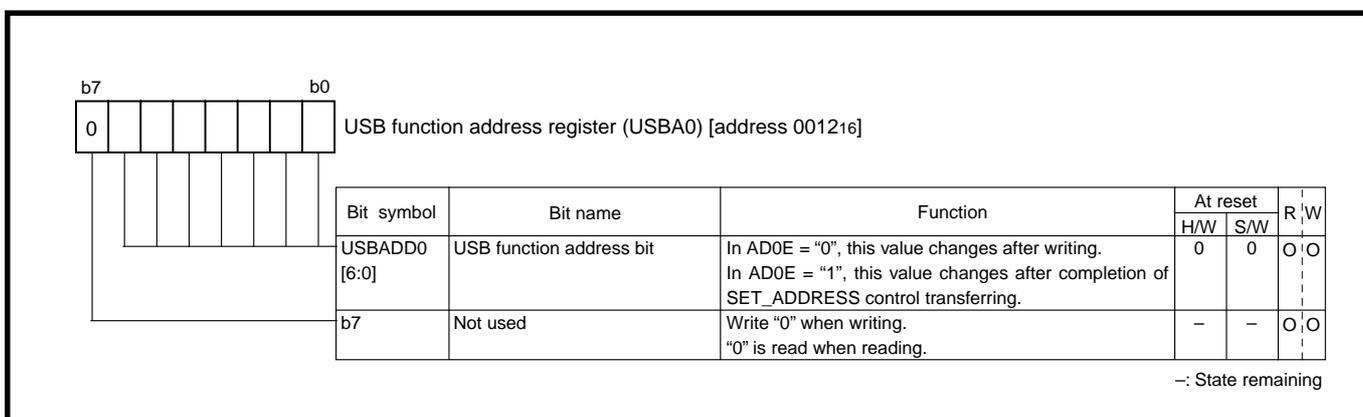


Fig. 3.4.5 Structure of USB function address register

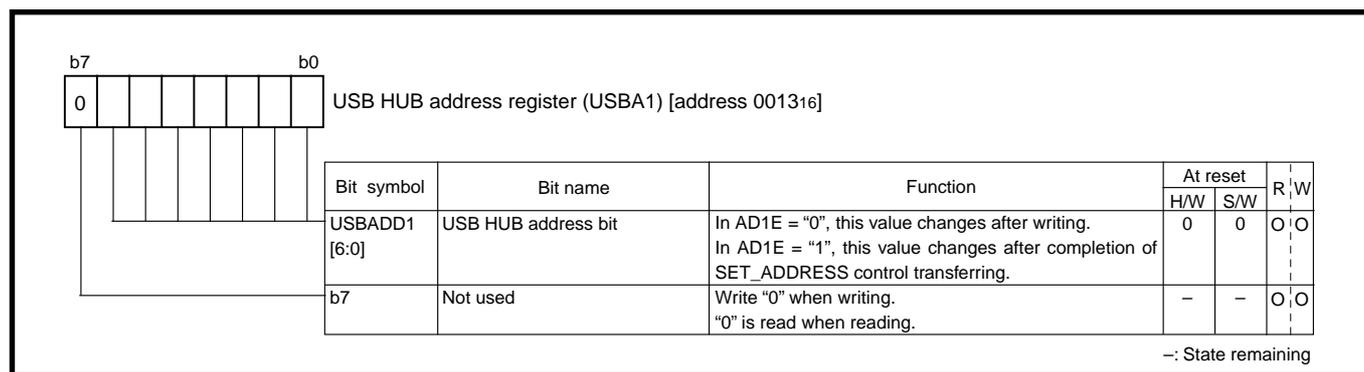


Fig. 3.4.6 Structure of USB HUB address register

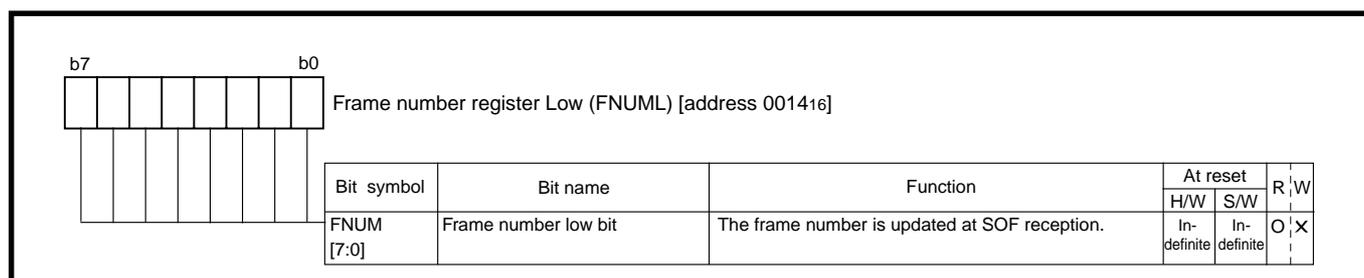


Fig. 3.4.7 Structure of Frame number register Low

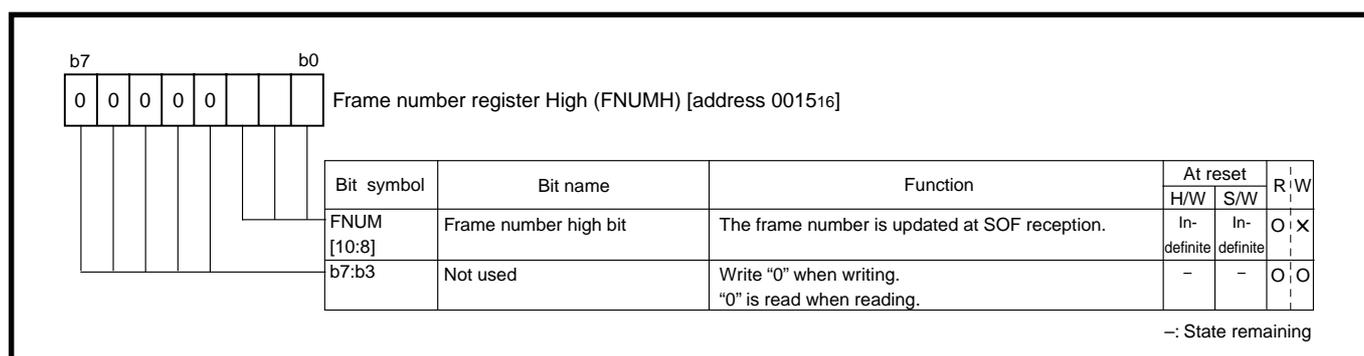


Fig. 3.4.8 Structure of Frame number register High

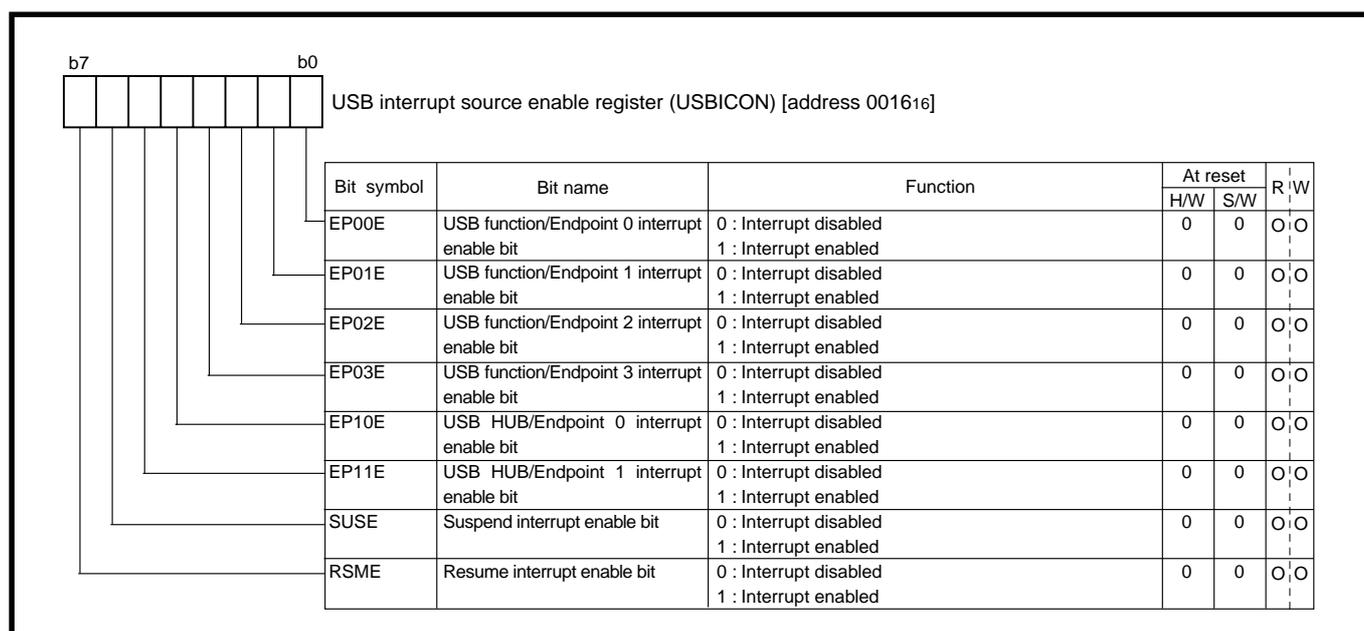


Fig. 3.4.9 Structure of USB interrupt source enable register

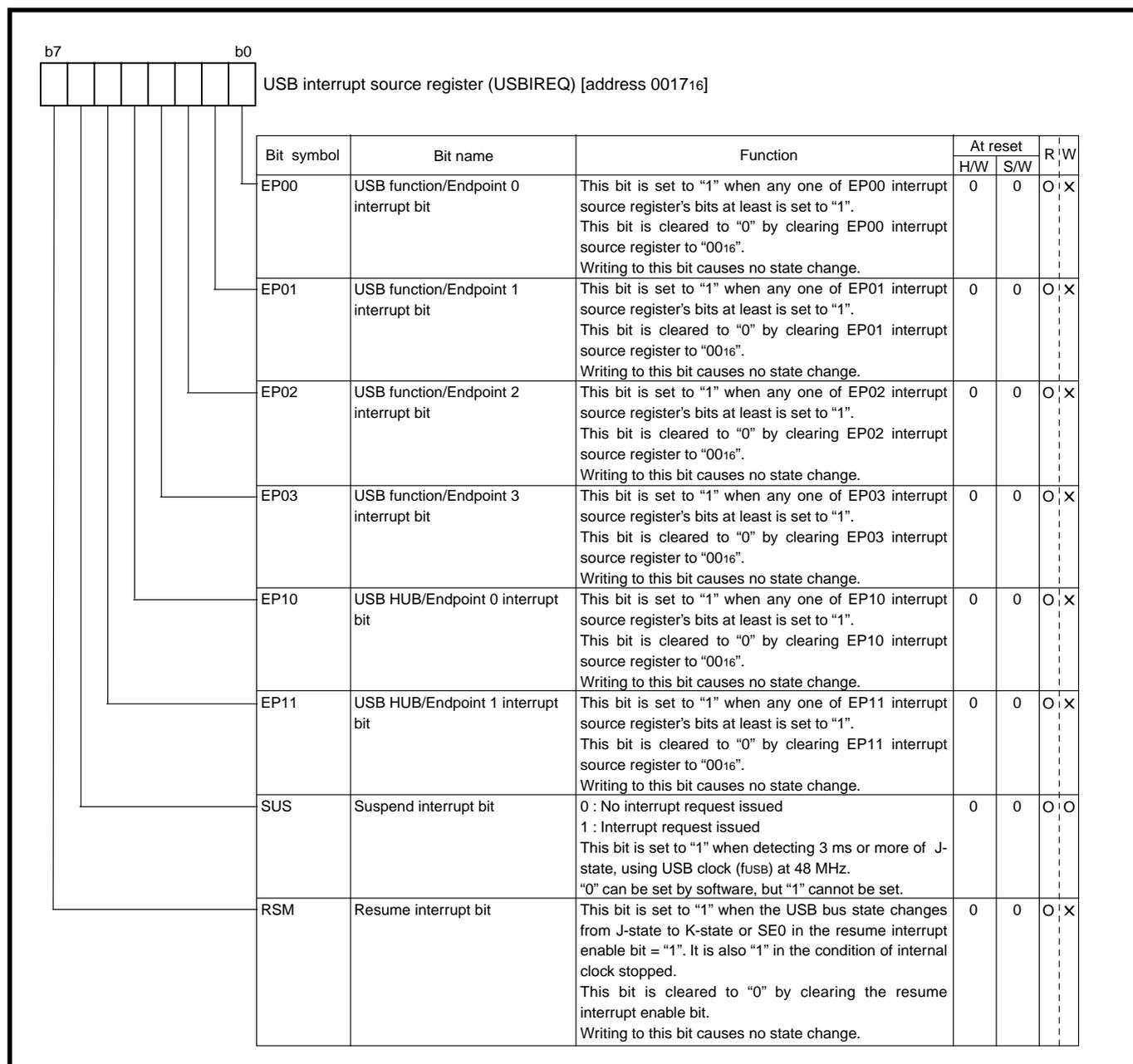


Fig. 3.4.10 Structure of USB interrupt source register

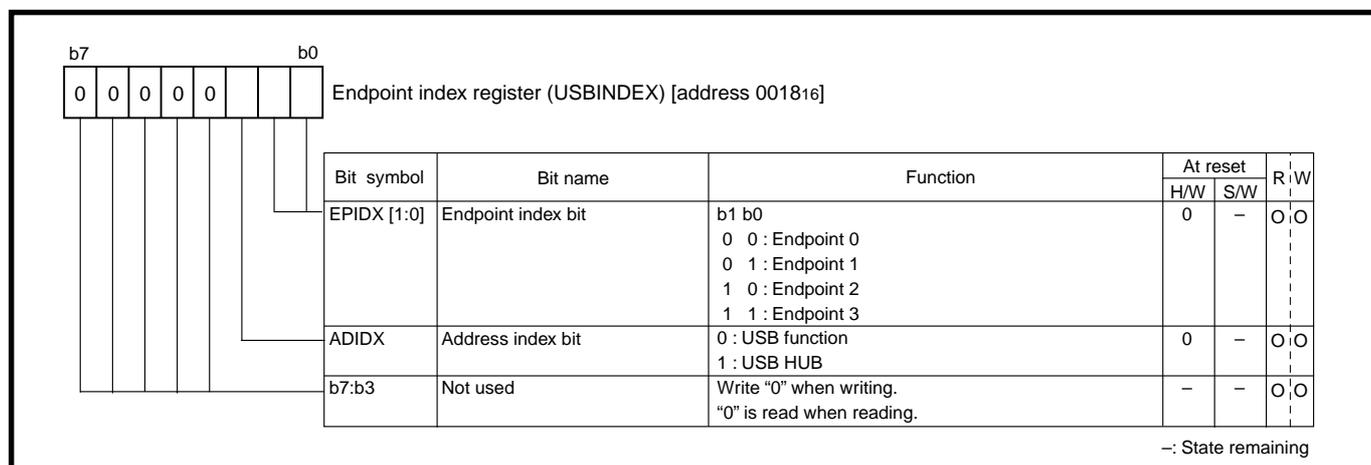


Fig. 3.4.11 Structure of Endpoint index register

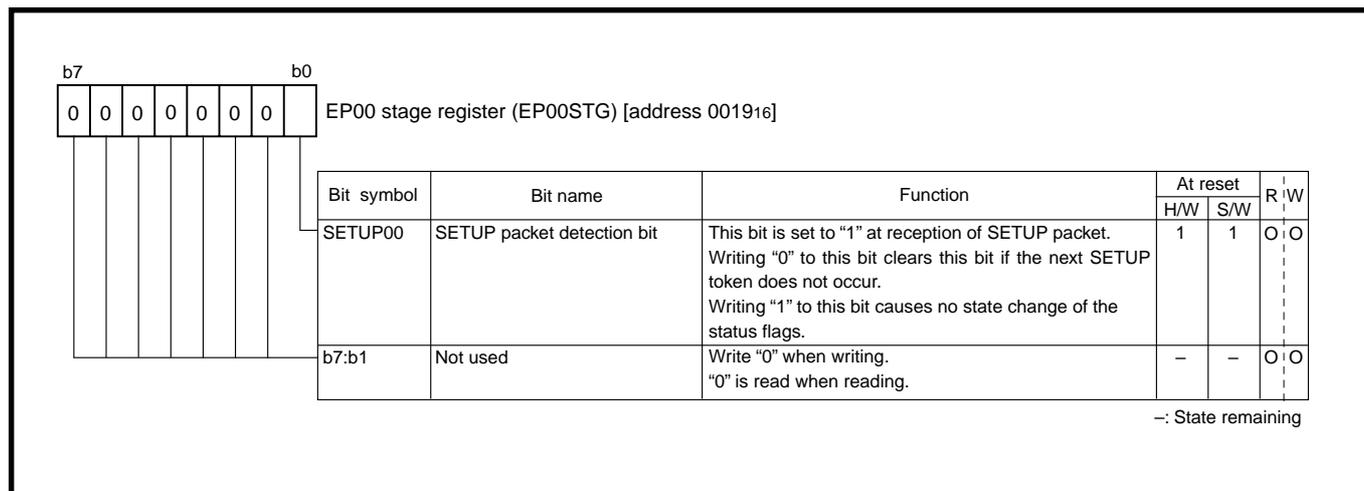


Fig. 3.4.12 Structure of EP00 stage register

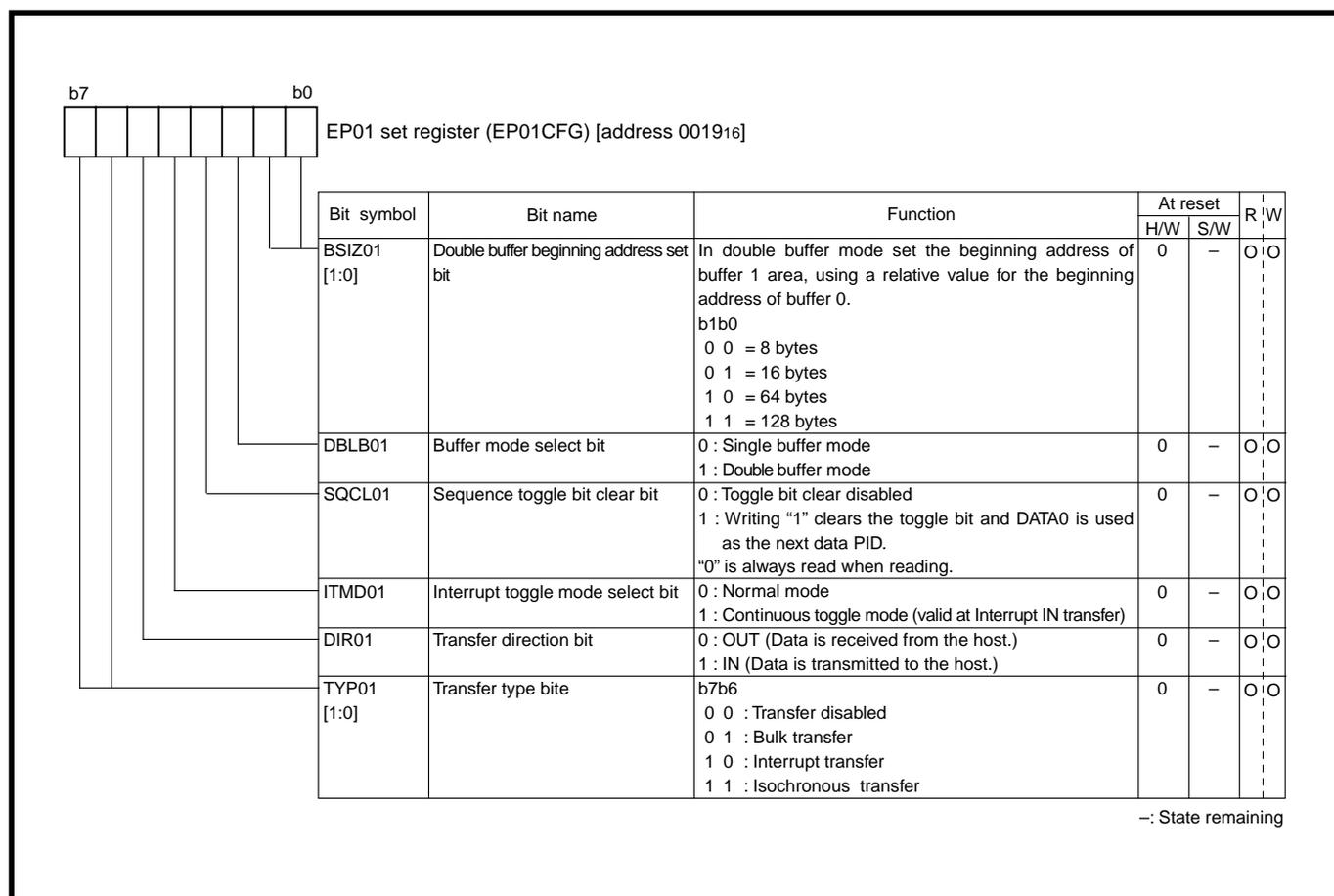


Fig. 3.4.13 Structure of EP01 set register

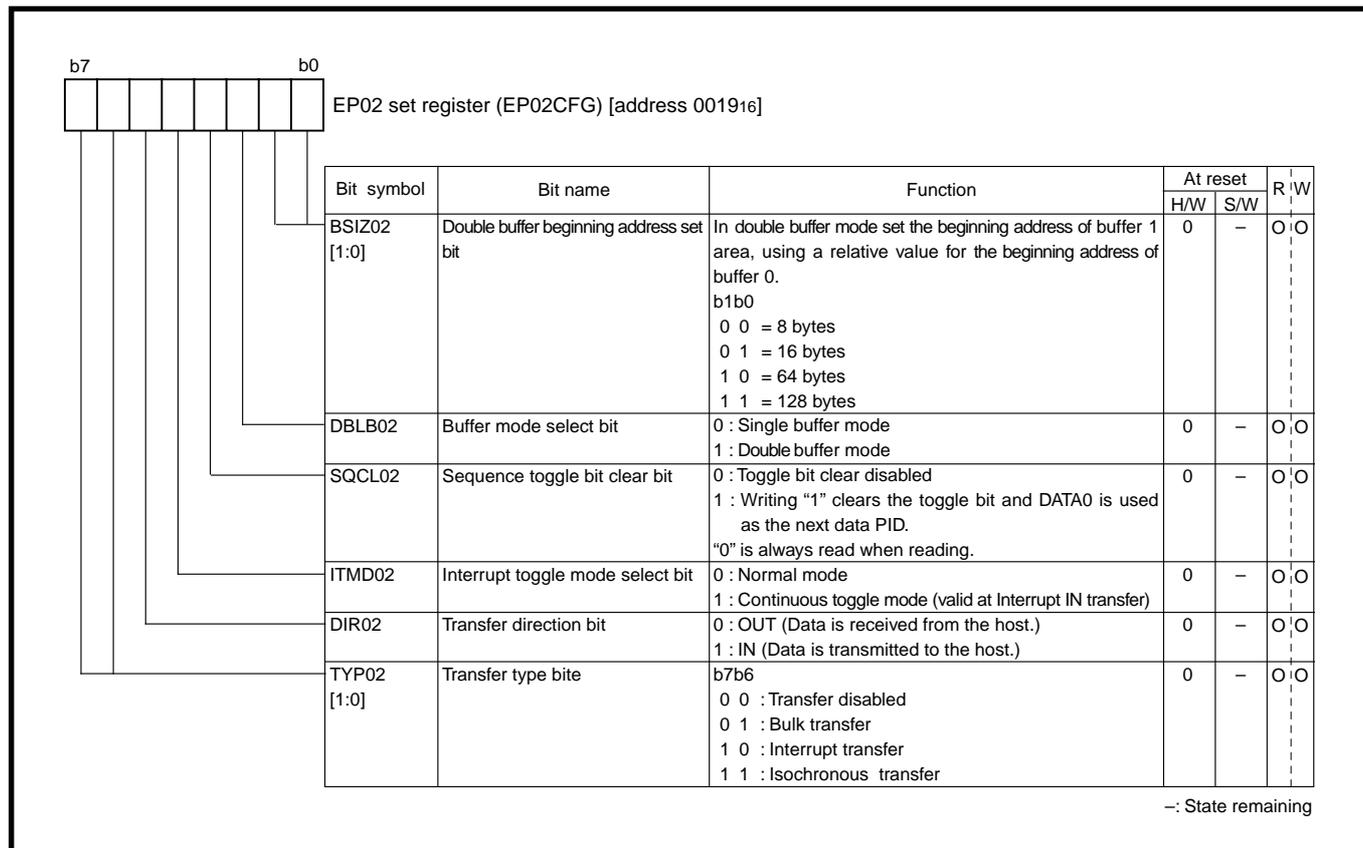


Fig. 3.4.14 Structure of EP02 set register

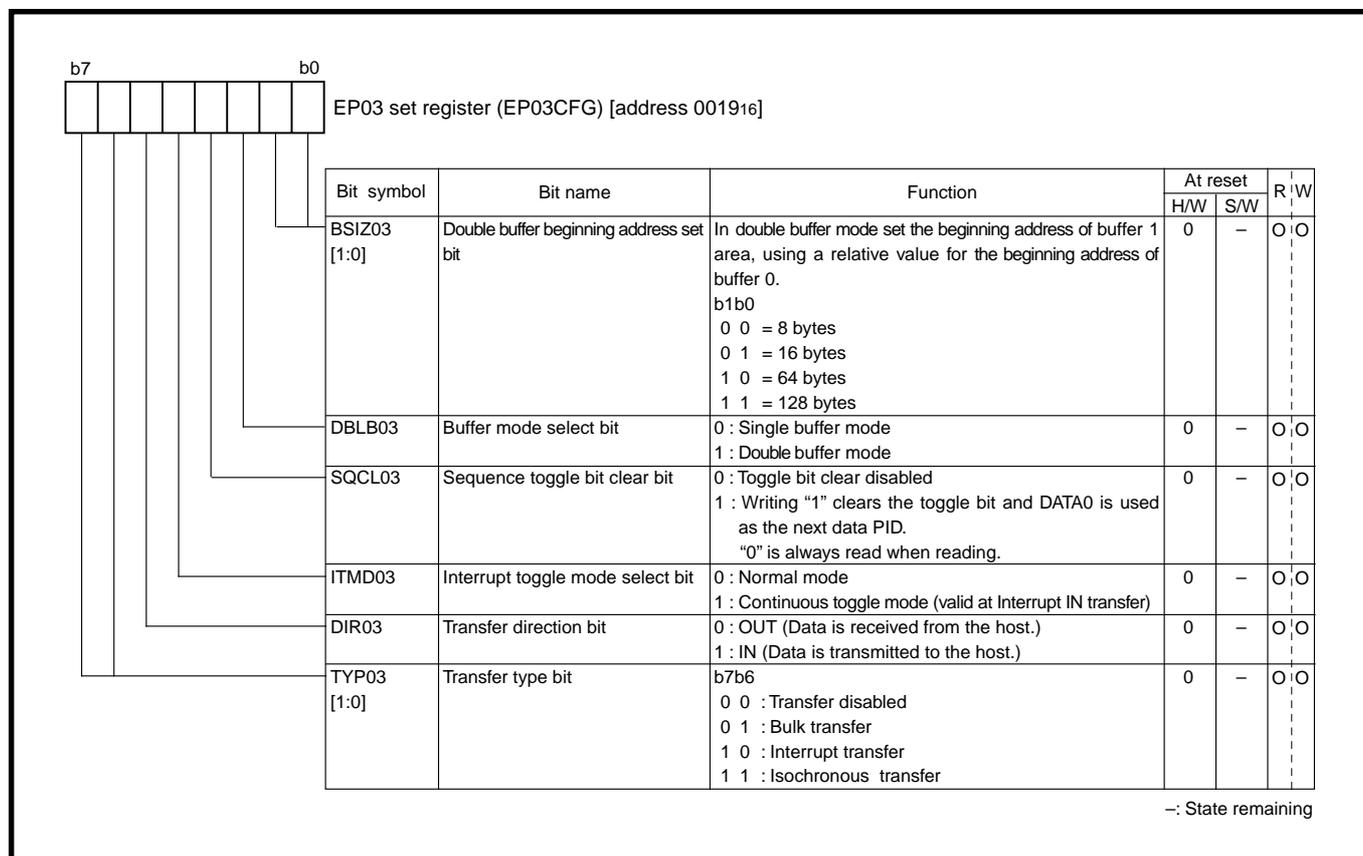


Fig. 3.4.15 Structure of EP03 set register

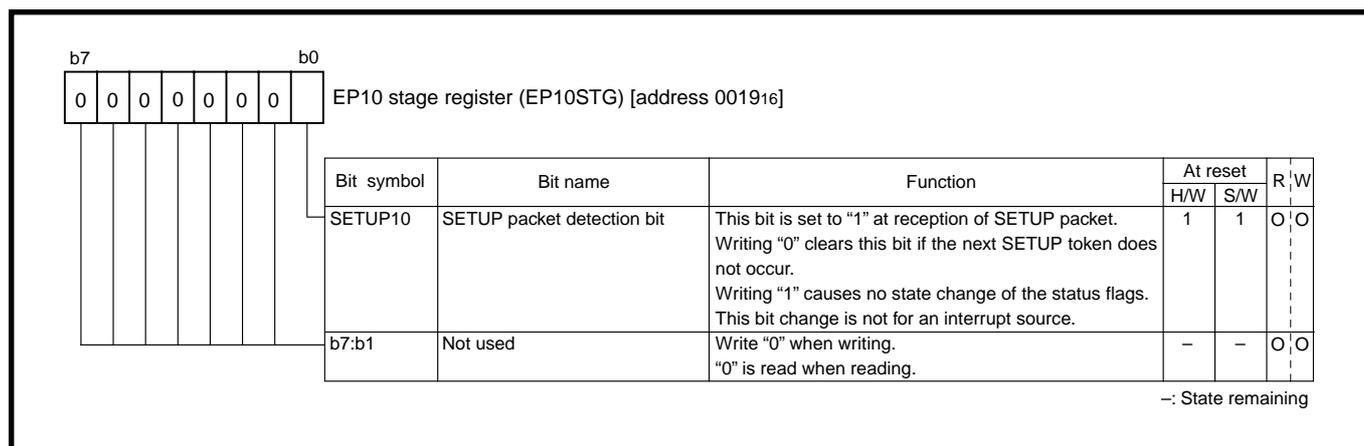


Fig. 3.4.16 Structure of EP10 stage register

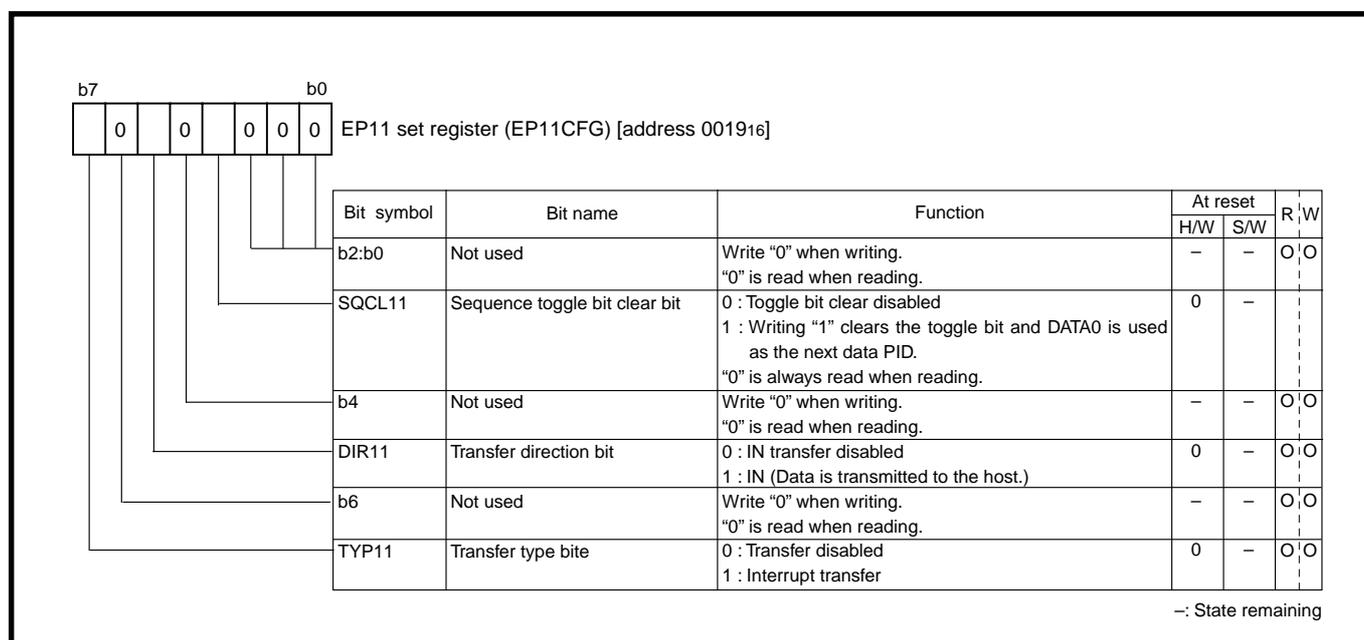


Fig. 3.4.17 Structure of EP11 set register

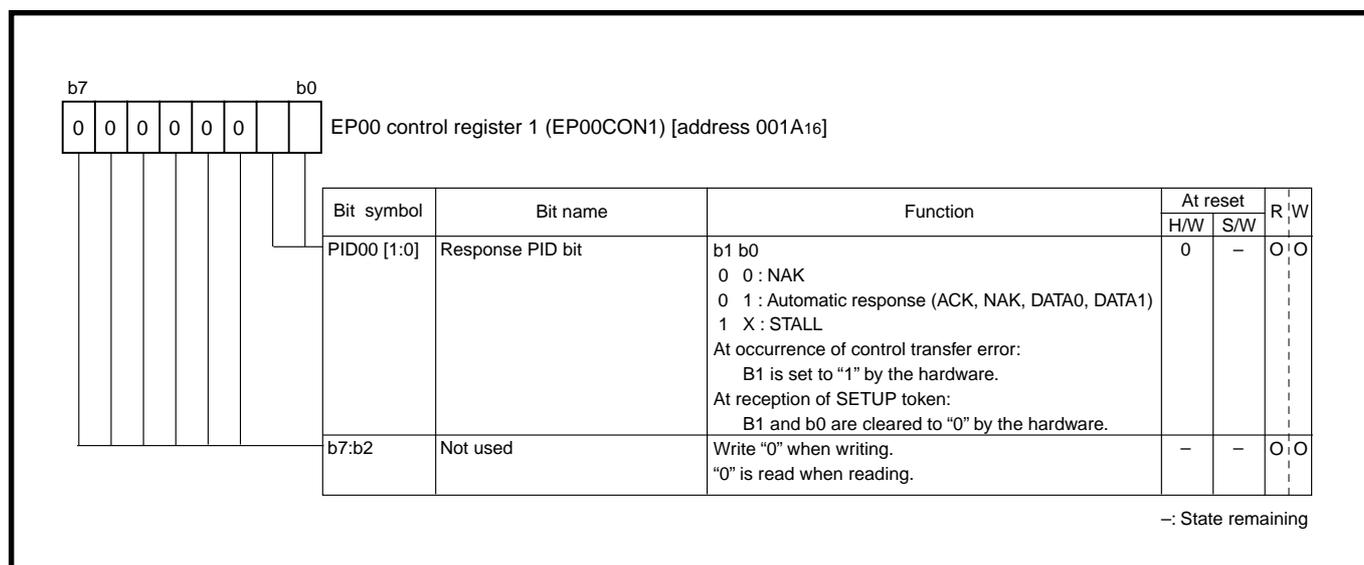


Fig. 3.4.18 Structure of EP00 control register 1

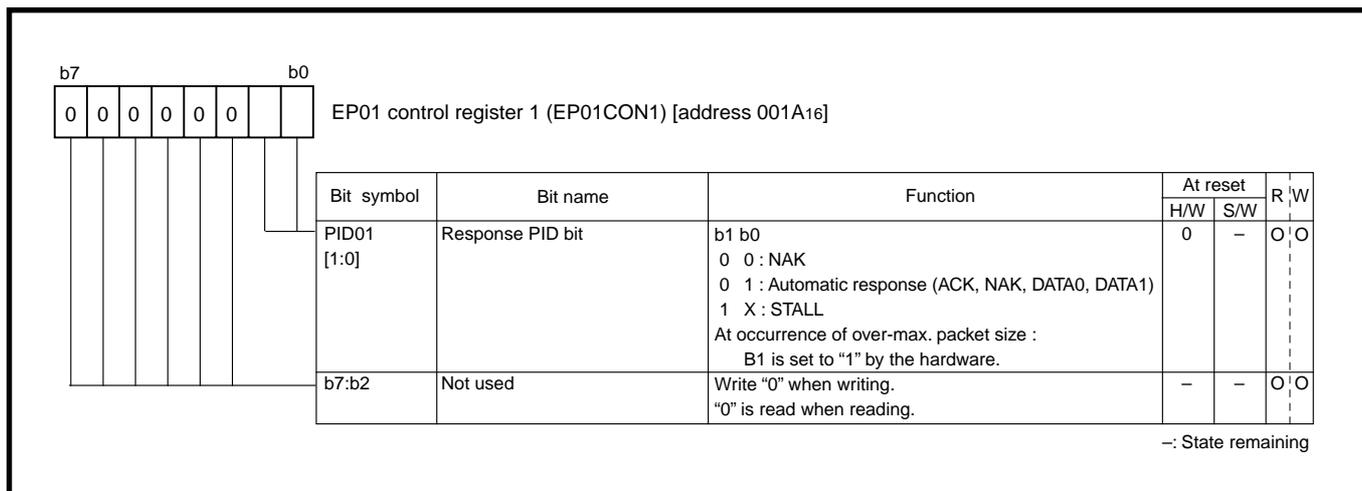


Fig. 3.4.19 Structure of EP01 control register 1

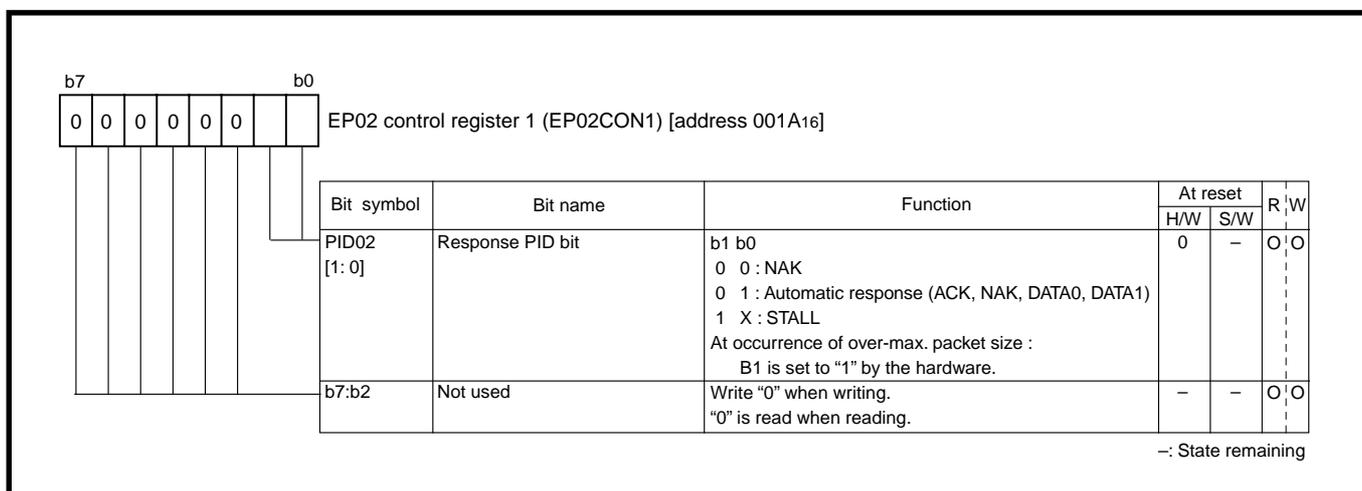


Fig. 3.4.20 Structure of EP02 control register 1

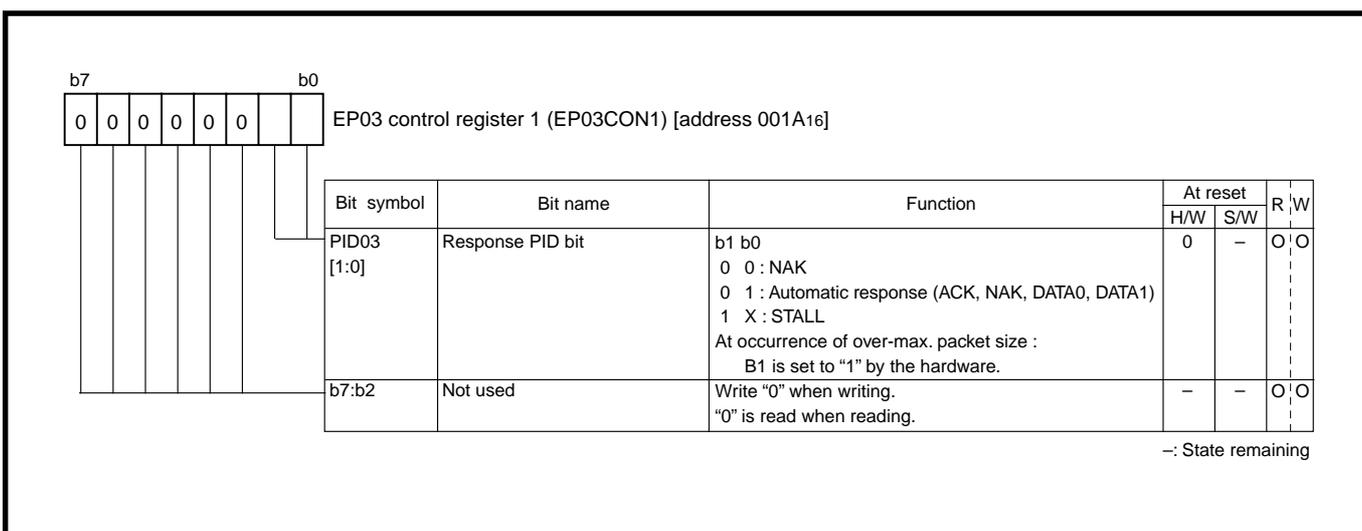


Fig. 3.4.21 Structure of EP03 control register 1

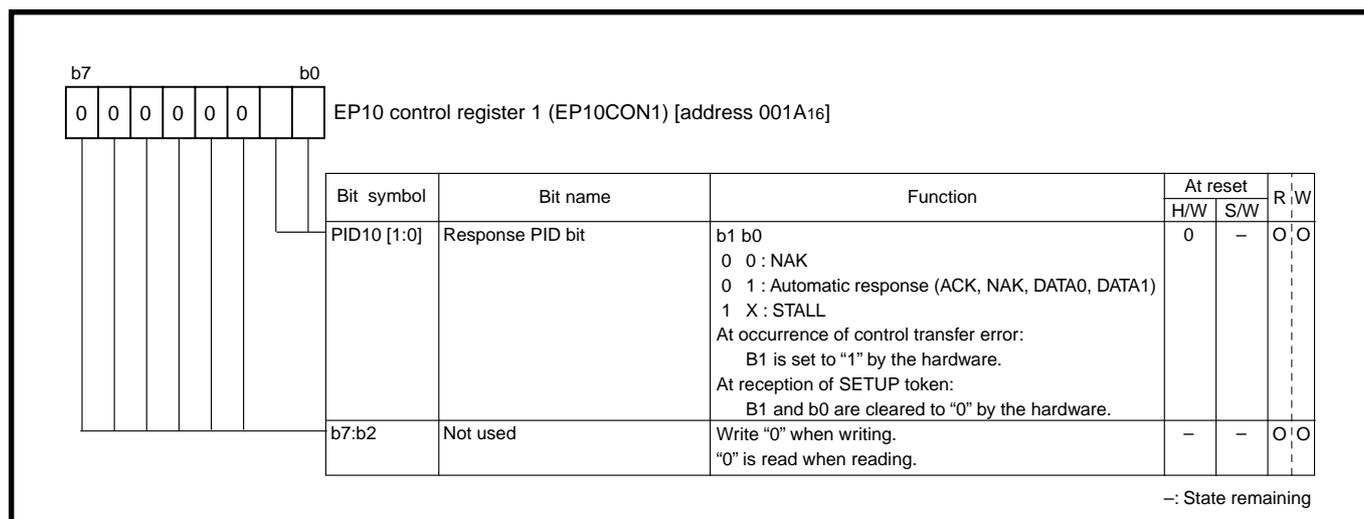


Fig. 3.4.22 Structure of EP10 control register 1

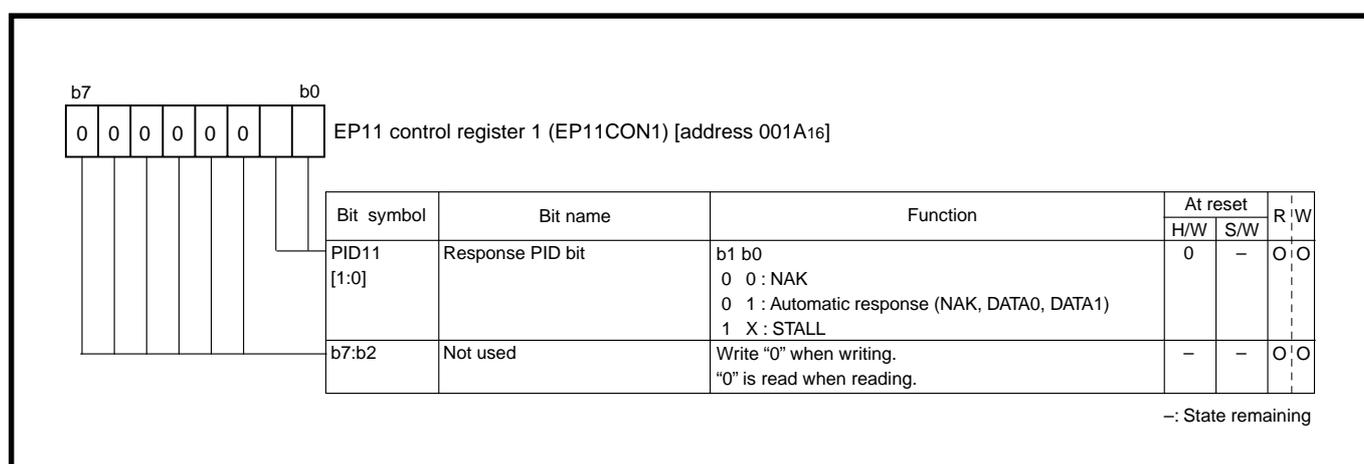


Fig. 3.4.23 Structure of EP11 control register 1

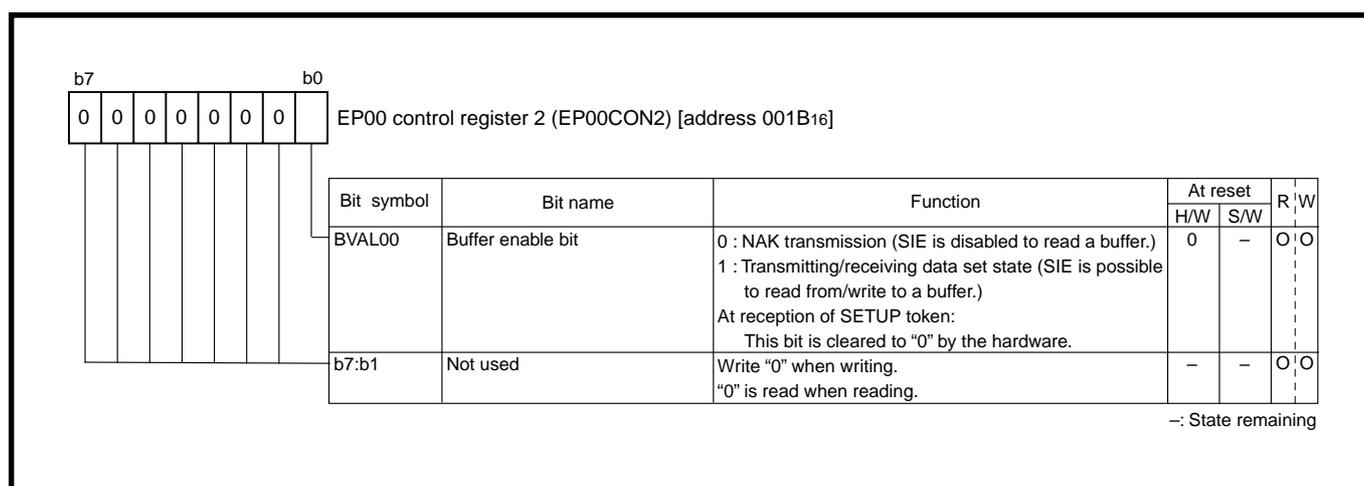


Fig. 3.4.24 Structure of EP00 control register 2

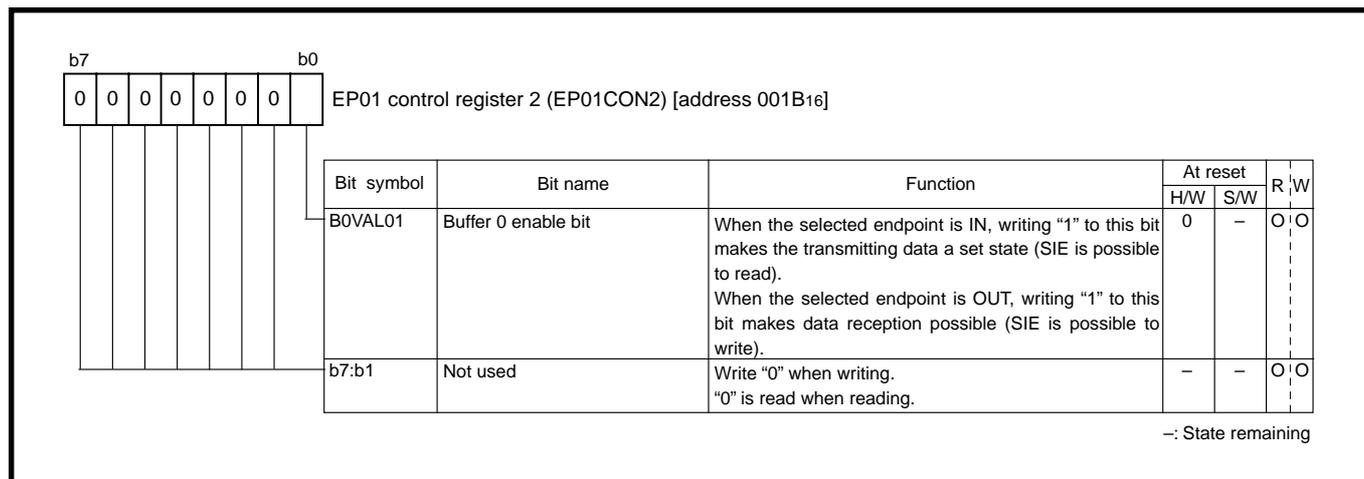


Fig. 3.4.25 Structure of EP01 control register 2

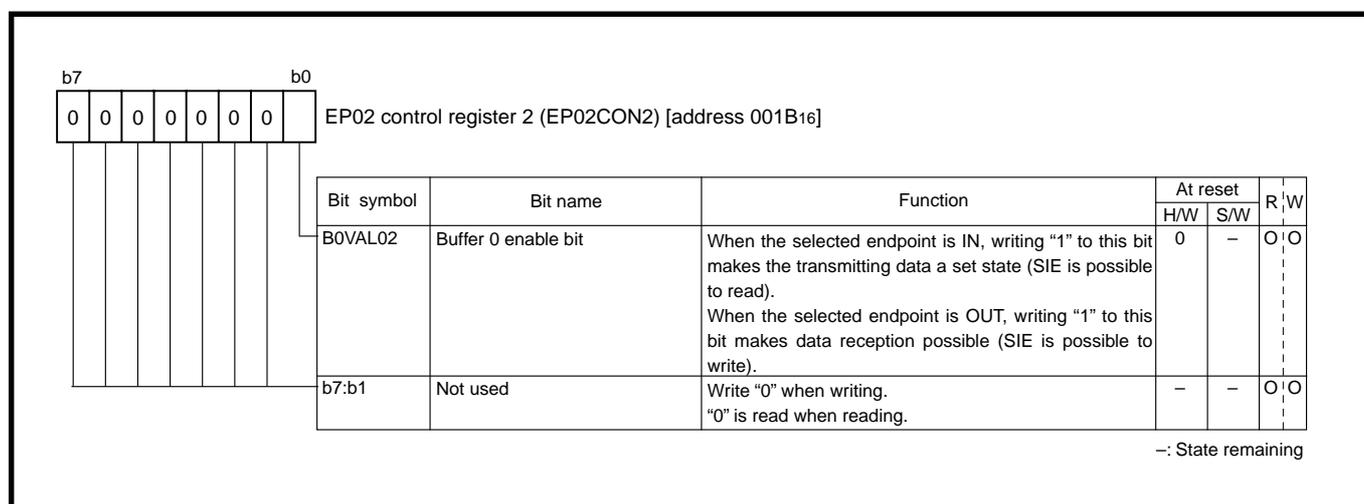


Fig. 3.4.26 Structure of EP02 control register 2

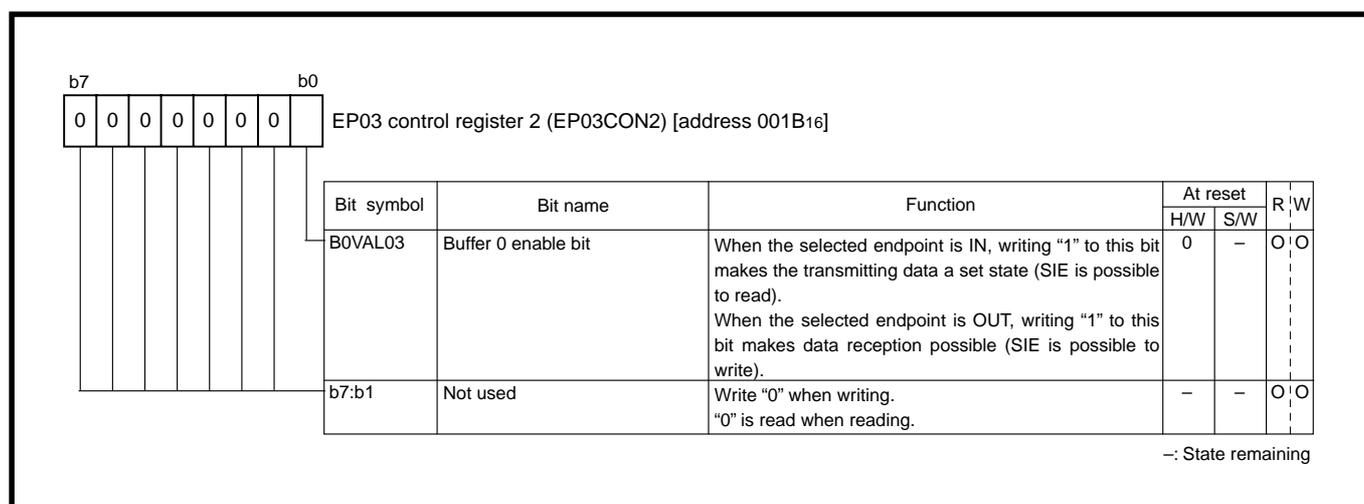


Fig. 3.4.27 Structure of EP03 control register 2

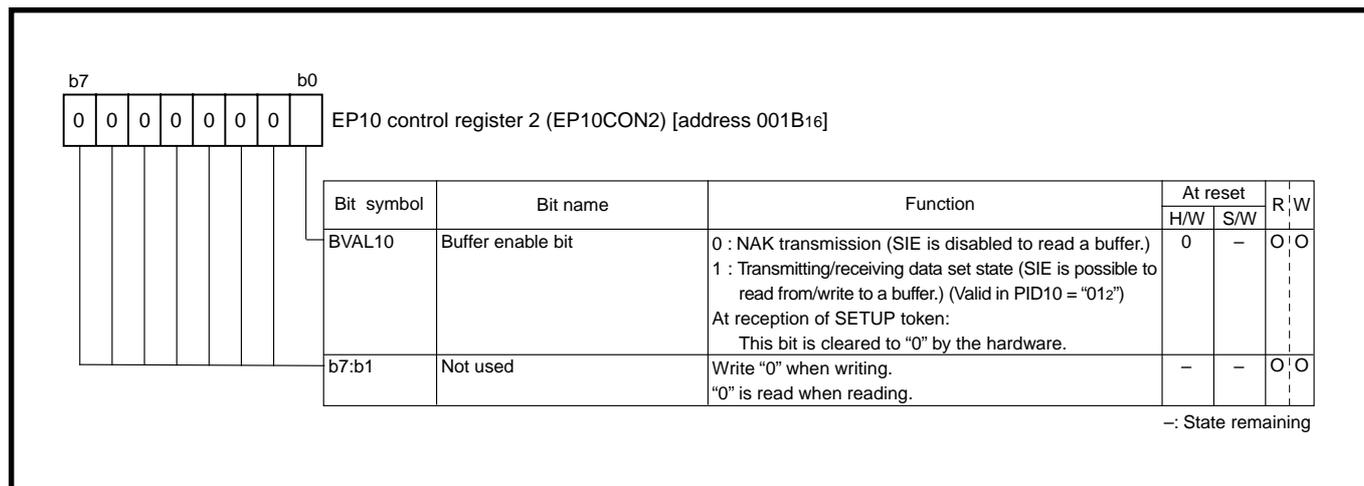


Fig. 3.4.28 Structure of EP10 control register 2

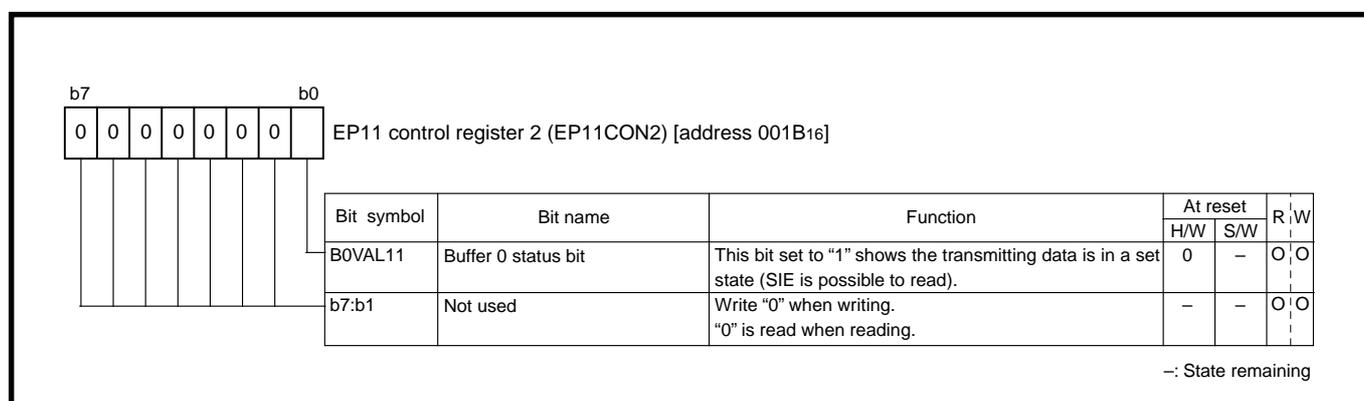


Fig. 3.4.29 Structure of EP11 control register 2

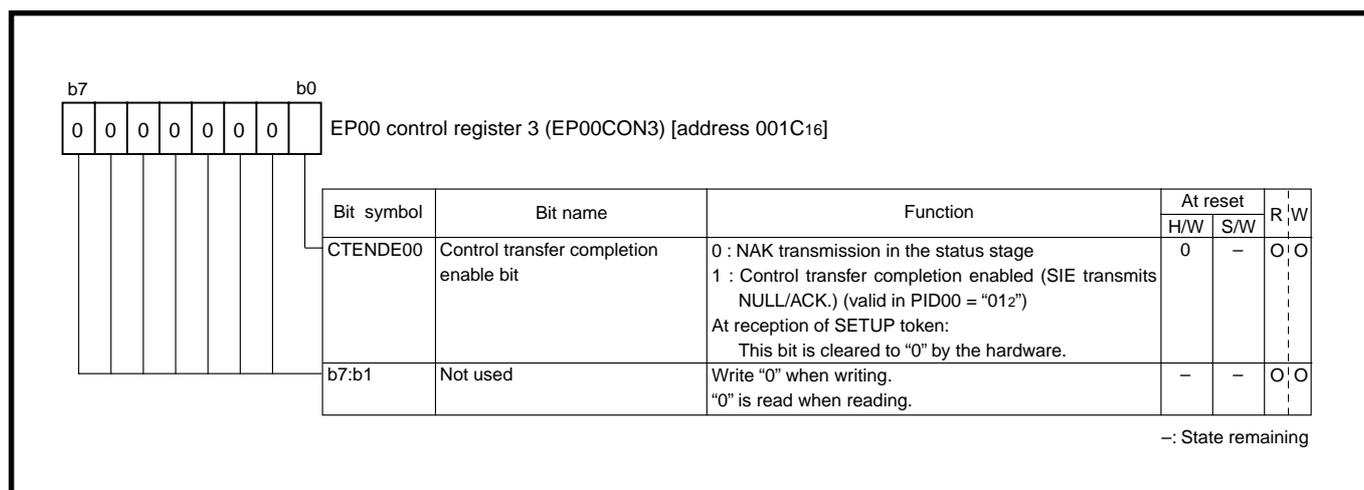


Fig. 3.4.30 Structure of EP00 control register 3

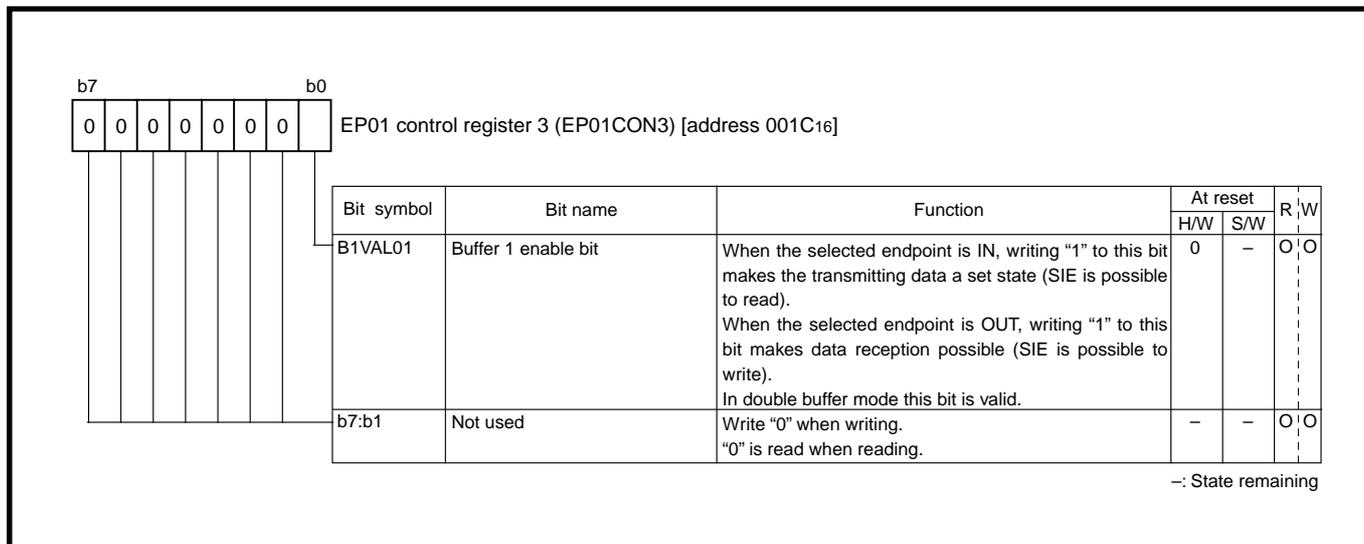


Fig. 3.4.31 Structure of EP01 control register 3

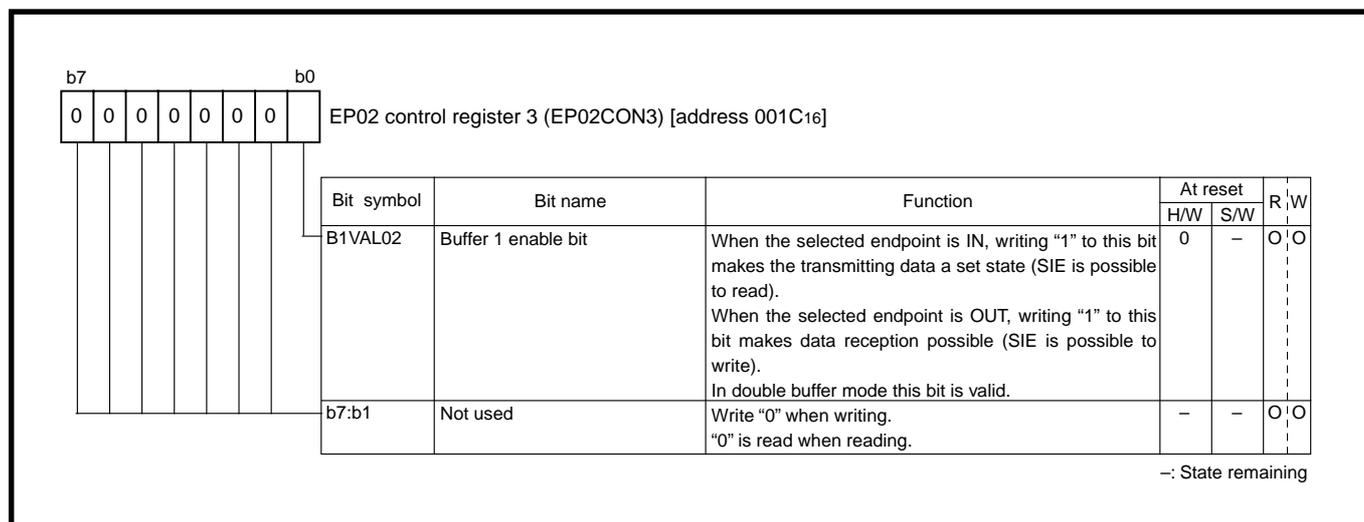


Fig. 3.4.32 Structure of EP02 control register 3

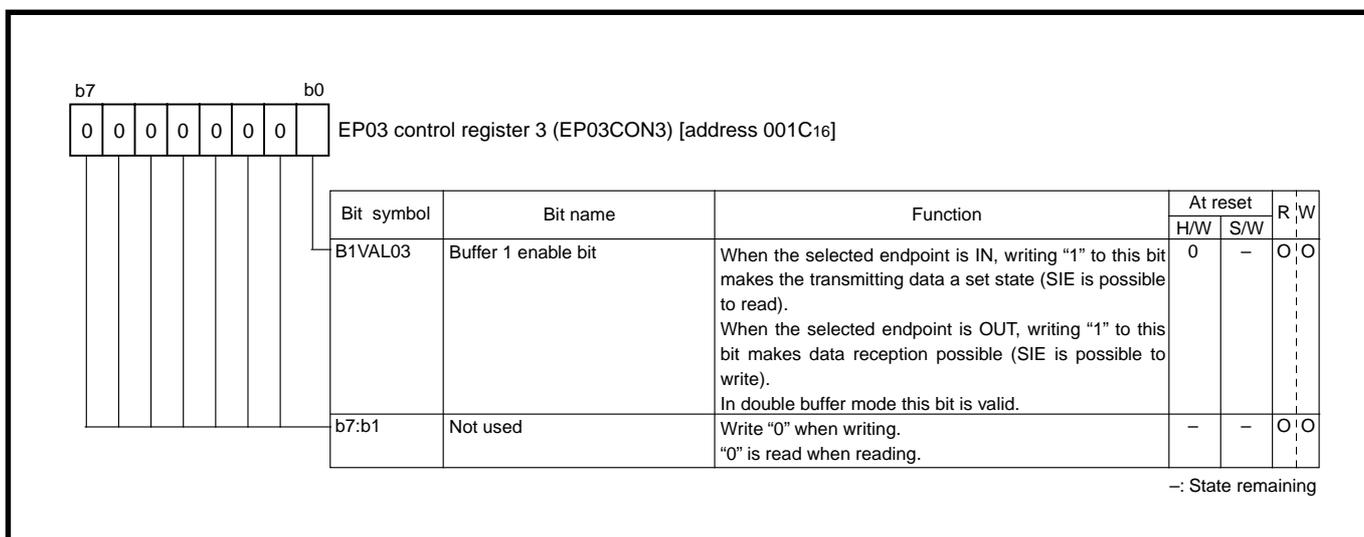


Fig. 3.4.33 Structure of EP03 control register 3

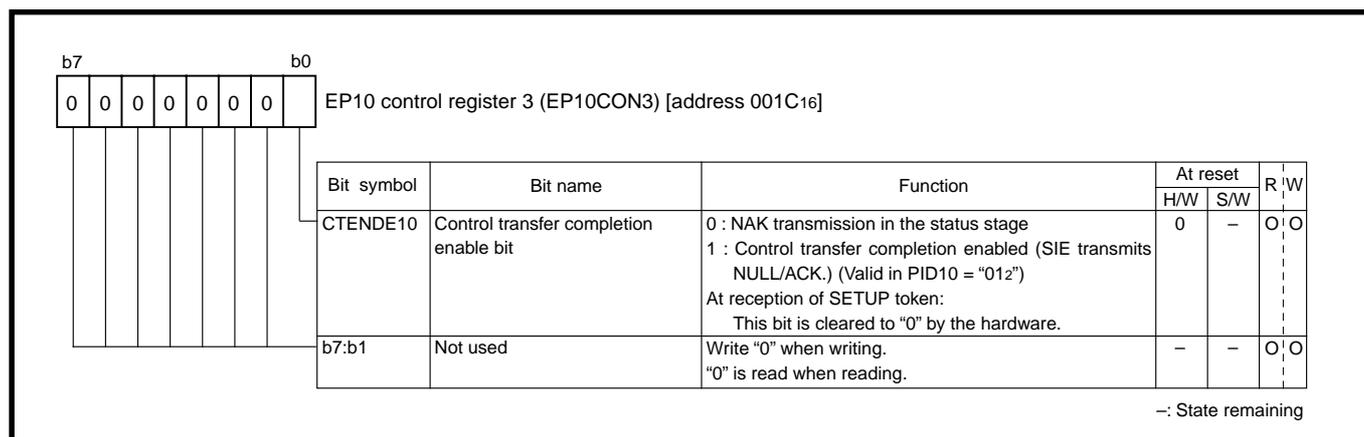


Fig. 3.4.34 Structure of EP10 control register 3

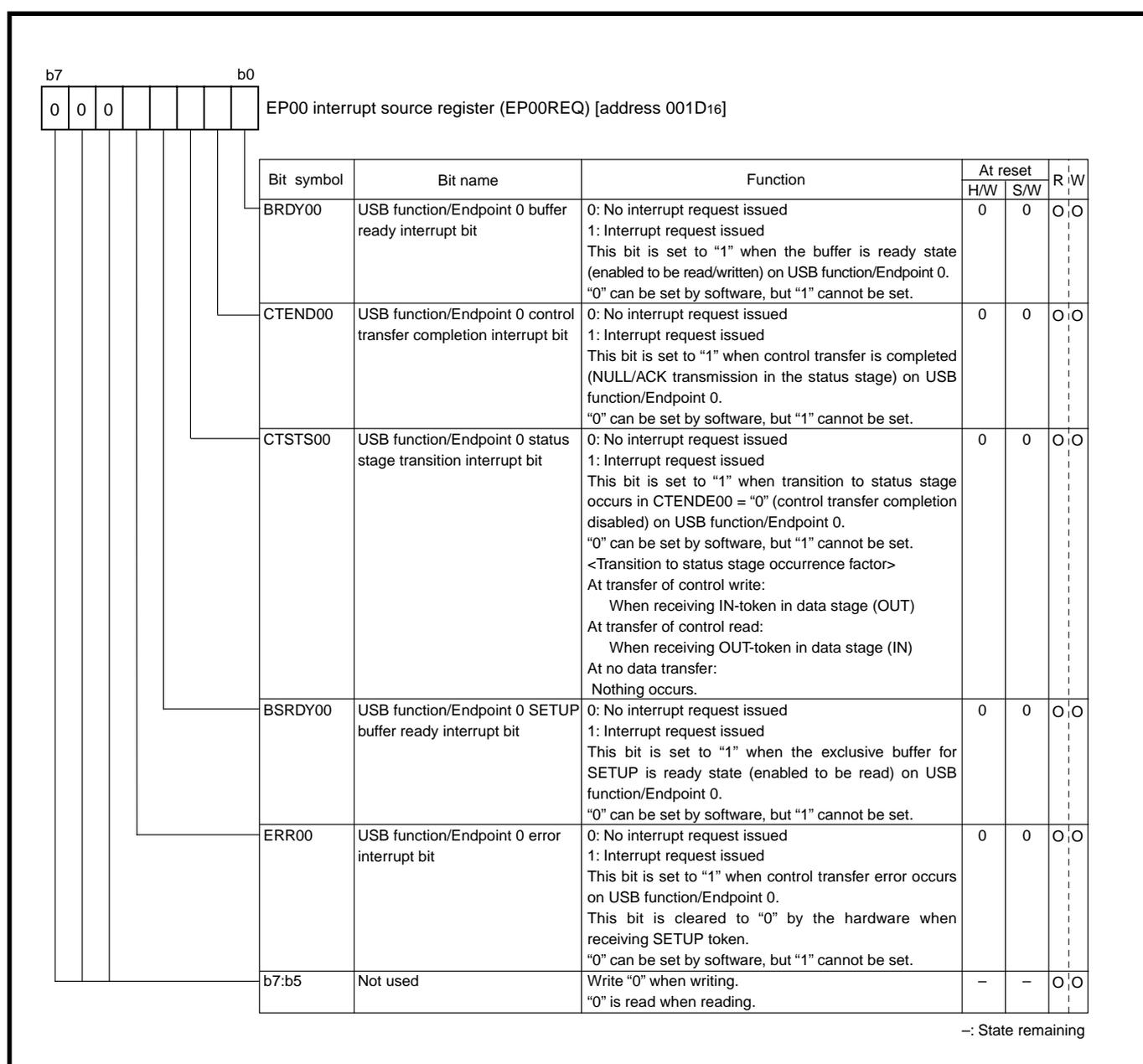


Fig. 3.4.35 Structure of EP00 interrupt source register

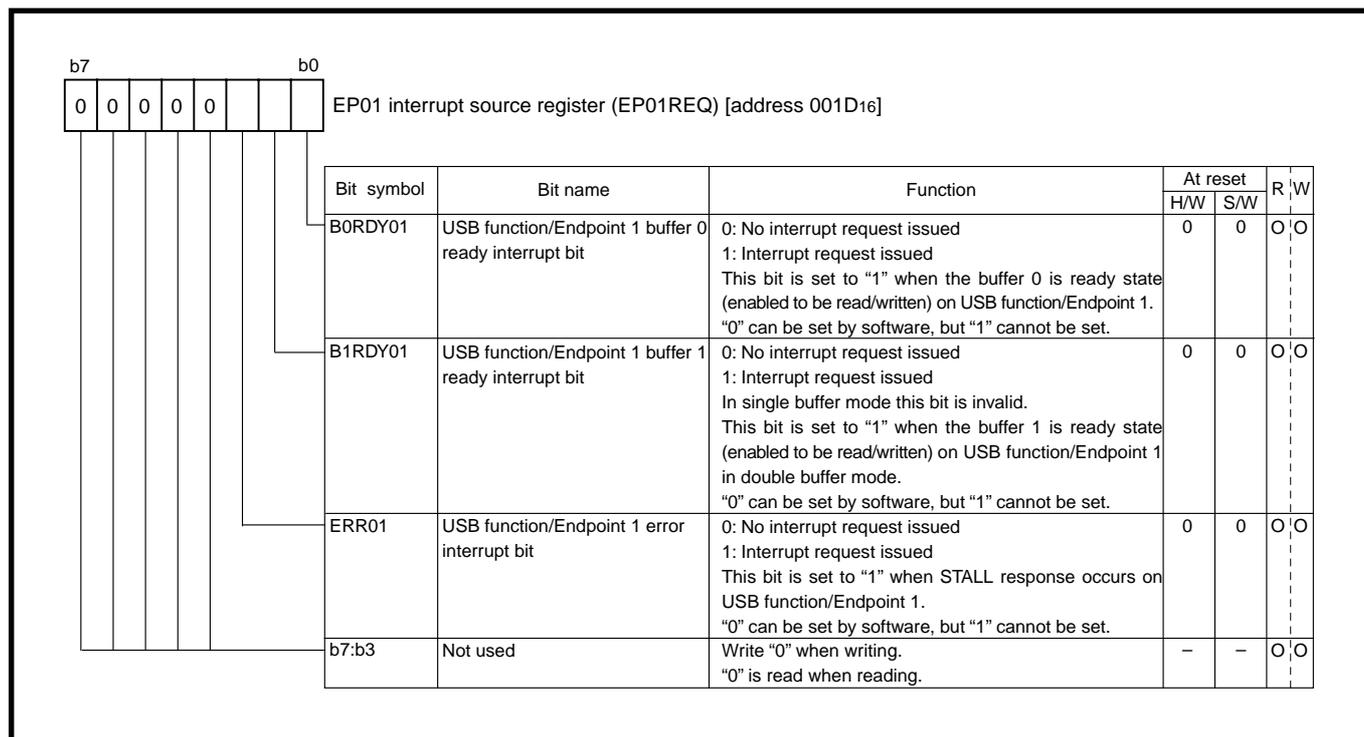


Fig. 3.4.36 Structure of EP01 interrupt source register

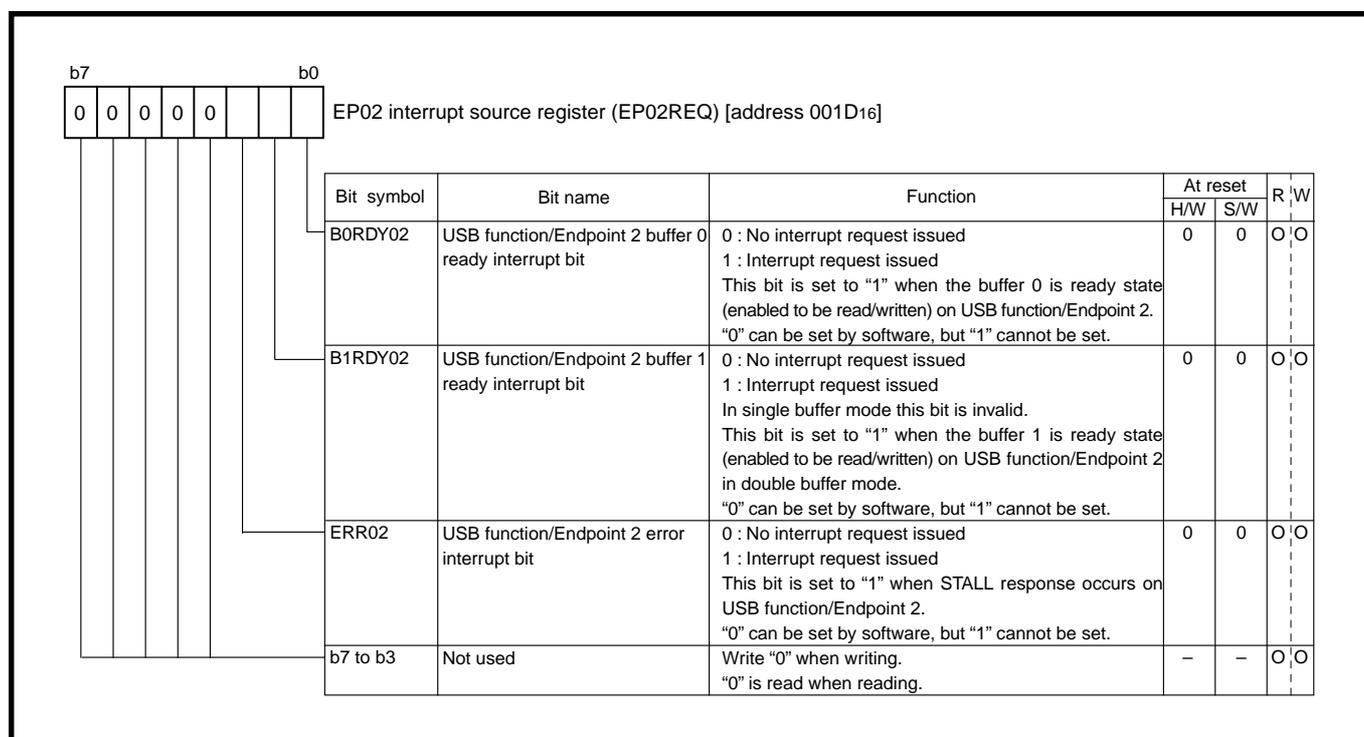


Fig. 3.4.37 Structure of EP02 interrupt source register

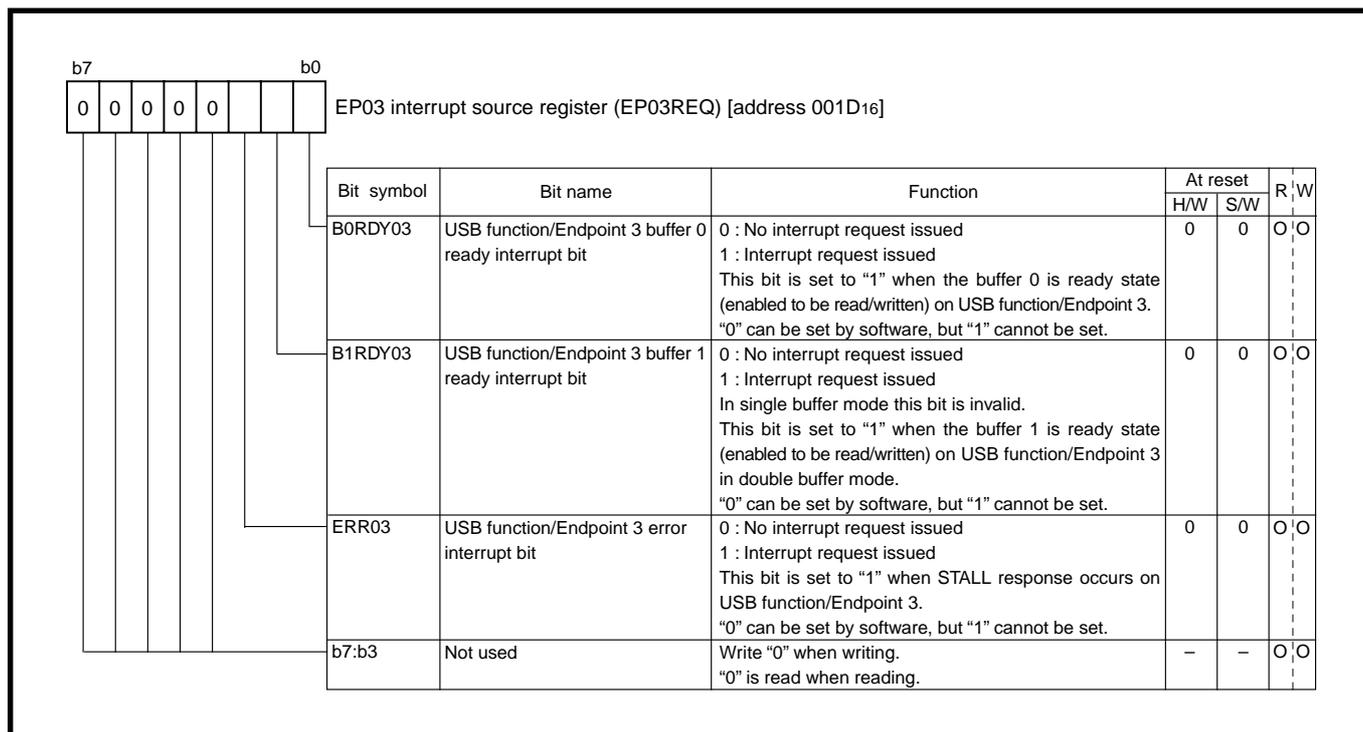


Fig. 3.4.38 Structure of EP03 interrupt source register

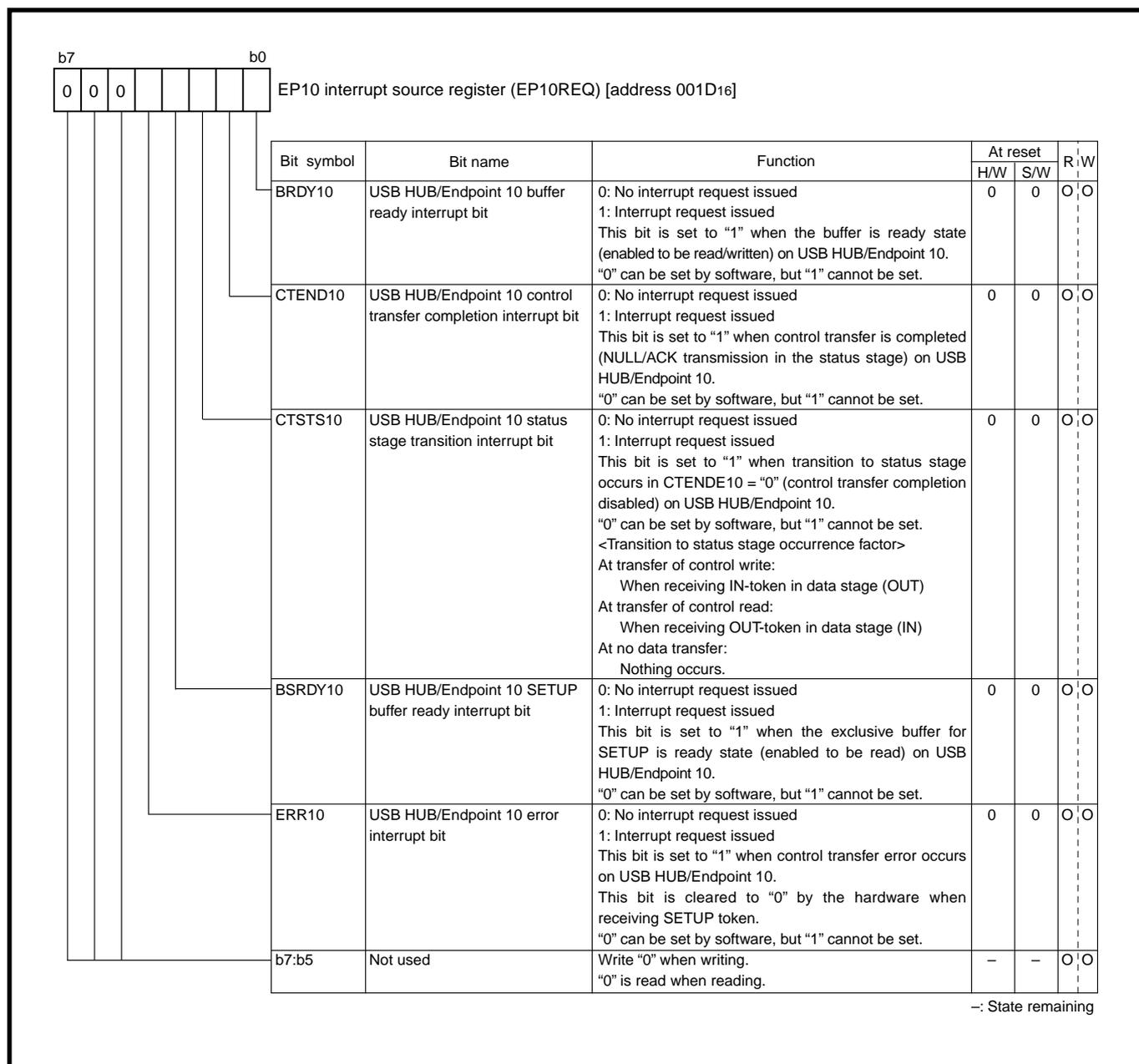


Fig. 3.4.39 Structure of EP10 interrupt source register

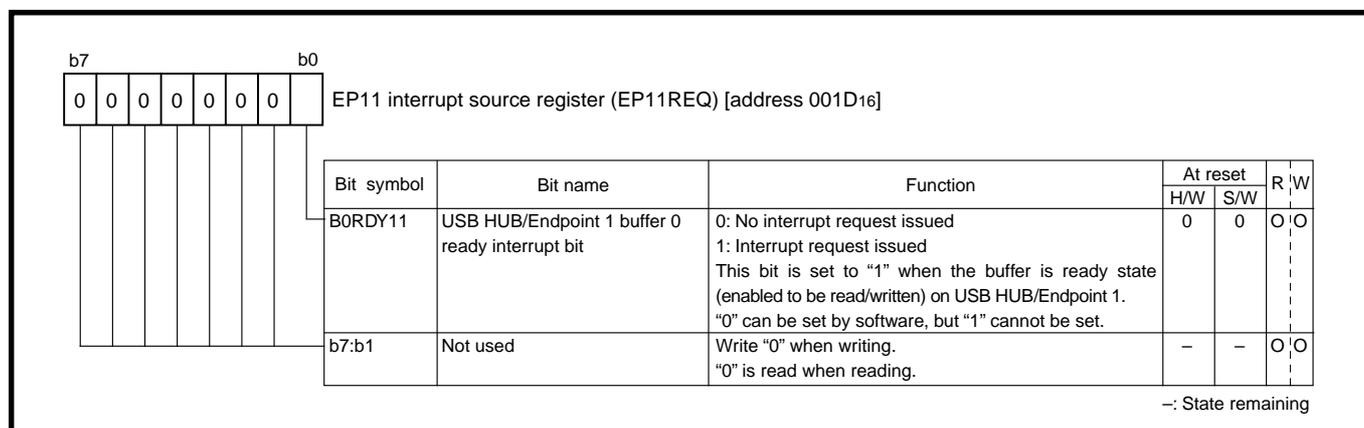


Fig. 3.4.40 Structure of EP11 interrupt source register

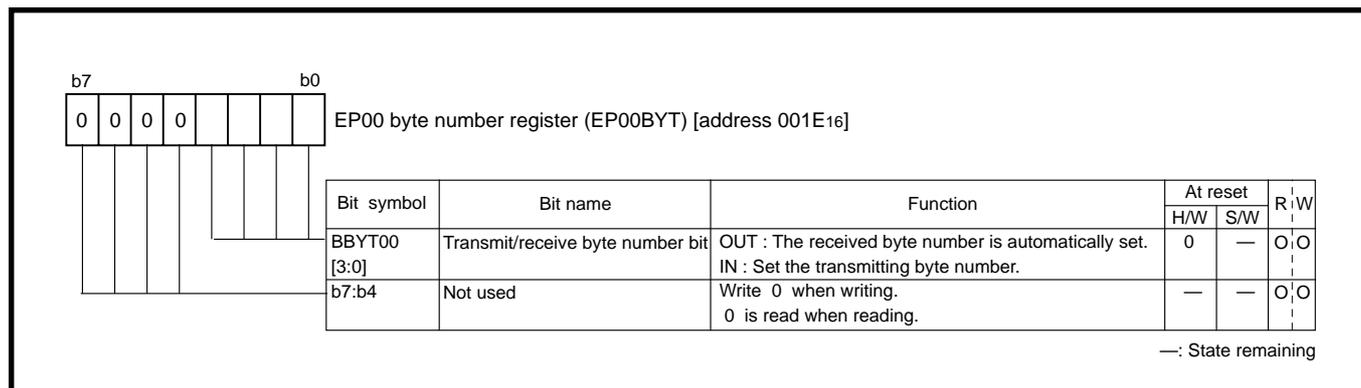


Fig. 3.4.41 Structure of EP00 byte number register

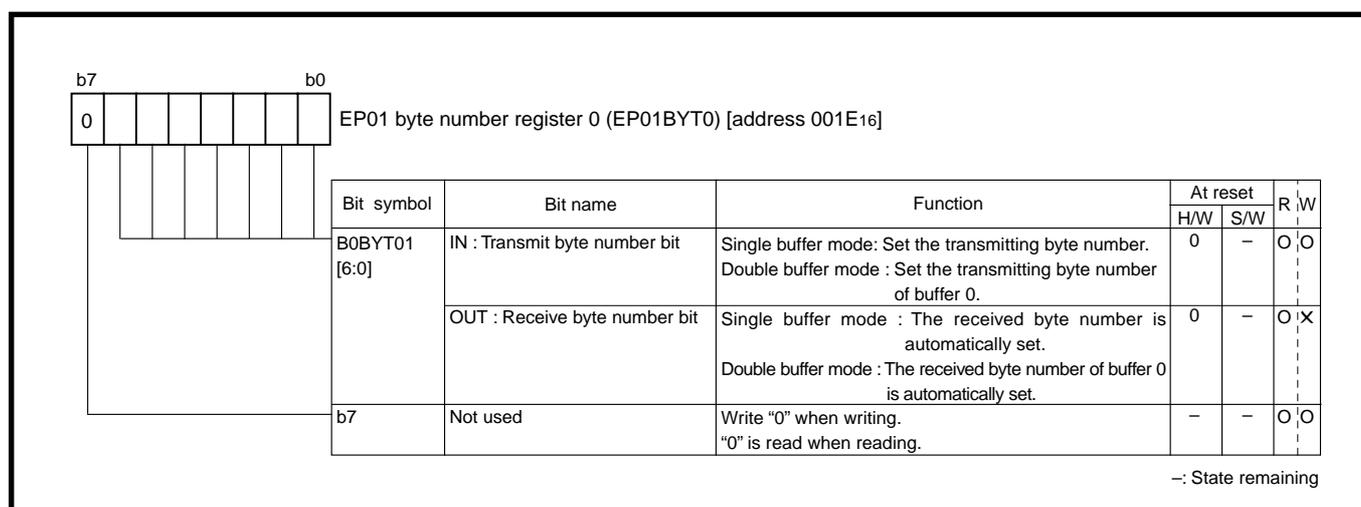


Fig. 3.4.42 Structure of EP01 byte number register 0

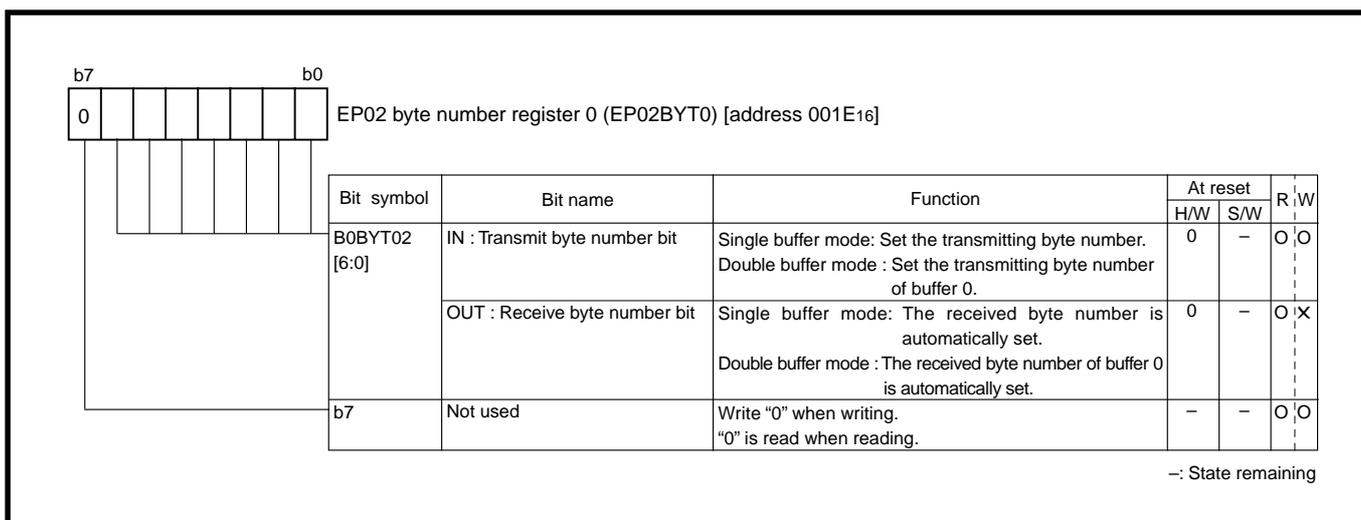


Fig. 3.4.43 Structure of EP02 byte number register 0

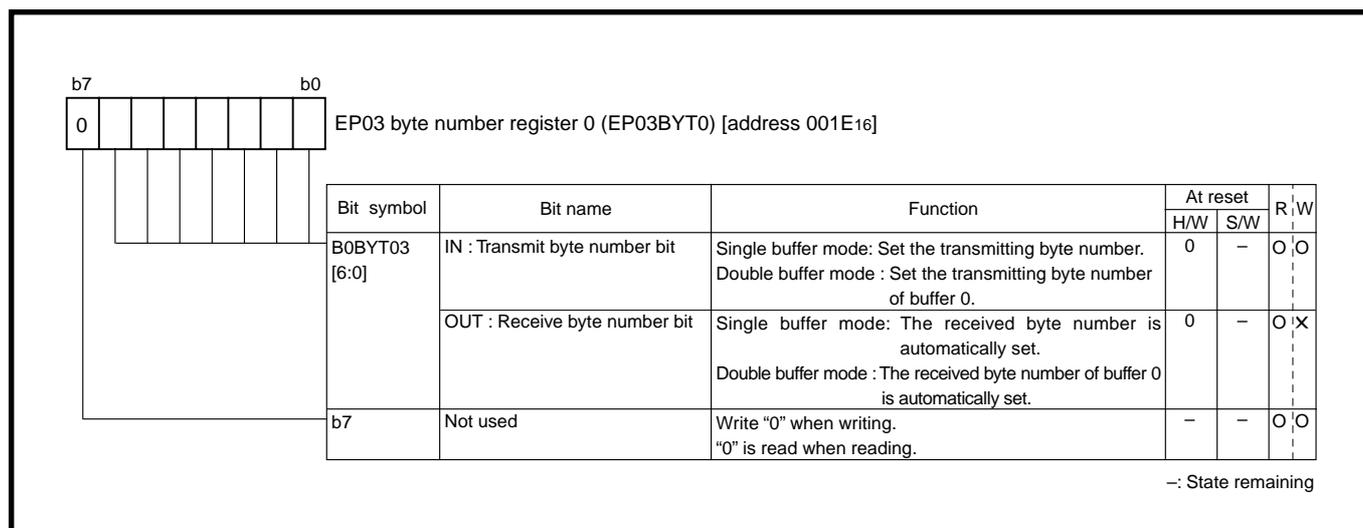


Fig. 3.4.44 Structure of EP03 byte number register 0

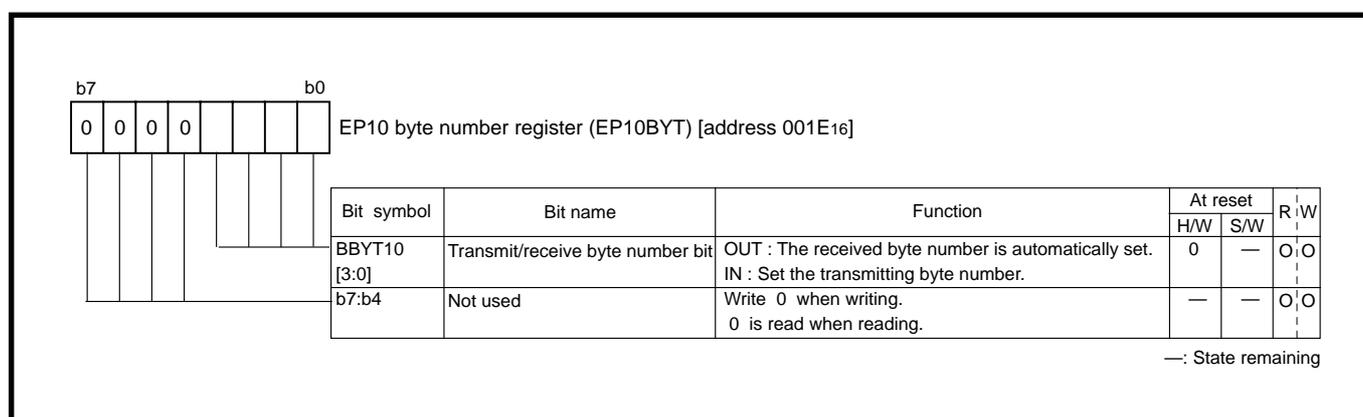


Fig. 3.4.45 Structure of EP10 byte number register

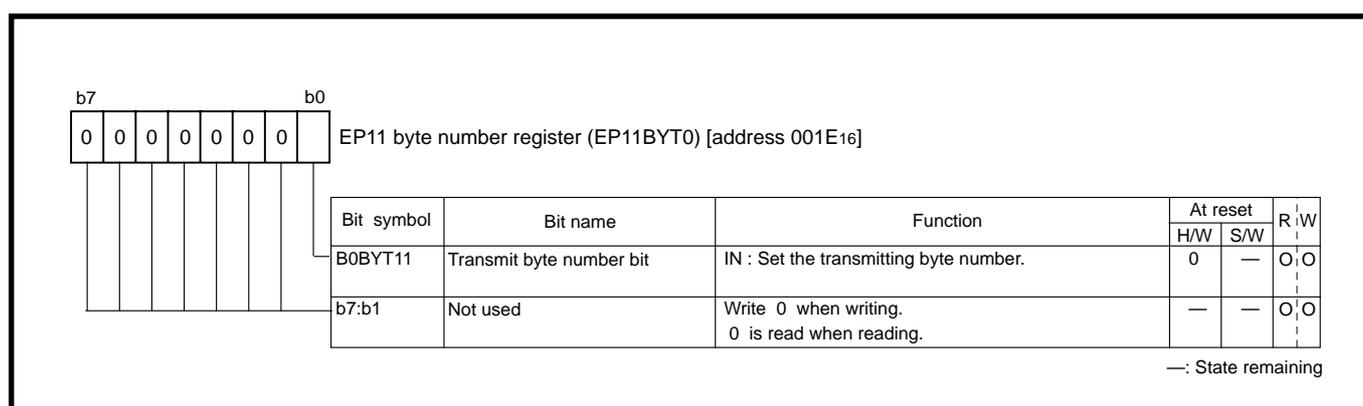


Fig. 3.4.46 Structure of EP11 byte number register 0

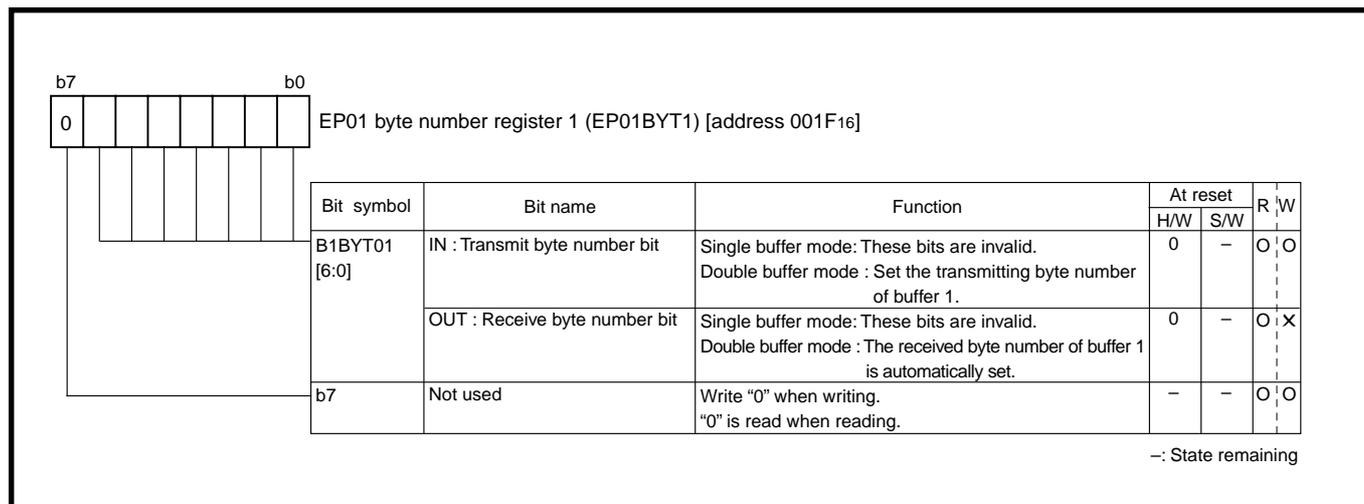


Fig. 3.4.47 Structure of EP01 byte number register 1

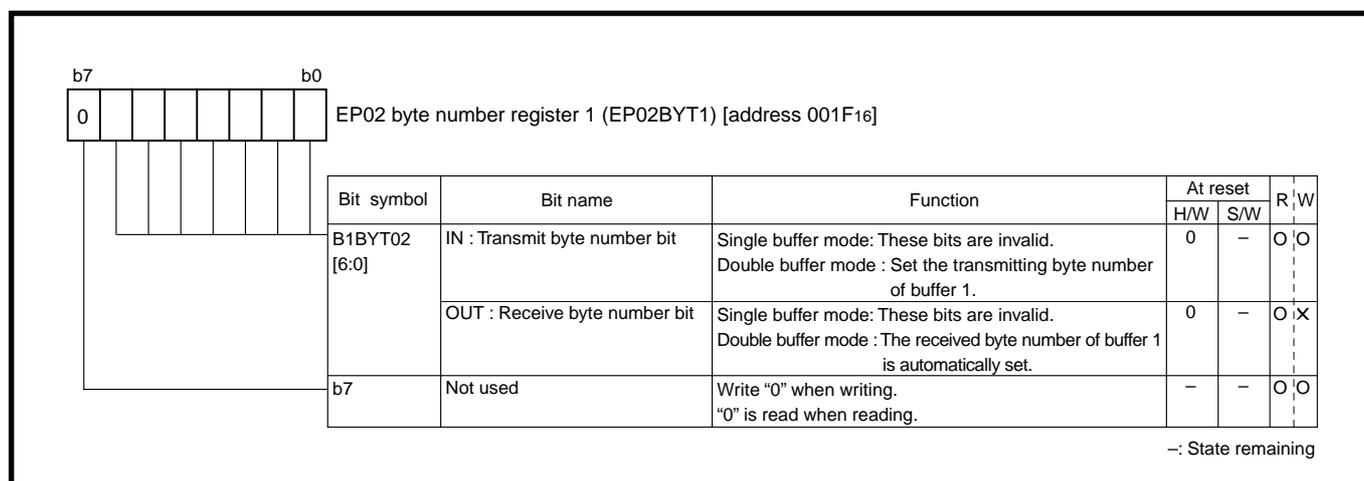


Fig. 3.4.48 Structure of EP02 byte number register 1

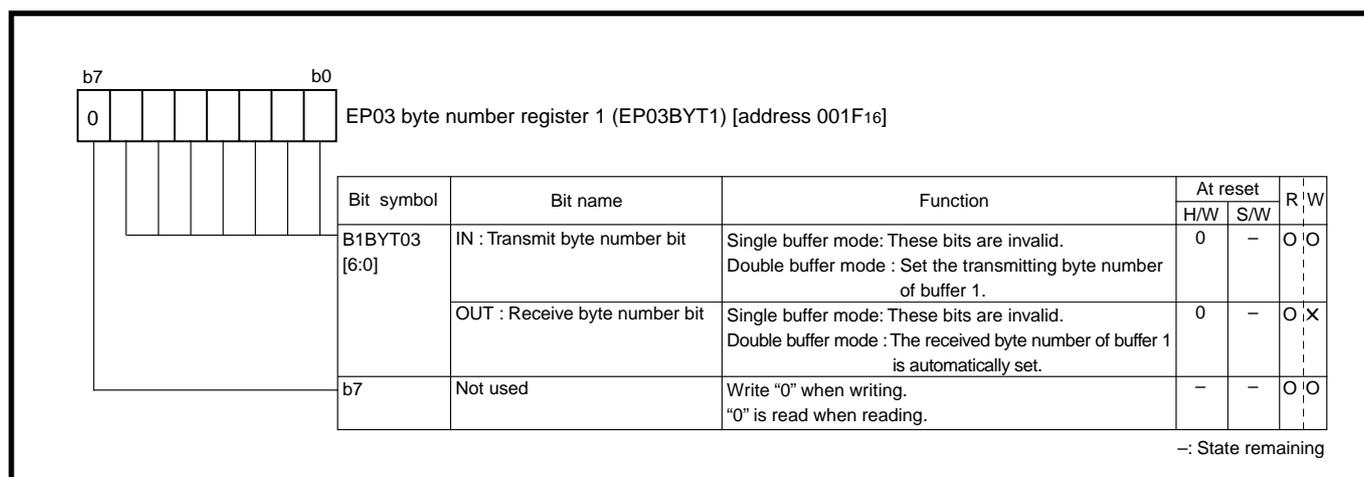


Fig. 3.4.49 Structure of EP03 byte number register 1

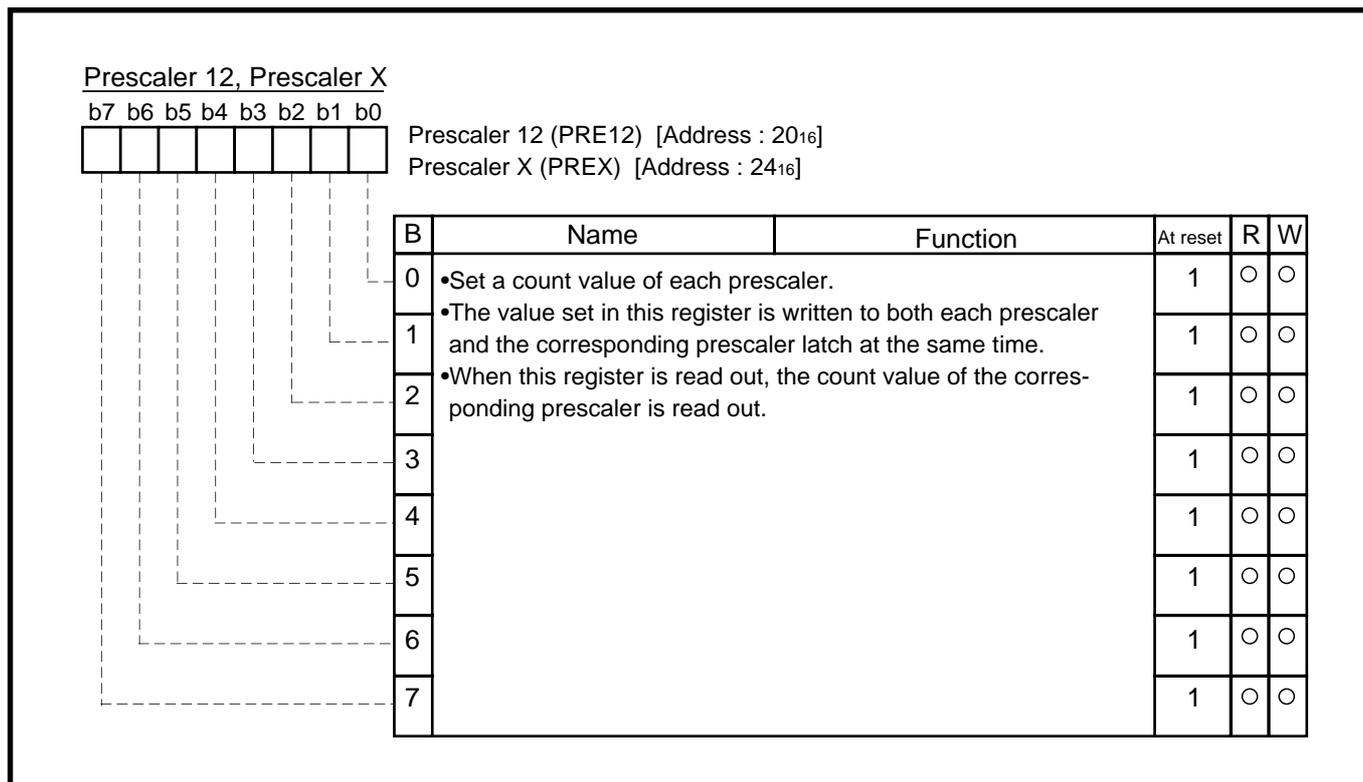


Fig. 3.4.50 Structure of Prescaler12, Prescaler X

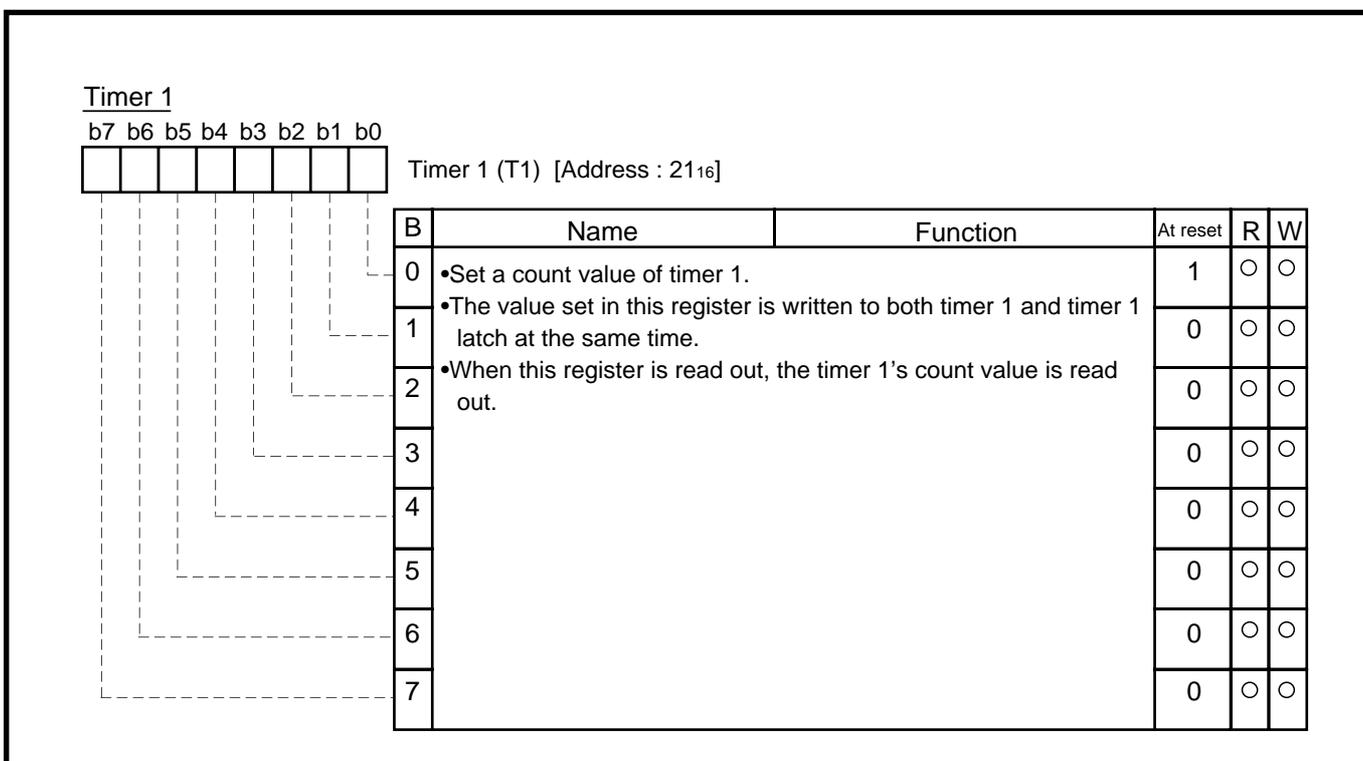


Fig. 3.4.51 Structure of Timer 1

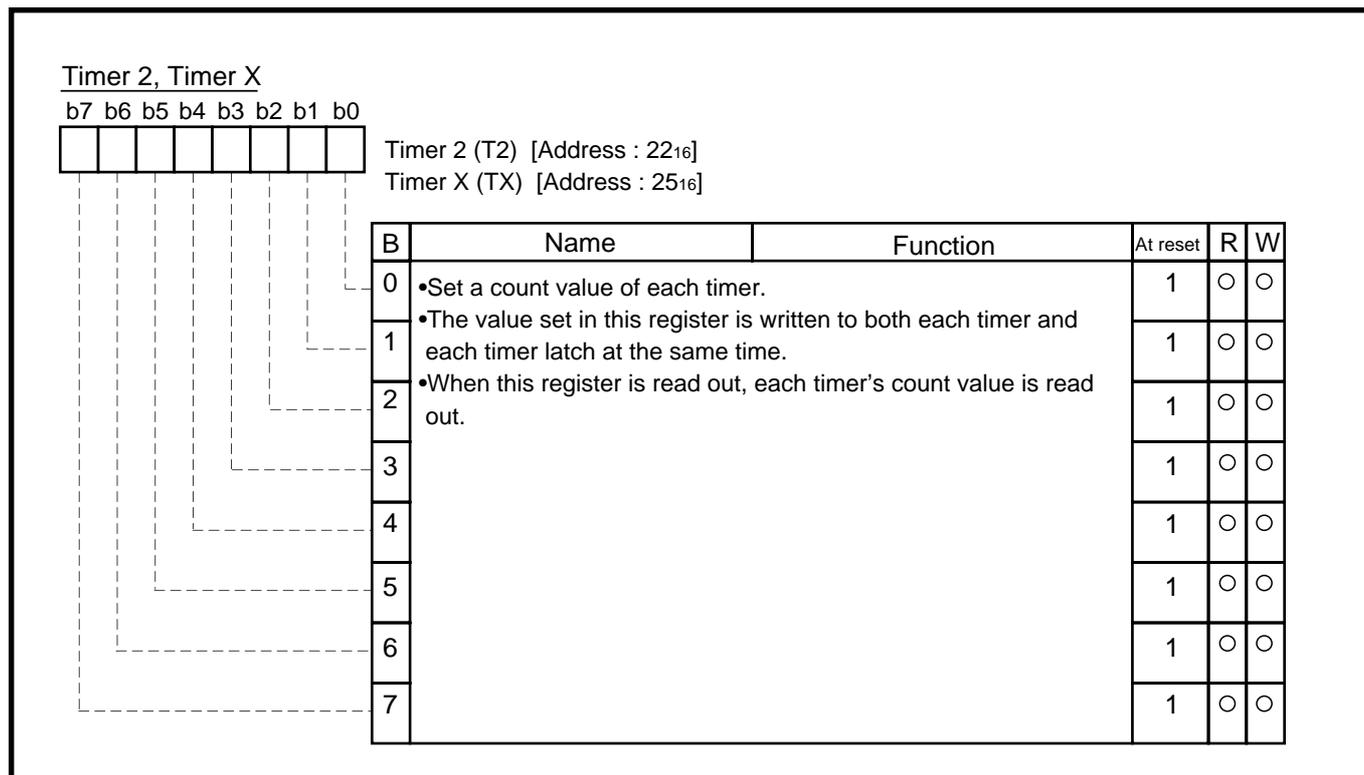


Fig. 3.4.52 Structure of Timer 2, Timer X

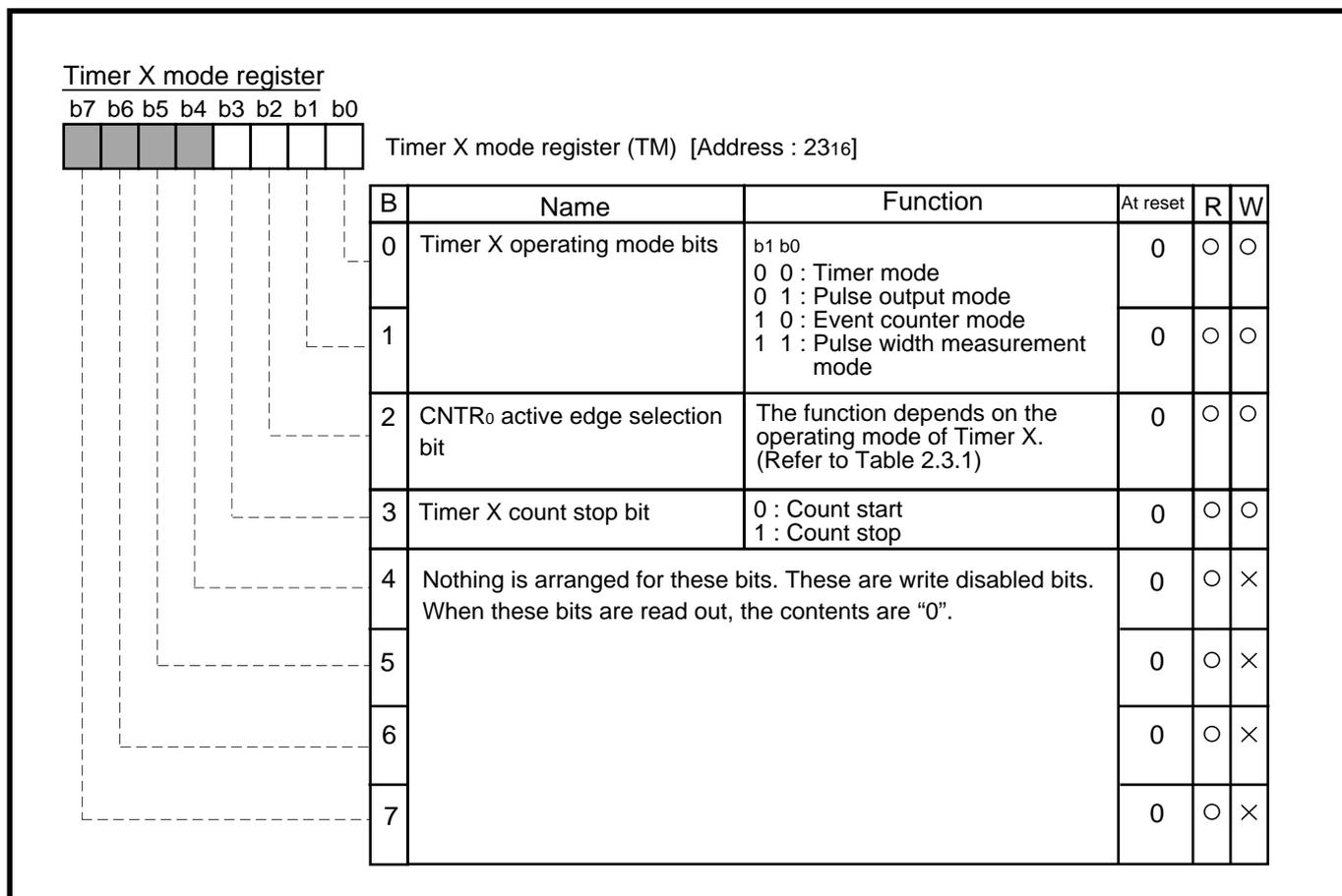


Fig. 3.4.53 Structure of Timer X mode register

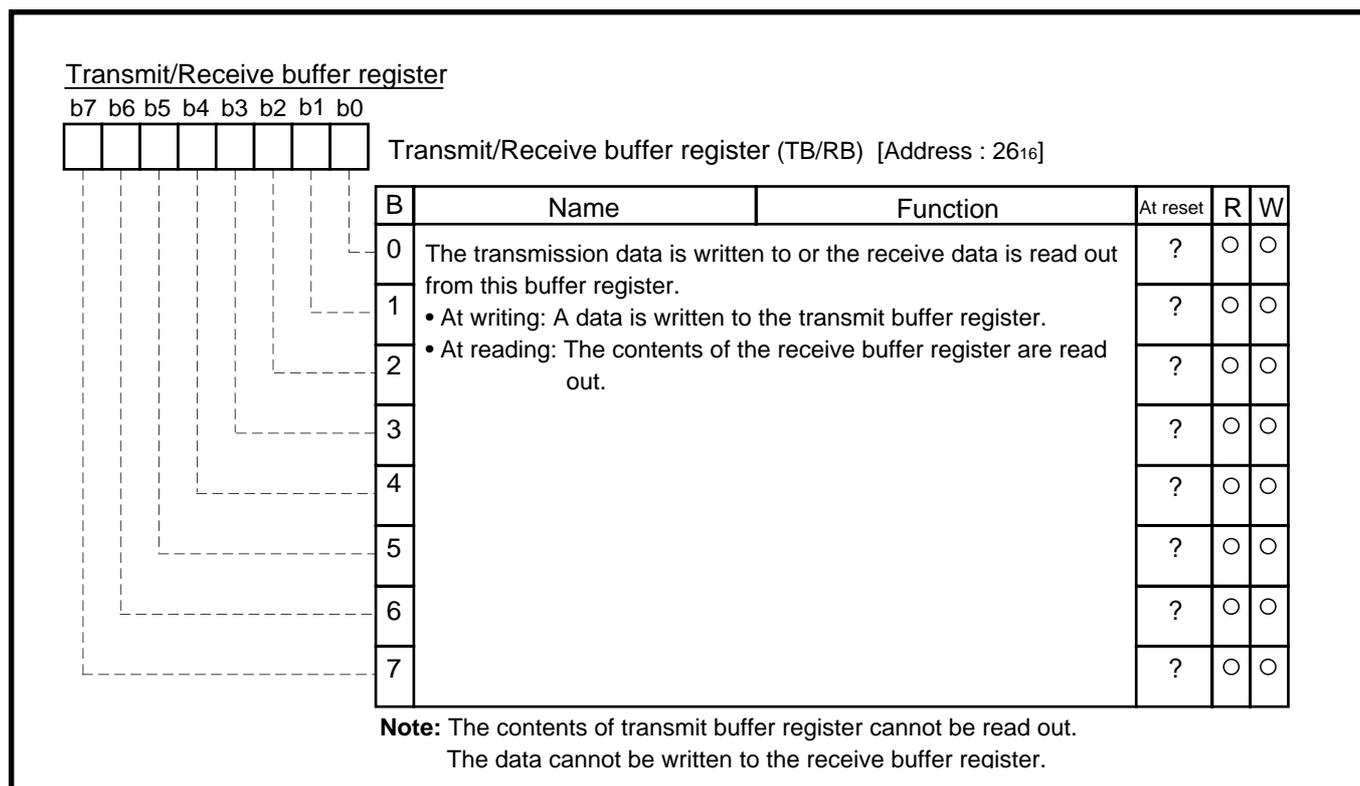


Fig. 3.4.54 Structure of Transmit/Receive buffer register

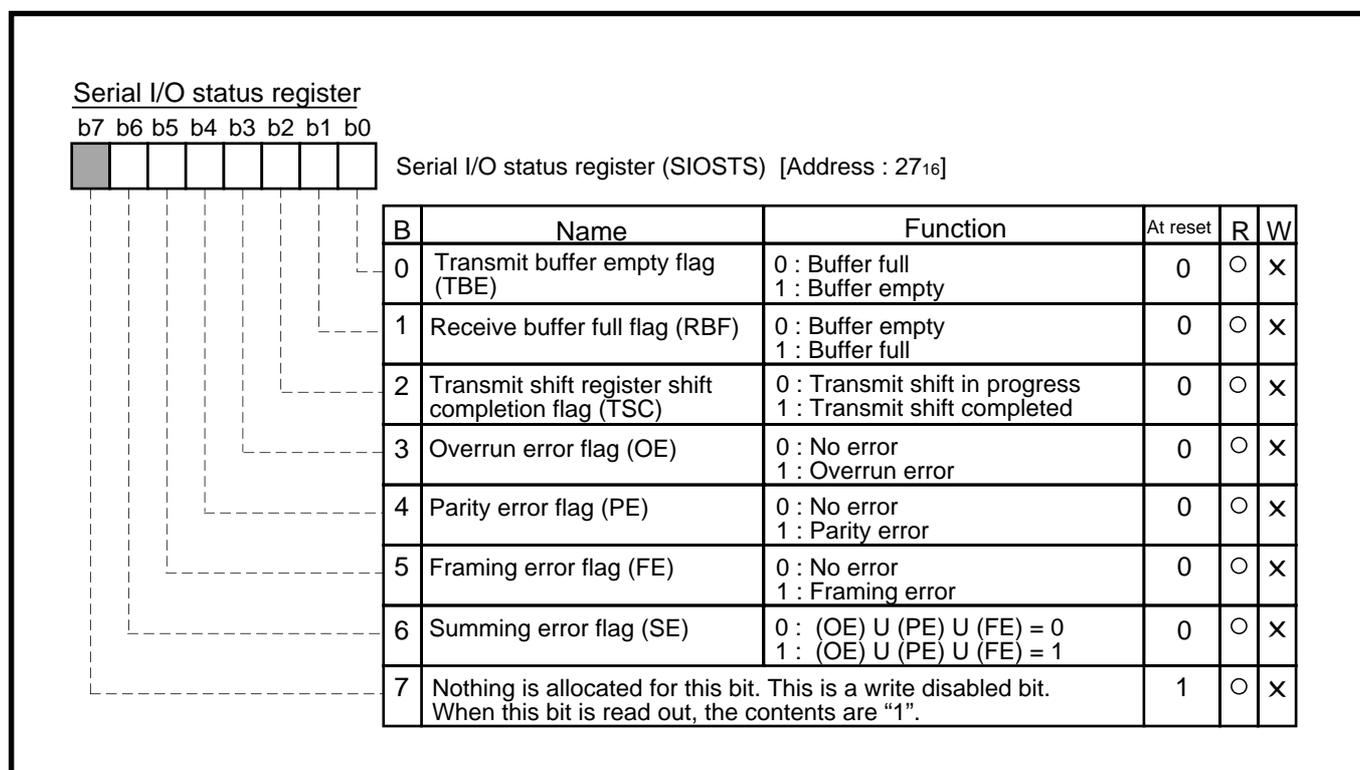


Fig. 3.4.55 Structure of Serial I/O status register

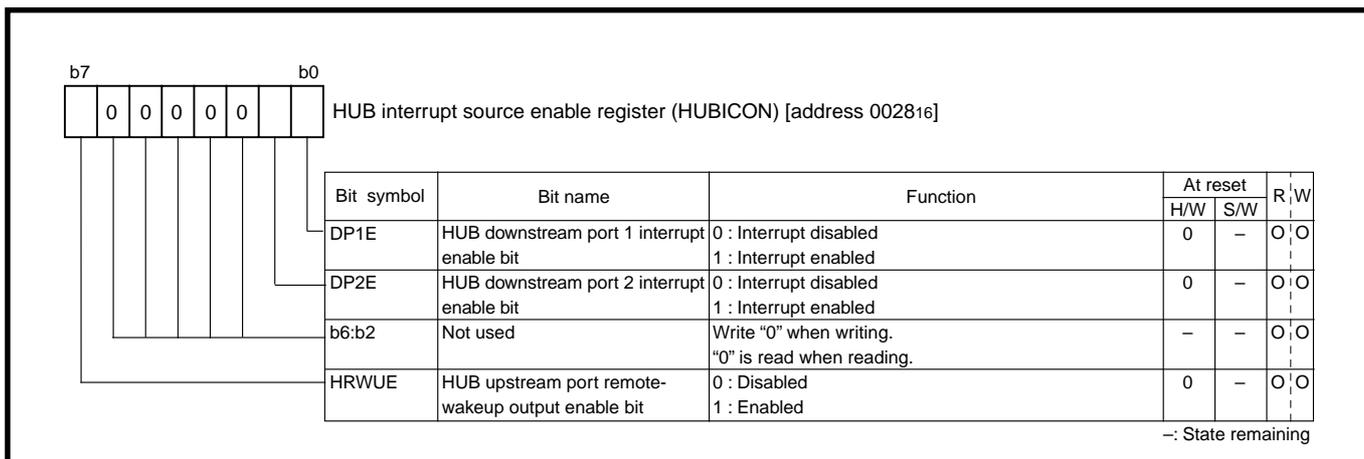


Fig. 3.4.56 Structure of HUB interrupt source enable register

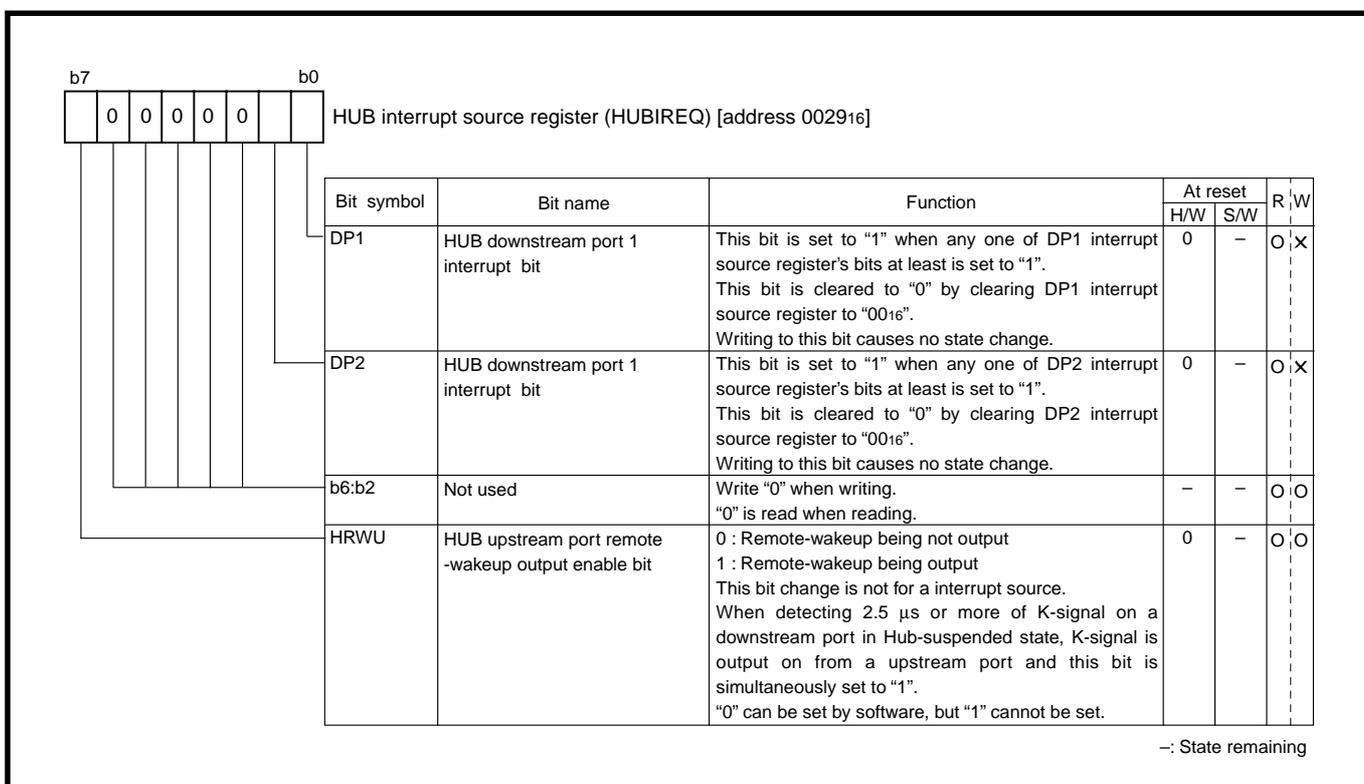


Fig. 3.4.57 Structure of HUB interrupt source register

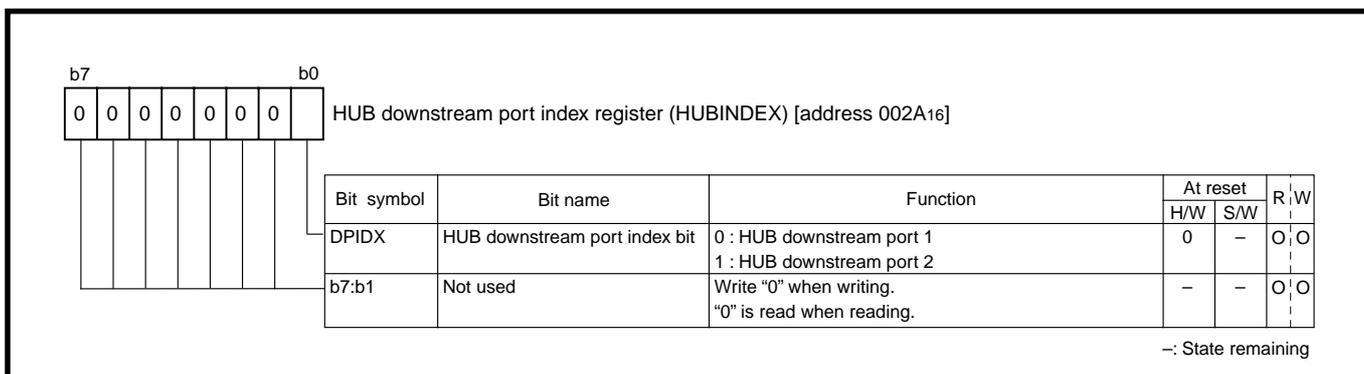


Fig. 3.4.58 Structure of HUB downstream port index register

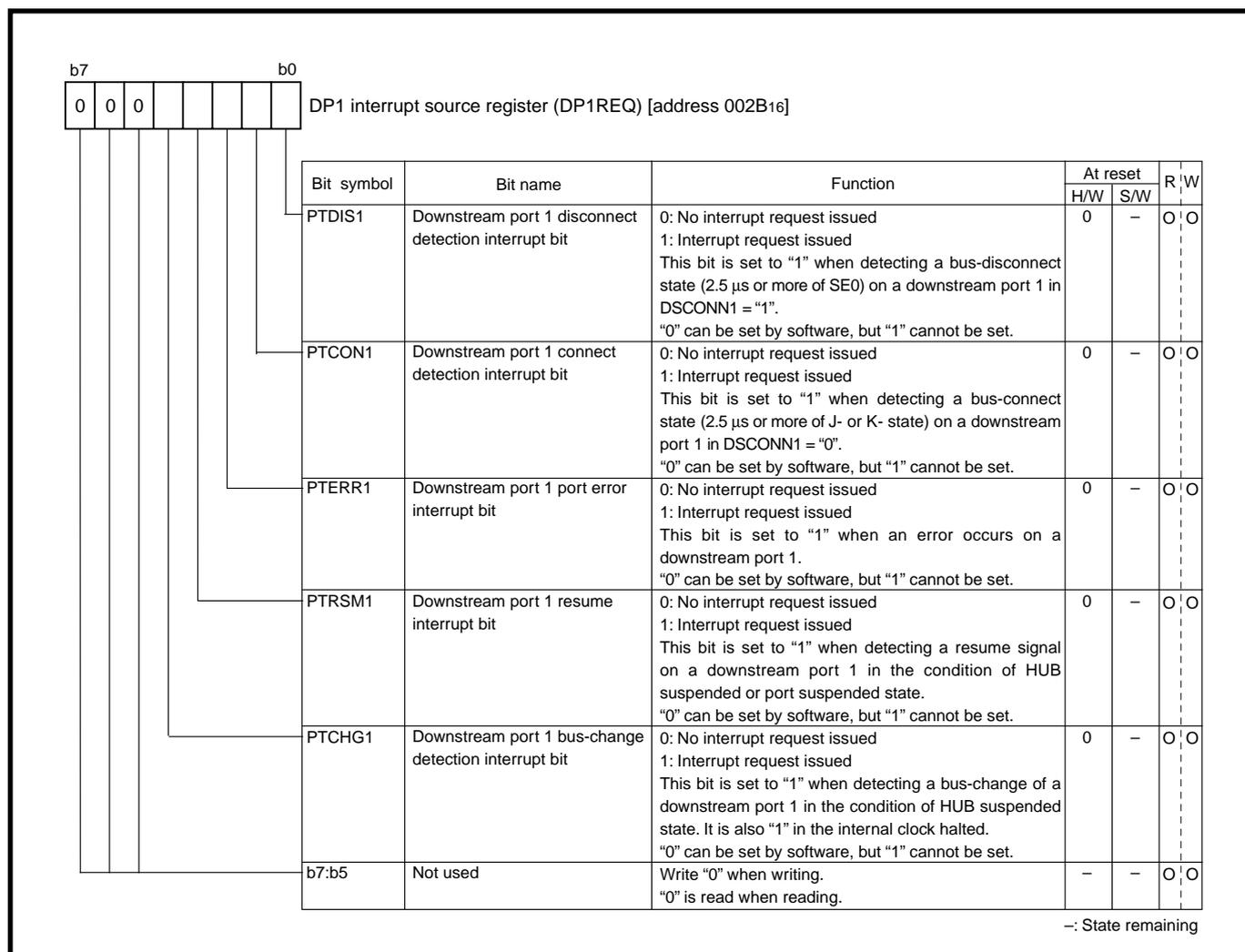


Fig. 3.4.59 Structure of DP1 interrupt source register

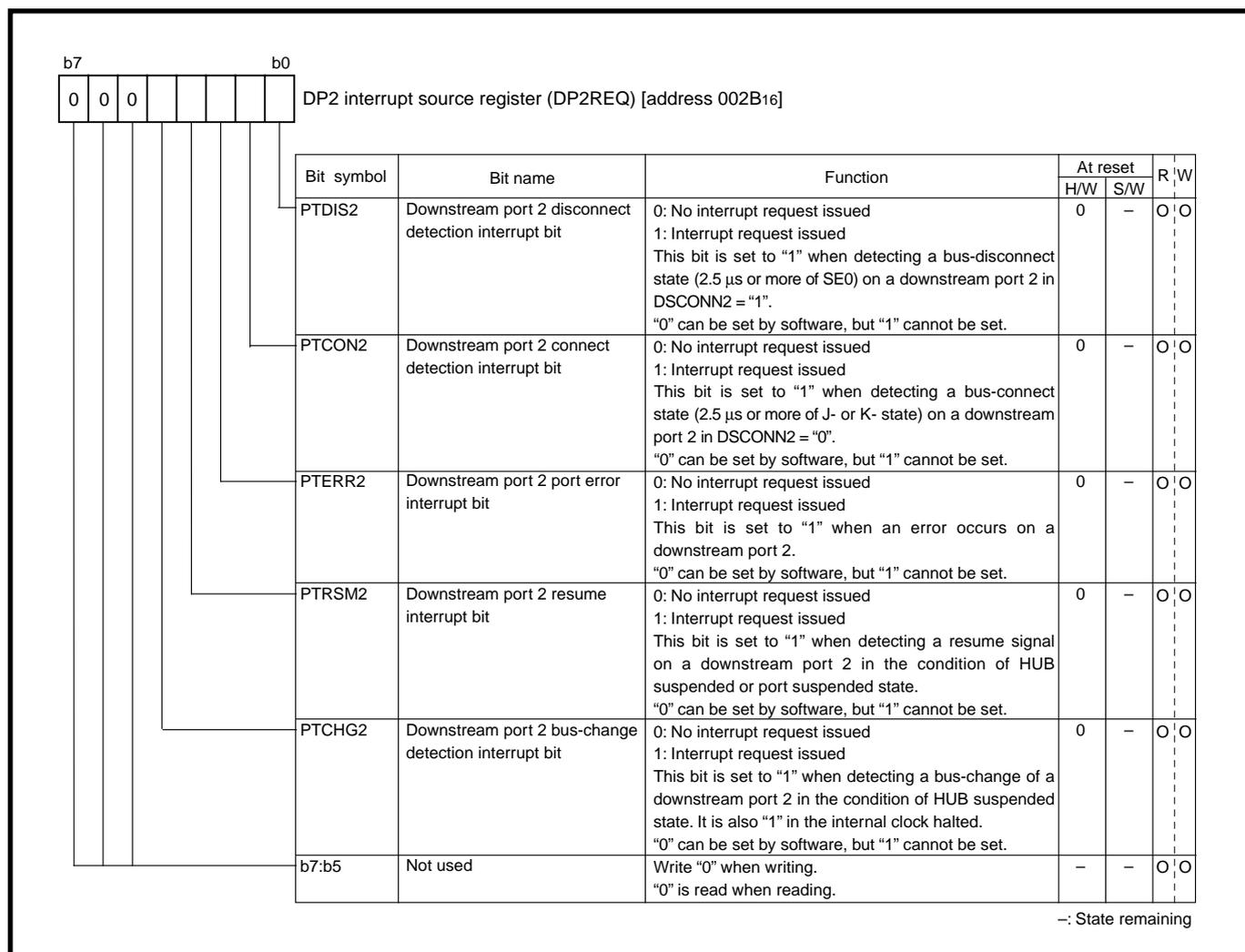


Fig. 3.4.60 Structure of DP2 interrupt source register

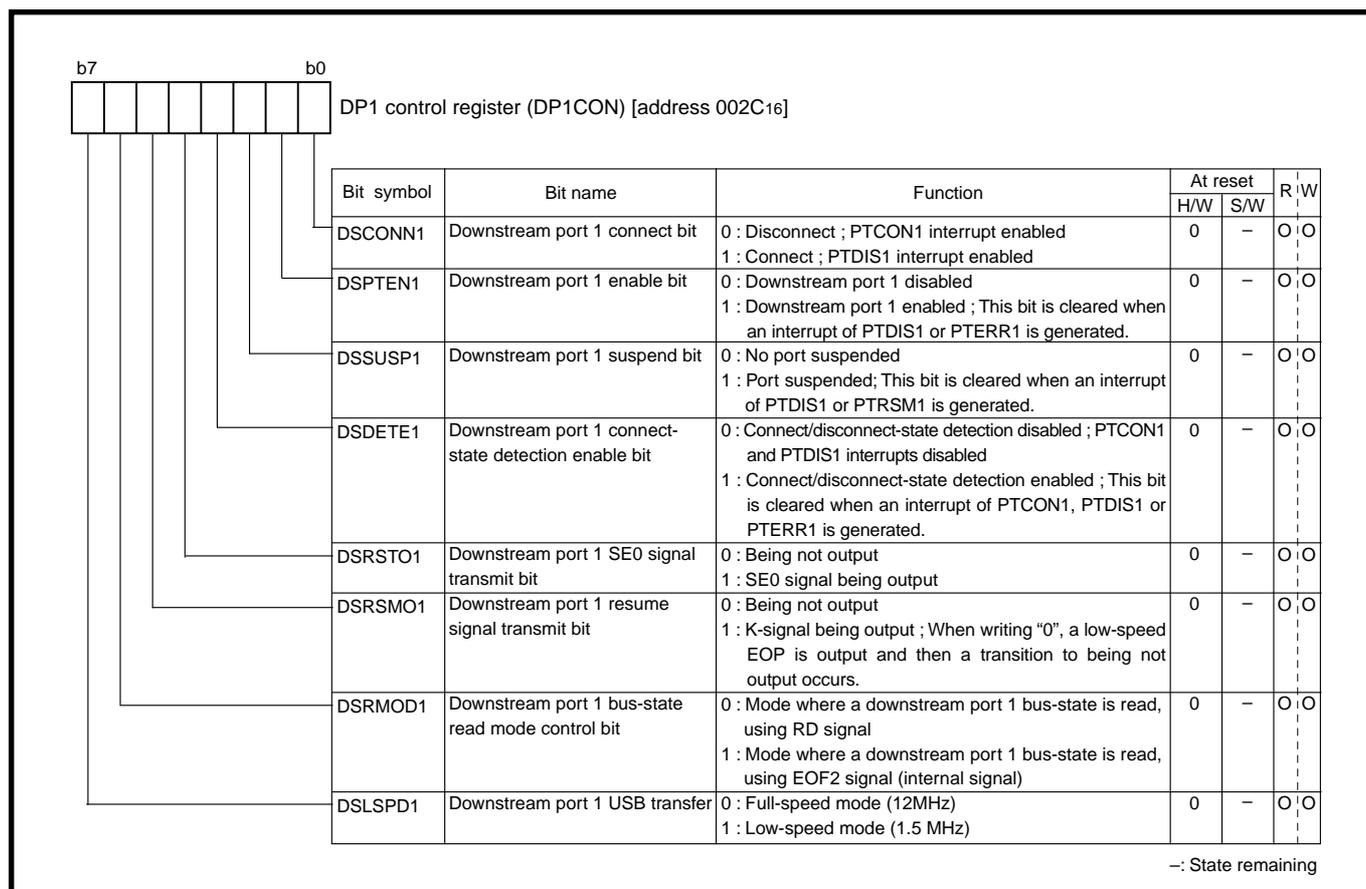


Fig. 3.4.61 Structure of DP1 control register

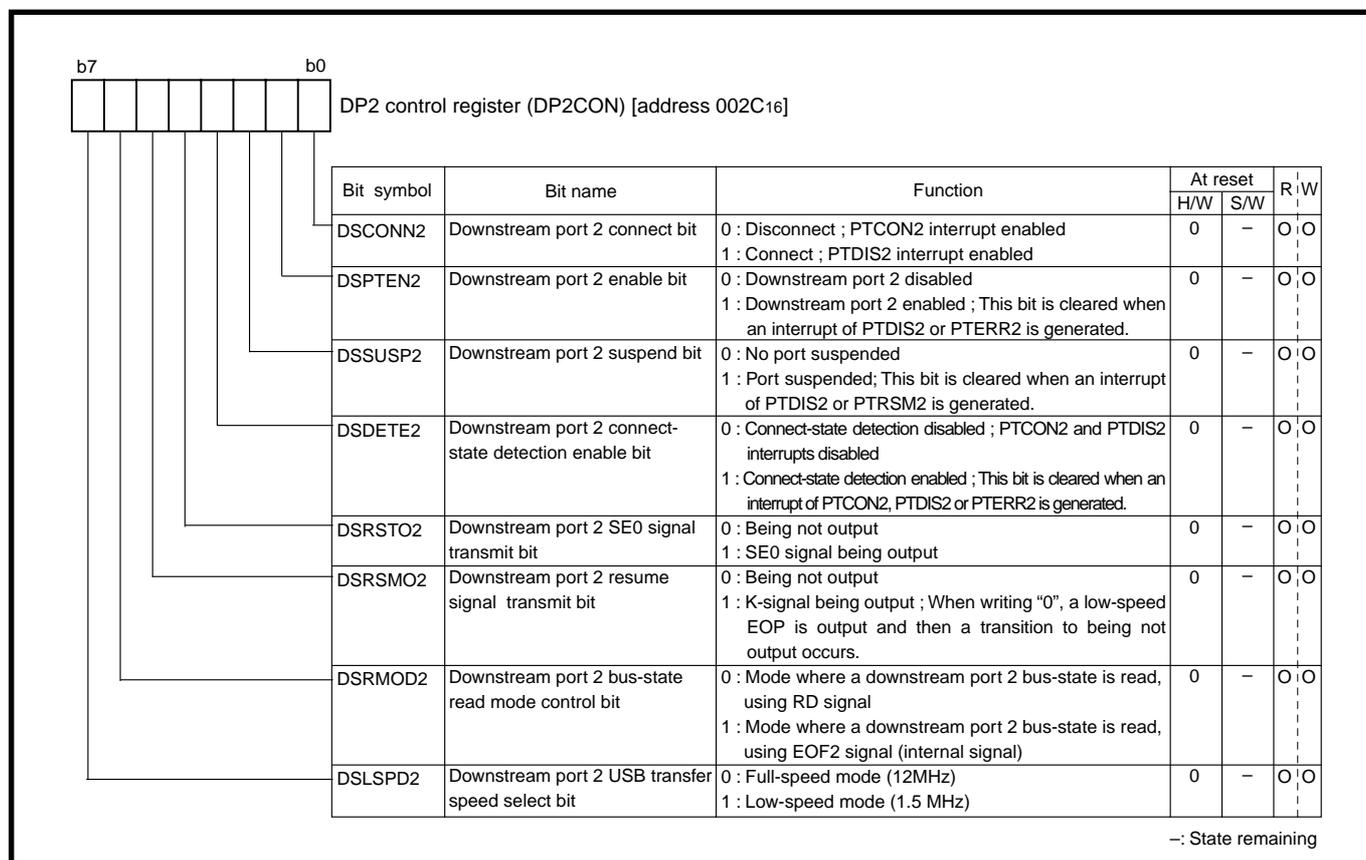


Fig. 3.4.62 Structure of DP2 control register

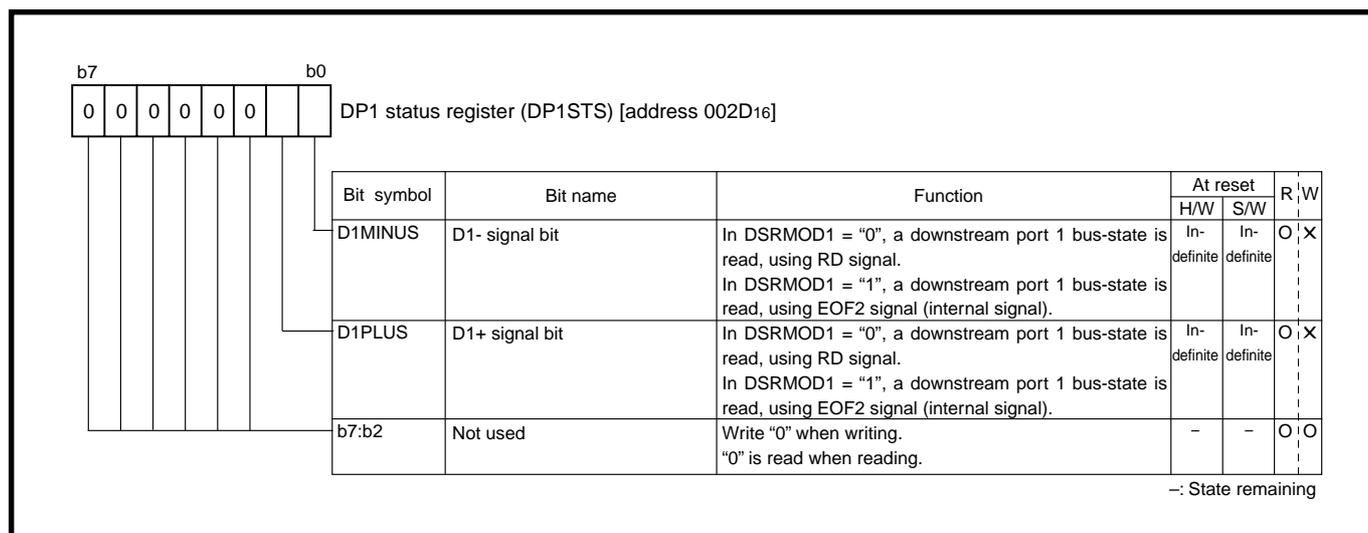


Fig. 3.4.63 Structure of DP1 status register

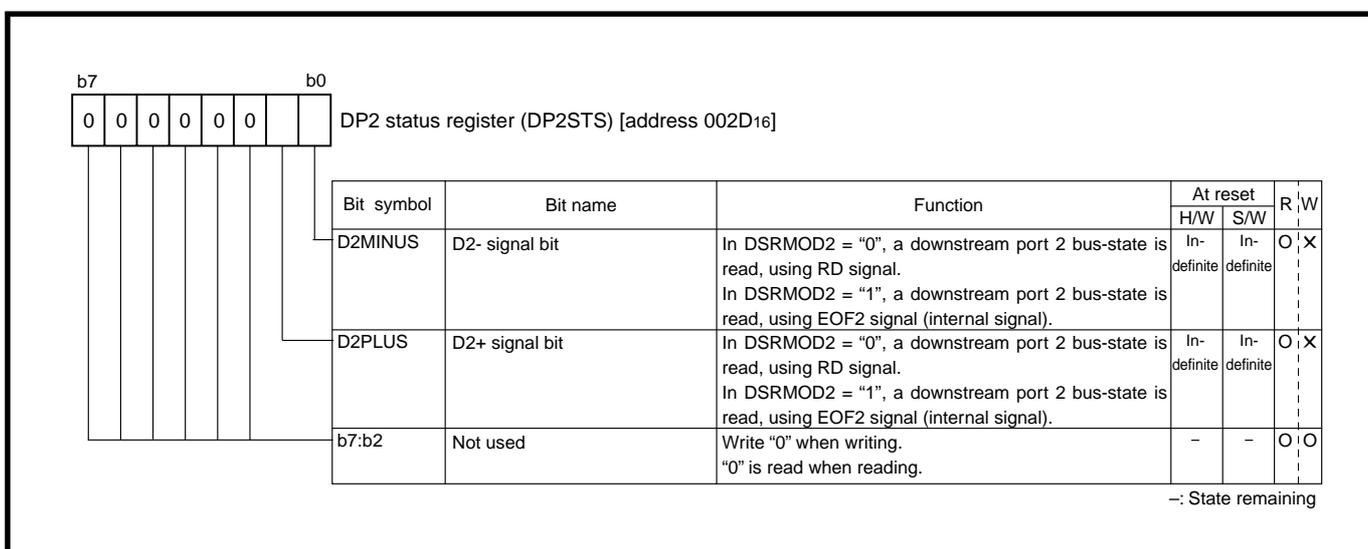


Fig. 3.4.64 Structure of DP2 status register

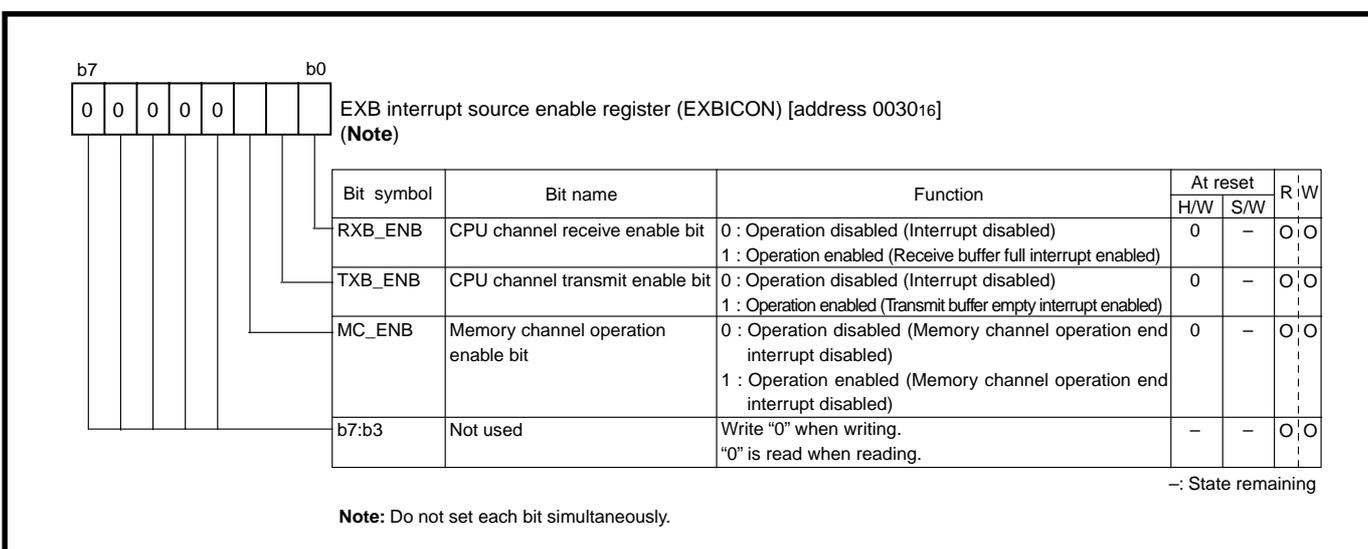


Fig. 3.4.65 Structure of EXB interrupt source enable register

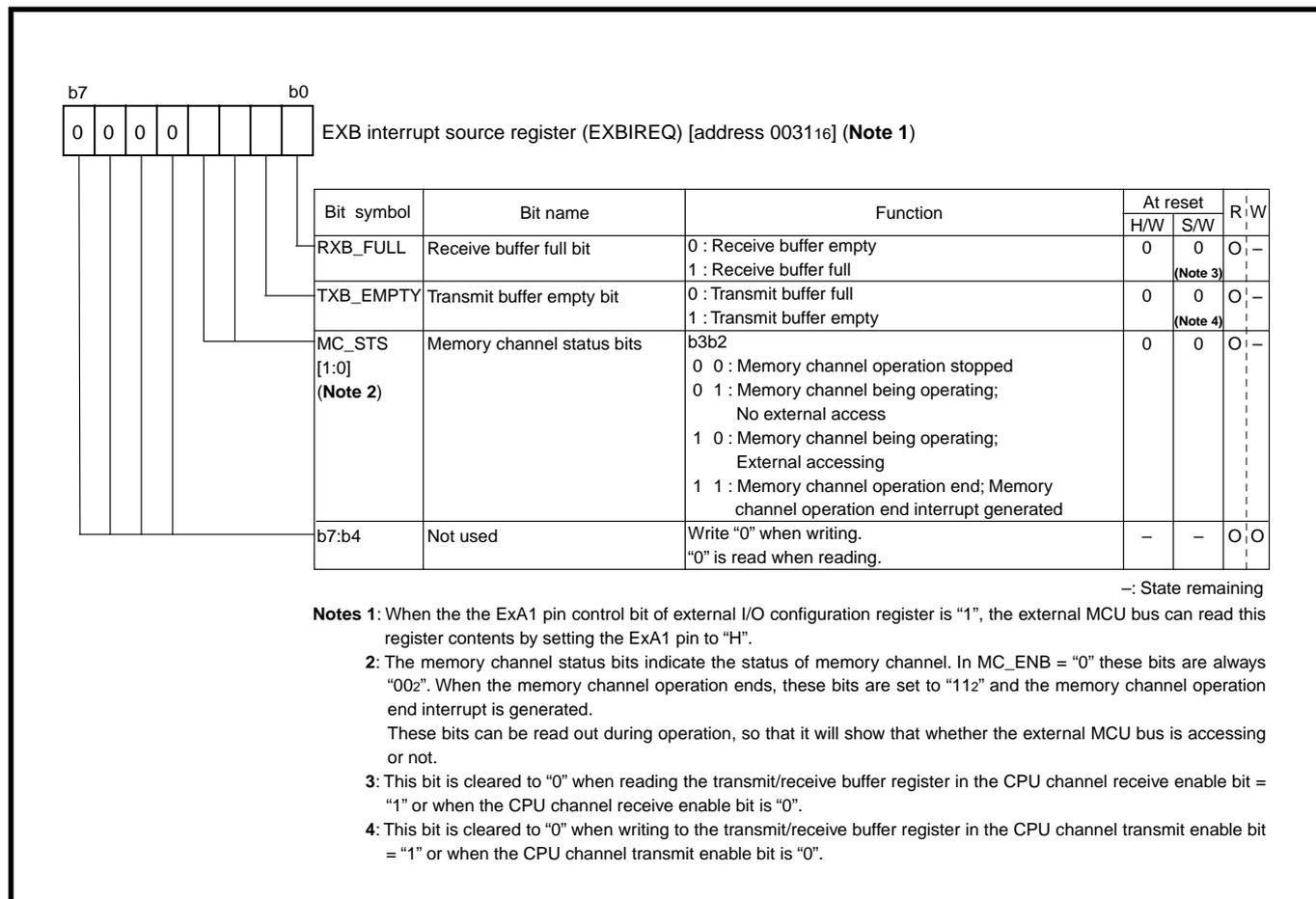


Fig. 3.4.66 Structure of EXB interrupt source register

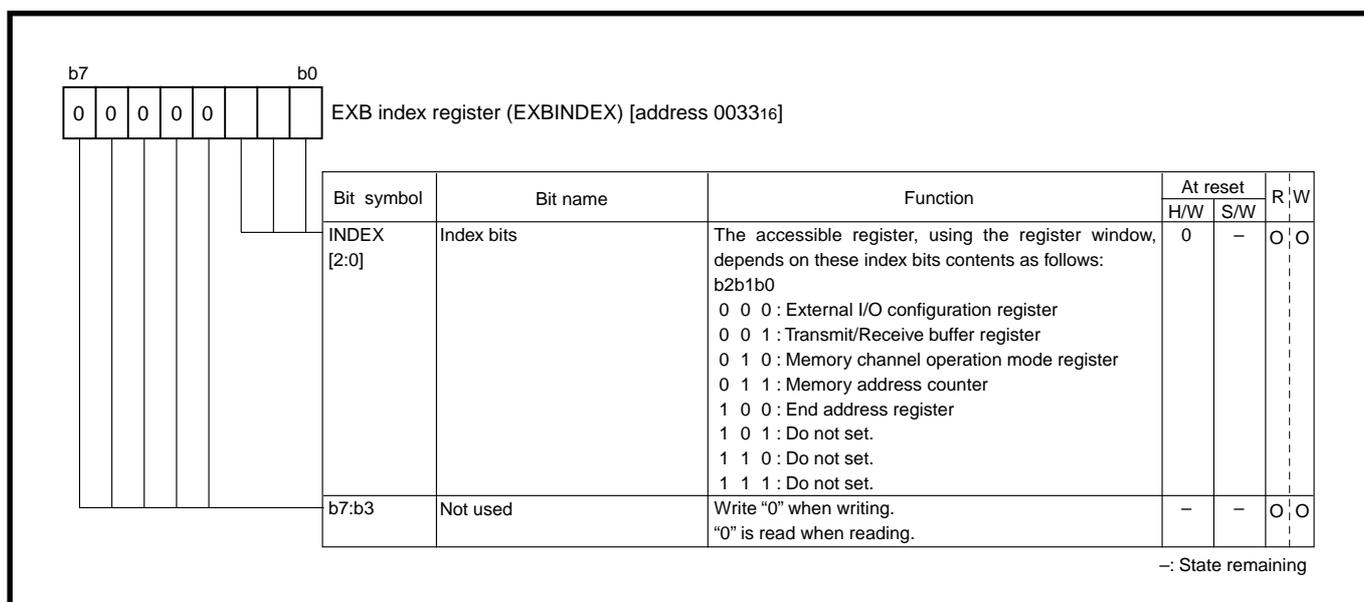


Fig. 3.4.67 Structure of EXB index register

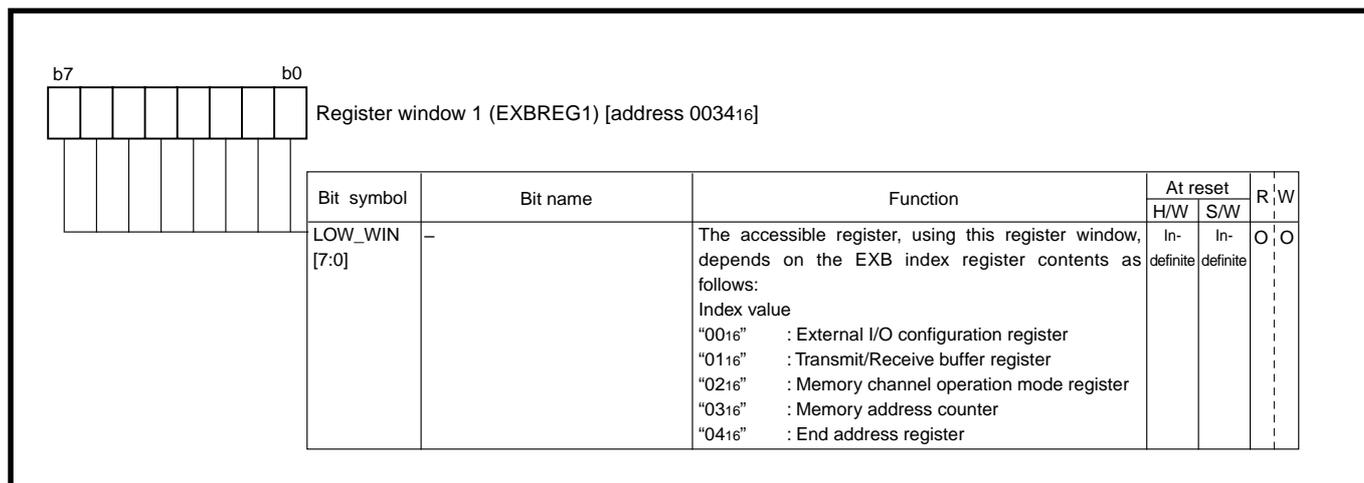


Fig. 3.4.68 Structure of Register window 1

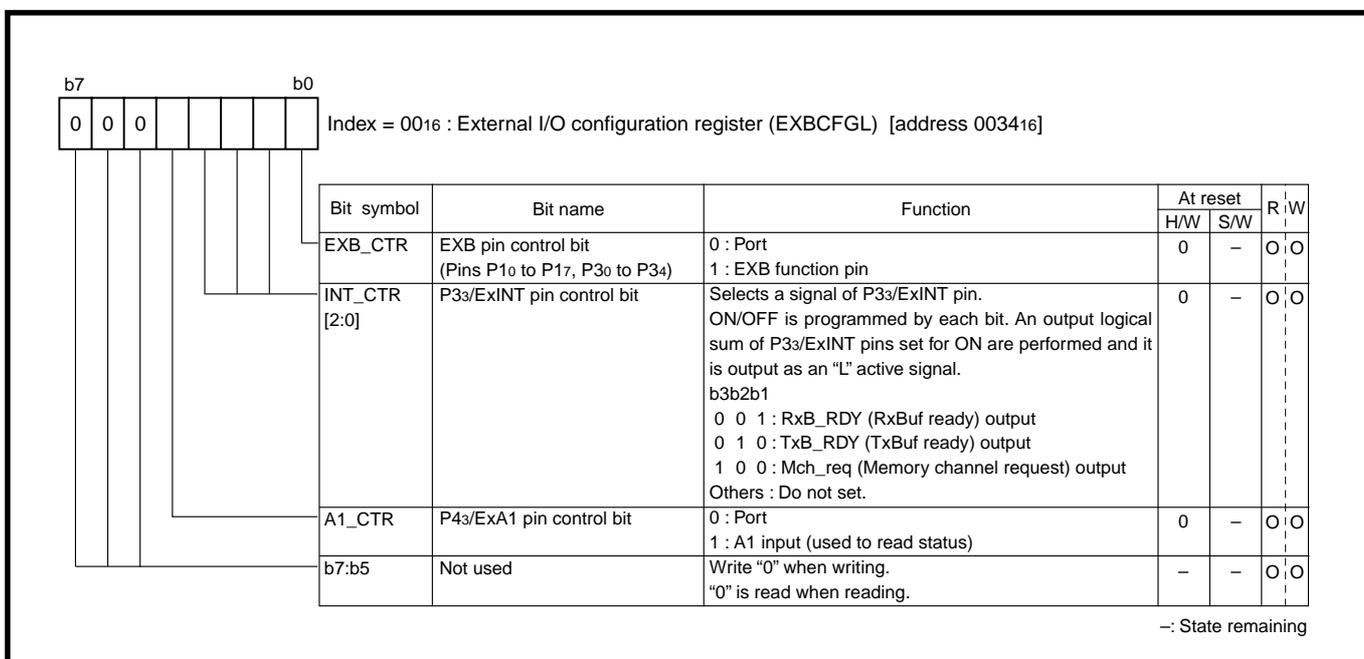


Fig. 3.4.69 Index00[low]; Structure of External I/O configuration register

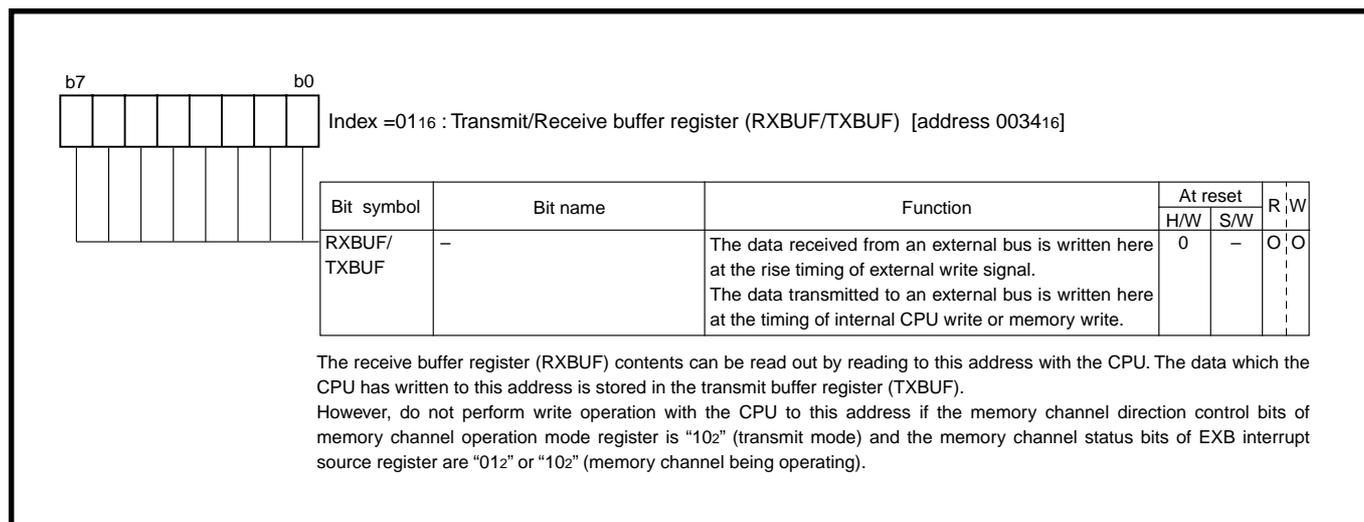


Fig. 3.4.70 Index01[low]; Structure of Transmit/Receive buffer register

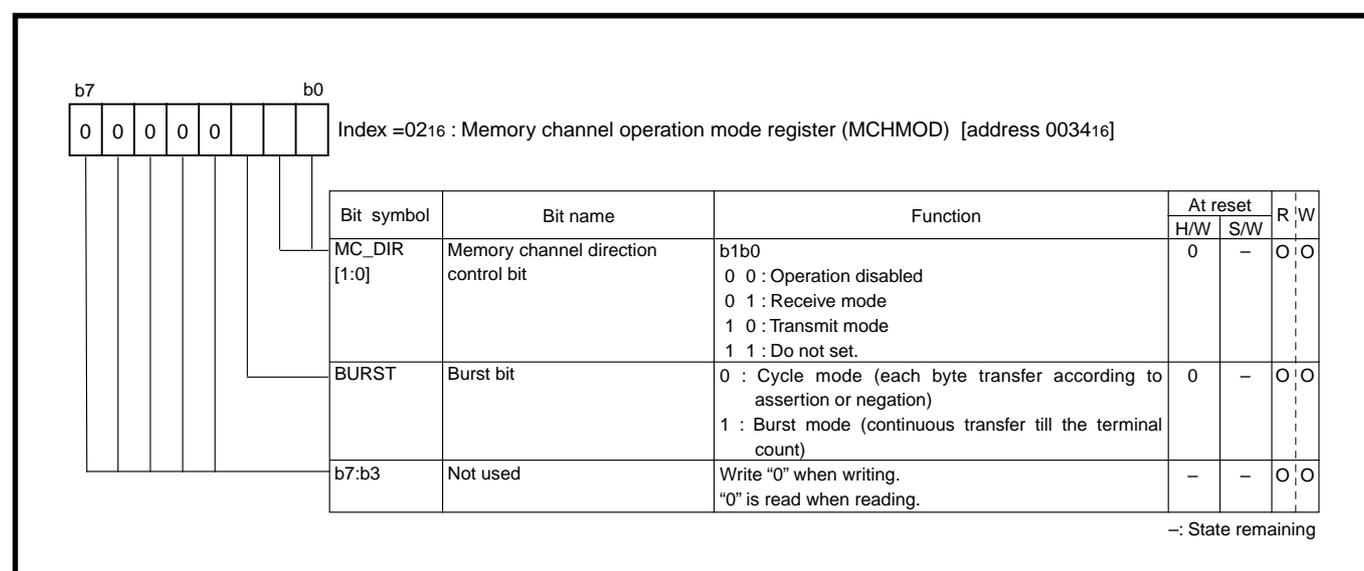


Fig. 3.4.71 Index02[low]; Structure of Memory channel operation mode register

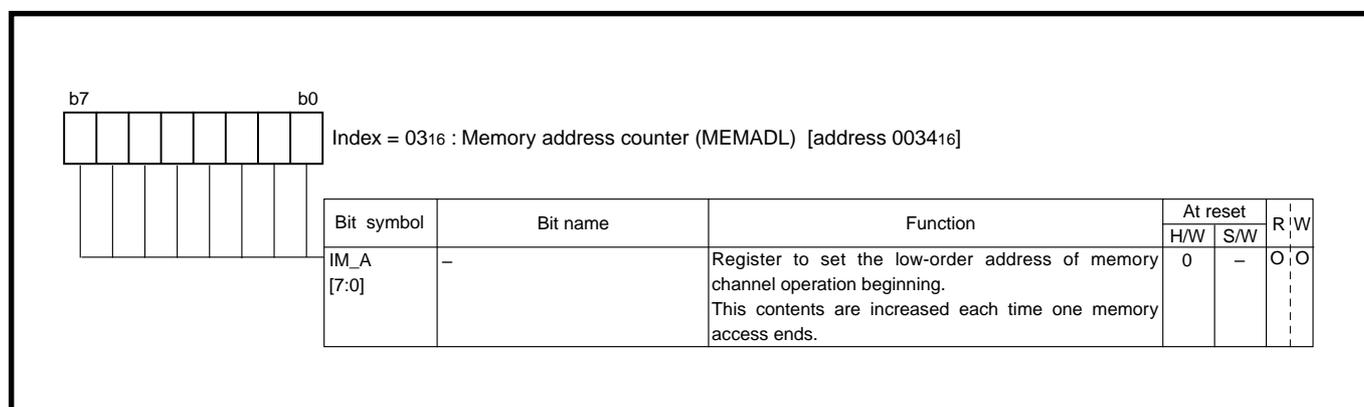


Fig. 3.4.72 Index03[low]; Structure of Memory address counter

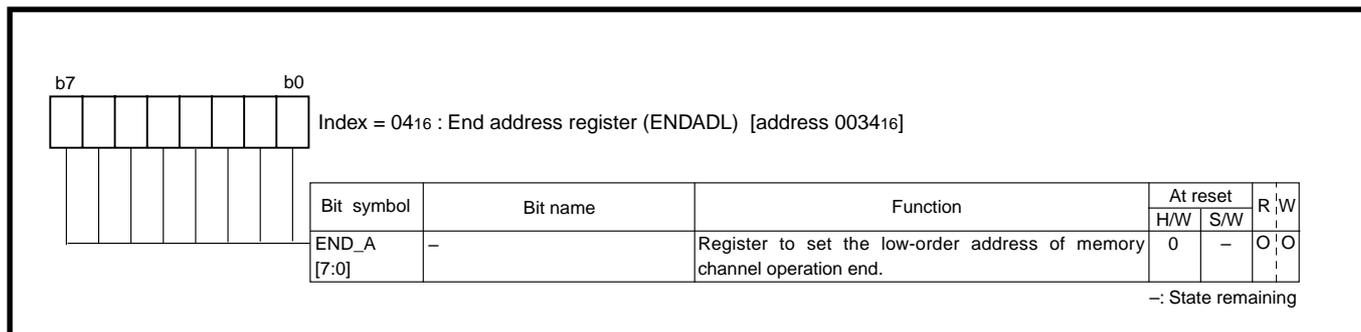


Fig. 3.4.73 Index04[low]; Structure of End address register

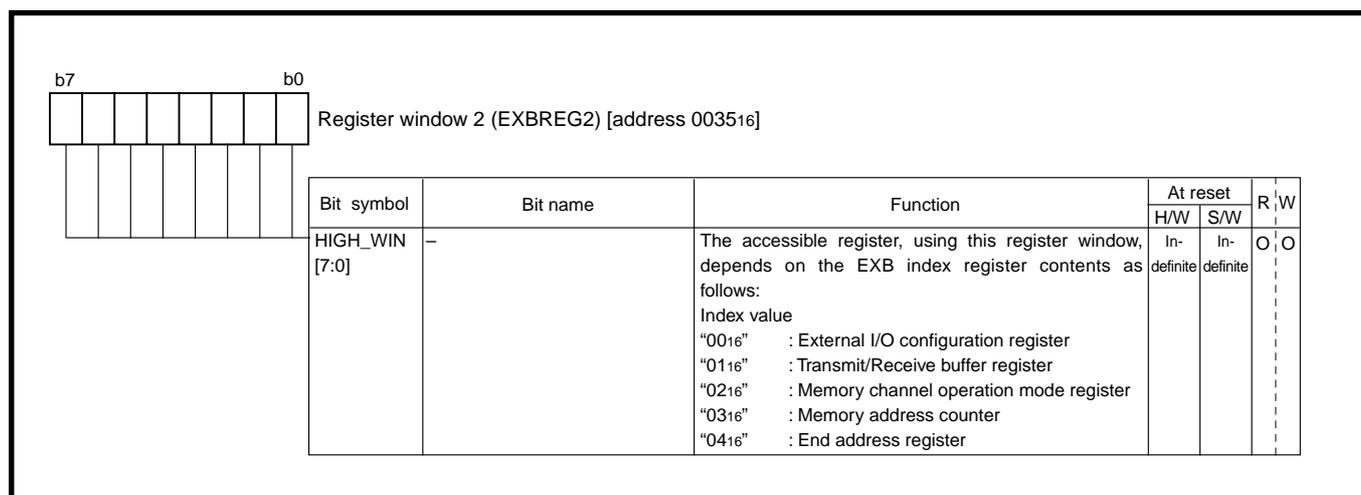


Fig. 3.4.74 Structure of Register window 2

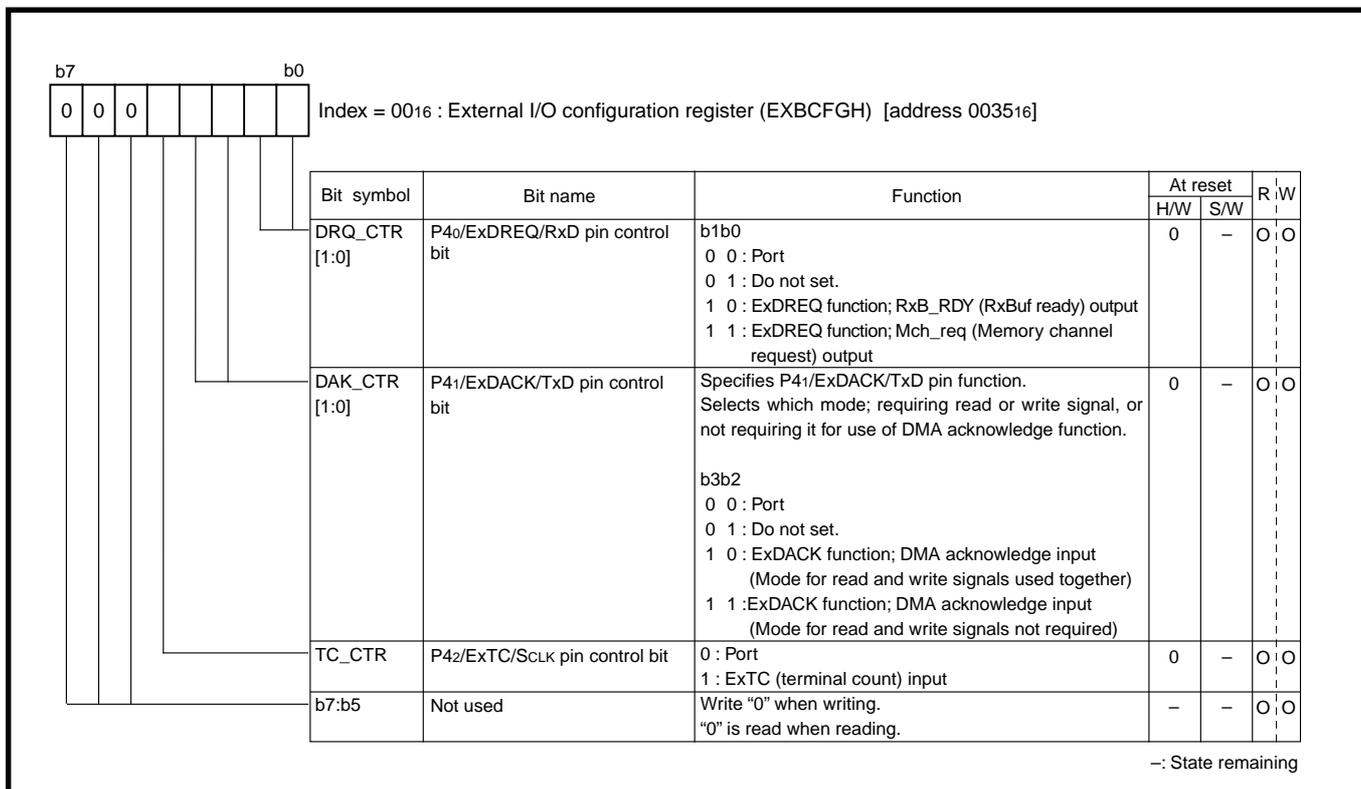


Fig. 3.4.75 Index00[high]; Structure of External I/O configuration register

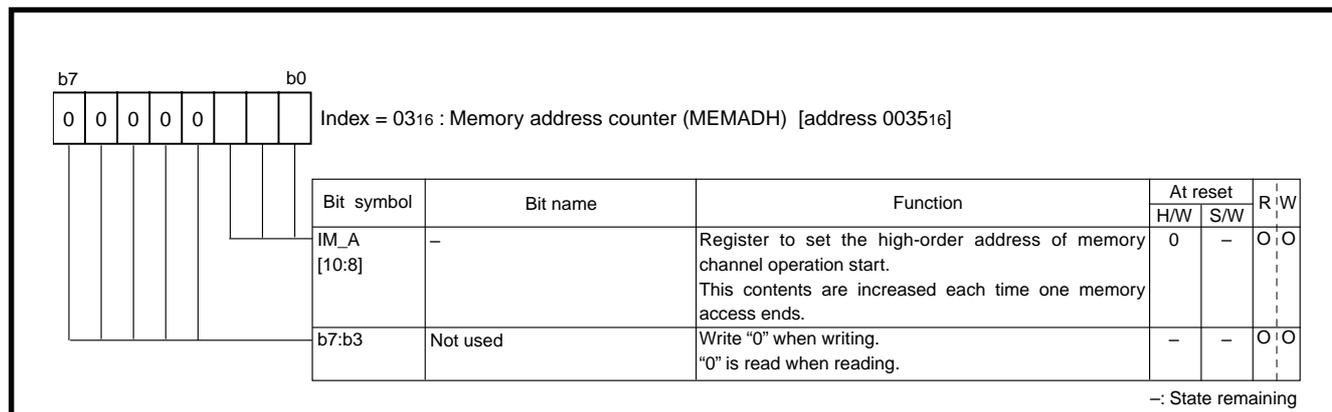


Fig. 3.4.76 Index03[high]; Structure of Memory address counter

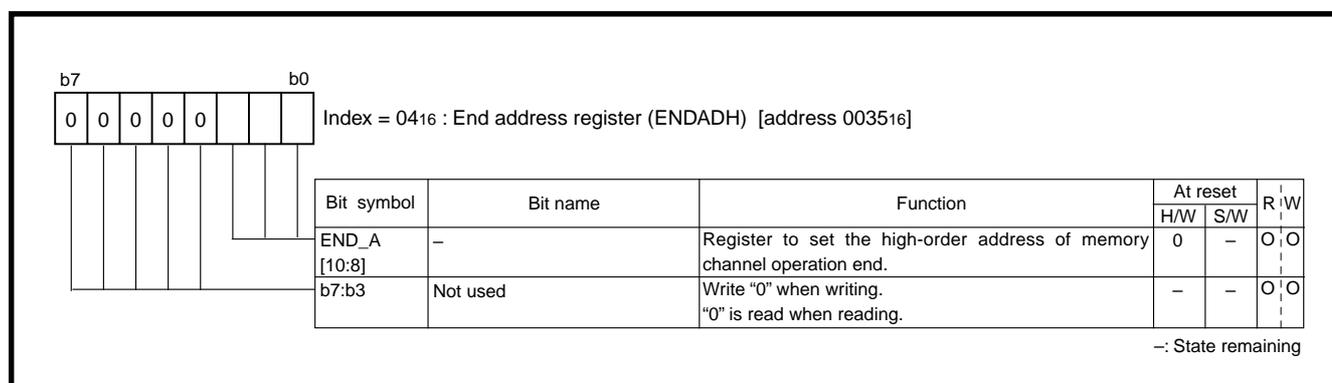


Fig. 3.4.77 Index04[high]; Structure of End address register

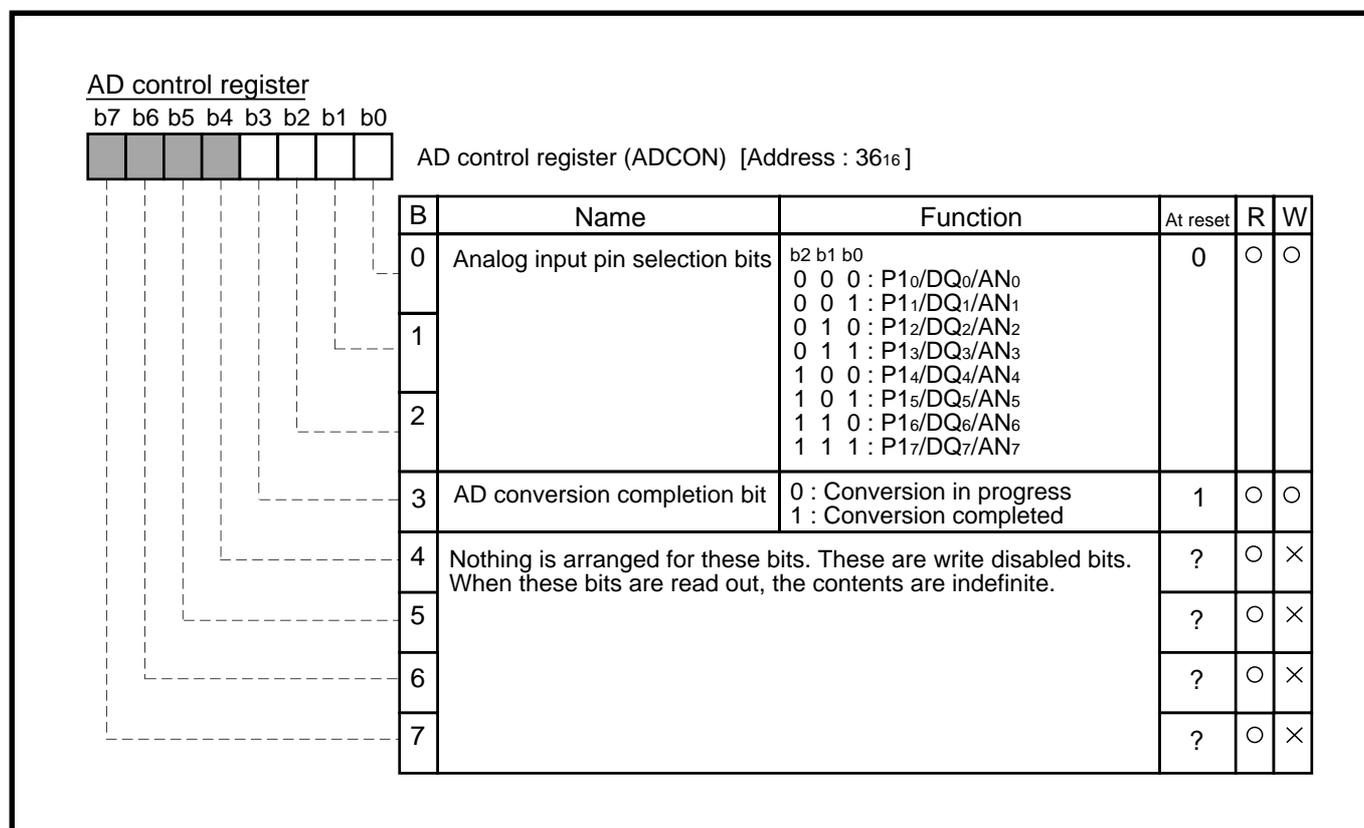


Fig. 3.4.78 Structure of AD control register



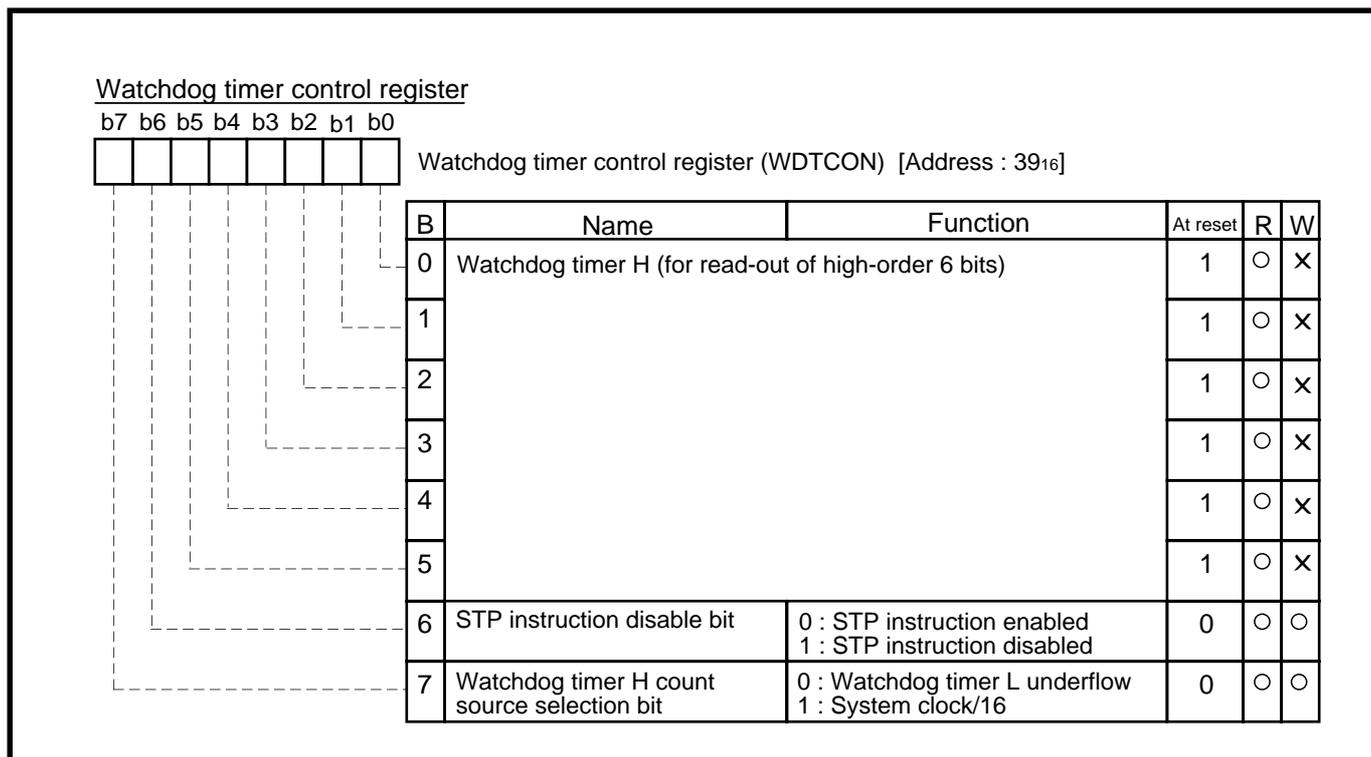


Fig. 3.4.81 Structure of Watchdog timer control register

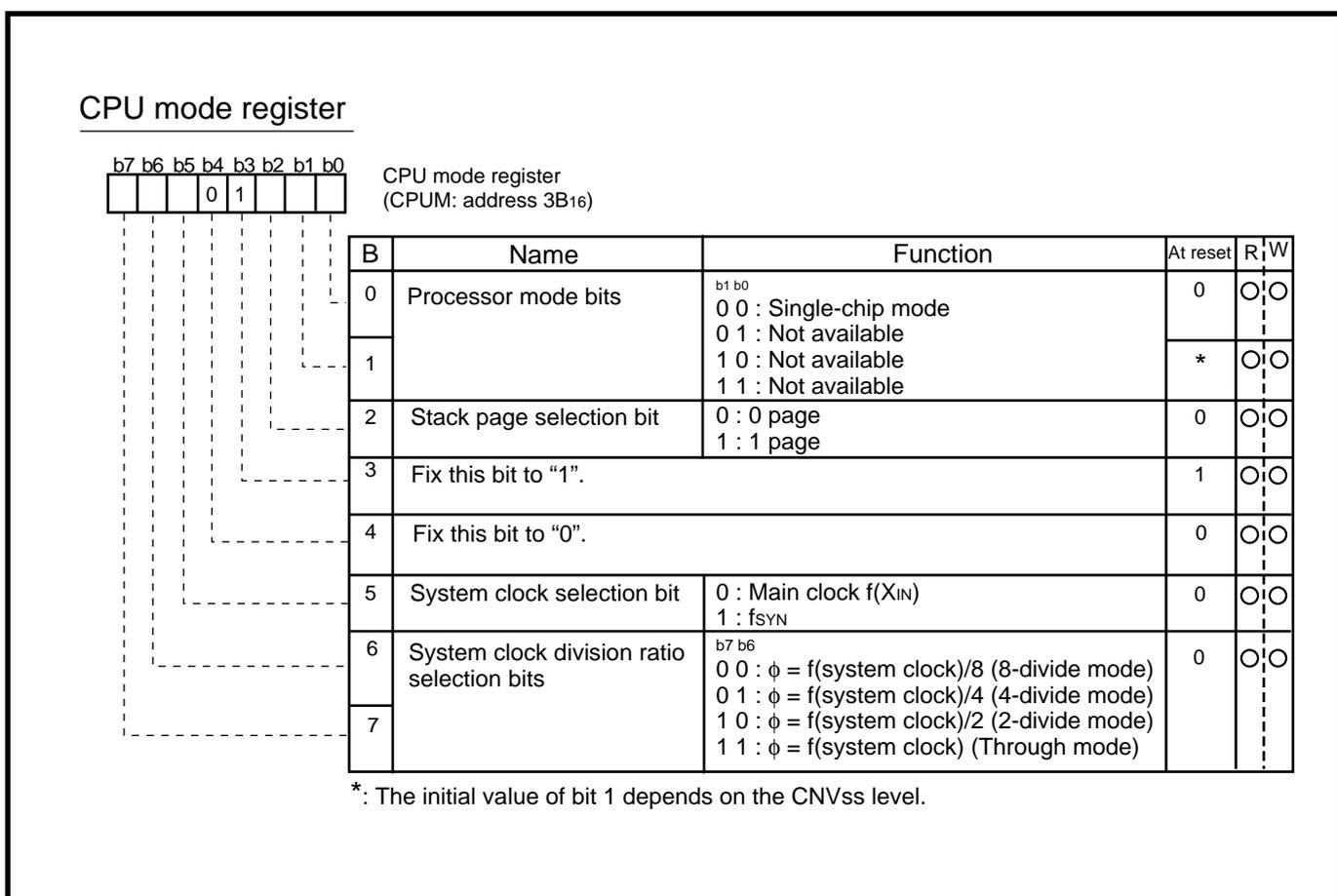


Fig. 3.4.82 Structure of CPU mode register

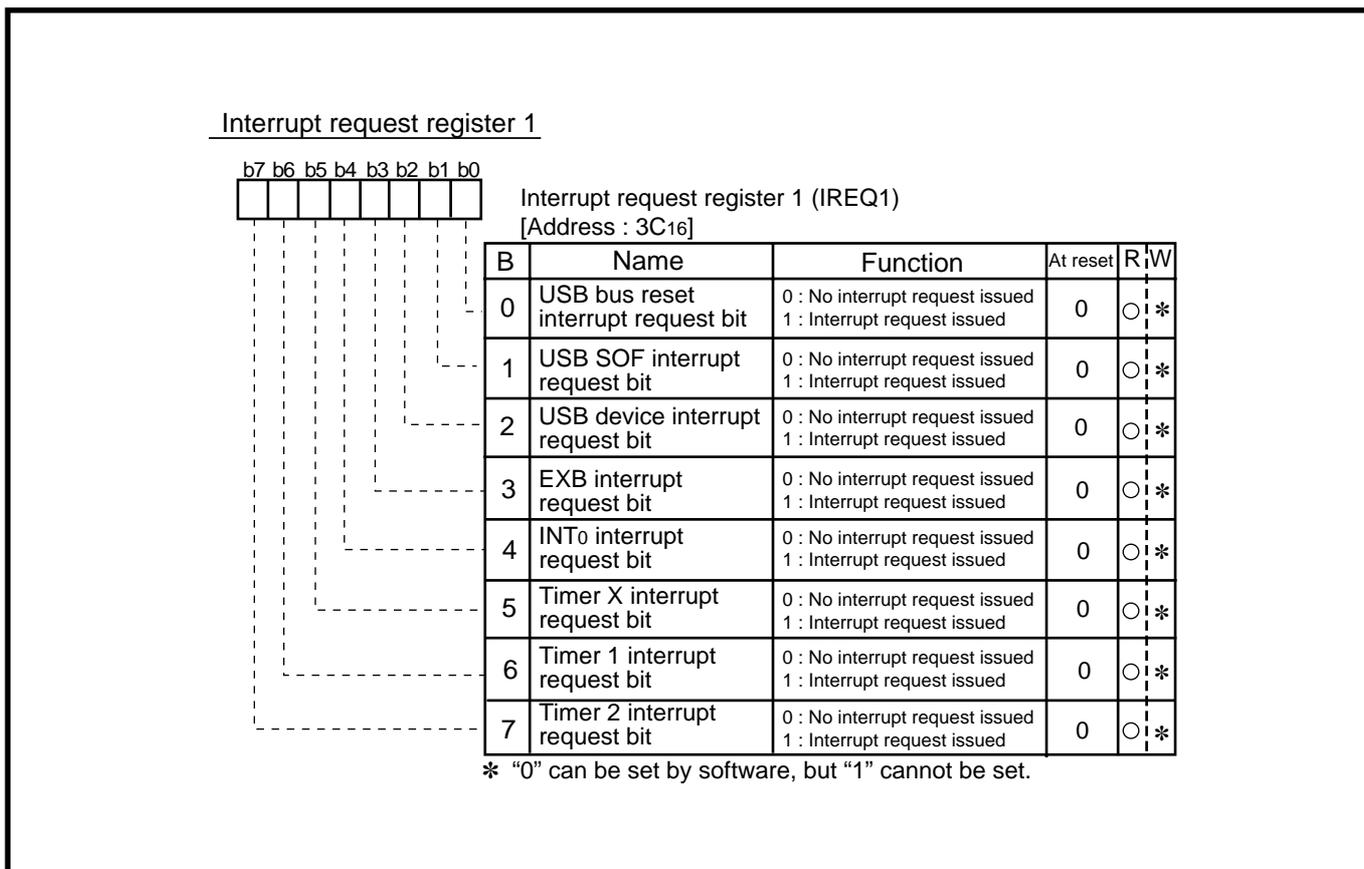


Fig. 3.4.83 Structure of Interrupt request register 1

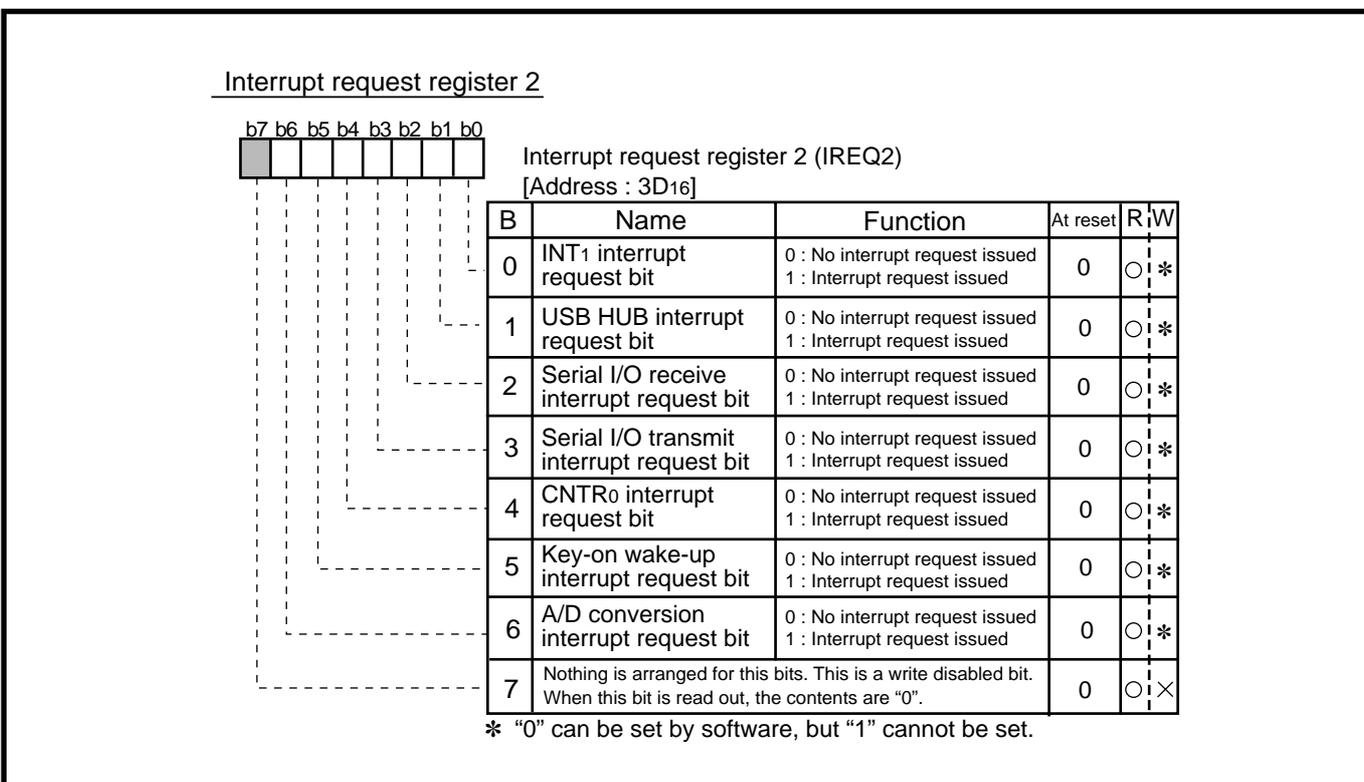


Fig. 3.4.84 Structure of Interrupt request register 2

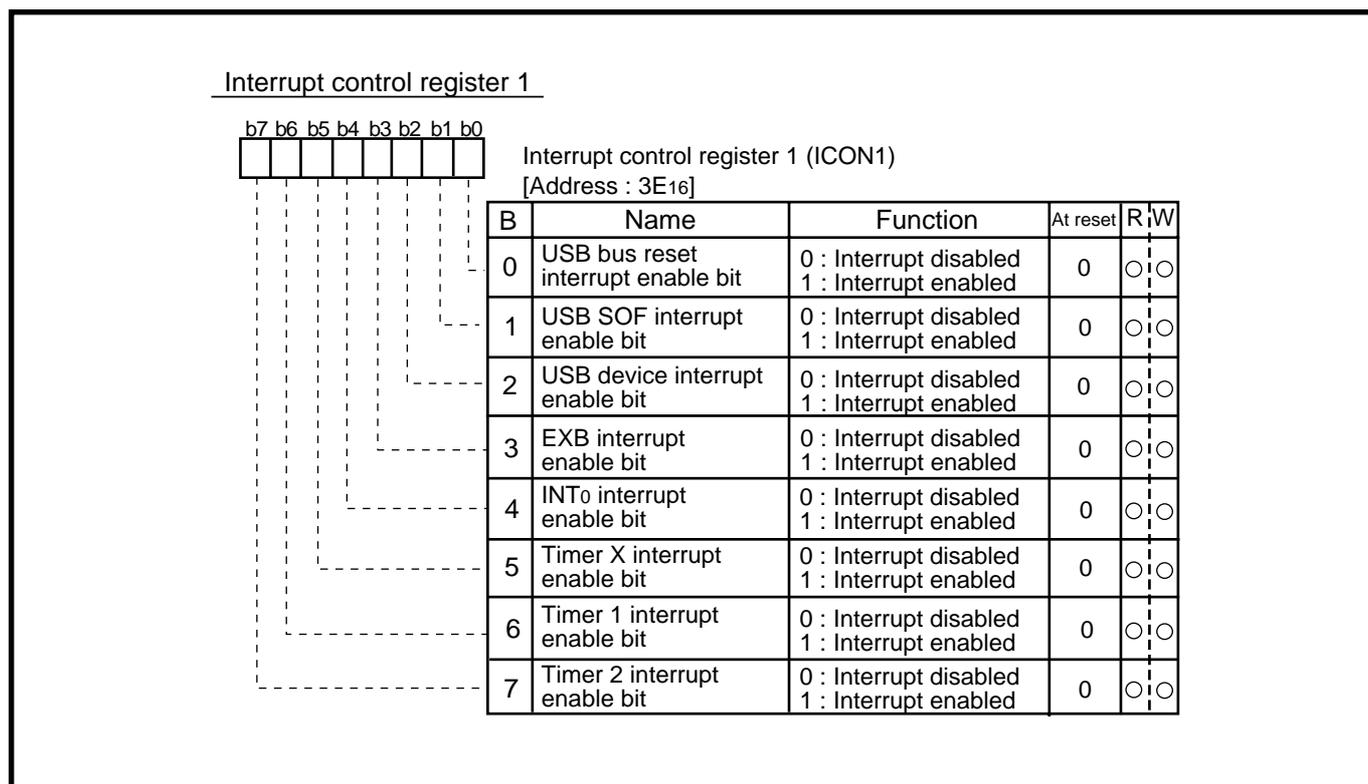


Fig. 3.4.85 Structure of Interrupt control register 1

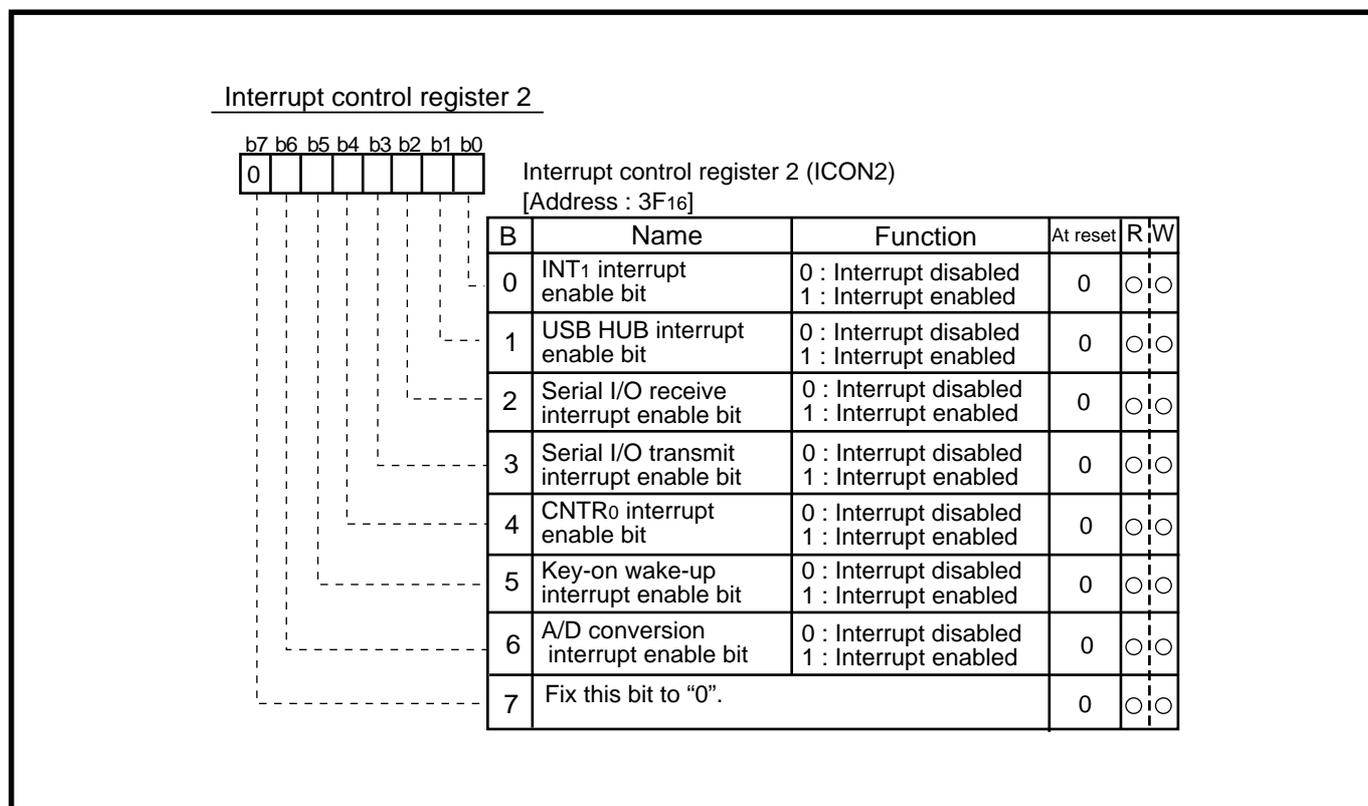


Fig. 3.4.86 Structure of Interrupt control register 2

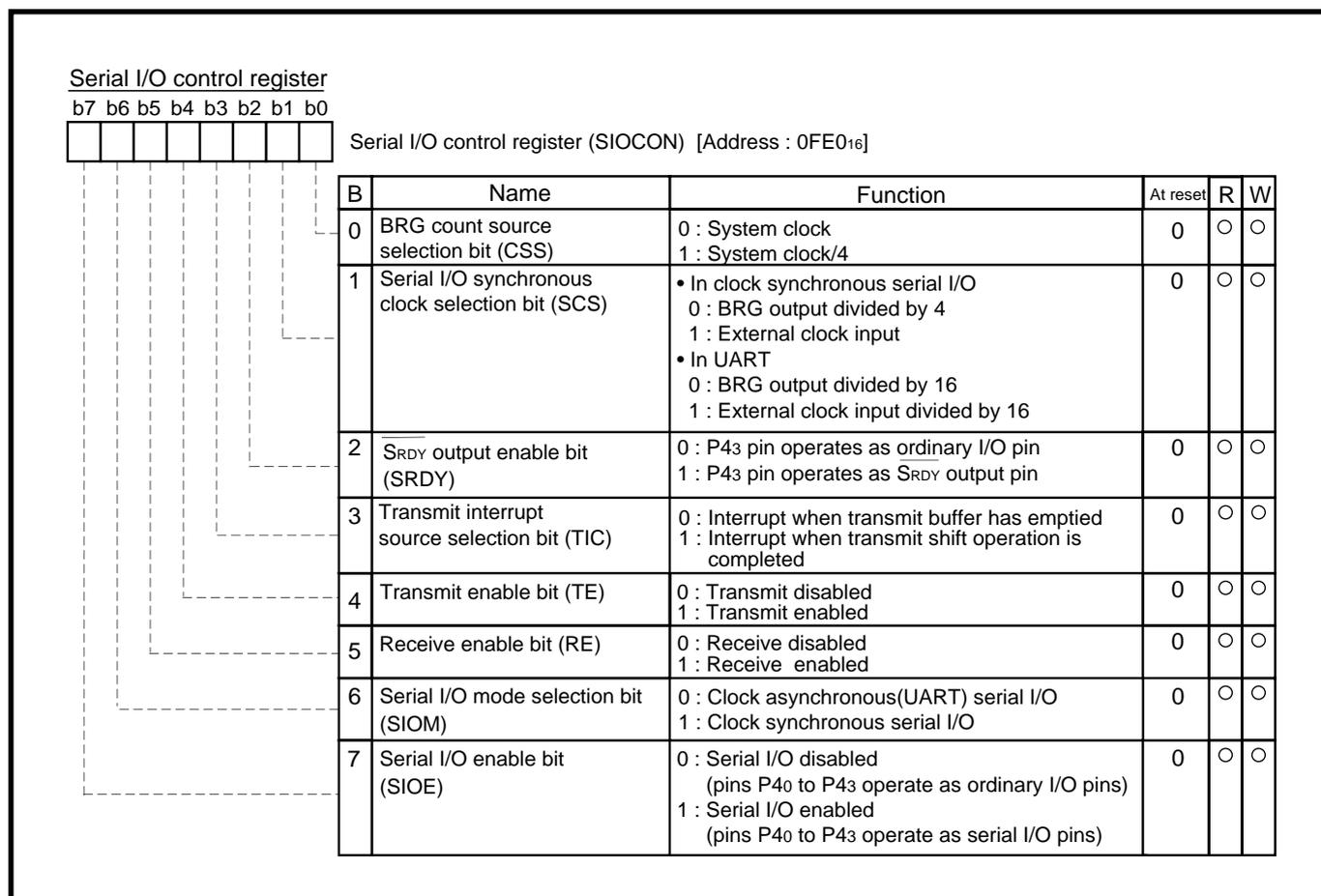


Fig. 3.4.87 Structure of Serial I/O control register

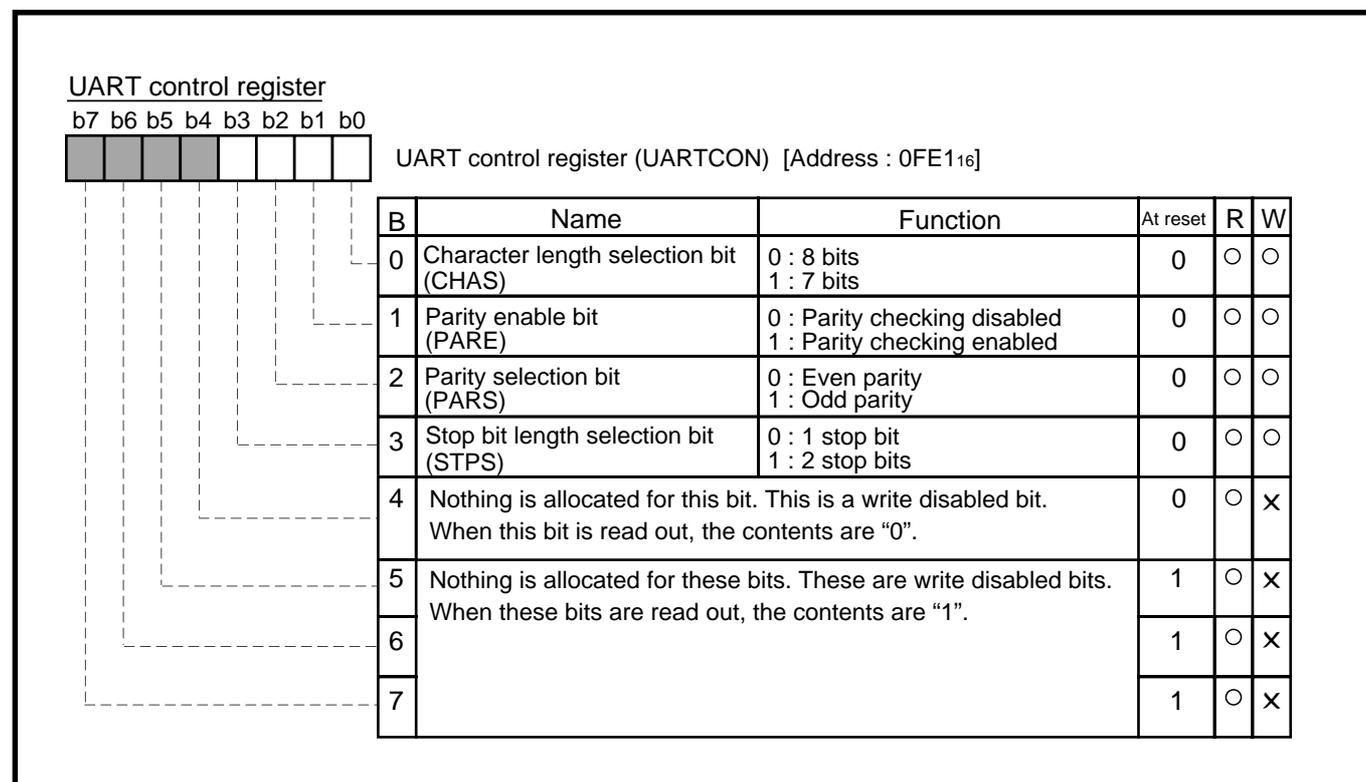


Fig. 3.4.88 Structure of UART control register

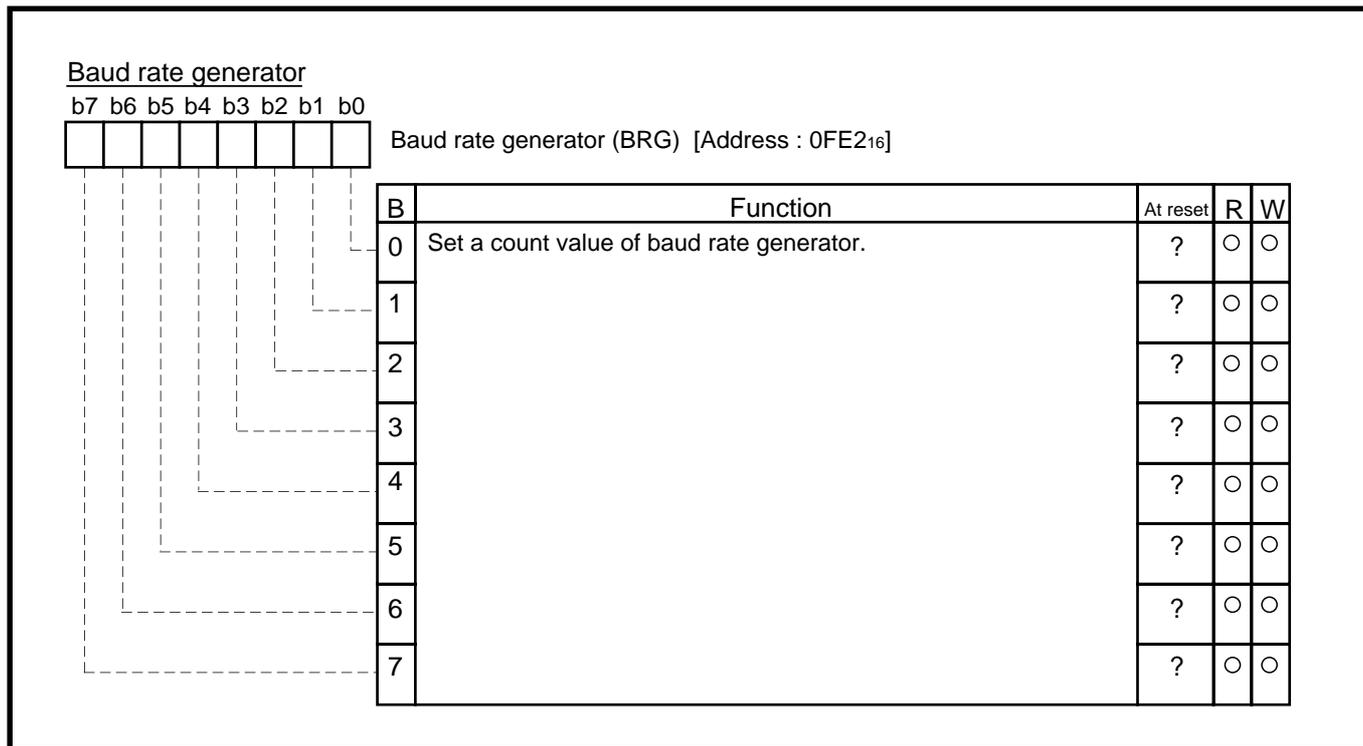


Fig. 3.4.89 Structure of Baud rate generator

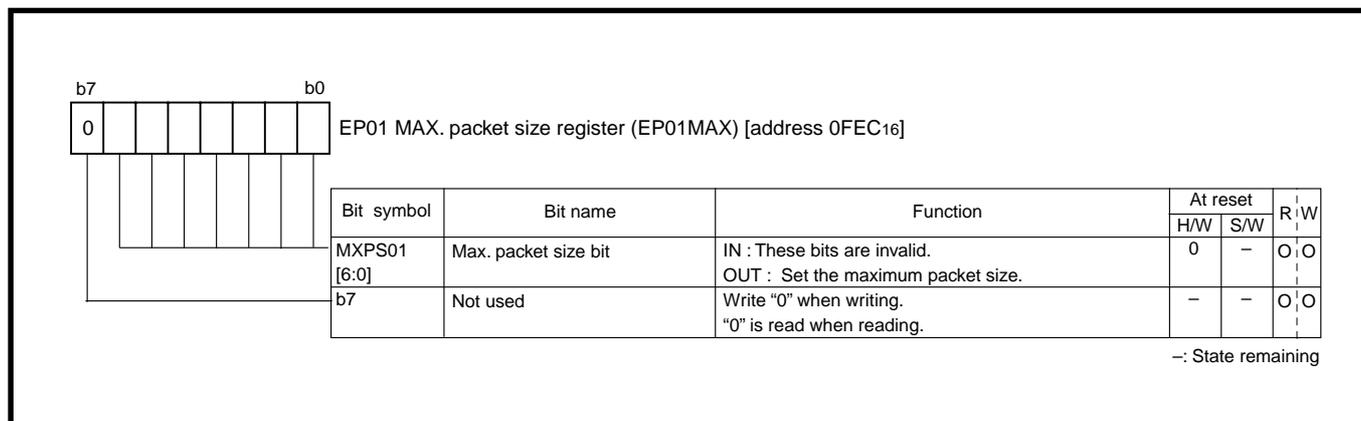


Fig. 3.4.90 Structure of EP01 MAX. packet size register

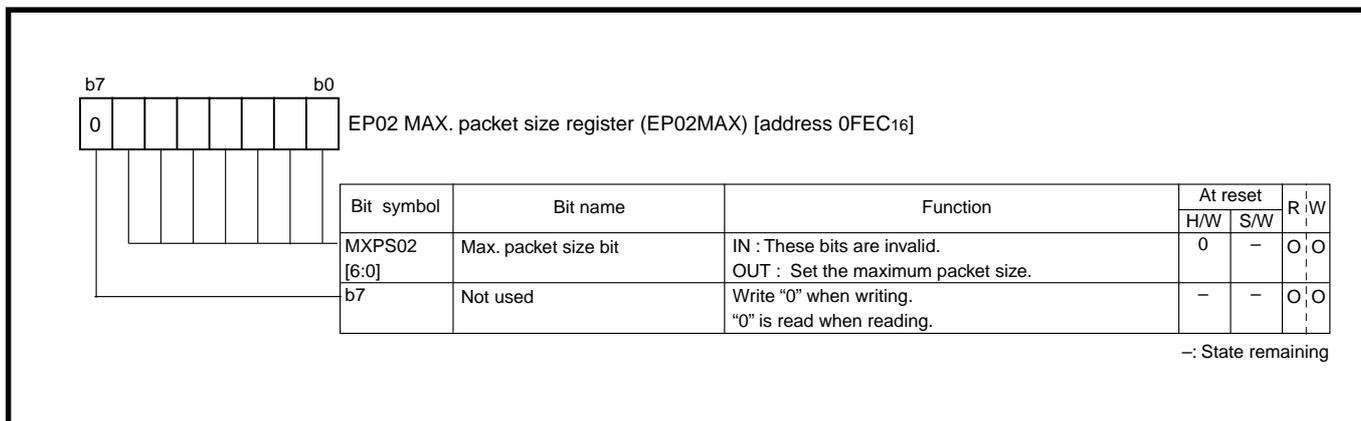


Fig. 3.4.91 Structure of EP02 MAX. packet size register

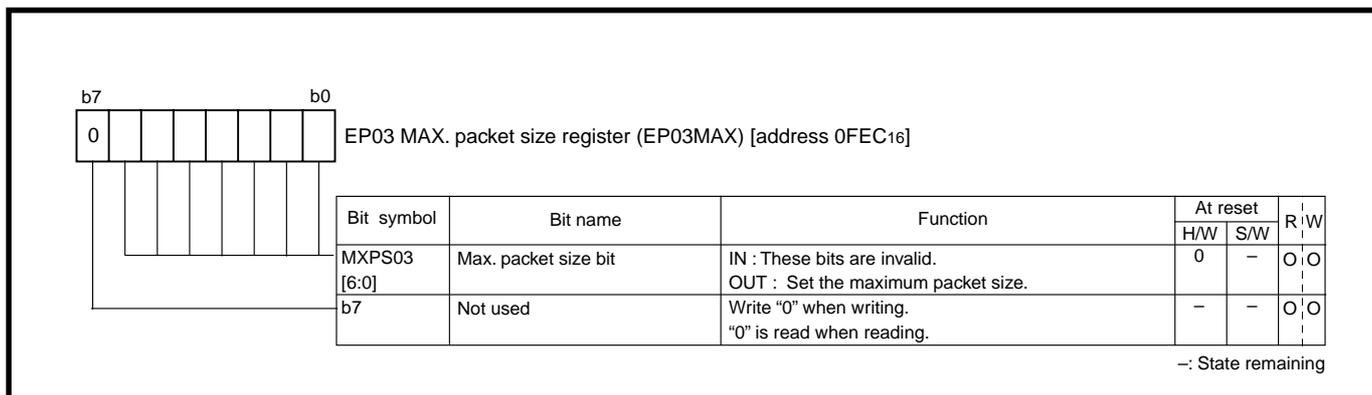


Fig. 3.4.92 Structure of EP03 MAX. packet size register

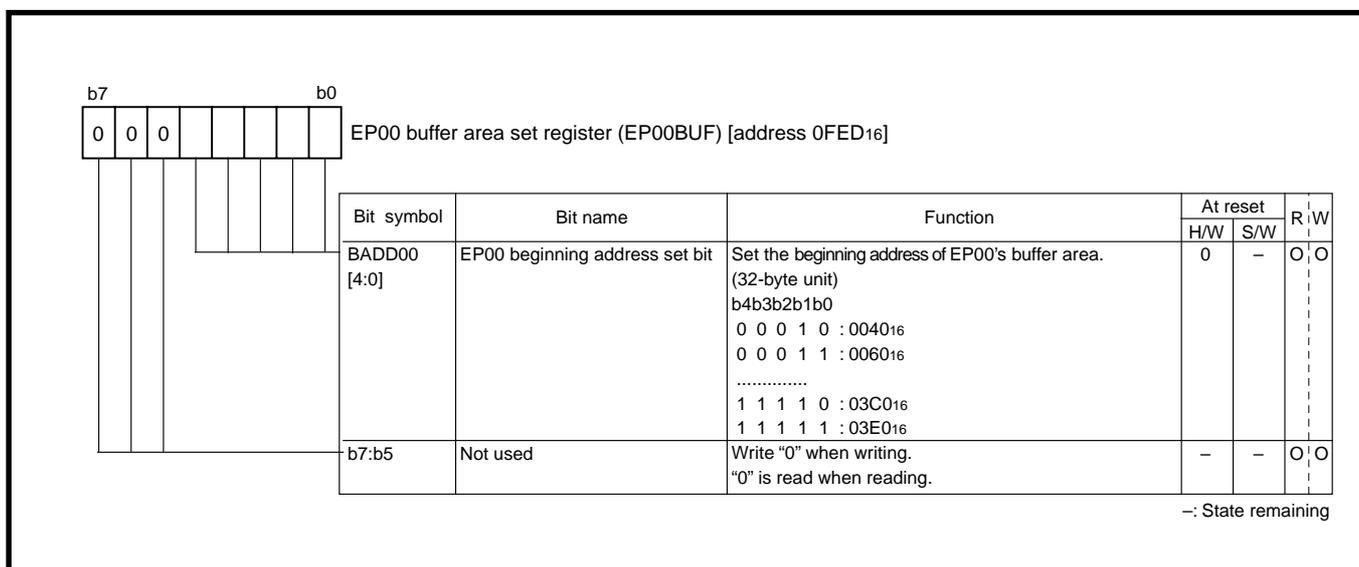


Fig. 3.4.93 Structure of EP00 buffer area set register

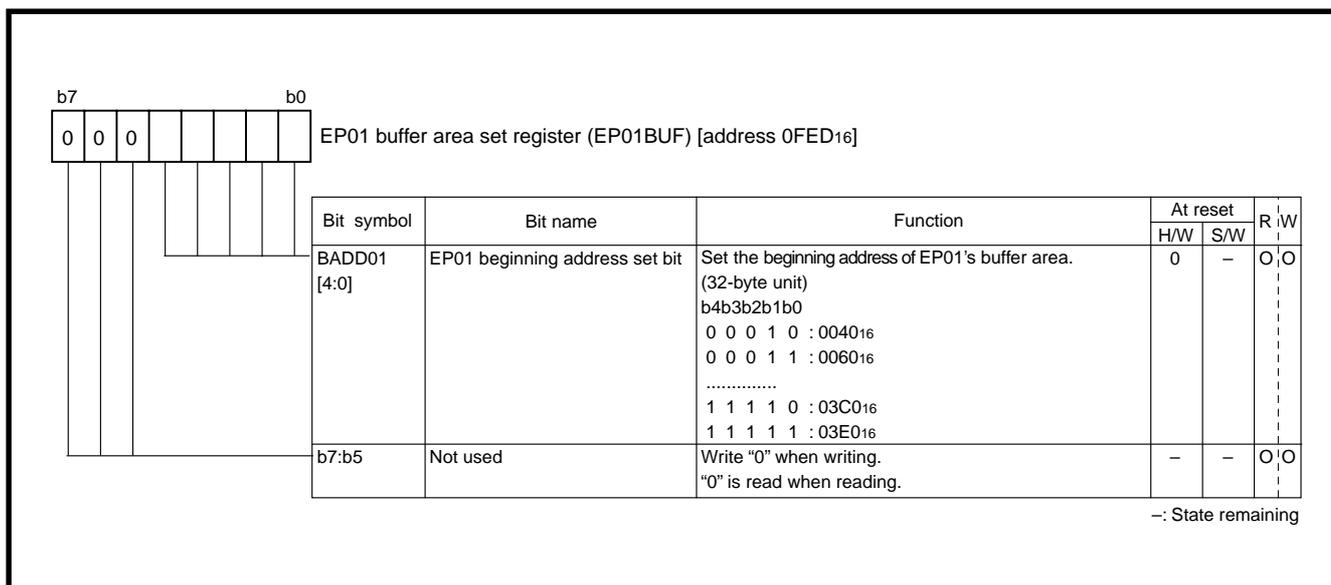


Fig. 3.4.94 Structure of EP01 buffer area set register

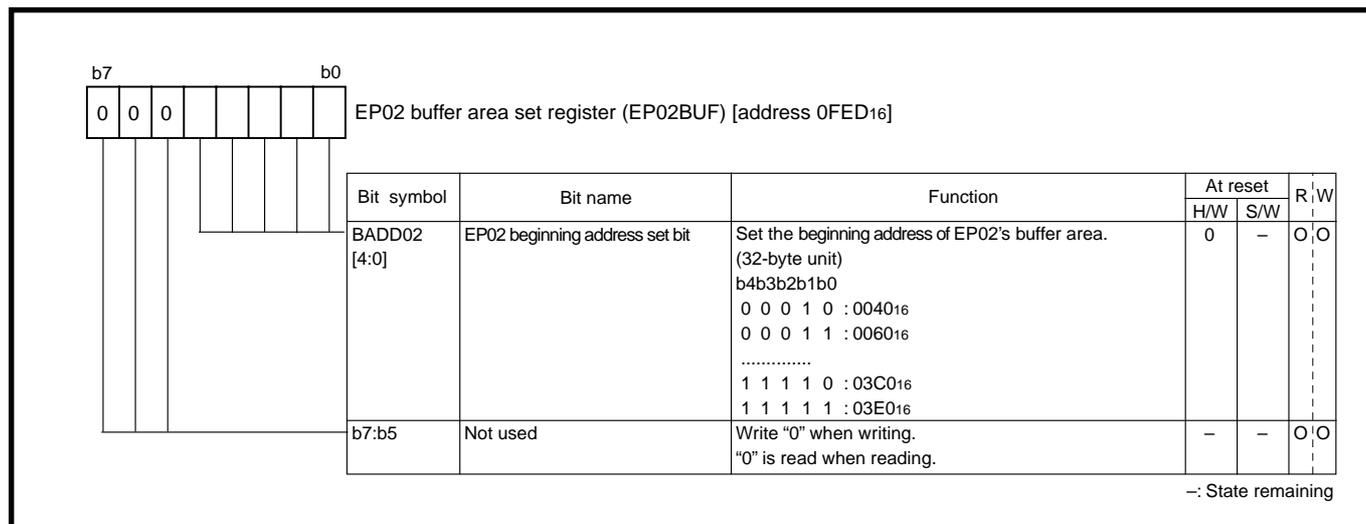


Fig. 3.4.95 Structure of EP02 buffer area set register

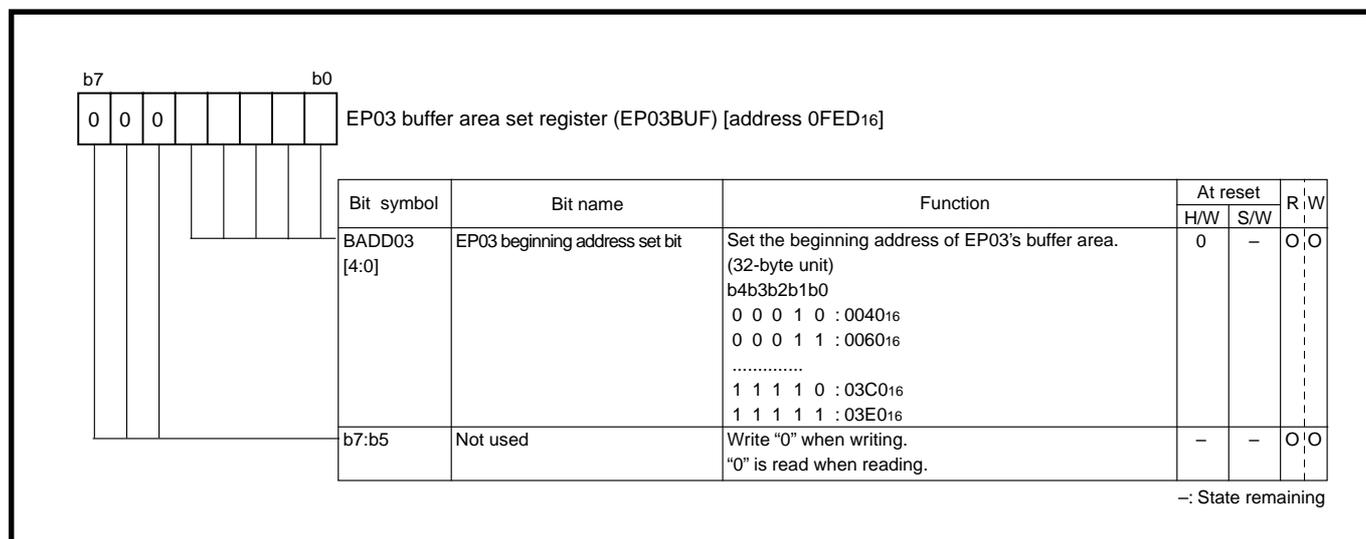


Fig. 3.4.96 Structure of EP03 buffer area set register

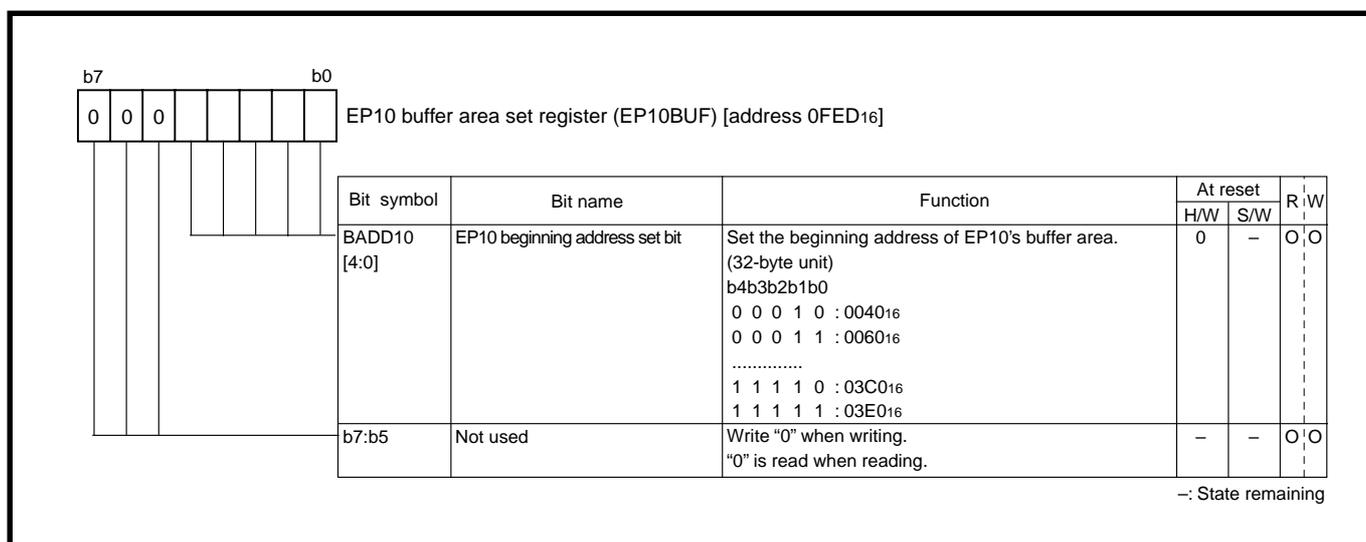


Fig. 3.4.97 Structure of EP10 buffer area set register

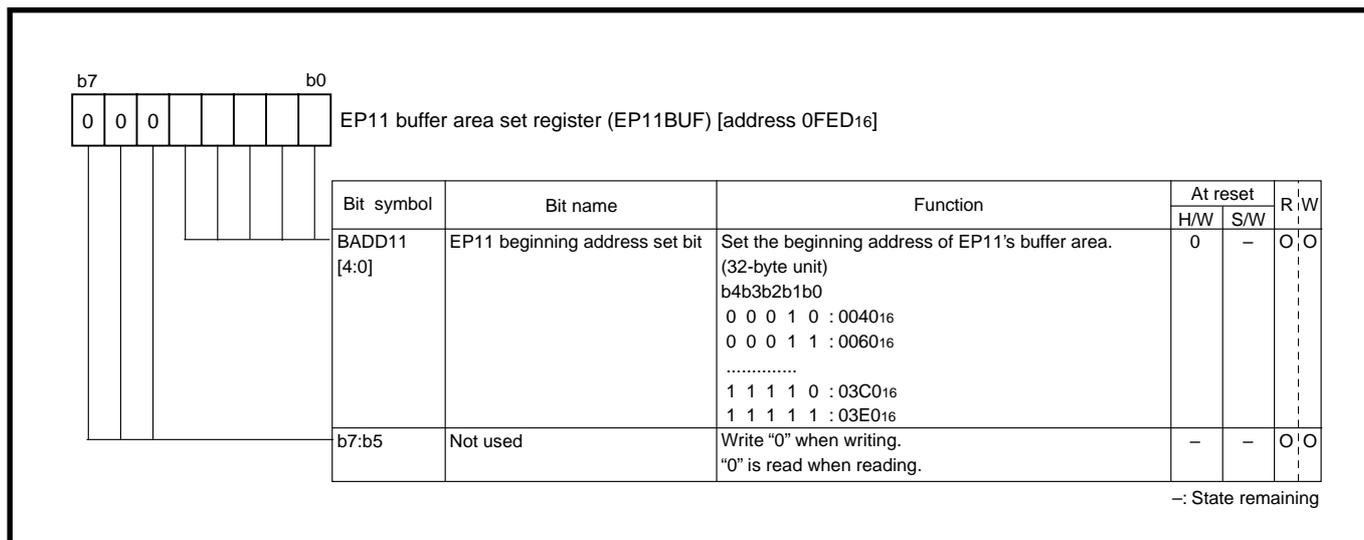


Fig. 3.4.98 Structure of EP11 buffer area set register

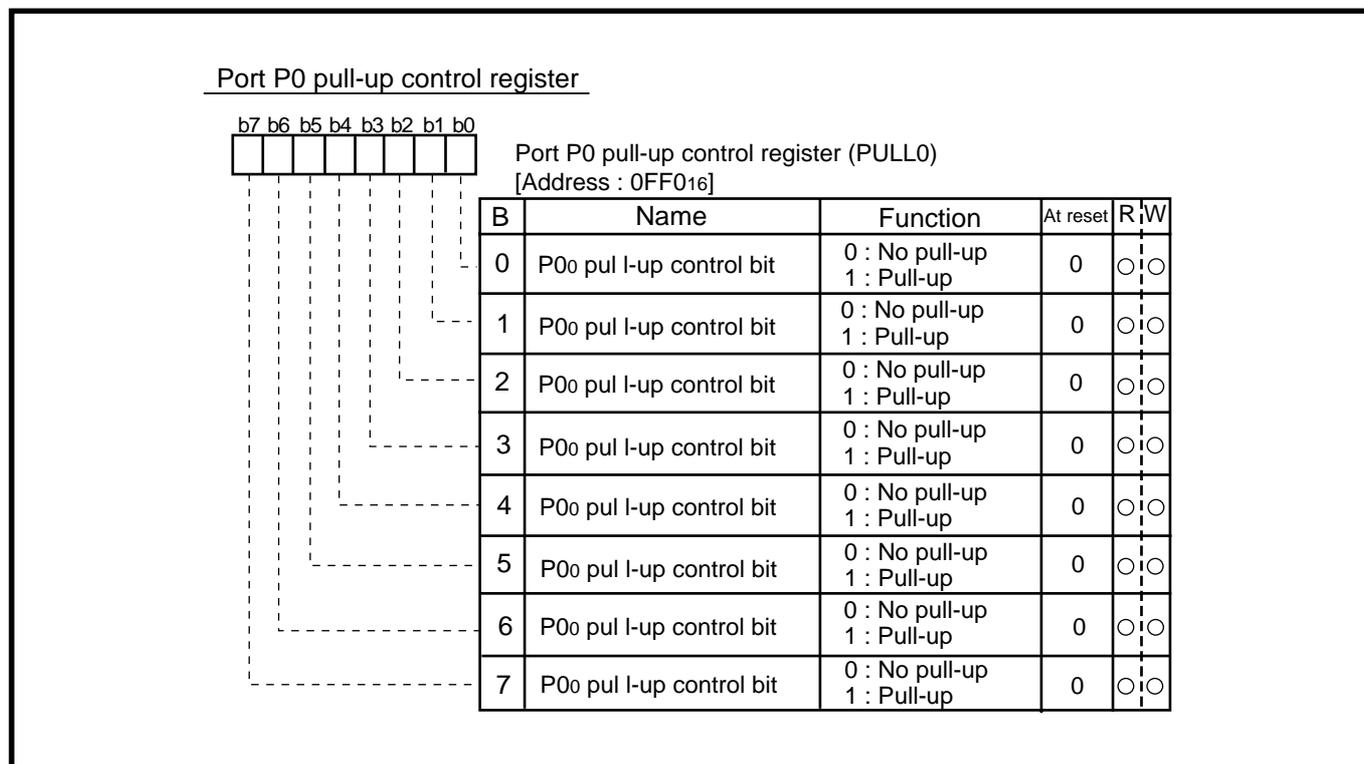


Fig. 3.4.99 Structure of Port P0 pull-up control register

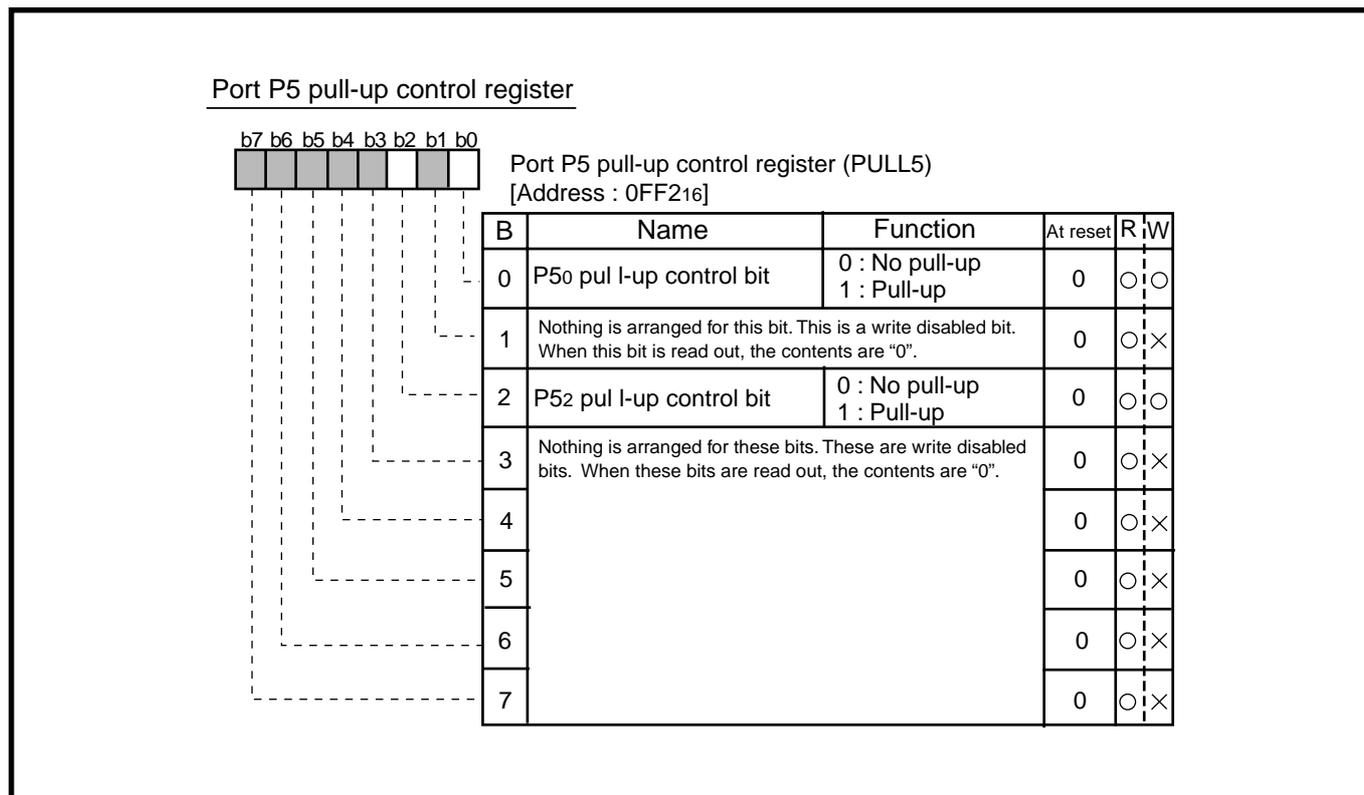


Fig. 3.4.100 Structure of Port P5 pull-up control register

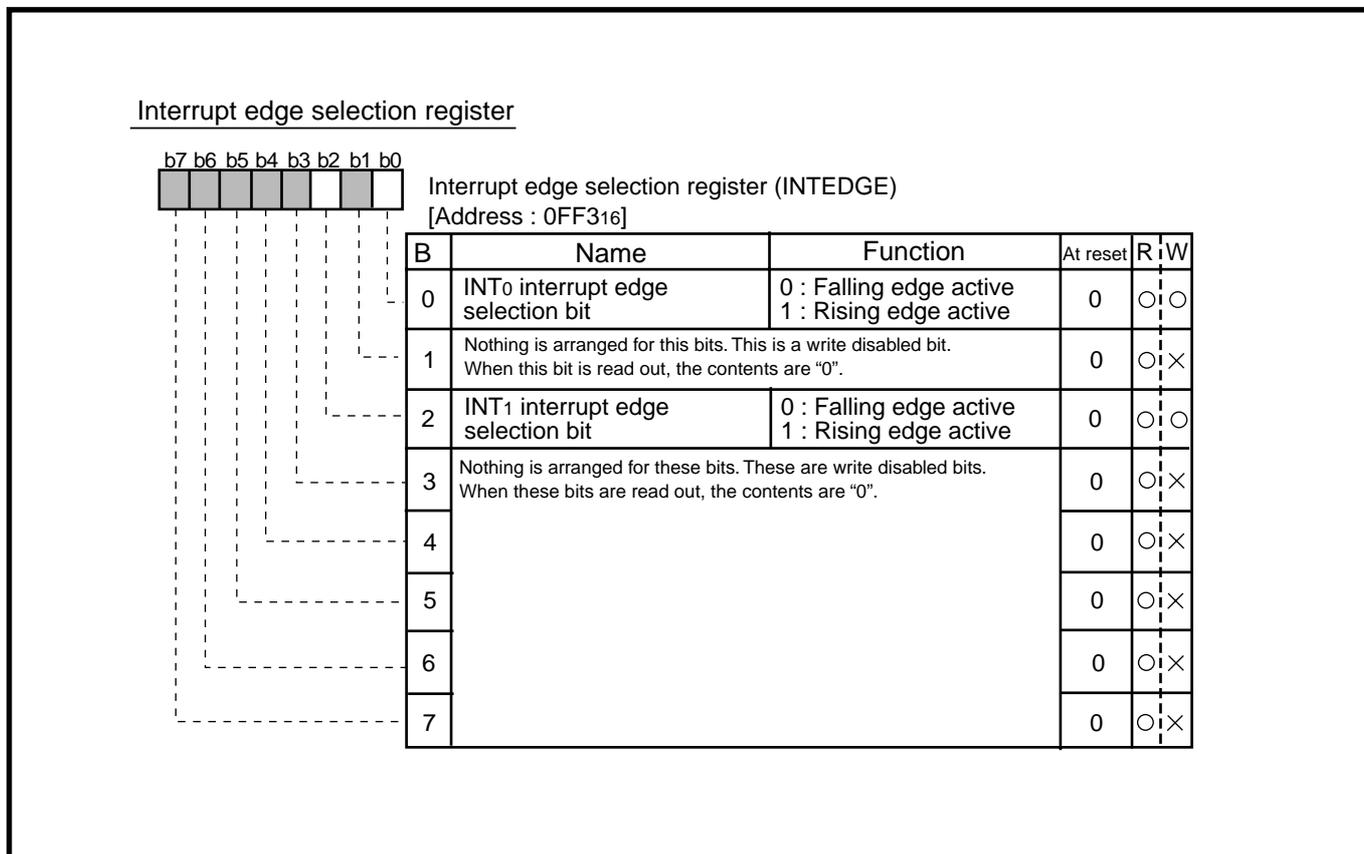


Fig. 3.4.101 Structure of Interrupt edge selection register

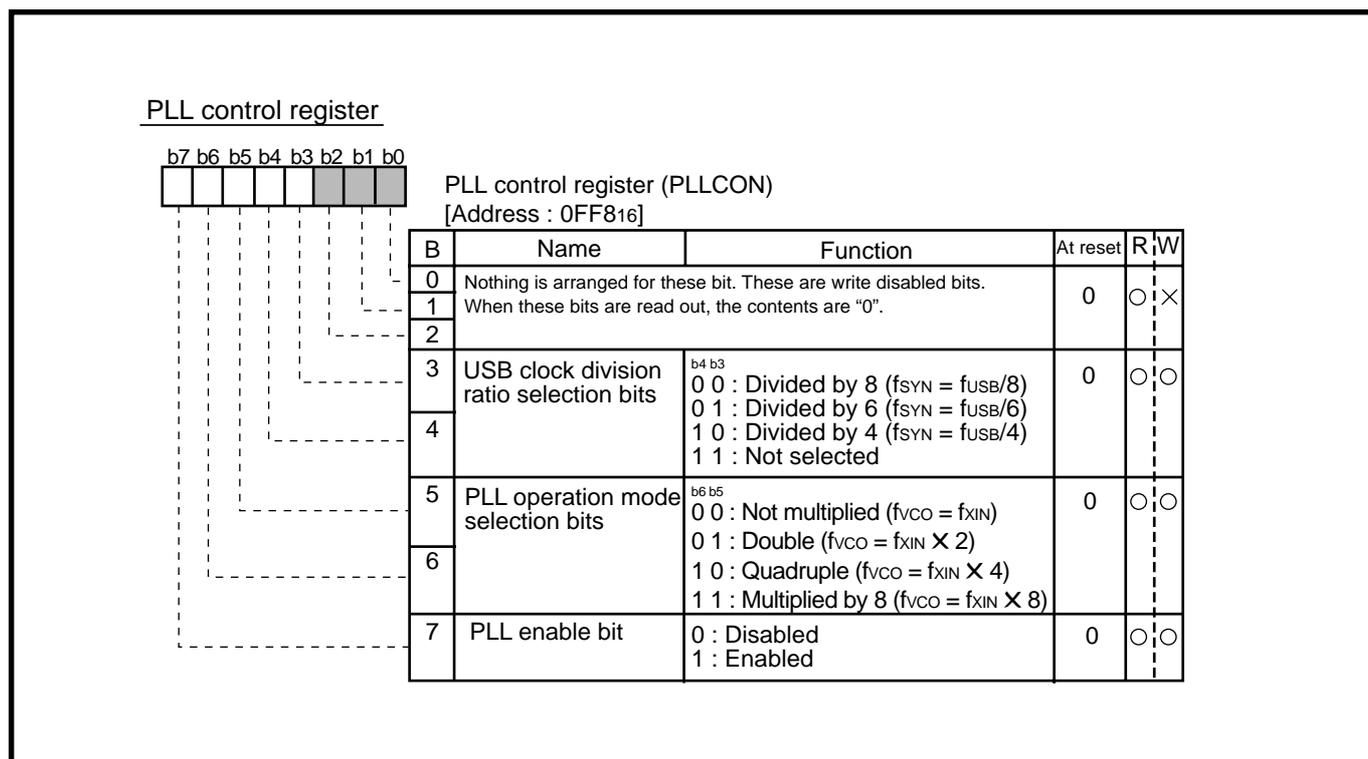


Fig. 3.4.102 Structure of PLL control register

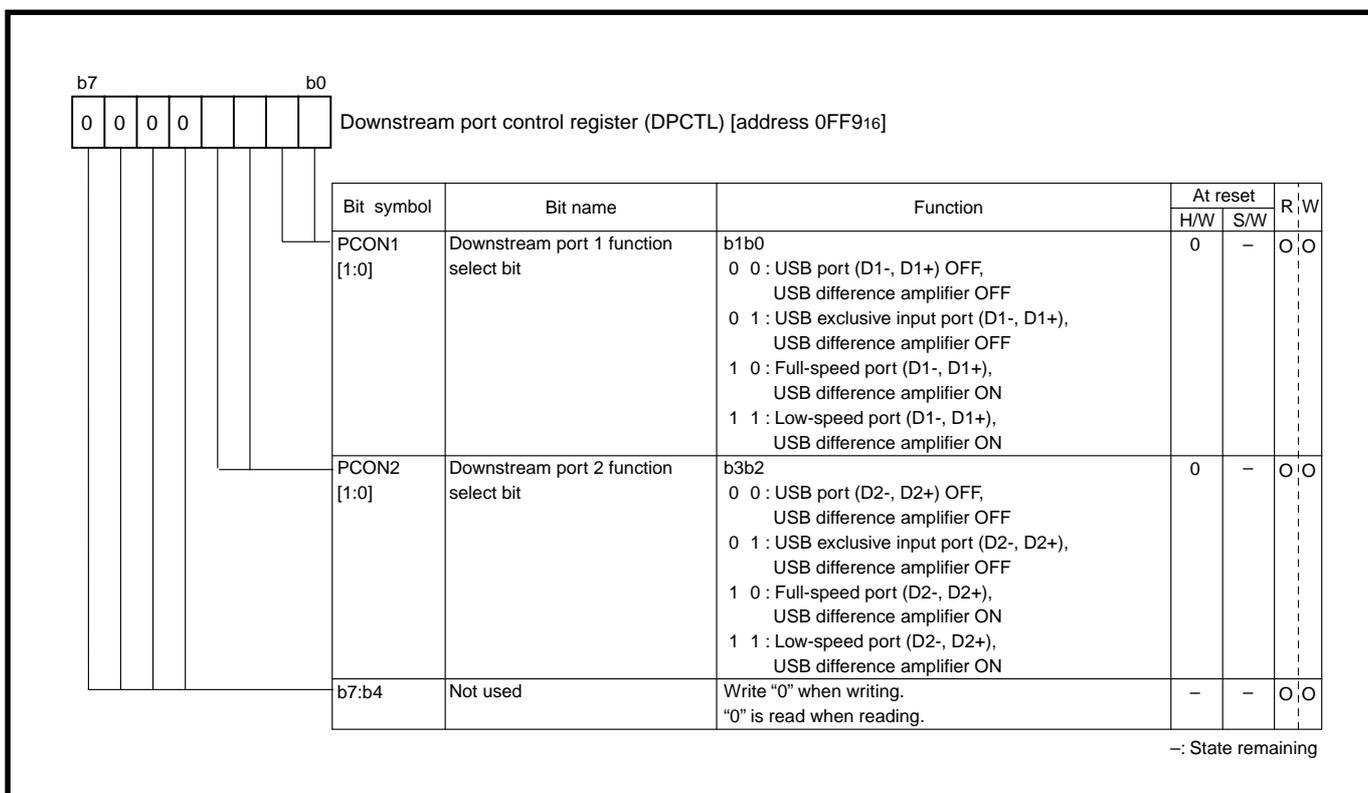


Fig. 3.4.103 Structure of Downstream port control register

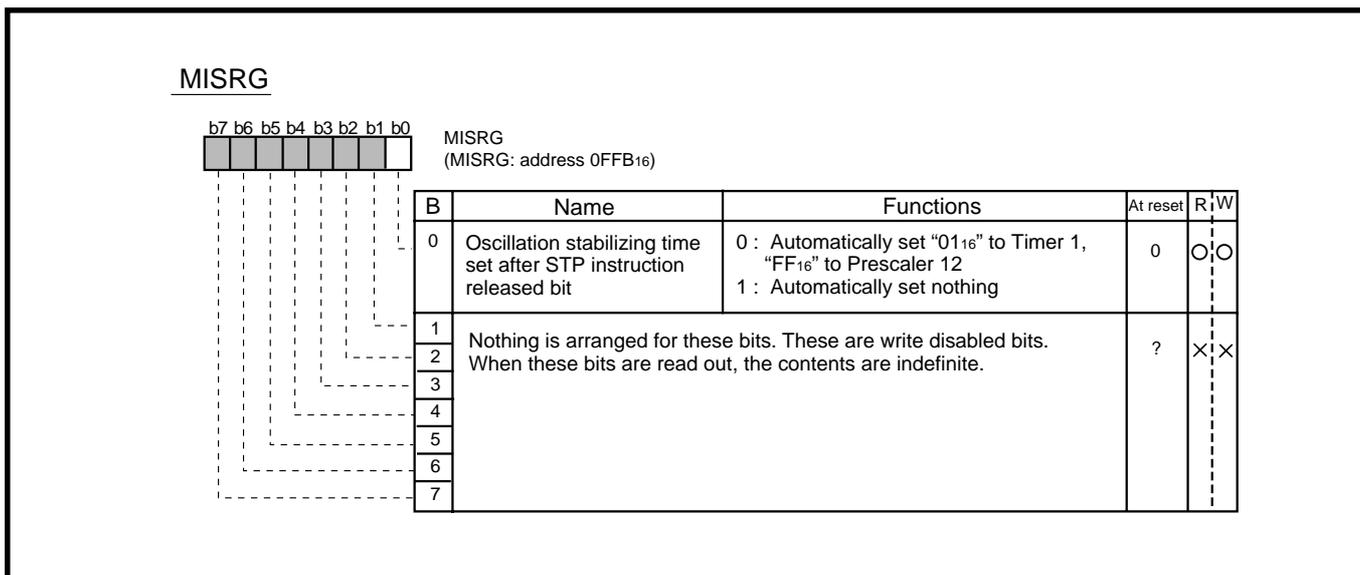


Fig. 3.4.104 Structure of MISRG

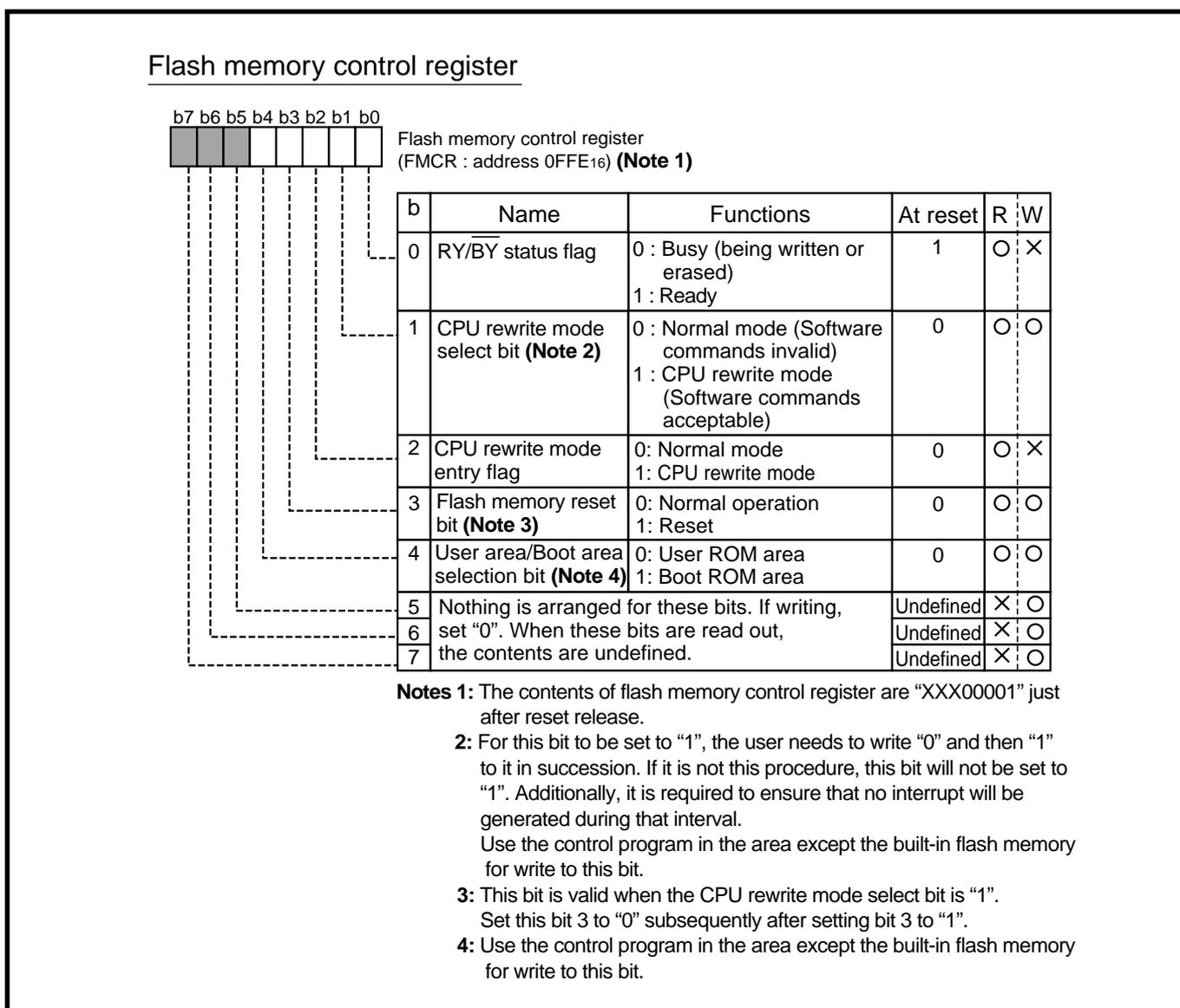
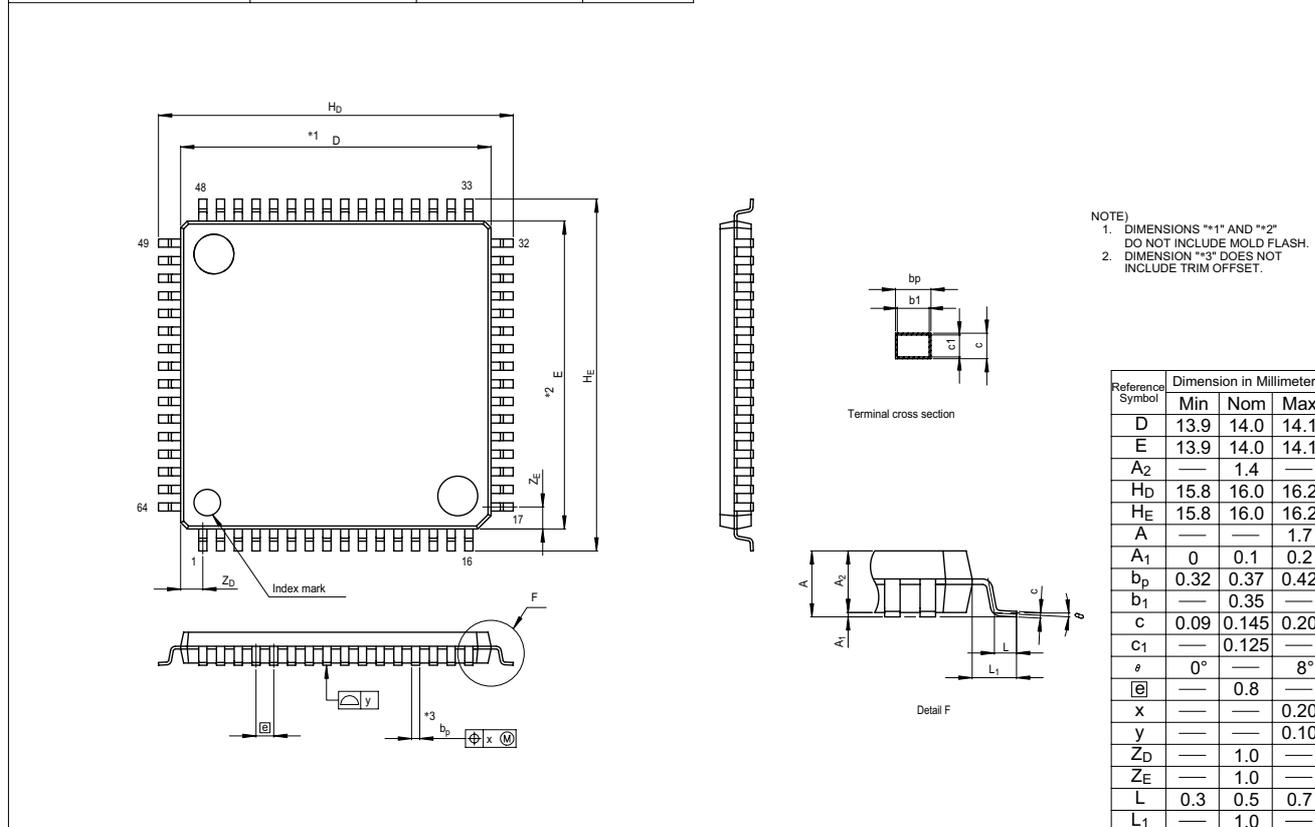


Fig. 3.4.105 Structure of Flash memory control register

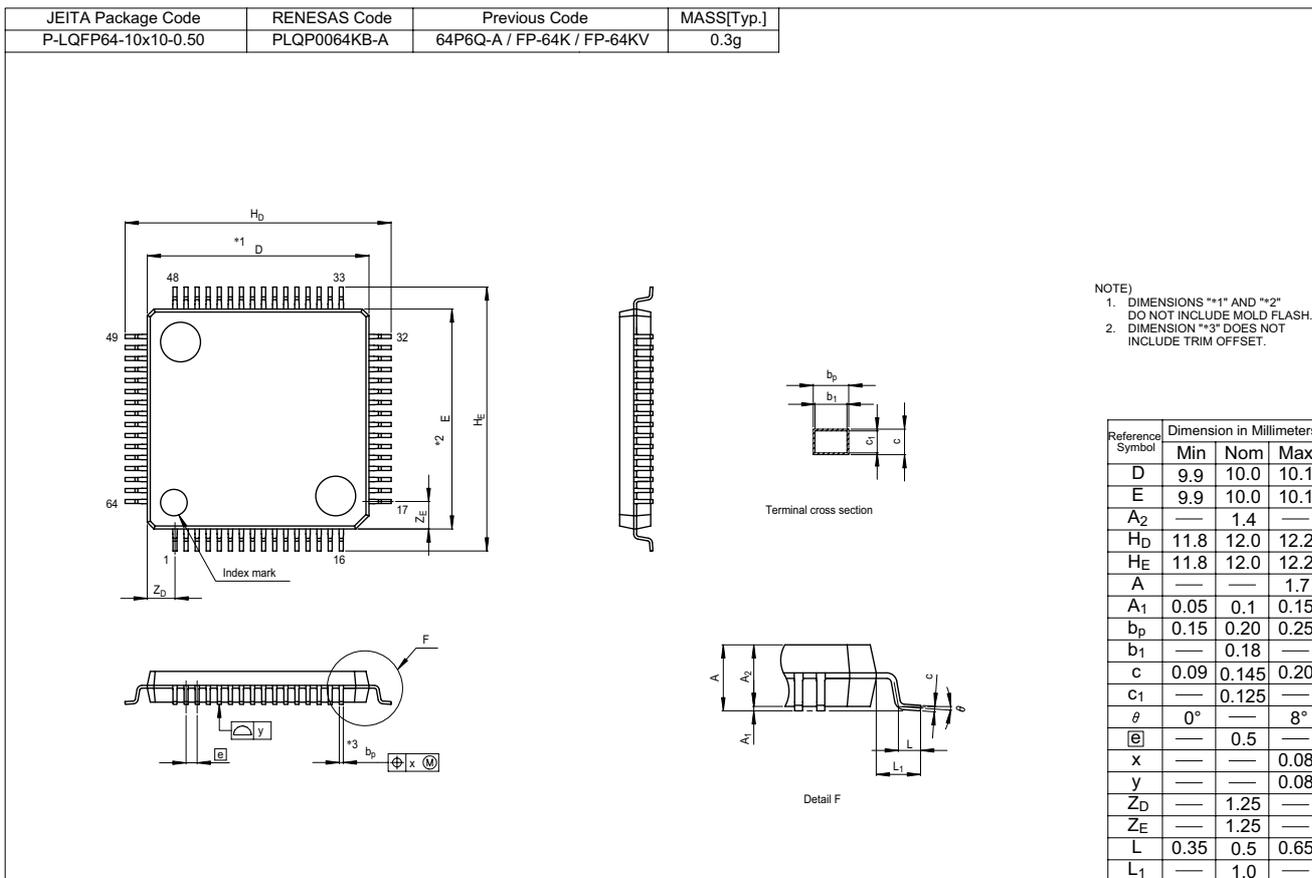
### 3.5 Package outline

#### PLQP0064GA-A

JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-LQFP64-14x14-0.80	PLQP0064GA-A	64P6U-A	0.7g

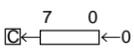


**PLQP0064KB-A**



THIS PAGE IS BLANK FOR REASONS OF LAYOUT.

3.6 Machine instructions

Symbol	Function	Details	Addressing mode																				
			IMP			IMM			A			BIT, A, R			ZP			BIT, ZP, R					
			OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#			
ADC (Note 1) (Note 5)	When T = 0 $A \leftarrow A + M + C$  When T = 1 $M(X) \leftarrow M(X) + M + C$	When T = 0, this instruction adds the contents M, C, and A; and stores the results in A and C. When T = 1, this instruction adds the contents of M(X), M and C; and stores the results in M(X) and C. When T=1, the contents of A remain unchanged, but the contents of status flags are changed. M(X) represents the contents of memory where is indicated by X.				69	2	2										65	3	2			
AND (Note 1)	When T = 0 $A \leftarrow A \wedge M$  When T = 1 $M(X) \leftarrow M(X) \wedge M$	When T = 0, this instruction transfers the contents of A and M to the ALU which performs a bit-wise AND operation and stores the result back in A. When T = 1, this instruction transfers the contents M(X) and M to the ALU which performs a bit-wise AND operation and stores the results back in M(X). When T = 1 the contents of A remain unchanged, but status flags are changed. M(X) represents the contents of memory where is indicated by X.				29	2	2										25	3	2			
ASL		This instruction shifts the content of A or M by one bit to the left, with bit 0 always being set to 0 and bit 7 of A or M always being contained in C.							0A	2	1							06	5	2			
BBC (Note 4)	Ai or Mi = 0?	This instruction tests the designated bit i of M or A and takes a branch if the bit is 0. The branch address is specified by a relative address. If the bit is 1, next instruction is executed.										13 20i	4	2							17 20i	5	3
BBS (Note 4)	Ai or Mi = 1?	This instruction tests the designated bit i of the M or A and takes a branch if the bit is 1. The branch address is specified by a relative address. If the bit is 0, next instruction is executed.										03 20i	4	2							07 20i	5	3
BCC (Note 4)	C = 0?	This instruction takes a branch to the appointed address if C is 0. The branch address is specified by a relative address. If C is 1, the next instruction is executed.																			90	2	2
BCS (Note 4)	C = 1?	This instruction takes a branch to the appointed address if C is 1. The branch address is specified by a relative address. If C is 0, the next instruction is executed.																			B0	2	2
BEQ (Note 4)	Z = 1?	This instruction takes a branch to the appointed address when Z is 1. The branch address is specified by a relative address. If Z is 0, the next instruction is executed.																			F0	2	2
BIT	$A \wedge M$	This instruction takes a bit-wise logical AND of A and M contents; however, the contents of A and M are not modified. The contents of N, V, Z are changed, but the contents of A, M remain unchanged.																24	3	2			
BMI (Note 4)	N = 1?	This instruction takes a branch to the appointed address when N is 1. The branch address is specified by a relative address. If N is 0, the next instruction is executed.																			30	2	2
BNE (Note 4)	Z = 0?	This instruction takes a branch to the appointed address if Z is 0. The branch address is specified by a relative address. If Z is 1, the next instruction is executed.																			D0	2	2

Addressing mode															Processor status register																									
ZP, X			ZP, Y			ABS			ABS, X			ABS, Y			IND		ZP, IND		IND, X		IND, Y		REL		SP		7	6	5	4	3	2	1	0						
OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	N	V	T	B	D	I	Z	C
75	4	2				6D	4	3	7D	5	3	79	5	3							61	6	2	71	6	2							N	V	.	.	.	.	Z	C
35	4	2				2D	4	3	3D	5	3	39	5	3							21	6	2	31	6	2							N	.	.	.	.	.	Z	.
16	6	2				0E	6	3	1E	7	3																						N	.	.	.	.	.	Z	C
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
																																	.	.	.	.	.	.	.	.
										</																														









Symbol	Contents	Symbol	Contents
IMP	Implied addressing mode	+	Addition
IMM	Immediate addressing mode	-	Subtraction
A	Accumulator or Accumulator addressing mode	*	Multiplication
BIT, A	Accumulator bit addressing mode	/	Division
BIT, A, R	Accumulator bit relative addressing mode	∧	Logical OR
ZP	Zero page addressing mode	∨	Logical AND
BIT, ZP	Zero page bit addressing mode	⊕	Logical exclusive OR
BIT, ZP, R	Zero page bit relative addressing mode	—	Negation
ZP, X	Zero page X addressing mode	←	Shows direction of data flow
ZP, Y	Zero page Y addressing mode	X	Index register X
ABS	Absolute addressing mode	Y	Index register Y
ABS, X	Absolute X addressing mode	S	Stack pointer
ABS, Y	Absolute Y addressing mode	PC	Program counter
IND	Indirect absolute addressing mode	PS	Processor status register
ZP, IND	Zero page indirect absolute addressing mode	PCH	8 high-order bits of program counter
IND, X	Indirect X addressing mode	PCL	8 low-order bits of program counter
IND, Y	Indirect Y addressing mode	ADH	8 high-order bits of address
REL	Relative addressing mode	ADL	8 low-order bits of address
SP	Special page addressing mode	FF	FF in Hexadecimal notation
C	Carry flag	nn	Immediate value
Z	Zero flag	zz	Zero page address
I	Interrupt disable flag	M	Memory specified by address designation of any addressing mode
D	Decimal mode flag	M(X)	Memory of address indicated by contents of index register X
B	Break flag	M(S)	Memory of address indicated by contents of stack pointer
T	X-modified arithmetic mode flag	M(ADH, ADL)	Contents of memory at address indicated by ADH and ADL, in ADH is 8 high-order bits and ADL is 8 low-order bits.
V	Overflow flag	M(00, ADL)	Contents of address indicated by zero page ADL
N	Negative flag	Ai	Bit i (i = 0 to 7) of accumulator
		Mi	Bit i (i = 0 to 7) of memory
		OP	Opcode
		n	Number of cycles
		#	Number of bytes

## 3.7 List of instruction code

D7 – D4	D3 – D0	Hexadecimal notation															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	BRK	ORA IND, X	JSR ZP, IND	BBS 0, A	—	ORA ZP	ASL ZP	BBS 0, ZP	PHP	ORA IMM	ASL A	SEB 0, A	—	ORA ABS	ASL ABS	SEB 0, ZP
0001	1	BPL	ORA IND, Y	CLT	BBC 0, A	—	ORA ZP, X	ASL ZP, X	BBC 0, ZP	CLC	ORA ABS, Y	DEC A	CLB 0, A	—	ORA ABS, X	ASL ABS, X	CLB 0, ZP
0010	2	JSR ABS	AND IND, X	JSR SP	BBS 1, A	BIT ZP	AND ZP	ROL ZP	BBS 1, ZP	PLP	AND IMM	ROL A	SEB 1, A	BIT ABS	AND ABS	ROL ABS	SEB 1, ZP
0011	3	BMI	AND IND, Y	SET	BBC 1, A	—	AND ZP, X	ROL ZP, X	BBC 1, ZP	SEC	AND ABS, Y	INC A	CLB 1, A	LDM ZP	AND ABS, X	ROL ABS, X	CLB 1, ZP
0100	4	RTI	EOR IND, X	STP	BBS 2, A	COM ZP	EOR ZP	LSR ZP	BBS 2, ZP	PHA	EOR IMM	LSR A	SEB 2, A	JMP ABS	EOR ABS	LSR ABS	SEB 2, ZP
0101	5	BVC	EOR IND, Y	—	BBC 2, A	—	EOR ZP, X	LSR ZP, X	BBC 2, ZP	CLI	EOR ABS, Y	—	CLB 2, A	—	EOR ABS, X	LSR ABS, X	CLB 2, ZP
0110	6	RTS	ADC IND, X	MUL ZP, X	BBS 3, A	TST ZP	ADC ZP	ROR ZP	BBS 3, ZP	PLA	ADC IMM	ROR A	SEB 3, A	JMP IND	ADC ABS	ROR ABS	SEB 3, ZP
0111	7	BVS	ADC IND, Y	—	BBC 3, A	—	ADC ZP, X	ROR ZP, X	BBC 3, ZP	SEI	ADC ABS, Y	—	CLB 3, A	—	ADC ABS, X	ROR ABS, X	CLB 3, ZP
1000	8	BRA	STA IND, X	RRF ZP	BBS 4, A	STY ZP	STA ZP	STX ZP	BBS 4, ZP	DEY	—	TXA	SEB 4, A	STY ABS	STA ABS	STX ABS	SEB 4, ZP
1001	9	BCC	STA IND, Y	—	BBC 4, A	STY ZP, X	STA ZP, X	STX ZP, Y	BBC 4, ZP	TYA	STA ABS, Y	TXS	CLB 4, A	—	STA ABS, X	—	CLB 4, ZP
1010	A	LDY IMM	LDA IND, X	LDX IMM	BBS 5, A	LDY ZP	LDA ZP	LDX ZP	BBS 5, ZP	TAY	LDA IMM	TAX	SEB 5, A	LDY ABS	LDA ABS	LDX ABS	SEB 5, ZP
1011	B	BCS	LDA IND, Y	JMP ZP, IND	BBC 5, A	LDY ZP, X	LDA ZP, X	LDX ZP, Y	BBC 5, ZP	CLV	LDA ABS, Y	TSX	CLB 5, A	LDY ABS, X	LDA ABS, X	LDX ABS, Y	CLB 5, ZP
1100	C	CPY IMM	CMP IND, X	WIT	BBS 6, A	CPY ZP	CMP ZP	DEC ZP	BBS 6, ZP	INY	CMP IMM	DEX	SEB 6, A	CPY ABS	CMP ABS	DEC ABS	SEB 6, ZP
1101	D	BNE	CMP IND, Y	—	BBC 6, A	—	CMP ZP, X	DEC ZP, X	BBC 6, ZP	CLD	CMP ABS, Y	—	CLB 6, A	—	CMP ABS, X	DEC ABS, X	CLB 6, ZP
1110	E	CPX IMM	SBC IND, X	DIV ZP, X	BBS 7, A	CPX ZP	SBC ZP	INC ZP	BBS 7, ZP	INX	SBC IMM	NOP	SEB 7, A	CPX ABS	SBC ABS	INC ABS	SEB 7, ZP
1111	F	BEQ	SBC IND, Y	—	BBC 7, A	—	SBC ZP, X	INC ZP, X	BBC 7, ZP	SED	SBC ABS, Y	—	CLB 7, A	—	SBC ABS, X	INC ABS, X	CLB 7, ZP

 : 3-byte instruction

 : 2-byte instruction

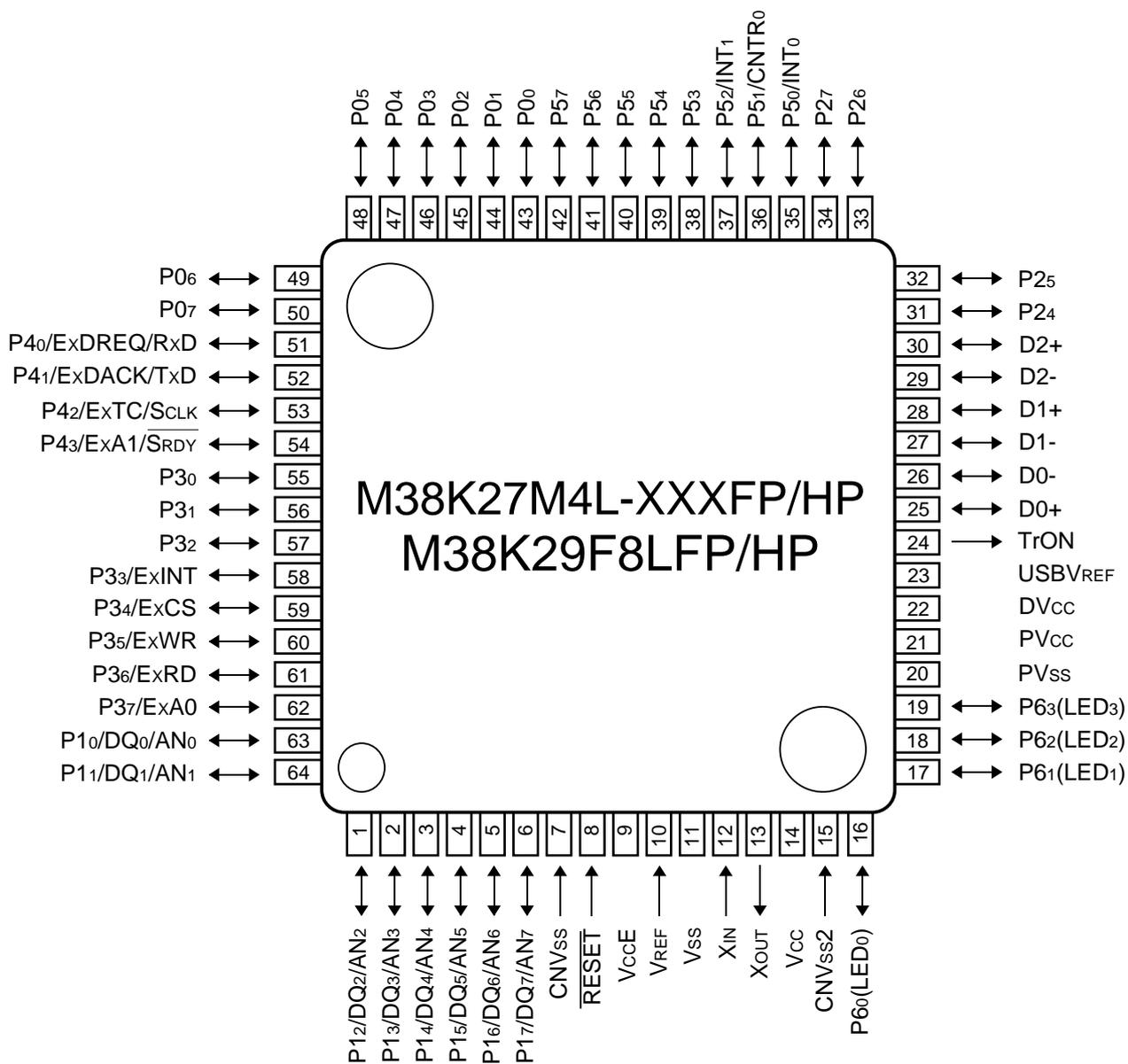
 : 1-byte instruction

## 3.8 SFR memory map

0000 <sub>16</sub>	Port P0 (P0)	0020 <sub>16</sub>	Prescaler 12 (PRE12)
0001 <sub>16</sub>	Port P0 direction register (P0D)	0021 <sub>16</sub>	Timer 1 (T1)
0002 <sub>16</sub>	Port P1 (P1)	0022 <sub>16</sub>	Timer 2 (T2)
0003 <sub>16</sub>	Port P1 direction register (P1D)	0023 <sub>16</sub>	Timer X mode register (TM)
0004 <sub>16</sub>	Port P2 (P2)	0024 <sub>16</sub>	Prescaler X (PREX)
0005 <sub>16</sub>	Port P2 direction register (P2D)	0025 <sub>16</sub>	Timer X (TX)
0006 <sub>16</sub>	Port P3 (P3)	0026 <sub>16</sub>	Transmit/Receive buffer register (TB/RB)
0007 <sub>16</sub>	Port P3 direction register (P3D)	0027 <sub>16</sub>	Serial I/O status register (SIOSTS)
0008 <sub>16</sub>	Port P4 (P4)	0028 <sub>16</sub>	HUB interrupt source enable register (HUBICON)
0009 <sub>16</sub>	Port P4 direction register (P4D)	0029 <sub>16</sub>	HUB interrupt source register (HUBIREQ)
000A <sub>16</sub>	Port P5 (P5)	002A <sub>16</sub>	HUB down stream port index register (HUBINDEX)
000B <sub>16</sub>	Port P5 direction register (P5D)	002B <sub>16</sub>	HUB port field register 1 (DPXREG1)
000C <sub>16</sub>	Port P6 (P6)	002C <sub>16</sub>	HUB port field register 2 (DPXREG2)
000D <sub>16</sub>	Port P6 direction register (P6D)	002D <sub>16</sub>	HUB port field register 3 (DPXREG3)
000E <sub>16</sub>	Reserved (Note)	002E <sub>16</sub>	Reserved (Note)
000F <sub>16</sub>	Reserved (Note)	002F <sub>16</sub>	Reserved (Note)
0010 <sub>16</sub>	USB control register (USBCON)	0030 <sub>16</sub>	EXB interrupt source enable register (EXBICON)
0011 <sub>16</sub>	USB function/Hub enable register (USBAE)	0031 <sub>16</sub>	EXB interrupt source register (EXBIREQ)
0012 <sub>16</sub>	USB function address register (USBA0)	0032 <sub>16</sub>	Reserved (Note)
0013 <sub>16</sub>	USB HUB address register (USBA1)	0033 <sub>16</sub>	EXB index register (EXBINDEX)
0014 <sub>16</sub>	Frame number register Low (FNUML)	0034 <sub>16</sub>	Register window 1 (EXBREG1)
0015 <sub>16</sub>	Frame number register High (FNUMH)	0035 <sub>16</sub>	Register window 2 (EXBREG2)
0016 <sub>16</sub>	USB interrupt source enable register (USBICON)	0036 <sub>16</sub>	AD control register (ADCON)
0017 <sub>16</sub>	USB interrupt source register (USBIREQ)	0037 <sub>16</sub>	AD conversion register 1 (AD1)
0018 <sub>16</sub>	Endpoint index register (USBINDEX)	0038 <sub>16</sub>	AD conversion register 2 (AD2)
0019 <sub>16</sub>	Endpoint field register 1 (EPXXREG1)	0039 <sub>16</sub>	Watchdog timer control register (WDTCN)
001A <sub>16</sub>	Endpoint field register 2 (EPXXREG2)	003A <sub>16</sub>	Reserved (Note)
001B <sub>16</sub>	Endpoint field register 3 (EPXXREG3)	003B <sub>16</sub>	CPU mode register (CPUM)
001C <sub>16</sub>	Endpoint field register 4 (EPXXREG4)	003C <sub>16</sub>	Interrupt request register 1 (IREQ1)
001D <sub>16</sub>	Endpoint field register 5 (EPXXREG5)	003D <sub>16</sub>	Interrupt request register 2 (IREQ2)
001E <sub>16</sub>	Endpoint field register 6 (EPXXREG6)	003E <sub>16</sub>	Interrupt control register 1 (ICON1)
001F <sub>16</sub>	Endpoint field register 7 (EPXXREG7)	003F <sub>16</sub>	Interrupt control register 2 (ICON2)
0FE0 <sub>16</sub>	Serial I/O control register (SIOCON)	0FF0 <sub>16</sub>	Port P0 pull-up control register (PULL0)
0FE1 <sub>16</sub>	UART control register (UARTCON)	0FF1 <sub>16</sub>	Reserved (Note)
0FE2 <sub>16</sub>	Baud rate generator (BRG)	0FF2 <sub>16</sub>	Port P5 pull-up control register (PULL5)
0FE3 <sub>16</sub>	Reserved (Note)	0FF3 <sub>16</sub>	Interrupt edge selection register (INTEDGE)
0FE4 <sub>16</sub>	Reserved (Note)	0FF4 <sub>16</sub>	Reserved (Note)
0FE5 <sub>16</sub>	Reserved (Note)	0FF5 <sub>16</sub>	Reserved (Note)
0FE6 <sub>16</sub>	Reserved (Note)	0FF6 <sub>16</sub>	Reserved (Note)
0FE7 <sub>16</sub>	Reserved (Note)	0FF7 <sub>16</sub>	Reserved (Note)
0FE8 <sub>16</sub>	Reserved (Note)	0FF8 <sub>16</sub>	PLL control register (PLLCON)
0FE9 <sub>16</sub>	Reserved (Note)	0FF9 <sub>16</sub>	Downstream port control register (DPCTL)
0FEA <sub>16</sub>	Reserved (Note)	0FFA <sub>16</sub>	Reserved (Note)
0FEB <sub>16</sub>	Reserved (Note)	0FFB <sub>16</sub>	MISRG
0FEC <sub>16</sub>	Endpoint field register 8 (EPXXREG8)	0FFC <sub>16</sub>	Reserved (Note)
0FED <sub>16</sub>	Endpoint field register 9 (EPXXREG9)	0FFD <sub>16</sub>	Reserved (Note)
0FEE <sub>16</sub>	Reserved (Note)	0FFE <sub>16</sub>	Flash memory control register (FMCR)
0FEF <sub>16</sub>	Reserved (Note)	0FFF <sub>16</sub>	Reserved (Note)

**Note:** Do not write any data to these addresses, because these areas are reserved.

### 3.9 Pin configurations



REVISION HISTORY

38K2 GROUP USER'S MANUAL

Rev.	Date	Description	
		Page	Summary
1.0	2/13/03		First Edition
2.0	10/15/06	All pages	Package names "64P6U-A" → "PLQP0064GA-A" revised Package names "64P6Q-A" → "PLQP0064KB-A" revised 38K2 group (Standard) deleted
		Chapter 1	
		94	Fig. 137 revised
		97	CLOCK GENERATING CIRCUIT; "No external resistor is needed .... resistor exists on-chip." → "No external resistor is needed .... depending on conditions.)
		98	Fig. 141; Pulled up added, NOTE added Fig. 144 revised
		128	NOTES ON USAGE; Power Source Voltage, USB Communication added
		Chapter 2	
		3	Fig. 2.1.3; "Do not set bits of .... If writing to these bits, write "0"." added
		Chapter 3	
		20	3.2 deleted
		35	3.3.6 (3) USB Communication added
		48	Fig. 3.5.2; "Do not set bits of .... If writing to these bits, write "0"." added
		92, 93	3.6 Package outline revised

---

**RENESAS 8-BIT SINGLE-CHIP MICROCOMPUTER  
USER'S MANUAL  
38K2 Group**

**Publication Data : Rev.1.00 Feb 13, 2003  
Rev.2.00 Oct 15, 2006**

**Published by : Sales Strategic Planning Div.  
Renesas Technology Corp.**

# 38K2 Group User's Manual



Renesas Electronics Corporation

1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ09B0338-0200