User's Manual

μPD784038, 784038Y Subseries

16-Bit Single-Chip Microcontrollers

Hardware

μPD784031
μPD784035
μPD784036
μPD784037
μPD784038
μPD78P4038

μPD784031Y
μPD784035Y
μPD784036Y
μPD784037Y
μPD784038Y
μPD78P4038Y

μPD784031(A)
μPD784035(A)
μPD784036(A)

Document No. U11316EJ4V1UD00 (4th edition)
Date Published January 2003 N CP(K)

Printed in Japan

[MEMO]

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

Strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it once, when it has occurred. Environmental control must be adequate. When it is dry, humidifier should be used. It is recommended to avoid using insulators that easily build static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work bench and floor should be grounded. The operator should be grounded using wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with semiconductor devices on it.

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

No connection for CMOS device inputs can be cause of malfunction. If no connection is provided to the input pins, it is possible that an internal input level may be generated due to noise, etc., hence causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using a pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND with a resistor, if it is considered to have a possibility of being an output pin. All handling related to the unused pins must be judged device by device and related specifications governing the devices.

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

Power-on does not necessarily define initial status of MOS device. Production process of MOS does not define the initial operation status of the device. Immediately after the power source is turned ON, the devices with reset function have not yet been initialized. Hence, power-on does not guarantee out-pin levels, I/O settings or contents of registers. Device is not initialized until the reset signal is received. Reset operation must be executed immediately after power-on for devices having reset function.

Purchase of NEC Electronics I²C components conveys a license under the Philips I²C Patent Rights to use these components in an I²C system, provided that the system conforms to the I²C Standard Specification as defined by Philips.

These commodities, technology or software, must be exported in accordance with the export administration regulations of the exporting country. Diversion contrary to the law of that country is prohibited.

- **The information in this document is current as of September, 2002. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC 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 NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".

  The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.

  "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.

  "Special": 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, life support systems and medical equipment for life support, etc.

The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact an NEC Electronics sales representative in advance to determine NEC Electronics' willingness to support a given application.

(Note)

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

(2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

M8E 02. 11-1

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC Electronics product in your application, please contact the NEC Electronics office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

NEC Electronics America, Inc. (U.S.)
Santa Clara, California
Tel: 408-588-6000
800-366-9782
Fax: 408-588-6130
800-729-9288

NEC Electronics (Europe) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 01
Fax: 0211-65 03 327

- **Sucursal en España**
  Madrid, Spain
  Tel: 091-504 27 87
  Fax: 091-504 28 60

- **Succursale Française**
  Vélizy-Villacoublay, France
  Tel: 01-30-67 58 00
  Fax: 01-30-67 58 99

- **Filiale Italiana**
  Milano, Italy
  Tel: 02-66 75 41
  Fax: 02-66 75 42 99

- **Branch The Netherlands**
  Eindhoven, The Netherlands
  Tel: 040-244 58 45
  Fax: 040-244 45 80

- **Tyskland Filial**
  Taeby, Sweden
  Tel: 08-63 80 820
  Fax: 08-63 80 388

- **United Kingdom Branch**
  Milton Keynes, UK
  Tel: 01908-691-133
  Fax: 01908-670-290

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Shanghai, Ltd.
Shanghai, P.R. China
Tel: 021-6841-1138
Fax: 021-6841-1137

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-2719-2377
Fax: 02-2719-5951

NEC Electronics Singapore Pte. Ltd.
Novena Square, Singapore
Tel: 6253-8311
Fax: 6250-3583

J02.11

Major Revisions in This Edition (1/2)

Page	Description
Throughout	Addition of the following special grade products to the target products • μPD784031GC(A)-3B9, 784035GC(A)-×××-3B9, 784036GC(A)-×××-3B9 Deletion of the following packages • μPD784031GC-3B9, 784031GK-BE9, 784035GC-×××-3B9, 784035GK-×××-BE9, 784036GC-×××-3B9, 784036GK-×××-BE9, 784037GC-×××-3B9, 784037GK-×××-BE9, 784038GC-×××-3B9, 784038GK-×××-BE9, 78P4038GC-3B9, 78P4038GC-×××-3B9, 78P4038GC-×××-8BT, 78P4038GK-BE9, 78P4038GK-×××-BE9, 78P4038KK-T • μPD784031YGC-3B9, 784031YGK-BE9, 784035YGC-×××-3B9, 784035YGK-×××-BE9, 784036YGC-×××-3B9, 784036YGK-×××-BE9, 784037YGC-×××-3B9, 784037YGK-×××-BE9, 784038YGC-×××-3B9, 784038YGK-×××-BE9, 78P4038YGC-3B9, 78P4038YGC-×××-3B9, 78P4038YGC-×××-8BT, 78P4038YGK-BE9, 78P4038YGK-×××-BE9, 78P4038YKK-T Addition of the following packages • μPD784031GK-9EU, 784035GK-×××-9EU, 784036GK-×××-9EU, 784037GK-×××-9EU, 784038GK-×××-9EU, 78P4038GK-9EU • μPD784031YGK-9EU, 784035YGK-×××-9EU, 784036YGK-×××-9EU, 784037YGK-×××-9EU, 784038YGK-×××-9EU, 78P4038YGK-9EU
	CHAPTER 1 GENERAL
p.39	• Update of **78K/IV Series Product Development Diagram**
p.41	• Addition and deletion of products in **1.2 ORDERING INFORMATION AND QUALITY GRADES**
p.53	• Addition of **1.7 DIFFERENCES BETWEEN STANDARD-GRADE PRODUCTS AND SPECIAL-GRADE PRODUCTS**
	CHAPTER 8 TIMER/COUNTER 0
p.245	• Addition of caution on compare register CR00 match interrupt to **8.9 CAUTIONS**
	CHAPTER 9 TIMER/COUNTER 1
p.283	• Addition of caution on compare register CR10 match interrupt to **9.8 CAUTIONS**
	CHAPTER 10 TIMER/COUNTER 2
p.357	• Addition of caution on compare register CR20 match interrupt to **10.10 CAUTIONS**
	CHAPTER 14 A/D CONVERTER
p.390	Modification of description in **Figure 14-3 A/D Converter Mode Register (ADM) Format**
	CHAPTER 17 ASYNCHRONOUS SERIAL INTERFACE/3-WIRE SERIAL I/O
p.445	• Addition of caution on successive reception in 3-wire serial I/O mode to **17.5 CAUTIONS**

Major Revisions in This Edition (2/2)

Page	Description
	CHAPTER 18 3-WIRE/2-WIRE SERIAL I/O MODE
pp.452, 453	• Modification of **Figure 18-6 3-Wire Serial I/O Mode Timing**
	• **18.6 CAUTIONS**
p.460	• Addition of caution on transmit data write in 3-wire serial I/O mode
p.460	• Addition of caution on serial clock count operation in 3-wire serial I/O mode
p.461	• Addition of caution on serial clock output in 3-wire serial I/O mode
p.462	• Addition of caution on successive reception in 3-wire serial I/O mode
	CHAPTER 21 EDGE DETECTION FUNCTION
p.498	Addition of description to **21.2 EDGE DETECTION FOR PINS P20, P25 AND P26**
p.684	Addition of **CHAPTER 28 ELECTRICAL SPECIFICATIONS**
p.711	Addition of **CHAPTER 29 PACKAGE DRAWINGS**
p.714	Addition of **CHAPTER 30 RECOMMENDED SOLDERING CONDITIONS**
	APPENDIX A DIFFERENCES WITH μPD784026 SUBSERIES
p.718	Addition of description in **Table A-1 Differences with μPD784026 Subseries**
p.719	Modification of description in **APPENDIX B DEVELOPMENT TOOLS**
p.734	Modification of description in **APPENDIX C EMBEDDED SOFTWARE**

The mark ★ shows major revised points.

PREFACE

Intended Readership

This manual is intended for user engineers who understand the functions of the μPD784038, 784038Y Subseries and wish to design application systems using these subseries.

★ The following are the target products in the μPD784038, 784038Y Subseries.

- Standard grade: μPD784031, 784035, 784036, 784037, 784038, 78P4038, 784031Y, 784035Y, 784036Y, 784037Y, 784038Y, 78P4038Y
- Special grade: μPD784031(A) ,784035(A), 784036(A)

Purpose

The purpose of this manual is to give users an understanding of the various hardware functions of the μPD784038, 784038Y Subseries.

Organization

The μPD784038, 784038Y Subseries user's manual is divided into two volumes – hardware (this manual) and instruction.

Hardware	Instruction
Pin functions	CPU functions
Internal block functions	Addressing
Interrupts	Instruction set
Other on-chip peripheral functions	
Electrical specifications	

Certain operating precautions apply to these products.
These precautions are stated at the relevant points in the text of each chapter, and are also summarized at the end of each chapter. Be sure to read them.

How to Read This Manual

Readers are required to have a general knowledge of electrical and logic circuits and microcontrollers.

- Unless otherwise specified
  The μPD784038 in the μPD784038 Subseries is treated as the representative model of the mask ROM models, the μPD784031 is treated as the representative model of the ROM-less model, and the μPD78P4038 is treated as the representative model of the PROM models.

- If there are functional differences
  The function of each model is described individually.
  Even in this case, the μPD784038 Subseries is treated as the representative model. If you use the μPD784038Y Subseries, take the μPD784031, 784035, 784036, 784037, 784038, and 78P4038 as the μPD784031Y, 784035Y, 784036Y, 784037Y, 784038Y, and 78P4038Y, respectively.

★ **The application examples presented in this manual are for the "standard" quality models in general-purpose electronic systems. If you wish to use the applications presented in this manual for electronic systems that require "special" quality models, thoroughly study the parts and circuits to be actually used, and their quality grade.**

♦ V_{DD} and V_{SS} pins
  This product is highly immune to noise and its power supply pins are classified into V_{DD} and V_{SS}, as follows. If there is no need to classify the power supply pins, V_{DD} is used as the representative pin name.
  • Positive power supply and GND of ports: V_{DD0}, V_{SS0}
  • Positive power supply and GND of function blocks other than ports: V_{DD1}, V_{SS1}

♦ For a general understanding of the functions:
  → Read in accordance with the **CONTENTS**.

♦ To find out about differences from the μPD784026 Subseries:
  → See **APPENDIX A DIFFERENCES WITH μPD784026 SUBSERIES**.

♦ If the device operates strangely after debugging:
  → Cautions are summarized at the end of each chapter, so refer to the cautions for the relevant function.

♦ To check the details of a register when the register name is known:
  → Use **APPENDIX D REGISTER INDEX**.

♦ For the details of the instruction functions:
  → Refer to the separate **78K/IV Series Instruction User's Manual (U10905E)**.

♦ To find out about the electrical characteristics:
  → Refer to **CHAPTER 28 ELECTRICAL SPECIFICATIONS**.

♦ To find out about application examples of each function:
  → Refer to Application Note separately available.

Differences between μPD784038 Subseries and μPD784038Y Subseries

The functions of the μPD784038 Subseries and μPD784038Y Subseries are the same except the clocked serial interface.

Caution

The clocked serial interface is described in the following two chapters:

- CHAPTER 18 3-/2-WIRE SERIAL I/O MODE
- CHAPTER 19 I^2C BUS MODE (μPD784038Y Subseries only)

Also refer to the general explanation on the serial interface in CHAPTER 16.

Legend

Significance in data notation	:	Higher digit on left, lower digit on right
Active-low notation	:	$\overline{\times\times\times}$ (Line above pin or signal name)
Note	:	Description of note in the text
Caution	:	Item to be especially noted
Remark	:	Supplementary information
Numeric notations	:	Binary ..... ××××B or ××××
		Decimal ..... ××××
		Hexadecimal..... ××××H

Register Notation

	7	6	5	4	③	2	1	0
EDC	B	1	0	×	A	1	0	×

Where the bit number is marked with a circle, the bit name is reserved for NEC Electronics assembler and is defined as an sfr variable by the #pragma sfr directive for C compiler.

Write Operation	Read Operation
0 or 1 is written. The operation is not affected by either value.	0 or 1 is read.
0 must be written	0 is read.
1 must be written	1 is read.
A value is written according to the function to be used.	A value is read according to the operating status.

Code combinations marked "Setting prohibited" in the register notations in the text must not be written.

Easily confused characters : 0 (Zero), O (Letter O)

: 1 (One), l (Lower-case letter L), I (Upper-case letter I)

Related Documents The related documents in this publication may include preliminary versions. However, preliminary versions are not marked as such.

Documents Related to Devices

Document Name	Document No.
μPD784038, 784038Y Subseries User's Manual - Hardware	This manual
78K/IV Series Application Note - Software Fundamentals	U10095E
78K/IV Series User's Manual - Instructions	U10905E

Documents Related to Development Tools (User's Manuals)

Document Name		Document No.
RA78K4 Assembler Package	Operation	U15254E
	Language	U15255E
	Structured Assembler Preprocessor	U11743E
CC78K4 C Compiler	Operation	U15557E
	Language	U15556E
SM78K Series Ver. 2.30 or Later System Simulator	Operation (Windows® Based)	U15373E
	External Part User Open Interface Specification	U15802E
ID78K Series Integrated Debugger Ver. 2.30 or Later	Operation (Windows Based)	U15185E
RX78K4 Real-time OS	Fundamentals	U10603E
	Installation	U10604E
Project Manager Ver 3.12 or Later (Windows Based)		U14610E

Caution The related documents listed above are subject to change without notice. Be sure to use the latest version of each document for designing.

Documents Related to Development Hardware Tools (User's Manuals)

Document Name	Document No.
IE-78K4-NS In-Circuit Emulator	U13356E
IE-784038-NS-EM1 Emulation Board	U13760E
IE-784000-R In-Circuit Emulator	U12903E
IE-784038-R-EM1 Emulation Board	U11383E

Documents Related to PROM Writing (User's Manuals)

Document Name		Document No.
PG-1500 PROM Programmer		U11940E
PG-1500 Controller	PC-9800 Series (MS-DOS)-Based	EEU-1291
	IBM PC Series (PC DOS)-Based	U10540E

Other Related Documents

Document Name	Document No.
SEMICONDUCTOR SELECTION GUIDE - Products & Packages -	X13769E
Semiconductor Device Mounting Technology Manual	C10535E
Quality Grades on NEC Semiconductor Devices	C11531E
NEC Semiconductor Device Reliability/Quality Control System	C10983E
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892E

Caution The related documents listed above are subject to change without notice. Be sure to use the latest version of each document for designing.

CONTENTS

LIST OF FIGURES (1/12)

LIST OF FIGURES (2/12)

LIST OF FIGURES (4/12)

LIST OF FIGURES (6/12)

LIST OF FIGURES (7/12)

LIST OF FIGURES (8/12)

LIST OF FIGURES (9/12)

LIST OF FIGURES (10/12)

LIST OF FIGURES (11/12)

LIST OF FIGURES (12/12)

LIST OF TABLES (1/3)

LIST OF TABLES (2/3)

LIST OF TABLES (3/3)

CHAPTER 1 GENERAL

The μPD784038 Subseries comprises 78K/IV Series products that can perform input/output directly with analog signals. The 78K/IV Series comprises 16-bit single-chip microcontrollers equipped with a high-performance CPU that has a function such as accessing a 1-Mbyte memory space. The μPD784038 Subseries is upward-compatible with the 78K/II Series, and has pin compatibility with μPD78234 Subseries of the 78K/II Series.

The μPD784038 incorporates 128-Kbyte mask ROM and 4,352-byte RAM, plus high-performance timer/counters, an 8-bit A/D converter, 8-bit D/A converter, PWM output function, two independent serial interface channels, etc.

The μPD784031 is a ROM-less model of the μPD784038 but is provided with RAM of 2,048 bytes.

The μPD784035 is based on the μPD784038 but is provided with 48 Kbytes of mask ROM and 2,048 bytes of RAM.

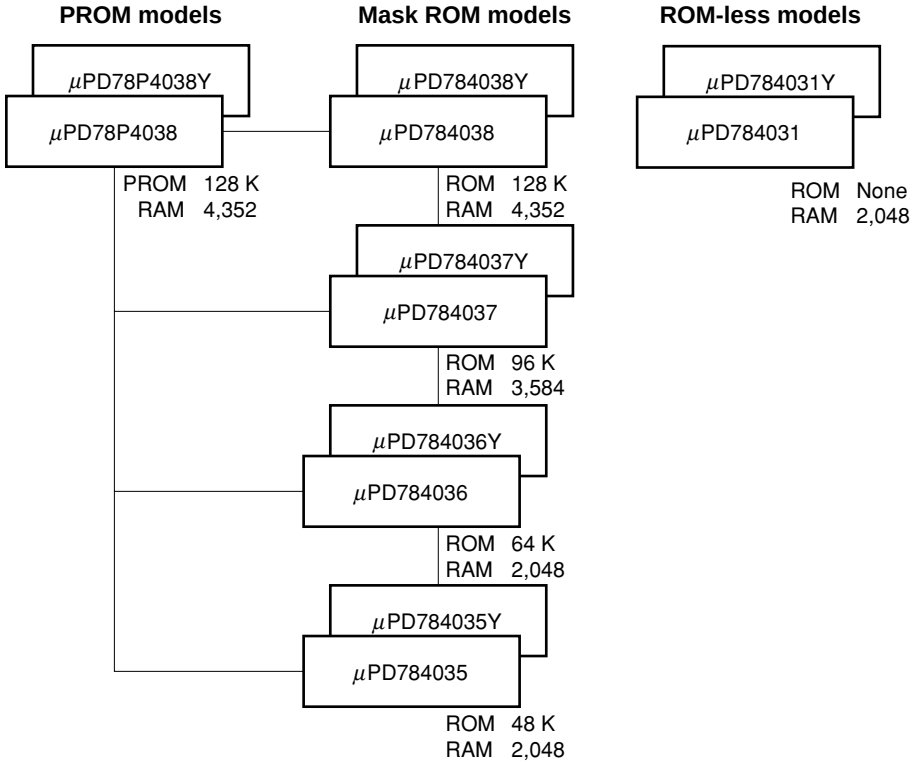
The μPD784036 is based on the μPD784038 but is provided with 64 Kbytes of mask ROM and 2,048 bytes of RAM.

The μPD784037 is based on the μPD784038 but is provided with 96 Kbytes of mask ROM and 3,584 bytes of RAM.

The μPD78P4038 replaces the mask ROM of the μPD784038 with PROM.

The μPD784038Y Subseries is based on the μPD784038 Subseries but is provided with an I²C bus control function.

The relation among these models is as shown below.



These models can be used in the following fields:

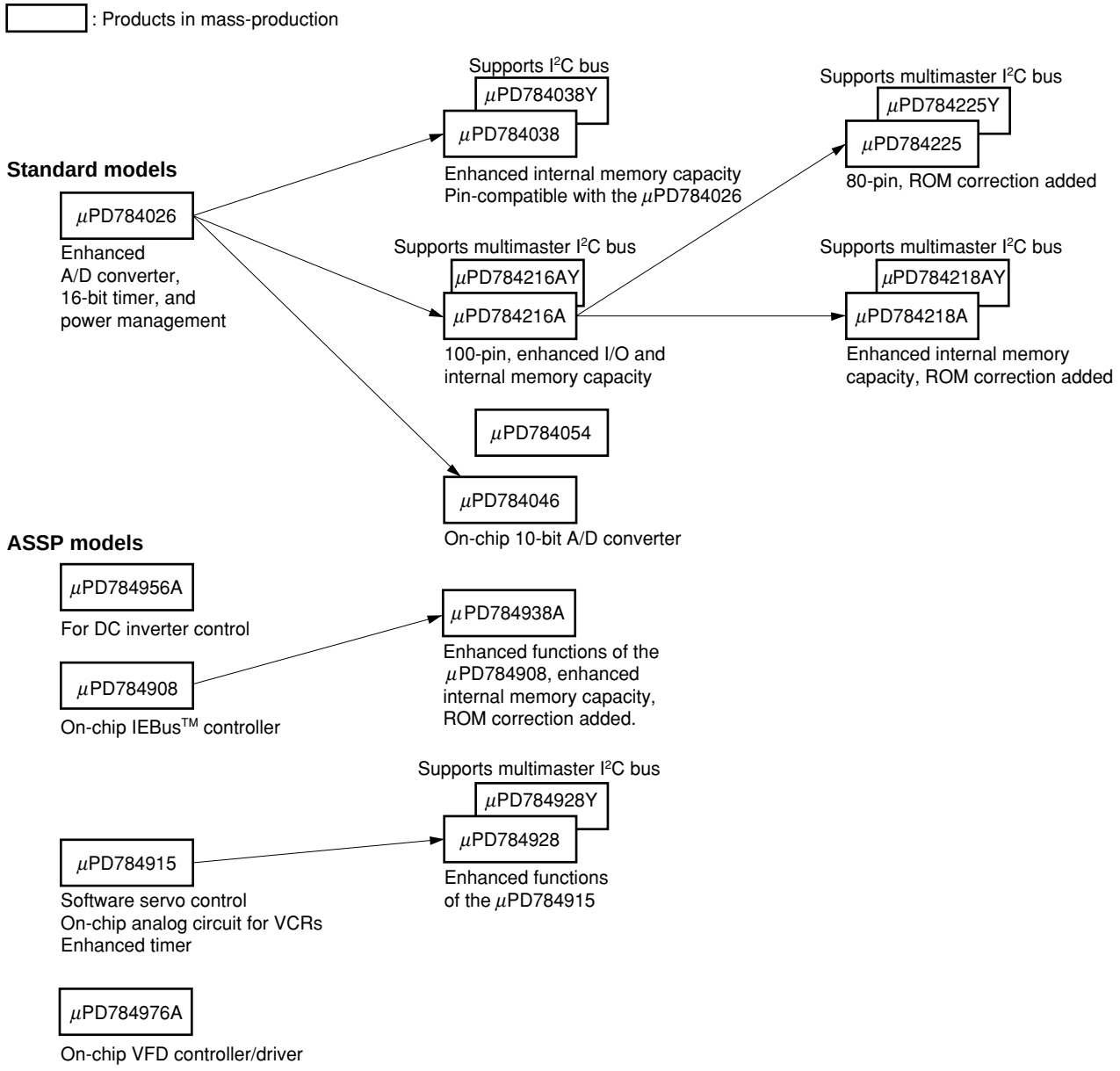
<μPD784038 Subseries>

- LBP
- Auto-focus camera
- PPC
- Printer
- Electronic typewriter
- Air conditioner
- Electronic musical instruments
- Cellular phone

<μPD784038Y Subseries>

- Cellular phone
- Cordless telephone
- Audio/visual systems

★ **78K/IV Series Product Development Diagram**



Remark VFD (Vacuum Florescent Display) is referred to as FIP™ (Florescent Indicator Panel) in some documents, but the functions of the two are the same.

1.1 FEATURES

- 78K/IV Series
- Pin-compatible with μPD78234 Subseries and μPD784026 Subseries
- Internal memory of μPD78234 Subseries and μPD784026 Subseries expanded
- High-speed instruction execution
  - Minimum instruction execution time (32-MHz operation): 125 ns
- Instruction set suitable for control applications
- Data memory extension function (1-Mbyte memory space: 2 bank specification pointers)
- Interrupt controller (4-level priority system)
  - Vectored interrupt service/macro service/context switching
- Standby functions: HALT/STOP/IDLE modes
- Internal memory:
  - ROM
    - Mask ROM : 128 Kbytes (μPD784038)
      96 Kbytes (μPD784037)
      64 Kbytes (μPD784036)
      48 Kbytes (μPD784035)
      Not provided (μPD784031)
    - PROM : 128 Kbytes (μPD78P4038)
  - RAM : 4,352 bytes (μPD784038, 78P4038)
    3,584 bytes (μPD784037)
    2,048 bytes (μPD784031, 784035, 784036)
- I/O pins
  - μPD784035, 784036, 784037, 784038, 78P4038 : 64
    Software programmable pull-up : 54 inputs
    Direct LED drive capability : 24 outputs
    Direct transistor drive capability : 8 outputs
  - μPD784031: 46
    Software programmable pull-up : 34 inputs
    Direct LED drive capability : 8 outputs
    Direct transistor drive capability : 8 outputs
- Serial interface
  - UART/IOE (3-wire serial I/O): 2 channels (with on-chip baud rate generator)
  - CSI (3-wire serial I/O, 2-wire serial I/O, I²C bus [Note]): 1 channel

Note μPD784038Y Subseries only

- Real-time output ports (combination with timer/counter allows independent control of 2-system stepping motors)
- A/D converter (8-bit resolution × 8 channels)
- D/A converter (8-bit resolution × 2 channels)
- PWM outputs (12-bit resolution × 2 channels)
- High-performance timer/counter
  - Timer/counter (16 bits) × 3 units
  - Timer (16 bits) × 1 unit
- Watchdog timer: 1 channel
- Clock output function: f_{CLK}, $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/16$ can be selected (other than μPD784031)

★ 1.2 ORDERING INFORMATION AND QUALITY GRADES

1.2.1 Ordering Information

(1) μPD784038 Subseries

Part Number	Package	Internal ROM
μPD784031GC-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	None
μPD784031GK-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	None
μPD784031GC(A)-3B9	80-pin plastic QFP (14 x 14, 2.7 mm thickness)	None
μPD784035GC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Mask ROM
μPD784035GK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Mask ROM
μPD784035GC(A)-×××-3B9	80-pin plastic QFP (14 x 14, 2.7 mm thickness)	Mask ROM
μPD784036GC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Mask ROM
μPD784036GK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Mask ROM
μPD784036GC(A)-×××-3B9	80-pin plastic QFP (14 x 14, 2.7 mm thickness)	Mask ROM
μPD784037GC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Mask ROM
μPD784037GK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Mask ROM
μPD784038GC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Mask ROM
μPD784038GK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Mask ROM
μPD78P4038GC-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	One-time PROM
μPD78P4038GK-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	One-time PROM

Remark ××× indicates ROM code suffix.

(2) μPD784038Y Subseries

Part Number	Package	Internal ROM
μPD784031YGC-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	None
μPD784031YGK-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	None
μPD784035YGC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Mask ROM
μPD784035YGK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Mask ROM
μPD784036YGC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Mask ROM
μPD784036YGK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Mask ROM
μPD784037YGC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Mask ROM
μPD784037YGK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Mask ROM
μPD784038YGC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Mask ROM
μPD784038YGK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Mask ROM
μPD78P4038YGC-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	One-time PROM
μPD78P4038YGK-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	One-time PROM

Remark ××× indicates ROM code suffix.

1.2.2 Quality Grades

(1) μPD784038 Subseries

Part Number	Package	Quality Grades
μPD784031GC-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784031GK-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD784031GC(A)-3B9	80-pin plastic QFP (14 x 14, 2.7 mm thickness)	Special
μPD784035GC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784035GK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD784035GC(A)-×××-3B9	80-pin plastic QFP (14 x 14, 2.7 mm thickness)	Special
μPD784036GC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784036GK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD784036GC(A)-×××-3B9	80-pin plastic QFP (14 x 14, 2.7 mm thickness)	Special
μPD784037GC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784037GK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD784038GC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784038GK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD78P4038GC-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD78P4038GK-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard

Please refer to the document "Quality Grades on NEC Semiconductor Devices" (Document No. C11531E) published by NEC Corporation for the specification of the quality grades of the devices and their recommended applications.

Remark ××× indicates ROM code suffix.

(2) μPD784038Y Subseries

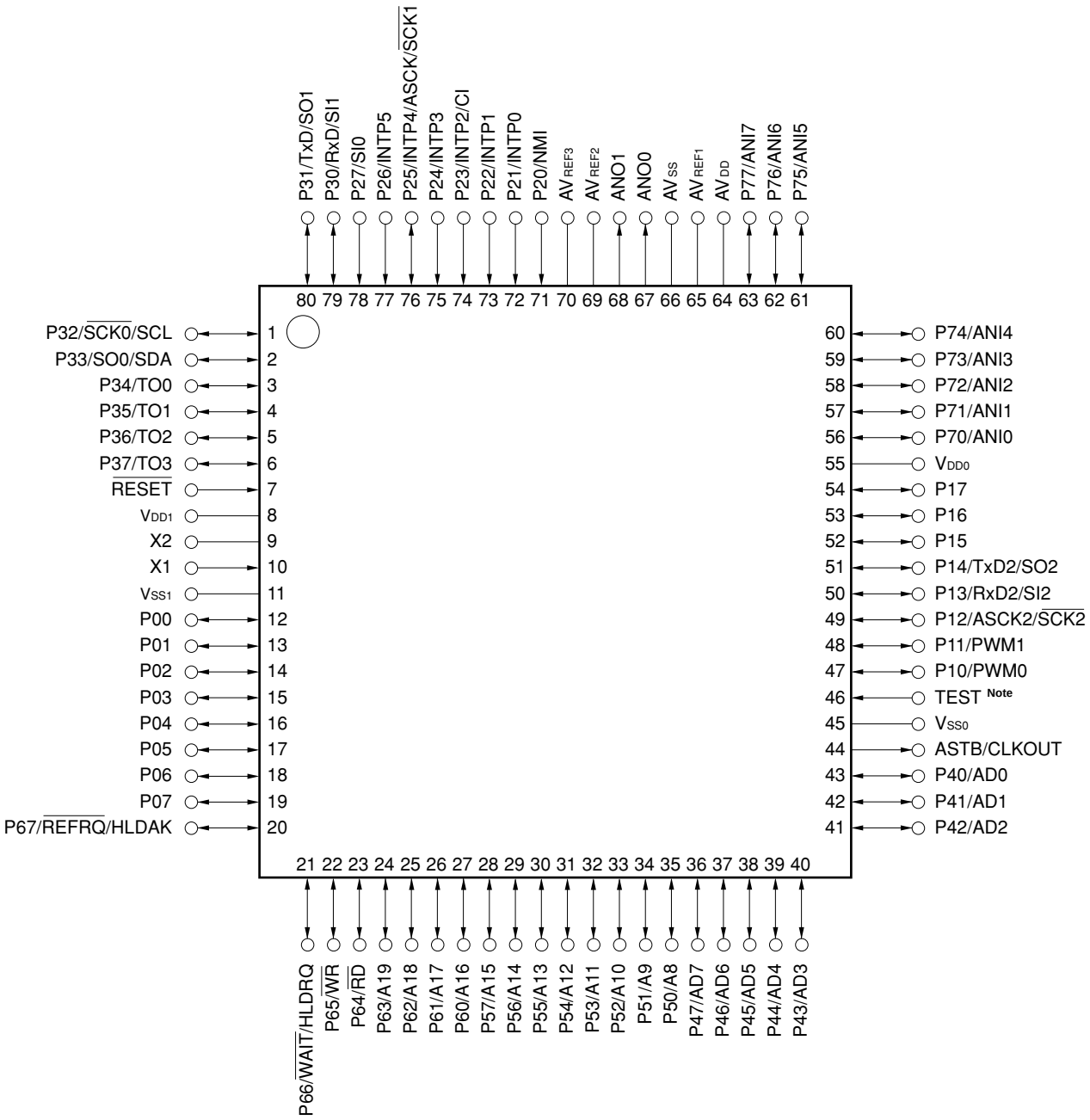
Part Number	Package	Quality Grades
μPD784031YGC-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784031YGK-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD784035YGC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784035YGK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD784036YGC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784036YGK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD784037YGC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784037YGK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD784038YGC-×××-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD784038YGK-×××-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard
μPD78P4038YGC-8BT	80-pin plastic QFP (14 x 14, 1.4 mm thickness)	Standard
μPD78P4038YGK-9EU	80-pin plastic TQFP (fine pitch) (12 x 12)	Standard

Remark ××× indicates ROM code suffix.

1.3 PIN CONFIGURATION (TOP VIEW)

1.3.1 Normal Operating Mode

- **80-pin plastic QFP (14 x 14, 2.7 mm thickness)**
  μPD784031GC(A)-3B9, 784035GC(A)-×××-3B9, 784036GC(A)-×××-3B9
- **80-pin plastic QFP (14 x 14, 1.4 mm thickness)**
  μPD784031GC-8BT, 784035GC-×××-8BT, 784036GC-×××-8BT, 784037GC-×××-8BT, 784038GC-×××-8BT, 78P4038GC-8BT, 784031YGC-8BT, 784035YGC-×××-8BT, 784036YGC-×××-8BT, 784037YGC-×××-8BT, 784038YGC-×××-8BT, 78P4038YGC-8BT
- **80-pin plastic TQFP (fine pitch)(12 x 12)**
  μPD784031GK-9EU, 784035GK-×××-9EU,784036GK-×××-9EU,784037GK-×××-9EU, 784038GK-×××-9EU, 78P4038GK-9EU, 784031YGK-9EU, 784035YGK-×××-9EU, 784036YGK-×××-9EU, 784037YGK-×××-9EU, 784038YGK-×××-9EU, 78P4038YGK-9EU



Note Connect the TEST pin directly to V_{SS0}.

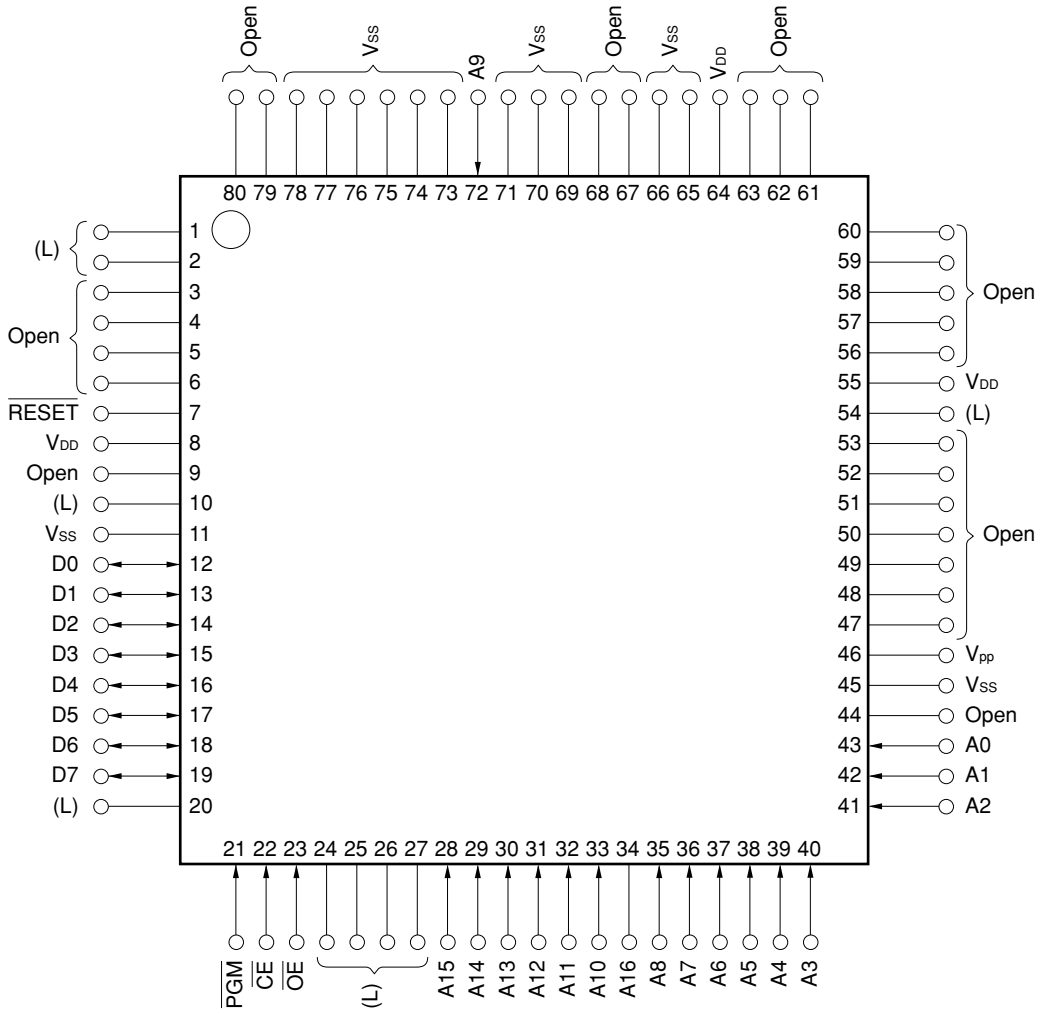
Caution With the μPD784031 and 784031Y, CLKOUT, P40 to P47, P50 to P57, P64, and P65 cannot be used.

P00 to P07	:	Port 0
P10 to P17	:	Port 1
P20 to P27	:	Port 2
P30 to P37	:	Port 3
P40 to P47	:	Port 4
P50 to P57	:	Port 5
P60 to P67	:	Port 6
P70 to P77	:	Port 7
TO0 to TO3	:	Timer Output
CI	:	Clock Input
RxD, RxD2	:	Receive Data
TxD, TxD2	:	Transmit Data
$\overline{\text{SCK0}}$ to $\overline{\text{SCK2}}$	:	Serial Clock
SCL	:	Serial Clock
ASCK, ASCK2	:	Asynchronous Serial Clock
SDA	:	Serial Data
SI0 to SI2	:	Serial Input
SO0 to SO2	:	Serial Output
PWM0, PWM1	:	Pulse Width Modulation Output
NMI	:	Non-maskable Interrupt
INTP0 to INTP5	:	Interrupt from Peripherals

AD0 to AD7	:	Address/Data Bus
A8 to A19	:	Address Bus
$\overline{\text{RD}}$	:	Read Strobe
$\overline{\text{WR}}$	:	Write Strobe
$\overline{\text{WAIT}}$	:	Wait
HLDRQ	:	Hold Request
HLDAK	:	Hold Acknowledge
CLKOUT	:	Clock Out
ASTB	:	Address Strobe
$\overline{\text{REFRQ}}$	:	Refresh Request
$\overline{\text{RESET}}$	:	Reset
X1, X2	:	Crystal
ANI0 to ANI7	:	Analog Input
ANO0, ANO1	:	Analog Output
AV_{REF1} to AV_{REF3}	:	Reference Voltage
AV_{DD}	:	Analog Power Supply
AV_{SS}	:	Analog Ground
V_{DD0}, V_{DD1}	:	Power Supply
V_{SS0}, V_{SS1}	:	Ground
TEST	:	Test

1.3.2 PROM Programming Mode ($V_{PP} \geq$ +5 V/+12.5 V, $\overline{RESET}$ = L)

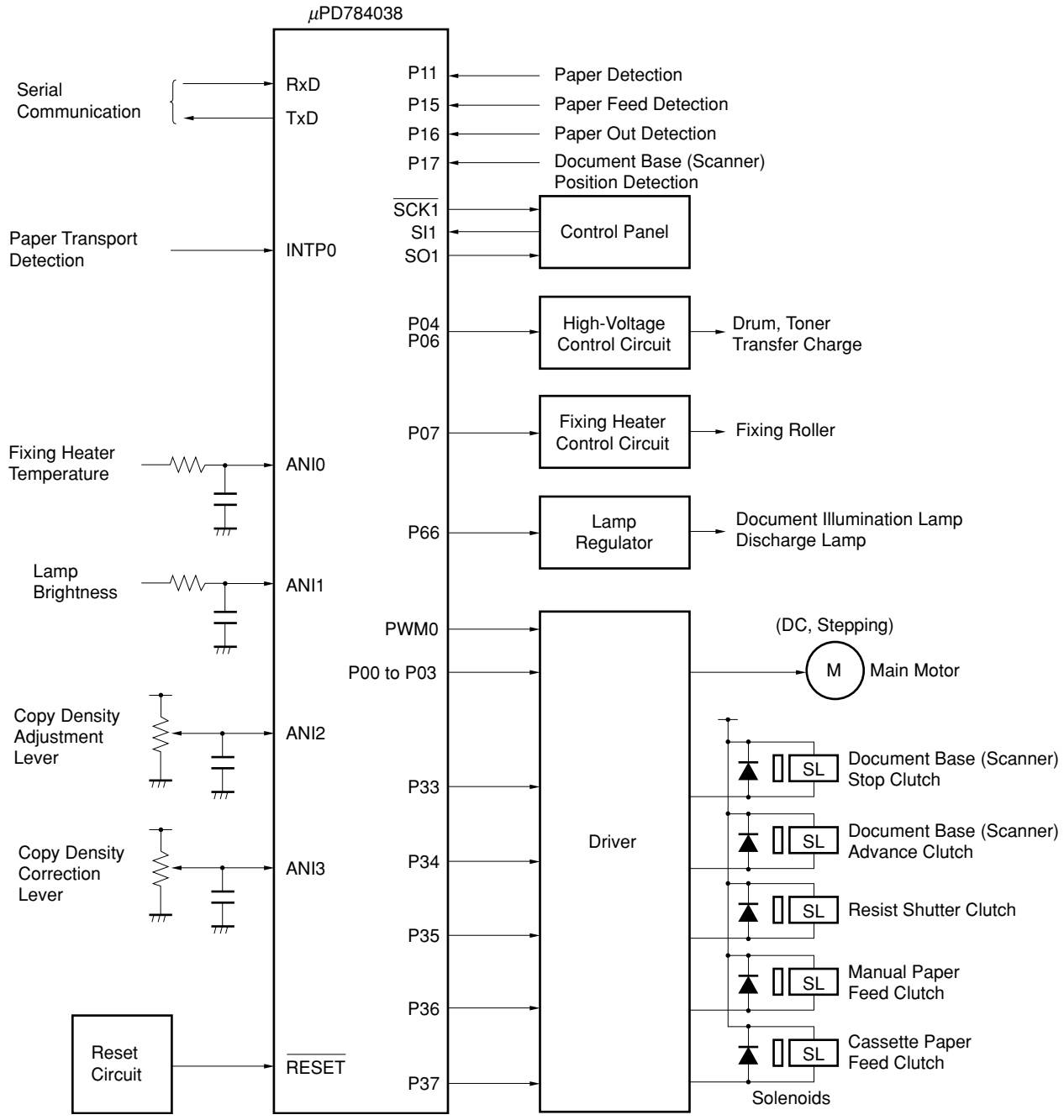
- **80-pin plastic QFP (14 × 14, 1.4 mm thick)**
  μPD78P4038GC-8BT, 78P4038YGC-8BT
- **80-pin plastic TQFP (fine pitch) (12 × 12)**
  μPD78P4038GK-9EU, 78P4038YGK-9EU



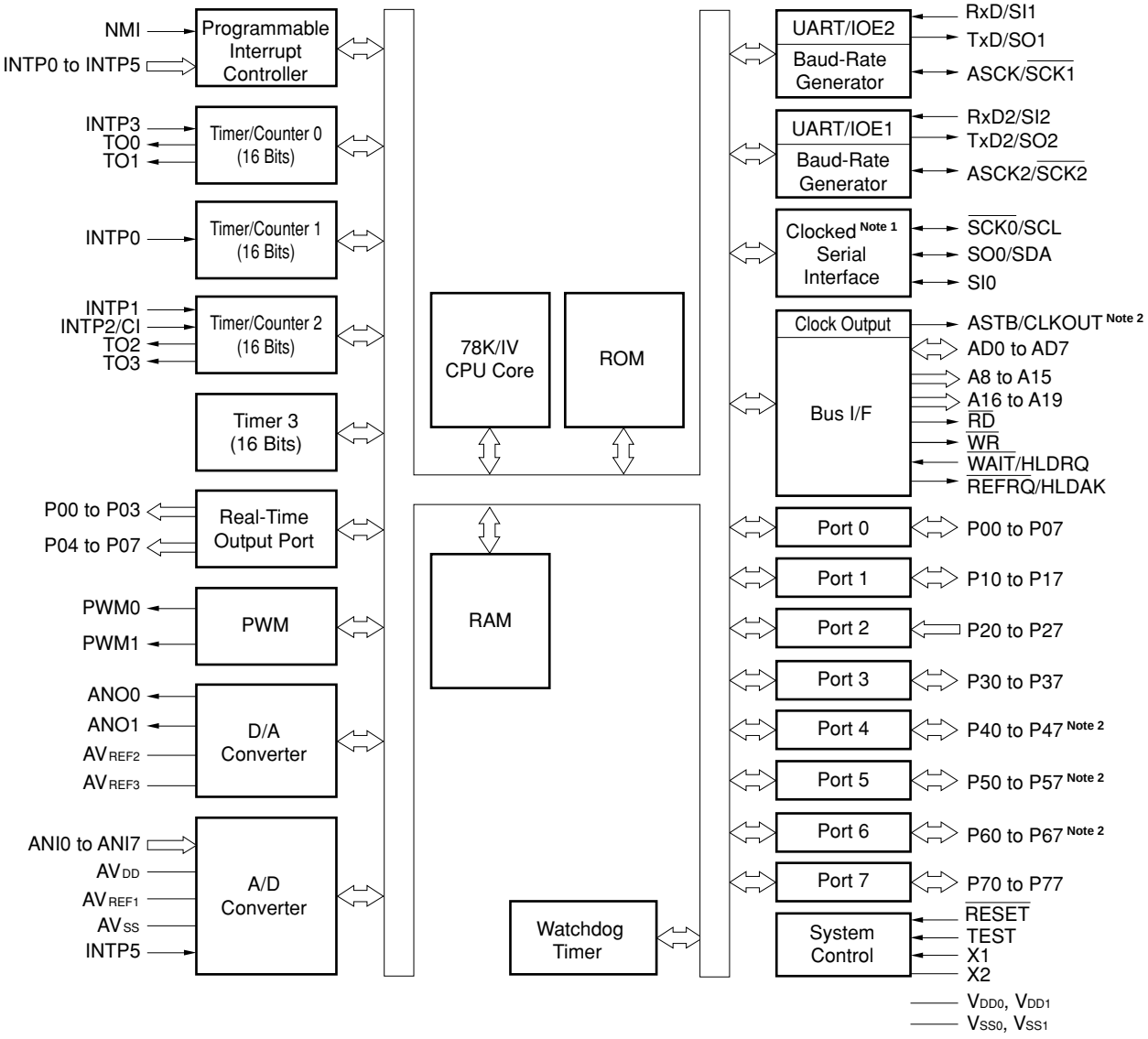
Caution L : Connect to V_{SS} individually via a 10 kΩ pull-down resistor.
V_{SS} : Connect to ground.
Open : Do not make any connection.
$\overline{RESET}$: Drive low.

V_{PP}	: Programming Power Supply	$\overline{PGM}$	: Program
$\overline{RESET}$	: Reset	$\overline{OE}$	: Output Enable
A0 to A16	: Address Bus	V_{DD}	: Power Supply
D0 to D7	: Data Bus	V_{SS}	: Ground
$\overline{CE}$	: Chip Enable		

1.4 APPLICATION SYSTEM CONFIGURATION EXAMPLE (PPC)



1.5 BLOCK DIAGRAM



Notes **1.** The μPD784038Y Subseries supports the I^2C bus mode.

2. When the μPD784031, CLKOUT, P40 to P47, P50 to P57, P64, and P65 cannot be used.

Remark The capacities of the internal ROM and RAM differ depending on the model.

1.6 LIST OF FUNCTIONS

(1/2)

Item \ Part Number		μPD784031 μPD784031Y	μPD784035 μPD784035Y	μPD784036 μPD784036Y	μPD784037 μPD784037Y	μPD784038 μPD784038Y	μPD78P4038 μPD78P4038Y
Number of basic instructions (mnemonics)		113					
General-purpose register		8 bits × 16 registers × 8 banks, or 16 bits × 8 registers × 8 banks (memory mapping)					
Minimum instruction execution time		125 ns (at 32 MHz operation)					
Internal memory	ROM	None	48 Kbytes (mask ROM)	64 Kbytes (mask ROM)	96 Kbytes (mask ROM)	128 Kbytes (mask ROM)	128 Kbytes (one-time PROM)
	RAM	2,048 bytes		3,584 bytes	4,352 bytes		
Memory space		1 Mbyte with program and data memories combined					
I/O port	Total	46 lines	64 lines				
	Input	8 lines					
	I/O	34 lines	56 lines				
	Output	4 lines	0 line				
Pins with ancillary functions Note 1	Pin with pull-up resistor	32 pins	54 pins				
	LED direct drive output	8 pins	24 pins				
	Transistor direct drive	8 pins					
Real-time output port		4 bits × 2, or 8 bits × 1					
Timer/counter		Timer/counter 0 (16 bits): Timer register × 1 Capture register × 1 Compare register × 2 Pulse output • Toggle output • PWM/PPG output • One-shot pulse output					
		Timer/counter 1 (8/16 bits): Timer register × 1 Capture register × 1 Capture/compare register × 1 Compare register × 1 Pulse output • Real-time output (4 bits × 2)					
		Timer/counter 2 (8/16 bits): Timer register × 1 Capture register × 1 Capture/compare register × 1 Compare register × 1 Pulse output • Toggle output • PWM/PPG output					
		Timer 3 (8/16 bits): Timer register × 1 Compare register × 1					
PWM output		12-bit resolution × 2 channels					
Serial interface		UART/IOE (3-wire serial I/O): 2 channels (with baud rate generator) CSI (3-wire serial I/O, 2-wire serial I/O, I²C bus Note 2): 1 channel					
A/D converter		8-bit resolution × 8 channels					
D/A converter		8-bit resolution × 2 channels					

Notes 1. The pins with ancillary functions are included in the I/O pins.

2. μPD784038Y Subseries only.

(2/2)

Item \ Part Number	μPD784031 μPD784031Y	μPD784035 μPD784035Y	μPD784036 μPD784036Y	μPD784037 μPD784037Y	μPD784038 μPD784038Y	μPD78P4038 μPD78P4038Y
Clock output	—	Selectable from f_{CLK}, $f_{CLK}/2$, $f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/16$ (also can be used as 1-bit output port)				
Watchdog timer	1 channel					
Standby	HALT/STOP/IDLE mode					
Interrupt (μPD784038 Subseries)						
Hardware causes	23 (Internal: 16, External: 7 (sampling clock variable input: 1))					
Software	BRK instruction, BRKCS instruction, operand error					
Non-maskable	Internal : 1, External : 1					
Maskable	Internal : 15, External: 6					
	• 4 levels of programmable priority • 3 processing types: vector interrupt/macro service/context switching					
Interrupt (μPD784038Y Subseries)						
Hardware causes	24 (Internal: 17, External: 7 (sampling clock variable input: 1))					
Software	BRK instruction, BRKCS instruction, operand error					
Non-maskable	Internal : 1, External : 1					
Maskable	Internal : 16, External: 6					
	• 4 levels of programmable priority • 3 processing types: vector interrupt/macro service/context switching					
Supply voltage	V_{DD} = 2.7 to 5.5 V					
Package	80-pin plastic QFP (14 × 14, 2.7 mm thick)[Note] 80-pin plastic QFP (14 × 14, 1.4 mm thick) 80-pin plastic TQFP (fine pitch) (12 × 12)					

Note μPD784031(A), 784035(A), and 784036(A) only

★ 1.7 DIFFERENCES BETWEEN STANDARD-GRADE PRODUCTS AND SPECIAL-GRADE PRODUCTS

Item \ Part Number	μPD784031(A), μPD784035(A), μPD784036(A)	μPD784031, μPD784035, μPD784036
Quality grade	Special	Standard
Package	80-pin plastic QFP (14 x 14, 2.7 mm thick)	80-pin plastic QFP (14 x 14, 1.4 mm thick) 80-pin plastic TQFP (fine pitch, 12 x 12)

1.8 MAJOR DIFFERENCES WITH μPD784026 SUBSERIES

Item \ Series Name	μPD784038 Subseries	μPD784038Y Subseries	μPD784026 Subseries
Minimum instruction execution time	125 ns (32-MHz operation)		160 ns (25-MHz operation)
Serial interface	UART/IOE (3-wire serial I/O) × 2 channels		
	CSI × 1 channel • 3-wire serial I/O • 2-wire serial I/O	CSI × 1 channel • 3-wire serial I/O • 2-wire serial I/O • I^2C bus	CSI × 1 channel • 3-wire serial I/O • SBI
Interrupts	23 + BRK instruction (Internal: 16, External: 7)	24 + BRK instruction (Internal: 17, External: 7)	23 + BRK instruction (Internal: 16, External: 7)
Packages	• 80-pin plastic QFP (14 × 14, 2.7 mm thick)[Note] • 80-pin plastic QFP (14 × 14, 1.4 mm thick) • 80-pin plastic TQFP (fine-pitch) (12 × 12)		• 80-pin plastic QFP (14 × 14, 2.7 mm thick) • 80-pin plastic TQFP (fine-pitch) (12 × 12): μPD784021 only

Note μPD784031(A), 784035(A), and 784036(A) only

CHAPTER 2 PIN FUNCTIONS

2.1 PIN FUNCTION TABLES

2.1.1 Normal Operating Mode

(1) Port pins (1/2)

Pin Name	Input/Output	Alternate Function	Functions
P00 to P07	Input/output	—	Port 0 (P0): • 8-bit input/output port • Can be used as real-time output ports (4 bits × 2) • Input/output specifiable bit-wise • For input mode pins, on-chip pull-up resistor connection can be specified at once by a software setting • Transistor drive capability
P10	Input/output	PWM0	Port 1 (P1): • 8-bit input/output port • Input/output specifiable bit-wise. • For input mode pins, on-chip pull-up resistor connection can be specified at once by a software setting • LED drive capability
P11		PWM1	
P12		ASCK2/$\overline{\text{SCK2}}$	
P13		RxD2/SI2	
P14		TxD2/SO2	
P15 to P17		—	
P20	Input	NMI	Port 2 (P2): • 8-bit input/output port • P20 cannot be used as a general-purpose port (non-maskable interrupt). Input level can be confirmed in the interrupt routine. • For P22 to P27, on-chip pull-up resistor connection can be specified by a software setting in 6-bit units • The P25/INTP4/ASCK/$\overline{\text{SCK1}}$ pin operates as the $\overline{\text{SCK1}}$ I/O pin in accordance with the CSIM1 register specification
P21		INTP0	
P22		INTP1	
P23		INTP2/CI	
P24		INTP3	
P25		INTP4/ASCK/$\overline{\text{SCK1}}$	
P26		INTP5	
P27		SI0	
P30	Input/output	RxD/SI1	Port 3 (P3): • 8-bit input/output port • Input/output specifiable bit-wise • For input mode pins, on-chip pull-up resistor connection can be specified at once by a software setting
P31		TxD/SO1	
P32		$\overline{\text{SCK0}}$/SCL	
P33		SO0/SDA	
P34 to P37		TO0 to TO3	

(1) Port pins (2/2)

Pin Name	Input/Output	Alternate Function	Functions
P40 to P47 Note 1	Input/output	AD0 to AD7	Port 4 (P4): • 8-bit input/output port • Input/output specifiable bit-wise • For input mode pins, on-chip pull-up resistor connection can be specified at once by a software setting • LEDs drive capability
P50 to P57 Note 1	Input/output	A8 to A15	Port 5 (P5): • 8-bit input/output port • Input/output specifiable bit-wise • For input mode pins, on-chip pull-up resistor connection can be specified at once by a software setting • LEDs drive capability
P60 to P63 Note 2	Input/output	A16 to A19	Port 6 (P6): • 8-bit input/output port • Input/output specifiable bit-wise • For input mode pins, on-chip pull-up resistor connection can be specified at once by a software setting
P64 Note 1		$\overline{RD}$	
P65 Note 1		$\overline{WR}$	
P66		$\overline{WAIT}$/HLDRQ	
P67		$\overline{REFRQ}$/HLDAK	
P70 to P77	Input/output	ANI0 to ANI7	Port 7 (P7): • 8-bit input/output port • Input/output specifiable bit-wise

Notes 1. With the μPD784031, P40 to P47, P50 to P57, P64, and P65 cannot be used as port pins.

2. These pins of the μPD784031 are output port pins.

(2) Non-port pins (1/2)

Pin Name	Input/Output	Alternate Function	Functions	
TO0/TO3	Output	P34 to P37	Timer output	
CI	Input	P23/INTP2	Count clock input to timer/counter 2	
RxD	Input	P30/SI1	Serial data input (UART0)	
RxD2		P13/SI2	Serial data input (UART2)	
TxD	Output	P31/SO1	Serial data output (UART0)	
TxD2		P14/SO2	Serial data output (UART2)	
ASCK	Input	P25/INTP4/$\overline{\text{SCK1}}$	Baud rate clock input (UART0)	
ASCK2		P12/$\overline{\text{SCK2}}$	Baud rate clock input (UART2)	
SDA	Input/output	P33/SO0	Serial data input/output (2-wire serial I/O, I²C bus Note)	
SI0	Input	P27	Serial data input (3-wire serial I/O0)	
SI1		P30/RxD	Serial data input (3-wire serial I/O1)	
SI2		P13/RxD2	Serial data input (3-wire serial I/O2)	
SO0	Output	P33/SDA	Serial data output (3-wire serial I/O0)	
SO1		P31/TxD	Serial data output (3-wire serial I/O1)	
SO2		P14/TxD2	Serial data output (3-wire serial I/O2)	
$\overline{\text{SCK0}}$	Input/output	P32/SCL	Serial clock input/output (3-wire serial I/O0)	
$\overline{\text{SCK1}}$		P25/INTP4/ASCK	Serial clock input/output (3-wire serial I/O1)	
$\overline{\text{SCK2}}$		P12/ASCK2	Serial clock input/output (3-wire serial I/O2)	
SCL		P32/$\overline{\text{SCK0}}$	Serial clock input/output (2-wire serial I/O, I²C bus Note)	
NMI	Input	P20	External interrupt requests	—
INTP0		P21		• Count clock input to timer/counter 1 • CR11 or CR12 capture trigger signal
INTP1		P22		• Count clock input to timer/counter 2 • CR22 capture trigger signal
INTP2		P23/CI		• Count clock input to timer/counter 2 • CR21 capture trigger signal
INTP3		P24		• Count clock input to timer/counter 0 • CR02 capture trigger signal
INTP4		P25/ASCK/$\overline{\text{SCK1}}$		—
INTP5		P26		A/D converter conversion start trigger input
AD0 to AD7	Input/output	P40 to P47	Time division address/data bus (external memory connection)	
A8 to A15	Output	P50 to P57	Upper address bus (external memory connection)	
A16 to A19	Output	P60 to P63	Upper address with address extension (external memory connection)	
$\overline{\text{RD}}$	Output	P64	External memory read strobe	
$\overline{\text{WR}}$	Output	P65	External memory write strobe	
$\overline{\text{WAIT}}$	Input	P66/HLDRQ	Wait insertion	
$\overline{\text{REFRQ}}$	Output	P67/HLDAK	External pseudo-static memory refresh pulse output	
HLDRQ	Input	P66/$\overline{\text{WAIT}}$	Bus hold request input	
HLDAK	Output	P67/$\overline{\text{REFRQ}}$	Bus hold response output	

Note μPD784038Y Subseries only

(2) Non-port pins (2/2)

Pin Name	Input/Output	Alternate Function	Functions
ASTB	Output	CLKOUT	Time division address (A0 to A7) latch timing output (during external memory access)
CLKOUT Note 1	Output	ASTB	Clock output
$\overline{\text{RESET}}$	Input	—	Chip reset
X1	Input	—	System clock oscillation crystal connections (clock can also be input to X1)
X2	—		
ANI0 to ANI7	Input	P70 to P77	A/D conversion analog voltage inputs
ANO0, ANO1	Output	—	D/A conversion analog voltage outputs
AV_{REF1}	—	—	A/D converter reference voltage application
AV_{REF2}, AV_{REF3}			D/A converter reference voltage application
AV_{DD}			A/D converter positive power supply
AV_{SS}			A/D converter GND
V_{DD0} Note 2			Positive power supply pin of ports
V_{DD1} Note 2			Positive power supply pin of function blocks other than ports
V_{SS0} Note 3			GND pin of ports
V_{SS1} Note 3			GND pin of function blocks other than ports
TEST			Connect directly to V_{SS0} (IC test pin).

Notes 1. With the μPD784031, CLKOUT cannot be used.

2. Keep V_{DD0} and V_{DD1} at the same potential.

3. Keep V_{SS0} and V_{SS1} at the same potential.

2.1.2 PROM Programming Mode (μPD78P4038 Only: $V_{PP} \geq$ +5 V/+12.5 V, $\overline{\text{RESET}}$ = L)

Pin Name	Input/Output	Functions
V_{PP}	Input	PROM programming mode setting High-voltage application pin in program write/verify
$\overline{\text{RESET}}$		PROM programming mode setting
A0 to A16		Address bus
D0 to D7	Input/output	Data bus
$\overline{\text{CE}}$	Input	PROM enable input/program pulse input
$\overline{\text{OE}}$		PROM read strobe input
$\overline{\text{PGM}}$		PROM program/program inhibit input
V_{DD}	—	Positive power supply
V_{SS}		GND

2.2 PIN FUNCTIONS

2.2.1 Normal Operating Mode

(1) P00 to P07 (Port 0) ... 3-state input/output

Port 0 is an 8-bit input/output port with an output latch, and has direct transistor drive capability. Input/output can be specified bit-wise by means of the port 0 mode register (PM0). Each pin incorporates a software programmable pull-up resistor. P00 to P03 and P04 to P07 can output the port 0 buffer register (P0L, P0H) contents at any time interval as 4-bit or 8-bit real-time output port. The real-time output port control register (RTPC) is used to select whether this port is used as a normal output port or a real-time output port.

When $\overline{\text{RESET}}$ is input, port 0 is set as an input port (output high-impedance state), and the output latch contents are undefined.

(2) P10 to P17 (Port 1) ... 3-state input/output

Port 1 is an 8-bit input/output port with an output latch. Input/output can be specified bit-wise by means of the port 1 mode register (PM1). Each pin incorporates a software programmable pull-up resistor. This port has direct LED drive capability. Pins P10 and P11 are also made to function as PWM output pins by means of the PWM control register (PWMC), and pins P12 to P14 can also be made to function as serial input/output pins by means of the port 1 mode control register (PMC1).

When $\overline{\text{RESET}}$ is input, port 1 is set as an input port (output high-impedance state), and the output latch contents are undefined.

Table 2-1 Port 1 Operating Modes

Pin Name	Port Mode	Control Signal Input/Output Mode	Operation to Operate as Control Pin
P10	Input/output port	PWM0 output	Set (to 1) EN0 bit of PWMC
P11		PWM1 output	Set (to 1) EN1 bit of PWMC
P12		ASCK2/$\overline{\text{SCK2}}$ input/output	Set (to 1) PMC12 bit of PMC1
P13		RxD2/SI2 input	Set (to 1) PMC13 bit of PMC1
P14		TxD2/SO2 output	Set (to 1) PMC14 bit of PMC1
P15 to P17		—	—

(a) Port mode

P10 and P11 operate as port mode pins when the EN0 and EN1 bits of the PWM control (PWMC) register are cleared (to 0), and P12 to P14 do the same when the relevant bits of the port 1 mode control (PMC1) register are cleared (to 0), and P15 to P17 always operate as port mode pins. Input/output can be specified bit-wise by means of the port 1 mode register (PM1).

(b) Control signal input/output mode

P10 and P11 operate as PWM signal output pins when the EN0 and EN1 bits, respectively, of the PWM control (PWMC) register are set (to 1).

P12 to P14 can be set as control pins bit-wise by setting the port 1 mode control (PMC1) register.

(i) PWM0, PWM1

PWM0 and PWM1 are PWM output pins.

(ii) ASCK2/$\overline{\text{SCK2}}$

ASCK2 is the asynchronous serial interface baud rate clock input pin.

$\overline{\text{SCK2}}$ is the serial clock input/output pin (in 3-wire serial I/O2 mode).

(iii) RxD2/SI2

RxD2 is the asynchronous serial interface serial data input pin.
SI2 is the serial data input pin (in 3-wire serial I/O2 mode).

(iv) TxD2/SO2

TxD2 is the asynchronous serial interface serial data output pin.
SO2 is the serial data output pin (in 3-wire serial I/O2 mode).

(3) P20 to P27 (Port 2) ... Input

Port 2 is an 8-bit input-only port. P22 to P27 incorporate a software programmable pull-up resistor. As well as operating as an input port, port 2 pins also operate as control signal input pins, such as external interrupt signal pins (see **Table 2-2**). All 8 pins are Schmitt-triggered inputs to prevent malfunction due to noise.
Also, pin P25 can also be made to function as a serial clock output pin by selecting the external clock as "serial operation enabled" with the clocked serial interface mode register 1 (CSIM1).

Table 2-2 Port 2 Operating Modes

Port	Functions
P20	Input port / NMI input Note
P21	Input port / INTP0 input / CR11 capture trigger input / timer/counter 1 count clock / real-time output port trigger signal
P22	Input port / INTP1 input / CR22 capture trigger input
P23	Input port / INTP2 input / CI input
P24	Input port / INTP3 input / CR02 capture trigger timer/input/counter 0 count clock
P25	Input port / INTP4 input / ASCK input / $\overline{SCK1}$ input/output
P26	Input port / INTP5 input / A/D converter external trigger input
P27	Input port / SI0 input

Note NMI input is acknowledged regardless of whether interrupts are enabled or disabled.

(a) Function as port pins

The pin level can always be read or tested regardless of the dual-function pin operation.

(b) Functions as control signal input pins

(i) NMI (Non-maskable Interrupt)

The external non-maskable interrupt request input pin. Rising edge detection or falling edge detection can be specified by means of the external interrupt mode register 0 (INTM0).

(ii) INTP0 to INTP5 (Interrupt from Peripherals)

External interrupt request input pins. When the valid edge specified by the external interrupt mode register 0, (INTM0/INTM1) is detected by pins INTP0 to INTP5, an interrupt is generated (see **CHAPTER 21 EDGE DETECTION FUNCTION**).

In addition, pins INTP0 to INTP3 and INTP5 are also used as external trigger input pins with the various functions shown below.

- INTP0 ..... Timer/counter 1 capture trigger input pin
  Timer/counter 1 external count clock input pin
  Real-time output port trigger input pin
- INTP1 ..... Timer/counter 2 capture trigger input pin to capture register (CR22)
- INTP2 ..... Timer/counter 2 external count clock input pin
  Capture trigger input pin to capture/compare register (CR21)
- INTP3 ..... Timer/counter 0 capture trigger input pin
  Timer/counter 0 external count clock input pin
- INTP5 ..... A/D converter external trigger input pin

(iii) CI (Clock Input)

The timer/counter 2 external clock input pin.

(iv) ASCK (Asynchronous Serial Clock)

The external baud rate clock input pin.

(v) $\overline{\text{SCK1}}$ (Serial Clock)

The serial clock input/output pin (in 3-wire serial I/O1 mode).

(vi) SIO (Serial Input 0)

The serial data input pin (in 3-wire serial I/O0 mode).

(4) P30 to P37 (Port 3) ... 3-state input/output

Port 3 is an 8-bit input/output port with an output latch. Input/output can be specified bit-wise by means of the port 3 mode register (PM3). Each pin incorporates a software programmable pull-up resistor.

In addition to its function as an input/output port, port 3 also has various alternate-function control signal pin functions.

The operating mode can be specified bit-wise by means of the port 3 mode control register (PMC3), as shown in Table 2-3. The pin level of any pin can always be read or tested regardless of the alternate-function pin operation.

When $\overline{\text{RESET}}$ is input, port 3 is set as an input port (output high-impedance state), and the output latch contents are undefined.

Table 2-3 Port 3 Operating Modes (n = 0 to 7)

Mode	Port Mode	Control Signal Input/Output Mode
Setting Condition	PMC3n = 0	PMC3n = 1
P30	Input/output port	RxD input / SI1 input
P31		TxD output / SO1 output
P32		$\overline{\text{SCK0}}$ input/output / SCL input/output
P33		SO0 output / SDA input/output
P34		TO0 output
P35		TO1 output
P36		TO2 output
P37		TO3 output

(a) Port mode

Each port specified as port mode by the port 3 mode control (PMC3) register can be specified as input/output bit-wise by means of the port 3 mode register (PM3).

(b) Control signal input/output mode

Pins can be set as control pins bit-wise by setting the port 3 mode control (PMC3) register.

(i) RxD (Receive Data)/SI1 (Serial Input 1)

RxD is the asynchronous serial interface serial data input pin.
SI1 is the serial data input pin (in 3-wire serial I/O1 mode).

(ii) TxD (Transmit Data)/SO1 (Serial Output 1)

TxD is the asynchronous serial interface serial data output pin.
SO1 is the serial data output pin (in 3-wire serial I/O1 mode).

(iii) $\overline{\text{SCK0}}$ (Serial Clock 0)/SCL (Serial Clock)

$\overline{\text{SCK0}}$ is the clocked serial interface serial clock input/output pin (in 3-wire serial I/O 0 mode).
SCL is the synchronous serial interface serial clock input/output pin (in 2-wire serial I/O mode/I^2C bus mode **Note**).

Note μPD784038Y Subseries only

(iv) SO0 (Serial Output 0)/SDA (Serial Data)

SO0 is the serial data output pin (in 3-wire serial I/O 0 mode), and SDA is the serial data input/output pin (in 2-wire serial I/O mode/I^2C bus mode **Note**).

Note μPD784038Y Subseries only

(v) TO0 to TO3 (Timer Output)

The timer output pins.

(5) P40 to P47 (Port 4) ... 3-state input/output

Port 4 is an 8-bit input/output port with an output latch. Input/output can be specified bit-wise by means of the port 4 mode register (PM4). Each pin incorporates a software programmable pull-up resistor. This port has direct LED drive capability.
Port 4 also functions as the time division address/data bus (AD0 to AD7) by the memory extension mode register (MM) when external memory or I/Os are extended.
With the μPD784031, P40 to P47 cannot be used as port pins. These pins function only as the time division address/data bus pins (AD0 to AD7).
When $\overline{\text{RESET}}$ is input, port 4 is set as an input port (output high-impedance state), and the output latch contents are undefined.

(6) P50 to P57 (Port 5) ... 3-state input/output

Port 5 is an 8-bit input/output port with an output latch. Input/output can be specified bit-wise by means of the port 5 mode register (PM5). Each pin incorporates a software programmable pull-up resistor. This port has direct LED drive capability.
In addition, P50 to P57 can be selected by means of the memory extension mode register (MM) in 2-bit units as pins that function as the address bus (A8 to A15) when external memory or I/Os are extended.
With the μPD784031, P50 to P57 cannot be used as port pins. These pins function only as the address bus pins (A8 to A15).
When $\overline{\text{RESET}}$ is input, port 5 is set as an input port (output high-impedance state), and the output latch contents are undefined.

(7) P60 to P67 (Port 6) ... 3-state input/output

• With μPD784031

P60 to P63 are output port pins and P66 and P67 are input/output port pins with output latch.

P64 to P67 incorporate a software programmable pull-up resistor.

In addition to the functions as port pins, these pins also have various alternate-function control signal pin functions, as shown in Table 2-4. Operations as control pins are performed by the respective function operations.

P64 and P65 cannot be used as port pins and function only as $\overline{RD}$ and $\overline{WR}$ output pins.

When $\overline{RESET}$ is input, the level of the above pins are set as follows:

- P60 to P63: Low
- P64, P65: High
- P66, P67: Input port (output high impedance)

The higher 4 bits of the contents are undefined, and the lower 4 bits are reset to 0H.

• With other than μPD784031

Port 6 is an 8-bit input/output port with an output latch. P60 to P67 incorporate a software programmable pull-up resistor.

In addition to its function as a port, port 6 also has various alternate-function control signal pin functions, as shown in Table 2-4. Operations as control pins are performed by the respective function operations.

When $\overline{RESET}$ is input, P60 to P67 are set as input port pins (output high-impedance state), and the output latch contents are undefined.

Table 2-4 Port 6 Operating Modes

Pin Name	Port Mode	Control Signal Input/Output Mode	Operation to Operate as Control Pin
P60 to P63 Note 1	Input/output ports	A16 to A19 output	Specified in 2-bit units by bits MM3 to MM0 of the MM
P64 Note 2		$\overline{RD}$ output	With the μPD784031, or when external memory extension mode is specified by bits MM3 to MM0 of the MM
P65 Note 2		$\overline{WR}$ output	
P66		$\overline{WAIT}$ input	Specified by setting bits PWn1 & PWn0 (n = 0 to 7) of the PWC1 & PWC2 and P66 to input mode
		HLDRQ input	Bus hold enabled by the HLDE bit of the HLDM
P67		HLDAK output	
		$\overline{REFRQ}$ output	Set (to 1) the RFEN bit of the RFM

Notes 1. These pins of the μPD784031 are output port pins only.

2. With the μPD784031, this pin cannot be used as a port pin.

Caution P60 to P63 of the μPD784031 are in the output high-impedance state while the $\overline{RESET}$ signal is input, but output a low level after the $\overline{RESET}$ signal has been cleared. Therefore, design the external circuit so that the low level may be output as the initial status.

Remark For details, refer to **CHAPTER 23 LOCAL BUS INTERFACE FUNCTION**.

(a) Port mode

• With μPD784031

Each port not specified as control mode, P66 and P67 serve as output port pins, and P66 and P67 can be specified as input/output bit-wise by means of the port 6 mode register (PM6).

• With other than μPD784031

Each port not set in the control mode can be set in the input or output mode in 1-bit units by using the port 6 mode register (PM6).

(b) Control signal input/output mode

(i) A16 to A19 (Address Bus)

Upper address bus output pins in case of external memory space extension (10000H to FFFFFH).
These pins operate in accordance with the memory extension mode register (MM).

(ii) RD (Read Strobe)

Pin that outputs the strobe signal for an external memory read operation.
Operates in accordance with the memory extension mode register (MM).
With the μPD784031, this pin always serves as an $\overline{RD}$ pin.

(iii) $\overline{WR}$ (Write Strobe)

Pin that outputs the strobe signal for an external memory write operation.
Operates in accordance with the memory extension mode register (MM).
With the μPD784031, this pin always serves as a $\overline{WR}$ pin.

(iv) $\overline{WAIT}$ (Wait)

Wait signal input pin. Operates in accordance with the programmable wait control registers (PWC1, PWC2).

(v) HLDRQ (Hold Request)

External bus hold request signal input pin. Operates in accordance with the hold mode register (HLDM).

(vi) HLDAK (Hold Acknowledge)

Bus hold acknowledge signal output pin. Operates in accordance with the hold mode register (HLDM).

(vii) $\overline{REFRQ}$ (Refresh Request)

This pin outputs refresh pulses to pseudo-static memory when this memory is connected externally. Operates in accordance with the refresh mode register (RFM).

(8) P70 to P77 (Port 7) ... 3-state input/output

Port 7 is an 8-bit input/output port. In addition to operating as an input/output port, it also operates as the A/D converter analog input pins (ANI0 to ANI7).
Input/output can be specified bit-wise by means of the port 7 mode register (PM7).
The levels of these pins can always be read or tested, regardless of the operation of the multiplexed pins.
When $\overline{RESET}$ is input, port 7 is set as an input port (output high-impedance state), and the output latch contents are undefined.

(9) ASTB (Address Strobe)/CLKOUT (Clock Output) ... Output

This pin outputs the timing signal that latches address information externally in order to access an external address. It also operates as the pin that supplies the clock to an external device.
With the μPD784031, CLKOUT cannot be used.

(10) X1, X2 (Crystal)

The internal clock oscillation crystal connection pins. When the clock is supplied externally, it is input to the X1 pin. Usually signal with the inverse phase of the X1 pin signal phase is input to the X2 pin (Refer to **4.3.1 Clock oscillation circuit**).

(11) $\overline{\text{RESET}}$ (Reset) ... Input

The active-low reset input.

(12) ANO0, ANO1 ... Output

The D/A converter analog voltage output pins.

(13) AV_{REF1}

The A/D converter reference voltage input pin.

(14) AV_{REF2}

The D/A converter reference voltage input (+ side) pin.

(15) AV_{REF3}

The D/A converter reference voltage input (– side) pin.

(16) AV_{DD}

The A/D converter power supply pin. This should be made at the same potential as the V_{DD} pin.

(17) AV_{SS}

The A/D converter GND pin. This should be made at the same potential as the V_{SS} pin.

(18) V_{DD0}

Positive power supply pins of the ports. These pins should be made at the same potential as the V_{DD1}.

(19) V_{DD1}

Positive power supply pins of the function blocks other than ports. These pins should be made at the same potential as the V_{DD0}.

(20) V_{SS0}

GND potential pins of the ports. These pins should be made at the same potential as the V_{SS1}.

(21) V_{SS1}

GND potential pins of the function blocks other than ports. These pins should be made at the same potential as the V_{SS0}.

(22) TEST

Pin used by NEC Electronics for IC testing. Must be directly connected to V_{SS0}.

Caution In the μPD78233 and 78237, the TEST pin is the MODE pin and is fixed high. When changing over from the μPD78233, 78237, the circuitry can be modified so that this pin is directly connected to V_{SS0}. Modification as shown below is needed if the μPD78234, 78238 was used with switching between the on-chip ROM mode and ROM-less mode performed by MODE pin switching (the TEST pin must be directly connected to V_{SS0}).

Modification examples:

- Incorporate all programs in ROM.
- Store all programs in external ROM.
- Change the location address of a program previously held in external ROM, shift the address to avoid overlapping internal ROM, and execute this program from the program internal ROM.

2.2.2 PROM Programming Mode (μPD78P4038)

(1) V_{PP} (Programming Power Supply) ... Input

Input pin that sets the μPD78P4038 to the PROM programming mode. When the input voltage of this pin is +5 V or more and the $\overline{RESET}$ input is driven low, the μPD78P4038 switches to the PROM programming mode.

If $\overline{CE}$ = L is set when V_{PP} = +12.5 V and $\overline{OE}$ = H, the program data on D0 to D7 can be written in the internal PROM cell selected by A0 to A16.

(2) $\overline{RESET}$ (Reset) ... Input

Input pin that sets the μPD78P4038 to the PROM programming mode. When the input voltage of the V_{PP} pin reaches +5 V or more and the input of this pin is low, the μPD78P4038 switches to the PROM programming mode.

(3) A0 to A16 (Address Bus) ... Input

The address bus. Selects an internal PROM address (00000H to 1FFFFH).

(4) D0 to D7 (Data Bus) ... Input/Output

The data bus. Internal PROM program reads and writes are performed via this bus.

(5) $\overline{CE}$ (Chip Enable) ... Input

Inputs the internal PROM enable signal. When this signal is active, program writing/reading is enabled.

(6) $\overline{OE}$ (Output Enable) ... Input

Inputs the internal PROM read strobe signal. When this signal is activated while $\overline{CE}$ = L, the program data (1 byte) in the internal PROM cell selected by A0 to A16 can be read onto D0 to D7.

(7) $\overline{PGM}$ (Program) ... Input

The internal PROM operating mode control signal input pin.

When this signal is active, it is possible to write to internal PROM.

When this signal is inactive, it is possible to read from internal PROM.

(8) V_{DD}

Positive power supply pins.

(9) V_{SS}

GND potential pins.

2.3 INPUT/OUTPUT CIRCUITS AND CONNECTION OF UNUSED PINS

Table 2-5 shows the input/output circuit types of the pins that have functions, and the connection method when that function is not used.

Each input/output circuit type is shown in Figure 2-1.

Table 2-5 Pin Input/Output Circuit Types and Recommended Connection When Not Used (1/2)

Pin Name		Input/Output Circuit Type	Input/Output	Recommended Connection When Not Used
P00 to P07		5-H	Input/output	Input : Connect to V_{DD0} Output : Leave open
P10/PWM0 P11/PWM1				
P12/ASCK2/$\overline{SCK2}$		8-C		
P13/RxD2/SI2		5-H		
P14/TxD2/SO2				
P15 to P17				
P20/NMI		2	Input	Connect to V_{DD0} or V_{SS0}
P21/INTP0				
P22/INTP1		2-C		Connect to V_{DD0}
P23/INTP2/CI				
P24/INTP3				
P25/INTP4/ASCK/$\overline{SCK1}$		8-C	Input/output	Input : Connect to V_{DD0} Output : Leave open
P26/INTP5		2-C	Input	Connect to V_{DD0}
P27/SI0				
P30/RxD/SI1		5-H	Input/output	Input : Connect to V_{DD0} Output : Leave open
P31/TxD/SO1				
P32/$\overline{SCK0}$/SCL		10-B		
P33/SO0/SDA				
P34/TO0 to P37/TO3		5-H		
P40/AD0 to P47/AD7				
P50/A8 to P57/A15				
P60/A16 to P63/A19	ROM-less version	4-B	Output	Leave open
	Mask ROM version	5-H	Input/output	Input : Connect to V_{DD0} Output : Leave open
P64/$\overline{RD}$				
P65/$\overline{WR}$				
P66/$\overline{WAIT}$/HLDRQ				
P67/$\overline{REFRQ}$/HLDAK				

Table 2-5 Pin Input/Output Circuit Types and Recommended Connection When Not Used (2/2)

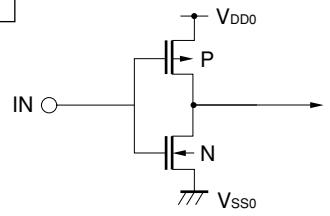
Pin Name	Input/Output Circuit Type	Input/Output	Recommended Connection When Not Used
P70/ANI0 to P77/ANI7	20-A	Input/output	Input : Connect to V_{DD0} or V_{SS0} Output : Leave open
ANO0, ANO1	12	Output	Leave open
ASTB/CLKOUT	4-B		
$\overline{\text{RESET}}$	2	Input	—
TEST	1-A		Directly connect to V_{SS0}
AV_{REF1} to AV_{REF3}	—		Connect to V_{SS0}
AV_{SS}			
AV_{DD}			Connect to V_{DD0}

Caution If the input/output mode is undefined for an input/output alternate-function pin, it should be connected to V_{DD0} via a resistor of several tens of kΩ (especially when the reset input pin goes to the low-level input voltage or over upon powering on, and when input/output is switched by software.)

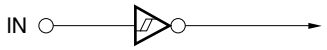
Remark The type numbers are standard for the 78K Series, and therefore are not necessarily serial numbers within each product (there are non-incorporated circuits).

Figure 2-1 Pin Input/Output Circuits

Type 1-A

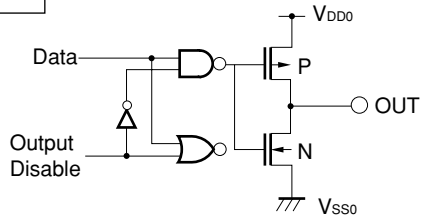


Type 2



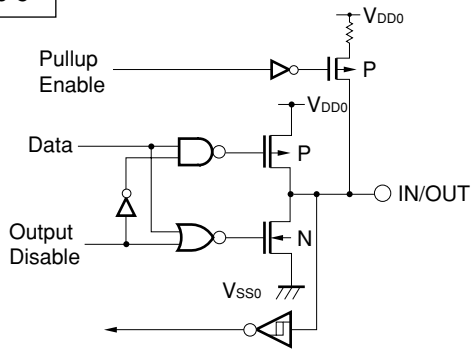
Schmitt-triggered input with hysteresis characteristics.

Type 4-B

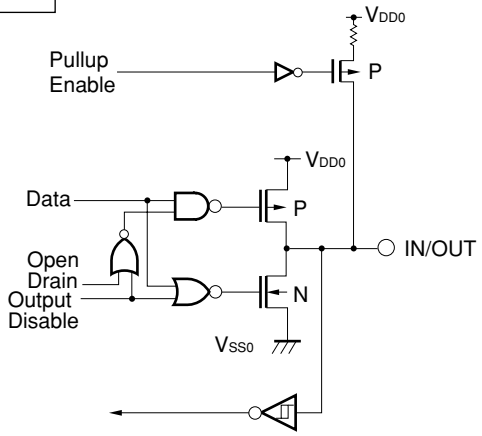


Push-pull output allowing output to be set to high impedance (P-ch & N-ch both off).

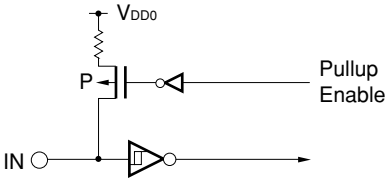
Type 8-C



Type 10-B

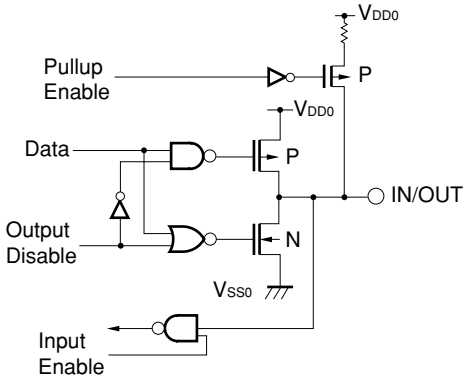


Type 2-C

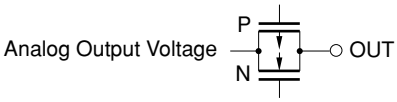


Schmitt-triggered input with hysteresis characteristics.

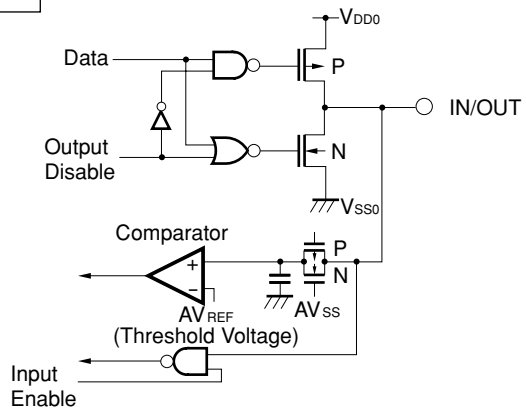
Type 5-H



Type 12



Type 20-A



2.4 CAUTIONS

(1) When connecting unused pins, if the input/output mode is undefined for an input/output alternate-function pin, it should be connected to V_{DD0} with a resistor of several tens of kΩ (especially when the reset input pin becomes the low-level input voltage or over upon powering on, and when input/output is switched by software.)

(2) P60 to P63 of the μPD784031 are in the output high-impedance state while the $\overline{\text{RESET}}$ signal is input, but output a low level after the $\overline{\text{RESET}}$ signal has been cleared. Therefore, design the external circuit so that the low level may be output as the initial status.

(3) In the μPD78233 and 78237, the TEST pin is the MODE pin and is fixed high. When changing over from the μPD78233/78237, the circuitry must be modified so that this pin is directly connected to V_{SS0}.
Modification as shown below is needed if the μPD78234/78238 was used with switching between the on-chip ROM mode and ROM-less mode performed by MODE pin switching (the TEST pin must be directly connected to V_{SS0}).

Modifications examples:

- Incorporate all programs in ROM
- Store all programs in external ROM
- Change the location address of a program previously held in external ROM, shift the address to avoid overlapping on-chip ROM, and execute this program from the program in on-chip ROM

CHAPTER 3 CPU ARCHITECTURE

3.1 MEMORY SPACE

The μPD784038 can access a 1-Mbyte memory space. The mapping of the internal data area (special function registers and internal RAM) depends on the LOCATION instruction. A LOCATION instruction must be executed after reset release, and can only be used once.

The program after reset release must be as follows:

```
RSTVCT    CSEG    AT  0
          DW      RSTSTRT
            to
INITSEG   CSEG    BASE
RSTSTRT:  LOCATION 0H              ; or LOCATION 0FH
          MOVG  SP, #STKBGN
```

(1) When LOCATION 0H instruction is executed

- **Internal memory**

  The internal data area and internal ROM area are follows:

Parts Number	Internal Data Area	Internal ROM Area
μPD784031	0F700H to 0FFFFH	—
μPD784035		00000H to 0BFFFH
μPD784036		00000H to 0F6FFH
μPD784037	0F100H to 0FFFFH	00000H to 0F0FFH 10000H to 17FFFH
μPD784038 μPD78P4038	0EE00H to 0FFFFH	00000H to 0EDFFH 10000H to 1FFFFH

Caution The following areas of the internal ROM that overlap the internal data area cannot be used when the LOCATION 0H instruction is executed.

Parts Number	Area That Cannot Be Used
μPD784035	—
μPD784036	0F700H to 0FFFFH (2,304 Bytes)
μPD784037	0F100H to 0FFFFH (3,840 Bytes)
μPD784038 μPD78P4038	0EE00H to 0FFFFH (4,608 Bytes)

- **External memory**

  The external memory is accessed in the external memory expansion mode.

(2) When LOCATION 0FH instruction is executed

- **Internal memory**

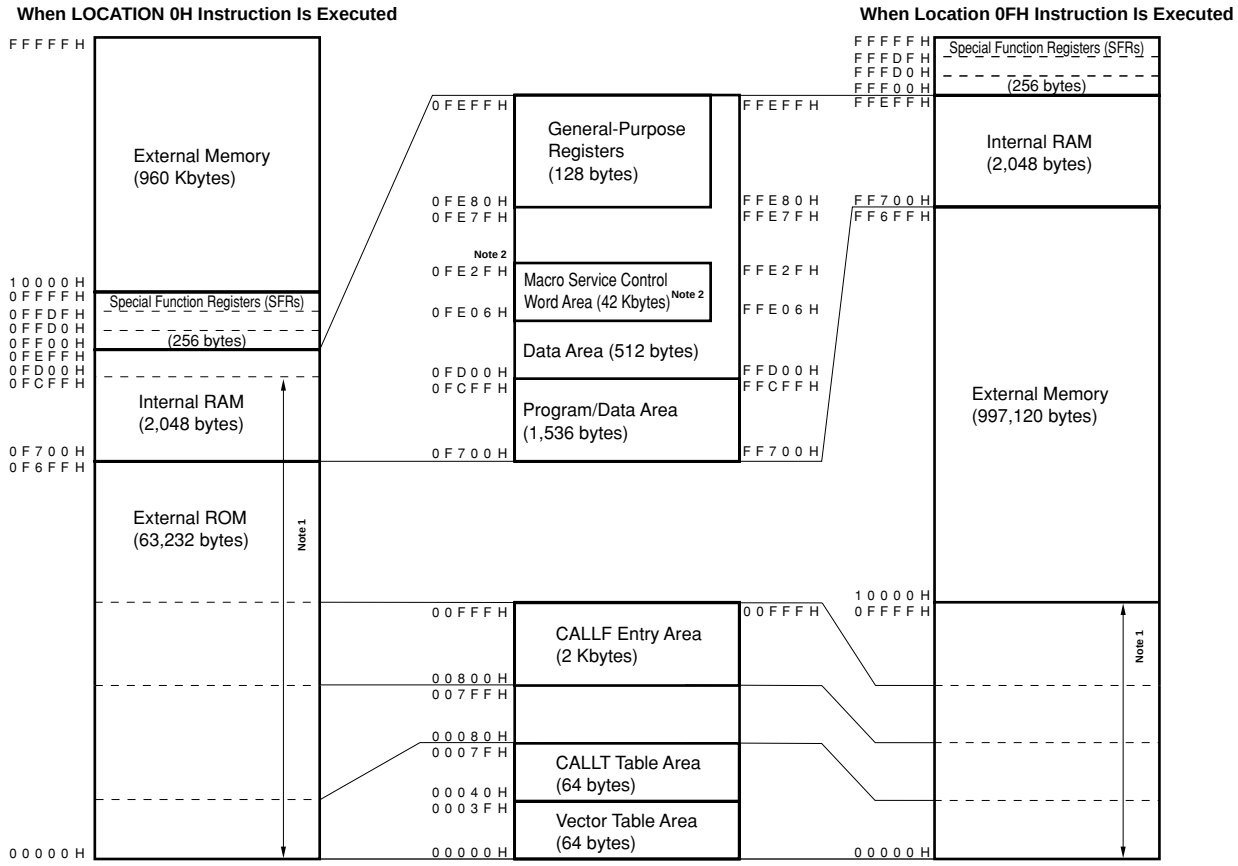
  The internal data area and internal ROM area are follows:

Parts Number	Internal Data Area	Internal ROM Area
μPD784031	FF700H to FFFFFH	—
μPD784035		00000H to 0BFFFH
μPD784036		00000H to 0FFFFH
μPD784037	FF100H to FFFFFH	00000H to 17FFFH
μPD784038 μPD78P4038	FEE00H to FFFFFH	00000H to 1FFFFH

- **External memory**

  The external memory is accessed in the external memory expansion mode.

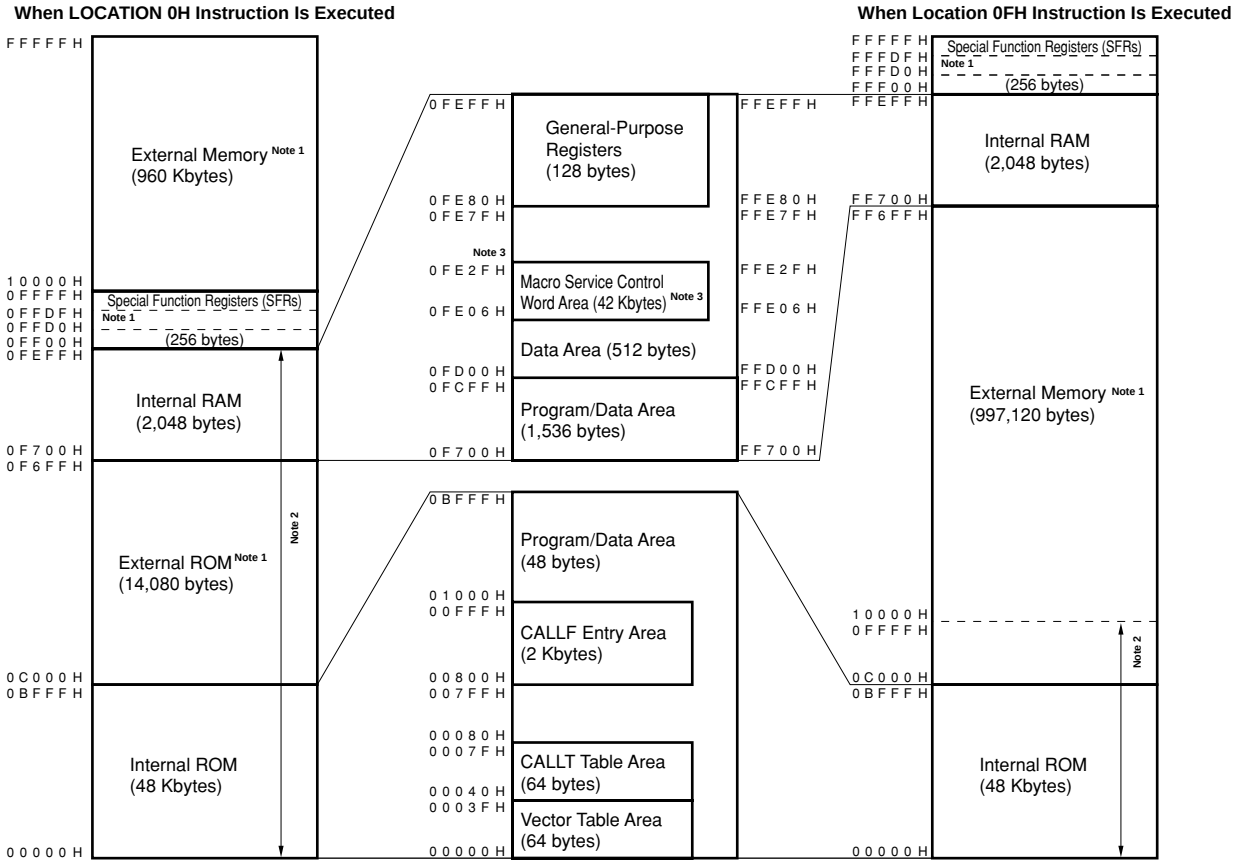
Figure 3-1 μPD784031 Memory Map



Notes 1. Base area, reset or interrupt entry area, excluding internal RAM in the case of reset.

2. 0FE31H (44 B) for the μPD784031Y.

Figure 3-2 μPD784035 Memory Map

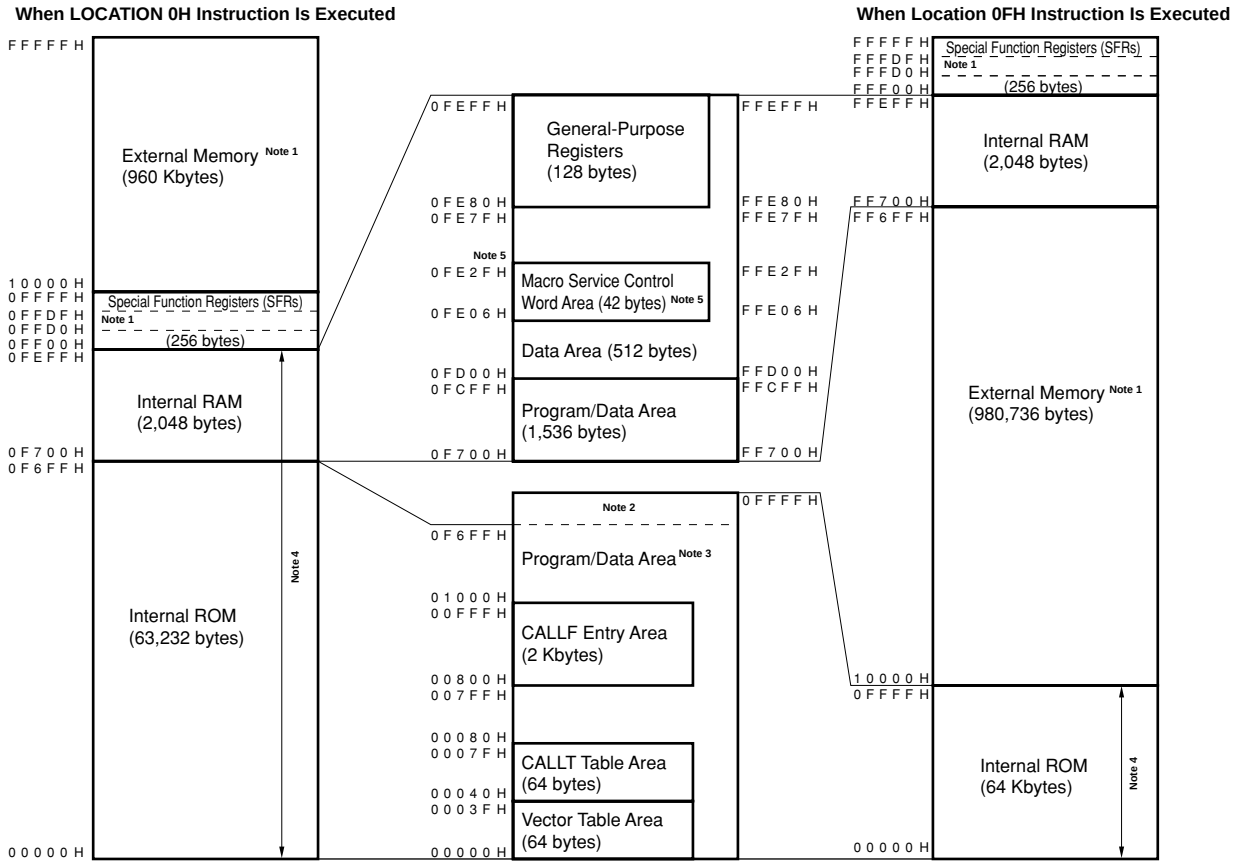


Notes 1. Accessed in external memory extension mode.

2. Base area, reset or interrupt entry area, excluding internal RAM in the case of reset.

3. 0FE31H (44 B) for the μPD784035Y.

Figure 3-3 μPD784036 Memory Map



Notes 1. Accessed in external memory extension mode.

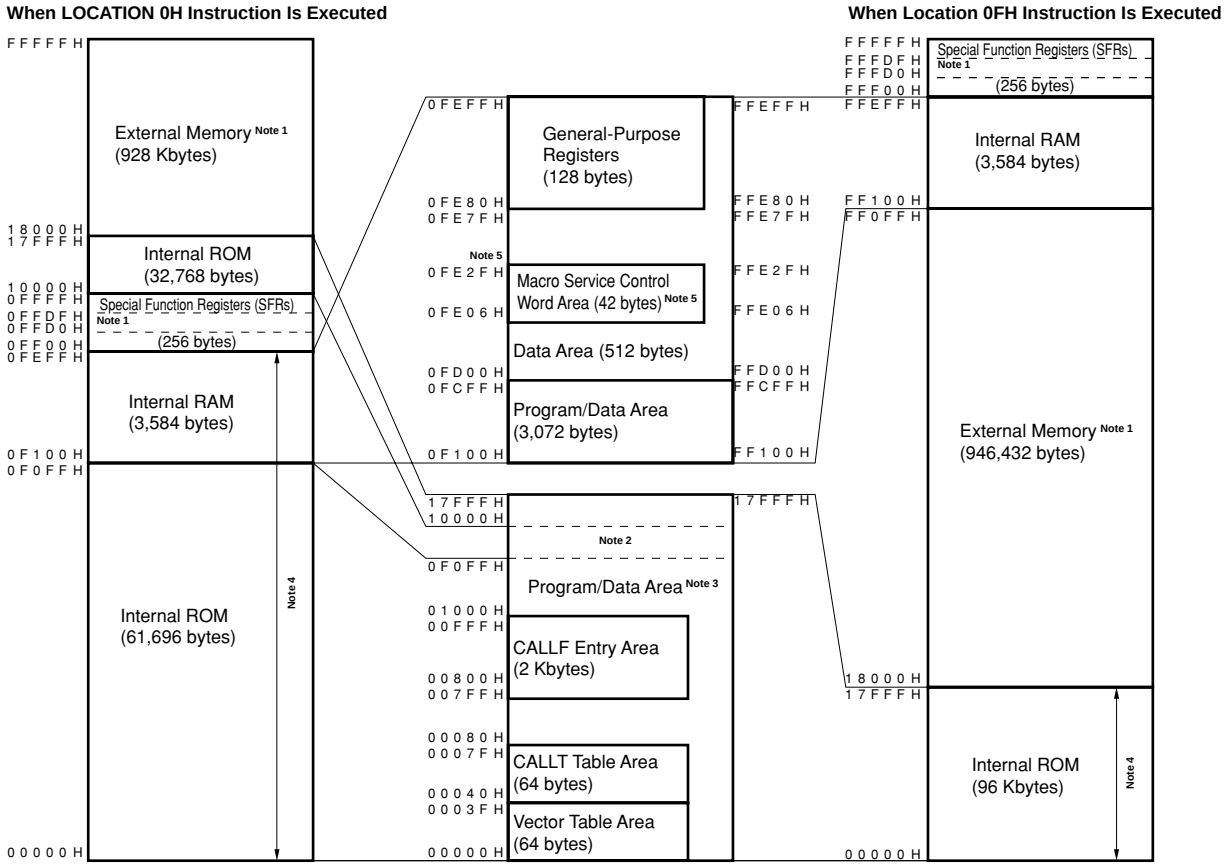
2. The 2,304 bytes of this area can be used as internal ROM only when the LOCATION 0FH instruction is executed.

3. 63,232 bytes when the LOCATION 0H is executed, and 65,536 bytes when the LOCATION 0FH instruction is executed.

4. Base area, reset or interrupt entry area, excluding internal RAM in the case of reset.

5. 0FE31H (44B) for the μPD784036Y.

Figure 3-4 μPD784037 Memory Map



Notes 1. Accessed in external memory extension mode.

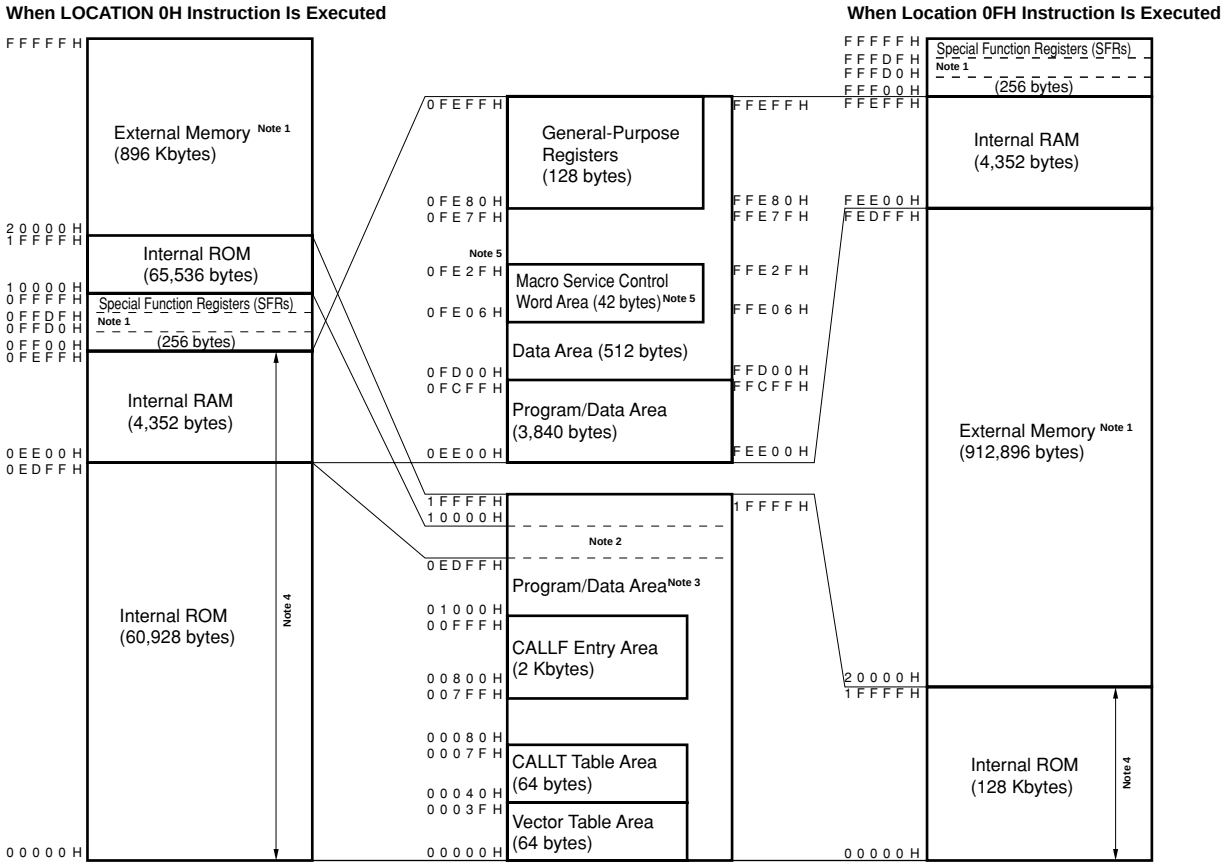
2. The 3,840 bytes of this area can be used as internal ROM only when the LOCATION 0FH instruction is executed.

3. 94,464 bytes when the LOCATION 0H is executed, and 98,304 bytes when the LOCATION 0FH instruction is executed.

4. Base area, reset or interrupt entry area, excluding internal RAM in the case of reset.

5. 0FE31H (44B) for the μPD784037Y.

Figure 3-5 μPD784038 Memory Map



Notes 1. Accessed in external memory extension mode.

2. The 4,608 bytes of this area can be used as internal ROM only when the LOCATION 0FH instruction is executed.

3. 126,464 bytes when the LOCATION 0H is executed, and 131,072 bytes when the LOCATION 0FH instruction is executed.

4. Base area, reset or interrupt entry area, excluding internal RAM in the case of reset.

5. 0FE31H (44 B) for the μPD784038Y.

3.2 INTERNAL ROM AREA

The μPD784038 Subseries products shown below incorporate ROM which is used to store programs, table data, etc.

If the internal ROM area and internal data area overlap when the LOCATION 0H instruction is executed, the internal data area is accessed, and the overlapping part of the internal ROM area cannot be accessed.

Product Name	Internal ROM	Address Space	
		LOCATION 0H Instruction	LOCATION 0FH Instruction
μPD784035	48 K × 8 bits	00000H to 0BFFFH	00000H to 0BFFFH
μPD784036	64 K × 8 bits	00000H to 0F6FFH	00000H to 0FFFFH
μPD784037	96 K × 8 bits 10000H to 17FFFH	00000H to 0F0FFH	00000H to 17FFFH
μPD784038 μPD78P4038	128 K × 8 bits	00000H to 0EDFFH 10000H to 1FFFFH	00000H to 1FFFFH

The internal ROM can be accessed at high speed. Normally, fetches are performed at the same speed as external ROM, but if the IFCH bit of the memory extension mode register (MM) is set (to 1), the high-speed fetch function is used and internal ROM fetches are performed at high speed (2-byte fetch performed in 2 system clocks).

When the instruction execution cycle equal to an external ROM fetch is selected, wait insertion is performed by the wait function, but when high-speed fetches are used, wait insertion is not performed for internal ROM. However, do not set external wait to the internal ROM area. Otherwise, the CPU may be in the deadlock status which can be cleared only by reset input.

$\overline{\text{RESET}}$ input sets the instruction execution cycle equal to the external ROM fetch cycle.

Remark This address space of the μPD784031 is in an external memory.

3.3 BASE AREA

The space from 0 to FFFFH comprises the base area. The base area is the object for the following uses:

- Reset entry address
- Interrupt entry address
- CALLT instruction entry address
- 16-bit immediate addressing mode (with instruction address addressing)
- 16-bit direct addressing mode
- 16-bit register addressing mode (with instruction address addressing)
- 16-bit register indirect addressing mode
- Short direct 16-bit memory indirect addressing mode

The vector table area, CALLT instruction table area and CALLF instruction entry area are allocated to the base area.

When the LOCATION 0H instruction is executed, the internal data area is located in the base area. Note that, in the internal data area, program fetches cannot be performed from the internal high-speed RAM area or special function register (SFR) area. Also, internal RAM area data should only be used after initialization has been performed.

3.3.1 Vector Table Area

The 64-byte area from 00000H to 0003FH is reserved as the vector table area. The vector table area stores the program start addresses used when a branch is made as the result of $\overline{\text{RESET}}$ input or generation of an interrupt request. When context switching is used by an interrupt, the number of the register bank to be switched to is stored here.

Any portion not used as the vector table can be used as program memory or data memory.

16-bit values can be written to the vector table. Therefore, branches can only be made within the base area.

Table 3-1 Vector Table

Vector Table Address	Interrupt Cause
0003CH	Operand error
0003EH	BRK
00000H	Reset ($\overline{\text{RESET}}$ input)
00002H	NMI
00004H	WDT
00006H	INTP0
00008H	INTP1
0000AH	INTP2
0000CH	INTP3
0000EH	INTC00
00010H	INTC01
00012H	INTC10
00014H	INTC11
00016H	INTC20
00018H	INTC21
0001AH	INTC30
0001CH	INTP4
0001EH	INTP5
00020H	INTAD
00022H	INTSER1
00024H	INTSR1/INTCSI1
00026H	INTST1
00028H	INTCSI
0002AH	INTSER2
0002CH	INTSR2/INTCSI2
0002EH	INTST2
00030H	INTSPC Note

Note μPD784038Y Subseries only

3.3.2 CALLT Instruction Table Area

The 1-byte call instruction (CALLT) subroutine entry addresses can be stored in the 64-byte area from 00040H to 0007FH.

The CALLT instruction references this table, and branches to a base area address written in the table as a subroutine. As the CALLT instruction is one byte in length, use of the CALLT instruction for subroutine calls written frequently throughout the program enables the program object size to be reduced. The table can contain up to 32 subroutine entry addresses, and therefore it is recommended that they be recorded in order of frequency.

If this area is not used as the CALLT instruction table, it can be used as ordinary program memory or data memory.

3.3.3 CALLF Instruction Entry Area

A subroutine call can be made directly to the area from 00800H to 00FFFH with the 2-byte call instruction (CALLF).

As the CALLF instruction is a two-byte call instruction, it enables the object size to be reduced compared with use of the direct subroutine call CALL instruction (3 or 4 bytes).

Writing subroutines directly in this area is an effective means of exploiting the high-speed capability of the device.

If you wish to reduce the object size, writing an unconditional branch (BR) instruction in this area and locating the subroutine itself outside this area will result in a reduced object size for subroutines that are called from five or more points. In this case, only the 4 bytes of the BR instruction are occupied in the CALLF entry area, enabling the object size to be reduced with a large number of subroutines.

3.4 INTERNAL DATA AREA

The internal data area consists of the internal RAM area and special function register area (see **Figures 3-1 to 3-5**).

The final address of the internal data area can be specified by means of the LOCATION instruction as either 0FFFFH (when a LOCATION 0H instruction is executed) or FFFFFH (when a LOCATION 0FH instruction is executed). Selection of the addresses of the internal data area by means of the LOCATION instruction must be executed once immediately after reset release, and once the selection is made, it cannot be changed. The program after reset release must be as shown in the example below. If the internal data area and another area are allocated to the same addresses, the internal data area is accessed and the other area cannot be accessed.

```
Example   RSTVCT    CSEG     AT  0
                   DW       RSTSTRT
                    to
          INITSEG   CSEG     BASE
          RSTSTRT:  LOCATION 0H              ; or LOCATION 0FH
                    MOVG     SP, #STKBGN
```

Caution When the LOCATION 0H instruction is executed, it is necessary to ensure that the program after reset release does not overlap the internal data area. It is also necessary to make sure that the entry addresses of the service routines for non-maskable interrupts such as NMI do not overlap the internal data area. Also, initialization must be performed for maskable interrupt entry areas, etc., before the internal data area is referenced.

3.4.1 Internal RAM Area

The μPD784038 incorporates general-purpose static RAM.

This area is configured as follows:

- Internal RAM area
  - Peripheral RAM (PRAM)
  - Internal high-speed RAM (IRAM)

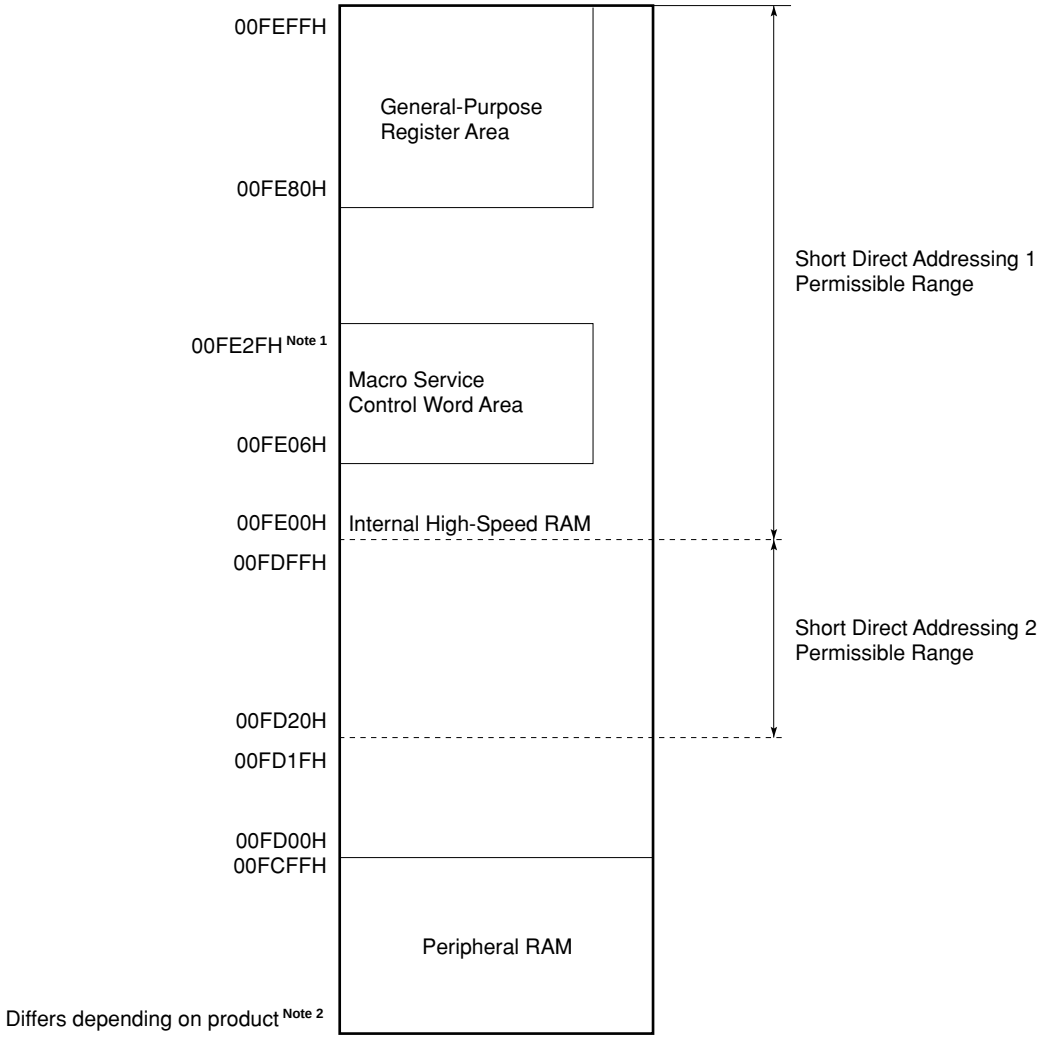
Table 3-2 Internal RAM Area

Internal RAM / Product Name	Internal RAM Area	Peripheral RAM: PRAM	Internal High-Speed RAM: IRAM
μPD784031 μPD784035 μPD784036	2,048 bytes (0F700H to 0FEFFH)	1,536 bytes (0F700H to 0FCFFH)	512 bytes (0FD00H to 0FEFFH)
μPD784037	3,584 bytes (0F100H to 0FEFFH)	3,072 bytes (0F100H to 0FCFFH)	
μPD784038 μPD78P4038	4,352 bytes (0EE00H to 0FEFFH)	3,840 bytes (0EE00H to 0FCFFH)	

Remark The addresses in the table are the values that apply when the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, 0F0000H should be added to the values shown above.

The internal RAM memory map is shown in Figure 3-6.

Figure 3-6 Internal RAM Memory Map



Notes 1. 00FE31H for μPD784038Y Subseries.

2. μPD784031, 784035, 784036 : 00F700H
μPD784037 : 00F100H
μPD784038, 78P4038 : 00EE00H

Remark The addresses in the figure are the values that apply when the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, 0F0000H should be added to the values shown above.

(1) Internal high-speed RAM (IRAM)

The internal high-speed RAM (IRAM) allows high-speed accesses to be made. The short direct addressing mode for high-speed accesses can be used on FD20H to FEFFH in this area. There are two kinds of short direct addressing mode, short direct addressing 1 and short direct addressing 2, according to the target address. The function is the same in both of these addressing modes. With some instructions, the word length is shorter with short direct addressing 2 than with short direct addressing 1. See the **78K/IV Series User's Manual Instruction Volume** for details.

A program fetch cannot be performed from IRAM. If a program fetch is performed from an address onto which IRAM is mapped, CPU inadvertent loop will result.

The following areas are reserved in IRAM.

- General-purpose register area : FE80H to FEFFH
- Macro service control word area : FE06H to FE2FH (excluding 0FE22H, 0FE23H, 0FE2AH, 0FE2BH)
- Macro service channel area : FE00H to FEFFH (the address is specified by the macro service control word)

If the reserved function is not used in these areas, they can be used as ordinary data memory.

Remark The addresses in this text are those that apply when the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, 0F0000H should be added to the values shown in the text.

(2) Peripheral RAM (PRAM)

The peripheral RAM (PRAM) is used as ordinary program memory or data memory. When used as program memory, the program must be written to the peripheral RAM beforehand by a program.

Program fetches from peripheral RAM are fast, with a 2-byte fetch being executed in 2 clocks.

3.4.2 Special Function Register (SFR) Area

The on-chip peripheral hardware special function registers (SFRs) are mapped onto the area from 0FF00H to 0FFFFH (see **Figures 3-1** to **3-5**).

The area from 0FFD0H to 0FFDFH is mapped as an external SFR area, and allows externally connected peripheral I/Os, etc., to be accessed in external memory extension mode (specified by the memory extension mode register (MM)) by the ROM-less product or on-chip ROM products.

Caution Addresses onto which SFRs are not mapped should not be accessed in this area. If such an address is accessed by mistake, the CPU may become deadlocked. A deadlock can only be released by reset input.

Remark The addresses in this text are those that apply when the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, 0F0000H should be added to the values shown in the text.

3.4.3 External SFR Area

In μPD784038 Subseries products, the 16-byte area from 0FFD0H to 0FFDFH in the SFR area (when the LOCATION 0H is executed; 0FFFD0H to 0FFFDFH when the LOCATION 0FH instruction is executed) is mapped as an external SFR area. When the external memory extension mode is set in a ROM-less product or on-chip ROM product, externally connected peripheral I/Os, etc., can be accessed using the address bus or address/data bus, etc.

As the external SFR area can be accessed by SFR addressing, peripheral I/O and similar operations can be performed easily, the object size can be reduced, and macro service can be used.

Bus operations for accesses to the external SFR area are performed in the same way as for ordinary memory accesses.

3.5 EXTERNAL MEMORY SPACE

The external memory space is a memory space that can be accessed in accordance with the setting of the memory extension mode register (MM). It can store programs, table data, etc., and can have peripheral I/O devices allocated to it.

3.6 μPD78P4038 MEMORY MAPPING

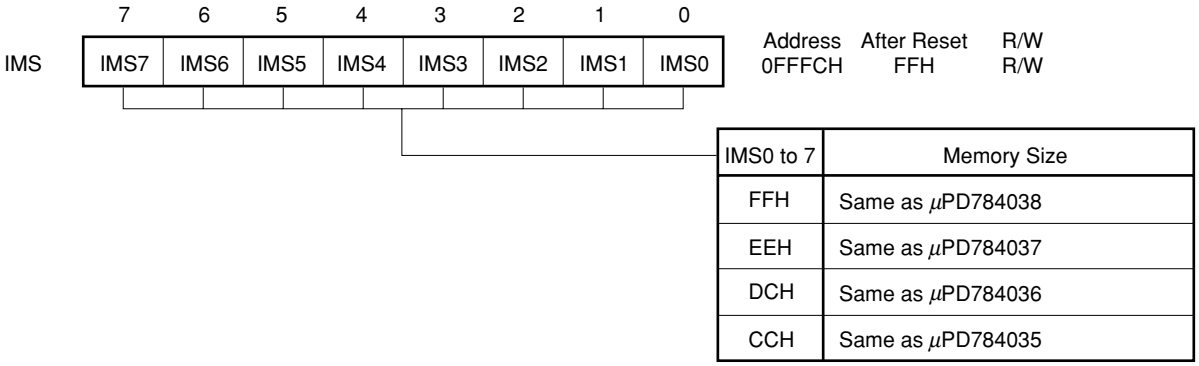
The μPD78P4038 incorporates 128-Kbyte internal ROM and 4,352-byte internal RAM. Therefore, the memory mapping of the μPD78P4038 is slightly different from that of the μPD784035, 784036, and 784037. In order to mask this difference, the μPD78P4038 has a function (the memory size switching function) which prevents part of the internal memory from being used by software.

Memory size switching is performed by means of the internal memory size switching register (IMS). To make the memory mapping of the μPD78P4038 the same as that of the μPD784035, 784036, and 784037, be sure to write this register immediately after reset. Do not change the written value.

The IMS can be written to with an 8-bit manipulation instruction. The IMS format is shown in **Figure 3-7**.

$\overline{\text{RESET}}$ input sets the IMS register to FFH.

Figure 3-7 Internal Memory Size Switching Register (IMS) Format



IMS0 to 7	Memory Size
FFH	Same as μPD784038
EEH	Same as μPD784037
DCH	Same as μPD784036
CCH	Same as μPD784035

IMS is not provided to the μPD784035, 784036, 784037, and 784038. However, the operation is not affected even if an instruction to write IMS is executed with these models.

Caution If the μPD78P4038 is selected as the emulation CPU when the in-circuit emulator is used, the memory size is always the same as the μPD784038 even if a write instruction other than FFH (EEH, DCH, or CCH) is executed to the IMS.

3.7 CONTROL REGISTERS

Control registers consist of the program counter (PC), program status word (PSW), and stack pointer (SP).

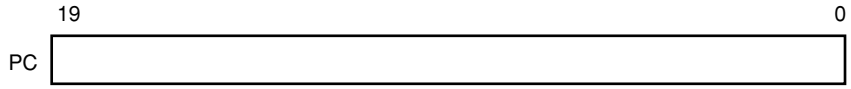
3.7.1 Program Counter (PC)

This is a 20-bit binary counter that holds address information on the next program to be executed (see **Figure 3-8**).

Normally, the PC is incremented automatically by the number of bytes in the fetched instruction. When an instruction associated with a branch is executed, the immediate data or register contents are set in the PC.

Upon $\overline{\text{RESET}}$ input, the 16-bit data in address 0 and 1 is set in the low-order 16 bits, and 0000 in the high-order 4 bits of the PC.

Figure 3-8 Program Counter (PC) Format



3.7.2 Program Status Word (PSW)

The program status word (PSW) is a 16-bit register comprising various flags that are set or reset according to the result of instruction execution.

Read accesses and write accesses are performed in high-order 8-bit (PSWH) and low-order 8-bit (PSWL) units. Individual flags can be manipulated by bit-manipulation instructions.

The contents of the PSW are automatically saved to the stack when a vectored interrupt request is acknowledged or a BRK instruction is executed, and automatically restored when an RETI or RETB instruction is executed. When context switching is used, the contents are automatically saved in RP3, and automatically restored when an RETCS or RETCSB instruction is executed.

$\overline{\text{RESET}}$ input resets (to 0) all bits.

"0" must always be written to the bits written as "0" in **Figure 3-9**. The contents of bits written as "-" are undefined when read.

Figure 3-9 Program Status Word (PSW) Format

Symbol	7	6	5	4	3	2	1	0
PSWH	UF	RBS2	RBS1	RBS0	–	–	–	–

	7	6	5	4	3	2	1	0
PSWL	S	Z	RSS	AC	IE	P/V	0	CY

The flags are described below.

(1) Carry flag (CY)

The carry flag records a carry or borrow resulting from an operation.

This flag also records the shifted-out value when a shift/rotate instruction is executed, and functions as a bit accumulator when a bit-manipulation instruction is executed.

The status of the CY flag can be tested with a conditional branch instruction.

(2) Parity/overflow flag (P/V)

The P/V flag performs the following two kinds of operation associated with execution of an operation instruction.
The status of the P/V flag can be tested with a conditional branch instruction.

- Parity flag operation
  Set (to 1) when the number of bits set (to 1) as the result of execution of a logical operation instruction, shift/rotate instruction, or a CHKL or CHKLA instruction is even, and reset (to 0) if odd. When a 16-bit shift instruction is executed, however, only the low-order 8 bits of the operation result are valid for the parity flag.

- Overflow flag operation
  Set (to 1) only when the numeric range expressed as a two's complement is exceeded as the result of execution of a arithmetic operation instruction, and reset (to 0) otherwise. More specifically, the value of this flag is the exclusive OR of the carry into the MSB and the carry out of the MSB. For example, the two's complement range in an 8-bit arithmetic operation is 80H (–128) to 7FH (+127), and the flag is set (to 1) if the operation result is outside this range, and reset (to 0) if within this range.

Example The operation of the overflow flag when an 8-bit addition instruction is executed is shown below.

When the addition of 78H (+120) and 69H (+105) is performed, the operation result is E1H (+225), and the two's complement limit is exceeded, with the result that the P/V flag is set (to 1). Expressed as a two's complement, E1H is -31.

```
    78H (+120)   =      0,111  1,000
+)  69H (+105)   = +)   0,110  1,001
                    0   1,110  0,001  =  –31   P/V = 1
                    ↑
                    CY
```

When the following two negative numbers are added together, the operation result is within the two's complement range, and therefore the P/V flag is reset (to 0).

```
    FBH (–5)     =      1,111  1,011
+)  F0H (–16)    = +)   1,111  0,000
                    1   1,110  1,011  =  –21   P/V = 0
                    ↑
                    CY
```

(3) Interrupt request enable flag (IE)

This flag controls CPU interrupt request acknowledgment operations.
When "0", interrupts are disabled, and only non-maskable interrupts and unmasked macro service can be acknowledged. All other interrupts are disabled.
When "1", the interrupt enabled state is set, and enabling of interrupt request acknowledgment is controlled by the interrupt mask flags corresponding to the individual interrupt requests and the priority of the individual interrupts.
The IE flag is set (to 1) by execution of an EI instruction, and reset (to 0) by execution of a DI instruction or acknowledgment of an interrupt.

(4) Auxiliary carry flag (AC)

The AC flag is set (to 1) when there is a carry out of bit 3 or a borrow into bit 3 as the result of an operation, and reset (to 0) otherwise.
This flag is used when the ADJBA or ADJBS instruction is executed.

(5) Register set selection flag (RSS)

The RSS flag specifies the general-purpose registers that function as X, A, C, and B, and the general-purpose register pairs (16-bit) that function as AX and BC.

This flag is provided to maintain compatibility with the 78K/III Series, and must be set to 0 except when using a 78K/III Series program.

(6) Zero flag (Z)

The Z flag records the fact that the result of an operation is “0”.

It is set (to 1) when the result of an operation is “0”, and reset (to 0) otherwise. The status of the Z flag can be tested with a conditional branch instruction.

(7) Sign flag (S)

The S flag records the fact that the MSB is “1” as the result of an operation.

It is set (to 1) when the MSB is “1” as the result of an operation, and reset (to 0) otherwise. The status of the S flag can be tested with a conditional branch instruction.

(8) Register bank selection flag (RBS0 to RBS2)

This is a 3-bit flag used to select one of the 8 register banks (register bank 0 to register bank 7) (see **Table 3-3**).

It stores 3-bit information which indicates the register bank selected by execution of a SEL RBn instruction, etc.

Table 3-3 Register Bank Selection

RBS2	RBS1	RBS0	Specified Register Bank
0	0	0	Register bank 0
0	0	1	Register bank 1
0	1	0	Register bank 2
0	1	1	Register bank 3
1	0	0	Register bank 4
1	0	1	Register bank 5
1	1	0	Register bank 6
1	1	1	Register bank 7

(9) User flag (UF)

This flag can be set and reset in the user program, and used for program control.

3.7.3 Use of RSS Bit

Basically, the RSS bit should be fixed at 0 at all times.

The following explanation refers to the case where a 78K/III Series program is used, and the program used sets the RSS bit to 1. This explanation can be skipped if the RSS bit is fixed at 0.

The RSS bit is provided to allow the functions of A (R1), X (R0), B (R3), C (R2), AX (RP0), and BC (RP1) to be used by registers R4 to R7 (RP2, RP3) as well. Effective use of this bit enables efficient programs to be written in terms of program size and program execution.

However, careless use can result in unforeseen problems. Therefore, the RSS bit should always be set to 0. The RSS bit should only be set to 1 when a 78K/III Series program is used.

Use of the RSS bit set to 0 in all programs will improve programming and debugging efficiency.

Even when using a program in which the RSS bit set to 1 is used, it is recommended that the program be amended if possible so that it does not set the RSS bit to 1.

(1) RSS bit specification

- Registers used by instructions for which the A, X, B, C, and AX registers are directly entered in the operand column of the operation list (see **27.2.**)

- Registers specified as implied by instructions that use the A, AX, B, and C registers by means of implied addressing

- Registers used in addressing by instructions that use the A, B, and C registers in indexed addressing and based indexed addressing

The registers used in these cases are switched as follows according to the RSS bit.

- When RSS = 0
  A→R1, X→R0, B→R3, C→R2, AX→RP0, BC→RP1

- When RSS = 1
  A→R5, X→R4, B→R7, C→R6, AX→RP2, BC→RP3

Registers used other than those mentioned above are always the same irrespective of the value of the RSS bit. With the NEC Electronics assembler (RA78K4), the register operation code generated when the A, X, B, C, AX, and BC registers are described by those names is determined by the assembler RSS pseudo-instruction.

When the RSS bit is set or reset, an RSS pseudo-instruction must be written immediately before (or immediately after) the relevant instruction (see example below).

<Program example>

- When RSS is set to 0

```
RSS   0        ;  RSS pseudo-instruction
CLR1  PSWL.5
MOV   B, A     ;  This description is equivalent to “MOV R3, R1”.
```

- When RSS is set to 1

```
RSS   1        ;  RSS pseudo-instruction
SET1  PSWL.5
MOV   B, A     ;  This description is equivalent to “MOV R7, R5”.
```

(2) Operation code generation method with RA78K4

- With RA78K4, if there is an instruction with the same function as an instruction for which A or AX is directly entered in the operand column of the instruction operation list, the operation code for which A or AX is directly entered in the operand column is generated first.

Example The function is the same when B is used as r in a MOV A, r instruction, and when A is used as r and B is used as r’ in a MOV r, r’ instruction, and the same description (MOV A, B) is used in the assembler source program. In this case, RA78K4 generates code equivalent to the MOV A, r instruction.

- If A, X, B, C, AX, or BC is written in an instruction for which r, r’, rp, and rp’ are specified in the operand column, the A, X, B, C, AX, and BC instructions generate an operation code that specifies the following registers according to the operand of the RA78K4 RSS pseudo-instruction.

Register	RSS = 0	RSS = 1
A	R1	R5
X	R0	R4
B	R3	R7
C	R2	R6
AX	RP0	RP2
BC	RP1	RP3

- If R0 to R7 or RP0 to RP4 is written as r, r’, rp or rp’ in the operand column, an operation code in accordance with that specification is output (an operation code for which A or AX is directly entered in the operand column is not output.)

- Descriptions R1, R3, R2 or R5, R7, R6 cannot be used for registers A, B, and C used in indexed addressing and based indexed addressing.

(3) Operating precautions

Switching the RSS bit has the same effect as having two register sets. However, when writing a program, care must be taken to ensure that the static program description and dynamic RSS bit changes at the time of program execution always coincide.

Also, a program that sets RSS to 1 cannot be used by a program that uses the context switching function, and therefore program usability is poor. Moreover, since different registers are used with the same name, program readability is poor and debugging is difficult. Therefore, if it is necessary to set RSS to 1, these disadvantages must be fully taken into consideration when writing a program.

A register not specified by the RSS bit can be accessed by writing its absolute name.

3.7.4 Stack Pointer (SP)

The stack pointer is a 24-bit register that holds the start address of the stack area (LIFO type: 00000H to FFFFFFH) (see **Figure 3-10**). It is used to address the stack area when subroutine processing or interrupt servicing is performed. Be sure to write “0” in the high-order 4 bits.

The contents of the SP are decremented before a write to the stack area and incremented after a read from the stack area (see **Figures 3-11 and 3-12**).

The SP is accessed by dedicated instructions.

The SP contents are undefined after $\overline{\text{RESET}}$ input, and therefore the SP must always be initialized by an initialization program directly after reset release (before a subroutine call or interrupt acknowledgment).

Example SP initialization

```
MOVG SP, #0FEE0H; SP ← 0FEE0H (when used from FEDFH)
```

Figure 3-10 Stack Pointer (SP) Format

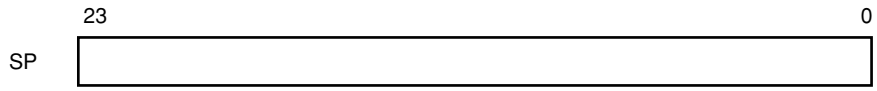


Figure 3-11 Data Saved to Stack Area

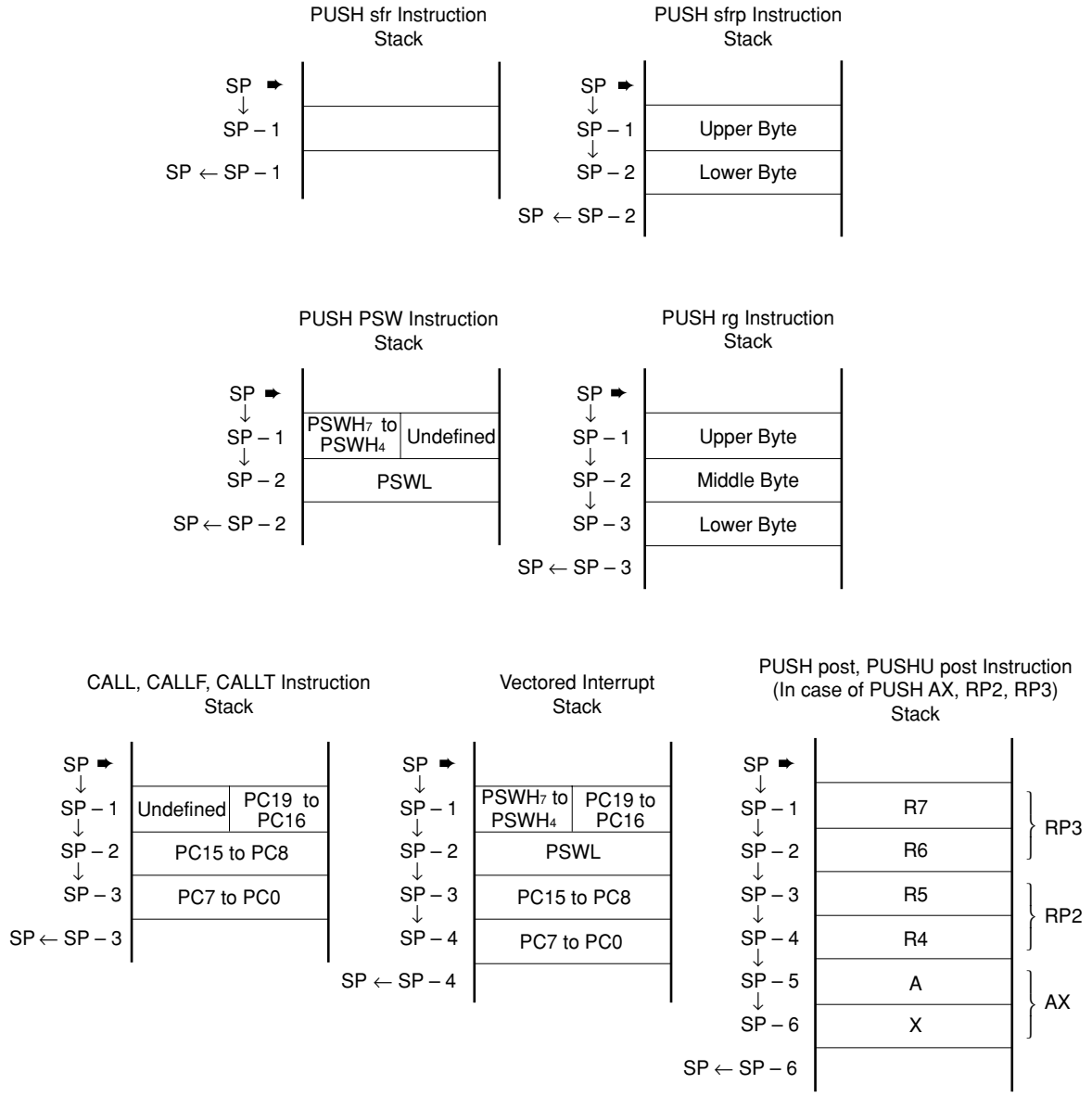
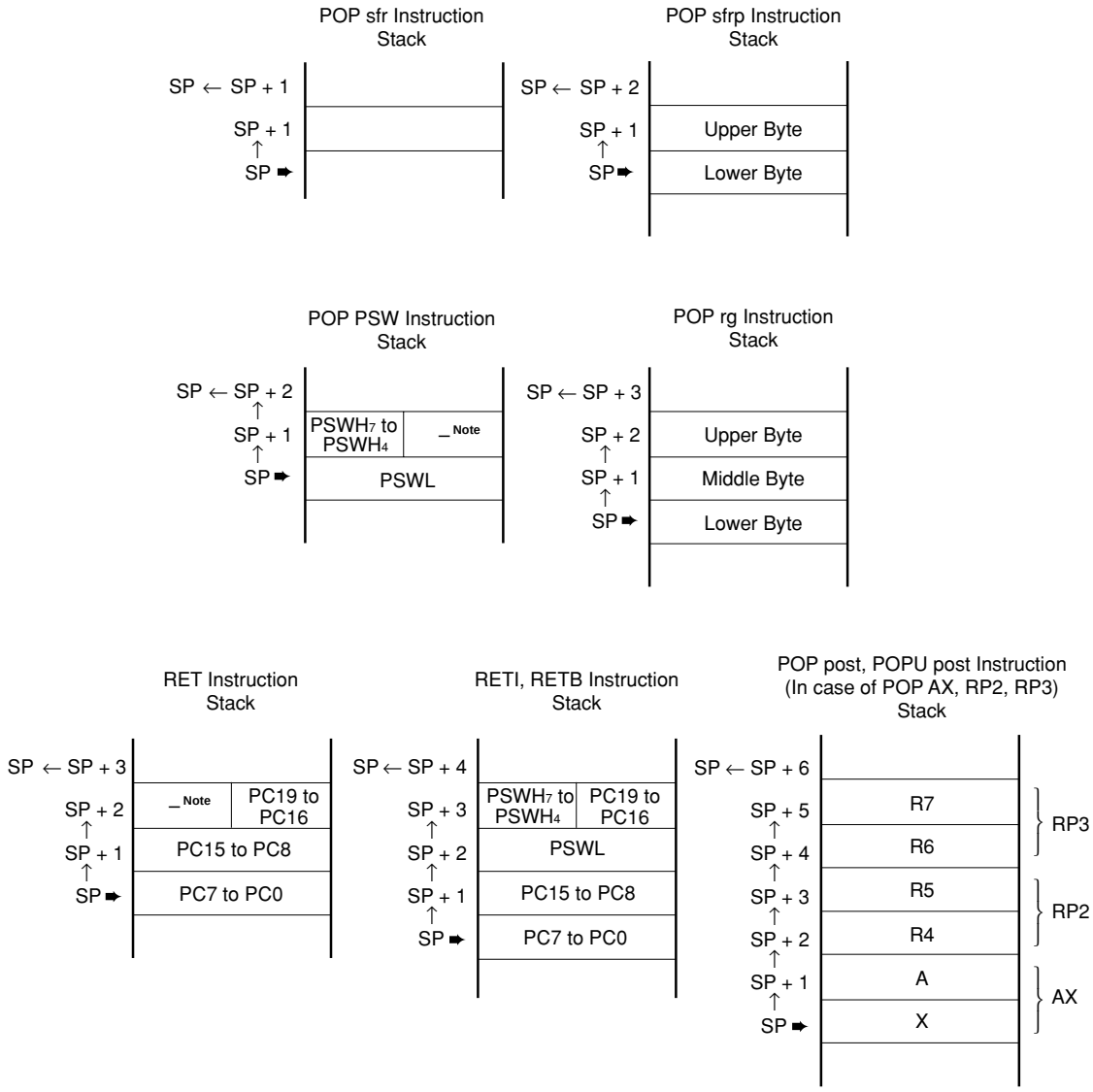


Figure 3-12 Data Restored from Stack Area



Note This 4-bit data is ignored.

Cautions 1. With stack addressing, the entire 1-Mbyte space can be accessed but a stack area cannot be reserved in the SFR area or internal ROM area.

2. The stack pointer (SP) is undefined after $\overline{\text{RESET}}$ input. Moreover, non-maskable interrupts can still be acknowledged when the SP is in an undefined state. An unanticipated operation may therefore be performed if a non-maskable interrupt request is generated when the SP is in the undefined state directly after reset release. To avoid this risk, the program after reset release must be written as follows.

```
RSTVCT    CSEG   AT       0
          DW     RSTSTRT
           to
INITSEG   CSEG   BASE
RSTSTRT : LOCATION 0H          ; or LOCATION 0FH
          MOVG SP, #STKBGN
```

3.8 GENERAL REGISTERS

3.8.1 Configuration

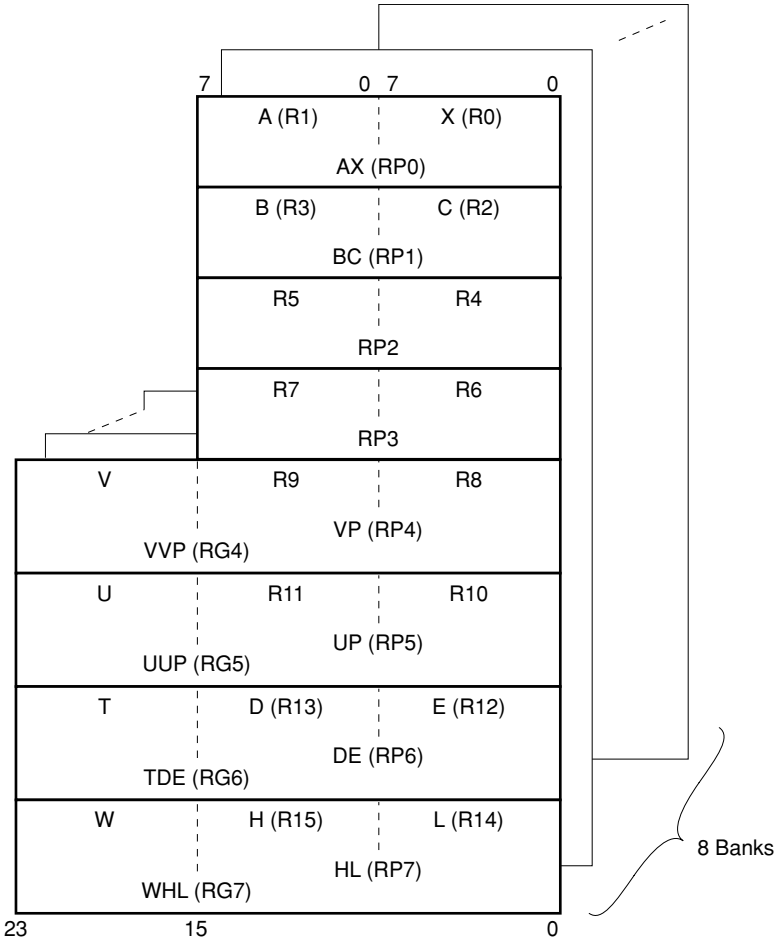
There are sixteen 8-bit general-purpose registers, and two 8-bit general-purpose registers can be used together as a 16-bit general-purpose register. In addition, four of the 16-bit general-purpose registers can be combined with an 8-bit register for address extension, and used as 24-bit address specification registers.

General-purpose registers other than the V, U, T, and W registers for address extension are mapped onto internal RAM.

These register sets are provided in 8 banks, and can be switched by means of software or the context switching function.

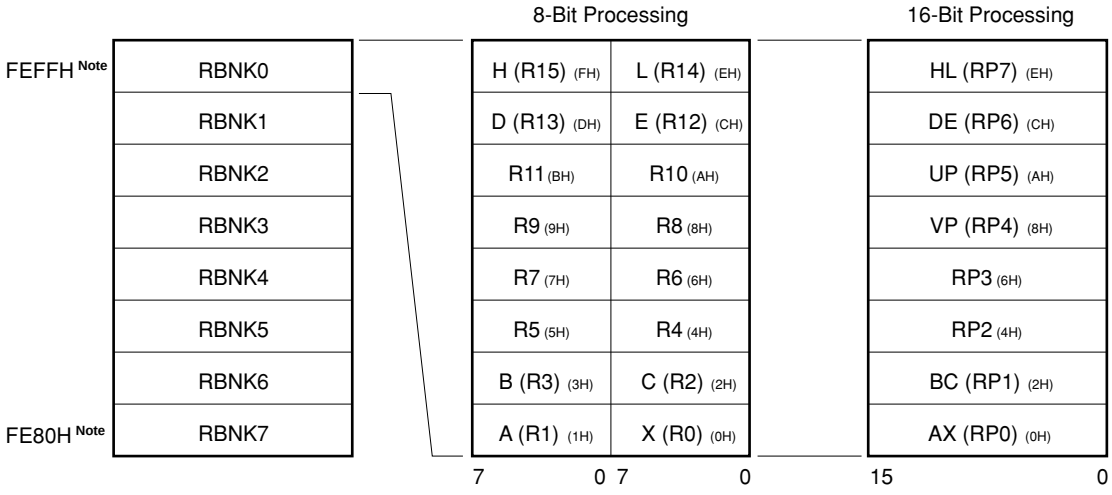
Upon $\overline{\text{RESET}}$ input, register bank 0 is selected. The register bank used during program execution can be checked by reading the register bank selection flag (RBS0, RBS1, RBS2) in the PSW.

Figure 3-13 General-Purpose Register Format



Remark Absolute names are shown in parentheses.

Figure 3-14 General-Purpose Register Addresses



Note When the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, 0F0000H should be added to the address values shown above.

Caution R4, R5, R6, R7, RP2, and RP3 can be used as the X, A, C, B, AX, and BC registers respectively by setting the RSS bit of the PSW to 1, but this function should only be used when using a 78K/III Series program.

Remark When the register bank is changed, and it is necessary to return to the original register bank, an SEL RBn instruction should be executed after saving the PSW to the stack with a PUSH PSW instruction. When returning to the original register bank, if the stack location does not change the POP PSW instruction should be used. When the register bank is changed by a vectored interrupt service program, etc., the PSW is automatically saved to the stack when an interrupt is acknowledged and restored by an RETI or RETB instruction, so that, if only one register bank is used in the interrupt service routine, only an SEL RBn instruction needs be executed, and execution of a PUSH PSW and POP PSW instruction is not necessary.

Example When register bank 2 is specified

```
      .
      .
      .
PUSH PSW
SEL RB2
      .
      .
      .                  } Operations in register bank 2
POP PSW
      .
      .
      .                    Operations in original register bank
```

3.8.2 Functions

In addition to being manipulated in 8-bit units, the general-purpose registers can also be manipulated in 16-bit units by pairing two 8-bit registers. Also, four of the 16-bit registers can be combined with an 8-bit register for address extension and manipulated in 24-bit units.

Each register can be used in a general-purpose way for temporary storage of an operation result and as the operand of an inter-register operation instruction.

The area from 0FE80H to 0FEFFH (when the LOCATION 0H instruction is executed; 0FFE80H to 0FFEFFH when the LOCATION 0FH instruction is executed) can be given an address specification and accessed as ordinary data memory irrespective of whether or not it is used as the general-purpose register area.

As 8 register banks are provided in the 78K/IV Series, efficient programs can be written by using different register banks for normal processing and processing in the event of an interrupt.

The registers have the following specific functions.

A (R1):

- Register mainly used for 8-bit data transfers and operation processing. Can be used in combination with all addressing modes for 8-bit data.
- Can also be used for bit data storage.
- Can be used as the register that stores the offset value in indexed addressing and based indexed addressing.

X (R0):

- Can be used for bit data storage.

AX (RP0):

- Register mainly used for 16-bit data transfers and operation processing. Can be used in combination with all addressing modes for 16-bit data.

AXDE:

- Used for 32-bit data storage when a DIVUX, MACW, or MACSW instruction is executed.

B (R3):

- Has a loop counter function, and can be used by the DBNZ instruction.
- Can be used as the register that stores the offset value in indexed addressing and based indexed addressing.
- Used as the MACW and MACSW instruction data pointer.

C (R2):

- Has a loop counter function, and can be used by the DBNZ instruction.
- Can be used as the register that stores the offset value in based indexed addressing.
- Used as the counter in a string instruction and the SACW instruction.
- Used as the MACW and MACSW instruction data pointer.

RP2:

- Used to save the low-order 16 bits of the program counter (PC) when context switching is used.

RP3:

- Used to save the high-order 4 bits of the program counter (PC) and the program status word (PSW) (excluding bits 0 to 3 of PSWH) when context switching is used.

VVP (RG4):

- Has a pointer function, and operates as the register that specifies the base address in register indirect addressing, based addressing and based indexed addressing.

UUP (RG5):

- Has a user stack pointer function, and enables a stack separate from the system stack to be implemented by means of the PUSHU and POPU instructions.
- Has a pointer function, and operates as the register that specifies the base address in register indirect addressing and based addressing.

DE (RP6), HL (RP7):

- Operate as the registers that store the offset value in indexed addressing and based indexed addressing.

TDE (RG6):

- Has a pointer function, and operates as the register that specifies the base address in register indirect addressing and based addressing.
- Used as the pointer in a string instruction and the SACW instruction.

WHL (RG7):

- Register used mainly for 24-bit data transfers and operation processing.
- Has a pointer function, and operates as the register that specifies the base address in register indirect addressing and based addressing.
- Used as the pointer in a string instruction and the SACW instruction.

In addition to the function name that emphasizes the specific function of the register (X, A, C, B, E, D, L, H, AX, BC, VP, UP, DE, HL, VVP, UUP, TDE, WHL), each register can also be described by its absolute name (R0 to R15, RP0 to RP7, RG4 to RG7). The correspondence between these names is shown in Table 3-4.

Table 3-4 Correspondence between Function Names and Absolute Names

(a) 8-bit registers

Absolute Name	Function Name	
	RSS = 0	RSS = 1 Note
R0	X	
R1	A	
R2	C	
R3	B	
R4		X
R5		A
R6		C
R7		B
R8		
R9		
R10		
R11		
R12	E	E
R13	D	D
R14	L	L
R15	H	H

(b) 16-bit registers

Absolute Name	Function Name	
	RSS = 0	RSS = 1 Note
RP0	AX	
RP1	BC	
RP2		AX
RP3		BC
RP4	VP	VP
RP5	UP	UP
RP6	DE	DE
RP7	HL	HL

(c) 24-bit registers

Absolute Name	Function Name
RG4	VVP
RG5	UUP
RG6	TDE
RG7	WHL

Note RSS should only be set to 1 when a 78K/III Series program is used.

Remark R8 to R11 have no function name.

3.9 SPECIAL FUNCTION REGISTERS (SFRS)

These are registers to which a special function is assigned, such as on-chip peripheral hardware mode registers, control registers, etc. They are mapped onto the 256-byte space from 0FF00H to 0FFFFH [Note].

Note When the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, the area is FFF00H to FFFFFH.

Caution Addresses onto which SFRs are not assigned should not be accessed in this area. If such an address is as accessed by mistake, the μPD784038 may become deadlocked. A deadlock can only be released by reset input.

A list of special function registers (SFRs) is given in Table 3-5. The meaning of the items in the table is as explained below.

- Symbol ..... Symbol that indicates the incorporated SFR. This is a reserved word in the NEC Electronics assembler (RA78K4). With the C compiler (CC78K4), this symbol can be used as an sfr variable by means of a #pragma sfr command.
- R/W ..... Indicates whether the corresponding SFR is read/write enabled.
  - R/W: Read/write enabled
  - R : Read-only
  - W : Write-only
- Manipulable Bit Units ..... Indicates the applicable manipulation bit units when the corresponding SFR is manipulated.
  A 16-bit-manipulable SFR can be written in the operand “sfrp”, and when specified by an address, an even address is specified.
  A bit-manipulable SFR can be written in a bit manipulation instruction.
- After Reset ..... Indicates the status of the register after $\overline{\text{RESET}}$ input.

Table 3-5 List of Special Function Registers (SFRs) (1/4)

Address [Note 1]	Special Function Register (SFR) Name	Symbol		R/W	Manipulable Bit Units			After Reset
					1 Bit	8 Bits	16 Bits	
0FF00H	Port 0	P0		R/W	√	√	—	Undefined
0FF01H	Port 1	P1			√	√	—	
0FF02H	Port 2	P2		R	√	√	—	
0FF03H	Port 3	P3		R/W	√	√	—	
0FF04H	Port 4 [Note 2]	P4			√	√	—	
0FF05H	Port 5 [Note 2]	P5			√	√	—	
0FF06H	Port 6	P6			√	√	—	00H
0FF07H	Port 7	P7			√	√	—	Undefined
0FF0EH	Port 0 buffer register	P0L			√	√	—	
0FF0FH	Port 0 buffer register H	P0H			√	√	—	
0FF10H	Compare register (timer/counter 0)	CR00			—	—	√	
0FF12H	Capture/compare register (timer/counter 0)	CR01			—	—	√	
0FF14H	Compare register L (timer/counter 1)	CR10	CR10W		—	√	√	
0FF15H	Compare register H (timer/counter 1)	—			—	—		
0FF16H	Capture/compare register L (timer/counter 1)	CR11	CR11W		—	√	√	
0FF17H	Capture/compare register H (timer/counter 1)	—			—	—		
0FF18H	Compare register L (timer/counter 2)	CR20	CR20W		—	√	√	
0FF19H	Compare register H (timer/counter 2)	—			—	—		
0FF1AH	Capture/compare register L (timer/counter 2)	CR21	CR21W		—	√	√	
0FF1BH	Capture/compare register H (timer/counter 2)	—			—	—		
0FF1CH	Compare register L (timer 3)	CR30	CR30W		—	√	√	
0FF1DH	Compare register H (timer 3)	—			—	—		
0FF20H	Port 0 mode register	PM0			√	√	—	FFH
0FF21H	Port 1 mode register	PM1			√	√	—	
0FF23H	Port 3 mode register	PM3			√	√	—	
0FF24H	Port 4 mode register [Note 2]	PM4			√	√	—	
0FF25H	Port 5 mode register [Note 2]	PM5			√	√	—	
0FF26H	Port 6 mode register	PM6			√	√	—	
0FF27H	Port 7 mode register	PM7			√	√	—	
0FF2EH	Real-time output port control register	RTPC			√	√	—	00H
0FF30H	Capture/compare control register 0	CRC0			—	√	—	10H
0FF31H	Timer output control register	TOC			√	√	—	00H
0FF32H	Capture/compare control register 1	CRC1			—	√	—	
0FF33H	Capture/compare control register 2	CRC2			—	√	—	10H

Notes 1. When the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, “F0000H” should be added to the value shown.

2. Not provided to the μPD784031.

Table 3-5 List of Special Function Registers (SFRs) (2/4)

Address [Note 1]	Special Function Register (SFR) Name	Symbol		R/W	Manipulable Bit Units			After Reset
					1 Bit	8 Bits	16 Bits	
0FF36H	Capture register (timer/counter 0)	CR02		R	—	—	√	0000H
0FF38H	Capture register L (timer/counter 1)	CR12	CR12W		—	√	√	
0FF39H	Capture register H (timer/counter 1)	—			—	—		
0FF3AH	Capture register L (timer/counter 2)	CR22	CR22W		—	√	√	
0FF3BH	Capture register H (timer/counter 2)	—			—	—		
0FF41H	Port 1 mode control register	PMC1		R/W	√	√	—	00H
0FF43H	Port 3 mode control register	PMC3			√	√	—	
0FF4EH	Pull-up resistor option register	PUO			√	√	—	
0FF50H	Timer register 0	TM0 [Note 3]		R	—	—	√	0000H
0FF51H					—	—		
0FF52H	Timer register 1	TM1 [Note 3]	TM1W [Note 3]		—	√	√	
0FF53H		—			—	—		
0FF54H	Timer register 2	TM2 [Note 3]	TM2W [Note 3]		—	√	√	
0FF55H		—			—	—		
0FF56H	Timer register 3	TM3 [Note 3]	TM3W [Note 3]		—	√	√	
0FF57H		—			—	—		
0FF5CH	Prescaler mode register 0	PRM0		R/W	—	√	—	11H
0FF5DH	Timer control register 0	TMC0			√	√	—	00H
0FF5EH	Prescaler mode register 1	PRM1W			—	√	—	11H
0FF5FH	Timer control register 1	TMC1			√	√	—	00H
0FF60H	D/A conversion value setting register 0	DACS0			—	√	—	
0FF61H	D/A conversion value setting register 1	DACS1			—	√	—	
0FF62H	D/A converter mode register	DAM			—	√	—	03H
0FF68H	A/D converter mode register	ADM			√	√	—	00H
0FF6AH	A/D conversion result register	ADCR		R	—	√	—	Undefined
0FF70H	PWM control register	PWMC		R/W	√	√	—	05H
0FF71H	PWM prescaler register	PWPR			—	√	—	00H
0FF72H	PWM modulo register 0	PWM0			—	—	√	Undefined
0FF74H	PWM modulo register 1	PWM1			—	—	√	
0FF7DH	One-shot pulse output control register	OSPC			√	√	—	00H
0FF80H	I^2C bus control register	IICC			√	√	—	
0FF81H	Prescaler mode register for serial clock	SPRM			—	√	—	04H
0FF82H	Clocked serial interface mode register	CSIM			√	√	—	00H
0FF83H [Note 2]	Slave address register	SVA		R/W [Note 4]	√ [Note 5]	√	—	01H

Notes
1. When the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, “F0000H” should be added to the value shown.
2. μPD784038Y Subseries only.
3. Use of TM0, TM1/TM1W, TM2/TM2W, and TM3/TM3W is limited. For details, refer to **(7)** in **3.10 CAUTIONS**.
4. Bit 0 is read-only.
5. Only bit 0 can be manipulated.

Table 3-5 List of Special Function Registers (SFRs) (3/4)

Address [Note 1]	Special Function Register (SFR) Name	Symbol	R/W	Manipulable Bit Units			After Reset
				1 Bit	8 Bits	16 Bits	
0FF84H	Clocked serial interface mode register 1	CSIM1	R/W	√	√	—	00H
0FF85H	Clocked serial interface mode register 2	CSIM2		√	√	—	
0FF86H	Serial shift register	SIO		—	√	—	Undefined
0FF88H	Asynchronous serial interface mode register	ASIM		√	√	—	00H
0FF89H	Asynchronous serial interface mode register 2	ASIM2		√	√	—	
0FF8AH	Asynchronous serial interface status register	ASIS	R	√	√	—	
0FF8BH	Asynchronous serial interface status register 2	ASIS2		√	√	—	
0FF8CH	Receive buffer: UART0	RXB		—	√	—	Undefined
	Transmit shift register: UART0	TXS	W	—	√	—	
	Shift register: IOE1	SIO1	R/W	—	√	—	
0FF8DH	Receive buffer: UART2	RXB2	R	—	√	—	Undefined
	Transmit shift register: UART2	TXS2	W	—	√	—	
	Shift register: IOE2	SIO2	R/W	—	√	—	
0FF90H	Baud rate generator control register	BRGC		—	√	—	00H
0FF91H	Baud rate generator control register 2	BRGC2		—	√	—	
0FFA0H	External interrupt mode register 0	INTM0		√	√	—	
0FFA1H	External interrupt mode register 1	INTM1		√	√	—	
0FFA4H	Sampling clock selection register	SCS0		—	√	—	
0FFA8H	In-service priority register	ISPR	R	√	√	—	
0FFAAH	Interrupt mode control register	IMC	R/W	√	√	—	80H
0FFACH	Interrupt mask register 0L	MK0L (MK0)		√	√	√	FFFFH
0FFADH	Interrupt mask register 0H	MK0H		√	√		
0FFAEH	Interrupt mask register 1L	MK1L		√	√	—	FFH
0FFC0H	Standby control register	STBC		—	√ [Note 2]	—	30H
0FFC2H	Watchdog timer mode register	WDM		—	√ [Note 2]	—	00H
0FFC4H	Memory extension mode register	MM		√	√	—	20H
0FFC5H	Hold mode register	HLDM		√	√	—	00H
0FFC6H	Clock output mode register	CLOM		√	√	—	
0FFC7H	Programmable wait control register 1	PWC1		—	√	—	AAH
0FFC8H	Programmable wait control register 2	PWC2		—	—	√	AAAAH
0FFCCH	Refresh mode register	RFM		√	√	—	00H

Notes
1. When the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, “F0000H” should be added to the value shown.
2. The write operation is possible by using the dedicated instruction “MOV STBC, #byte” or “MOV WDM, #byte” only. Instructions other than these cannot perform the write operation.

Table 3-5 List of Special Function Registers (SFRs) (4/4)

Address [Note 1]	Special Function Register (SFR) Name	Symbol	R/W	Manipulable Bit Units			After Reset
				1 Bit	8 Bits	16 Bits	
0FFCDH	Refresh area specification register	RFA	R/W	√	√	—	00H
0FFCEH	Oscillation stabilization time specification register	OSTS		—	√	—	
0FFD0H to	External SFR area 0FFDFH	—		√	√	—	—
0FFE0H	Interrupt control register (INTP0)	PIC0		√	√	—	43H
0FFE1H	Interrupt control register (INTP1)	PIC1		√	√	—	
0FFE2H	Interrupt control register (INTP2)	PIC2		√	√	—	
0FFE3H	Interrupt control register (INTP3)	PIC3		√	√	—	
0FFE4H	Interrupt control register (INTC00)	CIC00		√	√	—	43H
0FFE5H	Interrupt control register (INTC01)	CIC01		√	√	—	
0FFE6H	Interrupt control register (INTC10)	CIC10		√	√	—	
0FFE7H	Interrupt control register (INTC11)	CIC11		√	√	—	
0FFE8H	Interrupt control register (INTC20)	CIC20		√	√	—	
0FFE9H	Interrupt control register (INTC21)	CIC21		√	√	—	
0FFEAH	Interrupt control register (INTC30)	CIC30		√	√	—	
0FFEBH	Interrupt control register (INTP4)	PIC4		√	√	—	
0FFECH	Interrupt control register (INTP5)	PIC5		√	√	—	
0FFEDH	Interrupt control register (INTAD)	ADIC		√	√	—	
0FFEEH	Interrupt control register (INTSER)	SERIC		√	√	—	
0FFEFH	Interrupt control register (INTSR)	SRIC		√	√	—	
	Interrupt control register (INTCSI1)	CSIIC1		√	√	—	
0FFF0H	Interrupt control register (INTST)	STIC		√	√	—	
0FFF1H	Interrupt control register (INTCSI)	CSIIC		√	√	—	
0FFF2H	Interrupt control register (INTSER2)	SERIC2		√	√	—	
0FFF3H	Interrupt control register (INTSR2)	SRIC2		√	√	—	
	Interrupt control register (INTCSI2)	CSIIC2		√	√	—	
0FFF4H	Interrupt control register (INTST2)	STIC2		√	√	—	
0FFF5H [Note 2]	Interrupt control register (INTSPC)	SPCIC		√	√	—	
0FFFCH	Internal memory size switching register [Note 3]	IMS		—	√	—	FFH

Notes 1. When the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, “F0000H” should be added to the value shown.

2. μPD784038Y Subseries only.

3. Writing to this register is valid only when the μPD78P4038 is used.

3.10 CAUTIONS

(1) Program fetches cannot be performed from the internal high-speed RAM area (0FD00H to 0FEFFH when the LOCATION 0H instruction is executed; FFD00H to FFEFFH when the LOCATION 0FH instruction is executed).

(2) Special function registers (SFRs)
Addresses onto which SFRs are not assigned should not be accessed in the area 0FF00H to 0FFFFH [Note]. If such an address is accessed by mistake, the μPD784038 may become deadlocked. A deadlock can only be released by reset input.

Note When the LOCATION 0H instruction is executed; FFF00H to FFFFFH when the LOCATION 0FH instruction is executed.

(3) Stack pointer (SP) operation
With stack addressing, the entire 1-Mbyte space can be accessed, but a stack area cannot be reserved in the SFR area or internal ROM area.

(4) Stack pointer (SP) initialization
The SP is undefined after $\overline{\text{RESET}}$ input, while non-maskable interrupts can be acknowledged directly after reset release. Therefore, an unforeseen operation may be performed if a non-maskable interrupt request is generated while the SP is in the undefined state directly after reset release. To minimize this risk, the following program should be coded without fail after reset release.

```
RSTVCT      CSEG   AT     0
            DW     RSTSTRT
             to
INITSEG     CSEG   BASE
RSTSTRT :   LOCATION 0H          ; or LOCATION 0FH
            MOVG SP, #STKBGN
```

(5) The internal memory size switching register (IMS) that selects the internal memory size of the μPD78P4038 cannot be completely emulated by the in-circuit emulator and has the following restrictions. To debug products other than the μPD784038, select a mask version that performs debugging as the emulation CPU.
For the selection of an emulation CPU to the μPD78P4038, even if a write instruction other than FFH (EEH, DCH, CCH) to IMS is executed the memory size (FFH) is always identical to the μPD784038.

(6) Do not set external wait to the internal ROM area. Otherwise, the CPU may be in the deadlock status which can be cleared only by reset input.

(7) If the value of the timer register is read under the condition indicated by "×" in Table 3-6, the read value may be illegal. Do not read the timer register under condition "×".

Table 3-6 Limits of Reading Timer Register

(√: Can be read, ×: Must not be read)

f_{CLK} / Timer Count Clock	$f_{XX}/2$	$f_{XX}/4$	$f_{XX}/8$	$f_{XX}/16$
$f_{XX}/8$	√	√	×	×
$f_{XX}/16$	√	√	√	×
f_{XX}/n	√	√	√	√

Remarks **1.** f_{XX}: Oscillation frequency

2. f_{CLK}: Internal system clock frequency

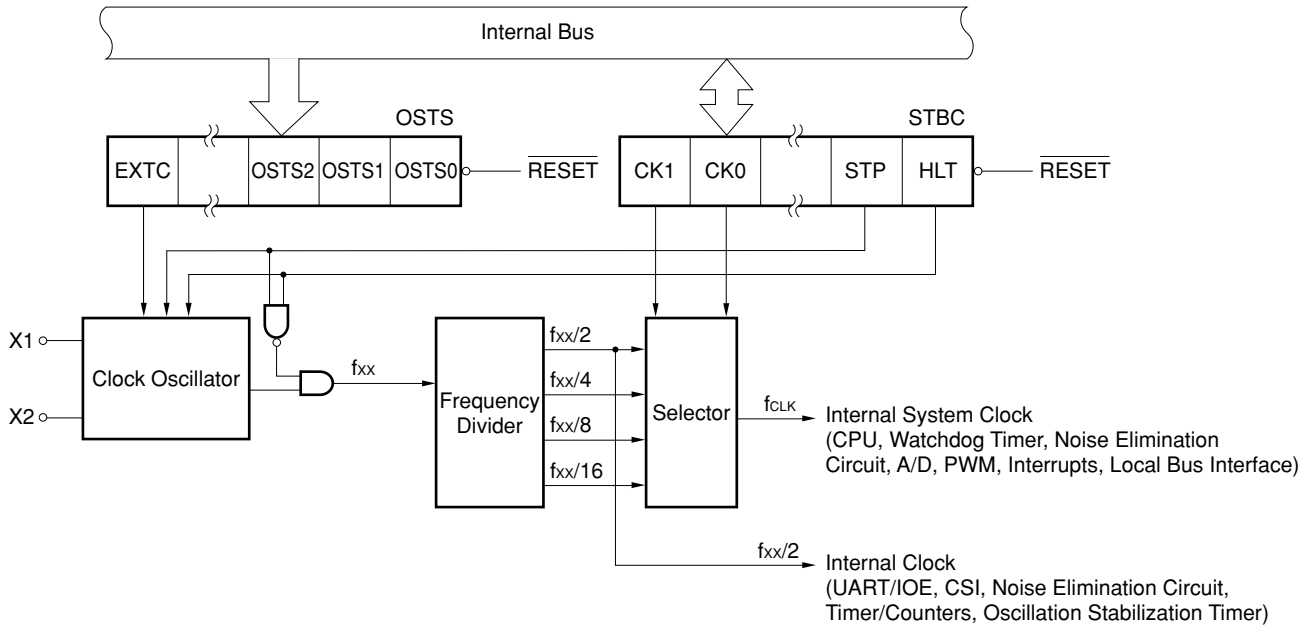
3. n = 32, 64, 128, 256, 512, 1,024, 2,048

CHAPTER 4 CLOCK GENERATOR

4.1 CONFIGURATION AND FUNCTION

The clock generator generates and controls the internal clock and internal system clock supplied to the CPU and on-chip hardware. The clock generator block diagram is shown in Figure 4-1.

Figure 4-1 Clock Generator Block Diagram



Remark f_{XX} : Crystal/ceramic oscillation frequency or internal clock frequency
f_{CLK} : Internal system clock frequency

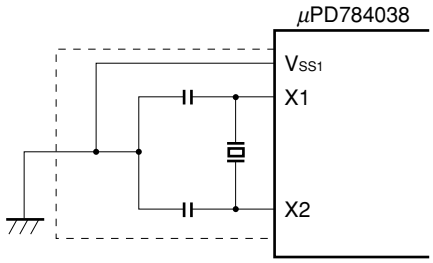
The clock oscillator oscillates by means of a crystal resonator/ceramic resonator connected to the X1 and X2 pins. When standby mode (STOP) is set, oscillation stops (see **CHAPTER 24 STANDBY FUNCTION**).

It is also possible to input an external clock. In this case, the clock signal is input to the X1 pin, and the inverse phase signal to the X2 pin.

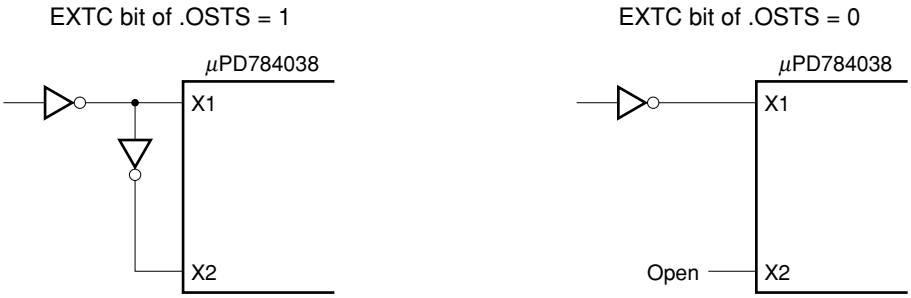
The frequency divider generates an internal system clock by 1/2, 1/4, 1/8, or 1/16 scaling of the clock oscillator output (f_{XX}) according to the setting of the standby control register (STBC).

Figure 4-2 Clock Oscillator External Circuitry

(a) Crystal/ceramic resonator oscillation



(b) External clock



Cautions 1. The oscillator should be as close as possible to the X1 and X2 pins.

2. No other signal lines should pass through the area enclosed by the dotted line.

Remark Differences between crystal resonator and ceramic resonator

Generally speaking, the oscillation frequency of a crystal resonator is extremely stable. It is therefore ideal for performing high-precision time management (in clocks, frequency meters, etc.).

A ceramic resonator is inferior to a crystal resonator in terms of oscillation frequency stability, but it has three advantages: a fast oscillation start-up time, small size, and low price. It is therefore suitable for general use (when high-precision time management is not required). In addition, there are products with a built-in capacitor, etc., which enable the number of parts and mounting area to be reduced.

4.2 CONTROL REGISTERS

4.2.1 Standby Control Register (STBC)

STBC is a register used to set the standby mode and select the internal system clock. See **Chapter 24 Standby Function** for details of the standby modes.

To prevent erroneous entry into standby mode due to an inadvertent program loop, the STBC register can only be written to by a dedicated instruction. This instruction is the MOV STBC, #byte instruction, and has a special code configuration (4 bytes). A write is only performed if the 3rd and 4th bytes of the op code are mutual complements. If the 3rd and 4th bytes of the op code are not mutual complements, a write is not performed, and an op error interrupt is generated. In this case, the return address saved in the stack area is the address of the instruction which is the source of the error. The error source address can thus be found from the return address saved on the stack area.

An endless loop will result if restore from an operand error is simply performed with an RETB instruction.

Because the operand error interrupt occurs only when the program hangs up (only the correct dedicated instruction is generated with the NEC Electronics assembler RA78K4 when MOV STBC, #byte is described), make sure that the operand error interrupt processing program initializes the system.

Other write instructions (“MOV STBC, A”, “AND STBC, #byte”, “SET1 STBC.7”, etc.) are ignored, and no operation is performed. That is, a write is not performed on the STBC, and an interrupt such as an operand error interrupt is not generated. The STBC can be read at any time with a data transfer instruction.

STBC is set by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets the STBC register contents to 30H.

The format of the STBC is shown in Figure 4-3.

Figure 4-3 Standby Control Register (STBC) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
STBC	0	0	CK1	CK0	×	0	STP	HLT	0FFC0H	30H	R/W

STP	HLT	Operating Mode
0	0	Normal mode
0	1	HALT mode
1	0	STOP mode
1	1	IDLE mode

CK1	CK0	Intermal System Clock Selection
0	0	f_{XX}/2 (16 MHz)
0	1	f_{XX}/4 (8 MHz)
1	0	f_{XX}/8 (4 MHz)
1	1	f_{XX}/16 (2 MHz)

Caution If the STOP mode is used when external clock input is used, the EXTC bit of the oscillation stabilization time specification register (OSTS) must be set (to 1) before setting the STOP mode. If the STOP mode is used when the EXTC bit of the OSTS is in the cleared (to 0) state when external clock input is used, the μPD784038 may be damaged or suffer reduced reliability.
When setting the EXTC bit to 1, be sure to input a clock in phase reverse to that of the clock input to the X1 pin, to the X2 pin.

4.2.2 Oscillation Stabilization Time Specification Register (OSTS)

OSTS is a register used to specify the operation of the oscillator. The EXTC bit of the OSTS specifies whether a crystal/ceramic resonator or an external clock is used. The STOP mode can be set during use of external clock input, only when the EXTC bit is set (to 1).

The OSTS can be written to only by an 8-bit transfer instruction.

$\overline{\text{RESET}}$ input clears the OSTS register contents to 00H.

The format of the OSTS is shown in Figure 4-4.

Figure 4-4 Oscillation Stabilization Time Specification Register (OSTS) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
OSTS	EXTC	0	0	0	0	OSTS2	OSTS1	OSTS0	0FFCFH	00H	R/W

Oscillation Stabilization Time Selection
(See **Figure 24-4** for details)

EXTC	External Clock Selection
0	Crystal/ceramic oscillation used External clock (drives X1 only)
1	External clock (drives X1 and X2)

Cautions 1. When using a crystal/ceramic oscillation, the EXTC bit must be cleared (to 0). If the EXTC bit is set (to 1), oscillation will stop.

2. If the STOP mode is used with external clock input, the EXTC bit must be set (to 1) before setting the STOP mode. If the STOP mode is used when the EXTC bit is in the cleared (to 0) state, the μPD784038 may be damaged or suffer reduced reliability.

3. When setting the EXTC bit to 1 during external clock input, be sure to input a clock in phase reverse to that of the clock input to the X1 pin, to the X2 pin. When the EXTC bit is set to 1, the μPD784038 operates on only the clock input to the X2 pin.

4.3 CLOCK GENERATOR OPERATION

4.3.1 Clock Oscillator

(1) When using crystal/ceramic oscillation

The clock oscillation circuit starts oscillating when the $\overline{\text{RESET}}$ signal is input, and stops oscillation when the STOP mode is set by the standby control register (STBC). Oscillation is resumed when the STOP mode is released.

(2) When using external clock

The clock oscillation circuits supplies the clock input from the X1 pin to the internal circuitry when the $\overline{\text{RESET}}$ signal is input. The oscillation circuit operates as follows when the EXTC bit of the oscillation stabilization time specification register (OSTS) is set to 1.

- The clock oscillation circuit is set in the external clock input mode.
- The clock oscillation circuit supplies the clock input to the X2 pin to the internal circuitry.
- The necessary circuit stops operating during the crystal/ceramic oscillation of the clock oscillation circuit, to reduce the power dissipation.
- The STOP mode can be used even when the external clock is input.
- The oscillation stabilization time is shortened when the system is released from the STOP mode.

Cautions 1. When using a crystal/ceramic oscillation, the EXTC bit of the Oscillation stabilization time specification register (OSTS) must be cleared (to 0). If the EXTC bit is set (to 1), oscillation will stop.

2. If the STOP mode is used with external clock input, the EXTC bit of the OSTS must be set (to 1) before setting the STOP mode. If the STOP mode is used when the EXTC bit is in the cleared (to 0) state, not only will the clock generator consumption current not be reduced, but the μPD784038 may also be damaged or suffer reduced reliability.

3. When setting the EXTC bit of OSTS to 1, be sure to input a clock in phase reverse to that of the clock input to the X1 pin, to the X2 pin.

4.3.2 Divider

The divider performs 1/2, 1/4, 1/8, or 1/16 scaling of the clock oscillator output, and supplies the resulting clock to the CPU, watchdog timer, noise elimination circuit, clocked serial interface (CSI), A/D converter, PWM, interrupt control circuit, and local bus interface. The division ratio is specified by the CK0 and CK1 bits of the standby control register (STBC).

Controlling the division ratio to match the speed required by the CPU enables the overall power consumption to be reduced. Also, the operating speed can be selected to match the supply voltage.

When $\overline{\text{RESET}}$ is input, the lowest speed (1/16) is selected.

If the division ratio of the divider circuit is changed, the maximum time shown in Table 4-1 is required to change the division ratio, depending on the clock selected before change.

Instruction execution continues even while the division ratio is changed, and the clock is supplied with the previous division ratio until the division ratio has been completely changed.

Table 4-1 Time Required to Change Division Ratio

Previous Division Ratio	Maximum Time Required for Change
1/2	22/f_{XX}
1/4	24/f_{XX}
1/8	16/f_{XX}
1/16	16/f_{XX}

4.4 CAUTIONS

The following cautions apply to the clock generator.

4.4.1 When an External Clock is Input

(1) If the STOP mode is used with external clock input, the EXTC bit of the oscillation stabilization time specification register (OSTS) must be set (to 1). If the STOP mode is used when the EXTC bit is in the cleared (to 0) state, the μPD784038 may be damaged or suffer reduced reliability.

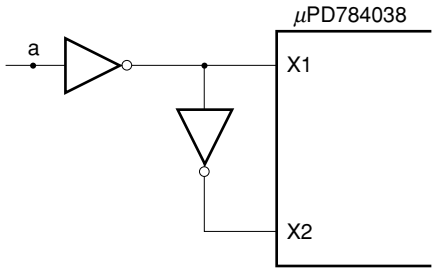
(2) When setting the EXTC bit of the OSTS to 1, be sure to input a clock in phase reverse to that of the clock input to the X1 pin, to the X2 pin.

(3) Even when inputting the external clock by clearing the EXTC bit of the oscillation stabilization time specification register (OSTS) to 0, input a signal in phase reverse to that of the signal input to the X1 pin, to the X2 pin, whenever possible. Otherwise, more malfunctioning may occur due to noise.

(4) When an external clock is input, this should be performed with a HCMOS device, or a device with the equivalent drive capability.

(5) A signal should not be extracted from the X1 and X2 pins. If a signal is extracted, it should be extracted from point a in Figure 4-5.

Figure 4-5 Signal Extraction with External Clock Input



(6) The wiring connecting the X1 pin to the X2 pin via an inverter, in particular, should be made as short as possible.

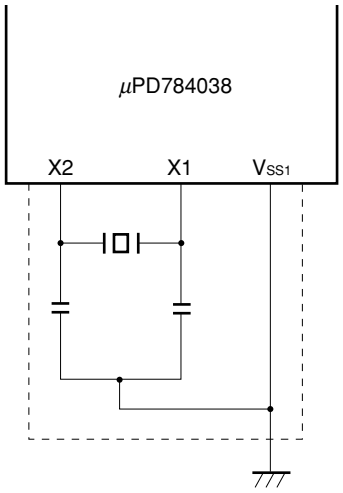
4.4.2 When Crystal/Ceramic Oscillation is Used

(1) As the oscillator is a high-frequency analog circuit, considerable care is required.
The following points, in particular, require attention.

- The wiring should be kept as short as possible.
- No other signal lines should be crossed.
- Avoid lines carrying a high fluctuating current.
- The oscillator capacitor grounding point should always be at the same potential as the V_{SS1} pin. Do not ground to a ground pattern carrying a high current.
- A signal should not be taken from the oscillator.

If oscillation is not performed normally and stably, the microcontroller will not be able to operate normally and stably, either. Also, if a high-precision oscillation frequency is required, consultation with the oscillator manufacturer is recommended.

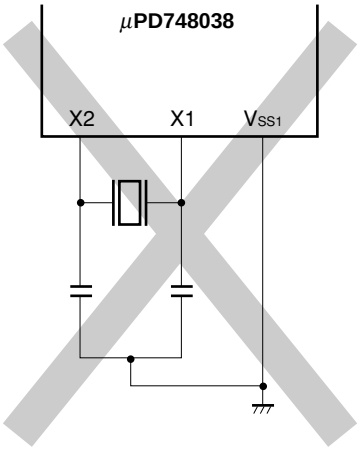
Figure 4-6 Cautions on Resonator Connection



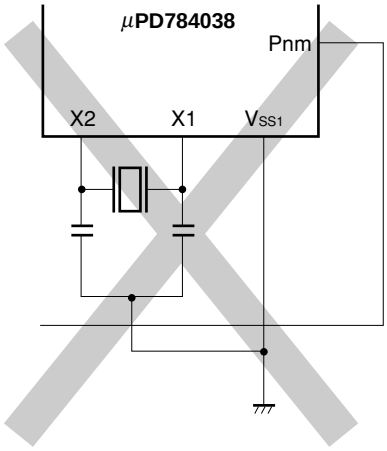
Cautions 1. The oscillator should be as close as possible to the X1 and X2 pins.
2. No other signal lines should pass through the area enclosed by the dotted line.

Figure 4-7 Incorrect Example of Resonator Connection

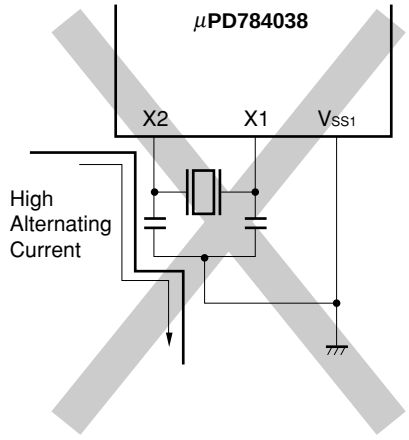
(a) Wiring of connected circuits is too long



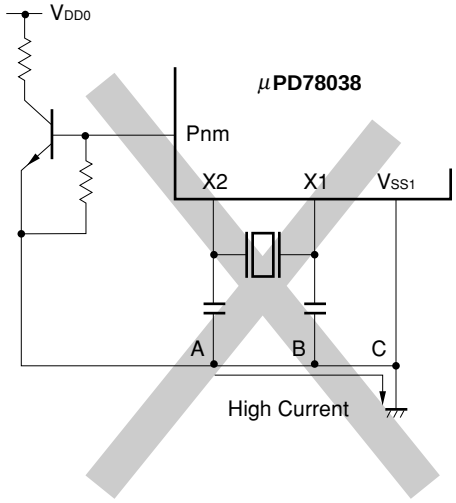
(b) Crossed signal lines



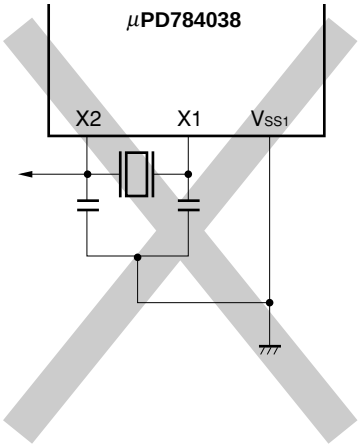
(c) Wiring near high alternating current



(d) Current flowing through ground line of oscillation circuit (Potentials at points A, B, and C fluctuate)



(e) Signal extracted



(2) When the device is powered on, and when restoring from the STOP mode, sufficient time must be allowed for the oscillation to stabilize. Generally speaking, the time required for oscillation stabilization is several milliseconds when a crystal resonator is used, and several hundred microseconds when a ceramic resonator is used.
An adequate oscillation stabilization period should be secured by the following means:

<1> When powering-on : $\overline{\text{RESET}}$ input (reset period)
<2> When returning from STOP mode :
(i) $\overline{\text{RESET}}$ input (reset period)
(ii) Time of the oscillation stabilization timer that automatically starts at the valid edge of NMI, INTP4, or INTP5 signal **Note** (set by the oscillation stabilization time specification register (OSTS))

Note For INTP4 and INTP5, when masking is released and macro service is disabled.

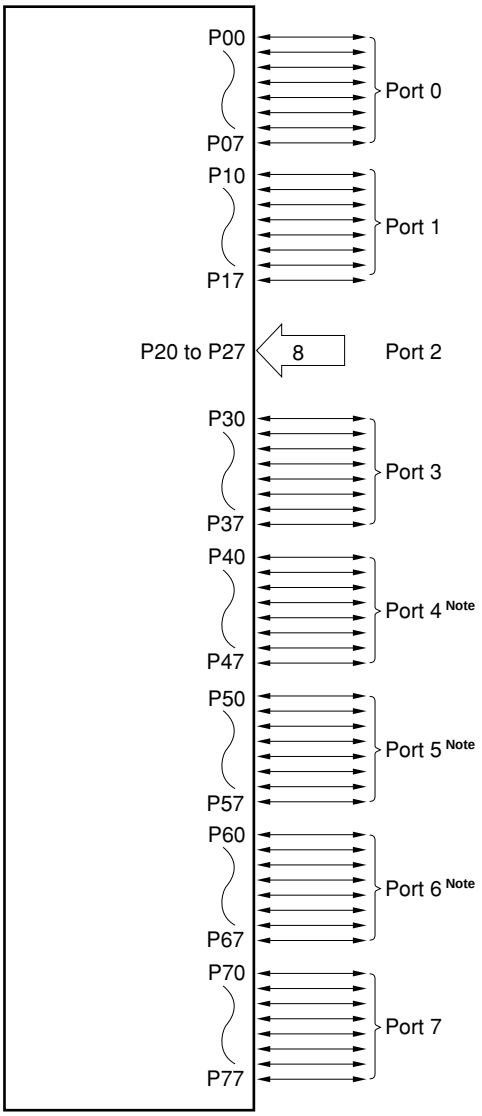
(3) The EXTC bit of the oscillation stabilization time specification register (OSTS) must be cleared (to 0). If the EXTC bit is set (to 1), oscillation will stop.

CHAPTER 5 PORT FUNCTIONS

5.1 DIGITAL INPUT/OUTPUT PORTS

The μPD784038 is provided with the ports shown in Figure 5-1, enabling various kinds of control to be performed. The function of each port is shown in Table 5-1. For ports 0 to 6, use of an internal pull-up resistor can be specified by software when used as input ports.

Figure 5-1 Port Configuration



Note With the μPD784031, P40 to P47 serve as address/data bus pins, P50 to P57, as address bus pins, P64, as $\overline{RD}$ pin, and P65, as $\overline{WR}$ pin.
P60 to P63 serve as output port pins.

Table 5-1 Port Functions

Port Name	Pin Names	Functions	Software Pull-up Specification
Port 0	P00 to P07	• Input or output specifiable bit-wise. • Can also operate as 4-bit real-time output ports (P00 to P03, P04 to P07). • Transistor drive capability.	Input mode pins specified at once
Port 1	P10 to P17	• Input or output specifiable bit-wise. • LED drive capability.	
Port 2	P20 to P27	Input port	6-bit unit (P22 to P27)
Port 3	P30 to P37	Input or output specifiable bit-wise.	Input mode pins specified at once
Port 4	P40 to P47 Note	• Input or output specifiable bit-wise. • LED drive capability.	
Port 5	P50 to P57 Note	• Input or output specifiable bit-wise. • LED drive capability.	
Port 6	P60 to P67 Note	Input or output specifiable bit-wise.	
Port 7	P70 to P77	Input or output specifiable bit-wise.	—

Note With the μPD784031, P40 to P47 serve as address/data bus pins, P50 to P57, as address bus pins, P64, as $\overline{RD}$ pin, and P65, as $\overline{WR}$ pin. These pins therefore cannot directly drive LEDs or be connected to a pull-up resistor by software. P60 to P63 serve as output port pins.

Table 5-2 Number of Input/Output Ports

Input/Output / Ports	Total	Input Mode	Output Mode	
		Software Pull-up Resistor	Direct LED Drive	Direct Transistor Drive
Input ports	8 (8)	6 (6)	—	—
Input/output ports	56 (34)	48 (26)	24 (8)	0 (0)
Output ports	0 (4)	—	0 (0)	8 (8)
Total	64 (46)	54 (32)	24 (8)	8 (8)

Remark ( ): μPD784031

5.2 PORT 0

Port 0 is an 8-bit input/output port with an output latch, and has direct transistor drive capability. Input/output can be specified bit-wise by means of the port 0 mode register (PM0). Each pin incorporates a software programmable pull-up resistor.

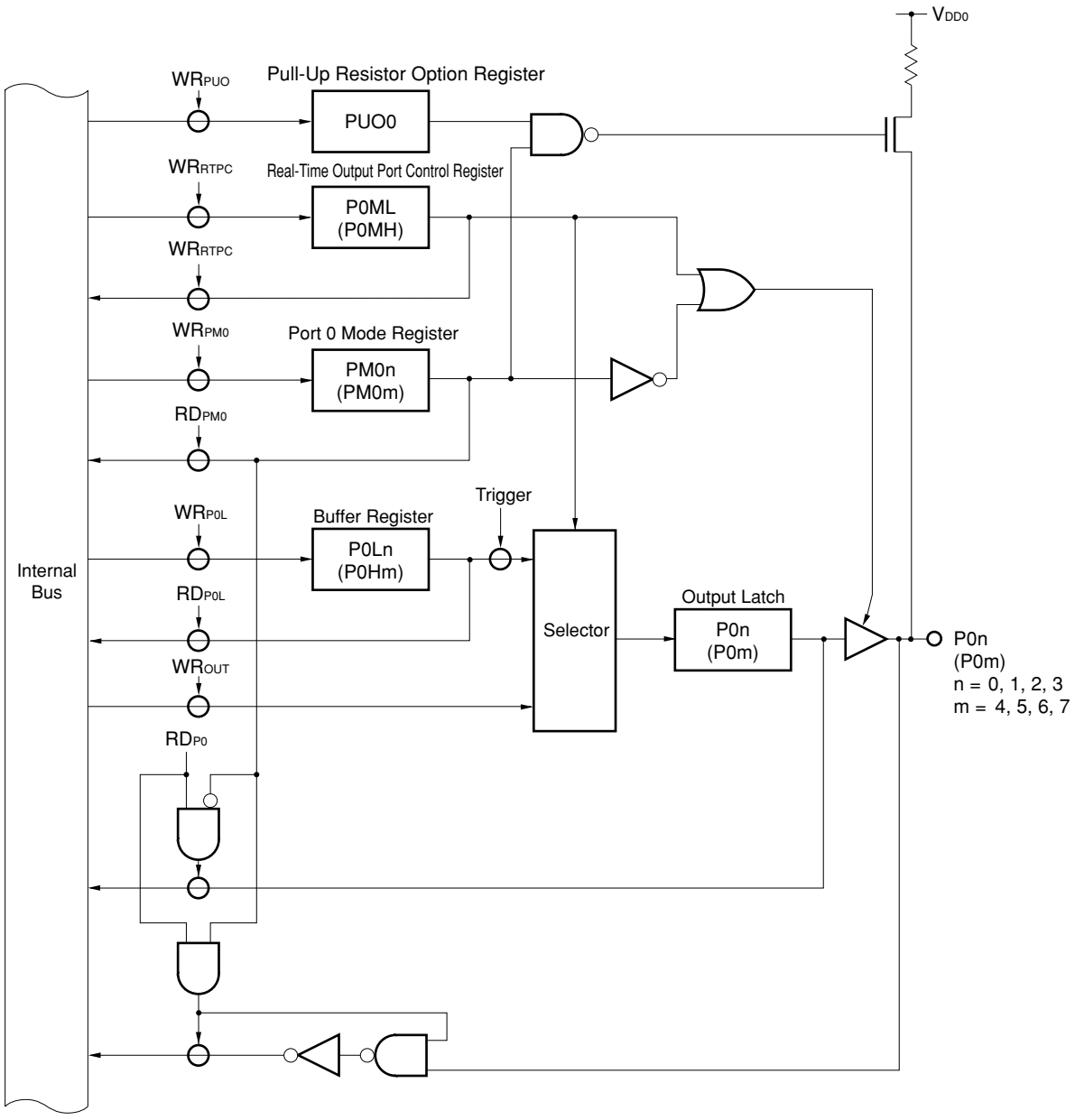
P00 to P03 and P04 to P07 can output the buffer register (P0L, P0H) contents at any time interval as 4-bit real-time output ports or one 8-bit real-time output port. The real-time output port control register (RTPC) is used to select whether this port is used as a normal output port or a real-time output port.

When $\overline{\text{RESET}}$ is input, port 0 is set as an input port (output high-impedance state), and the output latch contents are undefined.

5.2.1 Hardware Configuration

The port 0 hardware configuration is shown in Figure 5-2.

Figure 5-2 Port 0 Block Diagram



5.2.2 I/O Mode/Control Mode Setting

The port 0 input/output mode is set by means of the port 0 mode register (PM0) as shown in Figure 5-3.

Figure 5-3 Port 0 Mode Register (PM0) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	0FF20H	FFH	R/W

PM0n	P0n Pin Input/Output Mode Specification (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

When port 0 is used as a real-time output port, the P0ML and P0MH bits of the real-time output port control register (RTPC) should be set (to 1).

When P0ML and P0MH are set, the respective pin output buffer is turned on and the output latch contents are output to the pin irrespective of the contents of PM0.

5.2.3 Operating Status

Port 0 is an input/output port

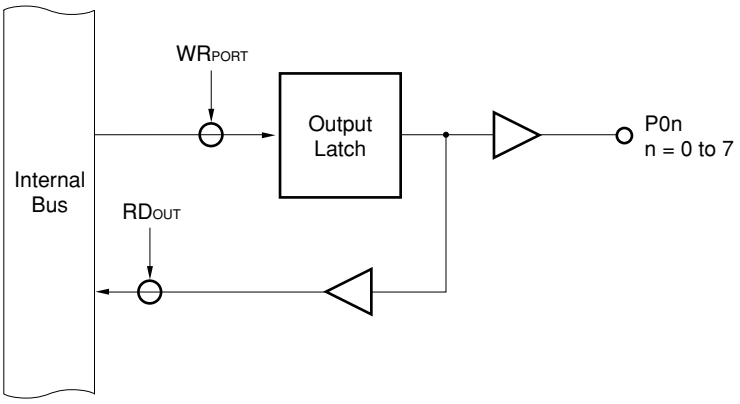
(1) When set as an output port

The output latch is enabled, and data transfers between the output latch and accumulator are performed by means of transfer instructions. The output latch contents can be freely set by means of logical operation instructions. Once data has been written to the output latch, it is retained until data is next written to the output latch **Note**.

Writes cannot be performed to the output latch of a port specified as a real-time output port. However, the output latch contents can be read even if it is set to the real-time output port mode.

Note Including the case where another bit of the same port is manipulated by a bit manipulation instruction.

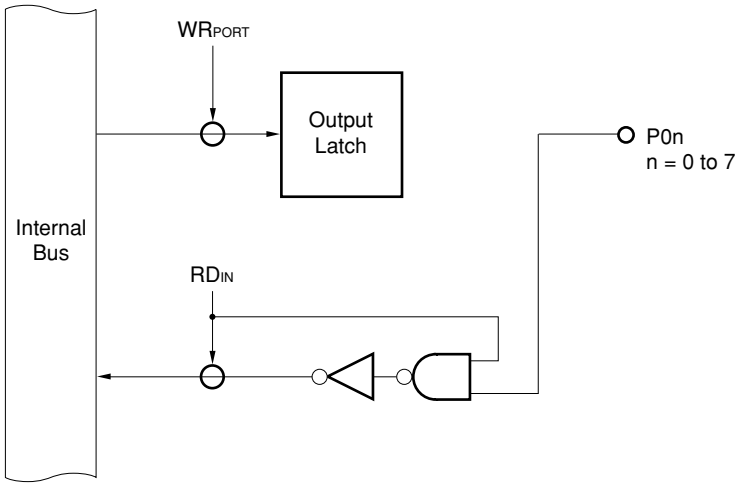
Figure 5-4 Port Specified as Output Port



(2) When set as an input port

The port pin level can be loaded into an accumulator by means of a transfer instruction, etc. In this case, too, writes can be performed to the output latch, and data transferred from the accumulator by a transfer instruction, etc., is stored in all output latches irrespective of the port input/output specification. However, since the output buffer of a bit specified as an input port is high-impedance, the data is not output to the port pin (when a bit specified as input is switched to an output port, the output latch contents are output to the port pin). Also, the contents of the output latch of a bit specified as an input port cannot be loaded into an accumulator.

Figure 5-5 Port Specified as Input Port



Caution A bit manipulation instruction manipulates one bit as the result, but accesses the port in 8-bit units. Therefore, if a bit manipulation instruction is used on a port with a mixture of input and output pins, the contents of the output latch of pins specified as inputs will be undefined (excluding bits manipulated with a SET1 or CLR1 instruction, etc.). Particular care is required when there are bits which are switched between input and output.

Caution is also required when manipulating the port with other 8-bit manipulation instructions.

5.2.4 Internal Pull-Up Resistors

Port 0 incorporates pull-up resistors. Use of these internal resistors when pull-up is necessary enables the number of parts and the mounting area to be reduced.

Whether or not an internal pull-up resistor is to be used can be specified for each pin by means of the PUO0 bit of the pull-up resistor option register (PUO) and the port 0 mode register (PM0). When PUO0 is 1, the internal pull-up resistors of the pins for which input is specified by PM0 are enabled (PM0n = 1, n = 0 to 7).

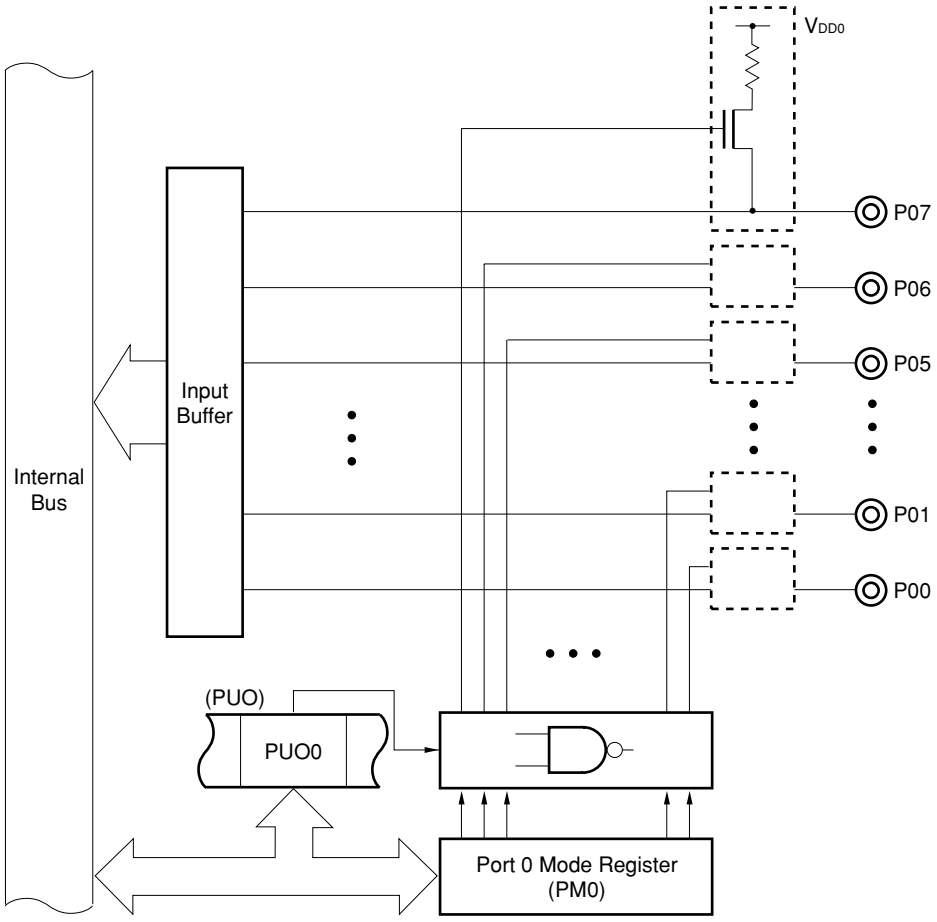
Figure 5-6 Pull-Up Resistor Option Register (PUO) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PUO	0	PUO6	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	0FF4EH	00H	R/W

PUO0	Port 0 Pull-Up Resistor Specification
0	Not used in port 0
1	Used in port 0

Remark When STOP mode is entered, setting 00H in PUO is effective in reducing the power consumption.

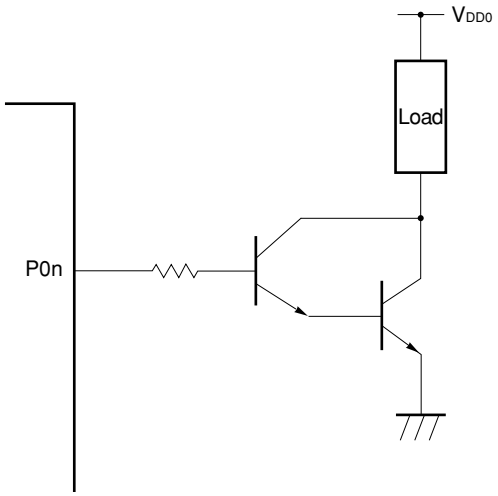
Figure 5-7 Pull-Up Resistor Specification (Port 0)



5.2.5 Transistor Drive

In port 0, the output buffer high-level side drive capability has been increased, allowing active-high direct transistor drive. An example of the connection is shown in Figure 5-8.

Figure 5-8 Example of Transistor Drive



5.3 PORT 1

Port 1 is an 8-bit input/output port with an output latch. Input/output can be specified bit-wise by means of the port 1 mode register (PM1). Each pin incorporates a programmable pull-up resistor. This port has direct LED drive capability.

In addition to their input/output port function, P10 to P14 also have an alternate function as PWM output pins and serial interface pins. The operating mode can be specified bit-wise by means of the PWM control register (PWMC) and the port 1 mode control register (PMC1), as shown in Table 5-3. The level of any pin can be read and tested at any time irrespective of the alternate-function pin operation.

When $\overline{\text{RESET}}$ is input, port 1 is set as an input port (output high-impedance state), and the output latch contents are undefined.

Table 5-3 Port 1 Operating Modes

Pin Name	Port Mode	Control Signal I/O Mode	Operation to Operate Control Pin
P10	I/O port	PWM0 output	Setting of EN0 bit of PWMC to 1
P11		PWM1 output	Setting of EN1 bit of PWMC to 1
P12		ASCK2 I/O/$\overline{\text{SCK2}}$ I/O	Setting PMC12 bit of PMC1 to 1
P13		RxD2 input/SI2 input	Setting PMC13 bit of PMC1 to 1
P14		TxD2 output/SO2 output	Setting PMC14 bit of PMC1 to 1
P15 to P17		—	—

(a) Port mode

P10 and P11 operate as port mode pins when the EN0 and EN1 bits of the PWM control (PWMC) register are cleared (to 0), and P12 to P14 do the same when the relevant bits of the port 1 mode control (PMC1) register are cleared (to 0), and P15 to P17 always operate as port mode pins. Input/output can be specified bit-wise by means of the port 1 mode register (PM1).

(b) Control signal input/output mode

P10 and P11 operate as PWM signal output pins when the EN0 and EN1 bits, respectively, of the PWM control (PWMC) register are set (to 1).

P12 to P14 can be set as control pins bit-wise by setting the port 1 mode control (PMC1) register.

(i) PWM0, PWM1

PWM0 and PWM1 are PWM output pins.

(ii) ASCK2/$\overline{\text{SCK2}}$

ASCK2 is the asynchronous serial interface baud rate clock input pin.

$\overline{\text{SCK2}}$ is the serial clock input/output pin (in 3-wire serial I/O2 mode).

(iii) RxD2/SI2

RxD2 is the asynchronous serial interface serial data input pin.

SI2 is the serial data input pin (in 3-wire serial I/O2 mode).

(iv) TxD2/SO2

TxD2 is the asynchronous serial interface serial data output pin.

SO2 is the serial data output pin (in 3-wire serial I/O2 mode).

5.3.1 Hardware Configuration

The port 1 hardware configuration is shown in Figures 5-9 to 5-13.

Figure 5-9 Block Diagram of P10 and P11 (Port 1)

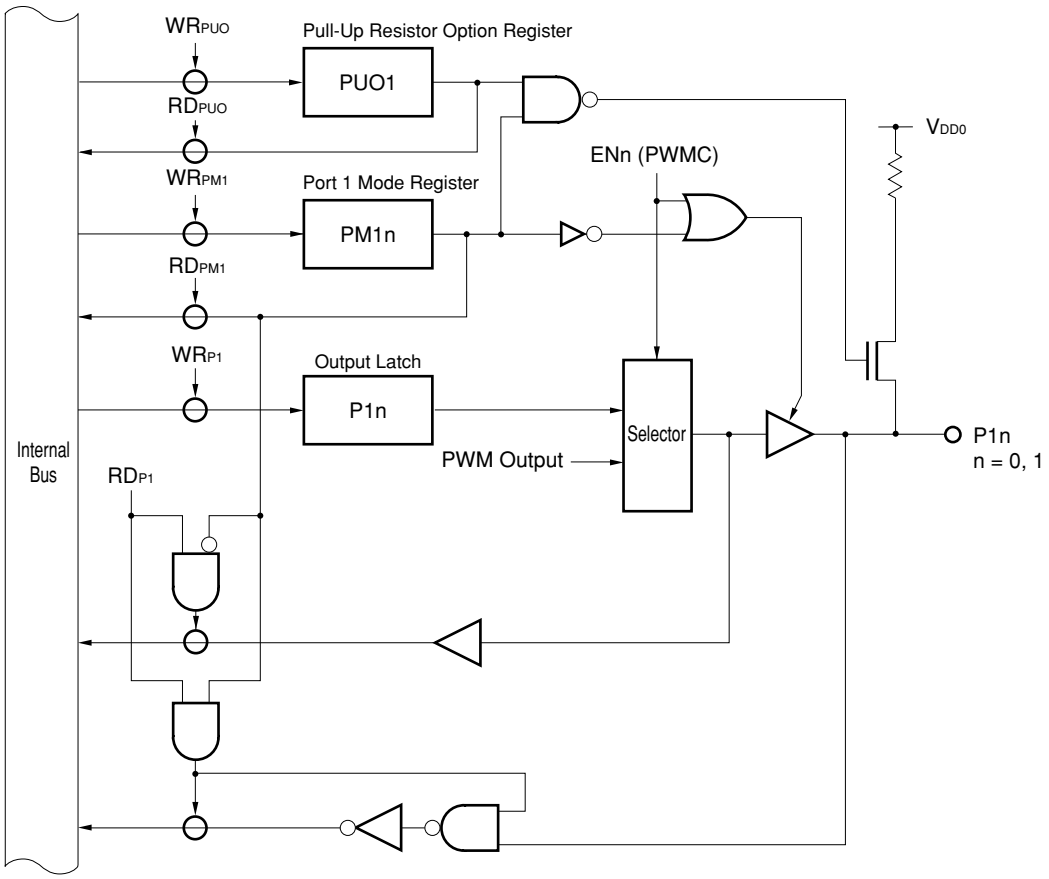


Figure 5-10 Block Diagram of P12 (Port 1)

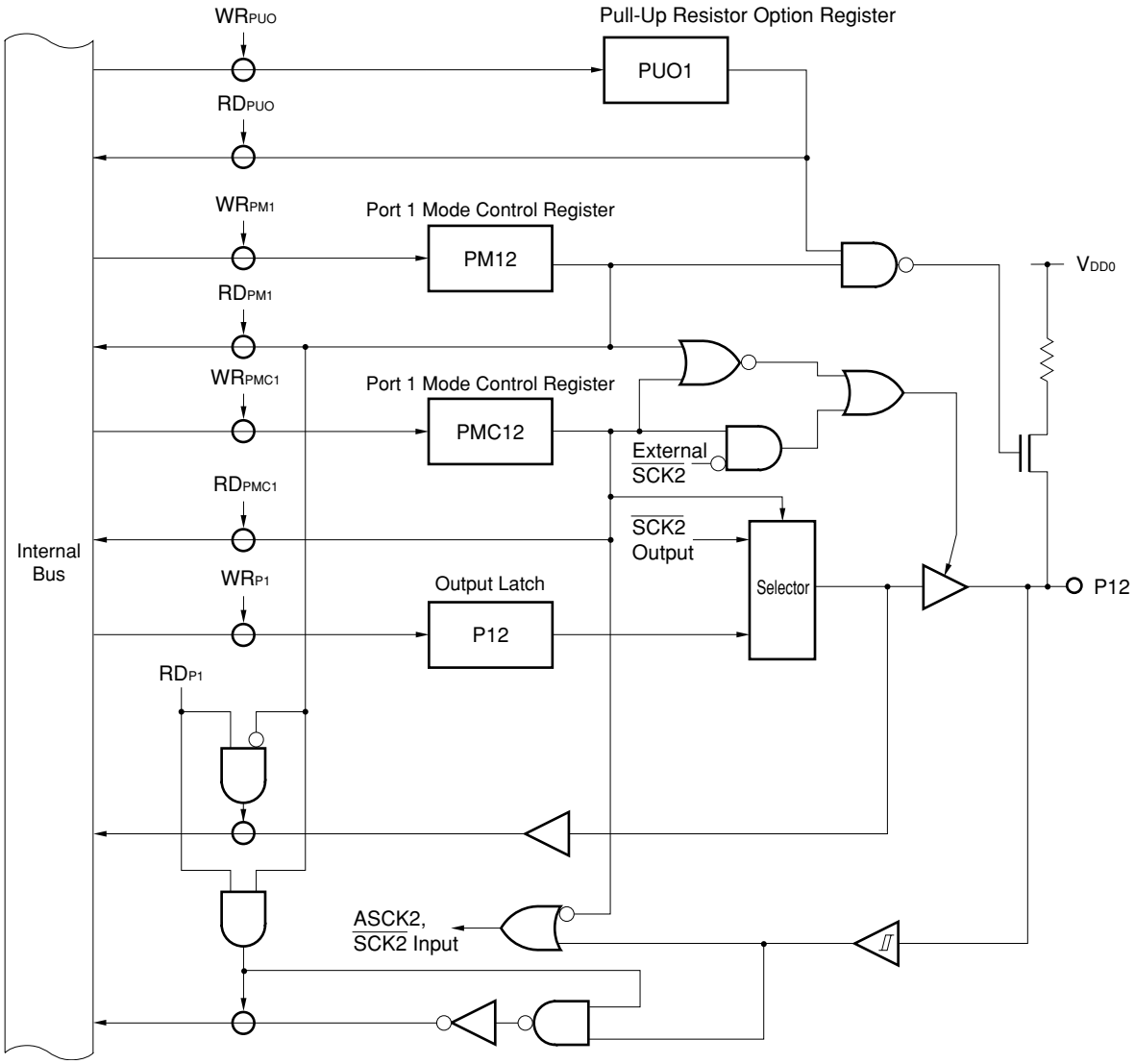


Figure 5-11 Block Diagram of P13 (Port 1)

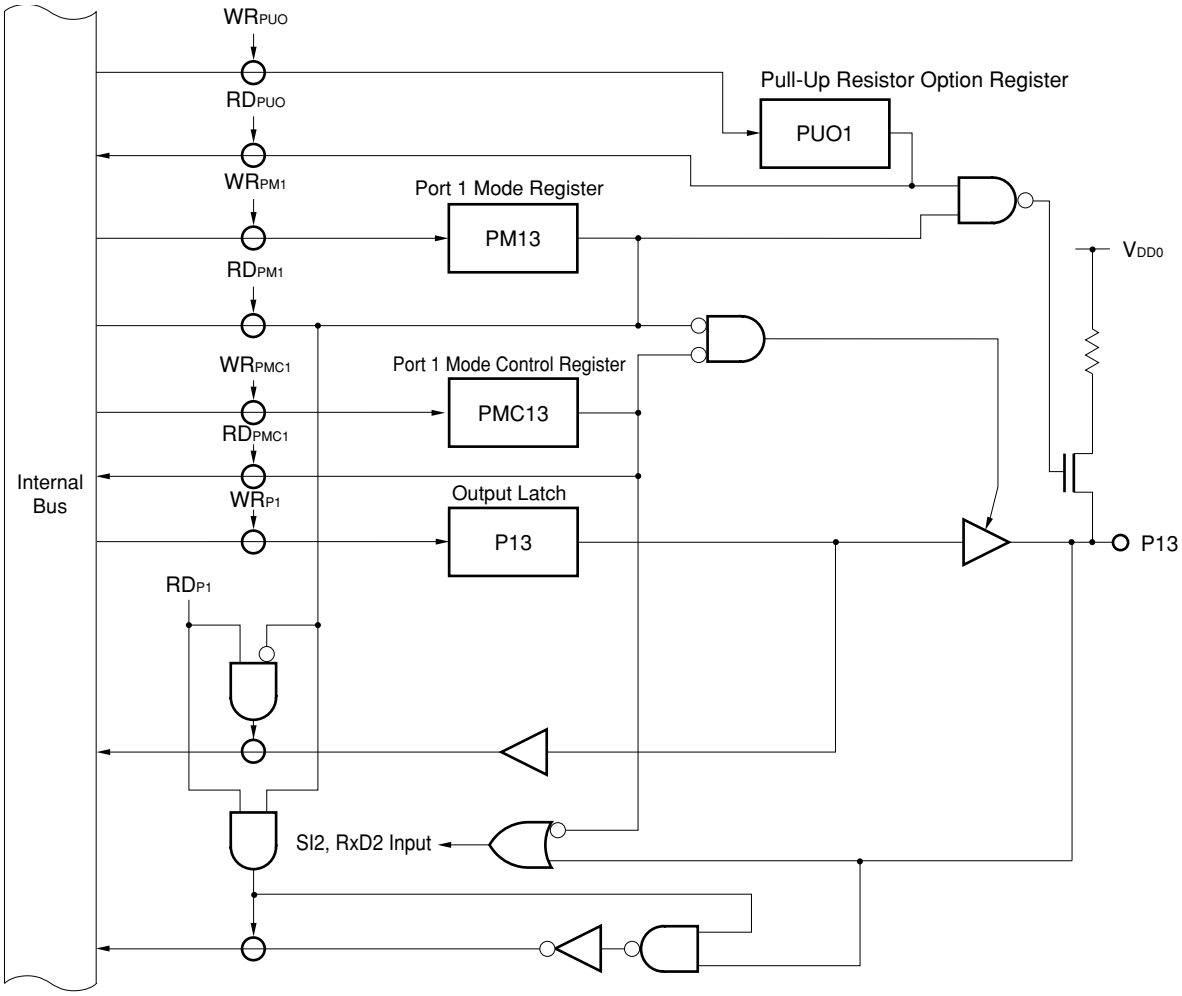


Figure 5-12 Block Diagram of P14 (Port 1)

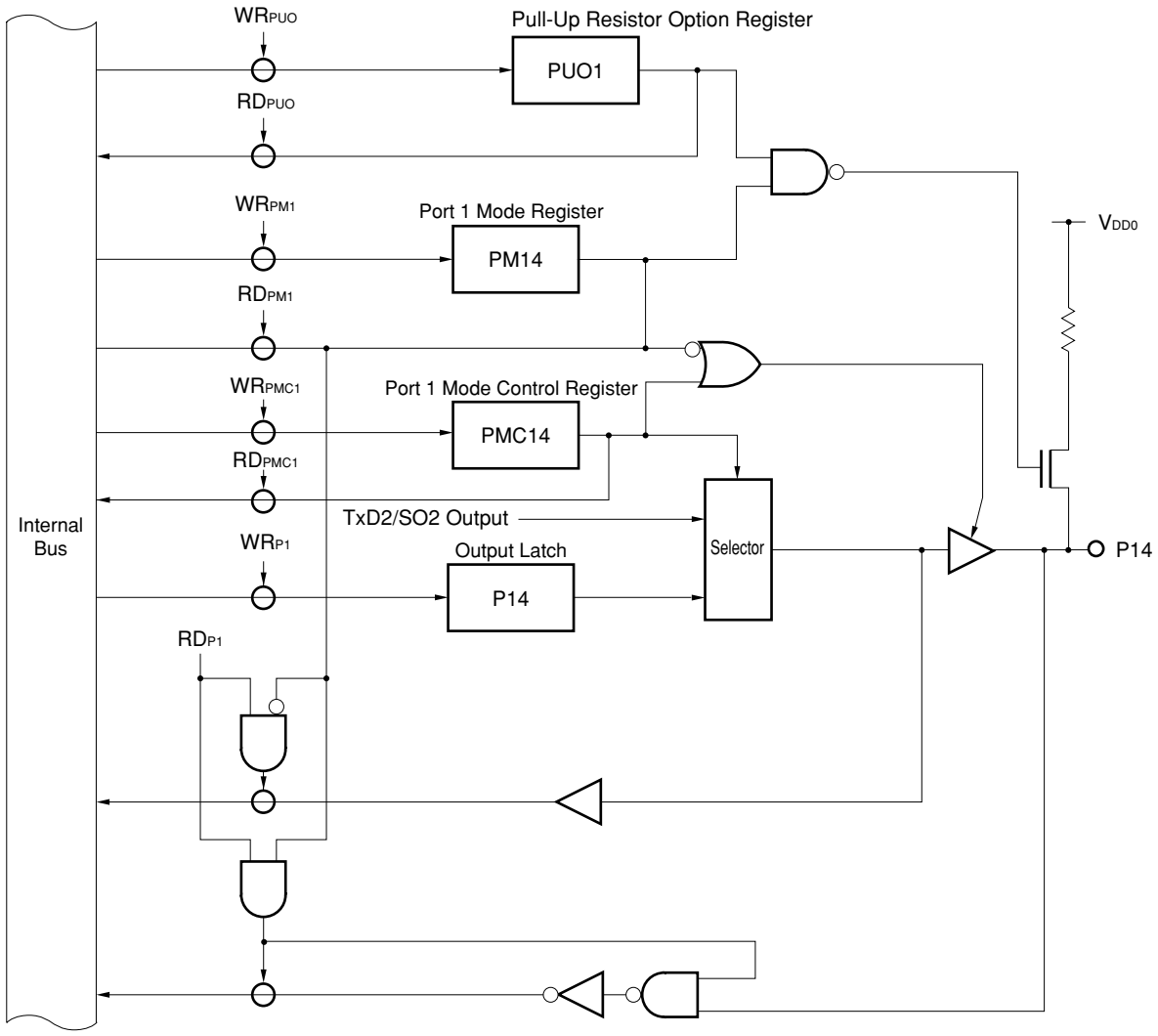
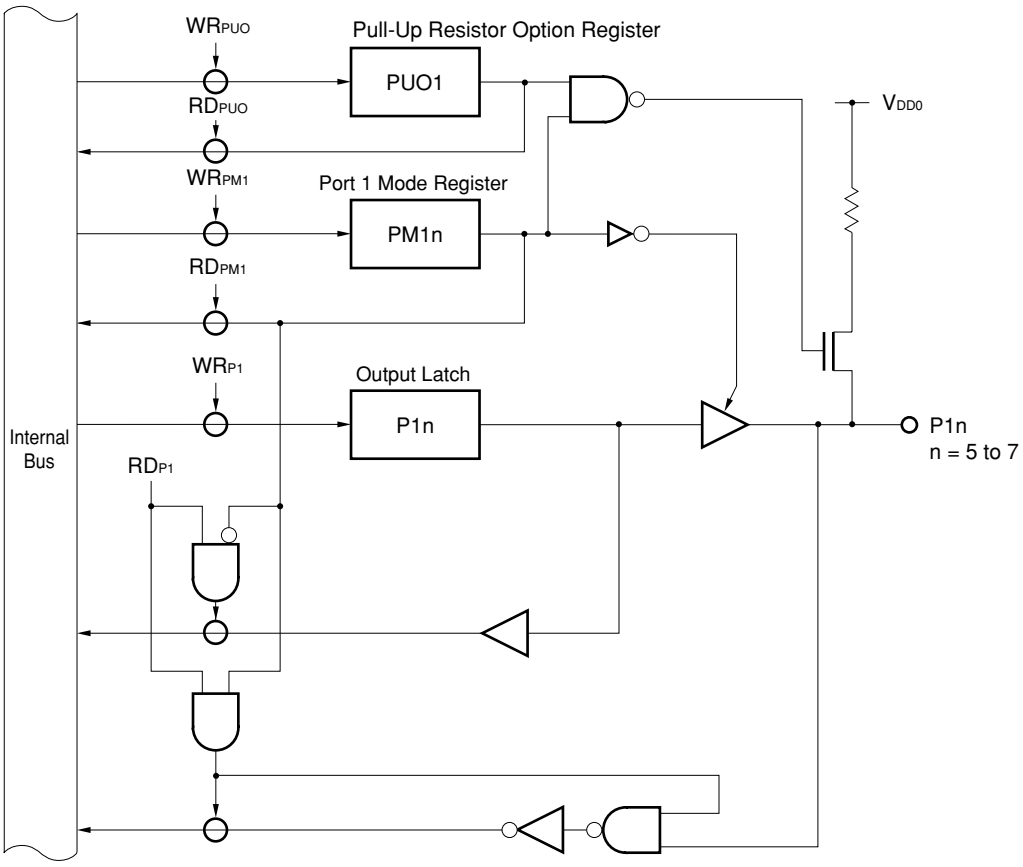


Figure 5-13 Block Diagram of P15 to P17 (Port 1)



5.3.2 I/O Mode/Control Mode Setting

The port 1 input/output mode is set for each pin by means of the port 1 mode register (PM1) as shown in Figure 5-14.

In addition to their input/output port function, P10 and P11 also have an alternate function as PWM signal output pins, and the control mode is specified by means of the PWM control register (PWMC) as shown in Table 5-4.

In addition to their input/output port function, P12 to P14 also have an alternate function as serial interface pins, and the control mode is specified by means of the port 1 mode control register (PMC1) as shown in Figure 5-15.

Figure 5-14 Port 1 Mode Register (PM1) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	0FF21H	FFH	R/W

PM1n	P1n Pin Input/Output Mode Specification (n = 0 to 7)
0	Output mode (output buffer on)
1	Intput mode (output buffer off)

Table 5-4 Method of Setting P10 & P11 PWM Signal Output Function

Pin Name	Function	Method of Setting PWM Signal Output Function
P10	PWM0	Set (to 1) EN0 bit of PWMC
P11	PWM1	Set (to 1) EN1 bit of PWMC

Figure 5-15 Port 1 Mode Control Register (PMC1) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PMC1	0	0	0	PMC14	PMC13	PMC12	0	0	0FF41H	00H	R/W

PMC12	P12 Pin Control Mode Specification
0	Input/output port mode
1	ASCK2/$\overline{\text{SCK2}}$ input/output mode

PMC13	P13 Pin Control Mode Specification
0	Input/output port mode
1	RxD2/SI2 input mode

PMC14	P14 Pin Control Mode Specification
0	Input/output port mode
1	TxD2/SO2 output mode

5.3.3 Operating Status

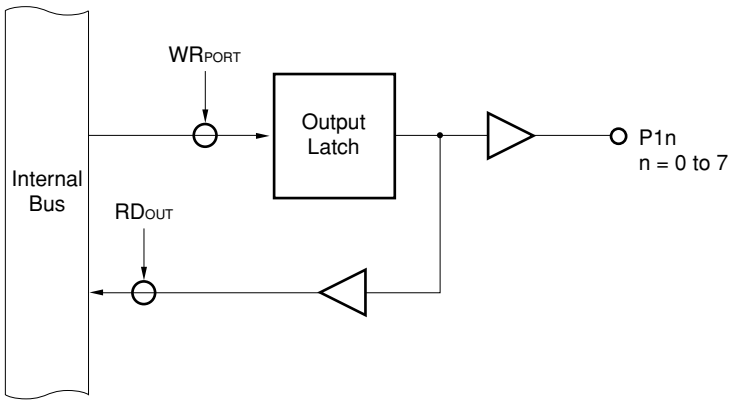
Port 1 is an input/output port. Pins P10 and P11 have an alternate function as PWM signal output pins, and pins P12 to P14 have an alternate function as serial interface pins.

(1) When set as an output port

The output latch is enabled, and data transfers between the output latch and accumulator are performed by means of transfer instructions. The output latch contents can be freely set by means of logical operation instructions. Once data has been written to the output latch, it is retained until data is next written to the output latch **Note**.

Note Including the case where another bit of the same port is manipulated by a bit manipulation instruction.

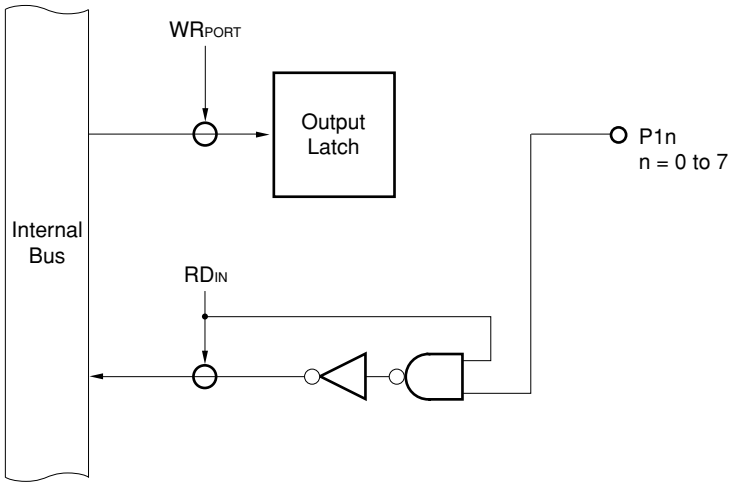
Figure 5-16 Port Specified as Output Port



(2) When set as an input port

The port pin level can be loaded into an accumulator by means of a transfer instruction, etc. In this case, too, writes can be performed to the output latch, and data transferred from the accumulator by a transfer instruction, etc., is stored in all output latches irrespective of the port input/output specification. However, since the output buffer of a bit specified as an input port is high-impedance, the data is not output to the port pin (when a bit specified as input is switched to an output port, the output latch contents are output to the port pin). Also, the contents of the output latch of a bit specified as an input port cannot be loaded into an accumulator.

Figure 5-17 Port Specified as Input Port

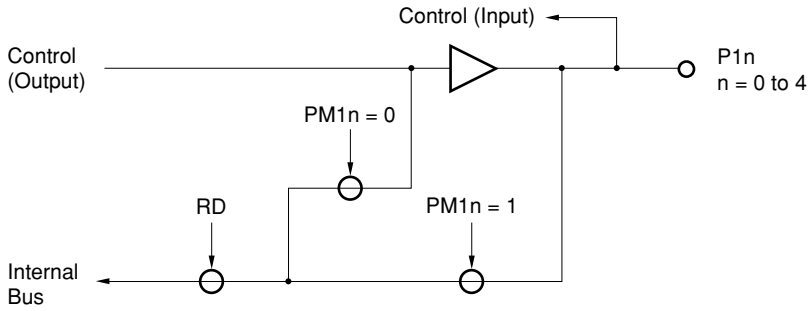


Caution A bit manipulation instruction manipulates one bit as the result, but accesses the port in 8-bit units. Therefore, if a bit manipulation instruction is used on a port that has the I/O mode or port mode and control mode, the contents of the output latch of the pin set in the input mode or control mode become undefined (excluding bits manipulated with a SET1 or CLR1 instruction, etc.). Particular care is required when there are bits which are switched between input and output.
Caution is also required when manipulating the port with other 8-bit manipulation instructions.

(3) When specified as control signal input/output

P10 and P11 (by setting (to 1) the ENn bit (n = 0 or 1) of the PWM control register (PWMC)) and P12 to P14 (by setting (to 1) bits of the port 1 mode control register (PMC1)) can be used as control signal inputs or outputs bit-wise irrespective of the setting of the port 1 mode register (PM1). When a pin is used as a control signal, the control signal status can be seen by executing a port read instruction.

Figure 5-18 Control Specification



(a) When port is control signal output

When the port 1 mode register (PM1) is set (to 1), the control signal pin level can be read by executing a port read instruction.

When PM1 is reset (to 0), the μPD784038 internal control signal status can be read by executing a port read instruction.

(b) When port is control signal input

When the port 1 mode register (PM1) is set (to 1), control signal pin level can be read by executing a port read instruction.

5.3.4 Internal Pull-Up Resistors

Port 1 incorporates pull-up resistors. Use of these internal resistors when pull-up is necessary enables the number of parts and the mounting area to be reduced.

Whether or not an internal pull-up resistor is to be used can be specified for each pin by means of the PUO1 bit of the pull-up resistor option register (PUO) and the port 1 mode register (PM1). When PUO1 is 1, the internal pull-up resistors of the pins for which input is specified by PM1 are enabled (PM1n = 1, n = 0 to 7).

Also, the specification for use of the pull-up resistor is also valid for pins specified as control signal output pins (pull-up resistors are also connected to pins that function as control signal output pins). Therefore, if you do not want to connect the pull-up resistors with the control signal output pin, the contents of the corresponding bits of PM1 should be set to 0 (output mode).

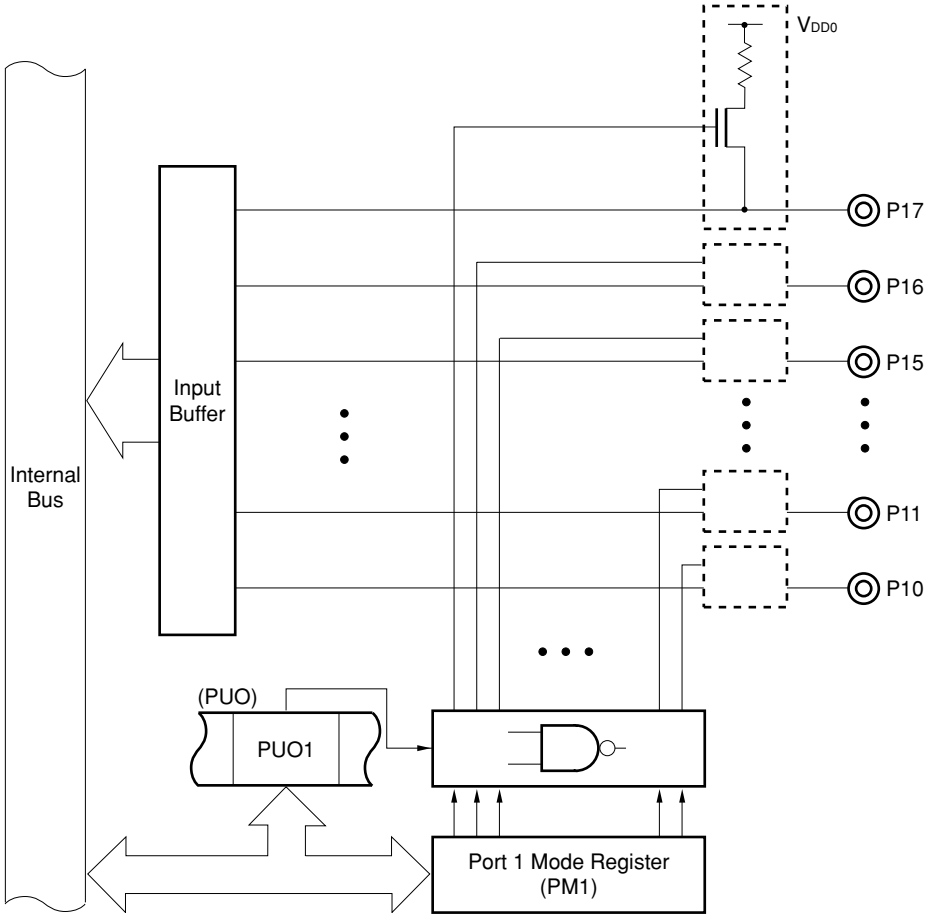
Figure 5-19 Pull-Up Resistor Option Register (PUO) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PUO	0	PUO6	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	0FF4EH	00H	R/W

PUO1	Port 1 Pull-Up Resistor Specification
0	Not used in port 1
1	Used in port 1

Remark When STOP mode is entered, setting 00H in PUO is effective in reducing the power consumption.

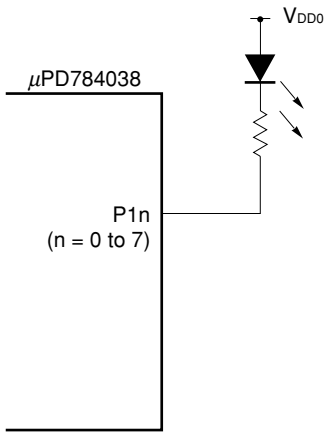
Figure 5-20 Pull-Up Resistor Specification (Port 1)



5.3.5 Direct LED Drive

In port 1, the output buffer low-level side drive capability has been reinforced allowing active-low direct LED drive. An example of such use is shown in Figure 5-21.

Figure 5-21 Example of Direct LED Drive



5.4 PORT 2

Port 2 is an 8-bit input-only port. P22 to P27 incorporate a software programmable pull-up resistor. As well as operating as input ports, port 2 pins also operate as control signal input pins, such as external interrupt signal pins (see Table 5-5). All 8 pins are Schmitt-triggered inputs to prevent malfunction due to noise.

Table 5-5 Port 2 Operating Modes

Port Name	Function
P20	Input port/NMI input Note
P21	Input port/INTP0 input/CR11 capture trigger input Timer/counter 1 count clock/real-time output port trigger signal
P22	Input port/INTP1 input/CR22 capture trigger input
P23	Input port/INTP2 input/CI input
P24	Input port/INTP3 input/CR02 capture trigger input/ Timer/count 0 count clock
P25	Input port/INTP4 input/ASCK input/$\overline{\text{SCK1}}$ input/output
P26	Input port/INTP5 input/A/D converter external trigger input
P27	Input port/SI0 input

Note NMI input is acknowledged regardless of whether interrupts are enabled or disabled.

(a) Function as port pins

The pin level can always be read or tested regardless of the alternate-function pin operation.

(b) Functions as control signal input pins

(i) NMI (Non-maskable Interrupt)

The external non-maskable interrupt request input pin. Rising edge detection or falling edge detection can be specified by means of the external interrupt mode register 0 (INTM0).

(ii) INTP0 to INTP5 (Interrupt from Peripherals)

External interrupt request input pins. When the valid edge specified by the external interrupt mode registers 0, 1 (INTM0/INTM1) is detected an interrupt is generated (see **CHAPTER 21 EDGE DETECTION FUNCTION**). In addition, pins INTP0 to INTP3 and INTP5 are also used as external trigger input pins with the various functions shown below.

- INTP0 ..... Timer/counter 1 capture trigger input pin
  External count clock input pin
  Real-time output port trigger input pin
- INTP1 ..... Timer/counter 2 capture register (CR22) capture trigger input pin
- INTP2 ..... Timer/counter 2 external count clock input pin
  Capture/compare register (CR21) capture trigger input pin
- INTP3 ..... Timer/counter 0 capture trigger input pin
  Timer/counter 0 external count clock input pin
- INTP5 ..... A/D converter external trigger input pin

(iii) CI (Clock Input)

The timer/counter 2 external clock input pin.

(iv) ASCK (Asynchronous Serial Clock)

The external baud rate clock input pin.

(v) $\overline{\text{SCK1}}$ (Serial Clock 1)

The serial clock input/output pin (in 3-wire serial I/O 1 mode).

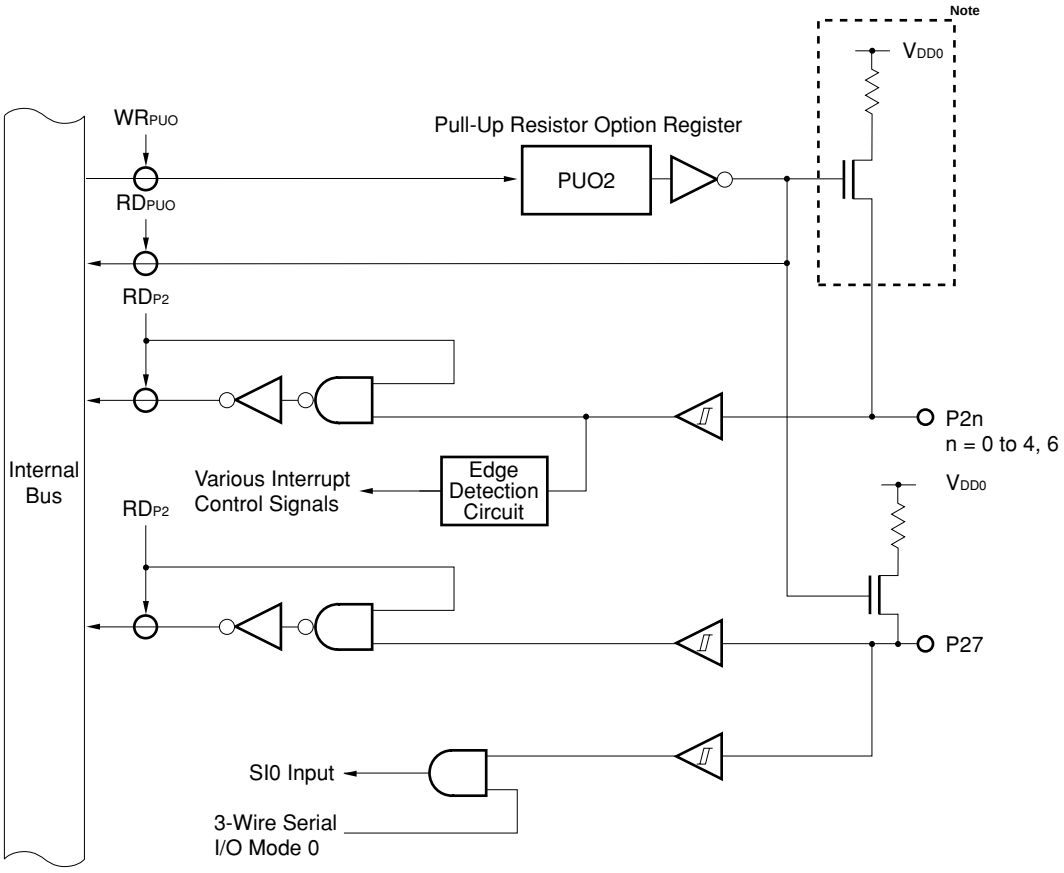
(vi) SI0 (Serial Input 0)

The serial data input pin (in 3-wire serial I/O 0 mode).

5.4.1 Hardware Configuration

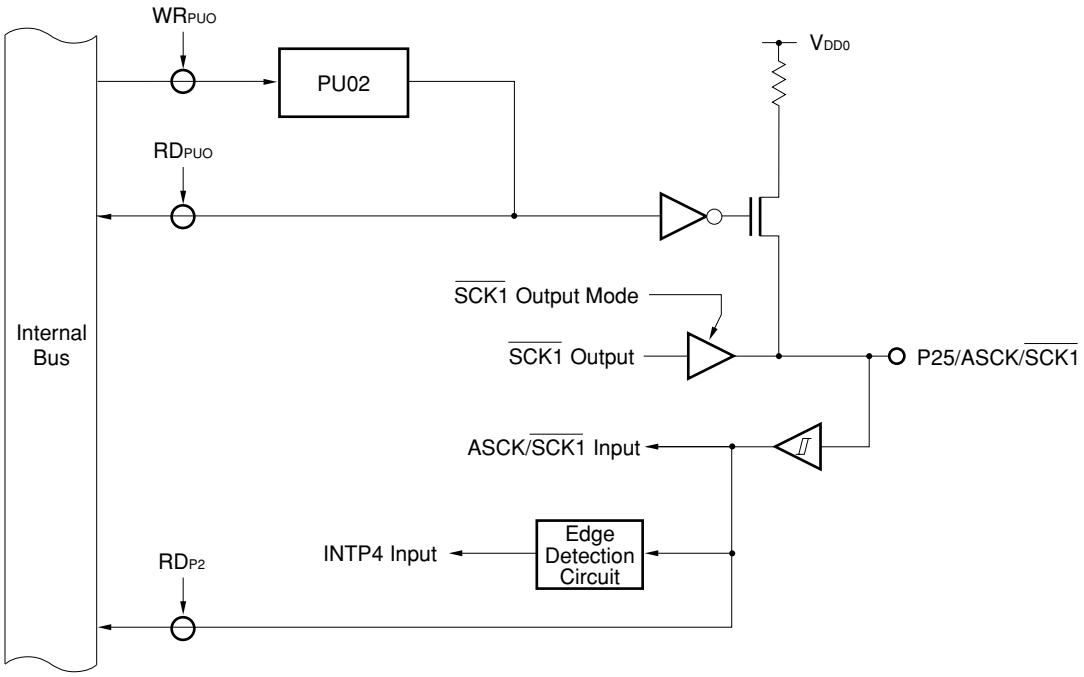
The port 2 hardware configuration is shown in Figure 5-22.

Figure 5-22 Block Diagram of P20 to P24, P26 and P27 (Port 2)



Note P20 and P21 do not have the circuitry enclosed by the dotted line.

Figure 5-23 Block Diagram of P25 (Port 2)



5.4.2 Input Mode/Control Mode Setting

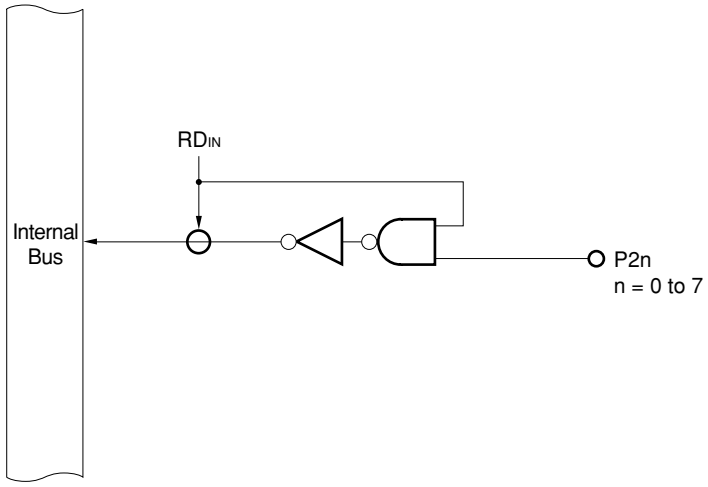
Port 2 is an input-only port, and there is no register for setting the input mode.

Also, control signal input is always possible, and therefore the signal to be used is determined by the control registers for individual on-chip hardware items.

5.4.3 Operating Status

Port 2 is an input-only port, and pin levels can always be read or tested.

Figure 5-24 Port Specified as Input Port



5.4.4 Internal Pull-Up Resistors

P22 to P27 incorporate pull-up resistors. Use of these internal resistors when pull-up is necessary enables the number of parts and the mounting area to be reduced.

Whether or not an internal pull-up resistor is to be used can be specified for all six pins, P22 to P27, together by means of the PUO2 bit of the pull-up resistor option register (PUO) (bit-wise specification is not possible).

P20 and P21 do not incorporate a pull-up resistor.

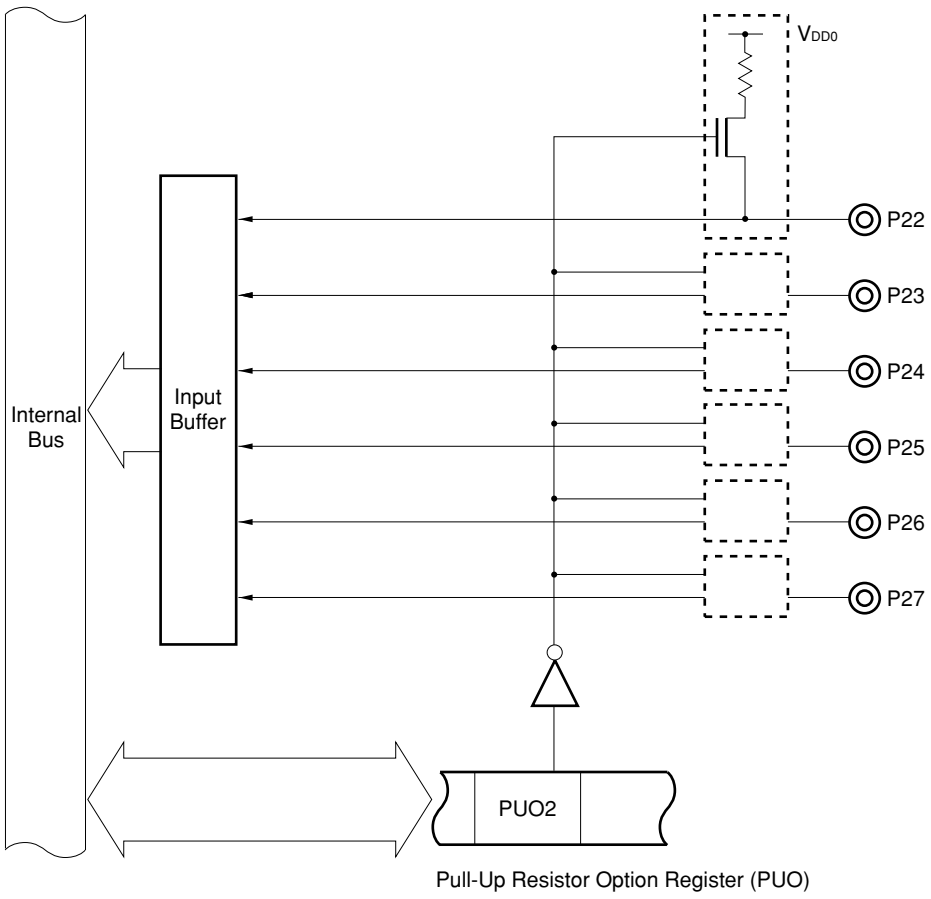
Figure 5-25 Pull-Up Resistor Option Register (PUO) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PUO	0	PUO6	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	0FF4EH	00H	R/W

PUO2	Port 2 Pull-Up Resistor Specification
0	Not used in port 2
1	Used in pins P22 to P27

Remark When STOP mode is entered, setting 00H in PUO is effective in reducing the power consumption.

Figure 5-26 Pull-Up Specification (Port 2)



Caution As P22 to P26 are not pulled up immediately after a reset, an interrupt request flag may be set depending on the function of the alternate-function pins (INTP1 to INTP5). Therefore, the interrupt request flags should be cleared after specifying pull-up in the initialization routine.

5.5 PORT 3

Port 3 is an 8-bit input/output port with an output latch. Input/output can be specified bit-wise by means of the port 3 mode register (PM3). Each pin incorporates a software programmable pull-up resistor.

In addition to its function as an input/output port, port 3 also has various alternate-function control signal pin functions.

The operating mode can be specified bit-wise by means of the port 3 mode control register (PMC3), as shown in Table 5-6. The pin level of all pins can always be read or tested regardless of the alternate-function pin operation.

When $\overline{\text{RESET}}$ is input, port 3 is set as an input port (output high impedance state), and the output latch contents are undefined.

Table 5-6 Port 3 Operating Modes

(n = 0 to 7)

Mode	Port Mode	Control Signal Input/Output Mode
Setting Condition	PMC3n = 0	PMC3n = 1
P30	Input/output port	RxD input/SI1 input
P31		TxD output/SO1 output
P32		$\overline{\text{SCK0}}$ input/output/SCL input/output
P33		SO0 output/SDA input/output
P34		TO0 output
P35		TO1 output
P36		TO2 output
P37		TO3 output

(a) Port mode

Each port specified as port mode by the port 3 mode control register (PMC3) can be specified as input/output bit-wise by means of the port 3 mode register (PM3).

(b) Control signal input/output mode

Pins can be set as control pins bit-wise by setting the port 3 mode control register (PMC3).

(i) RxD (Receive Data)/SI1 (Serial Input 1)

RxD is the asynchronous serial interface serial data input pin. SI1 is the serial data input pin (in 3-wire serial I/O 1 mode).

(ii) TxD (Transmit Data)/SO1 (Serial Output 1)

TxD is the asynchronous serial interface serial data output pin. SO1 is the serial data output pin (in 3-wire serial I/O 1 mode).

(iii) $\overline{\text{SCK0}}$ (Serial Clock 0)/SCL (Serial Clock)

$\overline{\text{SCK0}}$ is the clocked serial interface serial clock input/output pin (in 3-wire serial I/O 0 mode).

SCL is the serial clock I/O pin of the clocked serial interface (in 2-wire serial I/O mode/I^2C bus mode **Note**).

Note μPD784038Y Subseries only

Remark Bit 2 (P32) of port 3 is reserved for the NEC Electronics assembler package as “SCL”. It is also defined as a bit type sfr variable by the #pragma sfr command of the C compiler.

(iv) SO0 (Serial Output 0)/SDA (Serial Data)

SO0 is the serial data output pin (in 3-wire serial I/O 0 mode), and SDA is the serial data input/output pin (in 2-wire serial I/O mode/I²C bus mode [Note]).

Note μPD784038Y Subseries only

(v) TO0 to TO3 (Timer Output)

Timer output pins.

5.5.1 Hardware Configuration

The port 3 hardware configuration is shown in Figures 5-27 to 5-30.

Figure 5-27 Block Diagram of P30 (Port 3)

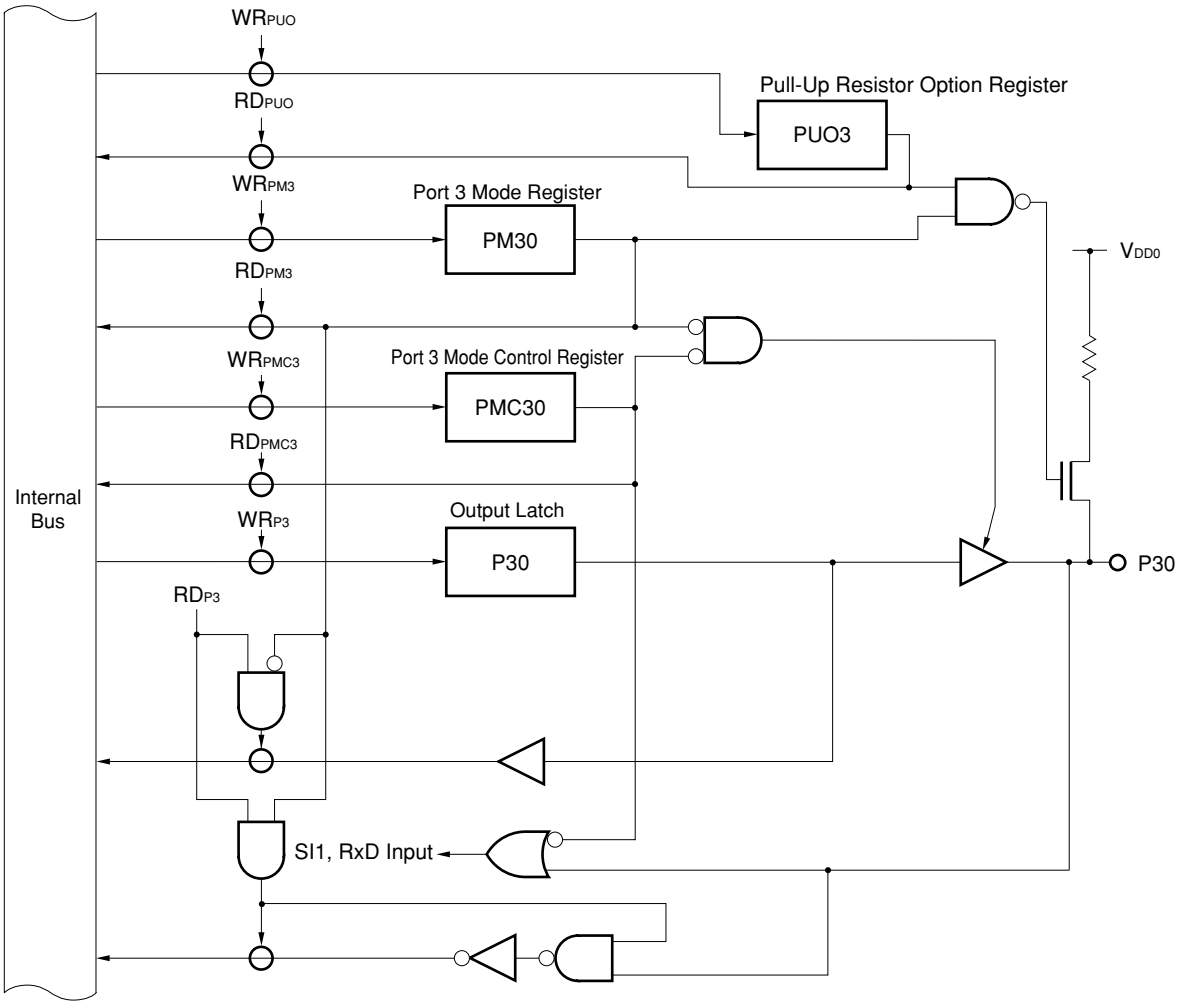


Figure 5-28 Block Diagram of P31 and P34 to P37 (Port 3)

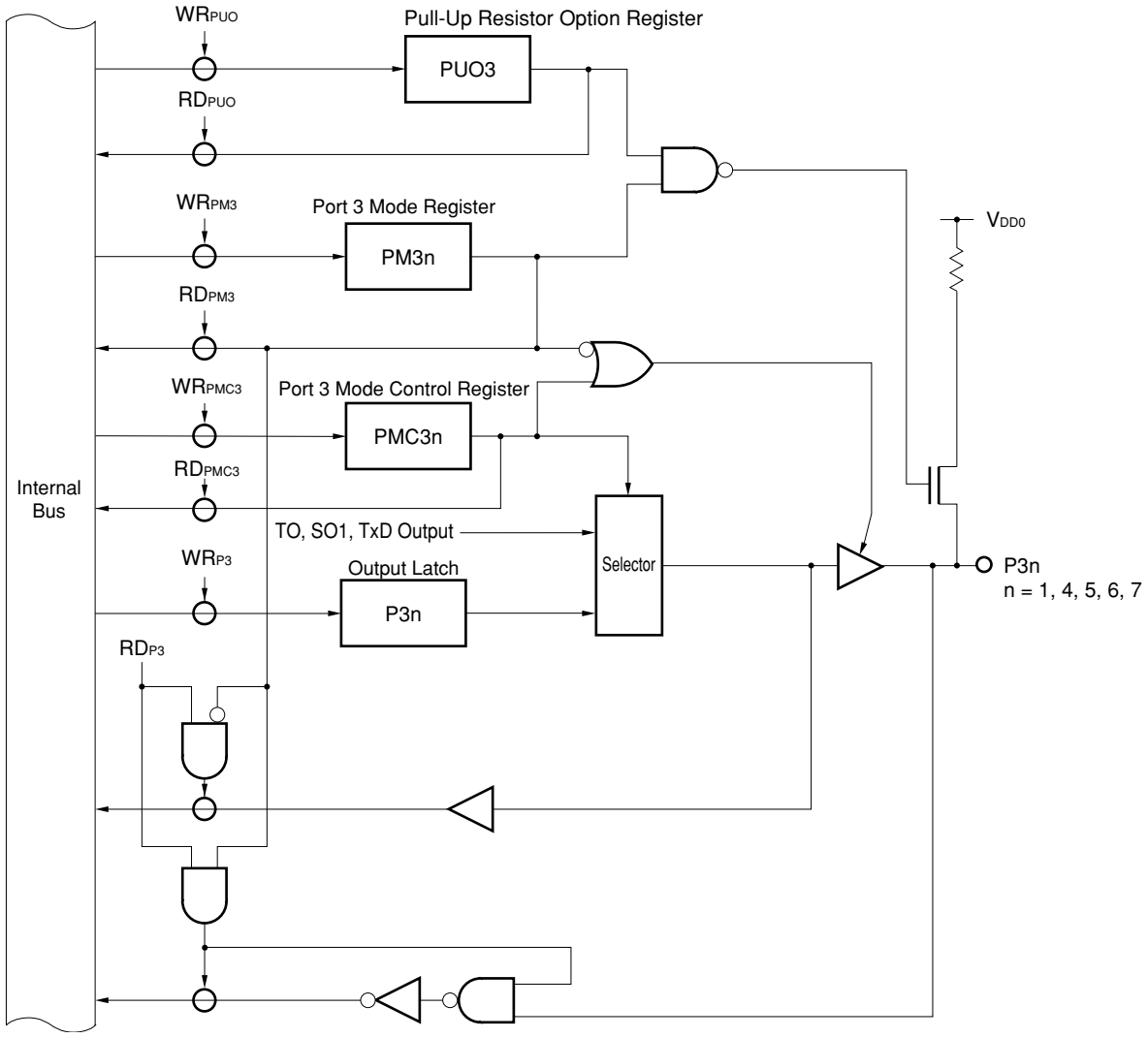


Figure 5-29 Block Diagram of P32 (Port 3)

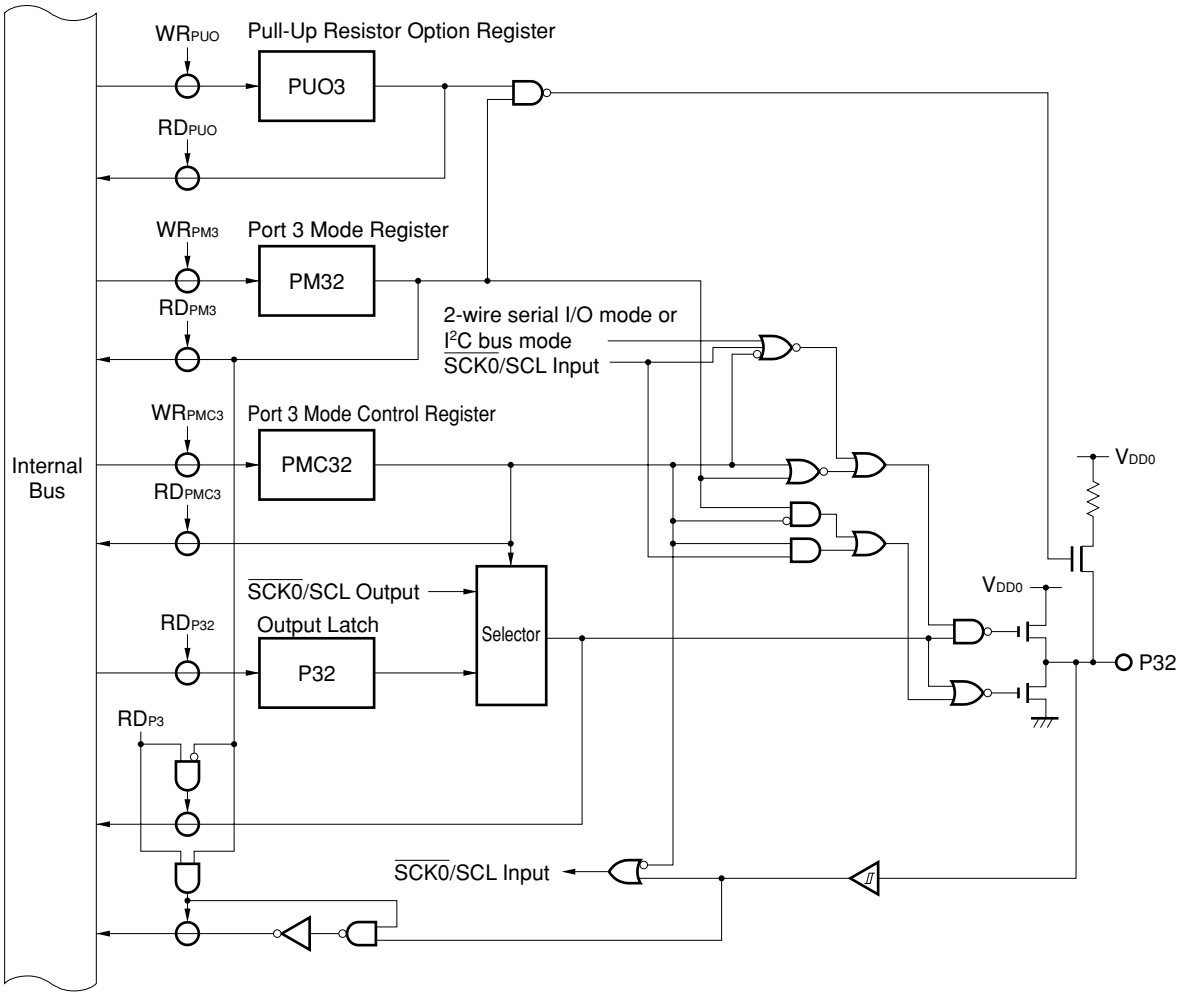
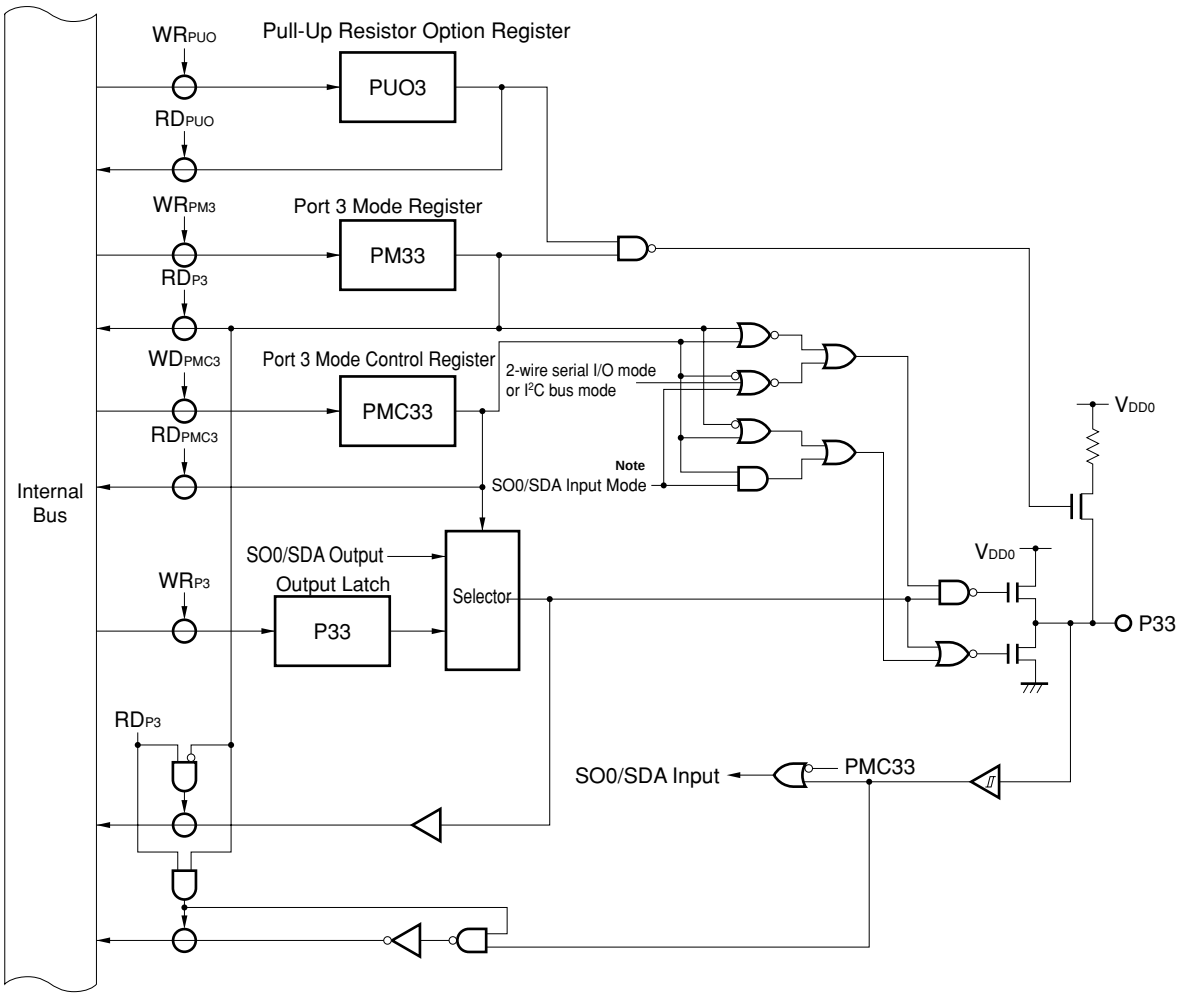


Figure 5-30 Block Diagram of P33 (Port 3)



Note Always 0 in the I^2C bus mode

5.5.2 I/O Mode/Control Mode Setting

The port 3 input/output mode is set for each pin by means of the port 3 mode register (PM3) as shown in Figure 5-31.

In addition to their input/output port function, port 3 pins also have an alternate function as various control signal pins, and the control mode is specified by means of the port 3 mode control register (PMC3) as shown in Figure 5-32.

Figure 5-31 Port 3 Mode Register (PM3) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	0FF23H	FFH	R/W

PM3n	P3n Pin Input/Output Mode Specification (n = 0 to 7)
0	Output mode (output buffer on)
1	Intput mode (output buffer off)

Figure 5-32 Port 3 Mode Control Register (PMC3) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PMC3	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	0FF43H	00H	R/W

PMC30	P30 Pin Control Mode Specification
0	Input/output port mode
1	RxD/SI1 input mode

PMC31	P31 Pin Control Mode Specification
0	Input/output port mode
1	TxD/SO1 output mode

PMC32	P32 Pin Control Mode Specification
0	Input/output port mode
1	$\overline{\text{SCK0}}$/SCL Input/output mode

PMC33	P33 Pin Control Mode Specification
0	Input/output port mode
1	SO0/SDA output mode

PMC3n	P3n Pin Control Mode Specification (n = 4 to 7)
0	Input/output port mode
1	TOm output mode (m = 0 to 3)

5.5.3 Operating Status

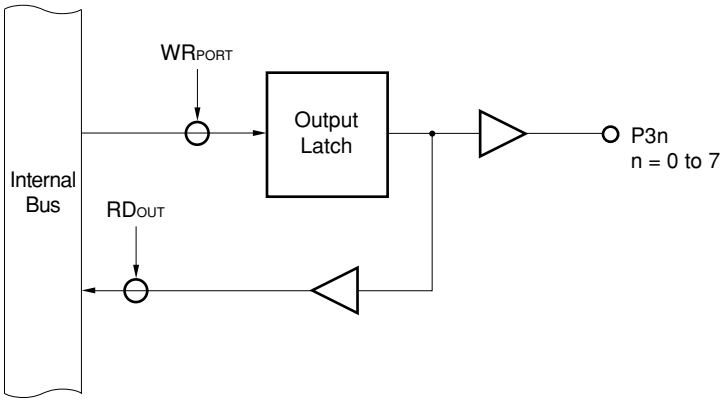
Port 3 is an input/output port, with an alternate function as various control pins.

(1) When set as an output port

The output latch is enabled, and data transfers between the output latch and accumulator are performed by means of transfer instructions. The output latch contents can be freely set by means of logical operation instructions. Once data has been written to the output latch, it is retained until data is next written to the output latch **Note**.

Note Including the case where another bit of the same port is manipulated by a bit manipulation instruction.

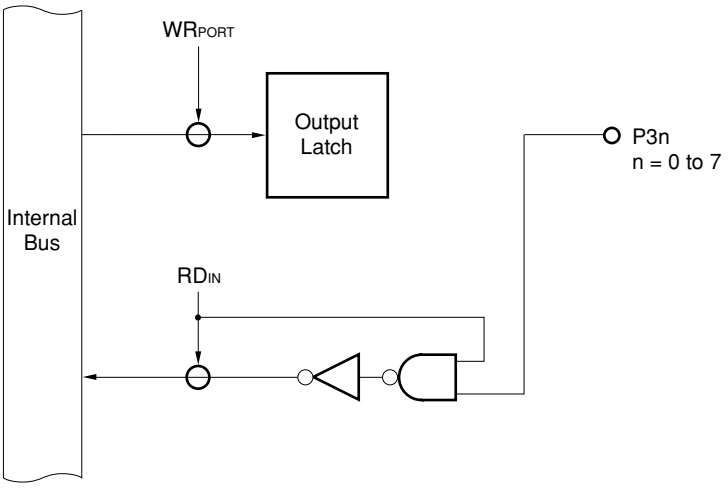
Figure 5-33 Port Specified as Output Port



(2) When set as an input port

The port pin level can be loaded into an accumulator by means of a transfer instruction. In this case, too, writes can be performed to the output latch, and data transferred from the accumulator by a transfer instruction, etc., is stored in all output latches irrespective of the port input/output specification. However, since the output buffer of a bit specified as an input port is high impedance, the data is not output to the port pin (when a bit specified as input is switched to an output port, the output latch contents are output to the port pin). Also, the contents of the output latch of a bit specified as an input port cannot be loaded into an accumulator.

Figure 5-34 Port Specified as Input Port



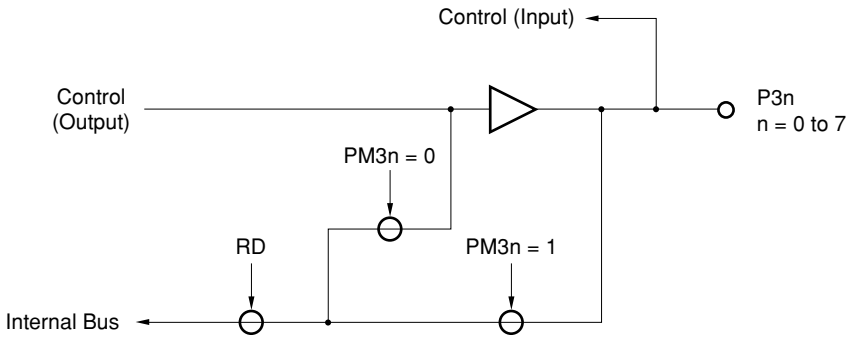
Caution A bit manipulation instruction manipulates one bit as the result, but accesses the port in 8-bit units. Therefore, if a bit manipulation instruction is used on a port with a mixture of input and output pins or port mode and control mode, the contents of the output latch of pins specified as inputs and pins specified as control mode will be undefined (excluding bits manipulated with a SET1 or CLR1 instruction, etc.). Particular care is required when there are bits which are switched between input and output.

Caution is also required when manipulating the port with other 8-bit manipulation instructions.

(3) When specified as control signal input/output

By setting (to 1) bits of the port 3 mode control register (PMC3), port 3 can be used as control signal input or output bit-wise irrespective of the setting of the port 3 mode register (PM3). When a pin is used as a control signal, the control signal status can be seen by executing a port read instruction.

Figure 5-35 Control Specification



(a) When port is control signal output

When the port 3 mode register (PM3) is set (to 1), the control signal pin level can be read by executing a port read instruction.

When PM3 is reset (to 0), the μPD784038 internal control signal status can be read by executing a port read instruction.

Remark For bit 2 (P32) of port 3, the name “SCL” is a reserved word in the NEC Electronics assembler package. In the C compiler, it is defined as a bit-type sfr variable by the # pragma sfr directive.

(b) When port is control signal input

Only the port 3 mode register (PM3) is set (to 1), control signal pin levels can be read by executing a port read instruction.

5.5.4 Internal Pull-Up Resistors

Port 3 incorporates pull-up resistors. Use of these internal resistors when pull-up is necessary enables the number of parts and the mounting area to be reduced.

Whether or not an internal pull-up resistor is to be used can be specified for each pin by means of the PUO3 bit of the pull-up resistor option register (PUO) and the port 3 mode register (PM3). When PUO3 is 1, the internal pull-up resistors of the pins for which input is specified by PM3 (PM3n = 1, n = 0 to 7) are enabled.

Also, the specification for use of the pull-up resistor is also valid for pins specified as control mode pins (pull-up resistors are also connected to pins that function as output pins in the control mode). Therefore, if you do not want to connect the pull-up resistors in the control mode, the contents of the corresponding bits of PM3 should be set to 0 (output mode).

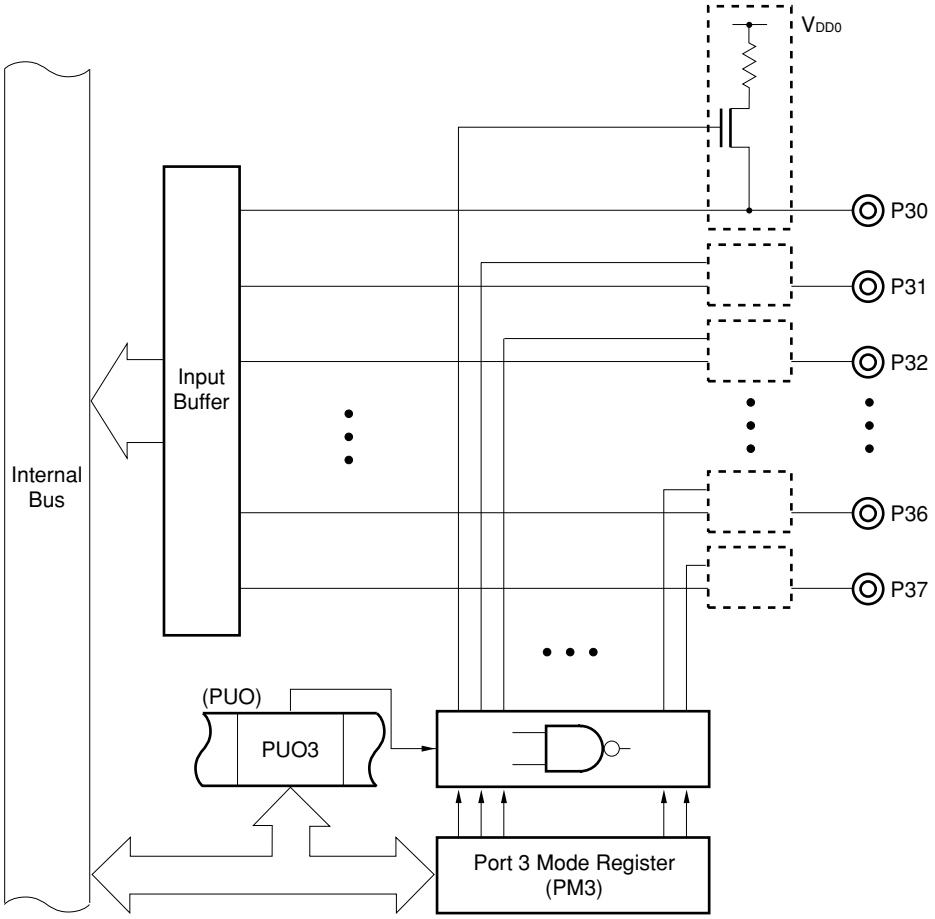
Figure 5-36 Pull-Up Resistor Option Register (PUO) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PUO	0	PUO6	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	0FF4EH	00H	R/W

PUO3	Port 3 Pull-Up Resistor Specification
0	Not used in port 3
1	Used in port 3

Remark When STOP mode is entered, setting 00H in PUO is effective in reducing the power consumption.

Figure 5-37 Pull-Up Specification (Port 3)



5.6 PORT 4

Port 4 is an 8-bit input/output port with an output latch. Input/output can be specified bit-wise by means of the port 4 mode register (PM4). Each pin incorporates a software programmable pull-up resistor. This port has direct LED drive capability.

Port 4 also functions as the time division address/data bus (AD0 to AD7) by the memory extension mode register (MM) when external memory or I/Os are extended.

With the μPD784031, P40 to P47 cannot be used as port pins. These pins function only as address/data bus pins (AD0 to AD7).

When $\overline{\text{RESET}}$ is input, port 4 is set as an input port (output high-impedance state), and the output latch contents are undefined.

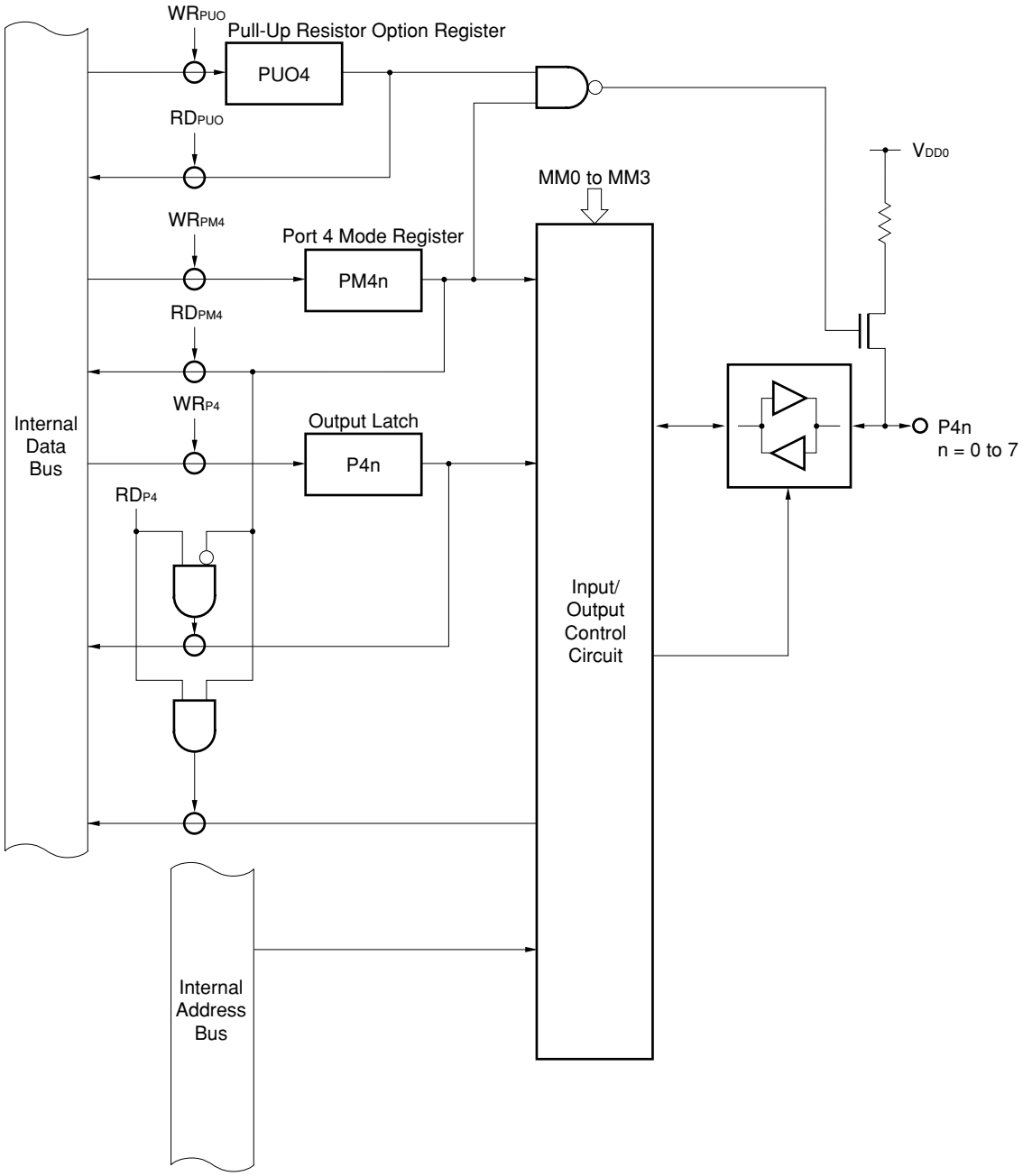
Table 5-7 Port 4 Operating Modes

MM Bits				Operating Mode
MM3	MM2	MM1	MM0	
0	0	0	0	Port
0	0	1	1	Address/data bus (AD0 to AD7)
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	

5.6.1 Hardware Configuration

The port 4 hardware configuration is shown in Figure 5-38.

Figure 5-38 Port 4 Block Diagram



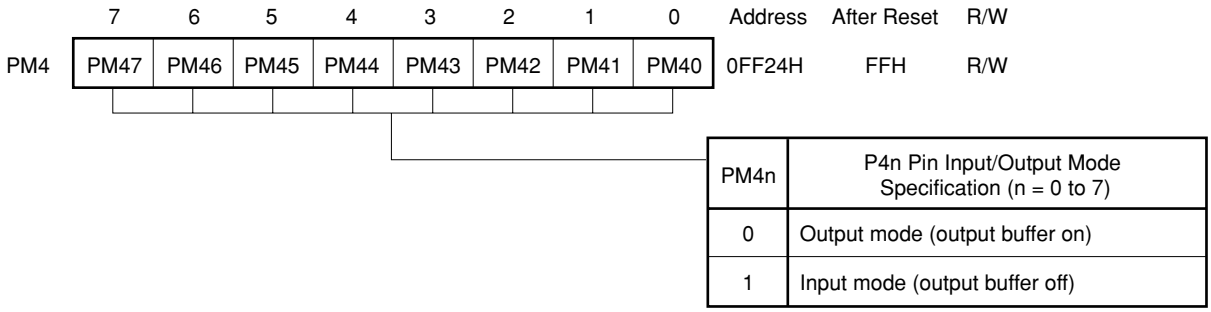
5.6.2 I/O Mode/Control Mode Setting

The port 4 input/output mode is set for each pin by means of the port 4 mode register (PM4) as shown in Figure 5-39.

When port 4 is used as the address/data bus, it is set by means of the memory extension mode register (MM: See **Figure 23-1**) as shown in Table 5-8.

With the μPD784031, this port functions only as the address/data bus (AD0 to AD7).

Figure 5-39 Port 4 Mode Register (PM4) Format



PM4n	P4n Pin Input/Output Mode Specification (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Table 5-8 Port 4 Operating Modes

MM Bits				Operating Mode
MM3	MM2	MM1	MM0	
0	0	0	0	Port
0	0	1	1	Address/data bus (AD0 to AD7)
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	

5.6.3 Operating Status

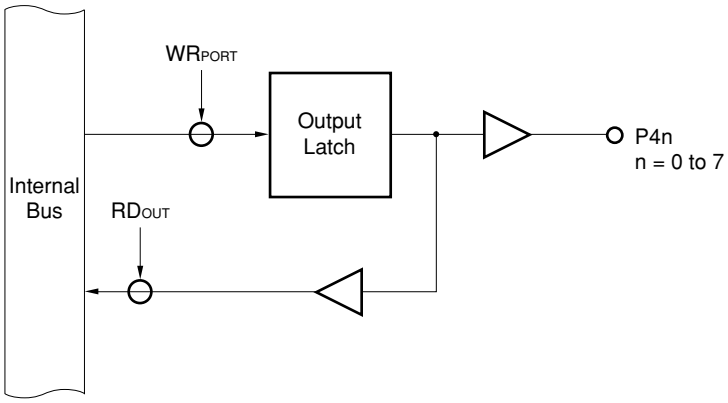
Port 4 is an input/output port, with an alternate function as the address/data bus (AD0 to AD7).

(1) When set as an output port

The output latch is enabled, and data transfers between the output latch and accumulator are performed by means of transfer instructions. The output latch contents can be freely set by means of logical operation instructions. Once data has been written to the output latch, it is retained until data is next written to the output latch **Note**.

Note Including the case where another bit of the same port is manipulated by a bit manipulation instruction.

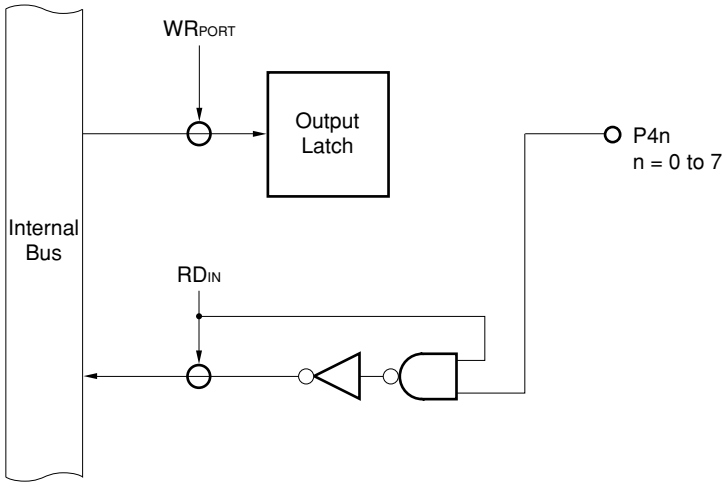
Figure 5-40 Port Specified as Output Port



(2) When set as an input port

The port pin level can be loaded into an accumulator by means of a transfer instruction. In this case, too, writes can be performed to the output latch, and data transferred from the accumulator by a transfer instruction, etc., is stored in all output latches irrespective of the port input/output specification. However, since the output buffer of a bit specified as an input port is high-impedance, the data is not output to the port pin (when a port specified as input is switched to an output port, the output latch contents are output to the port pin). Also, when specified as an input port, the output latch contents cannot be loaded into an accumulator.

Figure 5-41 Port Specified as Input Port



Caution A bit manipulation instruction manipulates one bit as the result, but accesses the port in 8-bit units. Therefore, if a bit manipulation instruction is used on a port with a mixture of input and output pins, the contents of the output latch of pins specified as inputs will be undefined (excluding bits manipulated with a SET1 or CLR1 instruction, etc.). Particular care is required when there are bits which are switched between input and output.
Caution is also required when manipulating the port with other 8-bit manipulation instructions.

(3) When used as address/data bus (AD0 to AD7)

Used automatically when an external access is performed.
Input/output instructions should not be executed on port 4.

5.6.4 Internal Pull-Up Resistors

Port 4 incorporates pull-up resistors. Use of these internal resistors when pull-up is necessary enables the number of parts and the mounting area to be reduced.

Whether or not an internal pull-up resistor is to be used can be specified for each pin by means of the PUO4 bit of the pull-up resistor option register (PUO) and the port 4 mode register (PM4).

When PUO4 is 1, the internal pull-up resistors of the pins for which input is specified by the PM4 for port 4 (PM4n = 1, n = 0 to 7) are enabled .

Figure 5-42 Pull-Up Resistor Option Register (PUO) Format

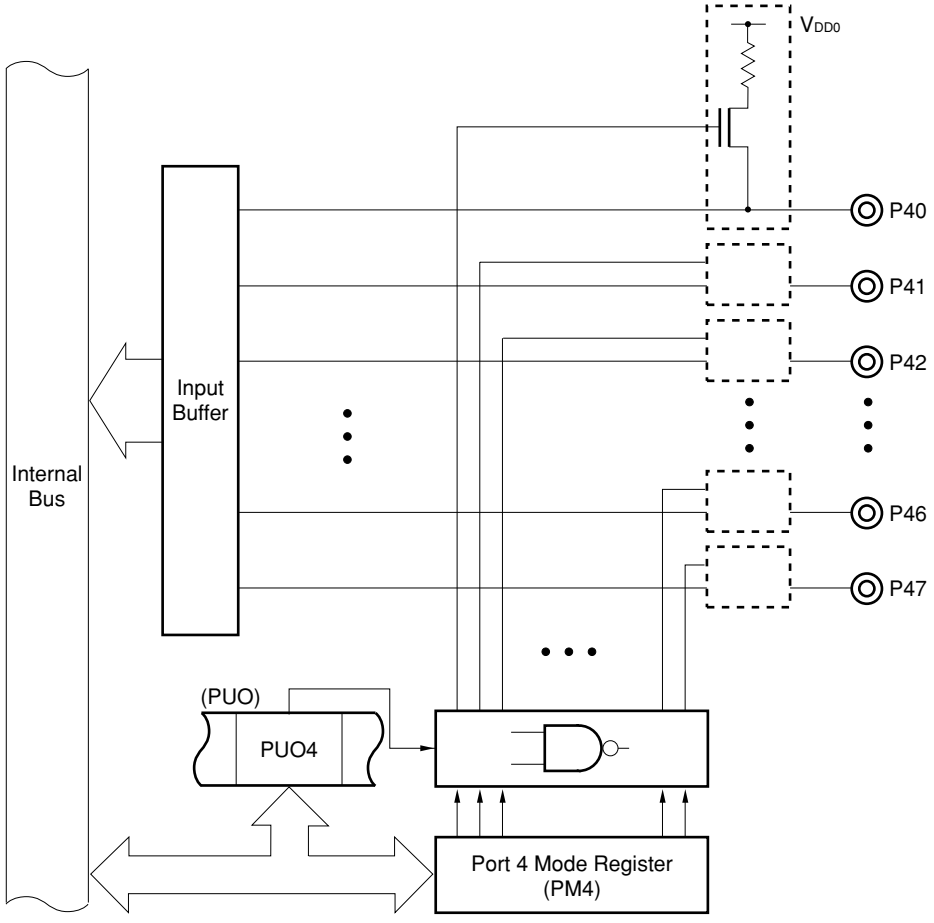
	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PUO	0	PUO6	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	0FF4EH	00H	R/W

PUO4	Port 4 Pull-Up Resistor Specification
0	Not used in port 4
1	Used in port 4

Caution When using the port 4 of the μPD784038 as an address/data bus pin, and with the μPD784031, be sure to clear PUO4 to 0 to disconnect the internal pull-up resistor.

Remark When STOP mode is entered, setting 00H in PUO is effective in reducing the power consumption.

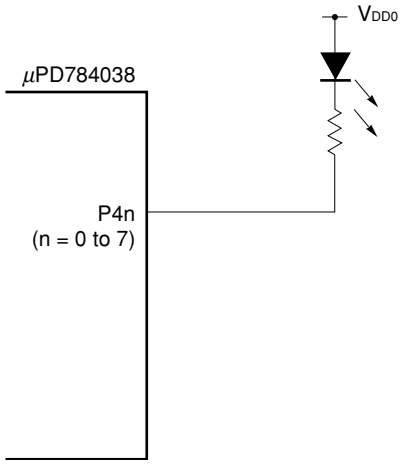
Figure 5-43 Pull-Up Specification (Port 4)



5.6.5 Direct LED Drive

In port 4, the output buffer low-level side drive capability has been reinforced, allowing active-low direct LED drive. An example of such use is shown in Figure 5-44.

Figure 5-44 Example of Direct LED Drive



5.7 PORT 5

Port 5 is an 8-bit input/output port with an output latch. Input/output can be specified bit-wise by means of the port 5 mode register (PM5). Each pin incorporates a software programmable pull-up resistor. This port has direct LED drive capability.

In addition, P50 to P57 function as the address bus (A8 to A15) when external memory or I/Os are extended.

With the μPD784031, P50 to P57 cannot be used as port pins. These pins function only as address bus pins (A8 to A15).

When $\overline{\text{RESET}}$ is input, port 5 is set as an input port (output high-impedance state), and the output latch contents are undefined.

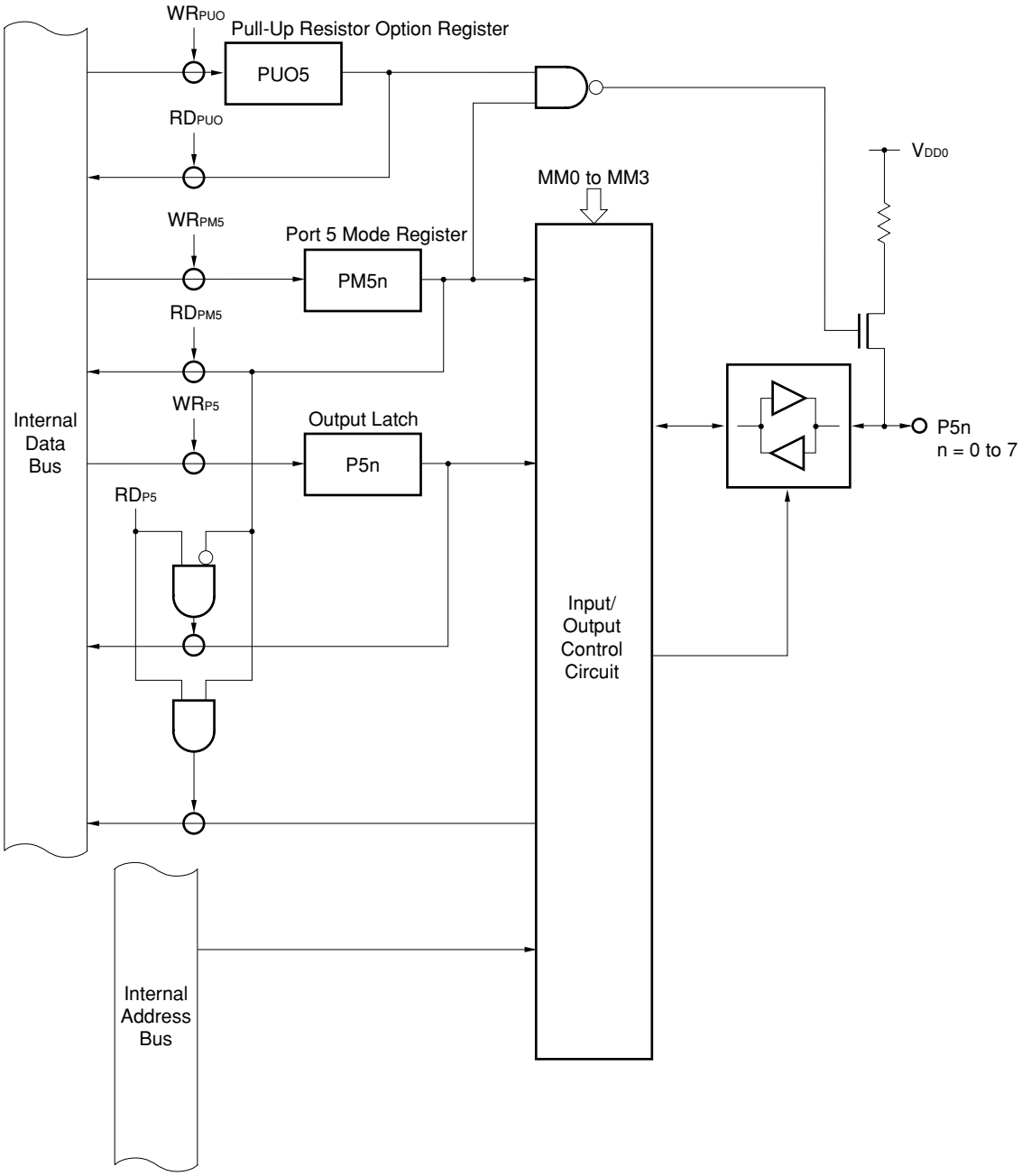
Table 5-9 Port 5 Operating Modes

MM Bits				Operating Mode							
MM3	MM2	MM1	MM0	P50	P51	P52	P53	P54	P55	P56	P57
0	0	0	0	Port (P50 to P57)							
0	0	1	1								
0	1	0	0	A8	A9	Port					
0	1	0	1	A8	A9	A10	A11	Port			
0	1	1	0	A8	A9	A10	A11	A12	A13	Port	
0	1	1	1	A8	A9	A10	A11	A12	A13	A14	A15
1	0	0	0								
1	0	0	1								

5.7.1 Hardware Configuration

The port 5 hardware configuration is shown in Figure 5-45.

Figure 5-45 Port 5 Block Diagram



5.7.2 I/O Mode/Control Mode Setting

The port 5 input/output mode is set for each pin by means of the port 5 mode register (PM5) as shown in Figure 5-46.

When port 5 pins can be used as port or address pins in 2-bit units, the setting is performed by means of the memory extension mode register (MM: See **Figure 23-1**) as shown in Table 5-10.

With the μPD784031, this port functions only as the address bus (A8 to A15).

Figure 5-46 Port 5 Mode Register (PM5) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	0FF25H	FFH	R/W

PM5n	P5n Pin Input/Output Mode Specification (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Table 5-10 Port 5 Operating Modes

MM Bits				Operating Mode							
MM3	MM2	MM1	MM0	P50	P51	P52	P53	P54	P55	P56	P57
0	0	0	0	Port (P50 to P57)							
0	0	1	1								
0	1	0	0	A8	A9	Port					
0	1	0	1	A8	A9	A10	A11	Port			
0	1	1	0	A8	A9	A10	A11	A12	A13	Port	
0	1	1	1	A8	A9	A10	A11	A12	A13	A14	A15
1	0	0	0								
1	0	0	1								

5.7.3 Operating Status

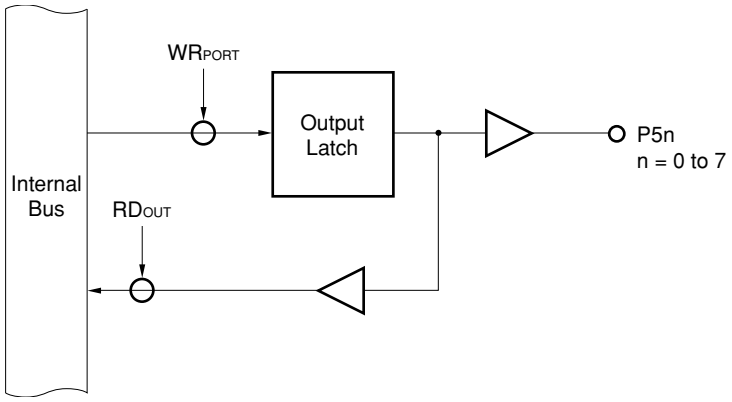
Port 5 is an input/output port, with an alternate function as the address bus (A8 to A15).

(1) When set as an output port

The output latch is enabled, and data transfers between the output latch and accumulator are performed by means of transfer instructions. The output latch contents can be freely set by means of logical operation instructions. Once data has been written to the output latch, it is retained until data is next written to the output latch **Note**.

Note Including the case where another bit of the same port is manipulated by a bit manipulation instruction.

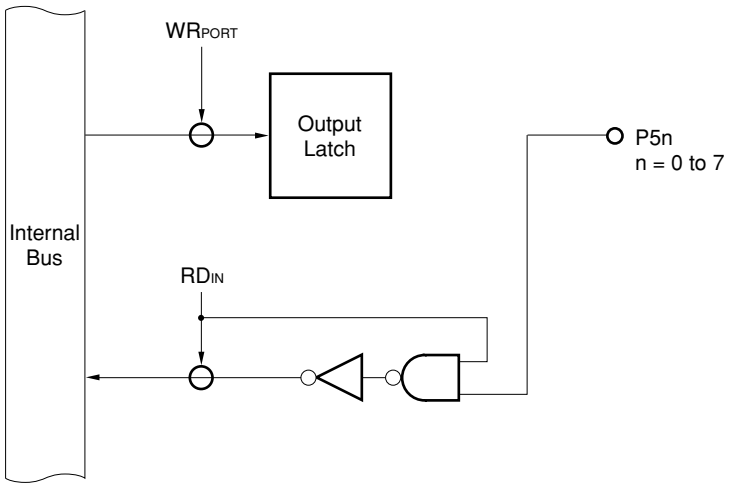
Figure 5-47 Port Specified as Output Port



(2) When set as an input port

The port pin level can be loaded into an accumulator by means of a transfer instruction. In this case, too, writes can be performed to the output latch, and data transferred from the accumulator by a transfer instruction, etc., is stored in all output latches irrespective of the port input/output specification. However, since the output buffer of a bit specified as an input port is high-impedance, the data is not output to the port pin (when a bit specified as input is switched to an output port, the output latch contents are output to the port pin). Also, the contents of the output latch of a bit specified as an input port cannot be loaded into an accumulator.

Figure 5-48 Port Specified as Input Port



Caution A bit manipulation instruction manipulates one bit as the result, but accesses the port in 8-bit units. Therefore, if a bit manipulation instruction is used on a port with a mixture of input and output pins, the contents of the output latch of pins specified as inputs will be undefined (excluding bits manipulated with a SET1 or CLR1 instruction, etc.). Particular care is required when there are bits which are switched between input and output.
Caution is also required when manipulating the port with other 8-bit operation instructions.

(3) When used as address bus (A8 to A15)

Used automatically when an external address is accessed.

5.7.4 Internal Pull-Up Resistors

Port 5 incorporates pull-up resistors. Use of these internal resistors when pull-up is necessary enables the number of parts and the mounting area to be reduced.

Whether or not an internal pull-up resistor is to be used can be specified for each pin by means of the PUO5 bit of the pull-up resistor option register (PUO) and the port 5 mode register (PM5).

When PUO5 is 1, the internal pull-up resistors of the pins for which input is specified by the PM5 for port 5 (PM5n = 1, n = 0 to 7) are enabled .

Figure 5-49 Pull-Up Resistor Option Register (PUO) Format

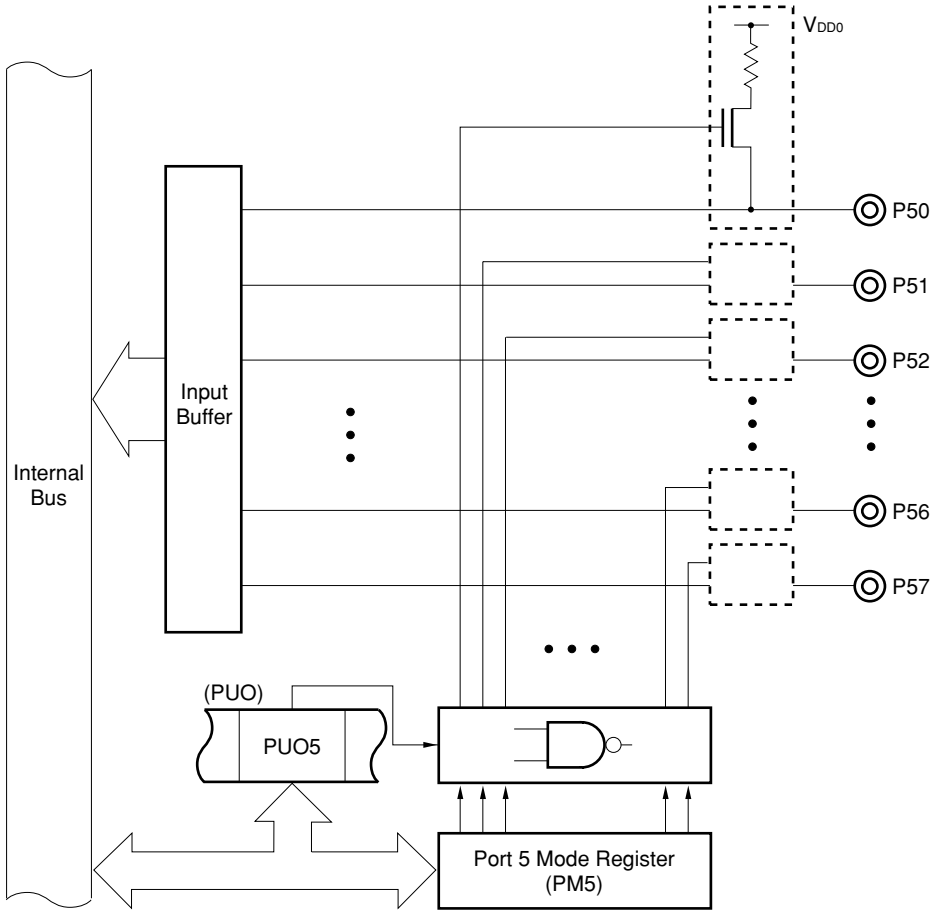
	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PUO	0	PUO6	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	0FF4EH	00H	R/W

PUO5	Port 5 Pull-Up Resistor Specification
0	Not used in port 5
1	Used in port 5

Caution When using the port 5 of the μPD784038 as an address bus, and with the μPD784031, be sure to clear PUO5 to 0 to disconnect the internal pull-up resistor.

Remark When STOP mode is entered, setting 00H in PUO is effective in reducing the power consumption.

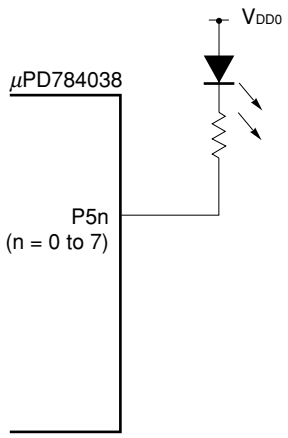
Figure 5-50 Pull-Up Specification (Port 5)



5.7.5 Direct LED Drive

In port 5, the output buffer low-level side drive capability has been reinforced, allowing active-low direct LED drive. An example of such use is shown in Figure 5-51.

Figure 5-51 Example of Direct LED Drive



5.8 PORT 6

- **With μPD784031**

  P60 to P63 are output port pins and P66 and P67 are input/output port pins with output latch.

  P64 to P67 incorporate a software programmable pull-up resistor.

  In addition to the functions as port pins, these pins also have various alternate-function control signal pin functions, as shown in Table 5-11. Operations as control pins are performed by the respective function operations.

  P64 and P65 cannot be used as port pins and function only as $\overline{RD}$ and $\overline{WR}$ output pins.

  When $\overline{RESET}$ is input, the level of the above pins are set as follows:

  • P60 to P63: Low
  • P64, P65: High
  • P66, P67: Input port (output high impedance)

  The higher 4 bits of the contents are undefined, and the lower 4 bits are reset to 0H.

- **With other than μPD784031**

  Port 6 is an 8-bit input/output port with an output latch. P60 to P67 incorporate a software programmable pull-up resistor.

  In addition to its function as a port, port 6 also has various alternate-function control signal pin functions as shown in Table 5-11. Operations as control pins are performed by the respective function operations.

  When $\overline{RESET}$ is input, P60 to P67 are set as input port pins (output high-impedance state), and the output latch contents are undefined.

Table 5-11 Port 6 Operating Modes

Pin Name	Port Mode Output Mode	Control Signal Input/	Operation to Operate as Control Pins
P60 to P63	Input/output ports Note	A16 to A19 outputs	Specified by bits MM3 to MM0 of the MM in 2-bit units
P64		$\overline{RD}$ output	With the μPD784031, or when external memory extension mode is specified by bits MM3 to MM0 of the MM
P65		$\overline{WR}$ output	
P66		$\overline{WAIT}$ input	Specified by bits PWn1 & PWn0 (n = 0 to 7) of the PWC1 & PWC2 or setting P66 in the input mode
		HLDRQ input	Bus hold enabled by the HLDE bit of the HLDM
P67		HLDAK output	
		$\overline{REFRQ}$ output	Set (to 1) the RFEN bit of the RFM

Notes 1. These pins of the μPD784031 are output port pins.

2. With the μPD784031, this pin cannot be used as a port pin.

Caution P60 to P63 of the μPD784031 are in the output high-impedance state while the $\overline{RESET}$ signal is input, but output a low level after the $\overline{RESET}$ signal has been cleared. Therefore, design the external circuit so that the low level may be output as the initial status.

Remark For details, refer to **CHAPTER 23 LOCAL BUS INTERFACE FUNCTION**.

Table 5-12 P60 to P65 Control Pin Specification

MM Bits				Operating Mode					
MM3	MM2	MM1	MM0	P60	P61	P62	P63	P64	P65
0	0	0	0	Port (P60 to P65)					
0	0	1	1						
0	1	0	0	Port (P60 to P63)				$\overline{RD}$	$\overline{WR}$
0	1	0	1						
0	1	1	0						
0	1	1	1						
1	0	0	0	A16	A17	Port			
1	0	0	1	A16	A17	A18	A19		

(a) Port mode

- **With μPD784031**

  Each port not specified as control mode, P66 and P67 serve as output port pins, and P66 and P67 can be specified as input/output bit-wise by means of the port 6 mode register (PM6).

- **With other than μPD784031**

  Each port not specified as in control mode can be specified as input/output bit-wise by means of the port 6 mode register (PM6).

(b) Control signal input/output mode

(i) A16 to A19 (Address Bus)

Upper address bus output pins when the external memory space is expanded (10000H to FFFFFH).
These pins operate in accordance with the memory extension mode register (MM).

(ii) $\overline{RD}$ (Read Strobe)

The strobe signal for an external memory read operation. The operation of this pin is controlled by the memory expansion mode register (MM).
With the μPD784031, this pin always serves as an $\overline{RD}$ pin.

(iii) $\overline{WR}$ (Write Strobe)

Pin that outputs the strobe signal for an external memory write operation. The operation of this pin is controlled by the memory expansion mode register (MM).
With the μPD784031, this pin always serves as a $\overline{WR}$ pin.

(iv) $\overline{WAIT}$ (Wait)

Wait signal input pin. Operates in accordance with the programmable wait control registers (PWC1, PWC2).

(v) HLDRQ (Hold Request)

External bus hold request signal input pin. Operates in accordance with the hold mode register (HLDM).

(vi) HLDAK (Hold Acknowledge)

Bus hold acknowledge signal output pin. Operates in accordance with the hold mode register (HLDM).

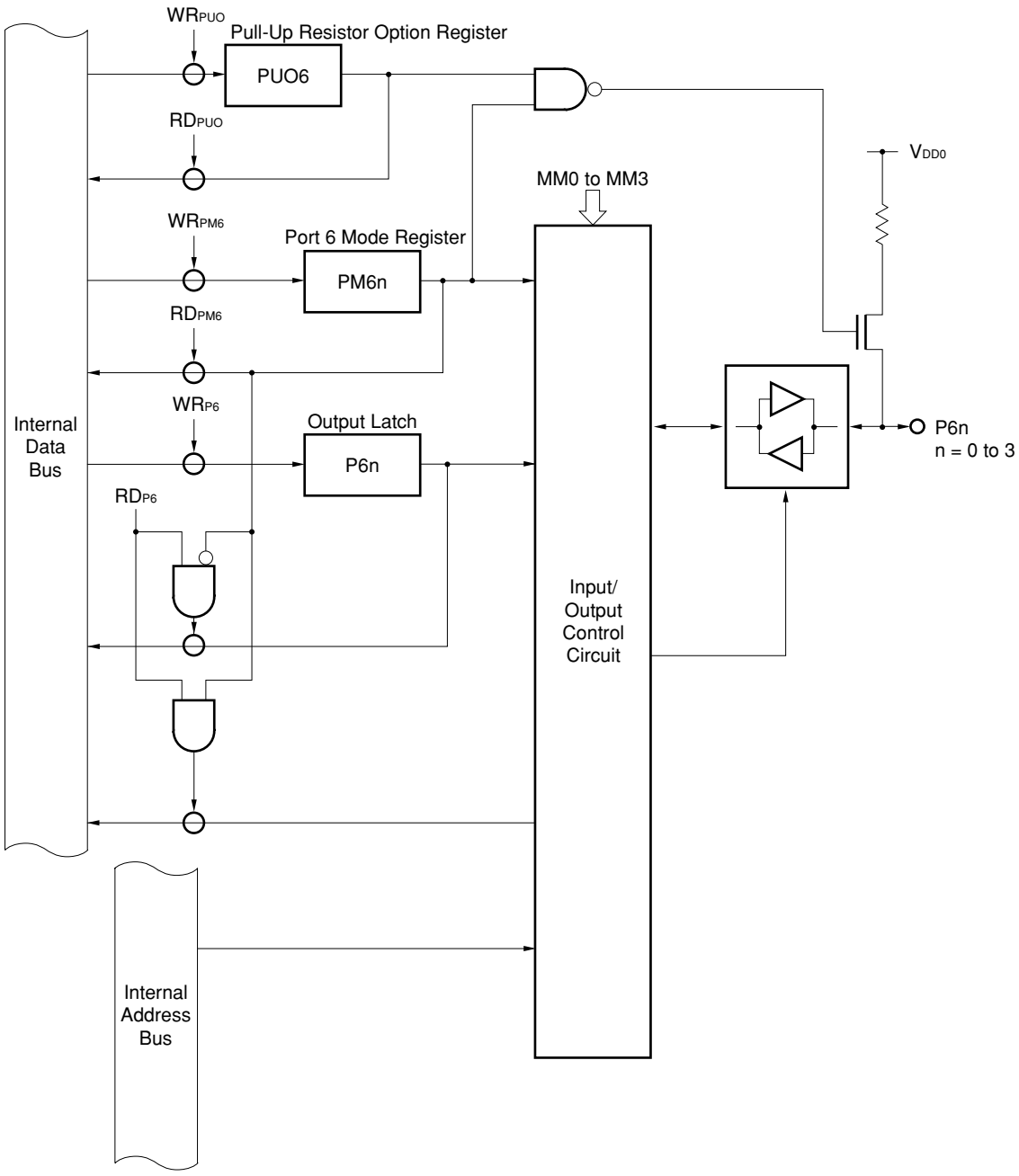
(vii) $\overline{\text{REFRQ}}$ (Refresh Request)

This pin outputs refresh pulses to pseudo-static memory when this memory is connected to it externally. Operates in accordance with the refresh mode register (RFM).

5.8.1 Hardware Configuration

The port 6 hardware configuration is shown in Figures 5-52 to 5-55.

Figure 5-52 Block Diagram of P60 to P63 (Port 6)



Remark The μPD784031 does not have a function for input operation.

Figure 5-53 Block Diagram of P64 and P65 (Port 6)

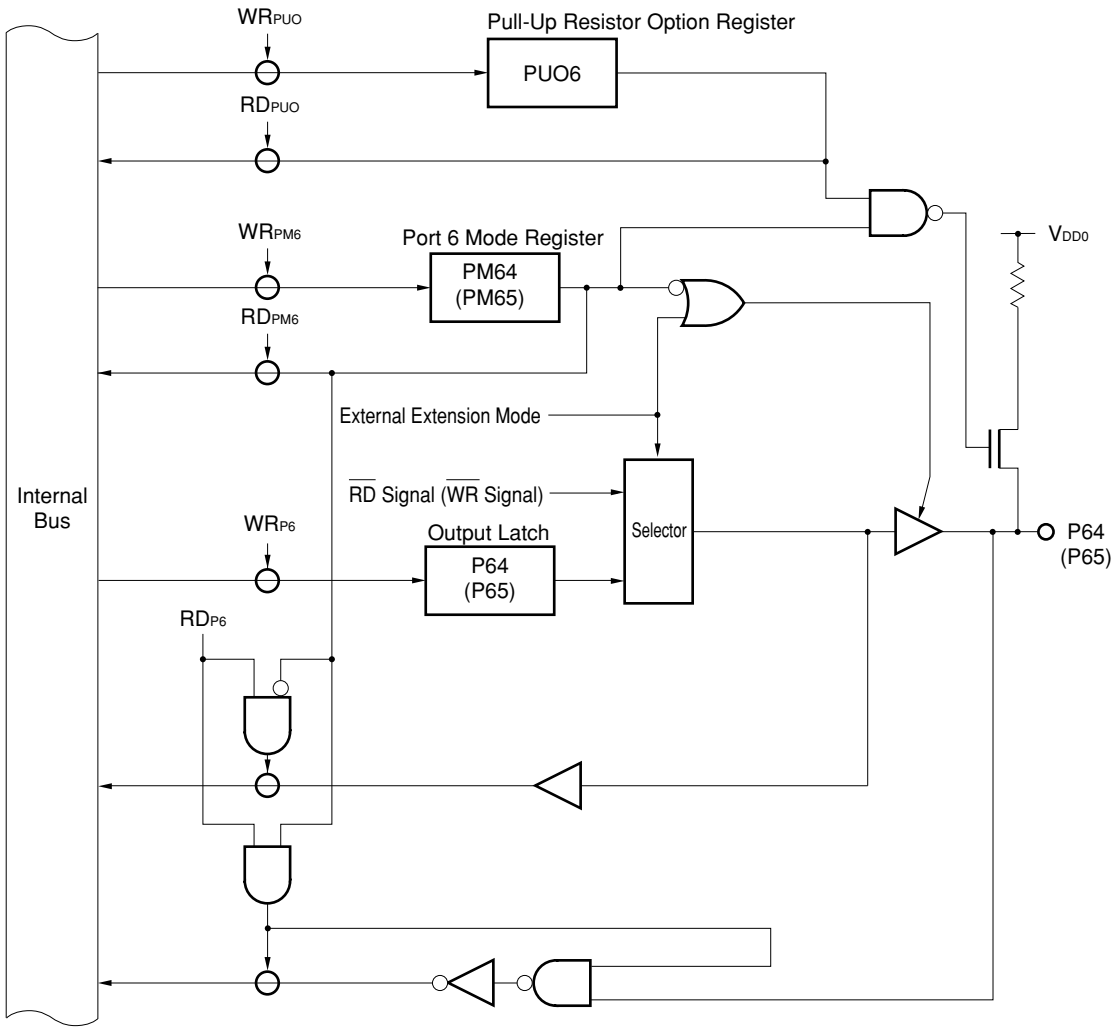


Figure 5-54 Block Diagram of P66 (Port 6)

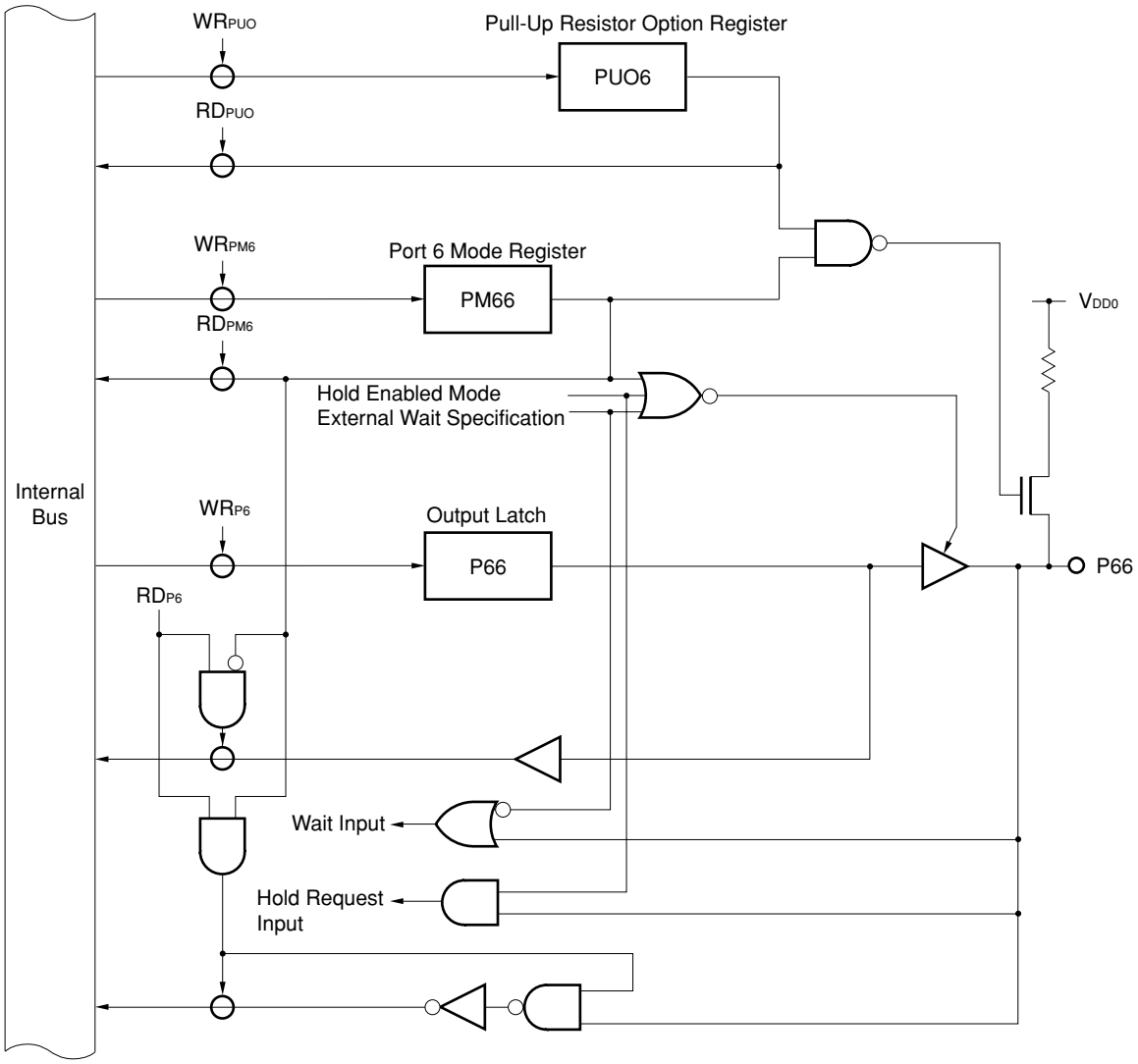
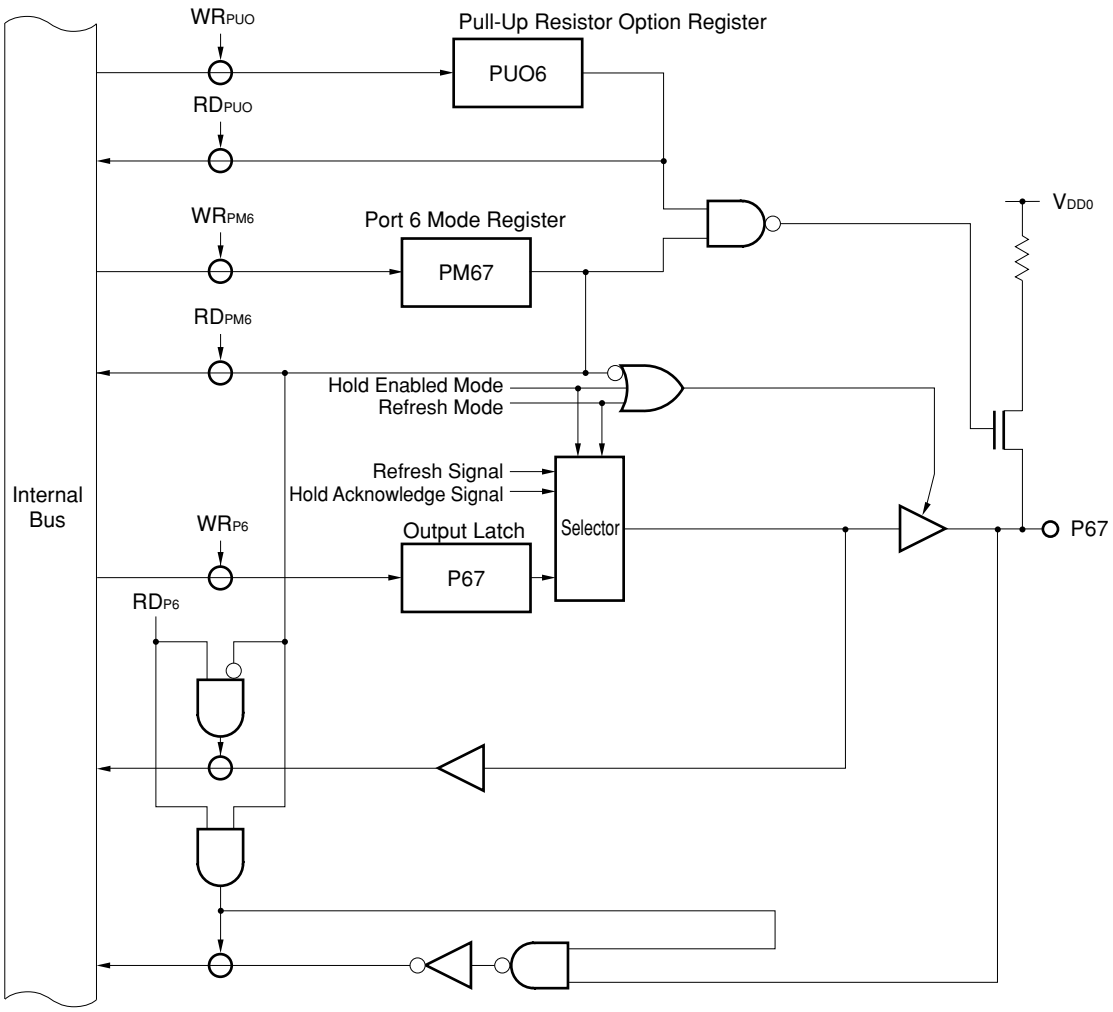


Figure 5-55 Block Diagram of P67 (Port 6)



5.8.2 I/O Mode/Control Mode Setting

The port 6 input/output mode is set by means of the port 6 mode register (PM6) as shown in Figure 5-56.

Operations for operating port 6 as control pins are shown in Table 5-13.

With the μPD784031, P64 functions only as $\overline{RD}$ signal output pin, and P65, as $\overline{WR}$ signal output pin.

Table 5-13 Port 6 Operating Modes

Pin Name	Control Signal I/O Mode	Port Mode	Operation to Operate as Control Pins
P60	A16	Input/output port Note	External memory extension mode specified by bits MM3 to MM0 of the MM (see **Table 5-14**)
P61	A17		
P62	A18		
P63	A19		
P64	$\overline{RD}$	Input/output port	With the μPD784031, external memory extension mode specified by bits MM3 to MM0 of the MM (see **Table 5-14**)
P65	$\overline{WR}$		
P66	$\overline{WAIT}$		External wait input is specified by setting bits PWn1 and PWn0 (n = 0 to 7) and P66 of the PWC1 and PWC2
	HLDRQ		Bus hold enabled by the HLDE bit of the HLDM
P67	HLDAK		
	$\overline{REFRQ}$		Set (to 1) the RFEN bit of the RFM

Note These pins of the μPD784031 are output port pins.

Table 5-14 P60 to P65 Control Pin Specification

MM Bits				Operating Mode					
MM3	MM2	MM1	MM0	P60	P61	P62	P63	P64	P65
0	0	0	0	Port (P60 to P65)					
0	0	1	1						
0	1	0	0	Port (P60 to P63)				$\overline{RD}$	$\overline{WR}$
0	1	0	1						
0	1	1	0						
0	1	1	1						
1	0	0	0	A16	A17	Port			
1	0	0	1	A16	A17	A18	A19		

Figure 5-56 Port 6 Mode Register (PM6) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM6	PM67	PM66	PM65	PM64	PM63	PM62	PM61	PM60	0FF26H	FFH	R/W

PM6n	P6n Pin Input/Output Mode Specification (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Remark The lower 4 bits (P60 to P63) of the μPD784031 are output port pins.

5.8.3 Operating Status

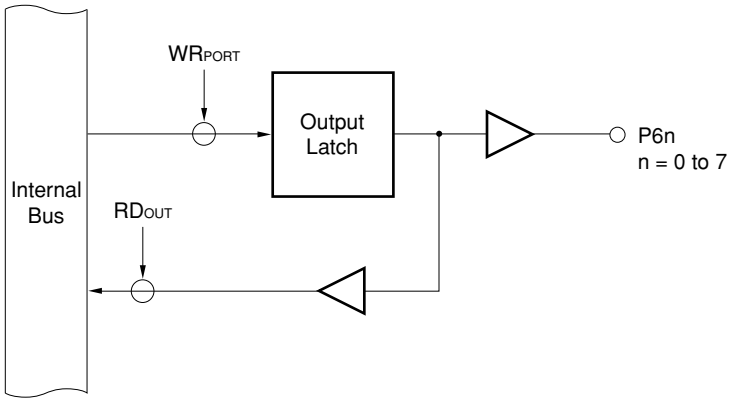
Port 6 is an input/output port, with an alternate function as various control pins.

(1) When set as an output port

The output latch is enabled, and data transfers between the output latch and accumulator are performed by means of transfer instructions. The output latch contents can be freely set by means of logical operation instructions. Once data has been written to the output latch, it is retained until data is next written to the output latch **Note**.

Note Including the case where another bit of the same port is manipulated by a bit manipulation instruction.

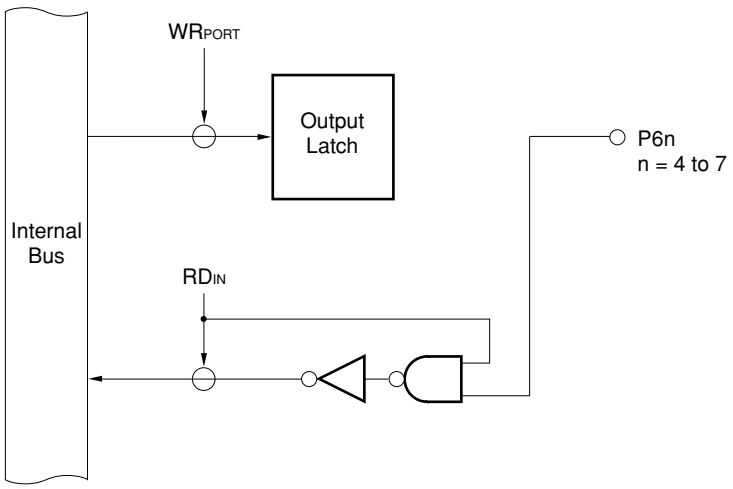
Figure 5-57 Port Specified as Output Port



(2) When set as an input port

The port pin level can be loaded into an accumulator by means of a transfer instruction. In this case, too, writes can be performed to the output latch, and data transferred from the accumulator by a transfer instruction, etc., is stored in all output latches irrespective of the port input/output specification. However, since the output buffer of a bit specified as an input port is high-impedance, the data is not output to the port pin (when a bit specified as input is switched to an output port, the output latch contents are output to the port pin). Also, the contents of the output latch of a bit specified as an input port cannot be loaded into an accumulator.

Figure 5-58 Port Specified as Input Port



Caution A bit manipulation instruction manipulates one bit as the result, but accesses the port in 8-bit units. Therefore, if a bit manipulation instruction is used on a port with a mixture of input and output pins, or port mode and control mode, the contents of the output latch of pins specified as inputs or pins specified as in the control mode will be undefined (excluding bits manipulated with a SET1 or CLR1 instruction, etc.). Particular care is required when there are bits which are switched between input and output.

Caution is also required when manipulating the port with other 8-bit manipulation instructions.

(3) When used as control pins

Cannot be manipulated or tested by software.

5.8.4 Internal Pull-Up Resistors

P60 to P67 (P64 to P67 with the μPD784031) incorporate pull-up resistors. Use of these internal resistors when pull-up is necessary enables the number of parts and the mounting area to be reduced.

Whether or not an internal pull-up resistor is to be used can be specified for each pin by means of the PUO6 bit of the pull-up resistor option register (PUO) and the port 6 mode register (PM6).

When PUO6 is 1, the internal pull-up resistors of the pins for which input is specified by the PM6 (PM6n = 1, n = 0 to 7) are enabled .

P60 to P63 of the μPD784031 are not connected to a pull-up resistor.

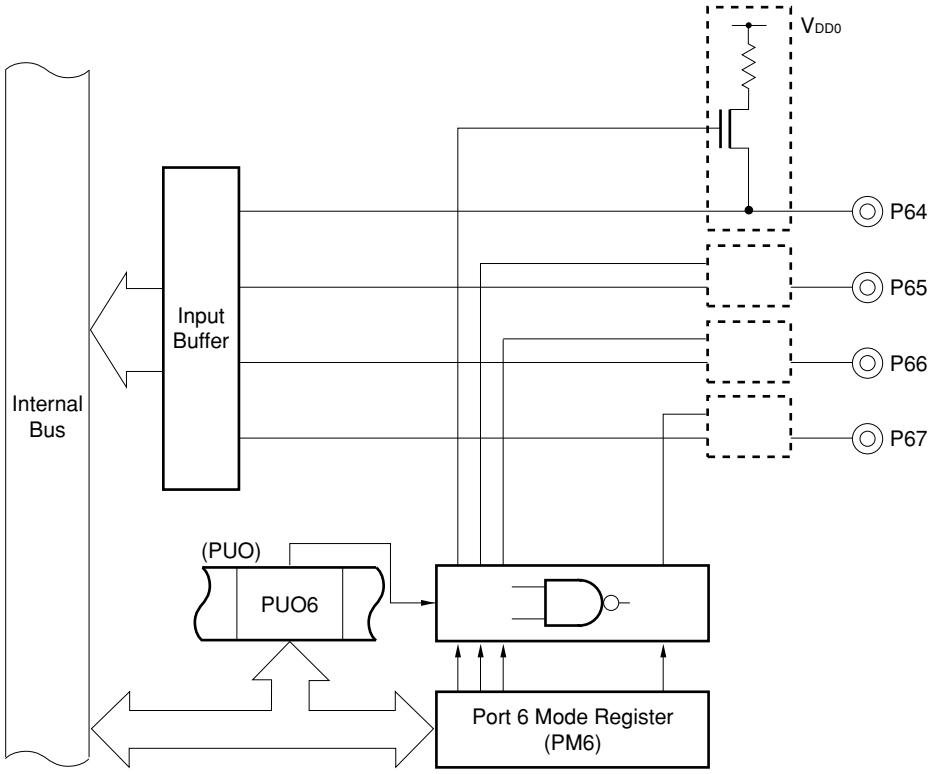
Figure 5-59 Pull-Up Resistor Option Register (PUO) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PUO	0	PUO6	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	0FF4EH	00H	R/W

PUO6	Port 6 Pull-Up Resistor Specification
0	Not used in port 6
1	Used in port 6

Remark When STOP mode is entered, setting 00H in PUO is effective in reducing the power consumption.

Figure 5-60 Pull-Up Specification (Port 6)



5.9 PORT 7

Port 7 is an 8-bit input/output port. In addition to operating as an input/output port, it also operates as the A/D converter analog input pins (ANI0 to ANI7).

Input/output can be specified bit-wise by means of the port 7 mode register (PM7).

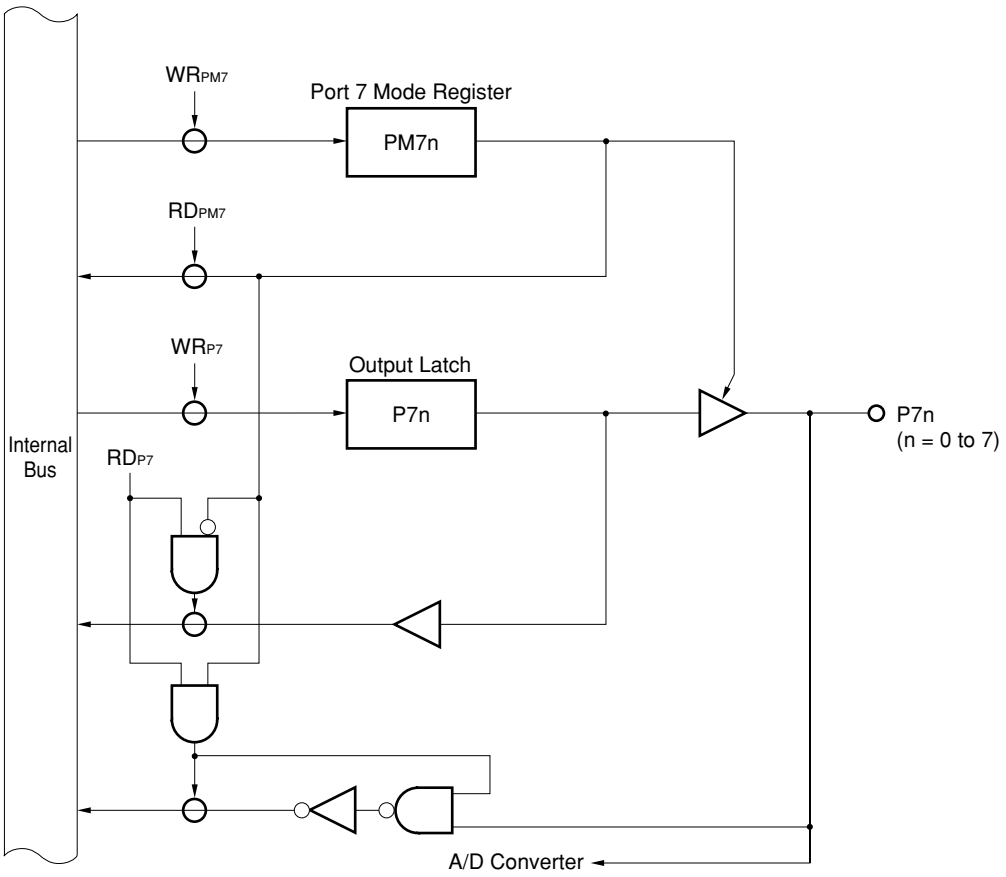
Pin levels can be read or tested at any time irrespective of alternate-function pin operations.

When $\overline{\text{RESET}}$ is input, port 7 is set as an input port (output high-impedance state), and the output latch contents are undefined.

5.9.1 Hardware Configuration

The port 7 hardware configuration is shown in Figure 5-61.

Figure 5-61 Port 7 Block Diagram



5.9.2 I/O Mode/Control Mode Setting

The port 7 input/output mode is set for each pin by means of the port 7 mode register (PM7) as shown in Figure 5-62.

In addition to the operation of port 7 as an input/output port, analog signal input can be performed at any time. Mode setting is not necessary.

Specification of the A/D conversion operation is performed by ADM of the A/D converter (see **Chapter 14 A/D Converter** for details).

Figure 5-62 Port 7 Mode Register (PM7) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70	0FF27H	FFH	R/W

PM7n	P7n Pin Input/Output Mode Specification (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

5.9.3 Operating Status

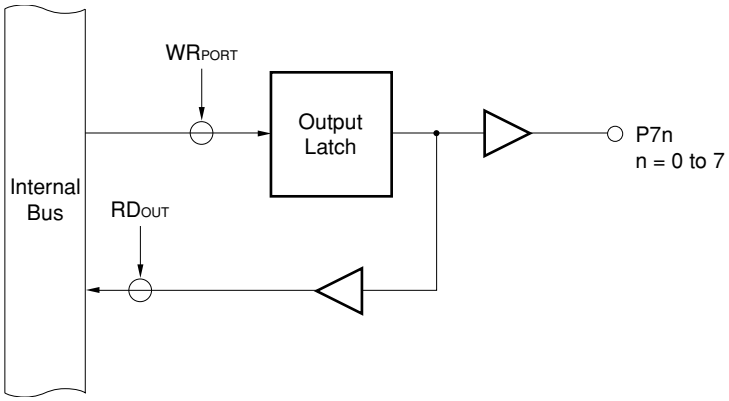
Port 7 is an input/output port, with an alternate function as the A/D converter analog input pins (ANI0 to ANI7).

(1) When set as an output port

The output latch is enabled, and data transfers between the output latch and accumulator are performed by means of transfer instructions. The output latch contents can be freely set by means of logical operation instructions. Once data has been written to the output latch, it is retained until data is next written to the output latch **Note**.

Note Including the case where another bit of the same port is manipulated by a bit manipulation instruction.

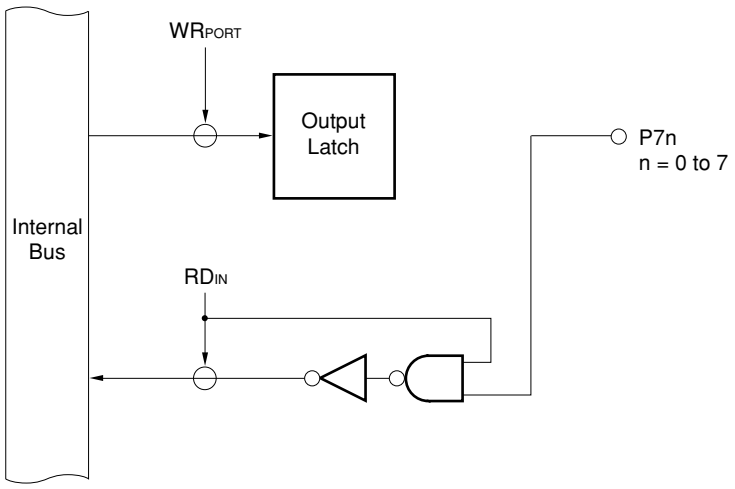
Figure 5-63 Port Specified as Output Port



(2) When set as an input port

The port pin level can be loaded into an accumulator by means of a transfer instruction. In this case, too, writes can be performed to the output latch, and data transferred from the accumulator by a transfer instruction, etc., is stored in all output latches-irrespective of the port input/output specification. However, since the output buffer of a bit specified as an input port is high-impedance, the data is not output to the port pin (when a bit specified as input is switched to an output port, the output latch contents are output to the port pin). Also, the contents of the output latch of a bit specified as an input port cannot be loaded into an accumulator.

Figure 5-64 Port Specified as Input Port



Caution A bit manipulation instruction manipulates one bit as the result, but accesses the port in 8-bit units. Therefore, if a bit manipulation instruction is used on a port with a mixture of input and output pins, the contents of the output latch of pins specified as inputs will be undefined (excluding bits manipulated with a SET1 or CLR1 instruction, etc.). Particular care is required when there are bits which are switched between input and output.

Caution is also required when manipulating the port with other 8-bit operation instructions.

5.9.4 Internal Pull-Up Resistors

Port 7 does not incorporate pull-up resistors.

5.9.5 Caution

A voltage outside the range AV_{SS} to AV_{REF1} must not be applied to pins for which P70 to P77 are used as ANI0 to AN17. See **14.5 CAUTIONS** in **CHAPTER 14 "A/D CONVERTER"** for details.

5.10 PORT OUTPUT CHECK FUNCTION

The μPD784038 has a function for reading and testing output port pin levels in order to improve the reliability of application systems. It is therefore possible to check the output data and the actual pin status as required. If there is a mismatch, appropriate action can be taken, such as replacement with another system.

Special instructions, CHKL and CHKLA, are provided to check the port status. These instructions perform a comparison by taking the exclusive OR of the pin status and the output latch contents (in port mode), or the pin status and the internal control output signal level (in control mode).

Example An example is shown below of a program that checks the pin status and output latch contents using the CHKL instruction and CHKLA instruction.

```
TEST  :   SET1    P0.3          ; Set bit 3 of port 0
          CHKL    P0            ; Check port 0
          BNE     $ ERR1        ; Branch to error processing (ERR1) in case of mismatch with output
                                  latch contents
              .
              .
              .
ERR1  :   CHKLA   P0            ; Faulty bit check
          BT      A.7, $BIT07   ; Bit 7?
          BT      A.6, $BIT06   ; Bit 6?
              .
              .
              .
          BT      A.1, $BIT01   ; Bit 1?
          BR      $BIT00        ; If none of the bits, bit 0 is faulty
```

Cautions

1. **If each port is set to input mode, a comparison of the pin status with the output latch contents (or control output level) using the CHKL or CHKLA instruction will always show a match whether the individual pins of the port are port pins or control pins.**
   Therefore, executing these instructions on a port set to input mode is actually ineffective.
2. **If the output levels of a port in which control outputs and port outputs are mixed in a single port are checked with the CHKL or CHKLA instruction, the input/output mode of control output pins should be set to input mode before executing these instructions (as the output levels of control outputs vary asynchronously, the output level cannot be checked with the CHKL or CHKLA instruction).**
3. **As port 2 is an input-only port, a comparison of the pin status with the output latch contents using the CHKL or CHKLA instruction will always show a match. Therefore, executing these instructions on port 2 is actually ineffective.**

5.11 CAUTIONS

(1) All port pins become high-impedance after $\overline{\text{RESET}}$ signal input (internal pull-up resistors are disconnected from the pins).
If there is a problem with pins becoming high-impedance during $\overline{\text{RESET}}$ input, this should be handled with external circuitry.

(2) Bit 7 of the pull-up resistor option register (PUO) that sets the internal pull-up resistor connection is fixed at 0, but if "1" is written to bit 7 of the PUO in the in-circuit emulator, "1" will be read.

(3) Output latch contents are not initialized by $\overline{\text{RESET}}$ input. When a port is used as an output port, the output latch must be initialized without fail before turning on the output buffer. If the output latch is not initialized before turning on the output buffer, unexpected data will be output to the output port.
Similarly, for pins used as control pins, internal peripheral hardware initialization must be performed before performing the control pin specification.

(4) As P22 to P26 are not pulled up immediately after a reset, an interrupt request flag may be set depending on the function of the alternate-function pins (INTP1 to INTP5). Therefore, the interrupt request flags should be cleared after specifying pull-up in the initialization routine.

(5) When P40 to P47 and P50 to P57 are used as the address/data bus and address bus respectively in the μPD784038, and with the μPD784038 bits PUO4 and PUO5 of the pull-up resistor option register (PUO) must be set to "0" so that internal pull-up resistor connection is not performed.

(6) P60 to P63 of the μPD784031 are in the output high-impedance state while the $\overline{\text{RESET}}$ signal is input, but output a low level after the $\overline{\text{RESET}}$ signal has been cleared. Therefore, design the external circuit so that the low level may be output as the initial status.

(7) A voltage outside the range AV_{SS} to AV_{REF1} must not be applied to pins for which P70 to P77 are used as ANI0 to ANI7. See **14.5 CAUTIONS** in **CHAPTER 14 A/D CONVERTER** for details.

(8) A bit manipulation instruction manipulates one bit as the result, but accesses the port in 8-bit units. Therefore, if a bit manipulation instruction is used on a port with a mixture of input and output pins or port mode and control mode, the contents of the output latch of pins specified as inputs or pins specified as in control mode will be undefined (excluding bits manipulated with a SET1 or CLR1 instruction, etc.). Particular care is required when there are bits which are switched between input and output.
Caution is also required when manipulating the port with other 8-bit operation instructions.

(9) If each port is set to input mode, a comparison of the pin status with the output latch contents (or control output level) using the CHKL or CHKLA instruction will always show a match whether the individual pins of the port are port pins or control pins. Therefore, executing these instructions on a port set to input mode is actually ineffective.

(10) If the output levels of a port in which control outputs and port outputs are mixed in a single port are checked with the CHKL or CHKLA instruction, the input/output mode of control output pins should be set to input mode before executing these instructions (as the output levels of control outputs vary asynchronously, the output level cannot be checked with the CHKL or CHKLA instruction).

(11) As port 2 is an input-only port, a comparison of the pin status with the output latch contents using the CHKL or CHKLA instruction will always show a match. Therefore, executing these instructions on port 2 is actually ineffective.

CHAPTER 6 REAL-TIME OUTPUT FUNCTION

6.1 CONFIGURATION AND FUNCTION

The real-time output function is implemented by hardware, including primarily port 0 and the port 0 buffer registers (P0H, P0L), shown in Figure 6-1.

The real-time output function refers to the transfer to the output latch by hardware of data prepared in the P0H and P0L beforehand, simultaneously with the generation of an interrupt from timer/counter 1 or external interrupt, and its output off-chip. The pins that output the data off-chip are called real-time output ports.

The following two kinds of real-time output data are handled:

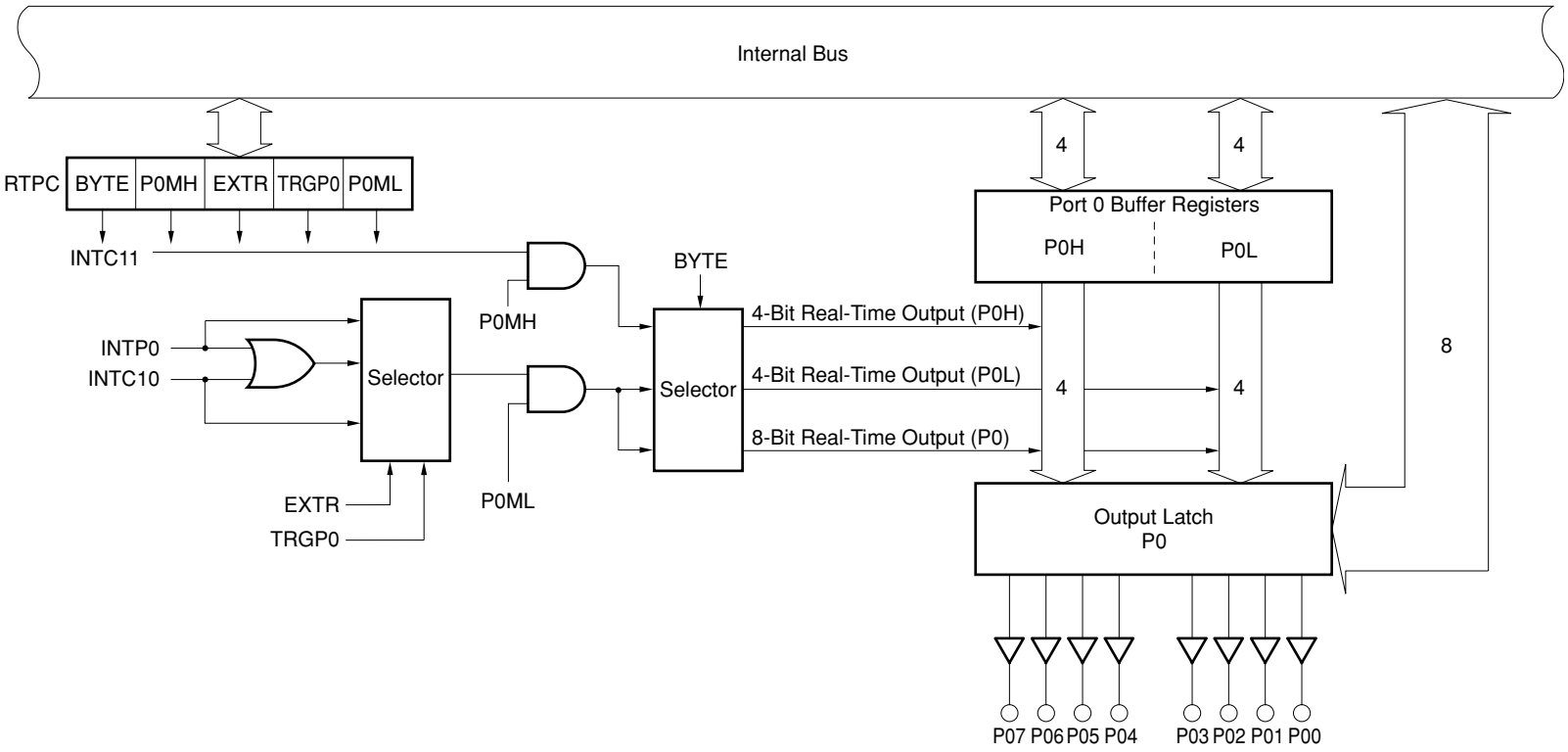
- 4 bits × 2 channels
- 8 bits × 1 channel

By combining the real-time output function with the macro service function described later, the functions of a pattern generator with programmable timing are implemented without software intermediation.

This is ideally suited to stepping motor control, for example.

Figure 6-1 shows the block diagram of the real-time output port.

Figure 6-1 Real-Time Output Port Block Diagram



6.2 REAL-TIME OUTPUT PORT CONTROL REGISTER (RTPC)

The RTPC is an 8-bit register that specifies the function of port 0.

RTPC can be read or written to by an 8-bit manipulation instruction or bit-manipulation instruction. Figure 6-2 shows the format of RTPC.

$\overline{\text{RESET}}$ input clears the RTPC register to 00H.

Figure 6-2 Real-Time Output Port Control Register (RTPC) Format

	7	6	5	④	3	2	1	⓪	Address	After Reset	R/W
RTPC	BYTE	0	0	P0MH	EXTR	0	TRGP0	P0ML	0FF2EH	00H	R/W

P0ML	P00 to P03 Function Specification
0	Port mode
1	Real-time output port mode

EXTR	TRGP0	Enabling of Data Transfer to Output Latch from P0H, P0L by INTP0	
0	0	Not enabled (data transfer by INTC10 only)	
1	0	Transfer by either INTP0 or INTC10	Enabled • BYTE = 0 : P0L only transferred • BYTE = 1 : P0L/P0H transferred
1	1	Transfer by INTP0 only	
0	1	Setting prohibited	

P0MH	P04 to P07 Function Specification
0	Port mode
1	Real-time output port mode

BYTE	Real-Time Output Port Operating Mode
0	4-bit separate real-time output ports
1	8-bit real-time output port

Caution When P0ML and P0MH bits are set (to 1), the corresponding port output buffer is turned on and the port 0 output latch contents are output irrespective of the contents of the port 0 mode register (PM0). The output latch contents should therefore be initialized before making a real-time output port specification.

6.3 REAL-TIME OUTPUT PORT ACCESSES

The port 0 buffer registers (P0H, P0L) are mapped onto mutually independent addresses in the SFR area as shown in Figure 6-3.

When the 4-bit × 2-channel real-time output function is specified, data can be set in the P0H, P0L independently of each other.

When the 8-bit ×1-channel real-time output function is specified, data can be set in P0H and P0L by writing 8-bit data to either one of the P0H or P0L.

Table 6-1 shows the operations when port 0, the P0H and P0L are manipulated.

Figure 6-3 Port 0 Buffer Register (P0H, P0L) Configuration

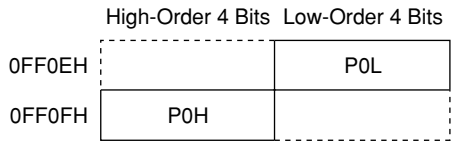


Table 6-1 Operations When Port 0 and Port 0 Buffer Registers (P0H, P0L) are Manipulated

Operating Mode	Register	Read Operation		Write Operation	
		High-Order 4 Bits	Low-Order 4 Bits	High-Order 4 Bits	Low-Order 4 Bits
8-bit port mode	P0	Output latch		Output latch	
	P0L	Buffer register Note		—	Buffer register
	P0H	Buffer register Note		Buffer register	—
8-bit real-time output port mode	P0	Output latch		—	
	P0L	Buffer register		Buffer register	
	P0H	Buffer register		Buffer register	
4-bit separate real-time output port mode	P0	Output latch		—	
	P0L	Buffer register Note		—	Buffer register
	P0H	Buffer register Note		Buffer register	—
P00 to P03: Ports P04 to P07: Real-time output port mode	P0	Output latch		—	Output latch
	P0L	Buffer register Note		—	Buffer register
	P0H	Buffer register Note		Buffer register	—
P00 to P03: Real-time output port mode P04 to P07: Ports	P0	Output latch		Output latch	—
	P0L	Buffer register Note		—	Buffer register
	P0H	Buffer register Note		Buffer register	—

Note The contents of P0H are read from the high-order 4 bits, and the contents of P0L from the low-order 4 bits.

Remark — : The output latch and port 0 buffer registers are not affected.

<Examples of setting data in port 0 buffer registers>

- 4-bit × 2-channel operation

```
MOV   P0L,   #05H     ; Sets 0101B in P0L
MOV   P0H,   #0C0H    ; Sets 1100B in P0H
```

- 8-bit × 1-channel operation

```
MOV   P0L,   #0C5H    ; Sets 0101B in P0L and 1100B in P0H
```

or

```
MOV   P0H,   #0C5H
```

The timing for transfer to the output latch can be determined by the following three sources:

- Interrupt from timer/counter 1 (INTC10 or INTC11)
- INTP0 external interrupt

6.4 OPERATION

When the port 0 function is specified as the real-time output port, the port 0 buffer register (P0H, P0L) contents are fetched into the output latch and output to the port 0 pins in synchronization with the generation of one of the trigger conditions shown in Table 6-2.

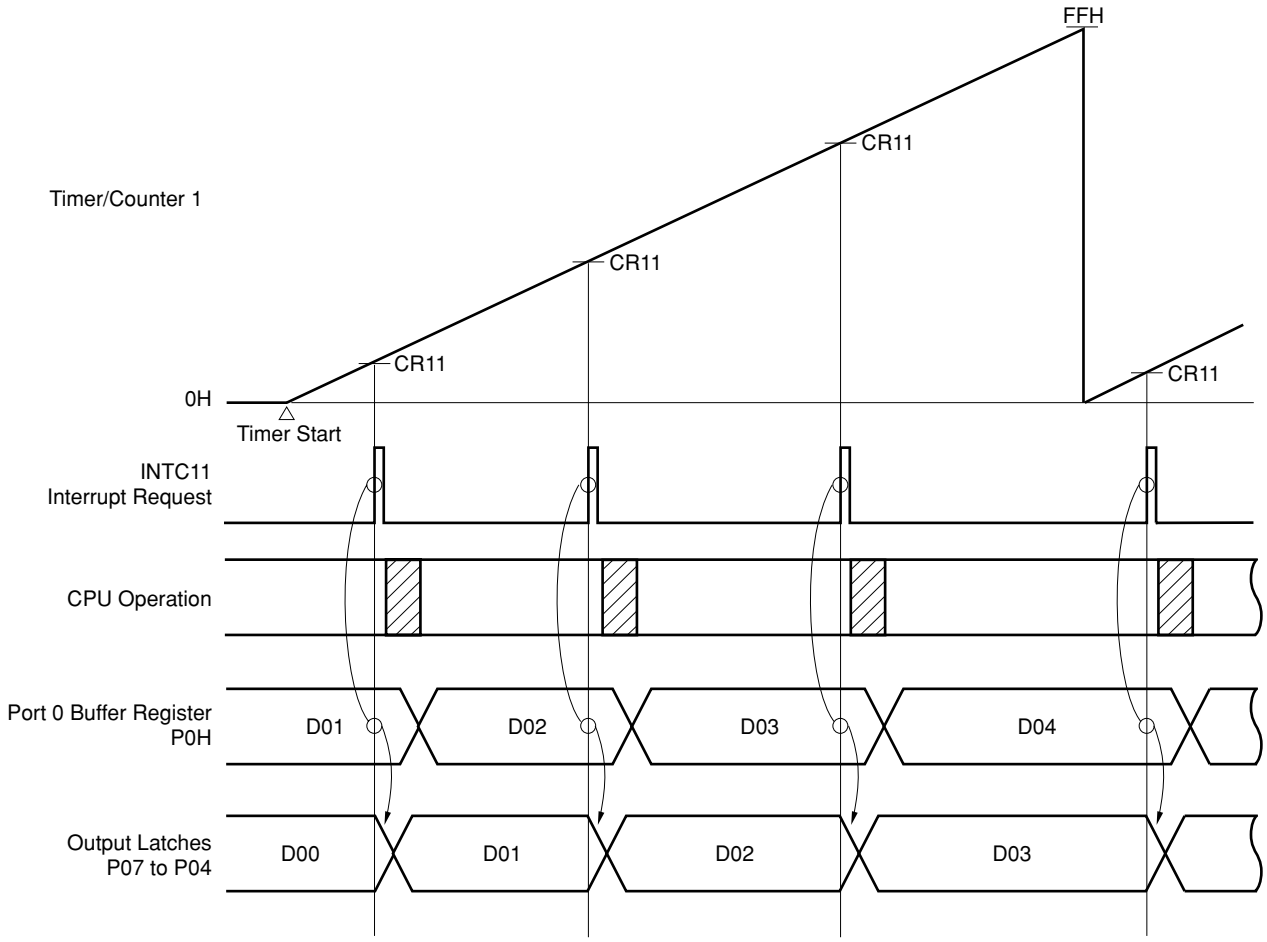
For example, the timer/counter 1 timer register 1 (TM1) and compare register (CR10, CR11) match signal (INTC10, INTC11) can be selected as the output trigger generation source. In this case, the port 0 pin output data can be changed to the P0H and P0L values using the value set in the CR10, CR11 beforehand as the timing interval. Combining this real-time output port function with the macro service function enables the port 0 output pin output data to be changed sequentially at any interval time (see **22.8 Macro Service Function**).

If the INTP0 external interrupt pin is selected as the output trigger source, port 0 output can be obtained in synchronization with an external event.

Table 6-2 Real-Time Output Port Output Triggers (When P0MH = P0ML = 1)

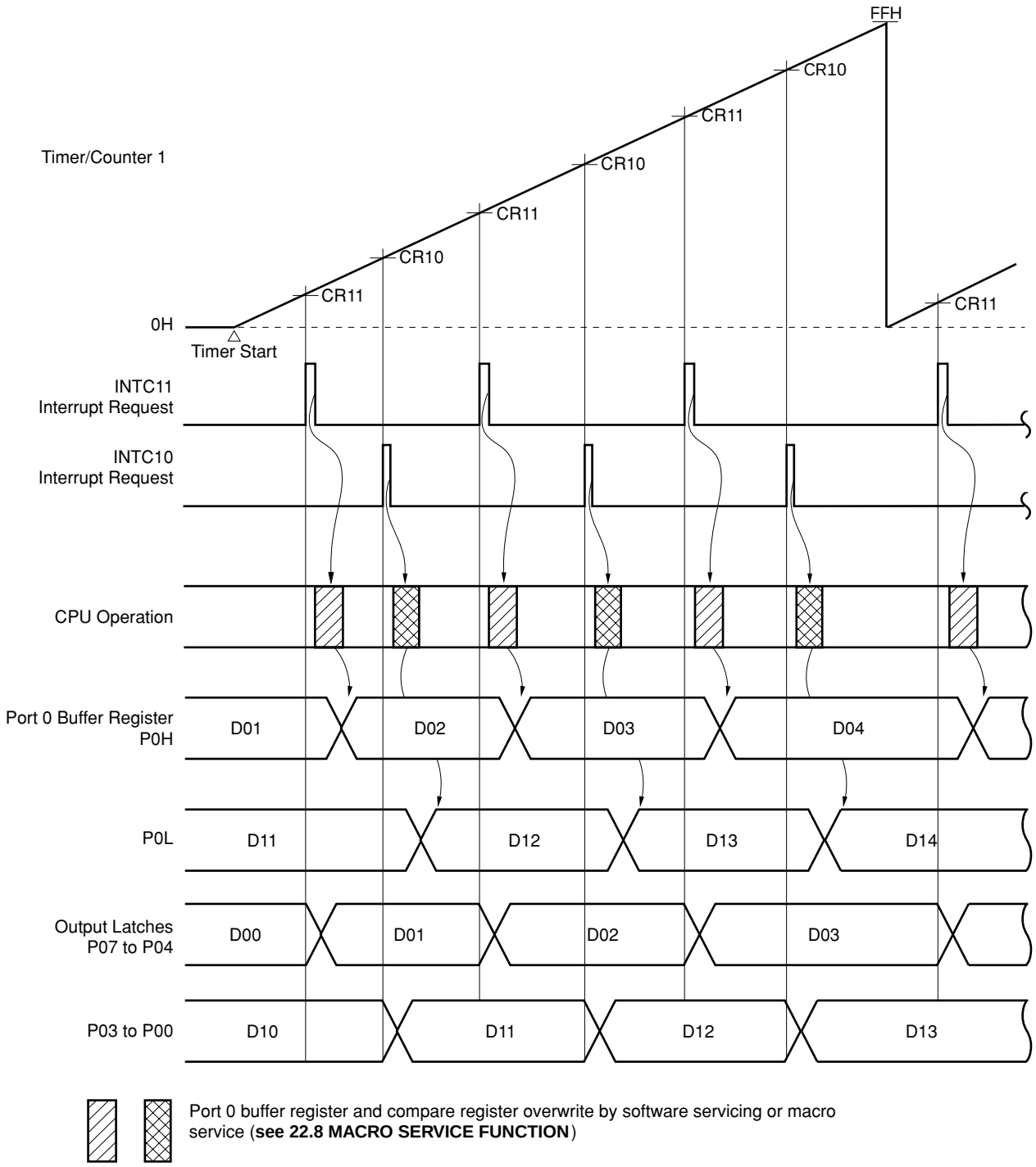
RTPC			Output Mode	P0H	P0L
BYTE	EXTR	TRGP0			
0	0	0	4-bit real-time output	INTC11	INTC10
0	1	0		INTC11	INTC10 or INTP0
0	1	1		INTC11	INTP0
1	0	0	8-bit real-time output	INTC10	
1	1	0		INTC10 or INTP0	
1	1	1		INTP0	

Figure 6-4 Real-Time Output Port Operation Timing



Port 0 buffer register and compare register overwrite by software servicing or macro service (see **22.8 MACRO SERVICE FUNCTION**)

Figure 6-5 Real-Time Output Port Operation Timing (2-Channel Independent Control Example)



6.5 EXAMPLE OF USE

The case in which P00 to P03 are used as a 4-bit real-time output port is shown here.

Each time the contents of timer/counter 1 timer register 1 (TM1) and compare register (CR10) match, the contents of port 0 buffer register (P0L) are output to P00 to P03. At this time, the next data to be output and the timing at which the output is to be changed next are set in the service routine for the simultaneously generated interrupt (see **Figure 6-6**).

See **CHAPTER 9 TIMER/COUNTER 1** for the method of using timer/counter 1.

The control register settings are shown in Figure 6-7, the setting procedure in Figure 6-8, and the processing in the interrupt service routine in Figure 6-9.

Figure 6-6 Real-Time Output Port Operation Timing

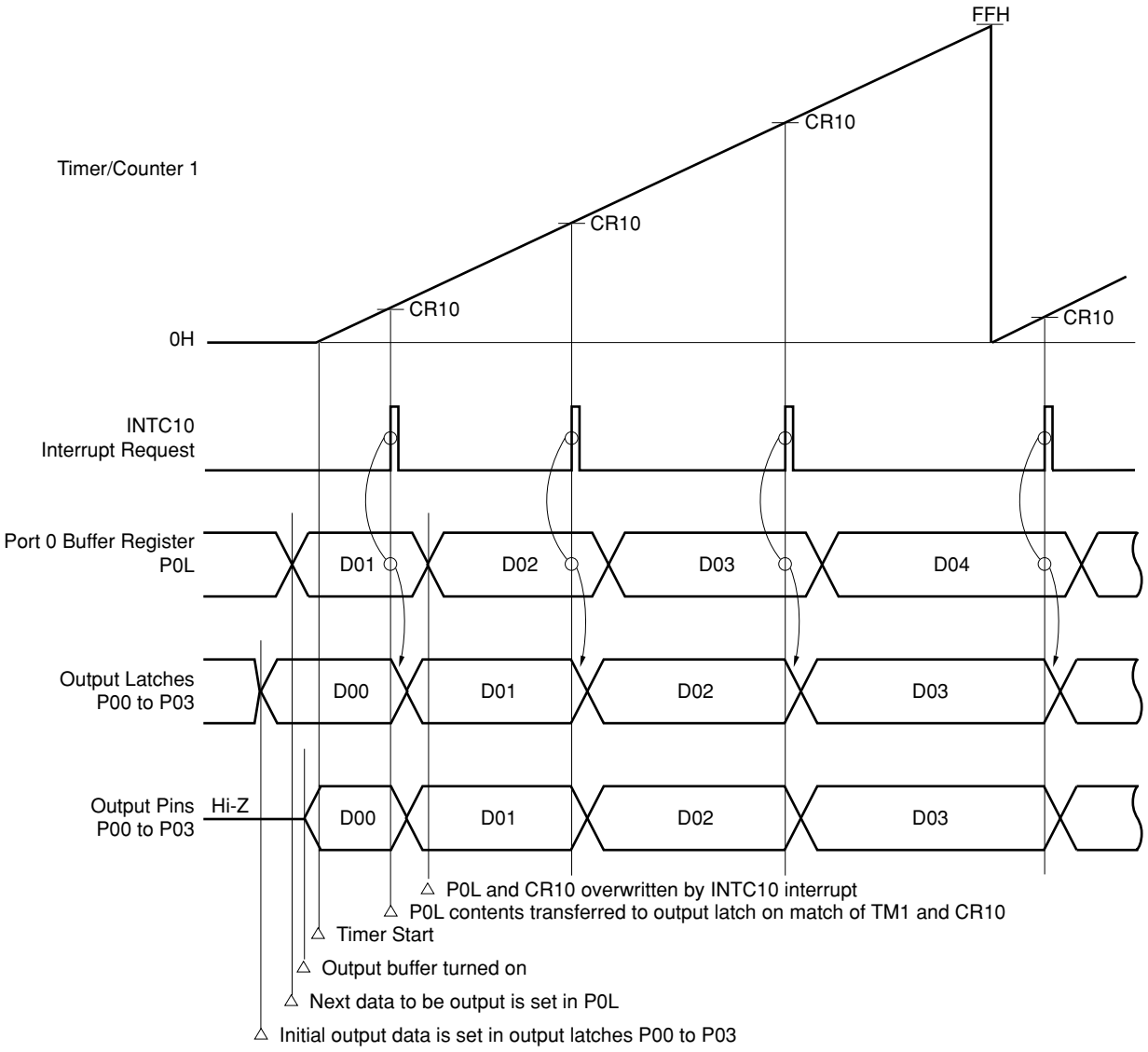


Figure 6-7 Real-Time Output Function Control Register Settings

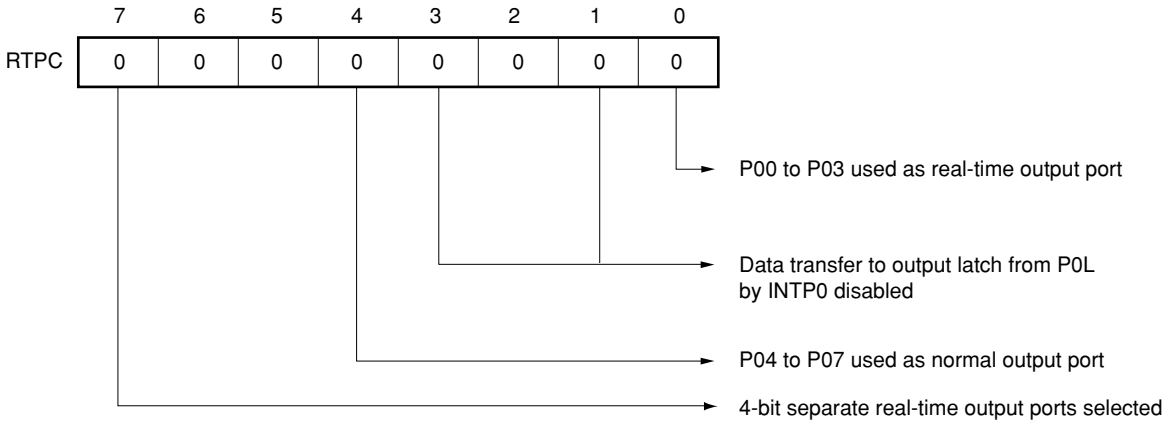


Figure 6-8 Real-Time Output Function Setting Procedure

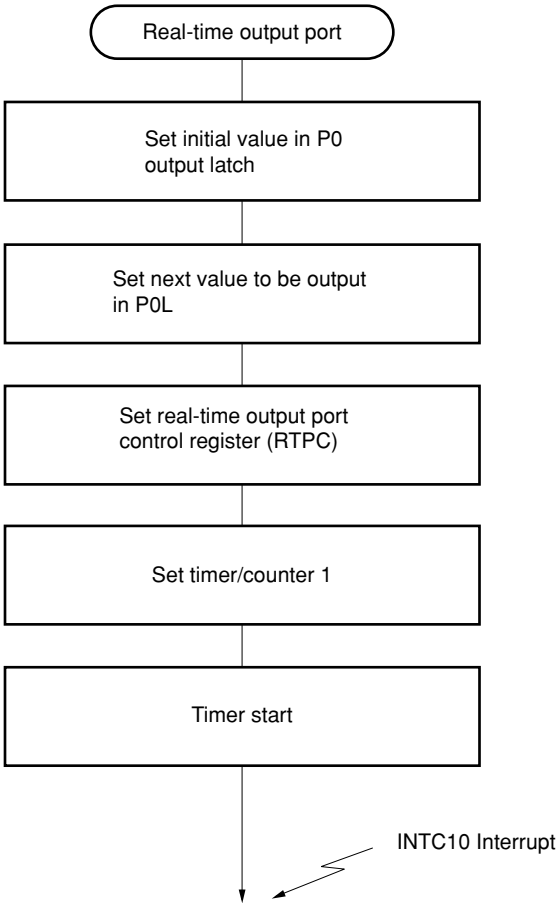
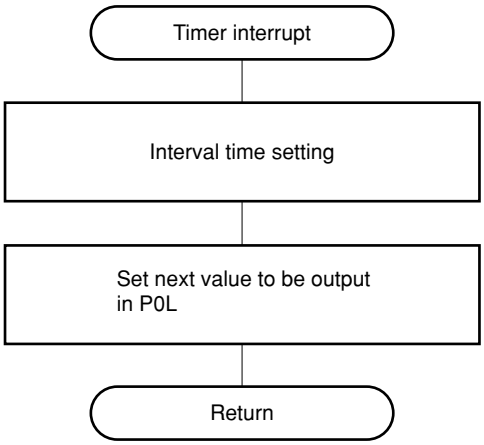


Figure 6-9 Interrupt Request Servicing when Real-Time Output Function is Used



6.6 CAUTIONS

(1) When P0ML and P0MH bits are set (to 1), the corresponding port output buffer is turned on and the port 0 output latch contents are output irrespective of the contents of the port 0 mode register (PM0). The output latch contents should therefore be initialized before making a real-time output port specification.

(2) When the port is specified as a real-time output port, values cannot be directly written to the output latch by software. Therefore, the initial value of the output latch must be set by software before specifying use as a real-time output port. Also, if the need arises to forcibly set the output data to a fixed value while the port is being used as a real-time output port, you should change the port to a normal output port by manipulating the real-time output port control register (RTPC), then write the value to be output to the output latch.

CHAPTER 7 OUTLINE OF TIMER/COUNTER

The μPD784038 incorporates three timer/counter units and one timer unit.

These timer/counter and timer units can be used as seven units of timer/counters because the μPD784038 supports seven interrupt requests.

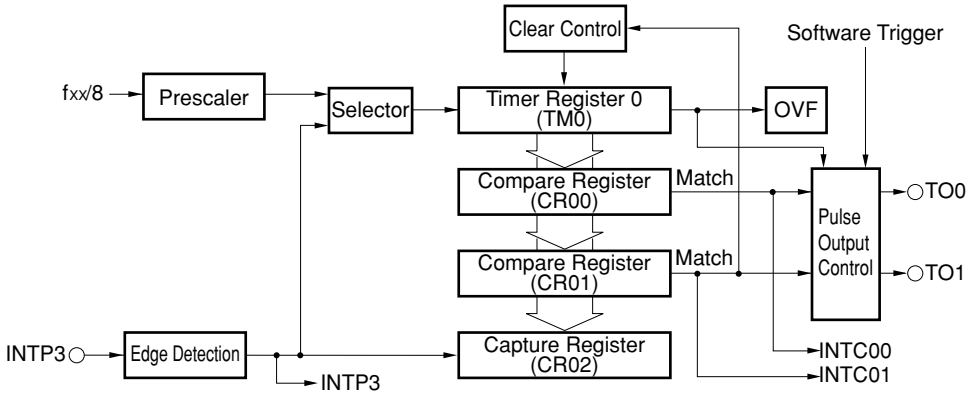
Table 7-1 Operations of Timer/Counters

Item \ Name		Timer/Counter 0	Timer/Counter 1	Timer/Counter 2	Timer 3
Count width	8 bits	—	√	√	√
	16 bits	√	√	√	√
Operation mode	Interval timer	2 ch	2 ch	2 ch	1 ch
	External event counter	√	√	√	—
	One-shot timer	—	—	√	—
Function	Timer output	2 ch	—	2 ch	—
	Toggle output	√	—	√	—
	PWM/PPG output	√	—	√	—
	One-shot pulse output Note	√	—	—	—
	Real-time output	—	√	—	—
	Pulse width measurement	1 input	1 input	2 inputs	—
	Number of interrupt requests	2	2	2	1

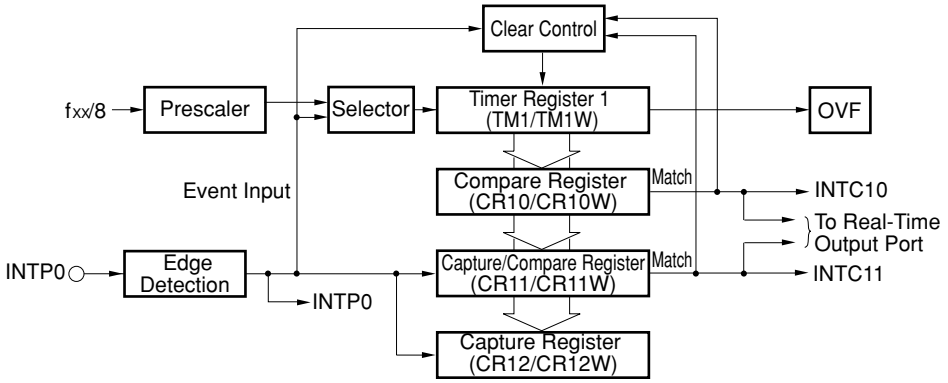
Note In the one-shot pulse output function, the pulse output level activated by software and inactivated by hardware (an interrupt request signal).
This function is different in nature from the one-shot timer function of timer/counter 2.

Figure 7-1 Timer/Counter Block Diagram

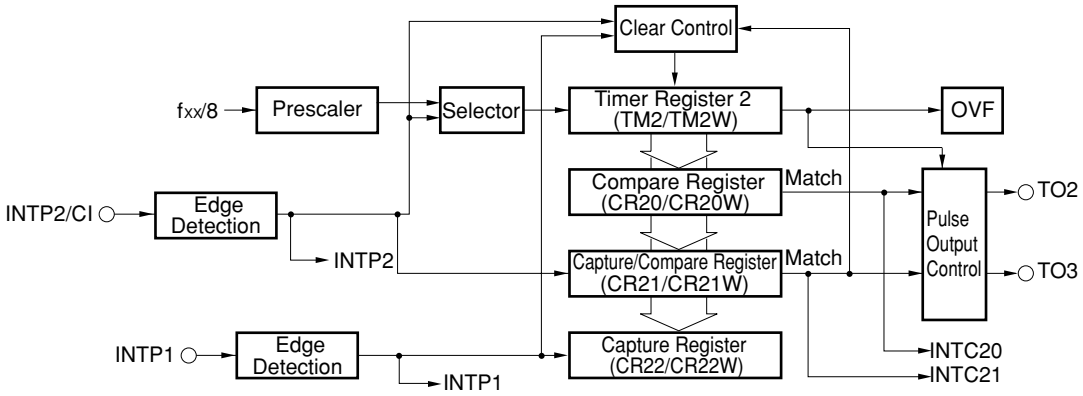
Timer/Counter 0



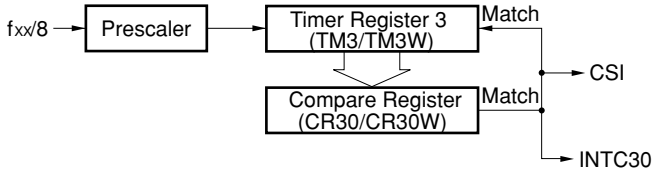
Timer/Counter 1



Timer/Counter 2



Timer 3



Remark OVF: Overflow flag

CHAPTER 8 TIMER/COUNTER 0

8.1 FUNCTIONS

Timer/counter 0 is a 16-bit timer/counter.

In addition to its basic functions of interval timer, programmable square-wave output, pulse width measurement and event counter, timer/counter 0 can be used for the following functions.

- PWM output
- Cycle measurement
- Soft triggered one-shot pulse output

(1) Interval timer

Generates internal interrupts at preset intervals.

Table 8-1 Timer/Counter 0 Interval Time

Minimum Interval Time	Maximum Interval Time	Resolution
8/fxx (0.25 μs)	$2^{16} \times 8$/fxx (16.40 ms)	8/fxx (0.25 μs)
16/fxx (0.50 μs)	$2^{16} \times 16$/fxx (32.80 ms)	16/fxx (0.50 μs)
32/fxx (1.00 μs)	$2^{16} \times 32$/fxx (65.50 ms)	32/fxx (1.00 μs)
64/fxx (2.00 μs)	$2^{16} \times 64$/fxx (131 ms)	64/fxx (2.00 μs)
128/fxx (4.00 μs)	$2^{16} \times 128$/fxx (262 ms)	128/fxx (4.00 μs)
256/fxx (8.00 μs)	$2^{16} \times 256$/fxx (524 ms)	256/fxx (8.00 μs)
512/fxx (16.00 μs)	$2^{16} \times 512$/fxx (1.05 s)	512/fxx (16.00 μs)
1,024/fxx (32.00 μs)	$2^{16} \times 1{,}024$/fxx (2.10 s)	1,024/fxx (32.05 μs)
2,048/fxx (64.00 μs)	$2^{16} \times 2{,}048$/fxx (4.19 s)	2,048/fxx (64.00 μs)

( ): When fxx = 32 MHz

(2) Programmable square-wave output

Outputs square waves independently to the timer output pins (TO0, TO1).

Table 8-2 Timer/Counter 0 Programmable Square-Wave Output Setting Range

Minimum Pulse Width	Maximum Pulse Width
8/fxx (0.25 μs)	$2^{16} \times 8$/fxx (16.40 ms)
16/fxx (0.50 μs)	$2^{16} \times 16$/fxx (32.80 ms)
32/fxx (1.00 μs)	$2^{16} \times 32$/fxx (65.50 ms)
64/fxx (2.00 μs)	$2^{16} \times 64$/fxx (131 ms)
128/fxx (4.00 μs)	$2^{16} \times 128$/fxx (262 ms)
256/fxx (8.00 μs)	$2^{16} \times 256$/fxx (524 ms)
512/fxx (16.00 μs)	$2^{16} \times 512$/fxx (1.05 s)
1,024/fxx (32.00 μs)	$2^{16} \times 1{,}024$/fxx (2.10 s)
2,048/fxx (64.00 μs)	$2^{16} \times 2{,}048$/fxx (4.19 s)

( ): When fxx = 32 MHz

(3) Pulse width measurement

Detects the pulse width of the signal input to the external interrupt request input pin (INTP3).

Table 8-3 Timer/Counter 0 Pulse Width Measurement Range

Measurable Pulse Width [Note]	Resolution
8/fxx to $2^{16} \times 8$/fxx (0.25 μs) (16.40 ms)	8/fxx (0.25 μs)
16/fxx to $2^{16} \times 16$/fxx (0.50 μs) (32.80 ms)	16/fxx (0.50 μs)
32/fxx to $2^{16} \times 32$/fxx (1.00 μs) (65.50 ms)	32/fxx (1.00 μs)
64/fxx to $2^{16} \times 64$/fxx (2.00 μs) (131 ms)	64/fxx (2.00 μs)
128/fxx to $2^{16} \times 128$/fxx (4.00 μs) (262 ms)	128/fxx (4.00 μs)
256/fxx to $2^{16} \times 256$/fxx (8.00 μs) (524 ms)	256/fxx (8.00 μs)
512/fxx to $2^{16} \times 512$/fxx (16.00 μs) (1.05 s)	512/fxx (16.00 μs)
1,024/fxx to $2^{16} \times 1{,}024$/fxx (32.00 μs) (2.10 s)	1,024/fxx (32.00 μs)
2,048/fxx to $2^{16} \times 2{,}048$/fxx (64.00 μs) (4.19 s)	2,048/fxx (64.00 μs)

( ): When fxx = 32 MHz

Note The minimum pulse width that can be measured differs depending on the selected value of f_{CLK}.
The minimum pulse width that can be measured is the value of $4/f_{CLK}$ or the value in the above table, whichever is greater.

(4) Software triggered one-shot pulse output

This is a one-shot pulse output function in which the pulse output level is activated by software and inactivated by hardware (an interrupt request signal). Control can be performed for the timer output pins (TO0, TO1) independently.

Caution The software triggered one-shot pulse output function is different in nature from the one-shot timer function of timer/counter 2.

(5) External event counter

Counts the clock pulses input from the external interrupt request input pin (INTP3).

The clocks that can be input to timer/counter 0 are shown in Table 8-4.

Table 8-4 Timer/Counter 0 Pulse Width Measurement Time

	When Counting One Edge	When Counting Both Edges
Maximum frequency	$f_{CLK}/8$ (2.00 MHz)	$f_{CLK}/8$ (2.00 MHz)
Minimum pulse width (High and low levels)	$4/f_{CLK}$ (0.25 μs)	$4/f_{CLK}$ (0.25 μs)

( ): When f_{CLK} = 16 MHz

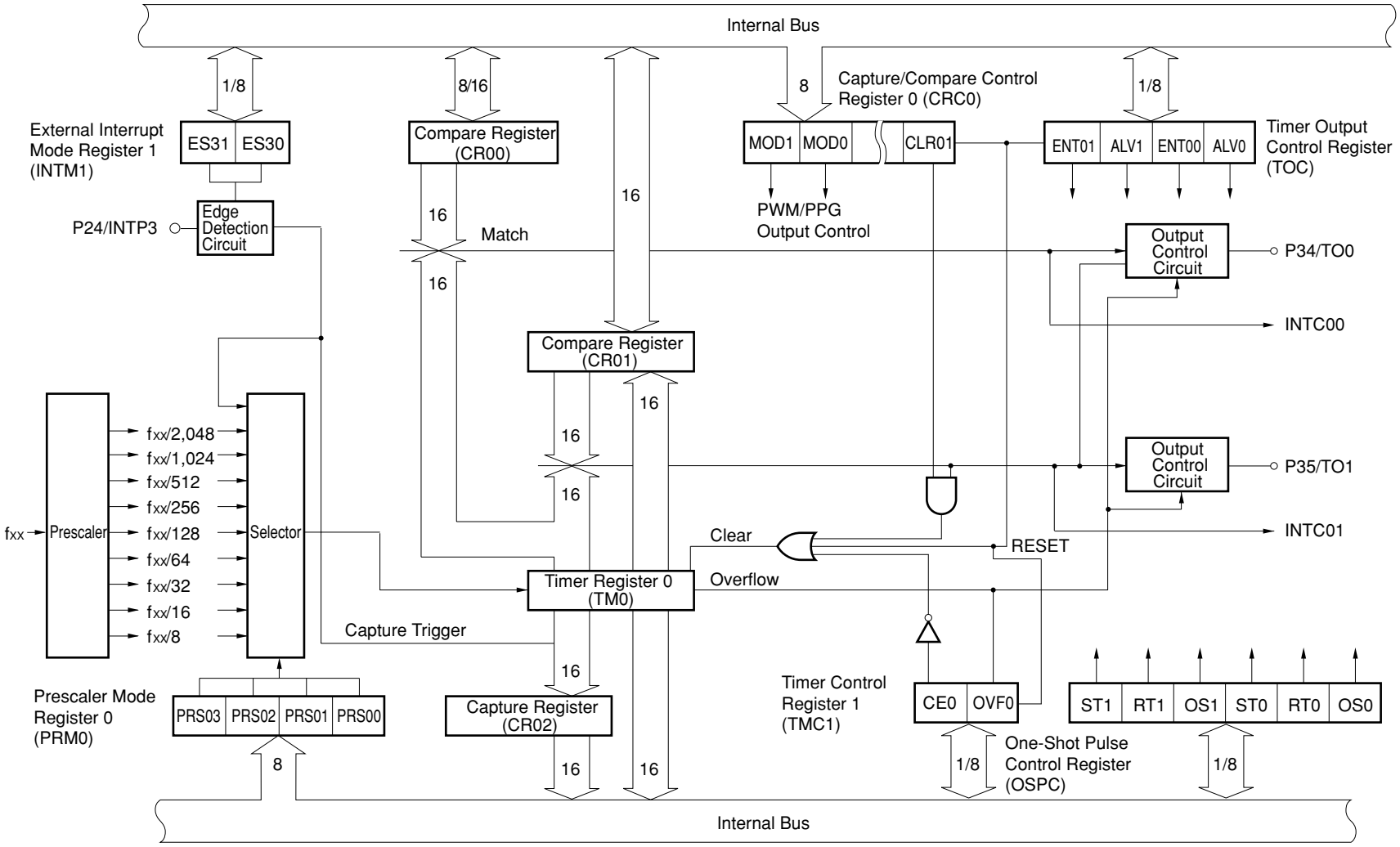
8.2 CONFIGURATION

Timer/counter 0 consists of the following registers:

- Timer register (TM0 × 1)
- Compare register (CR00, CR01) × 2
- Capture register (CR02) × 1

The block diagram of timer/counter 0 is shown in Figure 8-1.

Figure 8-1 Timer/Counter 0 Block Diagram



(1) Timer register 0 (TM0)

TM0 is a timer register that counts up using the count clock specified by the low-order 4 bits of prescaler mode register 0 (PRM0).

The count operation is stopped or enabled by means of timer control register 0 (TMC0).

TM0 can be read only with a 16-bit manipulation instruction. When $\overline{\text{RESET}}$ is input, TM0 is cleared to 0000H and the count is stopped.

Caution If the value of the timer register is read under the condition indicated by "×" in Table 8-5, the read value may be illegal. Do not read the timer register under condition "×".

Table 8-5 Limits of Reading Timer Register

(√: Can be read, ×: Must not be read)

Timer Count Clock \ f_{CLK}	$f_{XX}/2$	$f_{XX}/4$	$f_{XX}/8$	$f_{XX}/16$
$f_{XX}/8$	√	√	×	×
$f_{XX}/16$	√	√	√	×
f_{XX}/n	√	√	√	√

Remarks 1. f_{XX}: Oscillation frequency

2. f_{CLK}: Internal system clock frequency

3. n = 32, 64, 128, 256, 512, 1,024, 2,048

(2) Compare registers (CR00/CR01)

CR00 and CR01 are 16-bit registers that hold the values that determine the interval timer frequency.

If the CR00/CR01 contents match the contents of TM0, an interrupt request (INTC00/INTC01) and timer output control signal are generated. Also, the count value can be cleared by a content match (CR01).

CR00 and CR01 can be read or written with a 16-bit manipulation instruction. The contents of these registers are undefined after $\overline{\text{RESET}}$ input.

(3) Capture register (CR02)

CR02 is a 16-bit register that captures the contents of TM0.

The capture operation is synchronized with the input of a valid edge (capture trigger) on the external interrupt request input pin (INTP3). The contents of the CR02 are retained until the next capture trigger is generated.

CR02 can be read only with a 16-bit manipulation instruction. $\overline{\text{RESET}}$ input clears CR02 to 0000H.

(4) Edge detection circuit

The edge detection circuit detects an external input valid edge.

When the valid edge set by external interrupt mode register 1 (INTM1) is detected in the INTP3 pin input, the external interrupt request (INTP3), a capture trigger, and a external event count clock are generated (see **Figure 21-2** for details of the INTM1).

(5) Output control circuit

It is possible to invert the timer output when the compare register (CR00, CR01) register contents and the contents of the timer register (TM0) match. A square wave can be output from the timer output pins (TO0/TO1) in accordance with the setting of the low-order 4 bits of the timer output control register (TOC). At this time, PWM output or PPG output can be performed according to the specification of capture/compare control register 0 (CRC0).

In addition, one-shot pulse output can also be performed by means of a software trigger.

Timer output can be disabled/enabled by means of the TOC. When timer output is disabled, a fixed level is output to the TO0 and TO1 pins (the output level is set by the TOC).

(6) Prescaler

The prescaler generates the count clock from the internal system clock. The clock generated by this prescaler is selected by the selector, and is used as the count clock by the timer register 0 (TM0) to perform count operations.

(7) Selector

The selector selects a signal resulting from dividing the internal clock or the edge detected by the edge detection circuit as the count clock of timer register 0 (TM0).

8.3 TIMER/COUNTER 0 CONTROL REGISTERS

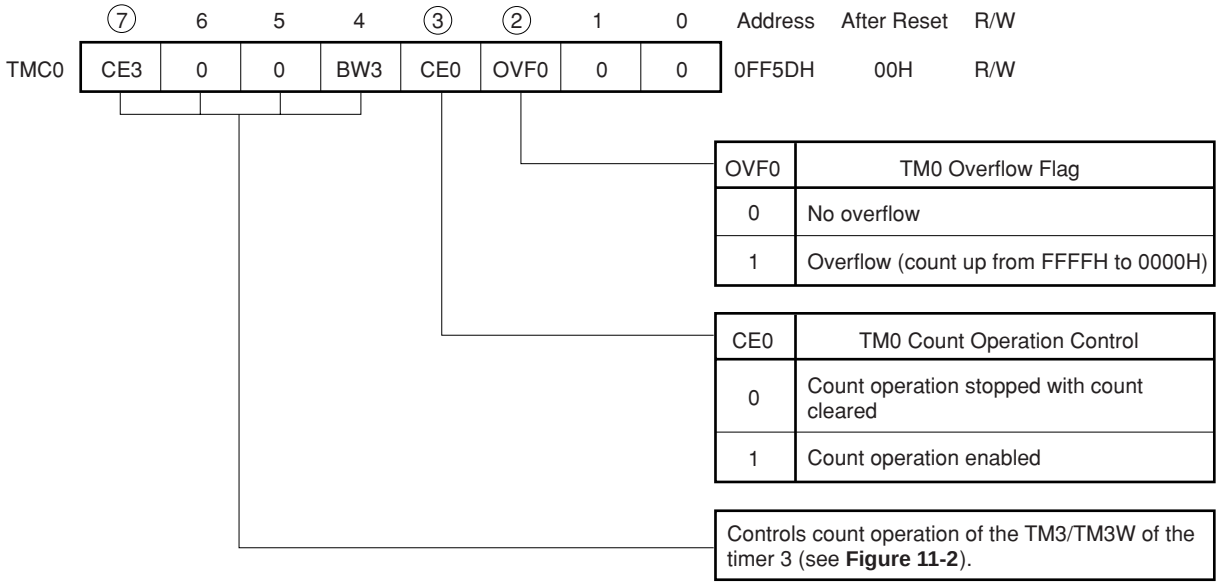
(1) Timer control register 0 (TMC0)

The timer/counter 0 TM0 count operation is controlled by the low-order 4 bits in the TMC0 (the high-order 4 bits control the count operation of the TM3/TM3W of the timer 3).

TMC0 can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. The format of the TMC0 is shown in Figure 8-2.

$\overline{\text{RESET}}$ input clears TMC0 to 00H.

Figure 8-2 Timer Control Register 0 (TMC0) Format



Remark The OVF0 bit is reset by software only.

(2) Prescaler mode register 0 (PRM0)

The count clock of the timer/counter 0, TM0, is specified by the low-order 4 bits of the PRM0 (the high-order 4 bits specify the count clock of the timer 3, TM3/TM3W).

PRM0 can be read/written with an 8-bit manipulation instruction. The format of the PRM0 is shown in Figure 8-3.

$\overline{\text{RESET}}$ input sets PRM0 to 11H.

Figure 8-3 Prescaler Mode Register 0 (PRM0) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PRM0	PRS3	PRS2	PRS1	PRS0	PRS03	PRS02	PRS01	PRS00	0FF5CH	11H	R/W

(f_{XX} = 32 MHz)

PRS03	PRS02	PRS01	PRS00	Timer/Counter 0 TM0 Count Clock Specification	
				Count Clock [Hz] Specification	Resolution [μs]
0	0	0	0	Setting prohibited	–
0	0	0	1	f_{XX}/8	0.25
0	0	1	0	f_{XX}/16	0.50
0	0	1	1	f_{XX}/32	1.00
0	1	0	0	f_{XX}/64	2.00
0	1	0	1	f_{XX}/128	4.00
0	1	1	0	f_{XX}/256	8.00
0	1	1	1	f_{XX}/512	16.00
1	0	0	0	f_{XX}/1,024	32.00
1	0	0	1	f_{XX}/2,048	64.00
1	1	1	1	External clock (INTP3)	–
Other than the above				Setting prohibited	

Specifies count clock of the TM3/TM3W of the timer 3 (see **Figure 11-3**).

Remark f_{XX}: X1 input frequency or oscillation frequency

(3) Capture/compare control register 0 (CRC0)

The CRC0 specifies the enabling conditions for the TM0 clear operation by a match signal between the contents of the compare register (CR01) and the timer register 0 (TM0) counter value, and the timer outputs (TO0/TO1) mode.

CRC0 can be read/written with an 8-bit manipulation instruction. The format of the CRC0 is shown in Figure 8-4.

$\overline{\text{RESET}}$ input sets CRC0 to 10H.

Figure 8-4 Capture/Compare Control Register 0 (CRC0) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
CRC0	MOD1	MOD0	0	1	CLR01	0	0	0	0FF30H	10H	R/W

MOD1	MOD0	CLR01	Timer Output Mode Specification		TM0 Clear Operation when TM0 = CR01
			TO0	TO1	
0	0	0	Toggle output	Toggle output	Disabled
0	0	1	Toggle output	Toggle output	Enabled
0	1	0	PWM output	Toggle output	Disabled
0	1	1	Setting prohibited		
1	0	0	PWM output	PWM output	Disabled
1	0	1	Setting prohibited		
1	1	0	Setting prohibited		
1	1	1	PPG output	Toggle output	Enabled

(4) Timer output control register (TOC)

TOC is an 8-bit register that controls the active level of timer output and output enabling/disabling.

The operation of the timer output pins (TO0 and TO1) by the timer/counter 0 is controlled by the low-order 4 bits (the high-order 4 bits control the operation of the timer output pins (TO2 and TO3 by the timer/counter 2).

TOC can be written to or read with an 8-bit manipulation instruction or bit manipulation instruction. The format of the TOC is shown in Figure 8-5.

$\overline{\text{RESET}}$ input clears TOC to 00H.

Figure 8-5 Timer Output Control Register (TOC) Format

	⑦	6	⑤	4	③	2	①	0	Address	After Reset	R/W
TOC	ENTO3	ALV3	ENTO2	ALV2	ENTO1	ALV1	ENTO0	ALV0	0FF31H	00H	R/W

ALV0	TO0 Pin Active Level	
	Toggle output specification or one-shot pulse output specification	PWM/PPG output specification
0	Low level	High level
1	High level	Low level

ENTO0	TO0 Pin Operation Specification
0	$\overline{\text{ALV0}}$ output
1	Pulse output enabled

ALV1	TO1 Pin Active Level	
	Toggle output specification or one-shot pulse output specification	PWM/PPG output specification
0	Low level	High level
1	High level	Low level

ENTO1	TO1 Pin Operation Specification
0	$\overline{\text{ALV1}}$ output
1	Pulse output enabled

Controls timer output pins (TO2, TO3) by timer/counter 2 (see **Figure 10-6**).

(5) One-shot pulse output control register (OSPC)

The OSPC is an 8-bit register that specifies enabling/disabling of one-shot pulse output by a software trigger and the output level, etc.

OSPC can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction.

The format of the OSPC is shown in Figure 8-6.

$\overline{\text{RESET}}$ input clears OSPC to 00H.

Figure 8-6 One-Shot Pulse Output Control Register (OSPC) Format

	⑦	⑥	5	4	③	②	1	0	Address	After Reset	R/W
OSPC	ST1	RT1	0	OS1	ST0	RT0	0	OS0	0FF7DH	00H	R/W

OS0	TO0 Pulse Output Type Selection
0	Toggle output/PWM output/PPG output selectable
1	Software triggered one-shot pulse selectable

ST0	RT0	TO0 Output Control
0	0	Output not changed
0	1	Inactive level output to TO0
1	0	Active level output to TO0
1	1	Setting prohibited

OS1	TO1 Pulse Output Type Selection
0	Toggle output/PWM output/PPG output selectable
1	Software triggered one-shot pulse output

ST1	RT1	TO1 Output Control
0	0	Output not changed
0	1	Inactive level output to TO1
1	0	Active level output to TO1
1	1	Setting prohibited

Remarks 1. The RT0, ST0, RT1, and ST1 bits are write-only, and show a value of “0” if read.

2. Pin pulse output disabling/enabling and active level setting are performed by means of the timer output control register (TOC).

8.4 16-BIT TIMER REGISTER 0 (TM0) OPERATION

8.4.1 Basic Operation

In the timer/counter 0 count operation, an up-count is performed using the count clock specified by the low-order 4 bits of prescaler mode register 0 (PRM0).

Count operation enabling/disabling is controlled by bit 3 (CE0) of timer control register 0 (TMC0). When the CE0 bit is set (to 1) by software, the contents of TM0 are cleared to 0000H on the first count clock, and then the up-count operation is performed.

When the CE0 bit is cleared (to 0), TM0 becomes 0000H immediately, and capture operations and match signal generation are stopped.

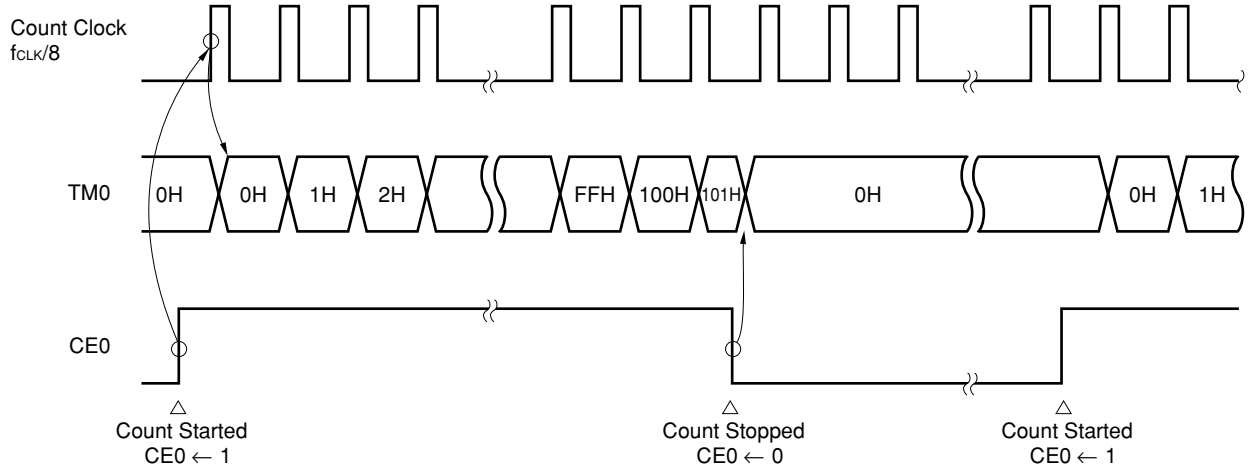
If the CE0 bit is set (to 1) again when it is already set (to 1), TM0 continues the count operation without being cleared.

If the count clock is input when TM0 is FFFFH, TM0 becomes 0000H. In this case, OVF0 bit is set (to 1) and an overflow signal is sent to the output control circuit. OVF0 bit is cleared by software only. The count operation is continued.

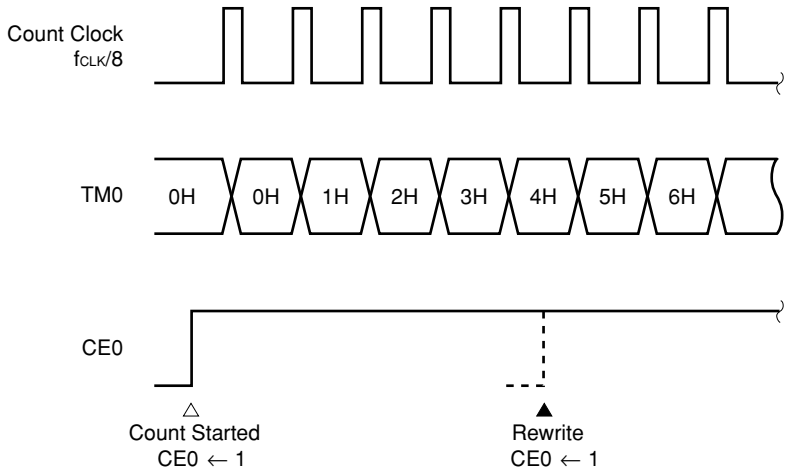
When $\overline{\text{RESET}}$ is input, TM0 is cleared to 0000H, and the count operation is stopped.

Figure 8-7 Basic Operation of Timer Register 0 (TM0)

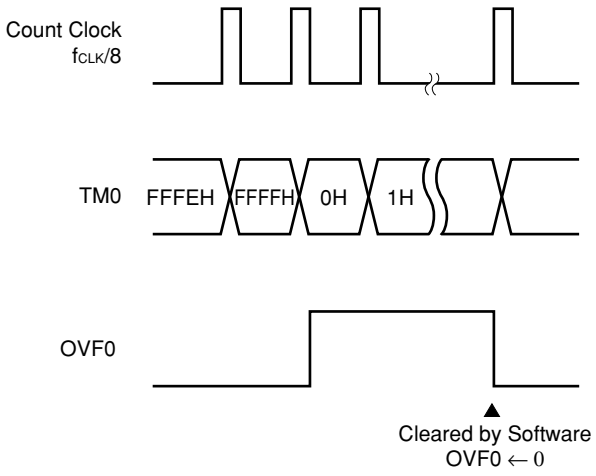
(a) Count started → count stopped → count started



(b) When "1" is written to the CE0 bit again after the count starts



(c) Operation when TM0 = FFFFH

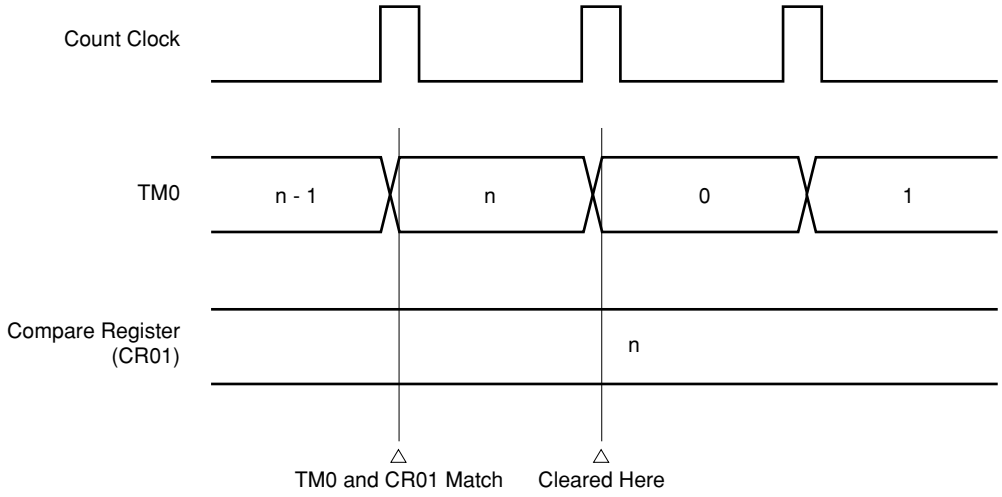


8.4.2 Clear Operation

(1) Clear operation after a match with the compare register

The timer register 0 (TM0) can be cleared automatically after a match with the compare register (CR01). When a clearance source arises, TM0 is cleared to 0000H on the next count clock. Therefore, even if a clearance source arises, the value at the point at which the clearance source arose is retained until the next count clock arrives.

Figure 8-8 TM0 Clearance by Match with Compare Register (CR01)

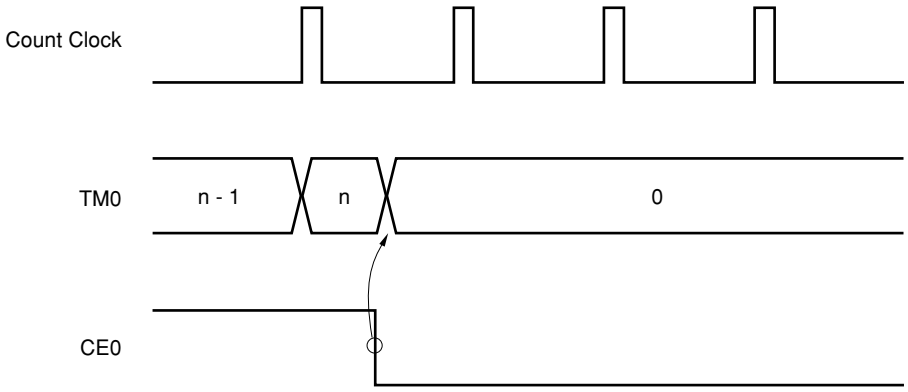


(2) Clear operation by the CE0 bit of the timer control register 0 (TMC0)

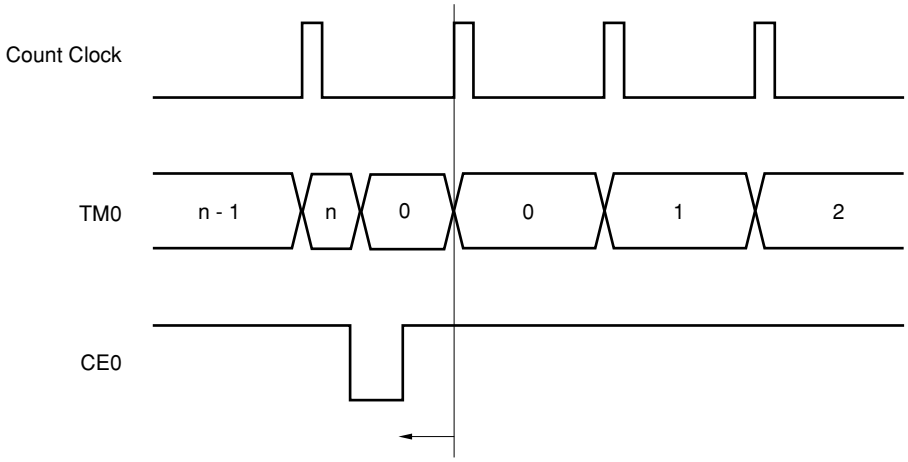
The timer register 0 (TM0) is also cleared when the CE0 bit of TMC0 is cleared (to 0) by software. The clear operation is performed immediately after clearance (to 0) of the CE0 bit.

Figure 8-9 Clear Operation When CE0 Bit is Cleared (0)

(a) Basic operation

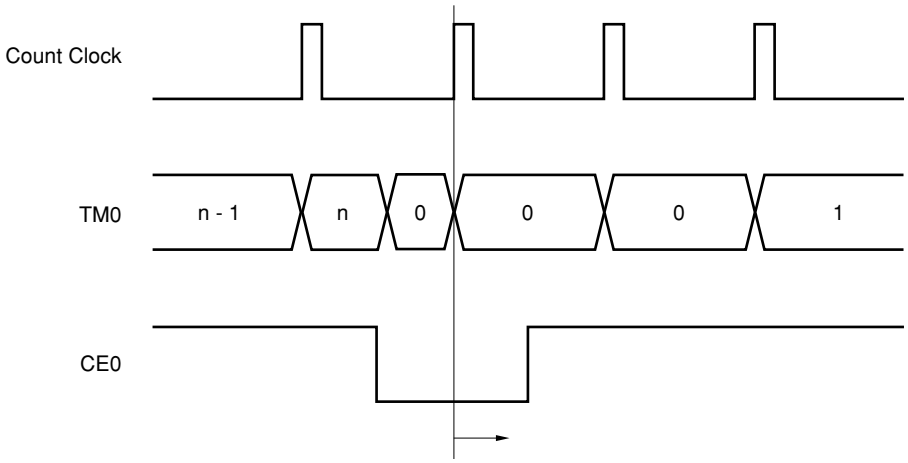


(b) Restart before count clock input after clearance



If the CE0 bit is set (to 1) before this count clock, the count starts from 0 on the count clock.

(c) Restart after count clock input after clearance



If the CE0 bit is set (to 1) from this count clock onward, the count starts from 0 on the count clock after the CE0 bit is set (to 1).

8.5 EXTERNAL EVENT COUNTER FUNCTION

The timer/counter 0 can count clock pulses input from the external interrupt request input pin (INTP3).

No special selection method is needed for the external event counter operating mode. When the timer register 0 (TM0) count clock is specified as external clock input by the setting of the low-order 4 bits of prescaler mode register 0 (PRM0), TM0 operates as an external event counter.

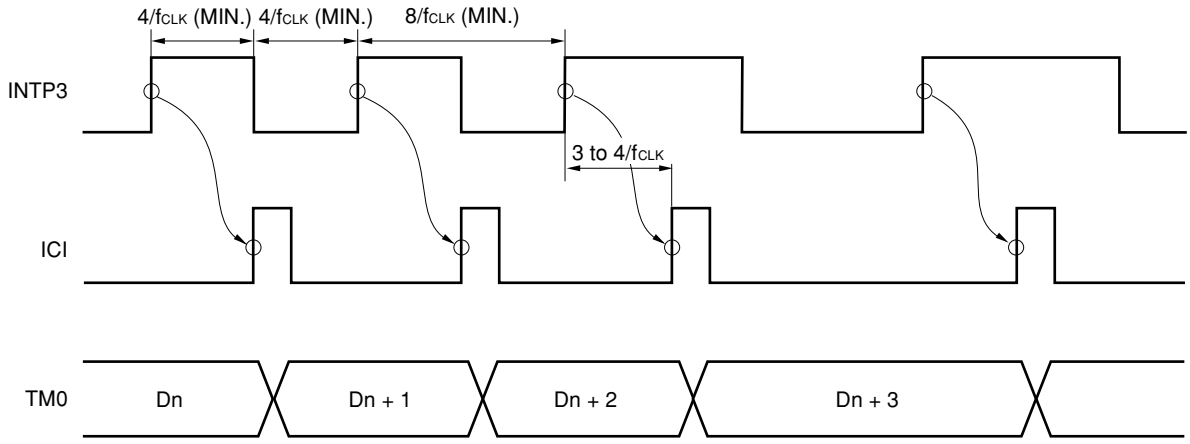
The maximum frequency of external clock pulses that can be counted by TM0 as the external event counter is 2.00 MHz (f_{CLK} = 16 MHz) irrespective of whether only one edge or both edges are counted on INTP3 input.

The pulse width of the INTP3 input must be at least 4 system clocks (0.25 μs: f_{CLK} = 16 MHz) for both the high level and low level. If the pulse width is shorter than this, the pulse may not be counted.

The timer/counter 0 external event counter timing is shown in Figure 8-10.

Figure 8-10 Timer/Counter 0 External Event Count Timing (1/2)

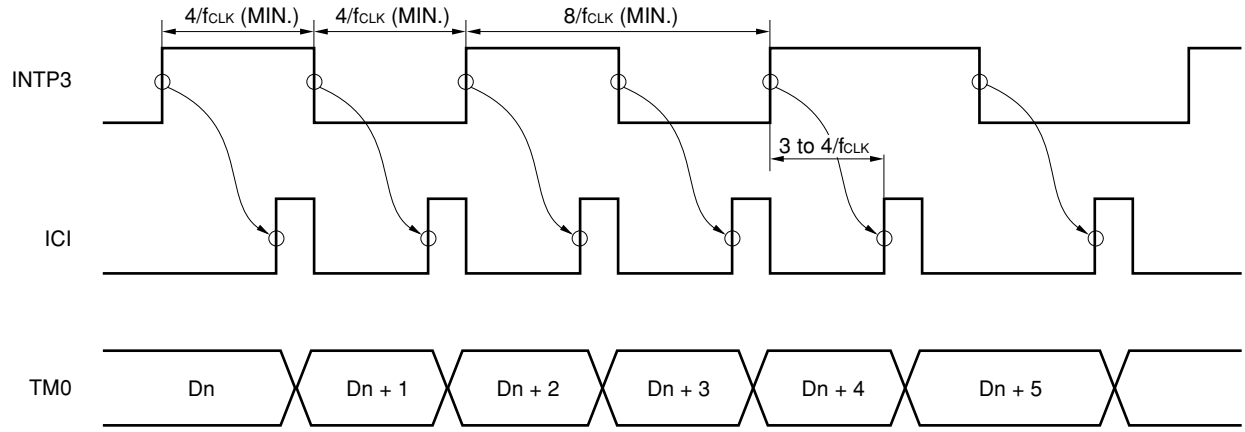
(1) Counting one edge (maximum frequency = $f_{CLK}/8$)



Remark ICI: INTP3 input signal after passing through edge detection circuit

Figure 8-10 Timer/Counter 0 External Event Count Timing (2/2)

(2) Counting both edges (maximum frequency = $f_{CLK}/8$)



Remark ICI: INTP3 input signal after passing through edge detection circuit

The TM0 count operation is controlled by the CE0 bit of the timer control register 0 (TMC0) in the same way as with basic operation.

When the CE0 bit is set (to 1) by software, the contents of TM0 are set to 0000H and the up-count is started on the initial count clock.

When the CE0 bit is cleared (to 0) by software during a TM0 count operation, the contents of TM0 are set to 0000H immediately and the stopped state is entered. The TM0 count operation is not affected if the CE0 bit is set (to 1) by software again when it is already set (to 1).

Caution When timer/counter 0 is used as an external event counter, it is not possible to distinguish between the case where there is no valid edge input at all and the case where there is a single valid edge input, using the timer register 0 (TM0) alone (see Figure 8-11), since the contents of TM0 are 0 in both cases. If it is necessary to make this distinction, the INTP3 interrupt request flag should be used. An example is shown in Figure 8-12.

Figure 8-11 Example of the Case Where the External Event Counter Does Not Distinguish Between One Valid Edge Input and No Valid Edge Input

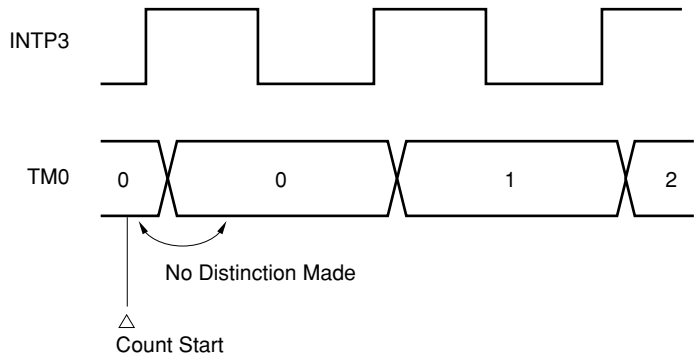
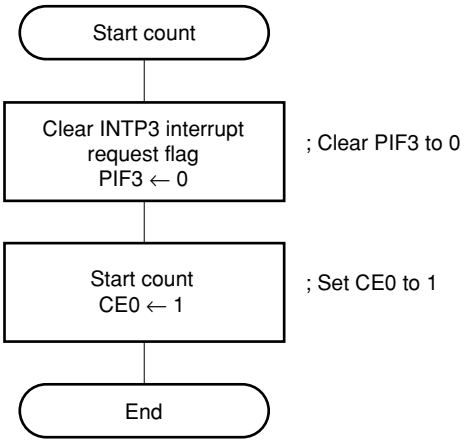
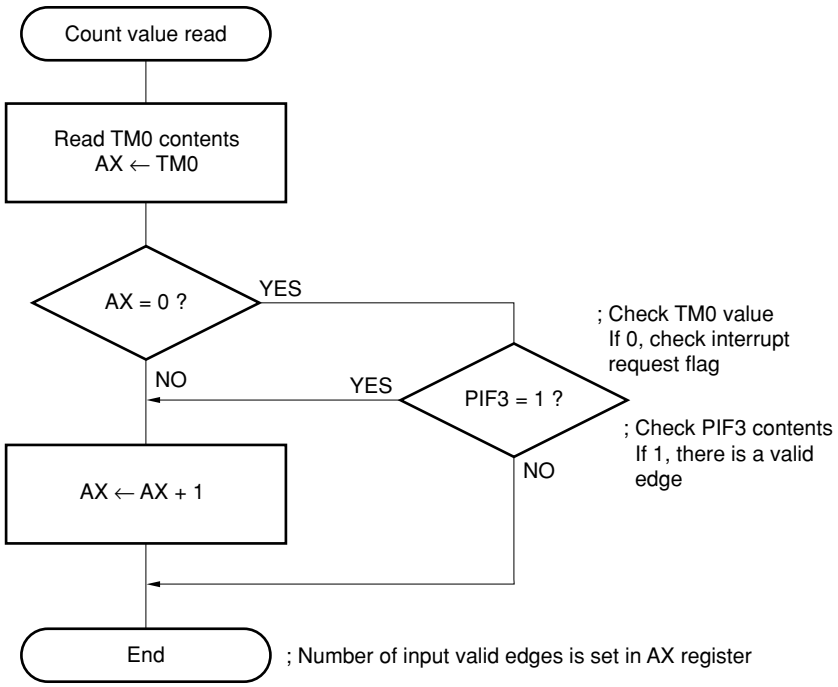


Figure 8-12 Methods of Enabling the External Event Counter to Distinguish No Valid Edge Input

(a) Processing when count is started



(b) Processing when count value is read



8.6 COMPARE REGISTER AND CAPTURE REGISTER OPERATION

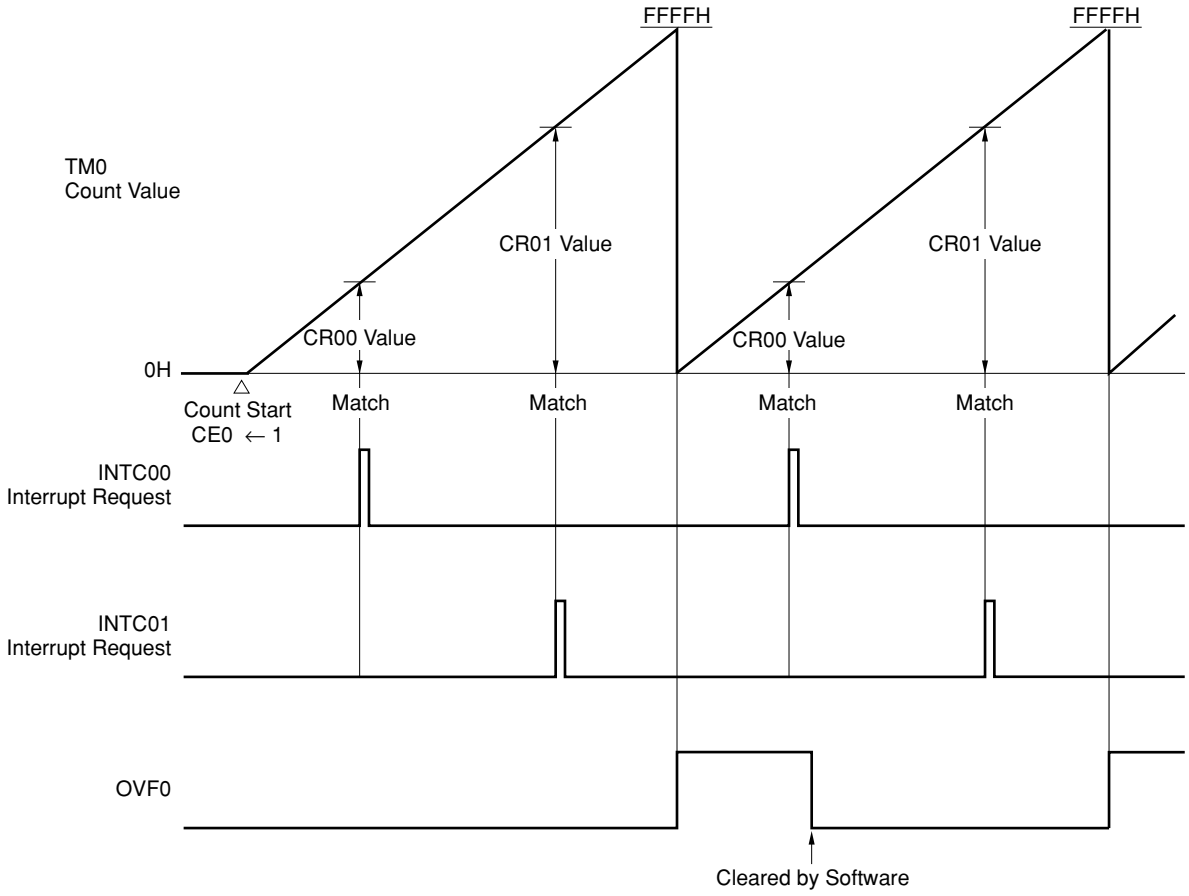
8.6.1 Compare Operations

Timer/counter 0 performs compare operations in which the value set in compare registers (CR00, CR01) are compared with the timer register 0 (TM0) count value.

If the count value of TM0 matches the preset CR0n (n = 0, 1) value as the result of the count operation, a match signal is sent to the output control circuit, and at the same time an interrupt request (INTC00/INTC01) is generated.

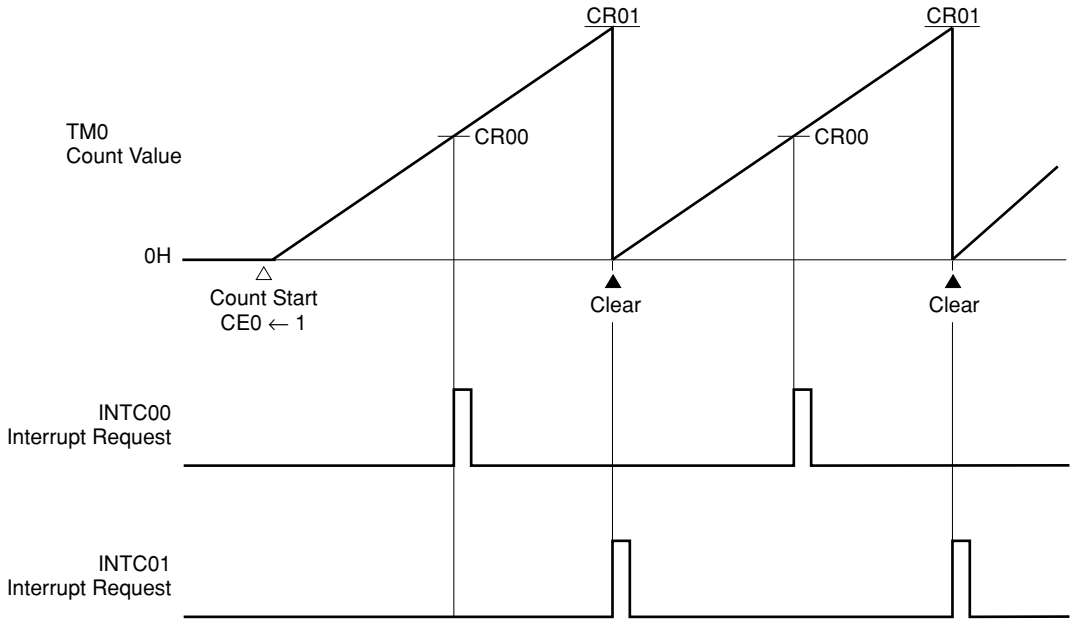
After a match with the CR01 value, the TM0 count value can be cleared, and the timer functions as an interval timer that repeatedly counts up to the value set in the CR01.

Figure 8-13 Compare Operation



Remark CLR01 = 0

Figure 8-14 TM0 Clearance After Match Detection



Remark CLR01 = 0

8.6.2 Capture Operations

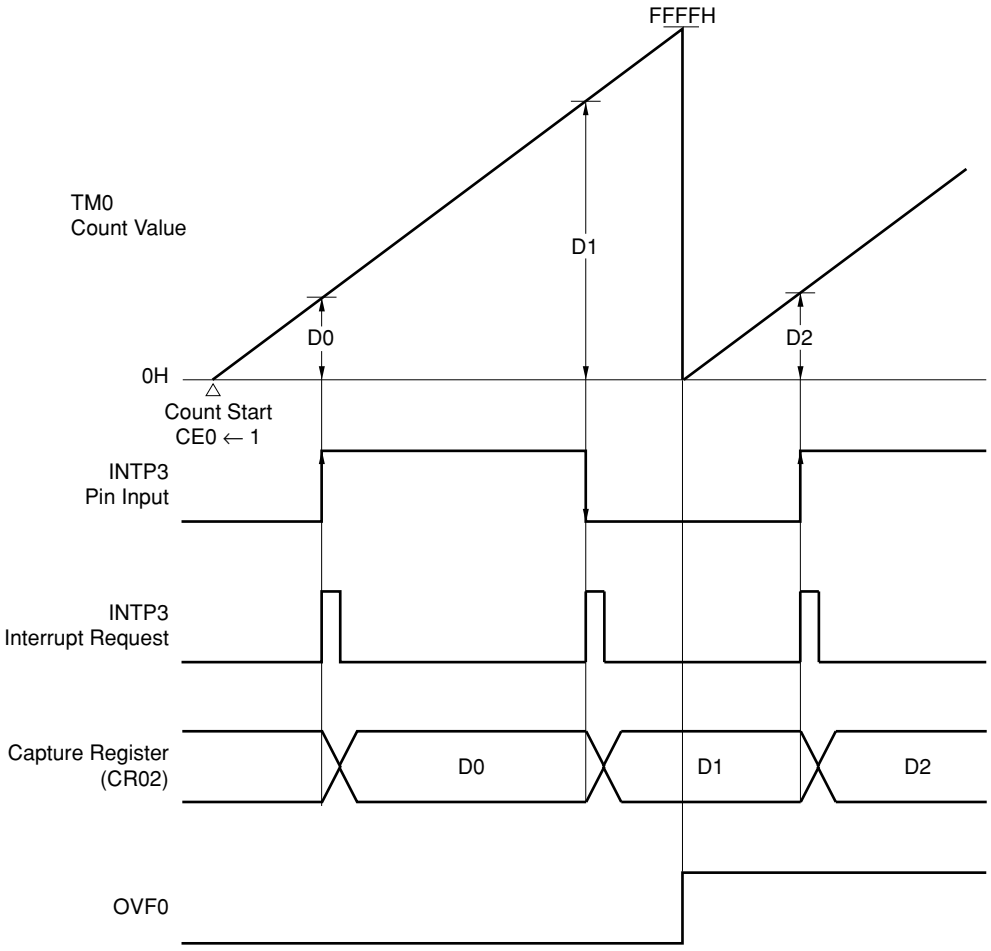
Timer/counter 0 performs capture operations in which the timer register 0 (TM0) count value is fetched into the capture register in synchronization with an external trigger, and retained there.

A valid edge detected from the input of the external interrupt request input pin (INTP3) is used as the external trigger (capture trigger). The count value of TM0 in the process of being counted is fetched into the capture register (CR02) in synchronization with the capture trigger, and is retained there. The contents of the CR02 are retained until the next capture trigger is generated.

The capture trigger valid edge is set by means of external interrupt mode register 1 (INTM1). If both rising and falling edges are set as capture triggers, the width of pulses input from off-chip can be measured. Also, if a capture trigger is generated by a single edge, the input pulse cycle can be measured.

See **Figure 21-2** in **CHAPTER 21 EDGE DETECTION FUNCTION** for details of the INTM1.

Figure 8-15 Capture Operation



Remark Dn: TM0 count value (n = 0, 1, 2, ...)
CLR01 = 0

8.7 BASIC OPERATION OF OUTPUT CONTROL CIRCUIT

The output control circuit controls the timer output pin (TO0/TO1) levels by means of overflow signals or match signals from the compare registers (CR00, CR01). The operation of the output control circuit is determined by the timer output control register (TOC), capture/compare control register 0 (CRC0), and the one-shot pulse output control register (OSPC) (see **Table 8-6**).

When TO0, TO1 signals are output to a pin, the relevant pin must be in control mode in the port 3 mode register (PMC3).

Table 8-6 Timer Output (TO0/TO1) Operations

TOC				OSPC		CRC0			TO1	TO0
ENTO1	ALV1	ENTO0	ALV0	OS1	OS0	MOD1	MOD0	CLR01		
0	0/1	0	0/1	×	×	×	×	×	High/low level fixed	High/low level fixed
0	0/1	1	0/1	×	0	0	0	×	High/low level fixed	Toggle output (active-low/high)
0	0/1	1	0/1	×	0	0	1	0	High/low level fixed	PWM output (active-high/low)
0	0/1	1	0/1	×	0	1	0	0	High/low level fixed	PWM output (active-high/low)
0	0/1	1	0/1	×	0	1	1	1	High/low level fixed	PPG output (active-high/low)
0	0/1	1	0/1	×	1	×	×	×	High/low level fixed	One-shot pulse output (active-low/high)
1	0/1	0	0/1	0	×	0	×	×	Toggle output (active-low/high)	High/low level fixed
1	0/1	0	0/1	0	×	1	0	0	PWM output (active-high/low)	High/low level fixed
1	0/1	0	0/1	0	×	1	1	×	Toggle output (active-low/high)	High/low level fixed
1	0/1	0	0/1	1	×	×	×	×	One-shot pulse output (active-low/high)	High/low level fixed
1	0/1	1	0/1	0	0	0	0	×	Toggle output (active-low/high)	Toggle output (active-low/high)
1	0/1	1	0/1	0	0	0	1	0	Toggle output (active-low/high)	PWM output (active-high/low)
1	0/1	1	0/1	0	0	1	0	0	PWM output (active-high/low)	PWM output (active-high/low)
1	0/1	1	0/1	0	0	1	1	1	Toggle output (active-low/high)	PPG output (active-high/low)
1	0/1	1	0/1	0	1	0	×	×	Toggle output (active-low/high)	One-shot pulse output (active-low/high)
1	0/1	1	0/1	0	1	1	0	0	PWM output (active-high/low)	One-shot pulse output (active-low/high)
1	0/1	1	0/1	0	1	1	1	1	Toggle output (active-low/high)	One-shot pulse output (active-low/high)
1	0/1	1	0/1	1	0	0	0	×	One-shot pulse output (active-low/high)	Toggle output (active-low/high)
1	0/1	1	0/1	1	0	0	1	0	One-shot pulse output (active-low/high)	PWM output (active-high/low)
1	0/1	1	0/1	1	0	1	0	0	One-shot pulse output (active-low/high)	PWM output (active-high/low)
1	0/1	1	0/1	1	0	1	1	1	One-shot pulse output (active-low/high)	PPG output (active-high/low)
1	0/1	1	0/1	1	1	×	×	×	One-shot pulse output (active-low/high)	One-shot pulse output (active-low/high)

Remarks 1. In the ALVn (n = 0, 1) columns, the figures on the left and right of the slash (“/”) correspond to the items on the left and right of the slash in the TOn (n = 0, 1) columns.

2. The “×” mark indicates that the operation is the same for either 0 or 1, but some prohibited combinations are included (see **Figure 8-4**).

3. Use with combinations not shown in this table is prohibited.

8.7.1 Basic Operation

Setting (to 1) the ENTOn (n = 0, 1) bit of the timer output control register (TOC) enables timer output (TOn: n = 0, 1) to be varied at a timing in accordance with the settings of MOD0, MOD1, and CLR01 bits of capture/compare control register 0 (CRC0) and the one-shot pulse output control register (OSPC).

Clearing (to 0) ENTOn sets the TOn to a fixed level. The fixed level is determined by the ALVn (n = 0, 1) bit of the TOC. The level is high when ALVn is 0, and low when 1.

8.7.2 Toggle Output

Toggle output is an operating mode in which the output level is inverted each time the compare register (CR00/CR01) value coincides with the timer register 0 (TM0) value. The output level of timer output (TO0) is inverted by a match between CR00 and TM0, and the output level of TO1 is inverted by a match between CR01 and TM0.

When timer/counter 0 is stopped by clearing (to 0) the CE0 bit of the timer control register 0 (TMC0), the inactive level ($\overline{\text{ALVn}}$: n = 0, 1) is output.

Figure 8-16 Toggle Output Operation

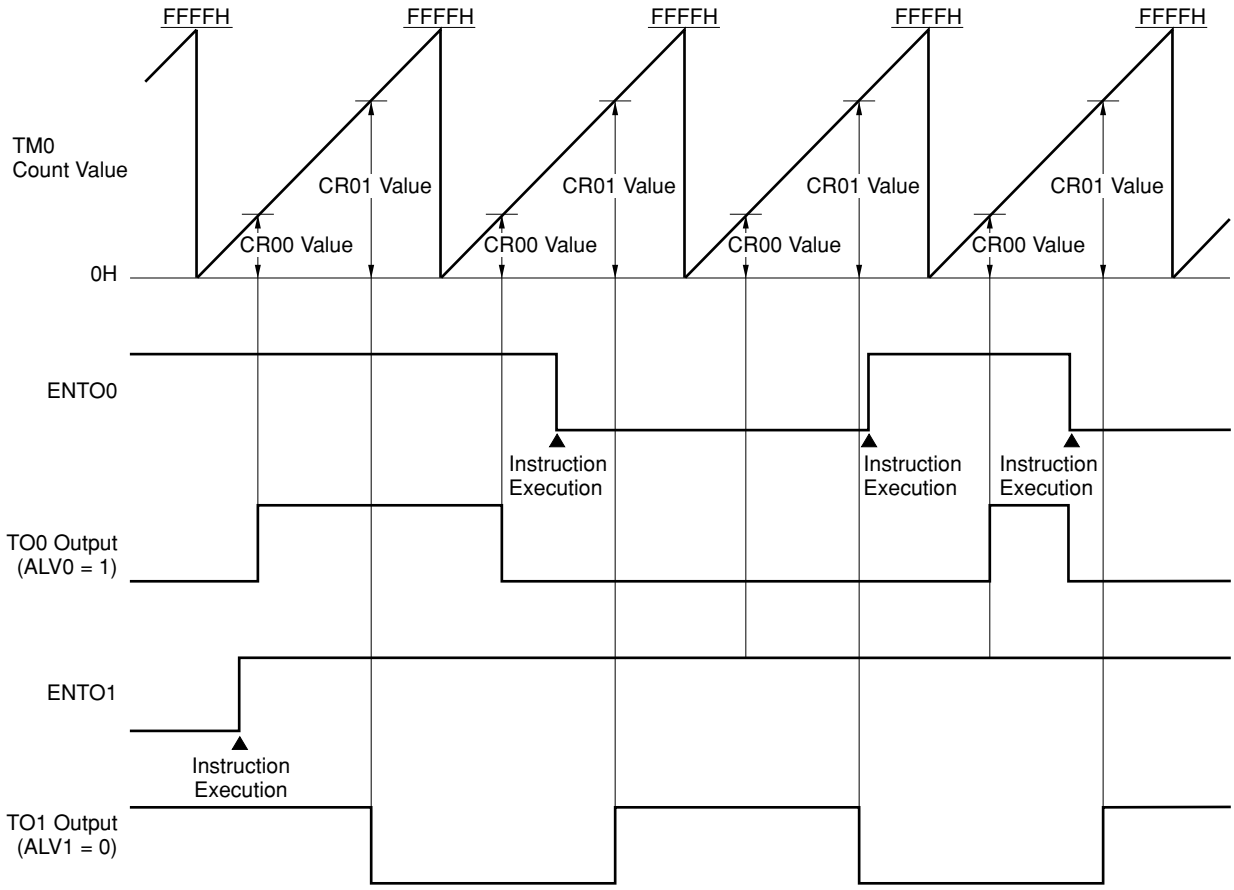


Table 8-7 TO0, TO1 Toggle Output (f_{XX} = 32 MHz)

Count Clock	Minimum Pulse Width	Maximum Interval Time
8/f_{XX}	0.25 μs	16.40 ms
16/f_{XX}	0.50 μs	32.80 ms
32/f_{XX}	1.00 μs	65.50 ms
64/f_{XX}	2.00 μs	131 ms
128/f_{XX}	4.00 μs	262 ms
256/f_{XX}	8.00 μs	524 ms
512/f_{XX}	16.00 μs	1.05 s
1,024/f_{XX}	32.00 μs	2.10 s
2,048/f_{XX}	64.00 μs	4.19 s

8.7.3 PWM Output

(1) Basic operation of PWM output

In this mode, a PWM signal with the period in which timer register 0 (TM0) reaches a full count used as one cycle is output. The timer output (TO0) pulse width is determined by the value of compare register (CR00), and the timer output (TO1) pulse width is determined by the value of compare register (CR01). When this function is used, the CLR01 bit of capture/compare control register 0 (CRC0) must be set to 0.

The pulse cycle and pulse width are as shown below.

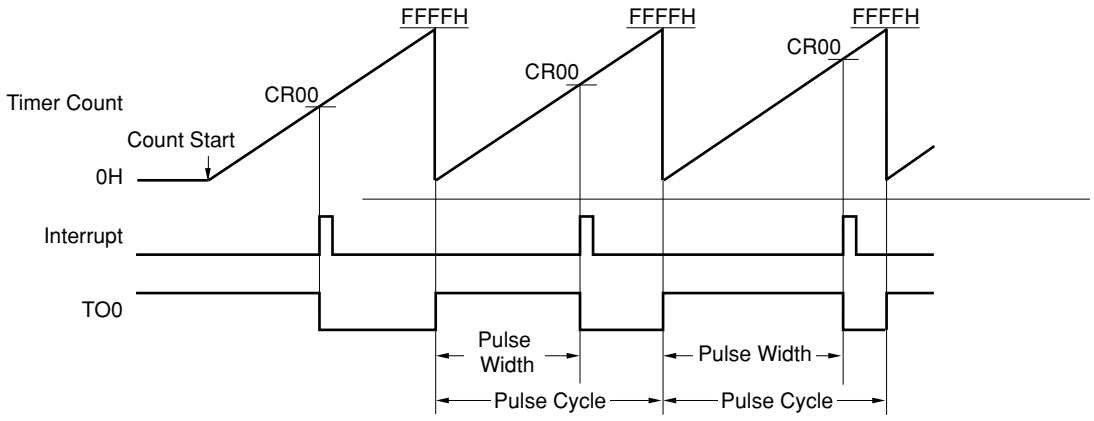
- PWM cycle = 65,536 × x/f_{XX}
- PWM pulse width = CR0n × x/f_{XX} **Note**; x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

Note 0 cannot be set in the CR0n.

- $\text{Duty} = \dfrac{\text{PWM pulse width}}{\text{PWM cycle}} = \dfrac{\text{CR0n}}{65{,}536}$

Remark n = 0, 1

Figure 8-17 PWM Pulse Output



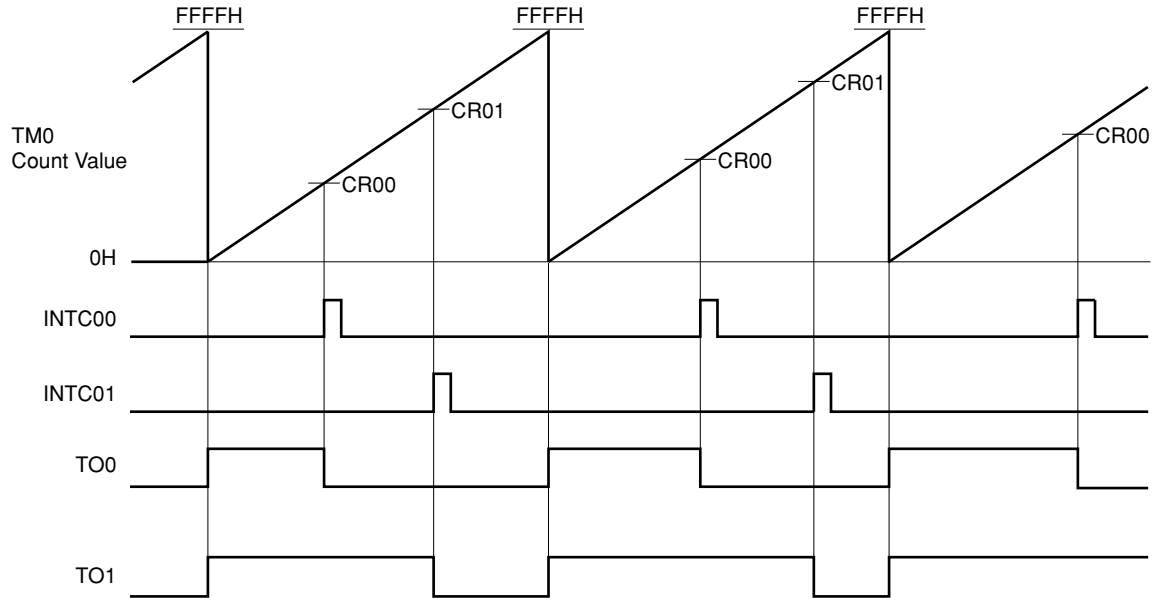
Remark ALV0 = 0

Table 8-8 TO0, TO1 PWM Cycle (f_{XX} = 32 MHz)

Count Clock	Minimum Pulse Width [μs]	PWM Cycle [s]	PWM Frequency [Hz]
f_{XX}/8	0.25	0.02	61.0
f_{XX}/16	0.50	0.03	30.5
f_{XX}/32	1.00	0.07	15.3
f_{XX}/64	2.00	0.13	7.6
f_{XX}/128	4.00	0.26	3.8
f_{XX}/256	8.00	0.52	1.9
f_{XX}/512	16.00	1.05	0.8
f_{XX}/1,024	32.00	2.10	0.5
f_{XX}/2,048	64.00	4.19	0.2

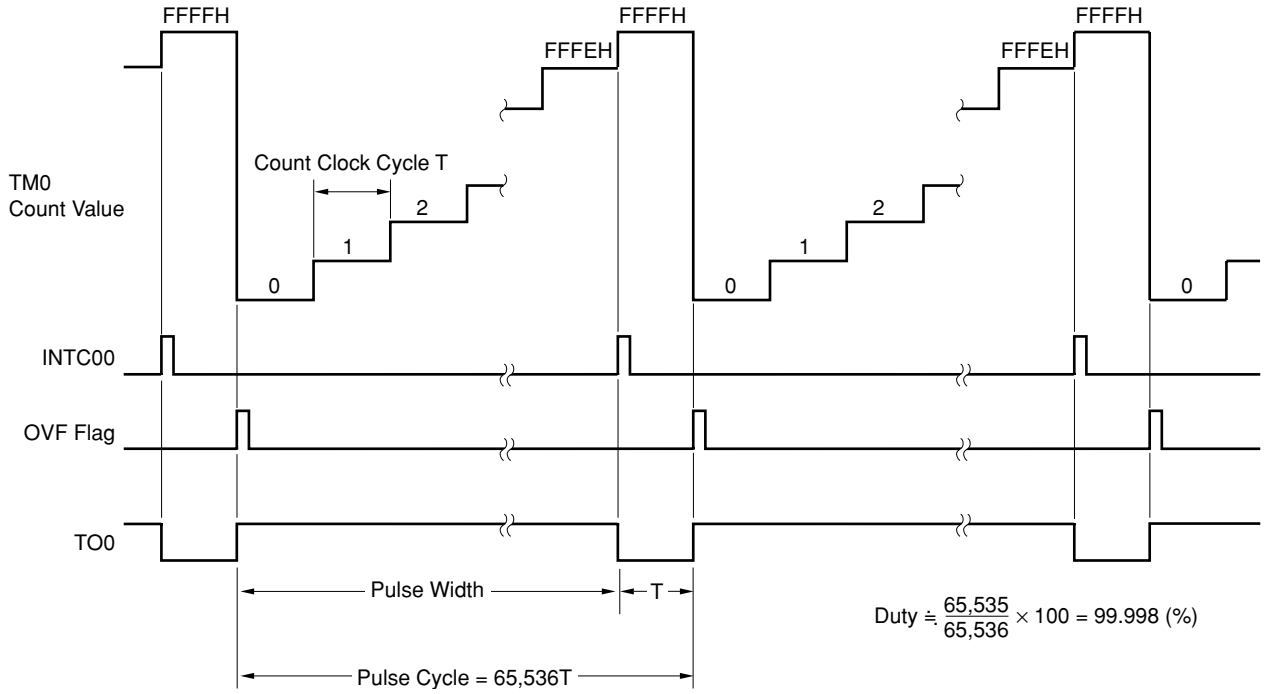
Figure 8-18 shows an example of 2-channel PWM output, and Figure 8-19 shows the operation of the case where FFFFH is set in the CR00.

Figure 8-18 Example of PWM Output Using TM0



Remark ALV0 = 0, ALV1 = 0

Figure 8-19 Example of PWM Output When CR00 = FFFFH



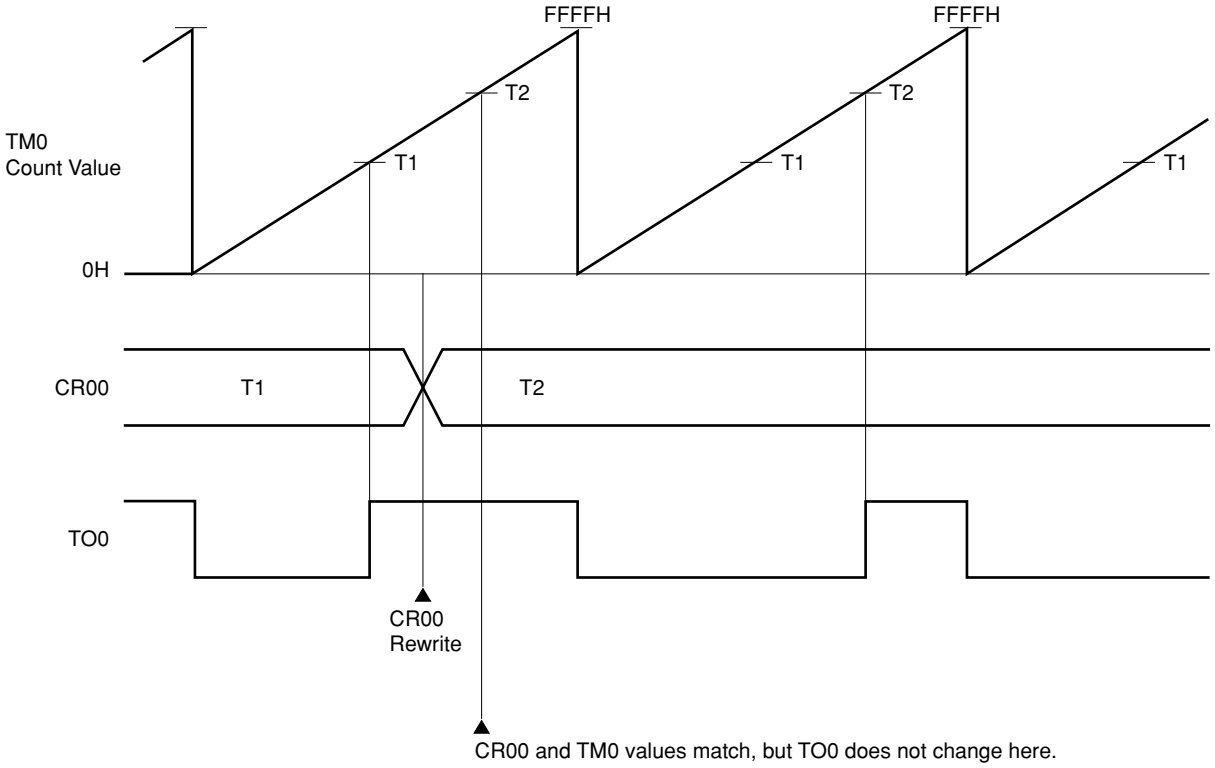
Remarks 1. ALV0 = 0

2. T = x/fxx (x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048)

(2) Rewriting compare registers (CR00, CR01)

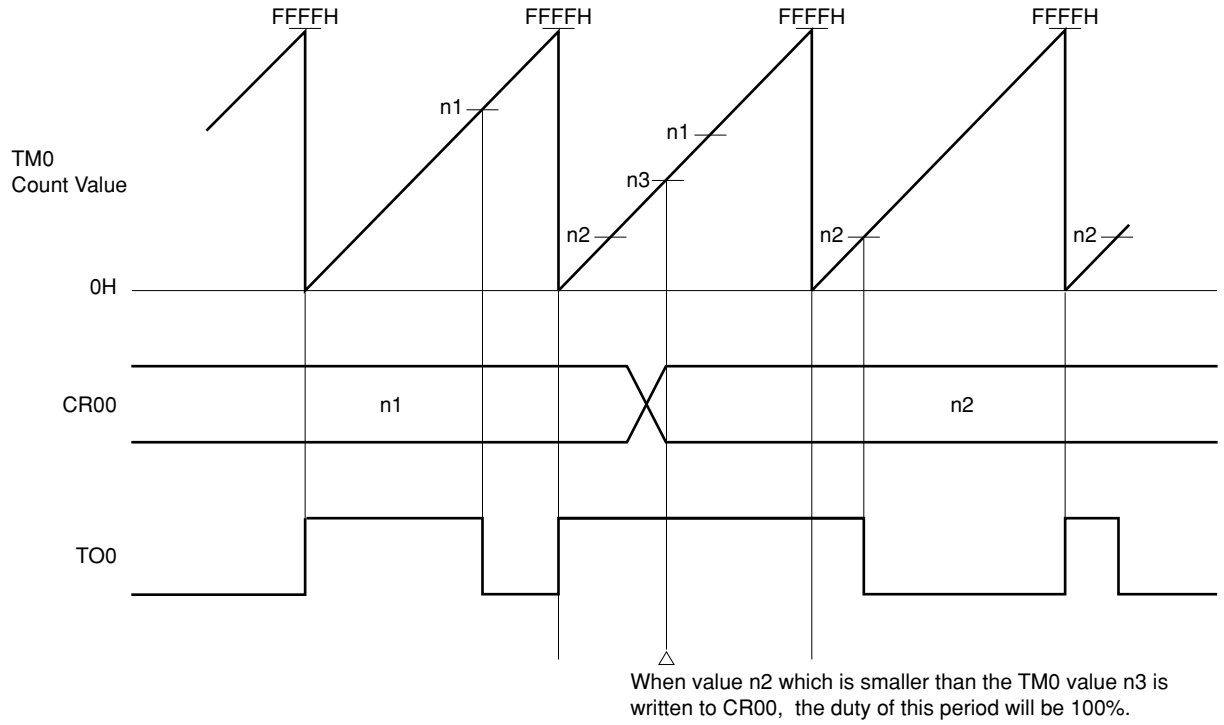
The output level of the timer output (TOn: n = 0, 1) does not change even if the CR0n (n = 0, 1) value matches the timer register 0 (TM0) value more than once during one PWM output cycle.

Figure 8-20 Example of Compare Register (CR00) Rewrite



If a value smaller than that of the TM0 is set as the CR0n value, a 100% duty PWM signal will be output. CR0n rewriting should be performed by the interrupt due to a match between TM0 and the CR0n on which the rewrite is performed.

Figure 8-21 Example of 100% Duty With PWM Output

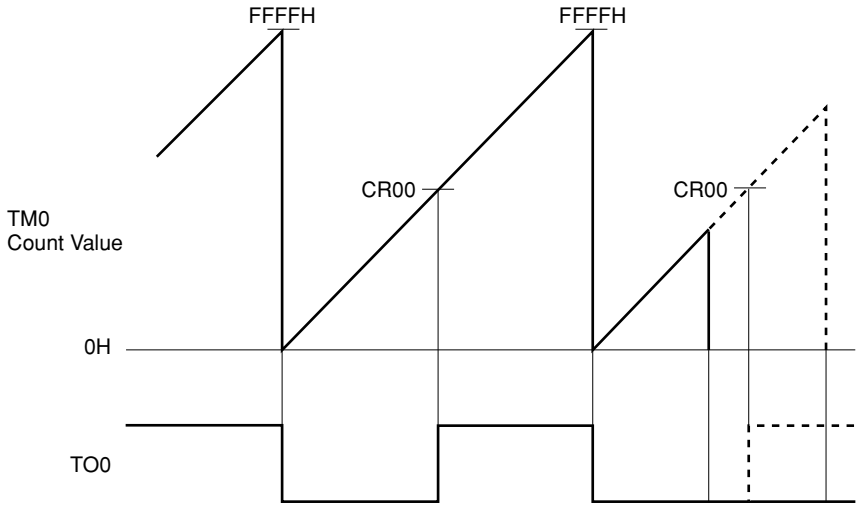


Remark ALV0 = 0

(3) Stopping PWM output

If timer/counter 0 is stopped by clearing (to 0) the CE0 bit of the timer control register 0 (TMC0) during PWM signal output, the active level is output.

Figure 8-22 When Timer/Counter 0 is Stopped During PWM Signal Output



Remark ALV0 = 1

Caution The output level of the TOn (n = 0, 1) pin when timer output is disabled (ENTOn = 0: n = 0, 1) is the inverse of the value set in ALVn (n = 0, 1) bit. Caution is therefore required as the active level is output when timer output is disabled when the PWM output function has been selected.

8.7.4 PPG Output

(1) Basic Operation of PPG Output

This function outputs a square-wave with the time determined by compare register CR01 value as one cycle, and the time determined by compare register CR00 value as the pulse width. The PWM cycle output by the PWM is made variable. This signal can only be output from the timer output (TO0).

When this function is used, the CLR01 bit of capture/compare control register 0 (CRC0) must be set to 1.

The pulse cycle and pulse width are as shown below.

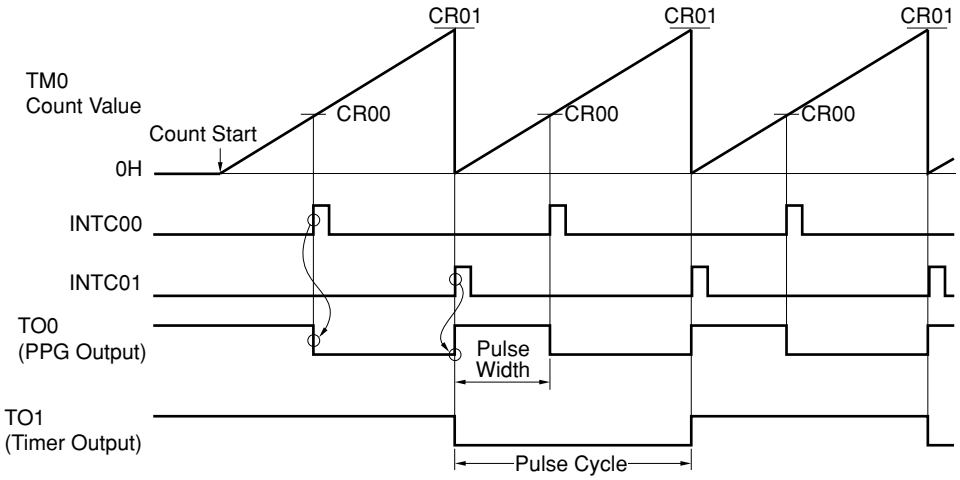
- PPG cycle = (CR01 + 1) × x/fxx; x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048
- PPG pulse width = CR00 × x/fxx

  where 1 ≤ CR00 ≤ CR01 **Note**
- Duty $= \dfrac{\text{PPG pulse width}}{\text{PPG cycle}} = \dfrac{\text{CR00}}{\text{CR01} + 1}$ **Note**

Note Both CR00 and CR01 cannot be cleared to “0”.

Figure 8-23 shows an example of PPG output using timer register 0 (TM0), Figure 8-24 shows an example of the case where CR00 = CR01.

Figure 8-23 Example of PPG Output Using TM0

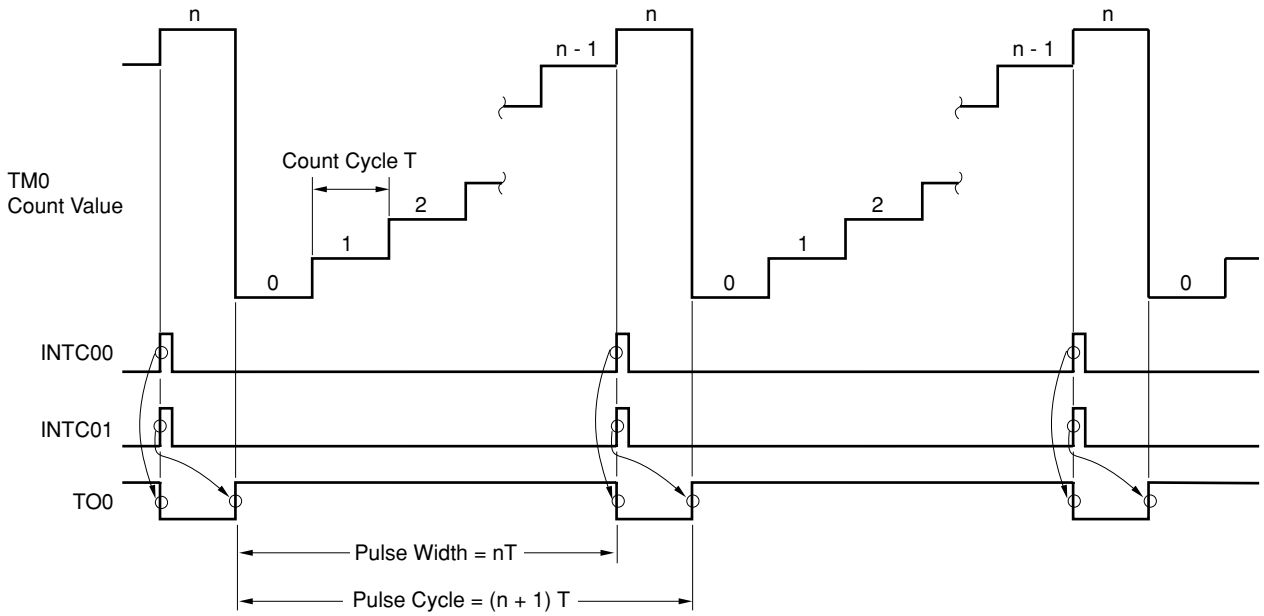


Remark ALV0 = 0, ALV1 = 0

Table 8-9 TO0 PPG Output (f_{XX} = 32 MHz)

Count Clock	Minimum Pulse Width	PPG Cycle	PPG Frequency
$f_{XX}/8$	0.25 μs	0.50 μs to 16.40 ms	2,000 kHz to 61.0 Hz
$f_{XX}/16$	0.50 μs	1.00 μs to 32.80 ms	1,000 kHz to 30.5 Hz
$f_{XX}/32$	1.00 μs	2.00 μs to 65.50 ms	500 kHz to 15.3 Hz
$f_{XX}/64$	2.00 μs	4.00 μs to 0.13 s	250 kHz to 7.6 Hz
$f_{XX}/128$	4.00 μs	8.00 μs to 0.26 s	125 kHz to 3.3 Hz
$f_{XX}/256$	8.00 μs	16.00 μs to 0.52 s	62.5 kHz to 1.9 Hz
$f_{XX}/512$	16.00 μs	32.00 μs to 1.05 s	31.3 kHz to 1.0 Hz
$f_{XX}/1,024$	32.00 μs	64.00 μs to 2.10 s	15.6 kHz to 0.5 Hz
$f_{XX}/2,048$	64.00 μs	128.00 μs to 4.19 s	7.8 kHz to 0.2 Hz

Figure 8-24 Example of PPG Output When CR00 = CR01



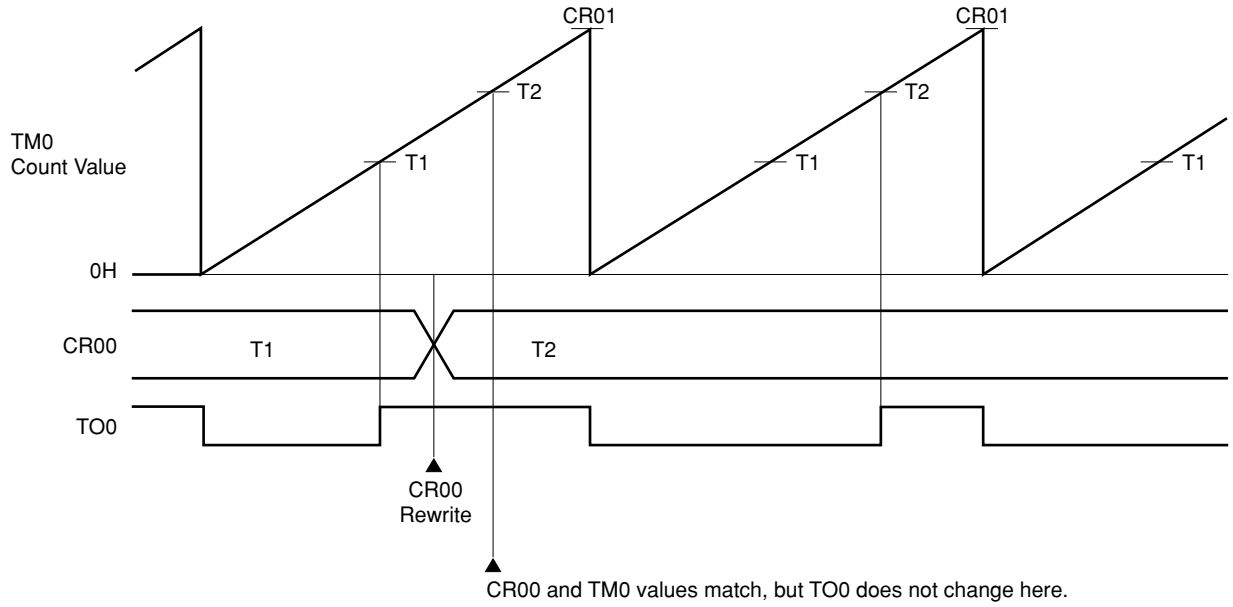
Remark ALV0 = 0

T = x/f_{XX} (x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048)

(2) Rewriting compare register (CR00)

The output level of the timer output (TO0) does not change even if the CR00 value matches the timer register 0 (TM0) value more than once during one PPG output cycle.

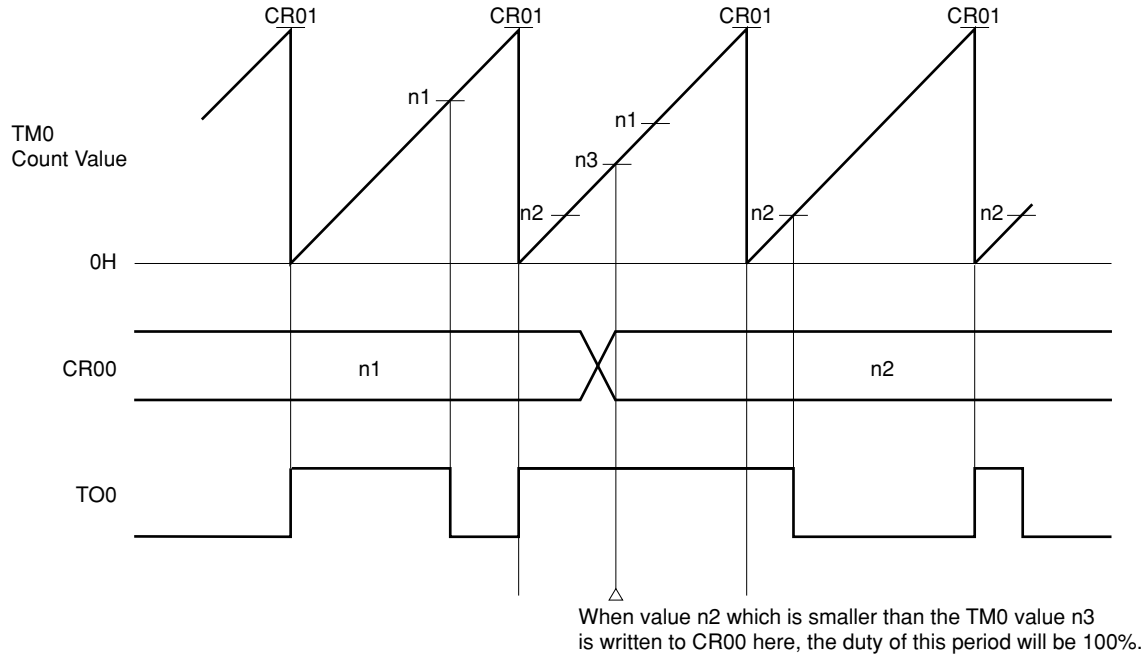
Figure 8-25 Example of Compare Register (CR00) Rewrite



Remark ALV0 = 1

If a value equal to or less than the TM0 value is written to CR00 before the CR00 and TM0 match, the duty of the PPG cycle will be 100%. CR00 rewriting should be performed by the interrupt due to a match between TM0 and CR00.

Figure 8-26 Example of 100% Duty With PPG Output



Remark ALV0 = 0

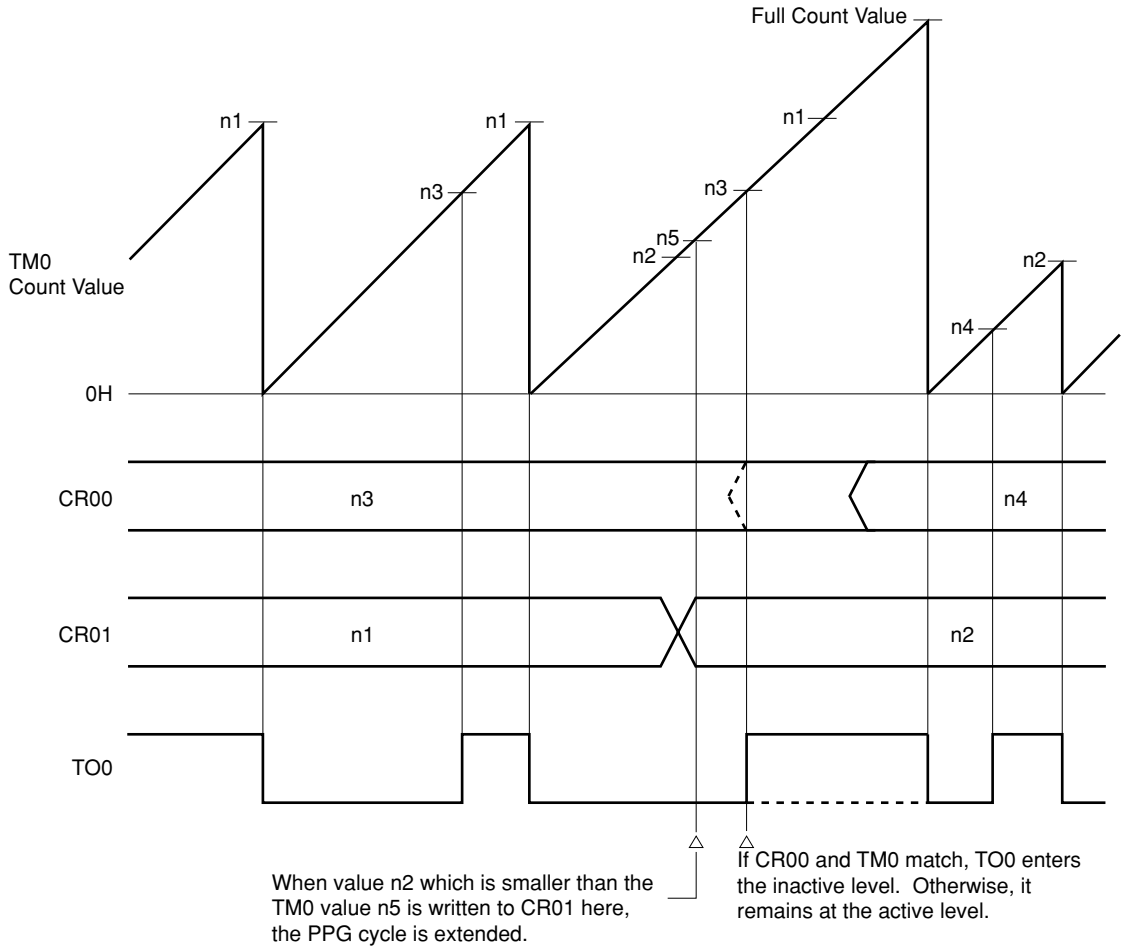
Caution If the PPG cycle is extremely short as compared with the time required to acknowledge an interrupt, the value of CR00 cannot be rewritten by interrupt processing that is performed on coincidence between TM0 and CR00. Use another method (for example, to poll the interrupt request flags by software with all the interrupts masked).

(3) Rewriting compare register (CR01)

If the current value of the CR01 is changed to a smaller value, and the CR01 value is made smaller than the timer register 0 (TM0) value, the PPG cycle at that time will be extended to the time equivalent to a full-count by TM0. If CR01 is rewritten after the compare register (CR00) and TM0 match, the output level at this time will be the inactive level until TM0 overflows and becomes 0, and will then return to normal PPG output.

If CR01 is rewritten before CR00 and TM0 match, the active level will be output until CR00 and TM0 match. If CR00 and TM0 match before TM0 overflows and becomes 0, the inactive level is output at that point. When TM0 overflows and becomes 0, the active level will be output, and normal PPG output will be restored. CR01 rewriting should be performed by the interrupt due to a match between TM0 and CR01, etc.

Figure 8-27 Example of Extended PPG Output Cycle



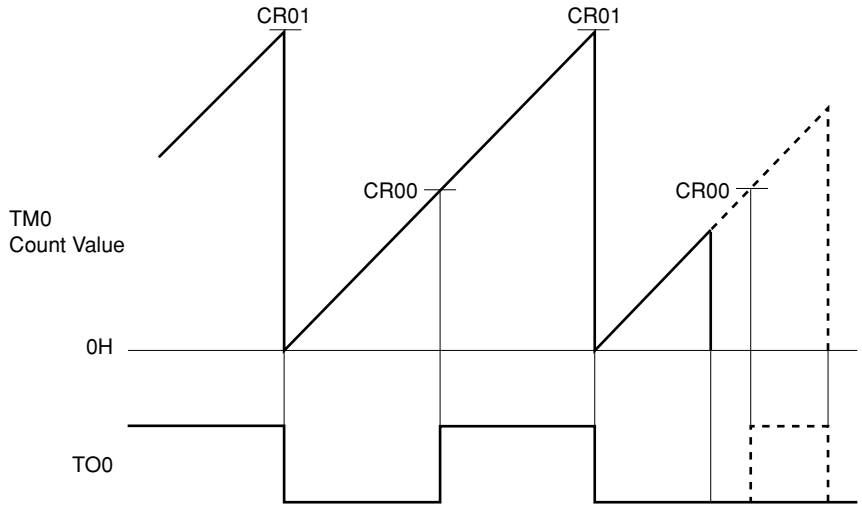
Remark ALV0 = 1

Caution If the PPG cycle is extremely short as compared with the time required to acknowledge an interrupt, the value of CR01 cannot be rewritten by interrupt processing that is performed on coincidence between the timer register (TM0) and compare register (CR01). Use another method (for example, to poll the interrupt request flags by software with all the interrupts masked).

(4) Stopping PPG output

If timer/counter 0 is stopped by clearing (to 0) the CE0 bit of the timer control register 0 (TMC0) during PPG signal output, the active level is output irrespective of the output level at the time it was stopped.

Figure 8-28 When Timer/Counter 0 is Stopped During PPG Signal Output



Caution The output level of the TOn (n = 0, 1) pin when timer output is disabled (ENTOn = 0: n = 0, 1) is the inverse of the value set in ALVn (n = 0, 1) bit. Caution is therefore required as the active level is output when timer output is disabled when the PPG output function has been selected.

8.7.5 Software Triggered One-Shot Pulse Output

In the software triggered one-shot pulse output mode, a one-shot pulse is output by software.

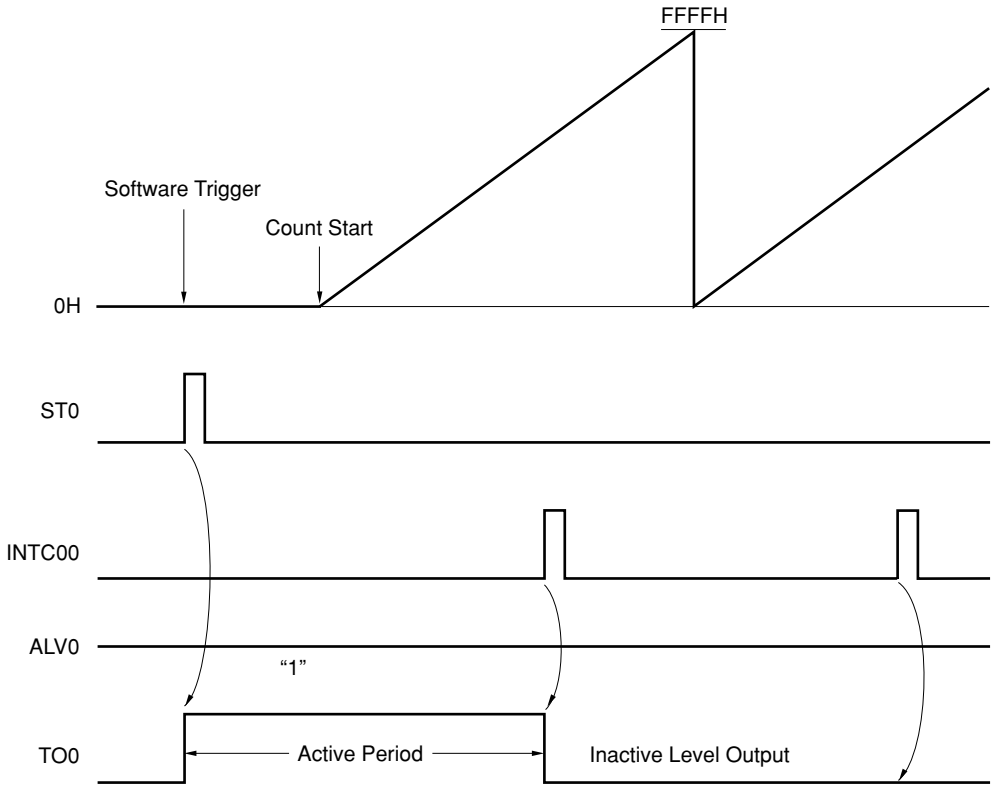
When the STn (n = 0/1) bit of the one-shot pulse output control register (OSPC) is set (to 1), timer output pin (TOn: n = 0, 1) is set to the active level. TOn then remains at the active level until the timer register 0 (TM0) value and the compare register (CR0n: n = 0, 1) value match, at which point TOn changes to the inactive level. TOn then remains at the inactive level until the STn bit is set again. TOn can also be set to the inactive level by setting (to 1) the RTn bit (n = 0/1), and in the same way, TOn remains at the inactive level until the STn bit is set again.

TO0 and TO1 can be controlled independently.

An example of software triggered one-shot pulse output is shown in Figure 8-29.

When timer/counter 0 is stopped by clearing (to 0) the CE0 bit of the TMC0, the level at the time was stopped is retained.

Figure 8-29 Example of Software Triggered One-Shot Pulse Output



Caution "1" should not be written to STn and RTn simultaneously.

8.8 EXAMPLES OF USE

8.8.1 Operation as Interval Timer (1)

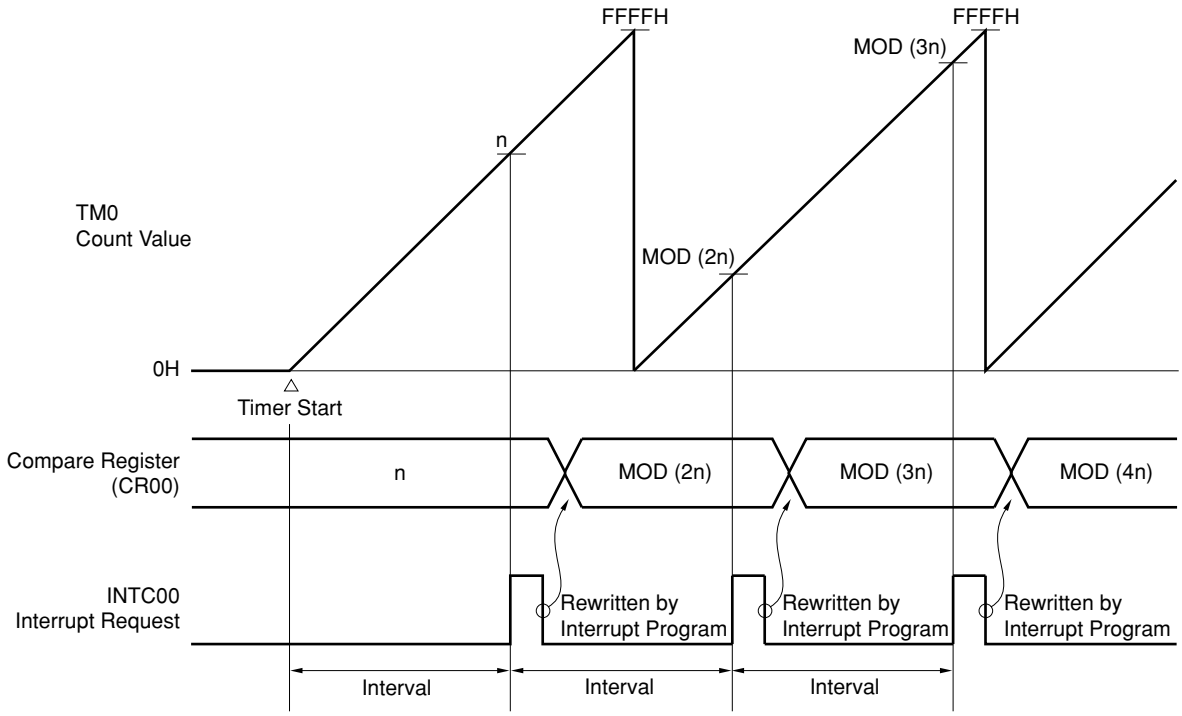
When timer register 0 (TM0) is made free-running and a fixed value is added to the compare register (CR0n: n = 0, 1) in the interrupt service routine, TM0 operates as an interval timer with the added fixed value as the cycle (see **Figure 8-30**).

This interval timer can count within the range shown in Table 8-1 (internal system clock fxx = 32 MHz).

Since TM0 has two compare registers, two interval timers with different cycles can be constructed.

The control register settings are shown in Figure 8-31, the setting procedure in Figure 8-32, and the processing in the interrupt service routine in Figure 8-33.

Figure 8-30 Interval Timer Operation (1) Timing



Remark Interval = n × 8/fxx, 1 ≤ n ≤ FFFFH

Figure 8-31 Control Register Settings for Interval Timer Operation (1)

Capture/compare control register 0 (CRC0)

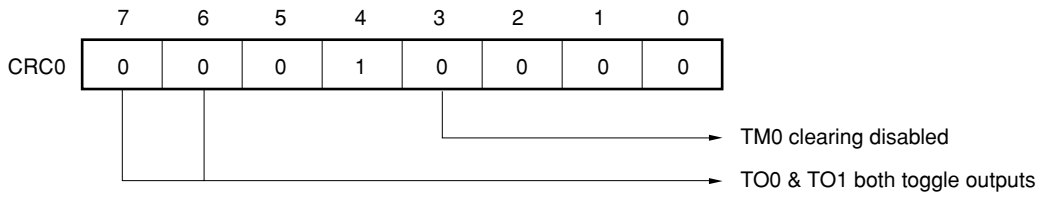


Figure 8-32 Interval Timer Operation (1) Setting Procedure

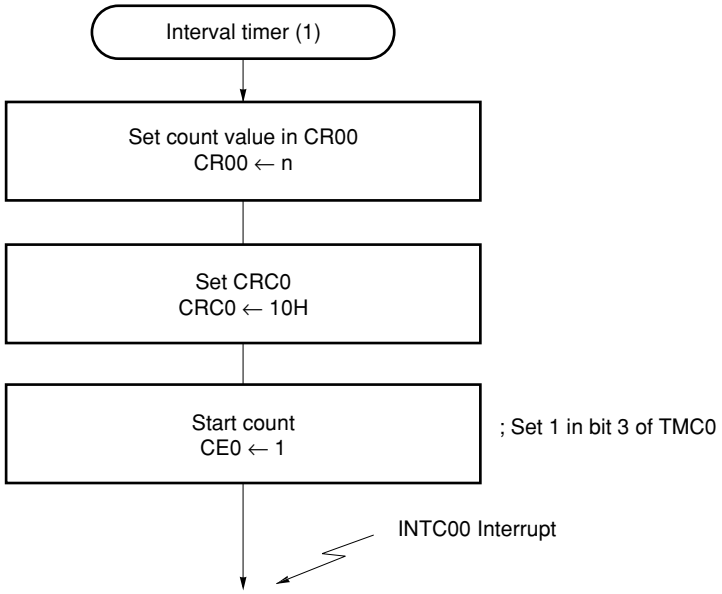
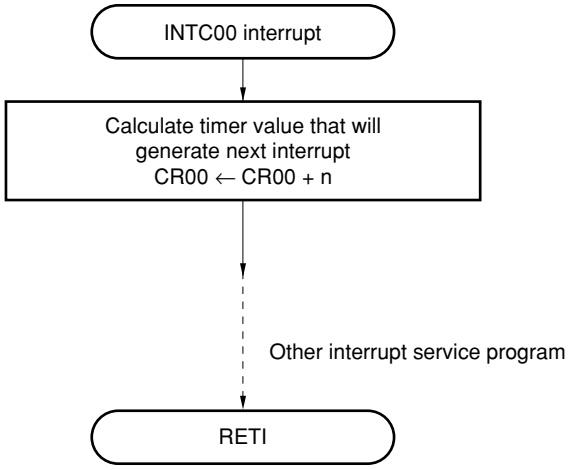


Figure 8-33 Interval Timer Operation (1) Interrupt Request Servicing



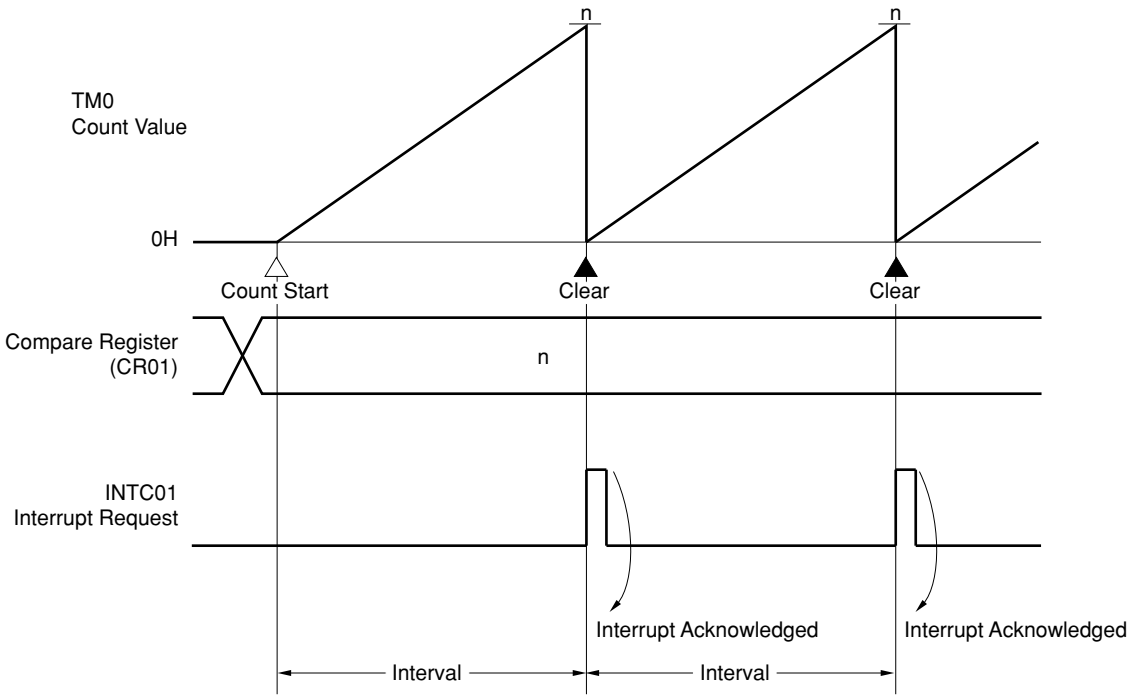
8.8.2 Operation as Interval Timer (2)

TM0 operates as an interval timer that generates interrupts repeatedly with the preset count time as the interval (see **Figure 8-34**).

This interval timer can count within the range shown in Table 8-1 (internal system clock fxx = 32 MHz).

The control register settings are shown in Figure 8-35, and the setting procedure in Figure 8-36.

Figure 8-34 Interval Timer Operation (2) Timing



Remark Interval = (n + 1) × 8/fxx, 0 ≤ n ≤ FFFFH

Figure 8-35 Control Register Settings for Interval Timer Operation (2)

Capture/compare control register 0 (CRC0)

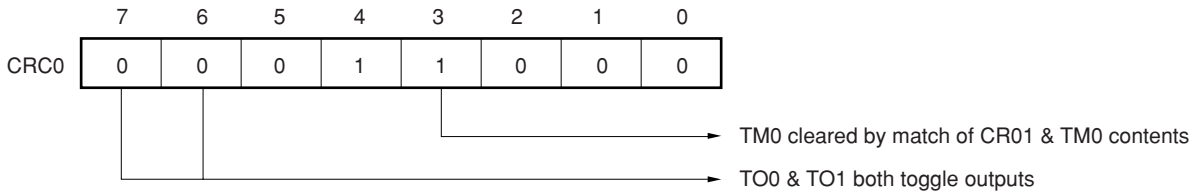
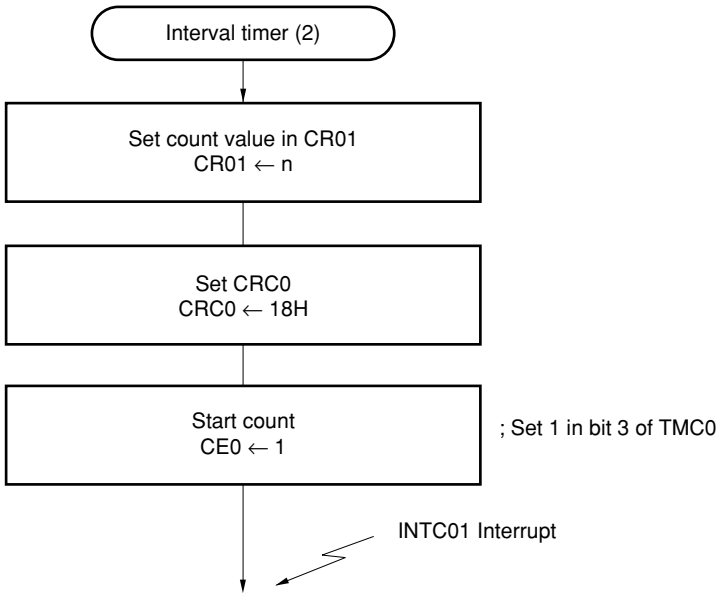


Figure 8-36 Interval Timer Operation (2) Setting Procedure



8.8.3 Pulse Width Measurement Operation

In pulse width measurement, the high-level or low-level width of external pulses input to the external interrupt request input pin (INTP3) is measured.

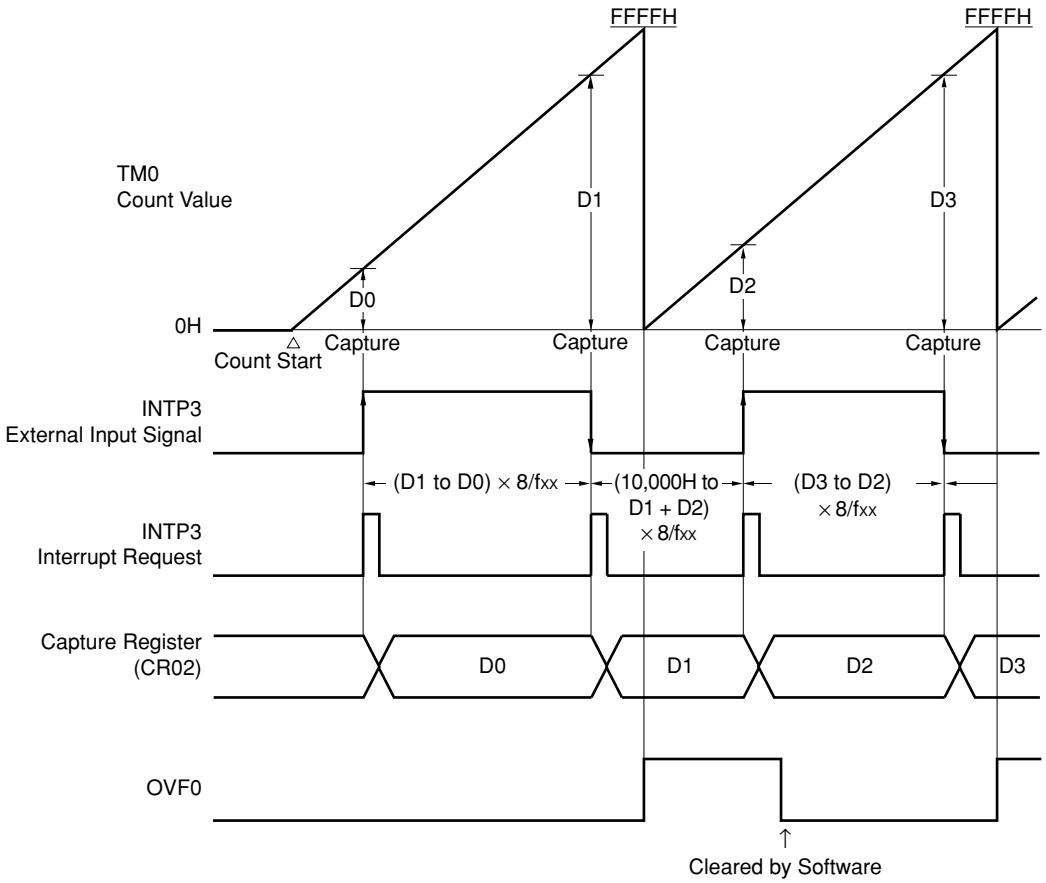
Both the high-level and low-level widths of pulses input to the INTP3 pin must be at least 3 system clocks (0.19 μs: f_{CLK} = 16 MHz); if shorter than this, the valid edge will not be detected and a capture operation will not be performed.

This pulse width measurement can be performed within the range shown in Table 8-3 (f_{CLK} = 16 MHz).

As shown in Figure 8-37, the timer register 0 (TM0) value being counted is fetched into the capture register (CR02) in synchronization with a valid edge (specified as both rising and falling edges) in the INTP3 pin input, and held there. The pulse width is obtained from the product of the difference between the TM0 count value (D_n) fetched into and held in the CR02 on detection of the nth valid edge and the count value (D_{n-1}) fetched and held on detection of valid edge n - 1, and the number of count clocks (x/fxx; x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048).

The control register settings are shown in Figure 8-38, and the setting procedure in Figure 8-39.

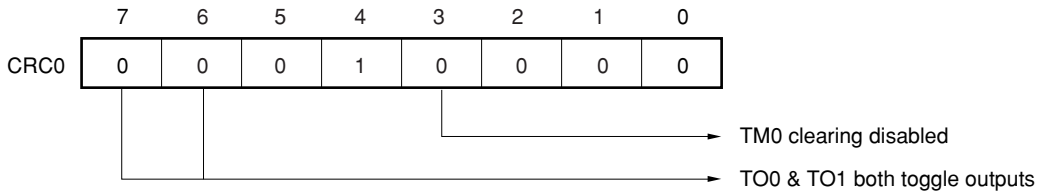
Figure 8-37 Pulse Width Measurement Timing



Remark Dn: TM0 count value (n = 0, 1, 2, ...)
x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

Figure 8-38 Control Register Settings for Pulse Width Measurement

(a) Capture/compare control register 0 (CRC0)



(b) External interrupt mode register 1 (INTM1)

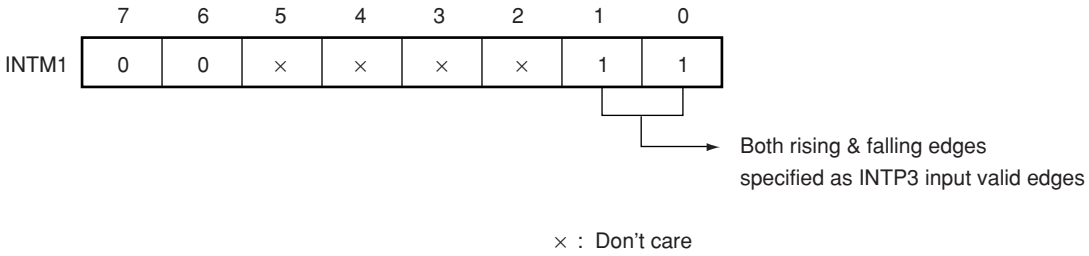


Figure 8-39 Pulse Width Measurement Setting Procedure

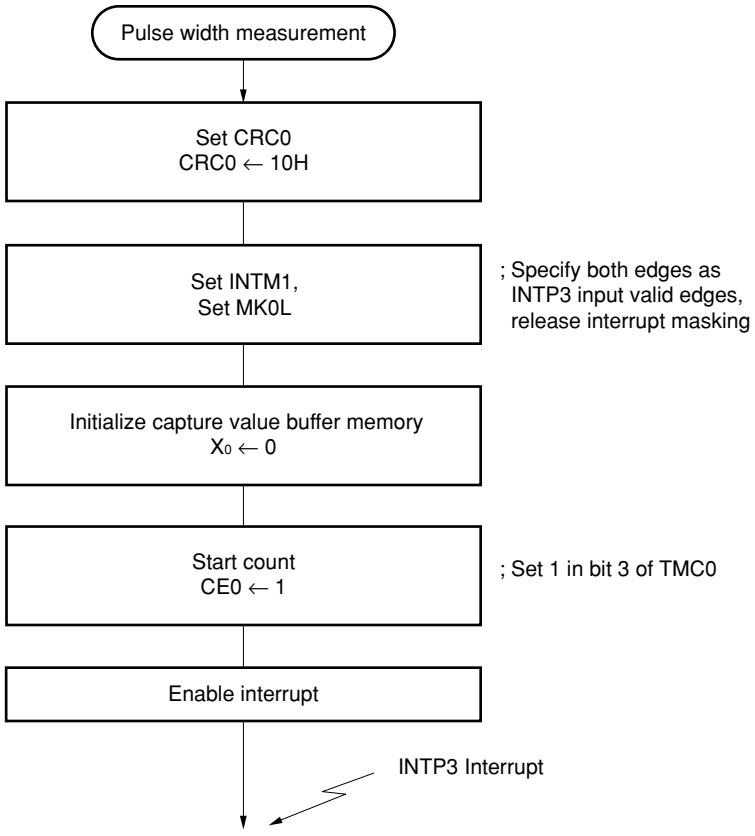
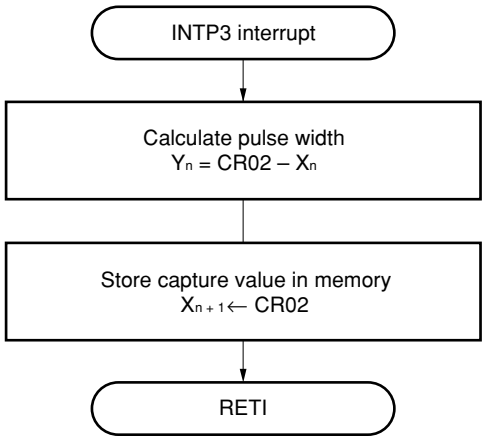


Figure 8-40 Interrupt Request Servicing that Calculates Pulse Width



8.8.4 Operation as PWM Output

In PWM output, pulses with the duty ratio determined by the value set in the compare register (CR0n: n = 0, 1) are output (see **Figure 8-41**).

This PWM output duty ratio can be varied in the range 1/65,536 to 65,535/65,536 in 1/65,536 units.

Since timer register 0 (TM0) has two compare registers, two different PWM signals can be output.

The control register settings are shown in Figure 8-42, the setting procedure in Figure 8-43, and the procedure for varying the duty in Figure 8-44.

Figure 8-41 Example of Timer/Counter 0 PWM Signal Output

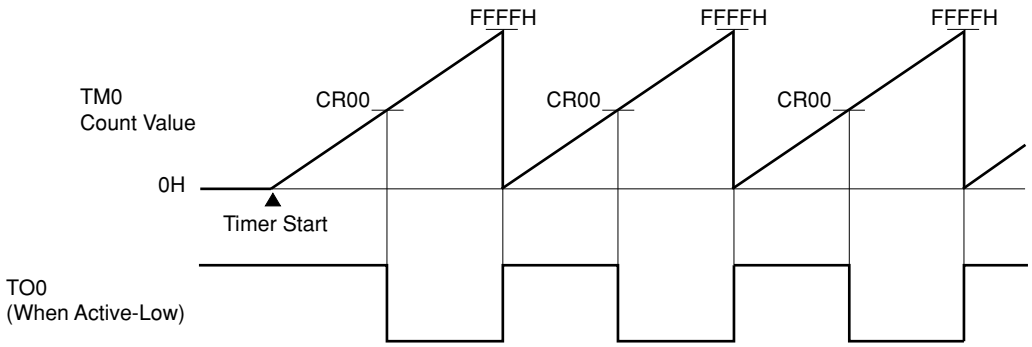
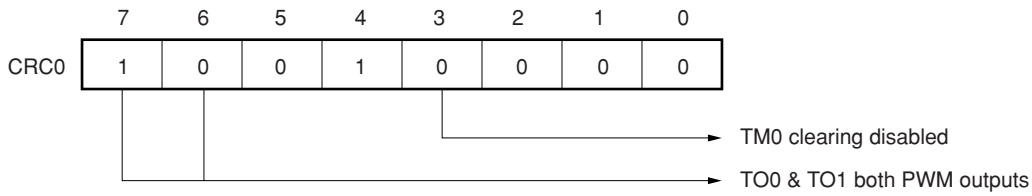
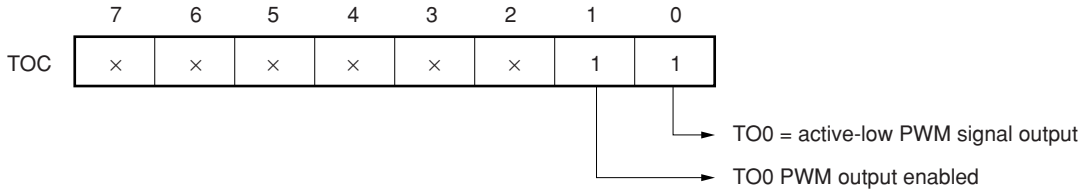


Figure 8-42 Control Register Settings for PWM Output Operation

(a) Capture/compare control register 0 (CRC0)



(b) Timer output control register (TOC)



(c) Port 3 mode control register (PMC3)

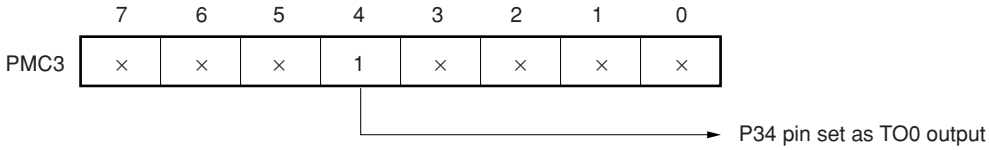


Figure 8-43 PWM Output Setting Procedure

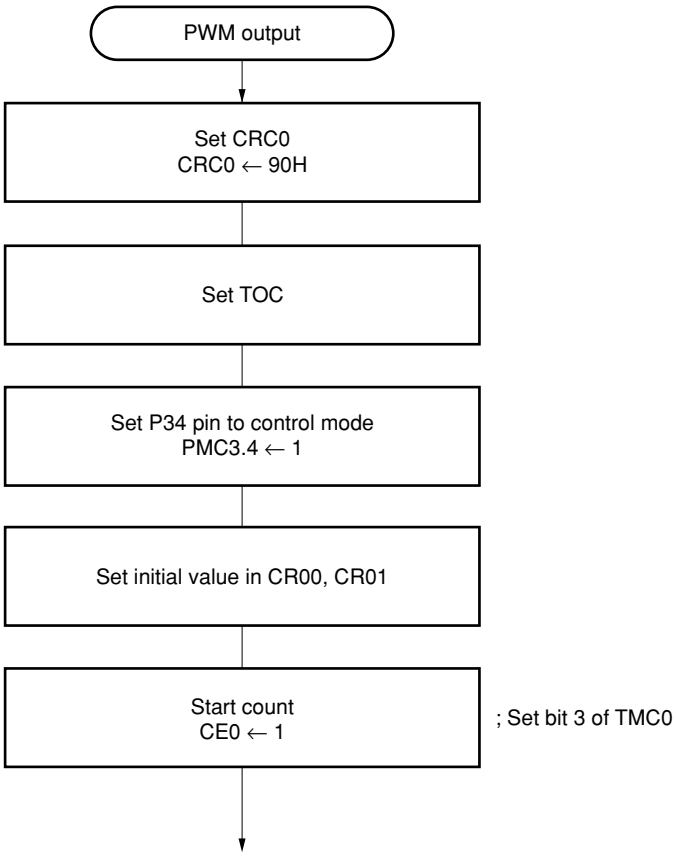
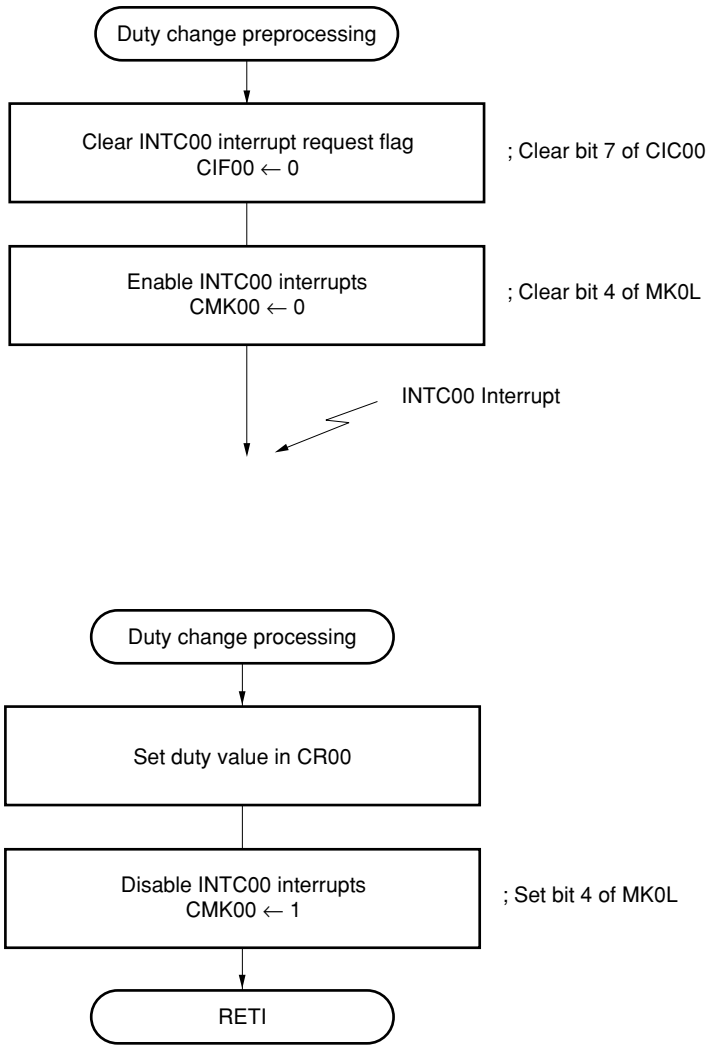


Figure 8-44 Changing PWM Output Duty



8.8.5 Operation as PPG Output

In PPG output, pulses with the cycle and duty ratio determined by the values set in the compare registers (CR0n: n = 0, 1) are output (see **Figure 8-45**).

The control register settings are shown in Figure 8-46, the setting procedure in Figure 8-47, and the procedure for varying the duty in Figure 8-48.

Figure 8-45 Example of Timer/Counter 0 PPG Signal Output

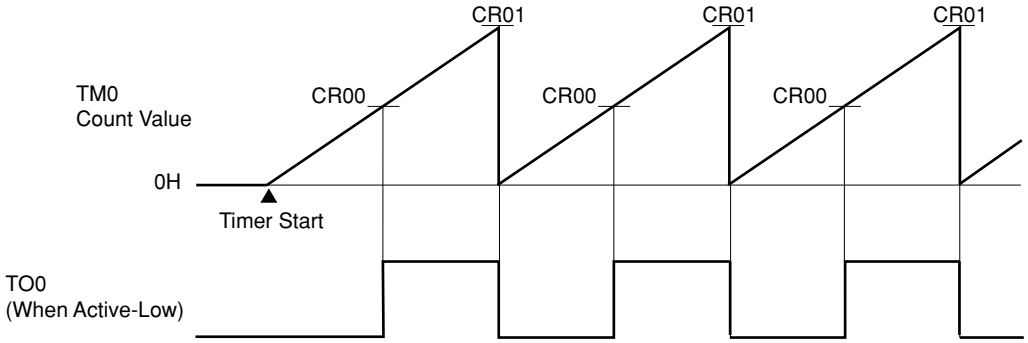
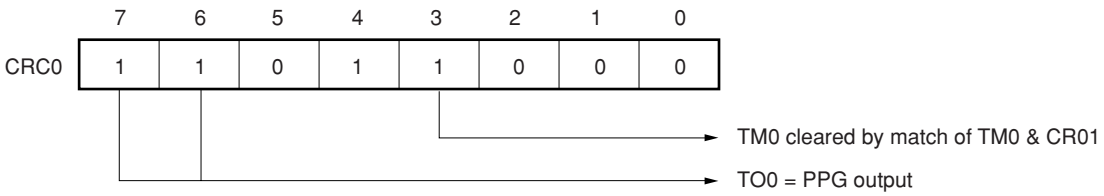
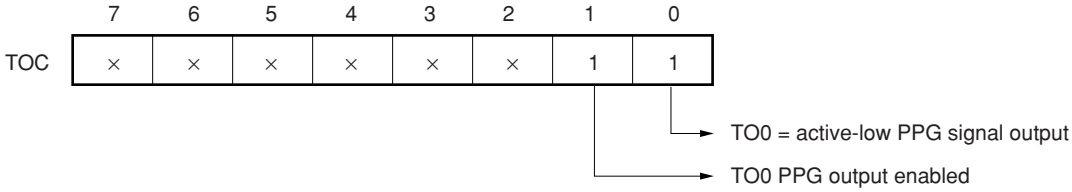


Figure 8-46 Control Register Settings for PPG Output Operation

(a) Capture/compare control register 0 (CRC0)



(b) Timer output control register (TOC)



(c) Port 3 mode control register (PMC3)

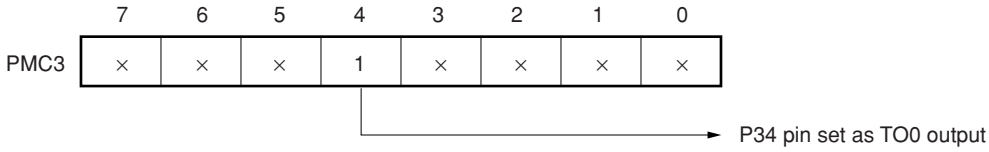


Figure 8-47 PPG Output Setting Procedure

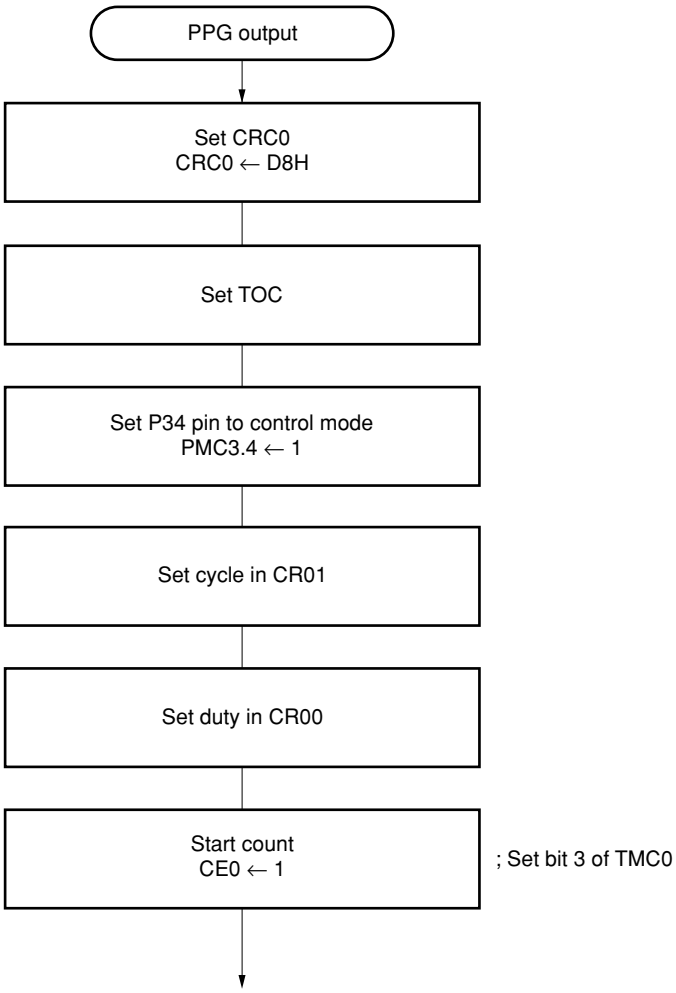
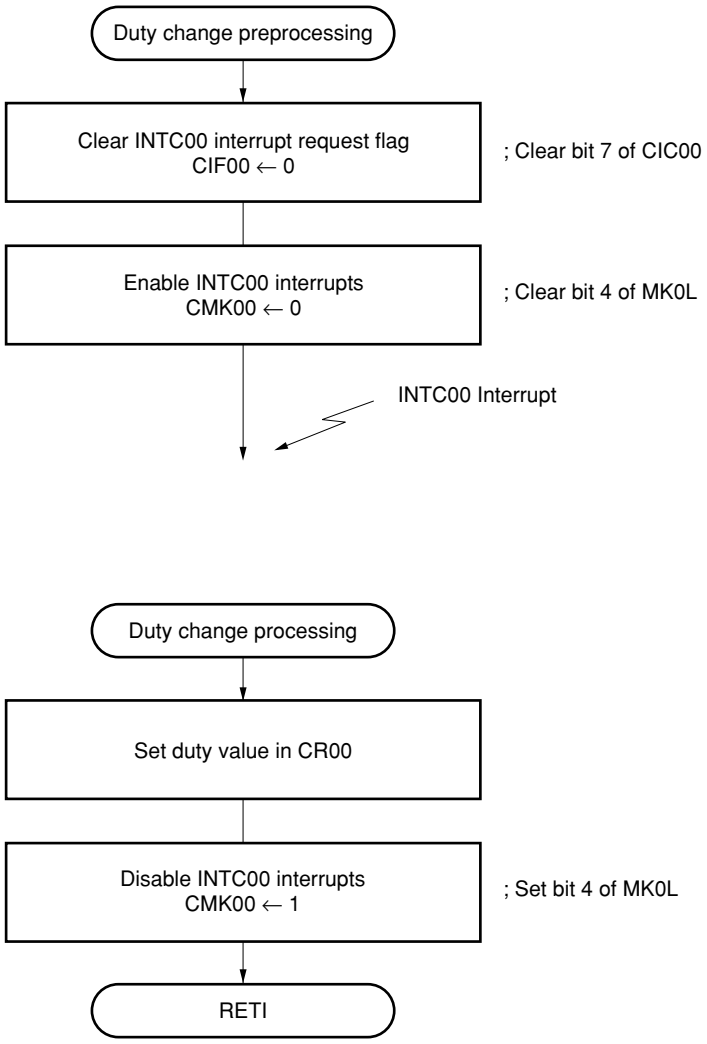


Figure 8-48 Changing PPG Output Duty



8.8.6 Example of Software Triggered One-Shot Pulse Output

In the software triggered one-shot pulse output mode, a one-shot pulse is output in response to a trigger activated by software (see **Figure 8-49**).

The control register settings are shown in Figure 8-50, and the setting procedure in Figure 8-51.

Figure 8-49 Example of Timer/Counter 0 One-Shot Pulse Output

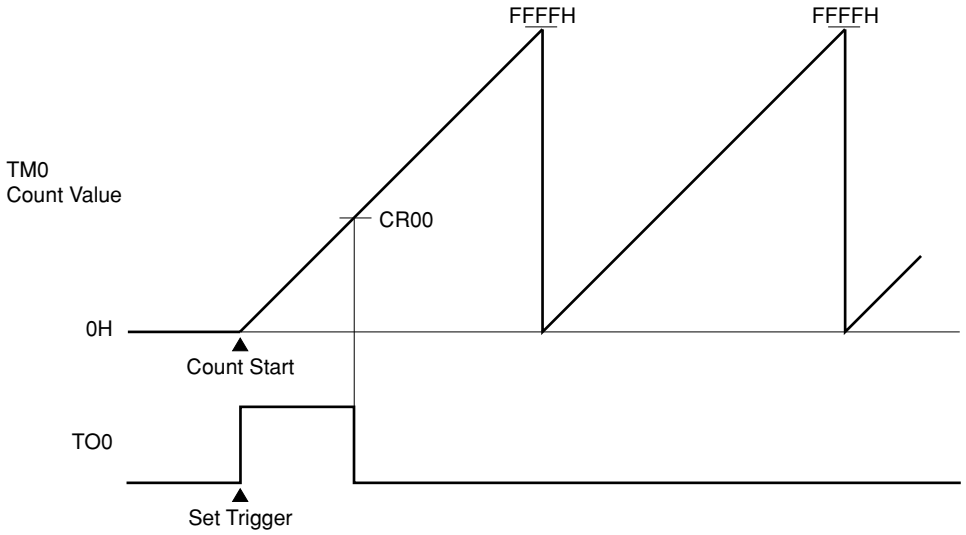
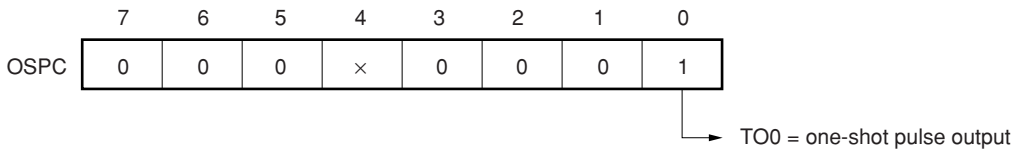
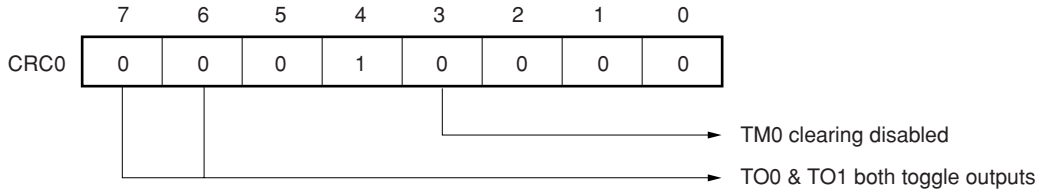


Figure 8-50 Control Register Settings for One-Shot Pulse Output

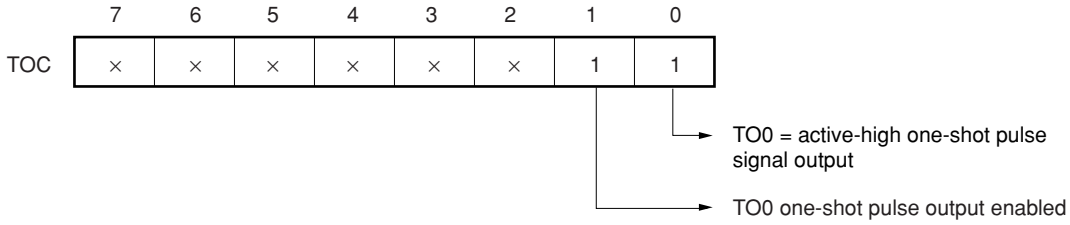
(a) One-shot pulse output control register (OSPC)



(b) Capture/compare control register 0 (CRC0)



(c) Timer output control register (TOC)



(d) Port 3 mode control register (PMC3)

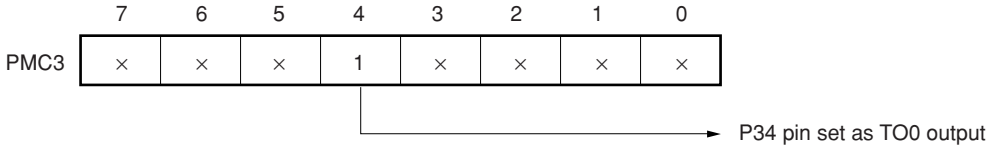
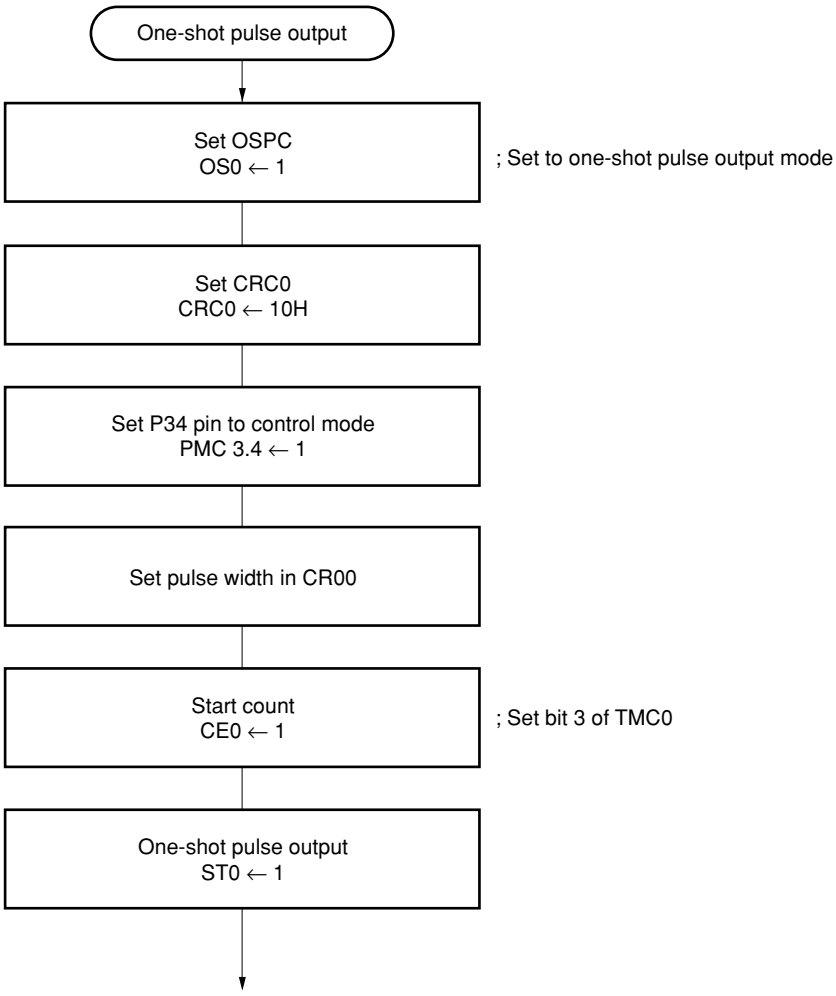


Figure 8-51 One-Shot Pulse Output Setting Procedure



8.9 CAUTIONS

(1) While timer/counter 0 is operating (while the CE0 bit of the timer control register 0 (TMC0) is set), malfunctioning may occur if the contents of the following registers are rewritten. This is because it is undefined which takes precedence in a contention the change in the hardware functions due to rewriting the register, or the change in the status because of the function before rewriting.
Therefore, be sure to stop the counter operation for the sake of safety before rewriting the contents of the following registers.

- Prescaler mode register 0 (PRM0)
- Capture/compare control register 0 (CRC0)
- Timer output control register (TOC)

(2) If the contents of the compare register (CR0n: n = 0 or 1) coincide with those of TM0 operation when an instruction that stops timer register 0 (TM0) operation is executed, the counting operation of TM0 stops, but an interrupt request is generated. In order not to generate the interrupt when stopping the operation of TM0, mask the interrupt in advance by using the interrupt mask register before stopping TM0.

Example

Program that may generate interrupt request

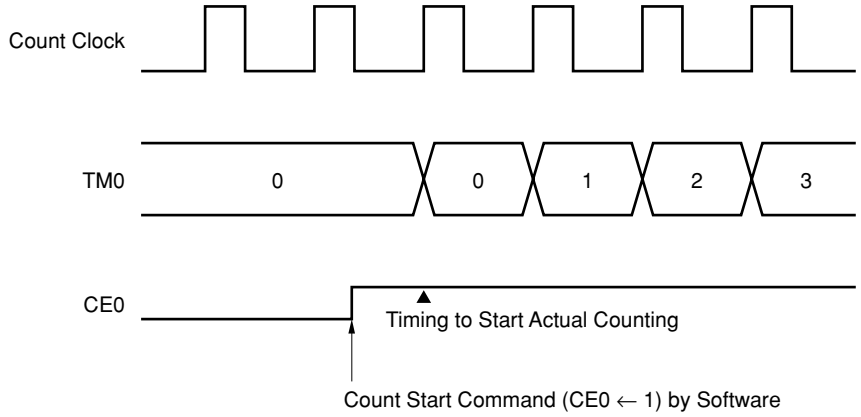
```
        ⋮
CLR1    CE0
OR      MK0L, #30H      ← Interrupt request
        ⋮                   from timer/counter 0
                            occurs between
                            these instructions
```

Program that does not generate interrupt request

```
        ⋮
OR      MK0L, #30H
CLR1    CE0             ← Disables interrupt
CLR1    CIF00               from timer/counter 0
CLR1    CIF01           ← Clears interrupt request
        ⋮                   flag for timer/counter 0
```

(3) Up to 1 count clock is required after an operation to start timer/counter 0 (CE0 ← 1) has been performed before timer/counter 0 actually starts (refer to **Figure 8-52**).
For example, when using timer/counter 0 as an interval timer, the first interval time is delayed by up to 1 clock. The second and those that follow are at the specified interval.

Figure 8-52 Operation When Counting is Started



(4) While an instruction that writes data to the compare register (CR0n: n = 0, 1) is executed, coincidence between CR0n, to which the data is to be written, and timer register 0 (TM0) is not detected. For example, if the contents of CR0n do not change before and after the writing, the interrupt request is not generated even if the value of TM0 coincides with the value of CR0n, nor does the timer output (TOn: n = 0, 1) change.
Write data to CR0n when timer/counter 0 is executing counting operation, in the timing that the contents of TM0 do not coincide with the value of CR0n before and after writing (e.g., immediately after an interrupt request has been generated because TM0 and CR0n have coincided).

(5) Coincidence between timer register 0 (TM0) and compare register (CR0n: n = 0, 1) is detected only when TM0 is incremented. Therefore, the interrupt request is not generated even if the same value as TM0 is written to CR0n, and the timer output (TOn: n = 0, 1) does not change.

(6) If the PPG cycle is extremely short as compared with the time required to acknowledge an interrupt, the value of the CR0n cannot be rewritten by interrupt processing that is performed on coincidence between the timer register 0 (TM0) and the compare register (CR0n: n = 0, 1). Use another method (for example, to poll the interrupt request flags by software with all the interrupts masked).

(7) The output level of the TOn (n = 0, 1) when the timer output is disabled (ENTOn = 0: n = 0, 1) is the reverse value of the value set to the ALVn (n = 0, 1) bit. Note, therefore, that an active level is output when the timer output is disabled with the PWM output function or PPG output function selected.

(8) If the value of the timer register is read under the condition indicated by "×" in Table 8-10, the read value may be illegal. Do not read the timer register under condition "×".

Table 8-10 Limits of Reading Timer Register

(√: Can be read, ×: Must not be read)

Timer Count Clock \ f_{CLK}	$f_{XX}/2$	$f_{XX}/4$	$f_{XX}/8$	$f_{XX}/16$
$f_{XX}/8$	√	√	×	×
$f_{XX}/16$	√	√	√	×
f_{XX}/n	√	√	√	√

Remarks 1. f_{XX}: Oscillation frequency
2. f_{CLK}: Internal system clock frequency
3. n = 32, 64, 128, 256, 512, 1,024, 2,048

(9) When timer/counter 0 is used as an external event counter, it is not possible to distinguish between the case where there is no valid edge input at all and the case where there is a single valid edge input, using the timer register 0 (TM0) alone (refer to **Figure 8-53**), since the contents of TM0 are 0 in both cases. If it is necessary to make this distinction, the INTP3 interrupt request flag should be used. To make a distinction, use the interrupt request flag of INTP3, as shown in Figure 8-54.

Figure 8-53 Example of the Case Where the External Event Counter Does Not Distinguish Between One Valid Edge Input and No Valid Edge Input

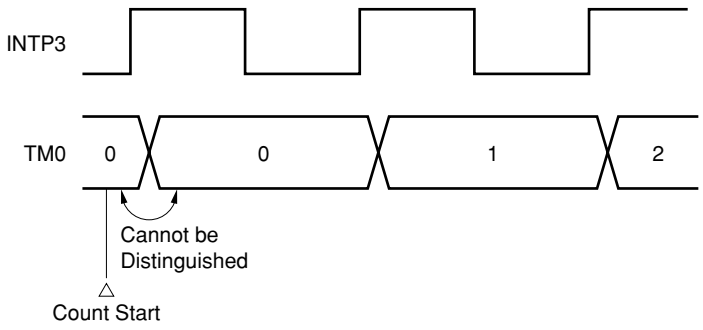
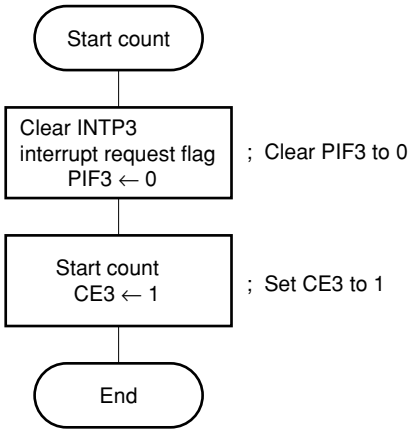
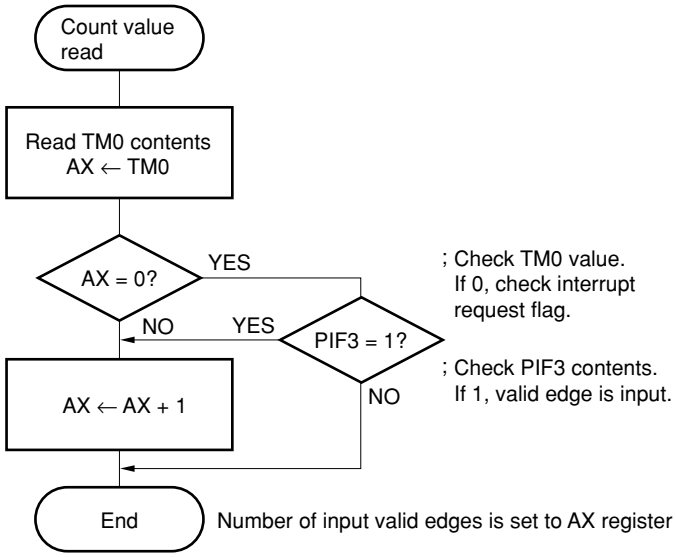


Figure 8-54 To Distinguish Whether One or No Valid Edge Has Been Input with External Event Counter

(a) Processing on starting counting



(b) Processing on reading count value

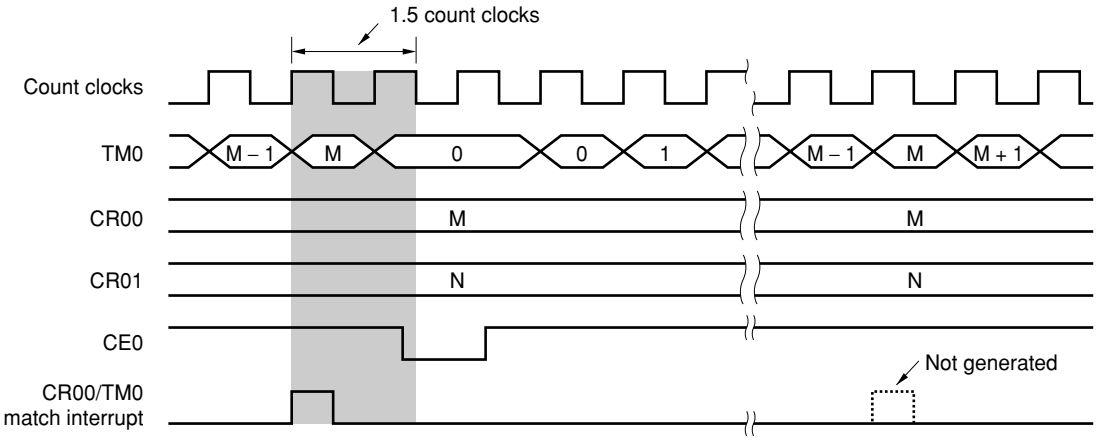


★ (10) If the count operation of TM0 stops at the timing at which compare register (CR00) and timer register 0 (TM0) match, the CR00/TM0 match interrupt may not be generated after timer/counter 0 is next started.

If the TM0 count operation is stopped within 1.5 count clocks after a match between CR00 and TM0, the first match interrupt after timer/counter 0 is next started will not be generated. The second and subsequent interrupts operate normally. Note that the timer output is unaffected by this bug.

This bug occurs because the timer interrupt controller inadvertently masks interrupts if timer/counter 0 is stopped in the period indicated by the shaded area in the figure below.

The interrupt controller is initialized by an overflow of timer/counter 0 or a match between CR01 and TM0.



Remark M < N

Do not stop timer/counter 0 within 1.5 count clocks after a match between CR00 and TM0.

Disable all interrupt requests (including macro servicing), read the value of the timer to be stopped, and wait until at least 1.5 count clocks have elapsed after a match between CR00 and TM0 before stopping timer/counter 0.

CHAPTER 9 TIMER/COUNTER 1

9.1 FUNCTIONS

Timer/counter 1 is 16-bit or 8-bit timer/counter.

In addition to its basic functions of interval timer, pulse width measurement and event counter, timer/counter 1 can be used as a real-time output port output trigger generation timer.

(1) Interval timer

Generates internal interrupts at preset intervals.

Table 9-1 Timer/Counter 1 Intervals

Minimum Interval	Maximum Interval	Resolution
8/fxx (0.25 μs)	$2^{16} \times 8$/fxx (16.40 ms)	8/fxx (0.25 μs)
16/fxx (0.50 μs)	$2^{16} \times 16$/fxx (32.80 ms)	16/fxx (0.50 μs)
32/fxx (1.00 μs)	$2^{16} \times 32$/fxx (65.50 ms)	32/fxx (1.00 μs)
64/fxx (2.00 μs)	$2^{16} \times 64$/fxx (131 ms)	64/fxx (2.00 μs)
128/fxx (4.00 μs)	$2^{16} \times 128$/fxx (262 ms)	128/fxx (4.00 μs)
256/fxx (8.00 μs)	$2^{16} \times 256$/fxx (524 ms)	256/fxx (8.00 μs)
512/fxx (16.00 μs)	$2^{16} \times 512$/fxx (1.05 s)	512/fxx (16.00 μs)
1,024/fxx (32.00 μs)	$2^{16} \times 1{,}024$/fxx (2.10 s)	1,024/fxx (32.00 μs)
2,048/fxx (64.00 μs)	$2^{16} \times 2{,}048$/fxx (4.19 s)	2,048/fxx (64.00 μs)

( ): When fxx = 32 MHz

(2) Pulse width measurement

Detects the pulse width of the signal input to the external interrupt request input pin INTP0.

Table 9-2 Timer/Counter 1 Pulse Width Measurement Range

Measurable Pulse Width [Note]	Resolution
$8/f_{XX}$ to $2^{16} \times 8/f_{XX}$ (0.25 μs) (16.40 ms)	$8/f_{XX}$ (0.25 μs)
$16/f_{XX}$ to $2^{16} \times 16/f_{XX}$ (0.50 μs) (32.80 ms)	$16/f_{XX}$ (0.50 μs)
$32/f_{XX}$ to $2^{16} \times 32/f_{XX}$ (1.00 μs) (65.50 ms)	$32/f_{XX}$ (1.00 μs)
$64/f_{XX}$ to $2^{16} \times 64/f_{XX}$ (2.00 μs) (131 ms)	$64/f_{XX}$ (2.00 μs)
$128/f_{XX}$ to $2^{16} \times 128/f_{XX}$ (4.00 μs) (262 ms)	$128/f_{XX}$ (4.00 μs)
$256/f_{XX}$ to $2^{16} \times 256/f_{XX}$ (8.00 μs) (524 ms)	$256/f_{XX}$ (8.00 μs)
$512/f_{XX}$ to $2^{16} \times 512/f_{XX}$ (16.00 μs) (1.05 s)	$512/f_{XX}$ (16.00 μs)
$1,024/f_{XX}$ to $2^{16} \times 1,024/f_{XX}$ (32.00 μs) (2.10 s)	$1,024/f_{XX}$ (32.00 μs)
$2,048/f_{XX}$ to $2^{16} \times 2,048/f_{XX}$ (64.00 μs) (4.19 s)	$2,048/f_{XX}$ (64.00 μs)

( ): When f_{XX} = 32 MHz

Note The minimum pulse width that can be measured changes depending on the sampling clock selected by the sampling clock select register (SCS0). The minimum pulse width that can be measured is the value in the table below or above, whichever is greater.

Sampling Clock		Minimum Pulse Width
f_{CLK}	$f_{CLK} = f_{XX}/2$	$4/f_{CLK} = 8/f_{XX}$ (0.25 μs)
	$f_{CLK} = f_{XX}/4$	$4/f_{CLK} = 16/f_{XX}$ (0.50 μs)
	$f_{CLK} = f_{XX}/8$	$4/f_{CLK} = 32/f_{XX}$ (1.00 μs)
	$f_{CLK} = f_{XX}/16$	$4/f_{CLK} = 64/f_{XX}$ (2.00 μs)
$f_{XX}/64$		$256/f_{XX}$ (8.00 μs)
$f_{XX}/28$		$512/f_{XX}$ (16.00 μs)
$f_{XX}/256$		$1,024/f_{XX}$ (32.00 μs)

(3) External event counter

Counts the clock pulses input from the external interrupt request input pin (INTP0).

The clocks that can be input to timer/counter 1 are shown in Table 9-3.

Table 9-3 Timer/Counter 1 Pulse Width Measurement Time

( ): When f_{CLK} = 16 MHz and f_{XX} = 32 MHz

Sampling Clock Note		When Counting One Edge	When Counting Both Edges
f_{CLK}	Maximum frequency	f_{CLK}/8 (2.00 MHz)	f_{CLK}/8 (2.00 MHz)
	Minimum pulse width (High and low levels)	4/f_{CLK} (0.25 μs)	4/f_{CLK} (0.25 μs)
f_{XX}/64	Maximum frequency	f_{XX}/512 (62.50 kHz)	f_{XX}/512 (62.50 kHz)
	Minimum pulse width (High and low levels)	256/f_{XX} (8.00 μs)	256/f_{XX} (8.00 μs)
f_{XX}/128	Maximum frequency	f_{XX}/1,024 (31.30 kHz)	f_{XX}/1,024 (31.30 kHz)
	Minimum pulse width (High and low levels)	512/f_{XX} (16.00 μs)	512/f_{XX} (16.00 μs)
f_{XX}/256	Maximum frequency	f_{XX}/2,048 (15.60 kHz)	f_{XX}/2,048 (15.60 kHz)
	Minimum pulse width (High and low levels)	1,024/f_{XX} (32.00 μs)	1,024/f_{XX} (32.00 μs)

Note Selected by means of the sampling clock selection register (SCS0)

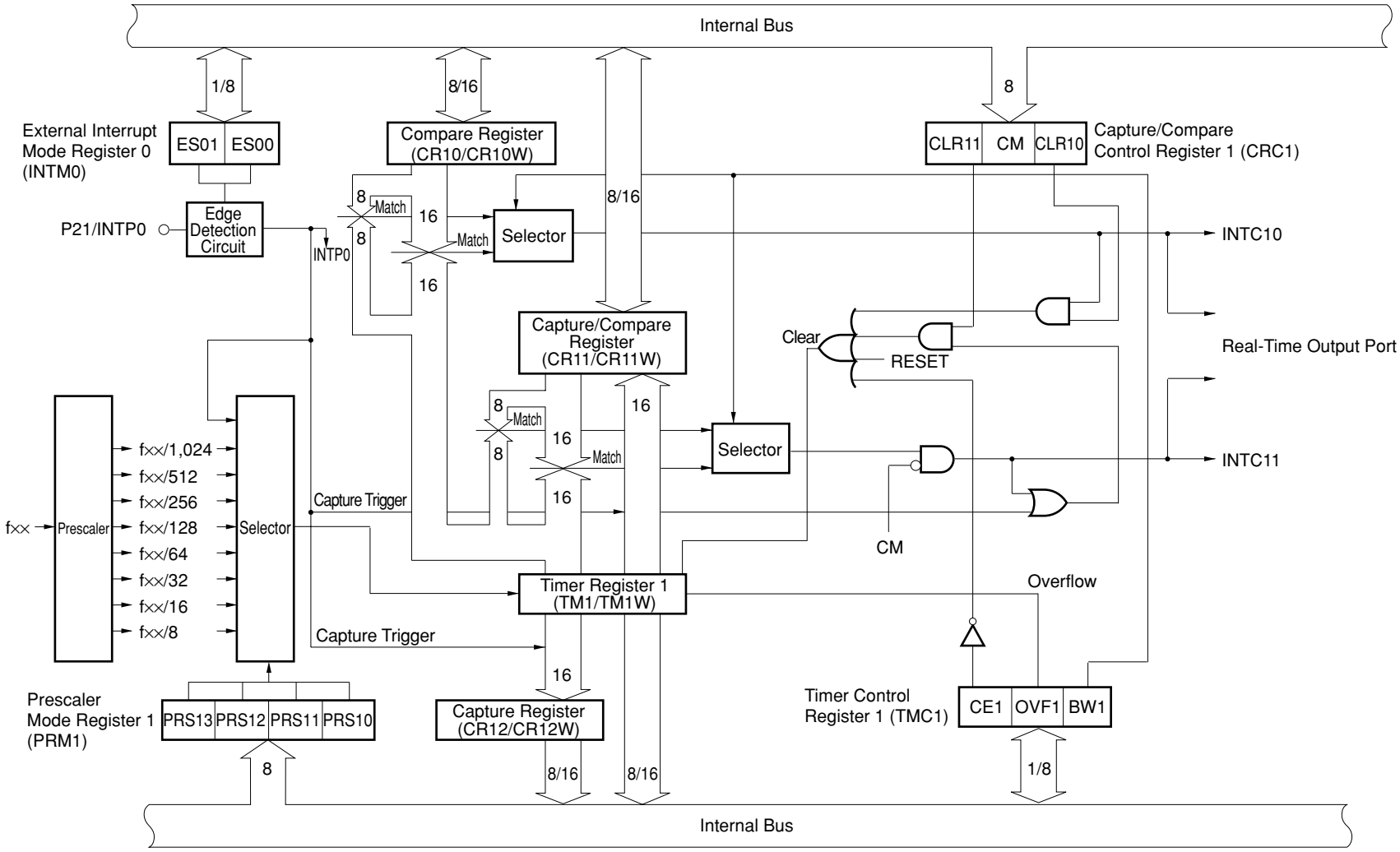
9.2 CONFIGURATION

Timer/counter 1 consists of the following registers:

- Timer register (TM1/TM1W) × 1
- Compare register (CR10/CR10W) × 1
- Capture/compare register (CR11/CR11W) × 1
- Capture register (CR12/CR12W) × 1

The block diagram of timer/counter 1 is shown in Figure 9-1.

Figure 9-1 Timer/Counter 1 Block Diagram



(1) Timer register 1 (TM1/TM1W)

TM1//TM1W is a timer register that counts up using the count clock specified by the low-order 4 bits of prescaler mode register 1 (PRM1).

The count operation can be specified to stop or enable, and an 8-bit operation mode (TM1)/16-bit operation mode (TM1W) can be selected, by means of timer control register 1 (TMC1).

TM1/TM1W can be read only with an 8/16-bit manipulation instruction. When $\overline{\text{RESET}}$ is input, TM1/TM1W is cleared to 00H and the count is stopped.

Caution If the value of the timer register is read under the condition indicated by "×" in Table 9-4, the read value may be illegal. Do not read the timer register under condition "×".

Table 9-4 Limits of Reading Timer Register

(√: Can be read, ×: Must not be read)

f_{CLK} / Timer Count Clock	$f_{XX}/2$	$f_{XX}/4$	$f_{XX}/8$	$f_{XX}/16$
$f_{XX}/8$	√	√	×	×
$f_{XX}/16$	√	√	√	×
f_{XX}/n	√	√	√	√

Remarks 1. f_{XX}: Oscillation frequency

2. f_{CLK}: Internal system clock frequency

3. n = 32, 64, 128, 256, 512, 1,024, 2,048

(2) Compare register (CR10/CR10W)

CR10/CR10W is an 8/16-bit register that holds the value that determines the interval timer operation cycle.

If the contents of the CR10/CR10W match the values of TM1/TM1W, an interrupt request (INTC10) is generated. This match signal is also a real-time output port trigger signal. Also, the count value can be cleared by a match.

This compare register operates as CR10 in the 8-bit operating mode, and CR10W in the 16-bit operating mode.

CR10/CR10W can be read or written to with an 8/16-bit manipulation instruction. The contents of this register are undefined after $\overline{\text{RESET}}$ input.

(3) Capture/compare register (CR11/CR11W)

CR11/CR11W is an 8/16-bit register that can be specified as a compare register for detecting a match with the TM1/TM1W count value or a capture register for capturing the TM1/TM1W count value according to the setting of capture/compare control register 1 (CRC1).

This capture/compare register operates as CR11 in the 8-bit operating mode, and CR11W in the 16-bit operating mode.

CR11/CR11W can be read or written to with an 8/16-bit manipulation instruction. The contents of this register are undefined after $\overline{\text{RESET}}$ input.

(a) When specified as compare register

CR11/CR11W functions as an 8/16-bit register that holds the value that determines the interval timer operation cycle.

An interrupt request (INTC11) is generated by a match between the contents of the CR11/CR11W register and the contents of TM1/TM1W.

Also, the count value can be cleared by a match. This match signal is also a real-time output port trigger signal.

(b) When specified as capture register

CR11/CR11W functions as an 8/16-bit register that captures the contents of TM1/TM1W in synchronization with the input of a valid edge (capture trigger) on the external interrupt request input pin (INTP0).

The contents of the CR11/CR11W are retained until the next capture trigger is generated. TM1/TM1W can be cleared after a capture operation.

(4) Capture register (CR12/CR12W)

CR12/CR12W is an 8/16-bit register that captures the contents of TM1/TM1W.

The capture operation is synchronized with the input of a valid edge (capture trigger) on the external interrupt request input pin (INTP0). The contents of the CR12/CR12W are retained until the next capture trigger is generated.

This capture/compare register operates as CR12 in the 8-bit operating mode, and CR12W in the 16-bit operating mode.

CR12/CR12W can be read only with an 8/16-bit manipulation instruction. $\overline{\text{RESET}}$ input clears this register to 0000H.

(5) Edge detection circuit

The edge detection circuit detects an external input valid edge.

When the valid edge set by external interrupt mode register 0 (INTM0) is detected in the INTP0 pin input, the external interrupt request (INTP0), a capture trigger and a count clock of the external event are generated (see **Figure 21-1** for details of the INTM0).

(6) Prescaler

The prescaler generates the count clock from the internal system clock. The clock generated by this prescaler is selected by the selector, and is used as the count clock by the timer register 1 (TM1/TM1W) to perform count operations.

(7) Selector

The selector selects a signal resulting from dividing the internal clock or the edge detected by the edge detection circuit as the count clock of timer register 1 (TM1/TM1W).

9.3 TIMER/COUNTER 1 CONTROL REGISTERS

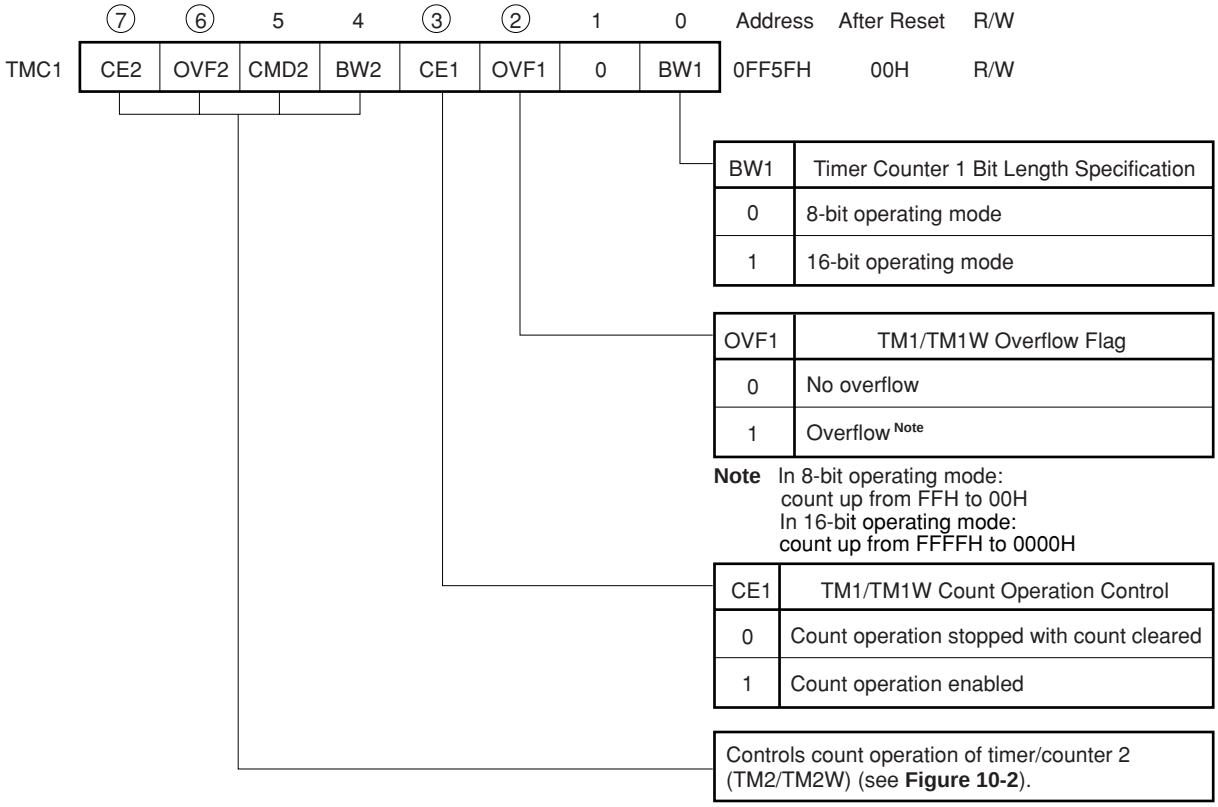
(1) Timer control register 1 (TMC1)

TMC1 controls the timer/counter 1, TM1/TM1W, count operation by the low-order 4 bits (the high-order 4 bits control the count operation of timer/counter 2, TM2/TM2W).

TMC1 can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. The format of the TMC1 is shown in Figure 9-2.

$\overline{\text{RESET}}$ input clears TMC1 to 00H.

Figure 9-2 Timer Control Register 1 (TMC1) Format



Remark The OVF1 bit is reset by software only.

(2) Prescaler mode register 1 (PRM1)

The count clock of PRM1 to timer/counter 1, TM1/TM1W, is specified by the low-order 4 bits (the high-order 4 bits specify the count clock to timer/counter 2, TM2/TM2W).

PRM1 can be read or written to with an 8-bit manipulation instruction. The format of the PRM1 is shown in Figure 9-3. $\overline{\text{RESET}}$ input sets PRM1 to 11H.

Figure 9-3 Prescaler Mode Register 1 (PRM1) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PRM1	PRS23	PRS22	PRS21	PRS20	PRS13	PRS12	PRS11	PRS10	0FF5EH	11H	R/W

(f_{XX} = 32MHz)

PRS13	PRS12	PRS11	PRS10	16-Bit Timer/Counter 1 TM1/TM1W Count Clock Specification	
				Count Clock [Hz] Specification	Resolution [μs]
0	0	0	0	Setting prohibited	–
0	0	0	1	f_{XX}/8	0.25
0	0	1	0	f_{XX}/16	0.50
0	0	1	1	f_{XX}/32	1.00
0	1	0	0	f_{XX}/64	2.00
0	1	0	1	f_{XX}/128	4.00
0	1	1	0	f_{XX}/256	8.00
0	1	1	1	f_{XX}/512	16.00
1	0	0	0	f_{XX}/1,024	32.00
1	0	0	1	f_{XX}/2,048	64.00
1	1	1	1	External clock (INTP0)	–
Other than the above				Setting prohibited	

Specifies count clock to TM2/TM2W of timer/counter 2 (see **Figure 10-3**).

Remark f_{XX}: X1 input frequency or oscillation frequency

(3) Capture/compare control register 1 (CRC1)

The CRC1 specifies the operation of the capture/compare register (CR11/CR11W) and the enabling condition for a timer register 1 (TM1/TM1W) clear operation.

CRC1 can be read or written to with an 8-bit manipulation instruction. The format of the CRC1 is shown in Figure 9-4.

$\overline{\text{RESET}}$ input clears CRC1 to 00H.

Figure 9-4 Capture/Compare Control Register 1 (CRC1) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
CRC1	0	0	0	0	CLR11	CM	CLR10	0	0FF32H	00H	R/W

CLR10	TM1 Clear Operation when TM1 = CR10 TM1W Clear Operation when TM1W = CR10W
0	Disabled
1	Enabled

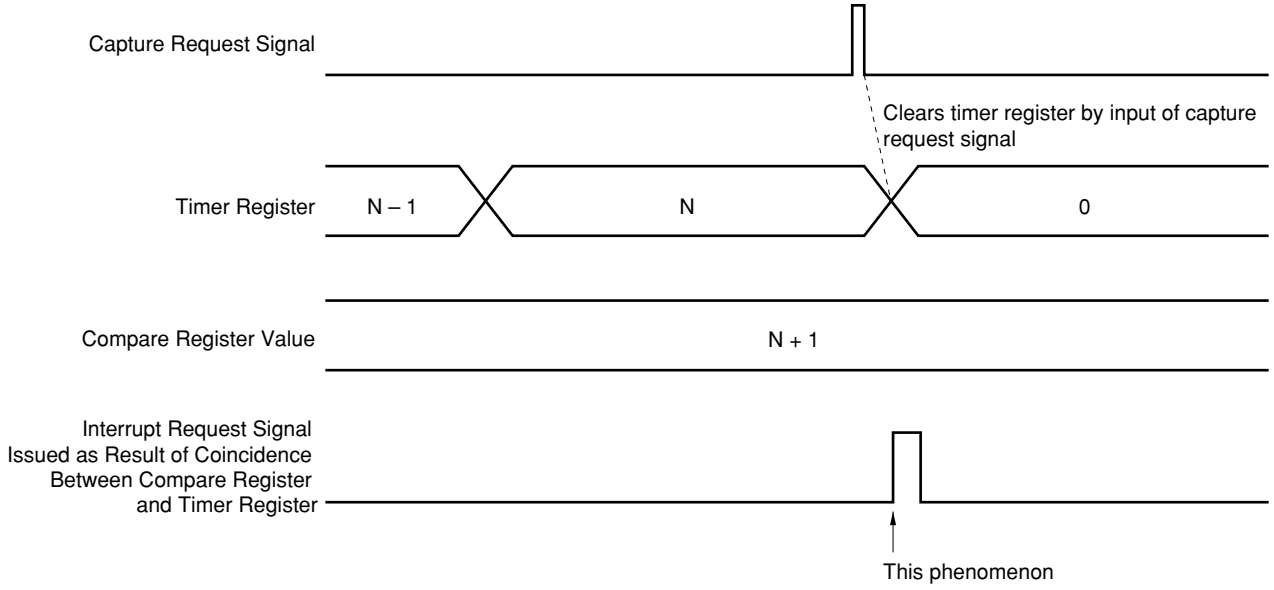
CLR11	CM	CR11/CR11W Operation Specification	TM1/TM1W Clearance Operation
0	0	Compare operation	Disabled
1	0		Enabled (when TM1 & CR11 or TM1W & CR11W contents match)
0	1	Capture operation	Disabled
1	1		Enabled (when TM1 contents are captured in CR11 or when TM1W contents are captured in CR11W)

Caution Even if an attempt is made to clear the timer register by inputting the capture request signal when the capture function of the timer is used, the timer register momentarily counts up immediately before it is cleared. Consequently, if a value greater than the value of the timer register by 1 is set to the compare register when the capture request signal is input, the values of the compare register and timer register coincide, and an unnecessary interrupt will be generated (refer to Figure 9-5). Therefore, take the following operation into consideration when creating a program.

<Operation>

Because the timer register is cleared at the next count if the capture request signal is generated when the value of timer register is "N" when the value "N + 1" is set to the compare register, no interrupt request is generated by the compare register. Actually, however, the timer register momentarily counts "N + 1" when the timer register is cleared. As a result, the values of the timer register and compare register coincide, and an interrupt request signal is generated by the compare register.

Figure 9-5 Example of Generation of Unnecessary Interrupt Request by Compare Register



9.4 TIMER REGISTER 1 (TM1) OPERATION

9.4.1 Basic Operation

8-bit operating mode/16-bit operating mode control can be performed for timer/counter 1 by means of bit 0 (BW1) of timer control register 1 (TMC1). **Note**

In the timer/counter 1 count operation, an up-count is performed using the count clock specified by the low-order 4 bits of prescaler mode register 1 (PRM1).

Count operation enabling/disabling is controlled by bit 3 (CE1) of TMC1 (timer/counter 1 operation control is performed by the low-order 4 bits of the TMC1). When the CE1 bit is set (to 1) by software, the contents of TM1 are cleared to 0H on the first count clock, and then the up-count operation is performed.

When the CE1 bit is cleared (to 0), TM1 becomes 0H immediately, and capture operations and match signal generation are stopped.

If the CE1 bit is set (to 1) again when it is already set (to 1), TM1 continues the count operation without being cleared.

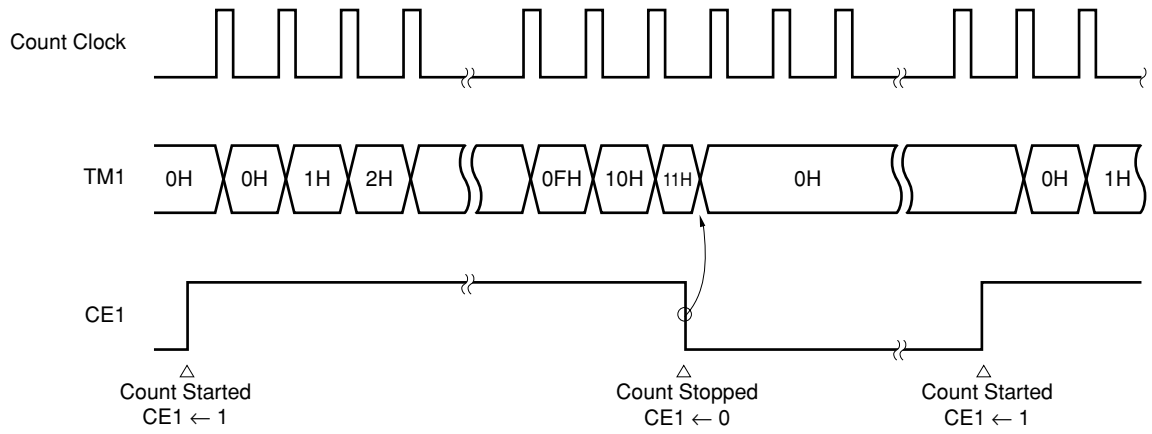
If the count clock is input when TM1 is FFH in 8-bit operating mode and when TM1W is FFFFH in 16-bit operating mode, TM1/TM1W becomes 0H. In this case, OVF1 bit is set. OVF1 bit is cleared by software only. The count operation is continued.

When $\overline{\text{RESET}}$ is input, TM1 is cleared to 0H, and the count operation is stopped.

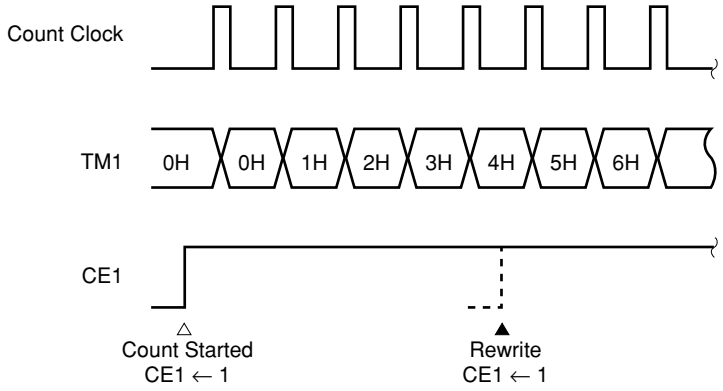
Note Unless otherwise specified, the functions of timer register 1 in the 8-bit operating mode are described hereafter. In the 16-bit operating mode, TM1, CR10, CR11, and CR12 operate as TM1W, CR10W, CR11W, and CR12W, respectively.

Figure 9-6 Basic Operation in 8-Bit Operating Mode (BW1 = 0)

(a) Count started → count disabled → count started



(b) When "1" is written to the CE1 bit again after the count starts



(c) Operation when TM1 = FFH

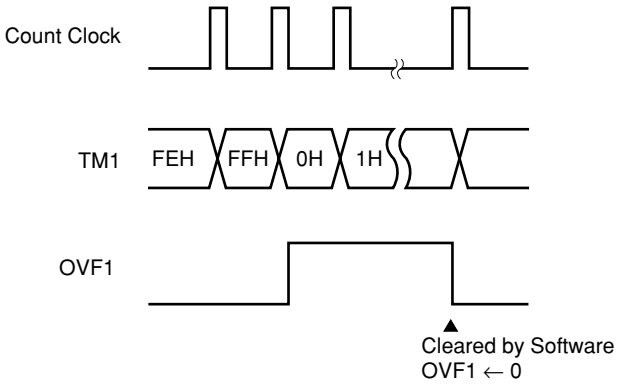
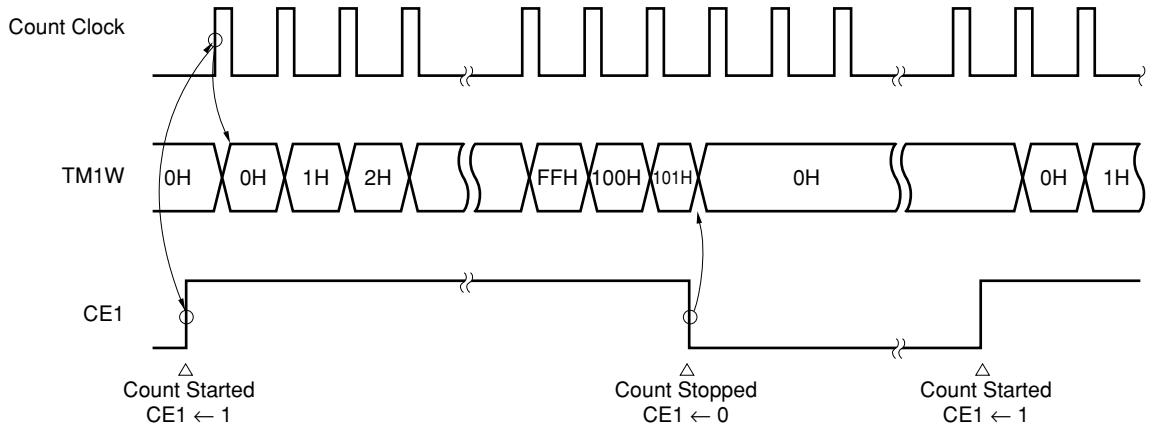
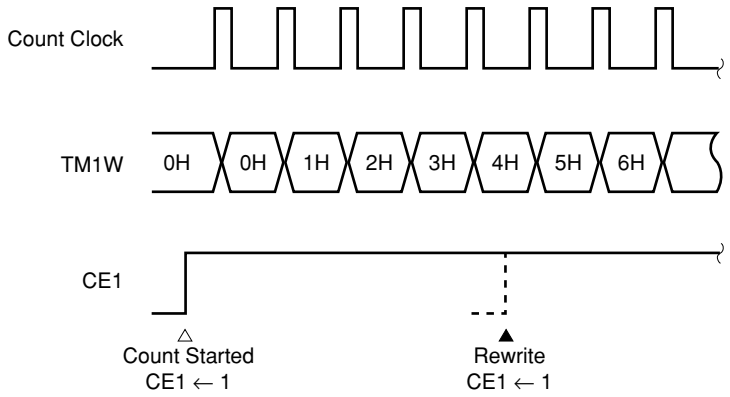


Figure 9-7 Basic Operation in 16-Bit Operating Mode (BW1 = 1)

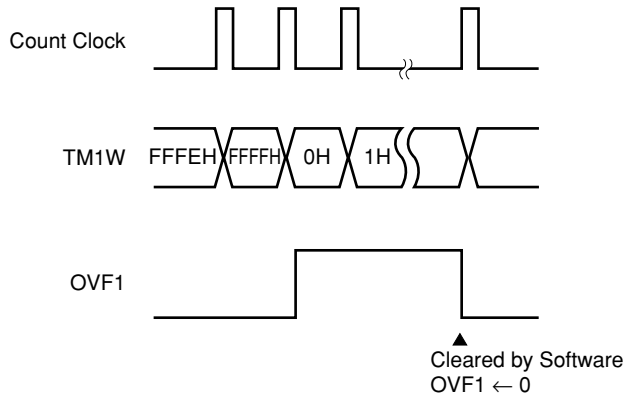
(a) Count started → count disabled → count started



(b) When "1" is written to the CE1 bit again after the count starts



(c) Operation when TM1W = FFFFH



9.4.2 Clear Operation

(1) Clear operation after match with compare register and after capture operation

Timer register 1 (TM1) can be cleared automatically after a match with the compare register (CR1n: n = 0, 1) and a capture operation. When a clearance source arises, TM1 is cleared to 0H on the next count clock. Therefore, even if a clearance source arises, the value at the point at which the clearance source arose is retained until the next count clock arrives.

Figure 9-8 TM1 Clearance by Match With Compare Register (CR10, CR11)

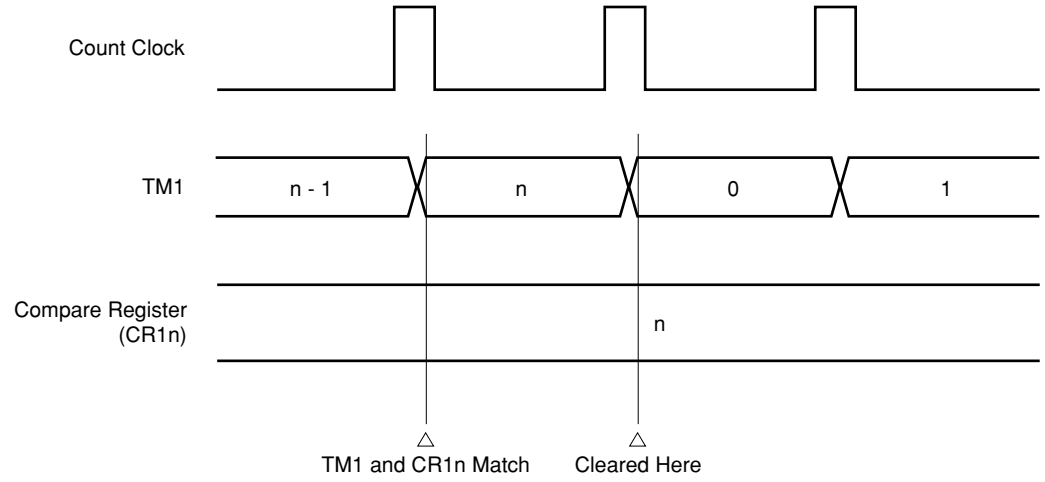
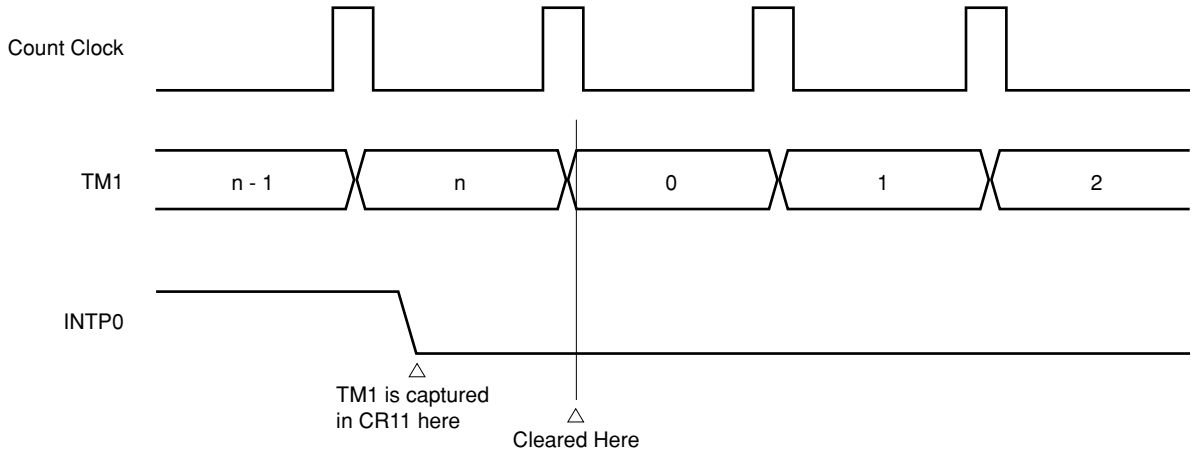


Figure 9-9 TM1 Clearance after Capture Operation

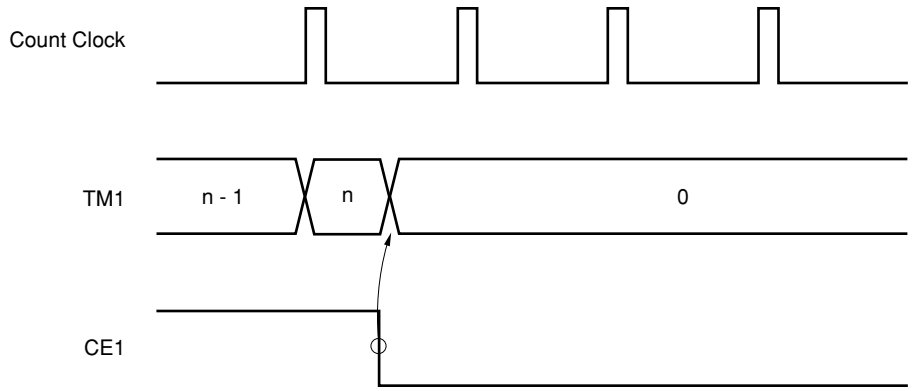


(2) Clear operation by CE1 bit of timer control register 1 (TMC1)

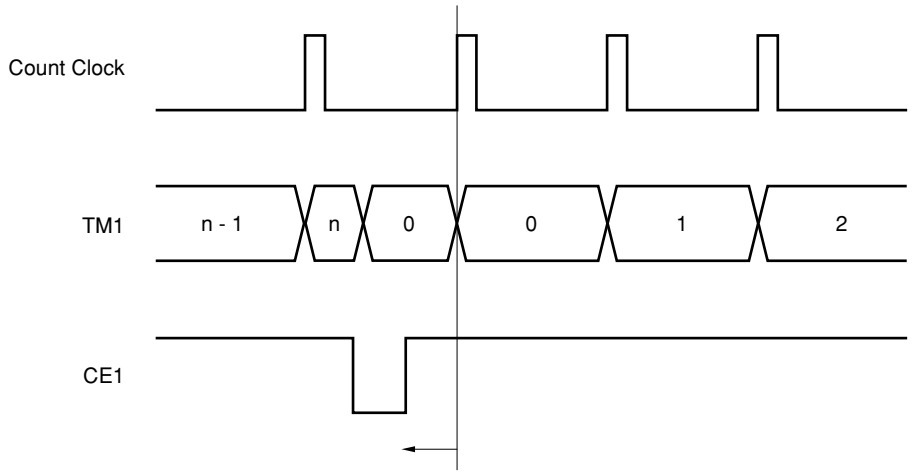
Timer register 1 (TM1) is also cleared when the CE1 bit of TMC1 is cleared (to 0) by software. The clear operation is performed immediately after the clearance (to 0) of the CE1 bit.

Figure 9-10 Clear Operation When CE1 Bit is Cleared (0)

(a) Basic operation

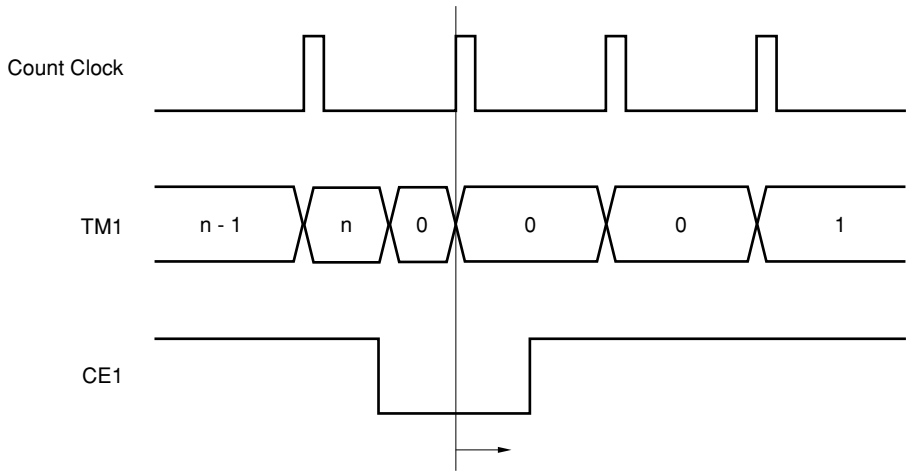


(b) Restart before count clock is input after clearance



If the CE1 bit is set (to 1) before this count clock, this count clock starts counting from 0.

(c) Restart after count clock is input after clearance



If the CE1 bit is set (to 1) from this count clock onward, the count clock starts counting from 0 after the CE1 bit is set (to 1).

9.5 EXTERNAL EVENT COUNTER FUNCTION

Timer/counter 1 can count clock pulses input from the external interrupt request input pin (INTP0) pin.

No special selection method is needed for the external event counter operating mode. When the timer register 1 (TM1) count clock is specified as external clock input by the setting of the low-order 4 bits of prescaler mode register 1 (PRM1), TM1 operates as an external event counter.

The maximum frequency of the external clock pulse that can be counted by the external event counter is determined by the sampling clock select register (SCS0) as shown in Table 9-5.

The maximum frequency is the same when both the edges of the INTP0 input are counted and when only one edge is counted.

The pulse width of the INTP0 input must be three or more sampling clocks selected by SCS0, regardless of whether the level is high or low. If the width is shorter than this, the pulse may not be counted.

Figure 9-11 shows the timing of the external event count by timer/counter 1.

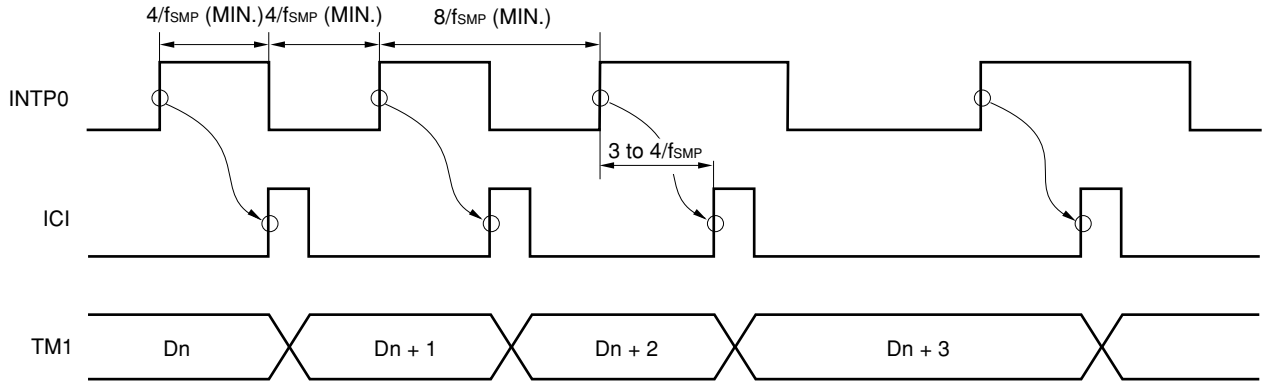
Table 9-5 Maximum Input Frequency and Minimum Input Pulse Width That Can be Counted as Events

( ): f_{XX} = 32 MHz, f_{CLK} = 16 MHz

Sampling Clock Selected by SCS0	Maximum Input Frequency	Minimum Pulse Width
f_{CLK}	f_{CLK}/8 (2.00 MHz)	4/f_{CLK} (0.25 μs)
f_{CLK}/64	f_{CLK}/512 (31.30 kHz)	256/f_{XX} (8.00 μs)
f_{CLK}/128	f_{CLK}/1,024 (15.60 kHz)	512/f_{XX} (16.00 μs)
f_{CLK}/256	f_{CLK}/2,048 (7.81 kHz)	1,024/f_{XX} (32.00 μs)

Figure 9-11 Timer/Counter 1 External Event Count Timing

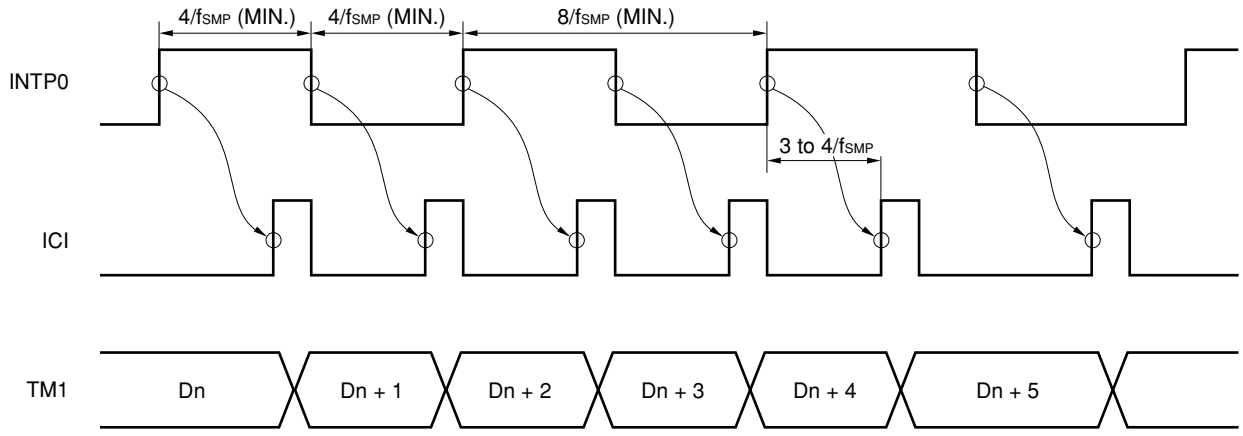
(1) Counting one edge (maximum frequency = f_{CLK}/8)



Remarks 1. ICI: INTP0 input signal after passing through edge detection circuit

2. f_{SMP} is selected by the sampling clock selection register (SCS0).

(2) Counting both edges (maximum frequency = $f_{CLK}/8$)



Remarks 1. ICI: INTP0 input signal after passing through edge detection circuit

2. f_{SMP} is selected by the sampling clock selection register (SCS0).

The TM1 count operation is controlled by the CE1 bit of the timer control register 1 (TMC1) in the same way as with the basic operation.

When the CE1 bit is set (to 1) by software, the contents of TM1 are set to 0H and the up-count operation is started on the initial count clock.

When the CE1 bit is cleared (to 0) by software during a TM1 count operation, the contents of TM1 are set to 0H immediately and the stopped state is entered. The TM1 count operation is not affected if the CE1 bit is set (to 1) by software again when it is already set (to 1).

Caution When timer/counter 1 is used as an external event counter, it is not possible to distinguish between the case where there is no valid edge input at all and the case where there is a single valid edge input using the timer register 1 (TM1) alone (see Figure 9-12), since the contents of TM1 are 0 in both cases. If it is necessary to make this distinction, the INTP0 interrupt request flag should be used. An example is shown in Figure 9-13.

Figure 9-12 Example of the Case Where the External Event Counter Does Not Distinguish Between One Valid Edge Input and No Valid Edge Input

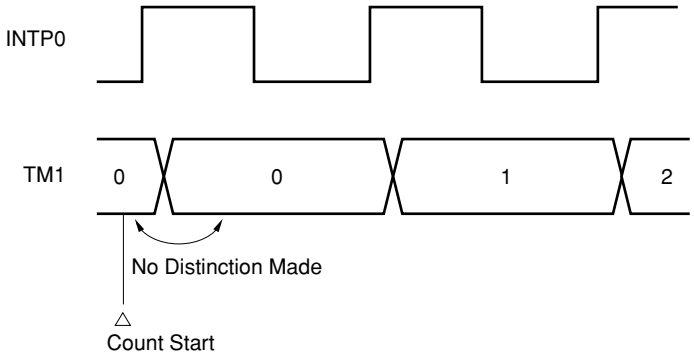
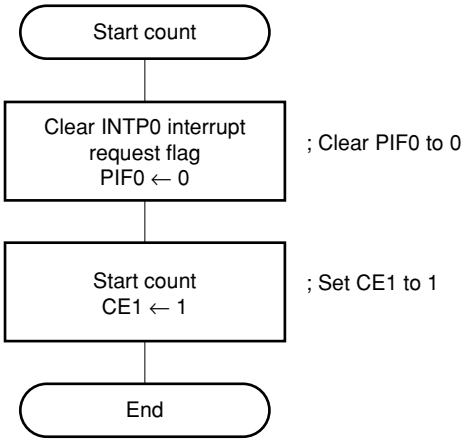
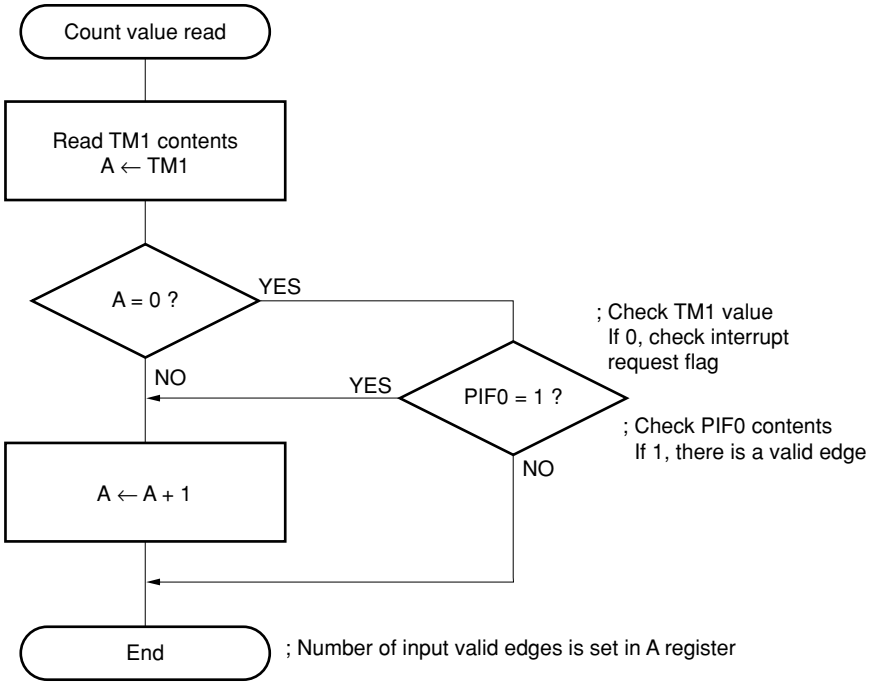


Figure 9-13 Methods of Enabling the External Event Counter to Distinguish No Valid Edge Input

(a) Processing when count is started



(b) Processing when count value is read



9.6 COMPARE REGISTER, CAPTURE/COMPARE REGISTER, AND CAPTURE REGISTER OPERATION

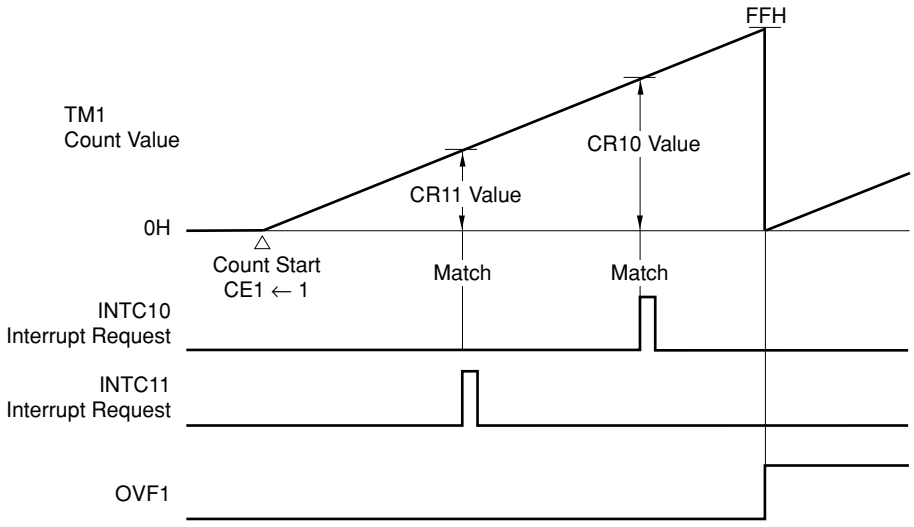
9.6.1 Compare Operations

Timer/counter 1 performs compare operations in which the value set in a compare register (CR10), capture/compare register (CR11), specified for compare operation is compared with the timer register 1 (TM1) count value.

If the count value of TM1 matches the preset value of the CR10, or the CR11 as the result of the count operation, an interrupt request signal (INTC10 or INTC11) is generated.

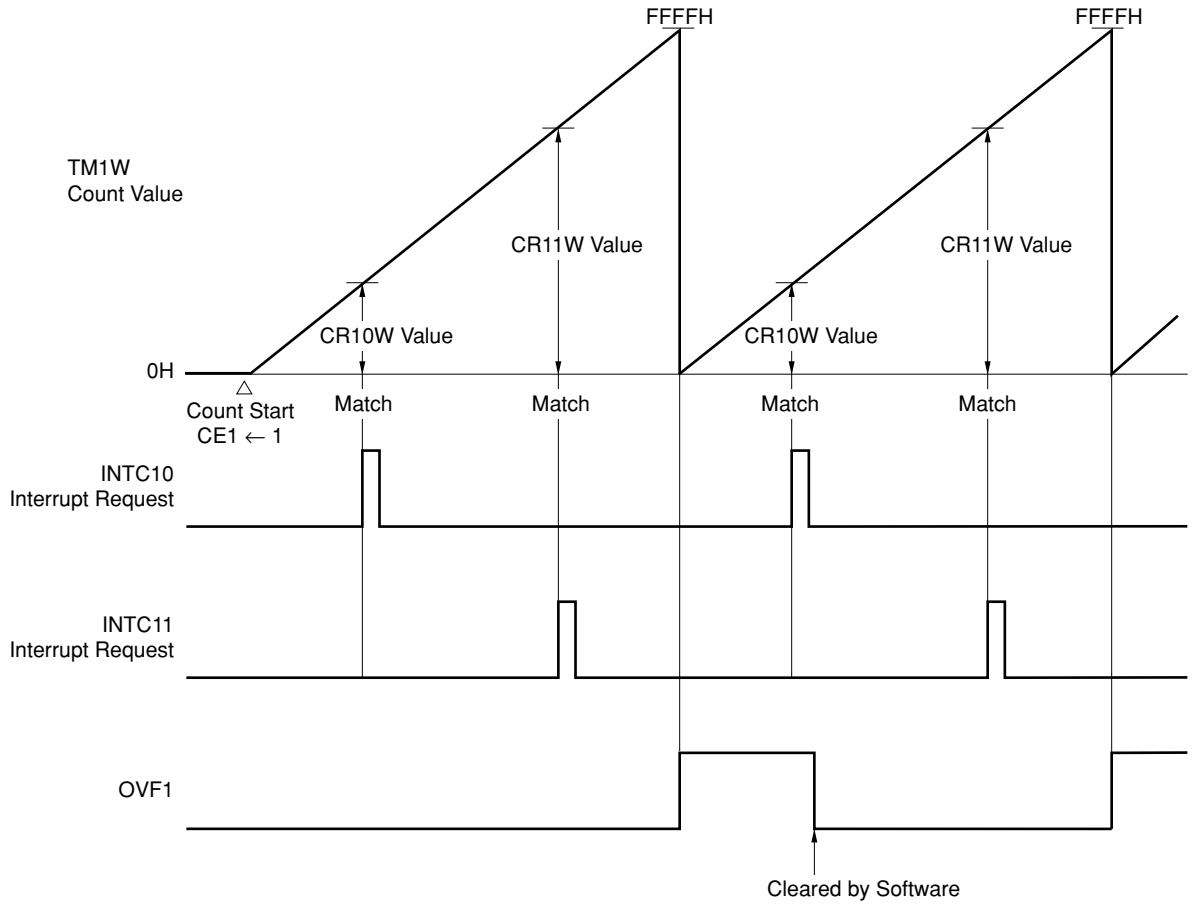
After a match with the CR10 or CR11 value, the TM1 contents can be cleared, and the timer functions as an interval timer that repeatedly counts up to the value set in the CR10 or CR11.

Figure 9-14 Compare Operation in 8-Bit Operating Mode



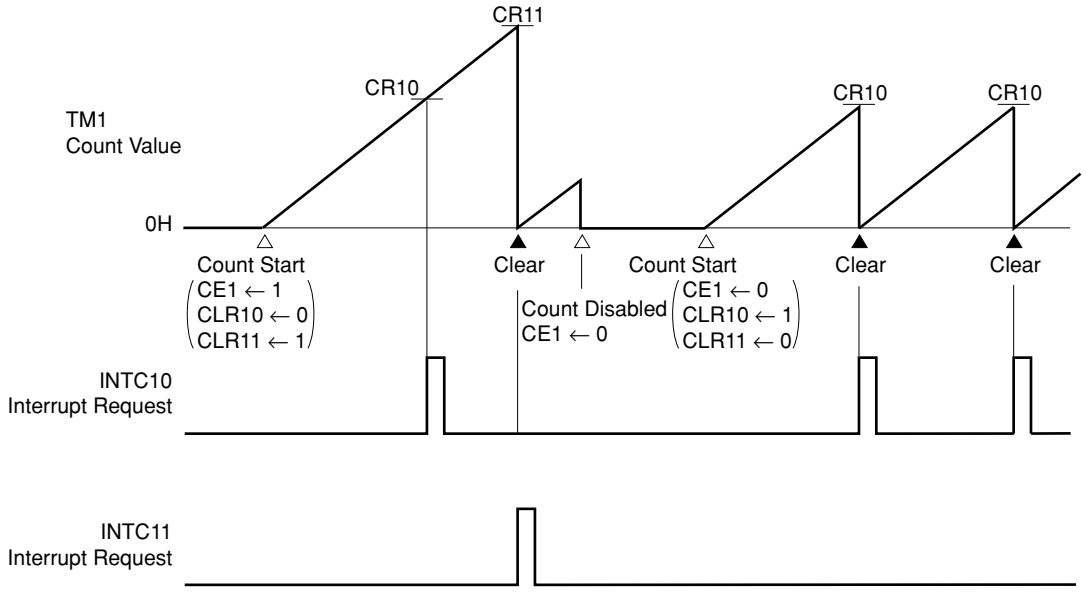
Remark CLR10 = 0, CLR11 = 0, CM = 0, BW1 = 0

Figure 9-15 Compare Operation in 16-Bit Operating Mode



Remark CLR10 = 0, CLR11 = 0, BW1 = 1

Figure 9-16 TM1 Clearance after Match Detection



9.6.2 Capture Operations

Timer/counter 1 performs capture operations in which the timer register 1 (TM1) count value is fetched into the capture register in synchronization with an external trigger, and retained there.

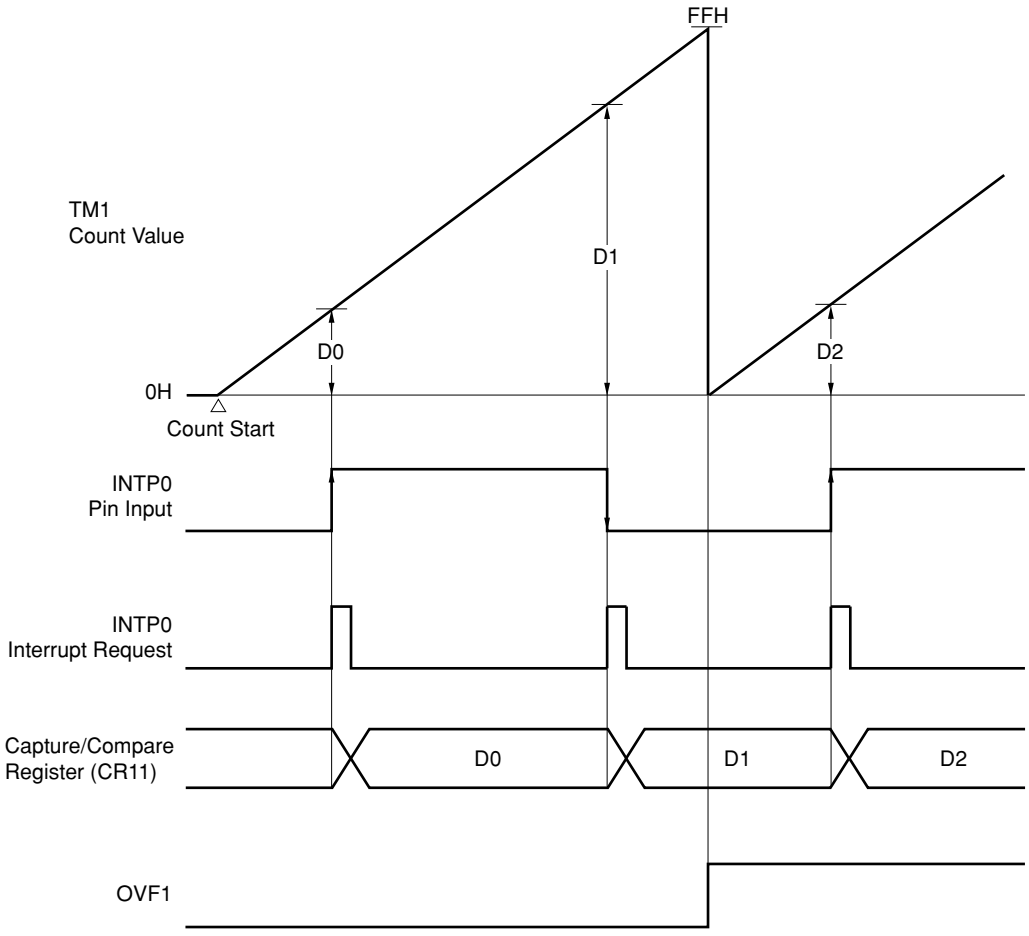
A valid edge detected from the input of the external interrupt request input pin (INTP0) is used as the external trigger (capture trigger). The count value of TM1 in the process of being counted is fetched into the capture register (CR12), or the capture/compare register (CR11) when a capture operation is specified, in synchronization with the capture trigger, and is retained there. The contents of the CR11 and CR12 are retained until the next capture trigger is generated.

The capture trigger valid edge is set by means of external interrupt mode register 0 (INTM0). If both rising and falling edges are set as capture triggers, the width of pulses input from off-chip can be measured, and if a capture trigger is generated by a single edge, the input pulse cycle can be measured.

See **Figure 21-1** in **CHAPTER 21 EDGE DETECTION FUNCTION** for details of the INTM0 format.

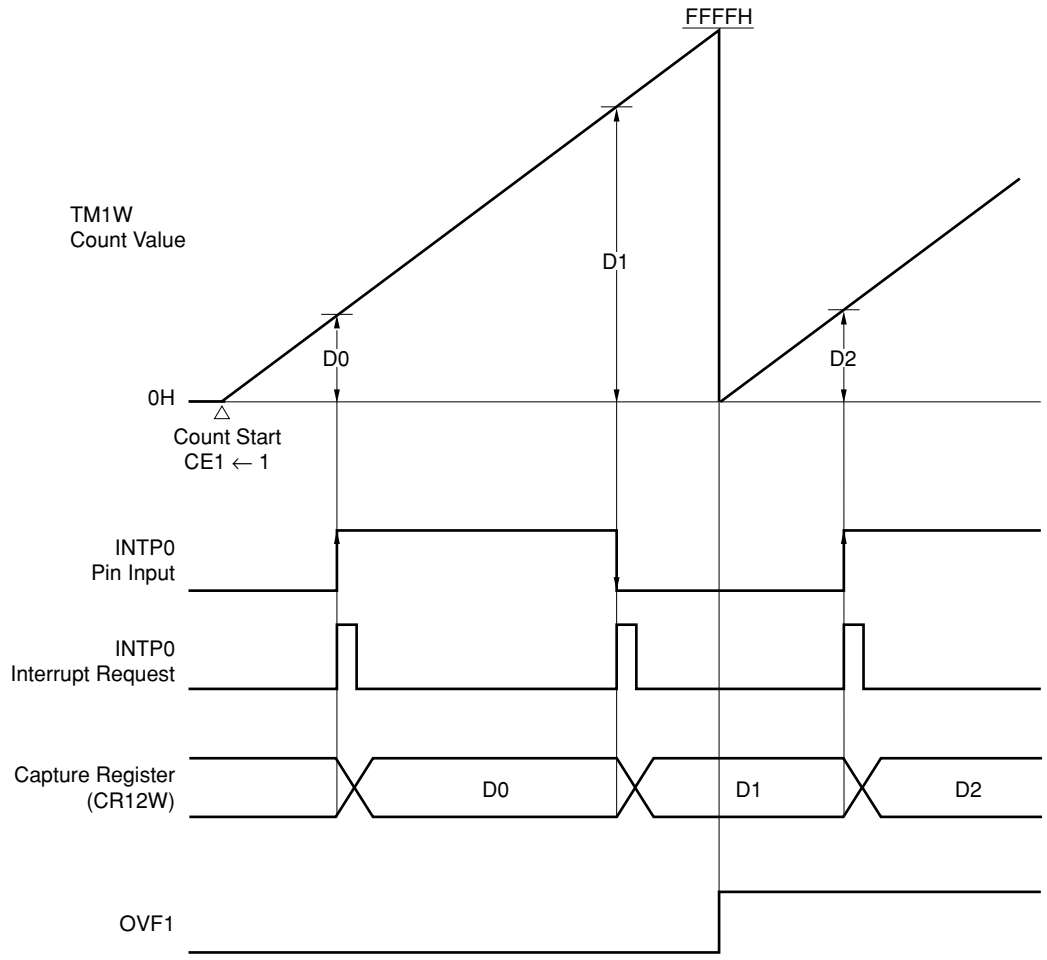
When CR11 is used as a capture register, TM1 can be cleared as soon as the contents of TM1 have been captured to CR11 by capture trigger.

Figure 9-17 Capture Operation in 8-Bit Operating Mode



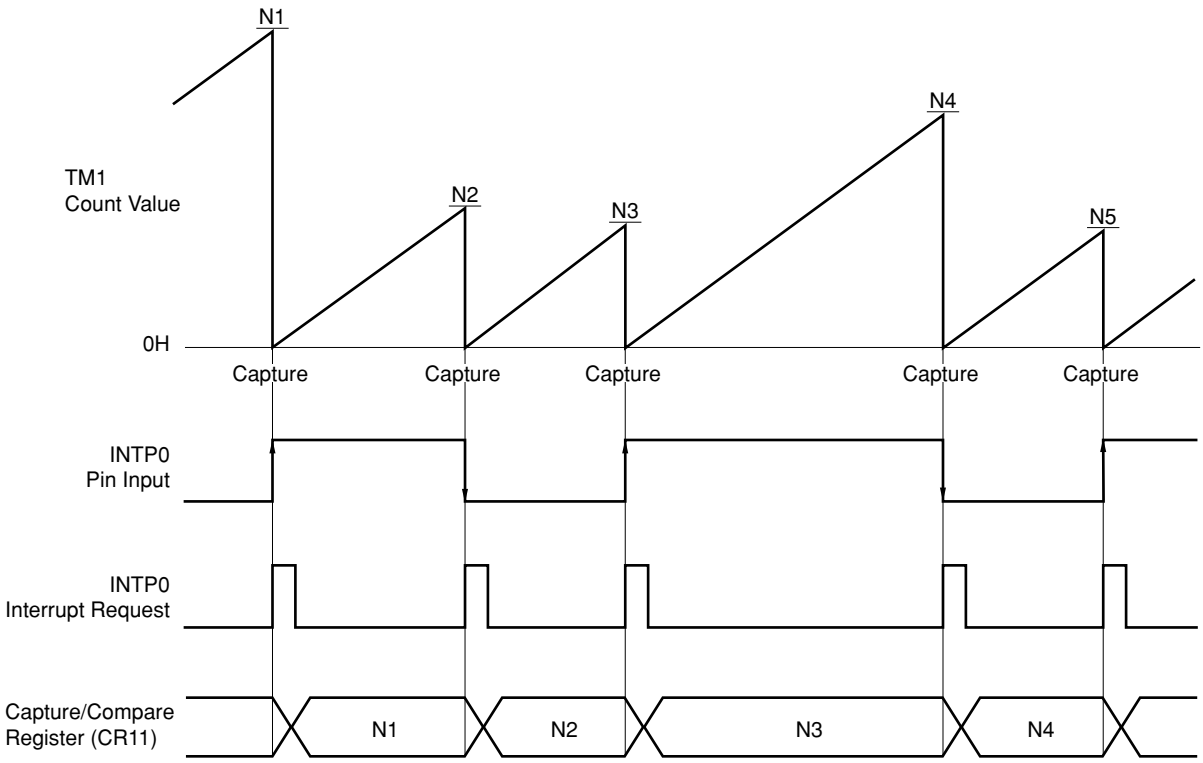
Remark Dn: TM1 count value (n = 0, 1, 2, ...)
CLR10 = 0, CLR11 = 0, CM = 1, BW1 = 0

Figure 9-18 Capture Operation in 16-Bit Operating Mode



Remark Dn: TM1W count value (n = 0, 1, 2, ...)
CLR10 = 0, CLR11 = 0, CM = 1, BW1 = 1

Figure 9-19 TM1 Clearance after Capture Operation



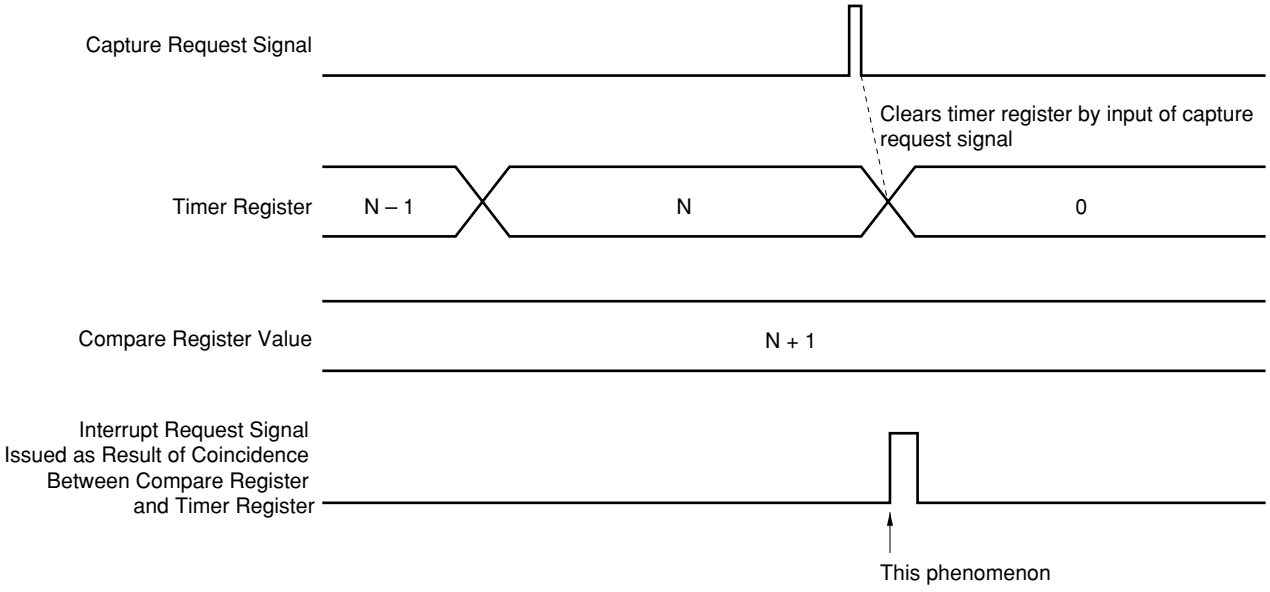
Remark NI: TM1 count value (n = 0, 1, 2, ...)
CLR10 = 0, CLR11 = 1, CM = 1

Caution Even if an attempt is made to clear the timer register by inputting the capture request signal when the capture function of the timer is used, the timer register momentarily counts up immediately before it is cleared. Consequently, if a value greater than the value of the timer register by 1 is set to the compare register when the capture request signal is input, the values of the compare register and timer register coincide, and an unnecessary interrupt will be generated (refer to Figure 9-20). Therefore, take the following operation into consideration when creating a program.

<Operation>

Because the timer register is cleared at the next count if the capture request signal is generated when the value of timer register is “N” when the value “N + 1” is set to the compare register, no interrupt request is generated by the compare register. Actually, however, the timer register momentarily counts “N + 1” when the timer register is cleared. As a result, the values of the timer register and compare register coincide, and an interrupt request signal is generated by the compare register.

Figure 9-20 Example of Generation of Unnecessary Interrupt Request by Compare Register



9.7 EXAMPLES OF USE

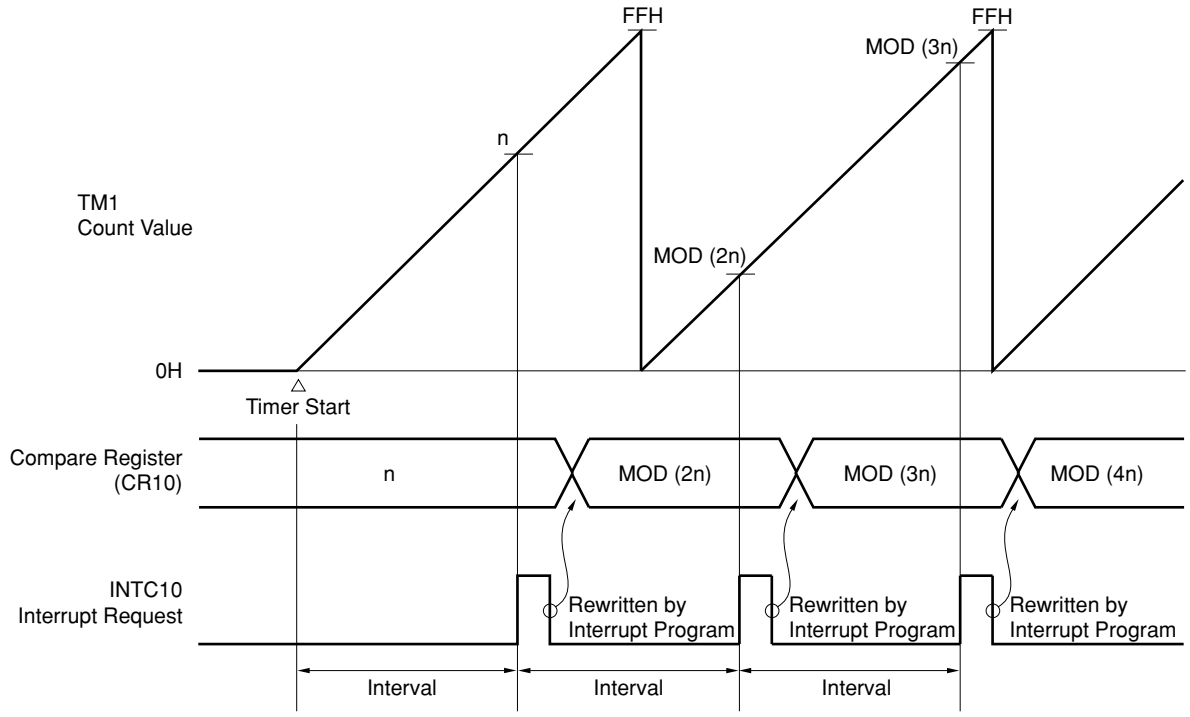
9.7.1 Operation as Interval Timer (1)

When timer register 1 (TM1) is made free-running and a fixed value is added to the compare register (CR1n: n = 0, 1) in the interrupt service routine, TM1 operates as an interval timer with the added fixed value as the cycle (see **Figure 9-21**).

Since TM1 has two compare registers, two interval timers with different intervals can be constructed.

The control register settings are shown in Figure 9-22, the setting procedure in Figure 9-23, and the processing in the interrupt service routine in Figure 9-24.

Figure 9-21 Interval Timer Operation (1) Timing



Remark Interval = n × x/fxx, 1 ≤ n ≤ FFH

x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

Figure 9-22 Control Register Settings for Interval Timer Operation (1)

(a) Prescaler mode register 1 (PRM1)

(b) Capture/compare control register 1 (CRC1)

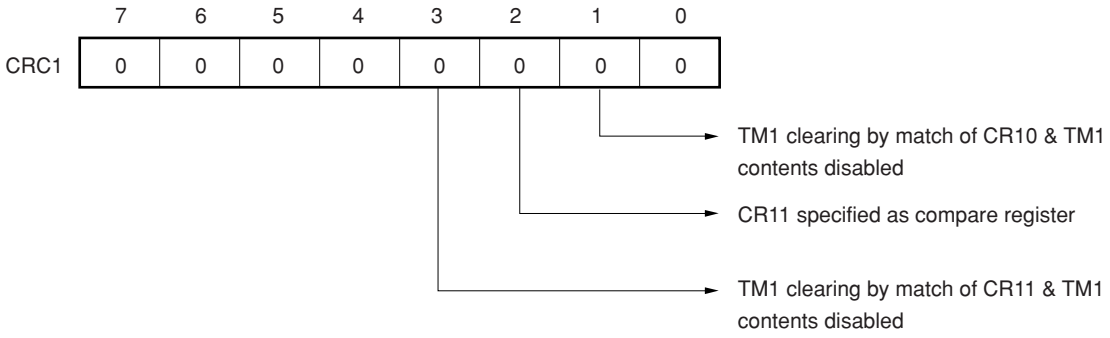


Figure 9-23 Interval Timer Operation (1) Setting Procedure

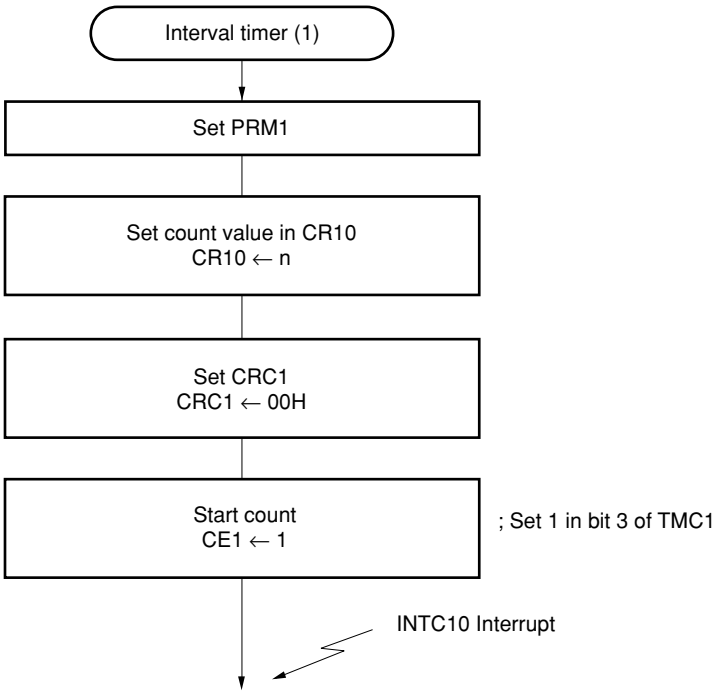
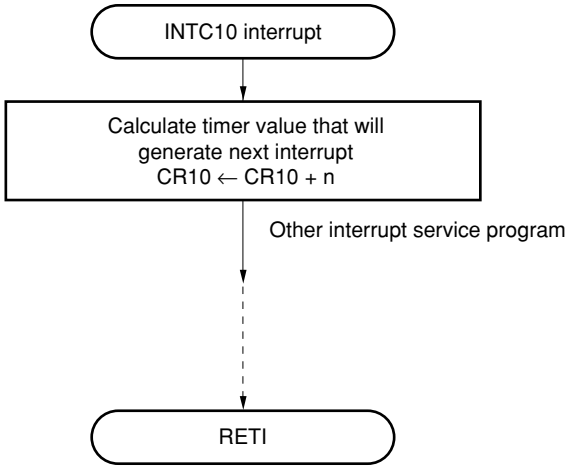


Figure 9-24 Interval Timer Operation (1) Interrupt Request Servicing

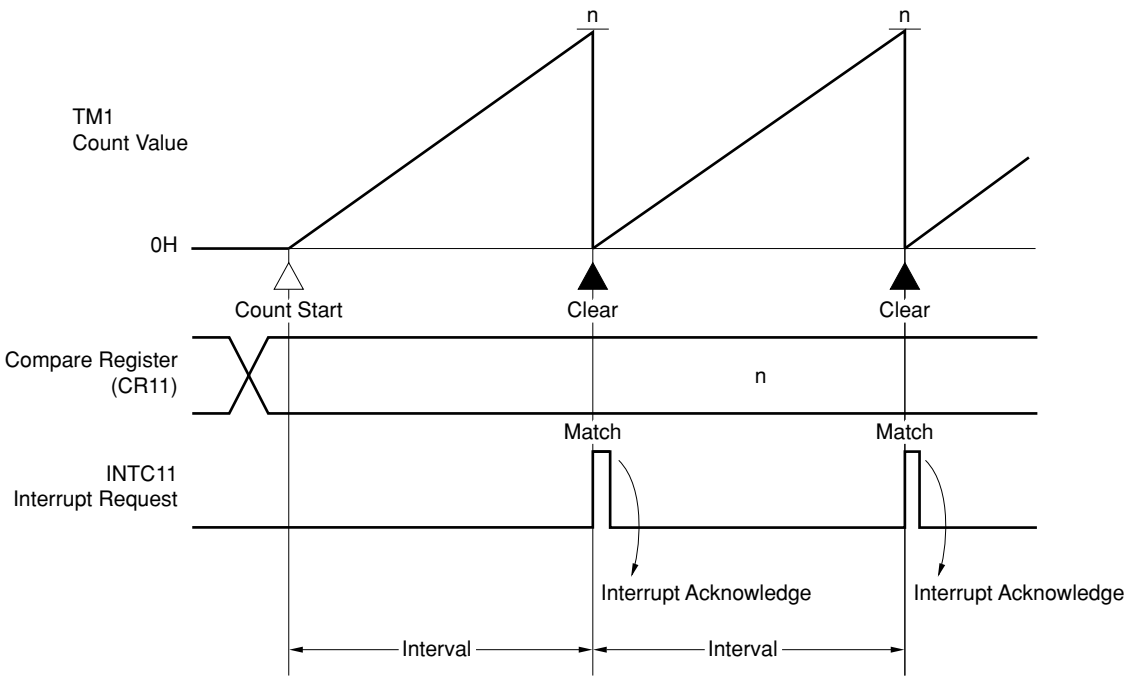


9.7.2 Operation as Interval Timer (2)

TM1 operates as an interval timer that generates interrupts repeatedly with the preset count time as the interval (see **Figure 9-25**).

The control register settings are shown in Figure 9-26, and the setting procedure in Figure 9-27.

Figure 9-25 Interval Timer Operation (2) Timing (When CR11 is Used as Compare Register)



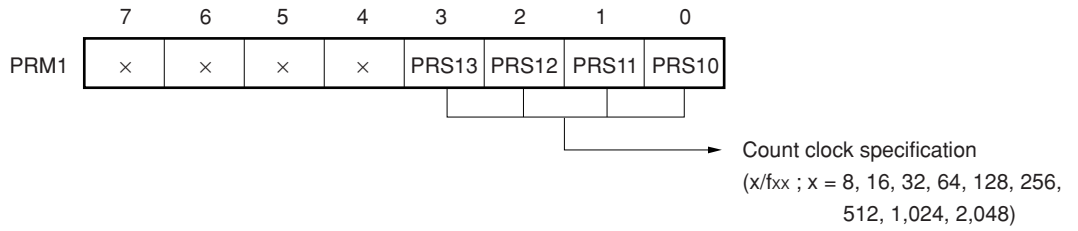
Remark Interval = (n + 1) × x/fxx

$0 \leq n \leq FFH$

x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

Figure 9-26 Control Register Settings for Interval Timer Operation (2)

(a) Prescaler mode register 1 (PRM1)



(b) Capture/compare control register 1 (CRC1)

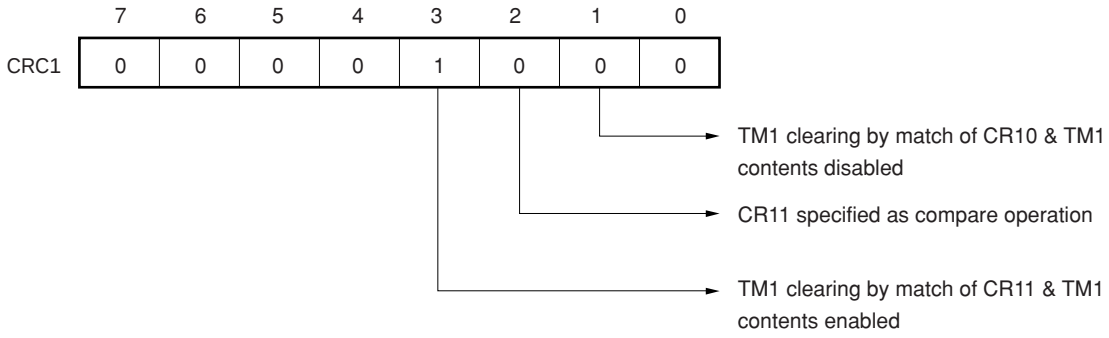
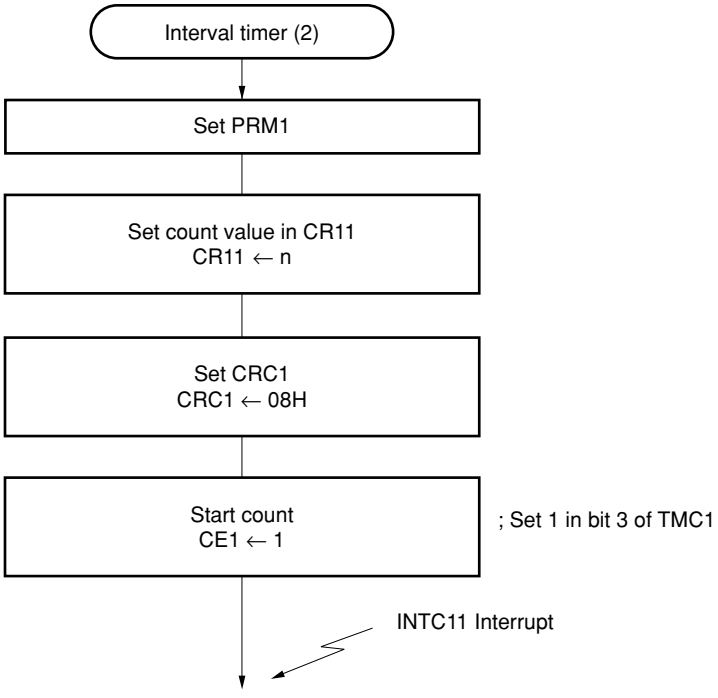


Figure 9-27 Interval Timer Operation (2) Setting Procedure



9.7.3 Pulse Width Measurement Operation

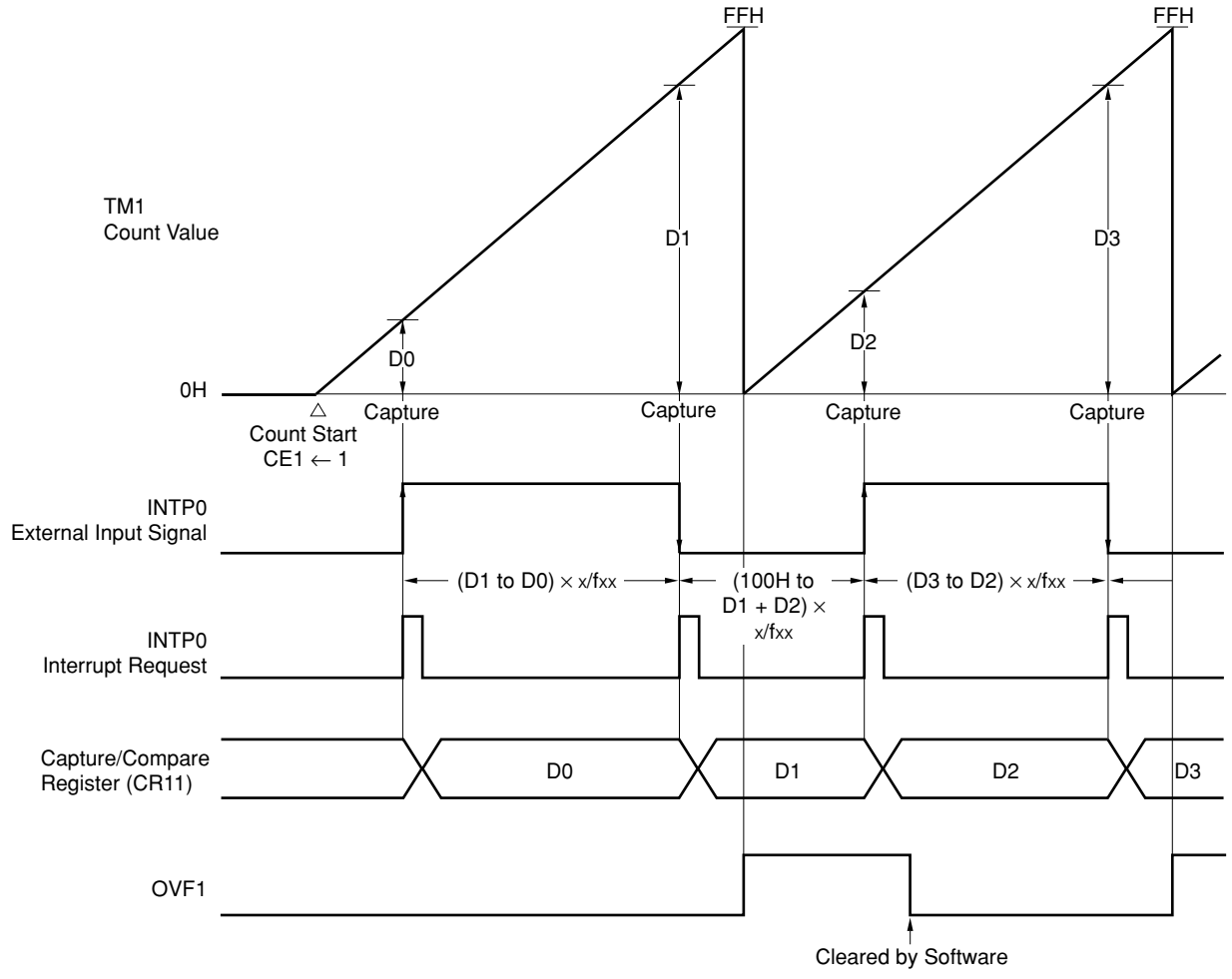
In pulse width measurement, the high-level or low-level width of external pulses input to the external interrupt request input pin (INTP0) is measured.

Both the high-level and low-level widths of pulses input to the INTP0 pin must be at least 3 sampling clocks selected by SCS0; if shorter than this, the valid edge will not be detected and a capture operation will not be performed.

As shown in Figure 9-28, the timer register 1 (TM1) value being counted is fetched into the capture/compare register (CR11) set as a capture register in synchronization with a valid edge (set as both rising and falling edges) in the INTP0 pin input, and held there. The pulse width is obtained from the product of the difference between the TM1 count value (D_n) fetched into and held in the CR11 on detection of the nth valid edge and the count value (D_{n-1}) fetched and held on detection of valid edge n - 1, and the number of count clocks (x/fxx; x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048).

The control register settings are shown in Figure 9-29, and the setting procedure in Figure 9-30.

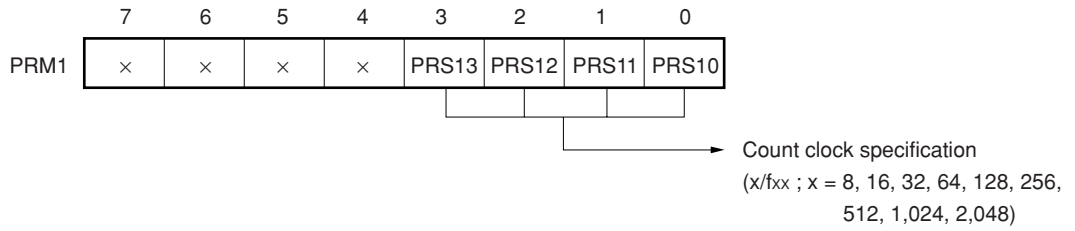
Figure 9-28 Pulse Width Measurement Timing (When CR11 is Used as Capture Register)



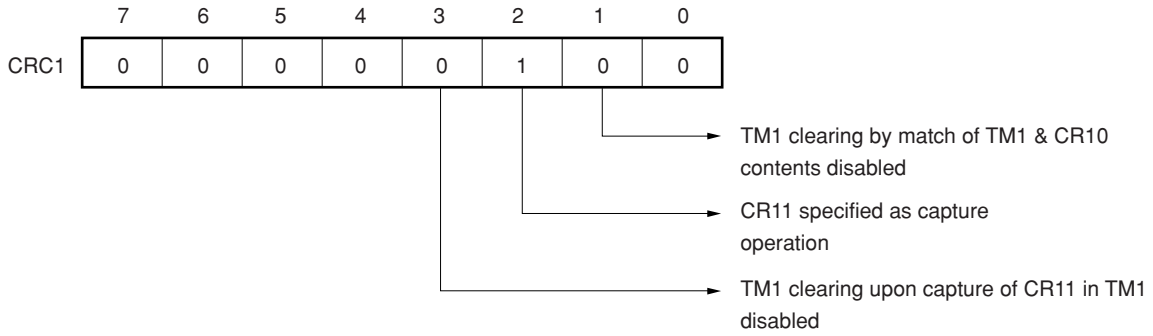
Remark Dn: TM1 count value (n = 0, 1, 2, ...)
x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

Figure 9-29 Control Register Settings for Pulse Width Measurement

(a) Prescaler mode register 1 (PRM1)



(b) Capture/compare control register 1 (CRC1)



(c) External interrupt mode register 0 (INTM0)

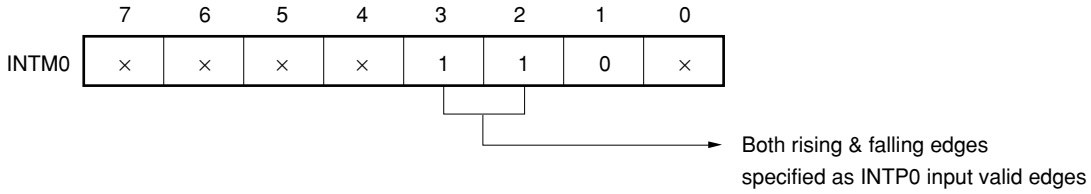


Figure 9-30 Pulse Width Measurement Setting Procedure

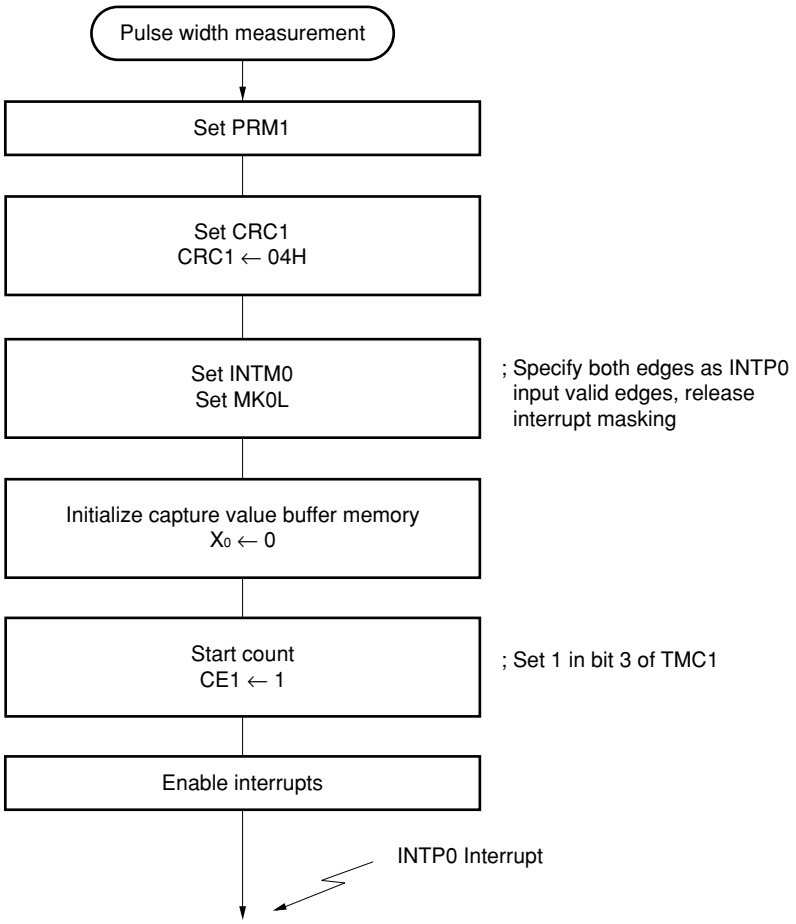
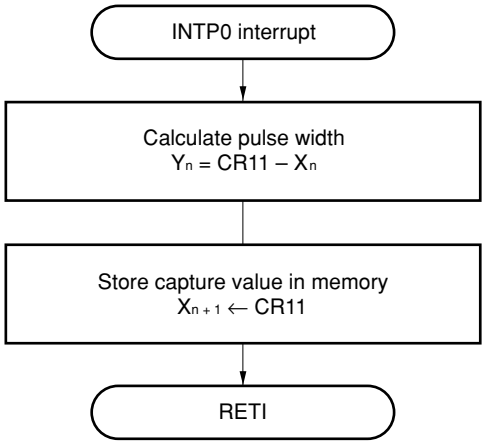


Figure 9-31 Interrupt Request Servicing that Calculates Pulse Width



9.8 CAUTIONS

(1) While timer/counter 1 is operating (while the CE1 bit of the timer control register 1 (TMC1) is set), malfunctioning may occur if the contents of the following registers are rewritten. This is because it is undefined which takes precedence in a contention, the change in the hardware functions due to rewriting the register, or the change in the status because of the function before rewriting.
Therefore, be sure to stop the counter operation for the sake of safety before rewriting the contents of the following registers.

- Prescaler mode register 1 (PRM1)
- Capture/compare control register 1 (CRC1)
- CMD2 bit of timer control register 1 (TMC1)

(2) If the contents of the compare register (CR1n: n = 0 or 1) coincide with those of TM1 when an instruction that stops timer register 1 (TM1) operation is executed, the counting operation of TM1 stops, but an interrupt request is generated.
In order not to generate the interrupt when stopping the operation of TM1, mask the interrupt in advance by using the interrupt mask register before stopping TM1.

Example

Program that may generate interrupt request

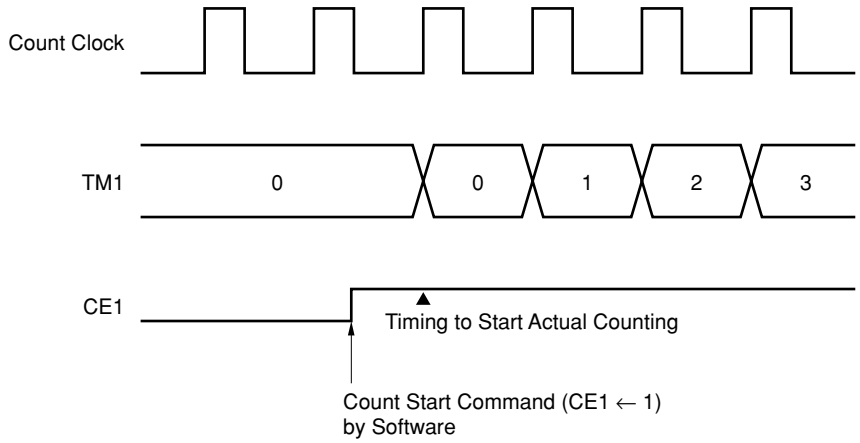
```
        ⋮
CLR1    CE1
OR      MK0L, #C0H      ← Interrupt request
        ⋮                   from timer/counter 1
                            occurs between
                            these instructions
```

Program that does not generate interrupt request

```
        ⋮
OR      MK0L, #C0H      ← Disables interrupt
CLR1    CE1                 from timer/counter 1
CLR1    CIF10           ← Clears interrupt request
CLR1    CIF11               flag from timer/counter 1
        ⋮
```

(3) Up to 1 count clock is required after an operation to start timer/counter 1 (CE1 ← 1) has been performed before timer/counter 1 actually starts (refer to **Figure 9-32**).
For example, when using timer/counter 1 as an interval timer, the first interval time is delayed by up to 1 clock. The second and those that follow are at the specified interval.

Figure 9-32 Operation When Counting is Started



(4) While an instruction that writes data to the compare register (CR1n: n = 0, 1) is executed, coincidence between CR1n, to which the data is to be written, and timer register 1 (TM1) is not detected.
Write data to CR1n when timer/counter 1 is executing counting operation in the timing that the contents of TM1 do not coincide with the value of CR1n before and after writing (e.g., immediately after an interrupt request has been generated because TM1 and CR1n have coincided).

(5) Coincidence between timer register 1 (TM1) and compare register (CR1n: n = 0, 1) is detected only when TM1 is incremented. Therefore, the interrupt request is not generated even if the same value as TM1 is written to CR1n.

(6) If the value of the timer register is read under the condition indicated by "×" in Table 9-6, the read value may be illegal. Do not read the timer register under condition "×".

Table 9-6 Limits of Reading Timer Register

(√: Can be read, ×: Must not be read)

Timer Count Clock \ f_{CLK}	$f_{XX}/2$	$f_{XX}/4$	$f_{XX}/8$	$f_{XX}/16$
$f_{XX}/8$	√	√	×	×
$f_{XX}/16$	√	√	√	×
f_{XX}/n	√	√	√	√

Remarks 1. f_{XX}: Oscillation frequency
2. f_{CLK}: Internal system clock frequency
3. n = 32, 64, 128, 256, 512, 1,024, 2,048

(7) When timer/counter 0 is used as an external event counter, it is not possible to distinguish between the case where there is no valid edge input at all and the case where there is a single valid edge input, using the timer register 0 (TM0) alone (refer to **Figure 9-33**), since the contents of TM0 are 0 in both cases. If it is necessary to make this distinction, the INTP3 interrupt request flag should be used. To make a distinction, use the interrupt request flag of INTP0, as shown in Figure 9-34.

Figure 9-33 Example of the Case Where the External Event Counter Does Not Distinguish Between One Valid Edge Input and No Valid Edge Input

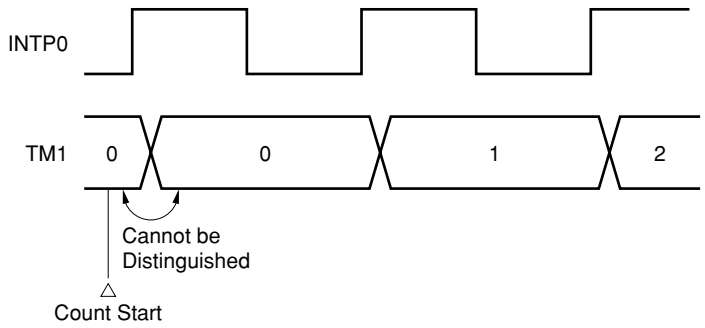
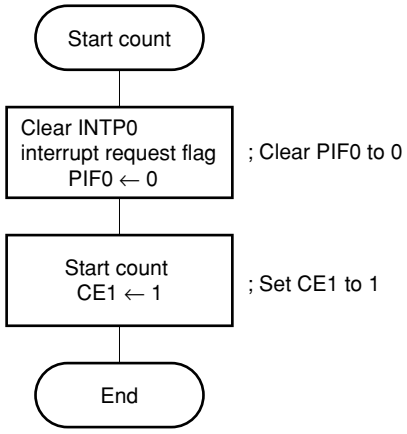
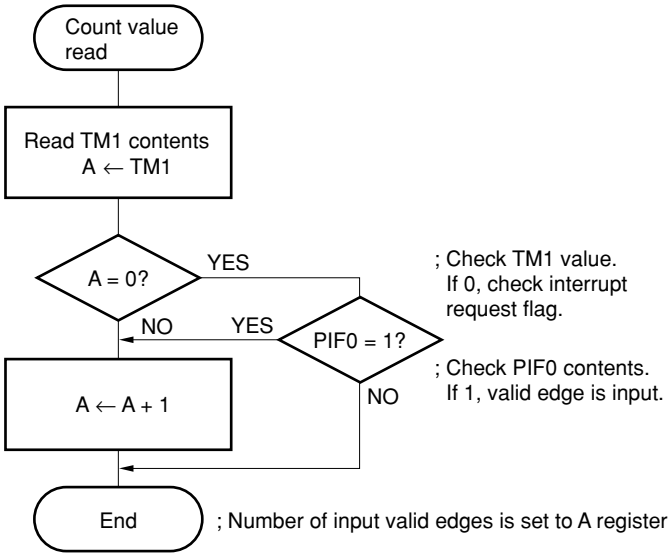


Figure 9-34 To Distinguish Whether One or No Valid Edge Has Been Input with External Event Counter

(a) Processing on starting counting



(b) Processing on reading count value

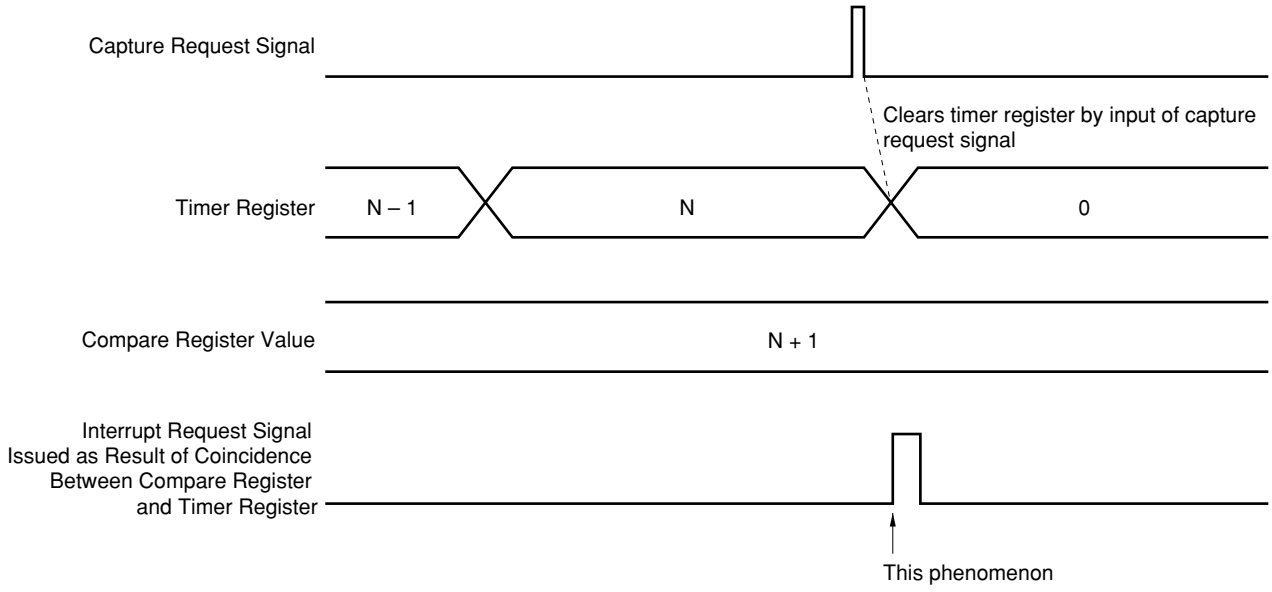


(8) Even if an attempt is made to clear the timer register by inputting the capture request signal when the capture function of the timer is used, the timer register momentarily counts up immediately before it is cleared. Consequently, if a value greater than the value of the timer register by 1 is set to the compare register when the capture request signal is input, the values of the compare register and timer register coincide, and an unnecessary interrupt will be generated (refer to Figure 9-35). Therefore, take the following operation into consideration when creating a program.

<Operation>

Because the timer register is cleared at the next count if the capture request signal is generated when the value of timer register is "N" when the value "N + 1" is set to the compare register, no interrupt request is generated by the compare register. Actually, however, the timer register momentarily counts "N + 1" when the timer register is cleared. As a result, the values of the timer register and compare register coincide, and an interrupt request signal is generated by the compare register.

Figure 9-35 Example of Generation of Unnecessary Interrupt Request by Compare Register

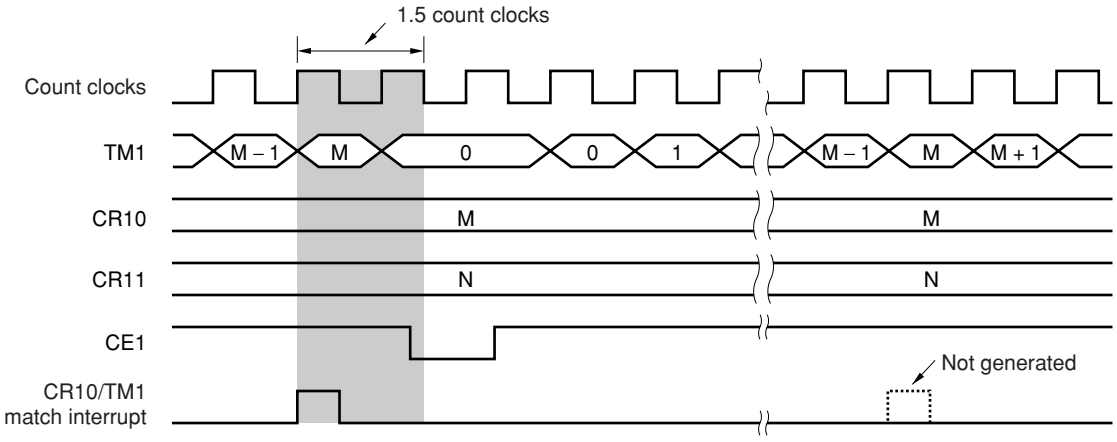


★ (9) If the count operation of TM1 stops at the timing at which compare register (CR10) and timer register 1 (TM1) match, the CR10/TM1 match interrupt may not be generated after timer/counter 1 is next started.

If the TM1 count operation is stopped within 1.5 count clocks after a match between CR10 and TM1, the first match interrupt after timer/counter 1 is next started will not be generated. The second and subsequent interrupts operate normally. Note that the timer output is unaffected by this bug.

This bug occurs because the timer interrupt controller inadvertently masks interrupts if timer/counter 1 is stopped in the period indicated by the shaded area in the figure below.

The interrupt controller is initialized by an overflow of timer/counter 1 or a match between CR11 and TM1.



Remark M < N

Do not stop timer/counter 1 within 1.5 count clocks after a match between CR10 and TM1.

Disable all interrupt requests (including macro servicing), read the value of the timer to be stopped, and wait until at least 1.5 count clocks have elapsed after a match between CR10 and TM1 before stopping timer/counter 1.

CHAPTER 10 TIMER/COUNTER 2

10.1 FUNCTIONS

Timer/counter 2 is 16-bit or 8-bit timer/counter, and has the following function which the other three timer/counters do not have:

- One-shot timer **Note**

Note The one-shot timer function is a count operation of timer/counter 2 (TM2/TM2W), and is thus different in nature from the one-shot pulse output function of timer/counter 0.

In this section, the following four basic functions are described in order:

- Interval timer
- Programmable square-wave output
- Pulse width measurement
- External event counter

(1) Interval timer

Generates internal interrupts at preset intervals.

Table 10-1 Timer/Counter 2 Intervals

Minimum Interval	Maximum Interval	Resolution
8/fxx (0.25 μs)	$2^{16} \times 8$/fxx (16.40 ms)	8/fxx (0.25 μs)
16/fxx (0.50 μs)	$2^{16} \times 16$/fxx (32.80 ms)	16/fxx (0.50 μs)
32/fxx (1.60 μs)	$2^{16} \times 32$/fxx (65.50 ms)	32/fxx (1.00 μs)
64/fxx (2.00 μs)	$2^{16} \times 64$/fxx (131 ms)	64/fxx (2.00 μs)
128/fxx (4.00 μs)	$2^{16} \times 128$/fxx (262 ms)	128/fxx (4.00 μs)
256/fxx (8.00 μs)	$2^{16} \times 256$/fxx (524 ms)	256/fxx (8.00 μs)
512/fxx (16.00 μs)	$2^{16} \times 512$/fxx (1.05 s)	512/fxx (16.00 μs)
1,024/fxx (32.00 μs)	$2^{16} \times 1{,}024$/fxx (2.10 s)	1,024/fxx (32.00 μs)
2,048/fxx (64.00 μs)	$2^{16} \times 2{,}048$/fxx (4.19 s)	2,048/fxx (64.00 μs)

( ): When fxx = 32 MHz

(2) Programmable square-wave output

Outputs square waves independently to the timer output pins (TO2 and TO3).

Table 10-2 Timer/Counter 2 Programmable Square-Wave Output Setting Range

Minimum Pulse Width	Maximum Pulse Width
$8/f_{XX}$ (0.25 μs)	$2^{16} \times 8/f_{XX}$ (16.40 ms)
$16/f_{XX}$ (0.50 μs)	$2^{16} \times 16/f_{XX}$ (32.80 ms)
$32/f_{XX}$ (1.00 μs)	$2^{16} \times 32/f_{XX}$ (65.50 ms)
$64/f_{XX}$ (2.00 μs)	$2^{16} \times 64/f_{XX}$ (131 ms)
$128/f_{XX}$ (4.00 μs)	$2^{16} \times 128/f_{XX}$ (262 ms)
$256/f_{XX}$ (8.00 μs)	$2^{16} \times 256/f_{XX}$ (524 ms)
$512/f_{XX}$ (16.00 μs)	$2^{16} \times 512/f_{XX}$ (1.05 s)
$1,024/f_{XX}$ (32.00 μs)	$2^{16} \times 1,024/f_{XX}$ (2.10 s)
$2,048/f_{XX}$ (64.00 μs)	$2^{16} \times 2,048/f_{XX}$ (4.19 s)

( ): When f_{XX} = 32 MHz

Caution The above table is applicable to use of an internal clock.

(3) Pulse width measurement

Detects the pulse width of the signal input to an external interrupt request input pins (INTP1/INTP2).

Table 10-3 Timer/Counter 2 Pulse Width Measurement Range

Measurable Pulse Width Note			Resolution
8/fxx (0.25 μs)	to	2^{16} × 8/fxx (16.40 ms)	8/fxx (0.25 μs)
16/fxx (0.50 μs)	to	2^{16} × 16/fxx (32.80 ms)	16/fxx (0.50 μs)
32/fxx (1.00 μs)	to	2^{16} × 32/fxx (65.50 ms)	32/fxx (1.00 μs)
64/fxx (2.00 μs)	to	2^{16} × 64/fxx (131 ms)	64/fxx (2.00 μs)
128/fxx (4.00 μs)	to	2^{16} × 128/fxx (262 ms)	128/fxx (4.00 μs)
256/fxx (8.00 μs)	to	2^{16} × 256/fxx (524 ms)	256/fxx (8.00 μs)
512/fxx (16.00 μs)	to	2^{16} × 512/fxx (1.05 s)	512/fxx (16.00 μs)
1,024/fxx (32.00 μs)	to	2^{16} × 1,024/fxx (2.10 s)	1,024/fxx (32.00 μs)
2,048/fxx (64.00 μs)	to	2^{16} × 2,048/fxx (4.19 s)	2,048/fxx (64.00 μs)

( ): When fxx = 32 MHz

Note The minimum pulse width that can be measured differs depending on the selected value of fCLK.
The minimum pulse width that can be measured is the value of 4/fCLK or the value in the above table, whichever greater.

(4) External event counter

Counts the clock pulses input from the external interrupt request input pin (INTP2) (CI pin input pulses). The clocks that can be input to timer/counter 2 are shown in Table 10-4.

Table 10-4 Clocks Enabled to be Input to Timer/Counter 2

	When Counting One Edge	When Counting Both Edges
Maximum frequency	fCLK/8 (2.00 MHz)	fCLK/8 (2.00 MHz)
Minimum pulse width (High and low levels)	4/fCLK (0.25 μs)	4/fCLK (0.25 μs)

( ): When fCLK = 16 MHz and fxx = 32 MHz

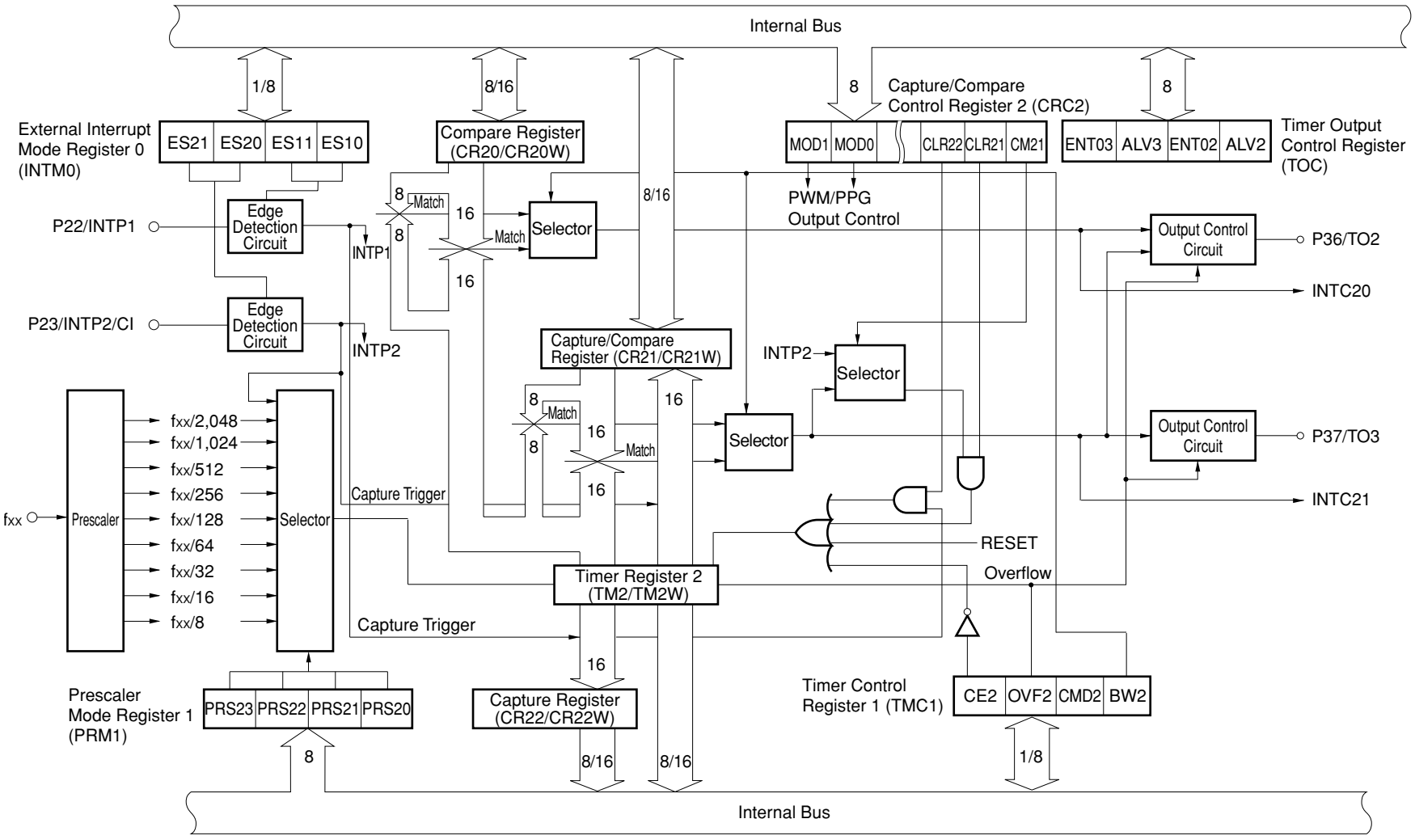
10.2 CONFIGURATION

Timer/counter 2 consists of the following registers.

- Timer register (TM2/TM2W) × 1
- Compare register (CR20/CR20W) × 1
- Capture/compare register (CR21/CR21W) × 1
- Capture register (CR22/CR22W) × 1

The block diagram of timer/counter 2 is shown in Figure 10-1.

Figure 10-1 Timer/Counter 2 Block Diagram



(1) Timer register 2 (TM2/TM2W)

TM2/TM2W is a timer register that counts up the count clock specified by the high-order 4 bits of prescaler mode register 1 (PRM1). An internal clock or external clock can be selected as the count clock.

The count operation can be stopped or enabled by means of timer control register 1 (TMC1). The timer register can select to operate in an 8-bit (TM2) or 16-bit (TM2W) mode. TM2/TM2W can be read only with an 8/16-bit manipulation instruction.

When $\overline{\text{RESET}}$ is input, TM2/TM2W is cleared to 00H and the count is stopped.

Caution If the value of the timer register is read under the condition indicated by "×" in Table 10-5, the read value may be illegal. Do not read the timer register under condition "×".

Table 10-5 Limits of Reading Timer Register

(√: Can be read, ×: Must not be read)

f_{CLK} / Timer Count Clock	$f_{XX}/2$	$f_{XX}/4$	$f_{XX}/8$	$f_{XX}/16$
$f_{XX}/8$	√	√	×	×
$f_{XX}/16$	√	√	√	×
f_{XX}/n	√	√	√	√

Remarks 1. f_{XX}: Oscillation frequency

2. f_{CLK}: Internal system clock frequency

3. n = 32, 64, 128, 256, 512, 1,024, 2,048

(2) Compare register (CR20/CR20W)

CR20/CR20W is an 8/16-bit register that holds the value that determines the interval timer operation cycle.

If the contents of the CR20/CR20W register match the contents of TM2/TM2W, an interrupt request (INTC20) and a timer output control signal are generated. This compare register operates as CR20 in the 8-bit mode, and CR20W in the 16-bit mode.

CR20/CR20W can be read or written to with an 8/16-bit manipulation instruction. The contents of this register are undefined after $\overline{\text{RESET}}$ input.

(3) Capture/compare register (CR21/CR21W)

CR21/CR21W is an 8/16-bit register that can be specified as a compare register for detecting a match with the TM2/TM2W count value or a capture register for capturing the TM2/TM2W count value according to the setting of the capture/compare control register 2 (CRC2).

This capture/compare register operates as CR21 in the 8-bit mode, and CR21W in the 16-bit mode.

CR21/CR21W can be read or written to with an 8/16-bit manipulation instruction.

The contents of this register are undefined after $\overline{\text{RESET}}$ input.

(a) When specified as compare register

CR21/CR21W functions as an 8/16-bit register that holds the value that determines the interval timer operation cycle.

An interrupt request (INTC21) and a timer output control signal are generated by a match between the contents of the CR21/CR21W register and the contents of TM2/TM2W.

Also, the count value can be cleared by a match of the contents.

(b) When specified as capture register

CR21/CR21W functions as an 8/16-bit register that captures the contents of TM2/TM2W in synchronization with the input of a valid edge on the external interrupt input pin (INTP2) (capture trigger).

The contents of the CR21/CR21W register are retained until the next capture trigger is generated.

Also, TM2/TM2W can be cleared after a capture operation.

(4) Capture register (CR22/CR22W)

CR22/CR22W is an 8/16-bit register that captures the contents of TM2/TM2W.

The capture operation is synchronized with the input of a valid edge to the external interrupt request input pin (INTP1) (capture trigger). The contents of the CR22/CR22W register are retained until the next capture trigger is generated.

Also, TM2/TM2W can be cleared after a capture operation.

This capture register operates as CR22 in the 8-bit mode, and CR22W in the 16-bit mode.

CR22/CR22W can be read only with an 8/16-bit manipulation instruction. $\overline{\text{RESET}}$ input clears this register to 0000H.

(5) Edge detection circuit

The edge detection circuit detects an external input valid edge.

This circuit generates an external interrupt request (INTP1) and capture trigger by detecting the valid edge of the INTP1 pin input specified by the external interrupt mode register 0 (INTM0). It also generates a capture trigger, the count clock of an external event, and external interrupt request (INTP2) by detecting the valid edge from an external interrupt request input pin (INTP2).

(6) Output control circuit

It is possible to invert the timer output when the CR20/CR21 register contents and the contents of TM2 match or the CR20W/CR21W contents and the contents of TM2W match.

A square wave can be output from the timer output pins (TO2/TO3) in accordance with the setting of the high-order 4 bits of the timer output control register (TOC). At this time, PWM output or PPG output can be performed according to the specification of the capture/compare control register 2 (CRC2).

Timer output can be disabled/enabled by means of the TOC register. When timer output is disabled, a fixed level is output to the TO2 and TO3 pins (the output level is set by the TOC register).

(7) Prescaler

The prescaler generates the count clock from the internal system clock. The clock generated by the prescaler is selected by the selector, and is used as the count clock by the timer register 2 (TM2/TM2W) to perform count operations.

(8) Selector

The selector selects a signal resulting from dividing the internal clock or the edge detected by the edge detection circuit as the count clock of timer register 2 (TM2/TM2W).

10.3 TIMER/COUNTER 2 CONTROL REGISTERS

(1) Timer control register 1 (TMC1)

In TMC1 the timer/counter 2, TM2/TM2W, count operation is controlled by the high-order 4 bits (the low-order 4 bits control the count operation of timer/counter 1, TM1/TM1W).

TMC1 can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. The format of the TMC1 is shown in Figure 10-2.

$\overline{\text{RESET}}$ input clears TMC1 to 00H.

Figure 10-2 Timer Control Register 1 (TMC1) Format

	⑦	⑥	5	4	③	②	1	0	Address	After Reset	R/W
TMC1	CE2	OVF2	CMD2	BW2	CE1	OVF1	0	BW1	0FF5FH	00H	R/W

Controls count operation of timer/counter 1 (TM1/TM1W) (see **Figure 9-2**).

BW2	Timer/Counter 2 Bit Length Specification
0	8-bit operating mode
1	16-bit operating mode

CMD2	TM2/TM2W Operating Mode Specification
0	Normal mode
1	One-shot mode

OVF2	TM2/TM2W Overflow Flag
0	No overflow
1	Overflow [Note]

Note 8-bit operating mode:
count up from FFH to 00H
In 16-bit operating mode:
count up from FFFFH to 0000H

CE2	TM2/TM2W Count Operation Control
0	Count operation stopped with count cleared
1	Count operation enabled

Remark The OVF2 bit is reset by software only.

(2) Prescaler mode register 1 (PRM1)

In PRM1 the count clock to timer/counter 2, TM2/TM2W, is specified by the high-order 4 bits (the low-order 4 bits specify the count clock to timer/counter 1, TM1/TM1W).

PRM1 can be read or written with an 8-bit manipulation instruction. The format of the PRM1 is shown in Figure 10-3.

$\overline{\text{RESET}}$ input sets PRM1 to 11H.

Figure 10-3 Prescaler Mode Register 1 (PRM1) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PRM1	PRS23	PRS22	PRS21	PR20	PRS13	PRS12	PRS11	PRS10	0FF5EH	11H	R/W

Specifies count clock to timer/counter 1 (TM1/TM1W) (see **Figure 9-3**).

(f_{XX} = 32MHz)

PRS23	PRS22	PRS21	PRS20	Timer/Counter 2 (TM2/TM2W) Count Clock Specification	
				Count Clock [Hz] Specification	Resolution [μs]
0	0	0	0	Setting prohibited	–
0	0	0	1	f_{XX}/8	0.25
0	0	1	0	f_{XX}/16	0.50
0	0	1	1	f_{XX}/32	1.00
0	1	0	0	f_{XX}/64	2.00
0	1	0	1	f_{XX}/128	4.00
0	1	1	0	f_{XX}/256	8.00
0	1	1	1	f_{XX}/512	16.00
1	0	0	0	f_{XX}/1,024	32.00
1	0	0	1	f_{XX}/2,048	64.00
1	1	1	1	External clock (CI/INTP2)	–
Other than the above				Setting prohibited	

Remark f_{XX}: X1 input frequency or oscillation frequency

(3) Capture/compare control register 2 (CRC2)

The CRC2 specifies the enabling condition for a timer register 2 (TM2/TM2W) clear operation by the capture/compare register (CR21/CR21W) or the capture register (CR22/CR22W) and the timer output (TO2/TO3) mode.

CRC2 can be read or written with an 8-bit manipulation instruction. The format of the CRC2 is shown in Figure 10-4.

$\overline{\text{RESET}}$ input sets CRC2 to 10H.

Figure 10-4 Capture/Compare Control Register 2 (CRC2) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
CRC2	MOD1	MOD0	CLR22	1	CLR21	CM21	0	0	0FF33H	10H	R/W

MOD1	MOD0	CLR22	CLR21	CM21	CR21 Operation Specification	Timer Output Mode Specification		TM2 Clear Operation
						TO2	TO3	
0	0	0	0	0	Compare operations	Toggle output	Toggle output	Not cleared
0	0	0	1	0		Toggle output	Toggle output	Cleared if TM2 and CR21 match
0	0	1	0	0		Toggle output	Toggle output	Cleared after TM2 contents are captured in CR22 by INTP1
0	0	1	1	0		Toggle output	Toggle output	Cleared by match of TM2 and CR21 or after TM2 contents are captured in CR22 by INTP1
0	1	0	0	0		PWM output	Toggle output	Not cleared
1	0	0	0	0		PWM output	PWM output	Not cleared
1	1	0	1	0		PPG output	Toggle output	Cleared if TM2 and CR21 match
0	0	0	0	1	Capture operations	Toggle output		Not cleared
0	0	0	1	1		Toggle output		Cleared after TM2 contents are captured in CR21 by INTP2
0	1	0	0	1		PWM output		Not cleared
Other than the above					Setting prohibited			

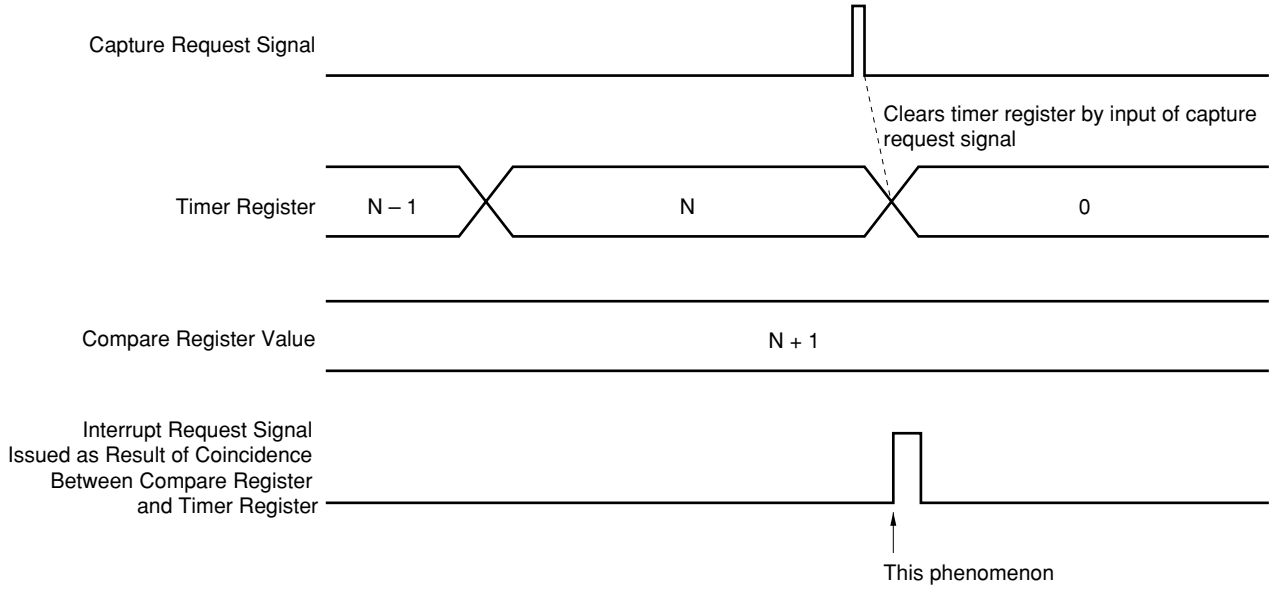
Remark The register names in the 8-bit operating mode are shown in this figure. In the 16-bit operating mode, the register names TM2, CR20, CR21, and CR22 are TM2W, CR20W, CR21W, and CR22W, respectively.

Caution Even if an attempt is made to clear the timer register by inputting the capture request signal when the capture function of the timer is used, the timer register momentarily counts up immediately before it is cleared. Consequently, if a value greater than the value of the timer register by 1 is set to the compare register when the capture request signal is input, the values of the compare register and timer register coincide, and an unnecessary interrupt will be generated (refer to Figure 10-5). Therefore, take the following operation into consideration when creating a program.

<Operation>

Because the timer register is cleared at the next count if the capture request signal is generated when the value of timer register is "N" when the value "N + 1" is set to the compare register, no interrupt request is generated by the compare register. Actually, however, the timer register momentarily counts "N + 1" when the timer register is cleared. As a result, the values of the timer register and compare register coincide, and an interrupt request signal is generated by the compare register.

Figure 10-5 Example of Generation of Unnecessary Interrupt Request by Compare Register



(4) Timer output control register (TOC)

TOC is an 8-bit register that controls output enabling/disabling of the active level of timer output.

The operation of the timer output pins (TO2 and TO3) by timer/counter 2 is controlled by the high-order 4 bits (the low-order 4 bits control the operation of the timer output pins (TO0 and TO1) by timer/counter 0).

TOC can be read or written with an 8-bit manipulation instruction or bit manipulation instruction. The format of the TOC is shown in Figure 10-6.

$\overline{\text{RESET}}$ input clears TOC to 00H.

Figure 10-6 Timer Output Control Register (TOC) Format

	⑦	6	⑤	4	③	2	①	0	Address	After Reset	R/W
TOC	ENTO3	ALV3	ENTO2	ALV2	ENTO1	ALV1	ENTO0	ALV0	0FF31H	0000H	R/W

Control operation of timer output pins (TO0 & TO1) by timer/counter 0 (see **Figure 8-5**).

ALV2	TO2 Pin Active Level	
	Toggle output specification	PWM/PPG output specification
0	Low level	High level
1	High level	Low level

ENTO2	TO2 Pin Operation Specification
0	$\overline{\text{ALV2}}$ output
1	Pulse output enabled

ALV3	TO3 Pin Active Level	
	Toggle output specification	PWM/PPG output specification
0	Low level	High level
1	High level	Low level

ENTO3	TO3 Pin Operation Specification
0	$\overline{\text{ALV3}}$ output
1	Pulse output enabled

10.4 TIMER REGISTER 2 (TM2) OPERATION

10.4.1 Basic Operation

8-bit operating mode/16-bit operating mode control can be performed for timer/counter 2 by means of bit 0 (BW2) of timer control register 2 (TMC2). **Note**

In the timer/counter 2 count operation, an up-count is performed using the count clock specified by the high-order 4 bits of prescaler mode register 1 (PRM1).

Count operation enabling/disabling is controlled by bit 3 (CE2) of TMC2 (timer/counter 2 operation control is performed by the high-order 4 bits of the timer control register 1 (TMC1). When the CE2 bit is set (to 1) by software, the contents of TM2 are cleared to 0H on the first count clock, and then the up-count operation is performed.

When the CE2 bit is cleared (to 0) by software, TM2 becomes 0H immediately, and capture operations and match signal generation are stopped.

If the CE2 bit is set (to 1) again when it is already set (to 1), the TM2 count operation is not affected (see **Figure 10-7 (b)**).

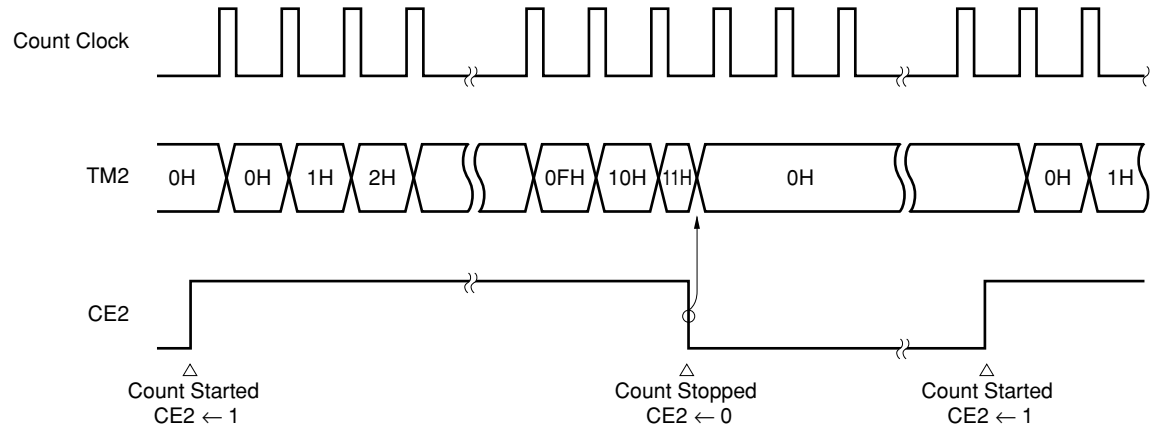
TM2/TM2W is cleared to 0H when the count clock is input while the value of TM2 is FFH in the 8-bit operating mode or while the value of TM2W is FFFFH in the 16-bit operating mode. At this time, OVF2 bit is set and the overflow signal is sent to the output control circuit. OVF2 bit is cleared by software only. The count operation is continued.

When $\overline{\text{RESET}}$ is input, TM2 is cleared to 0H, and the count operation is stopped.

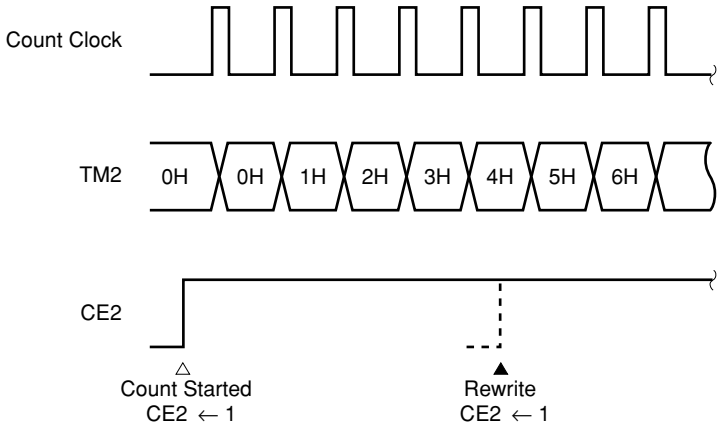
Note Unless otherwise specified, the functions of timer register 2 in the 8-bit operating mode are described hereafter. In the 16-bit operating mode, TM2, CR20, CR21, and CR22 operate as TM2W, CR20W, CR21W, and CR22W, respectively.

Figure 10-7 Basic Operation in 8-Bit Operating Mode (BW2 = 0)

(a) Count started → count disabled → count started



(b) When "1" is written to the CE2 bit again after the count starts



(c) Operation when TM2 = FFH

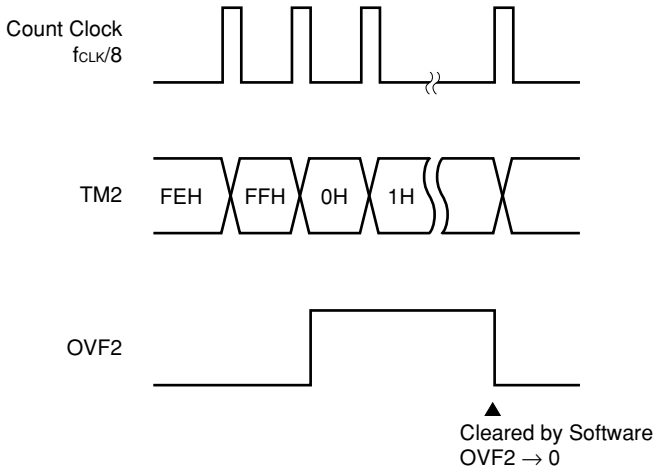
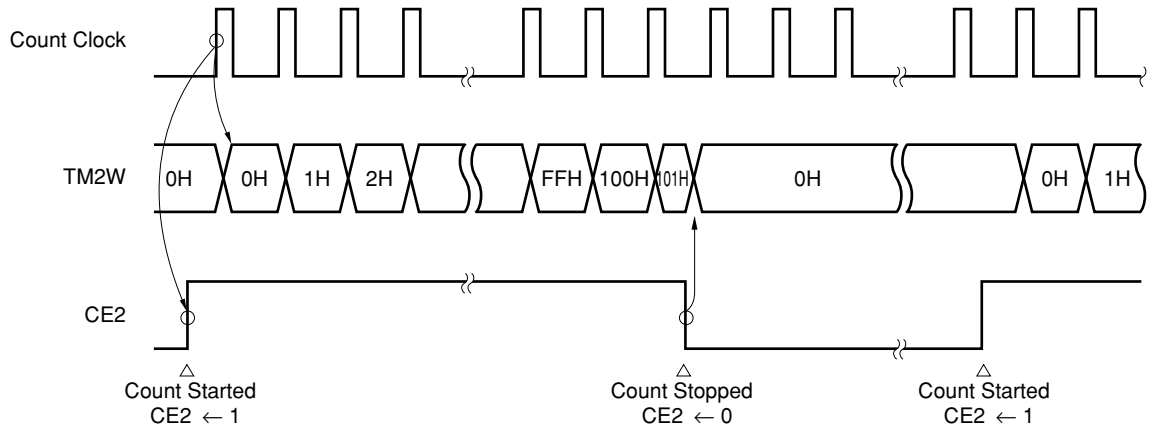
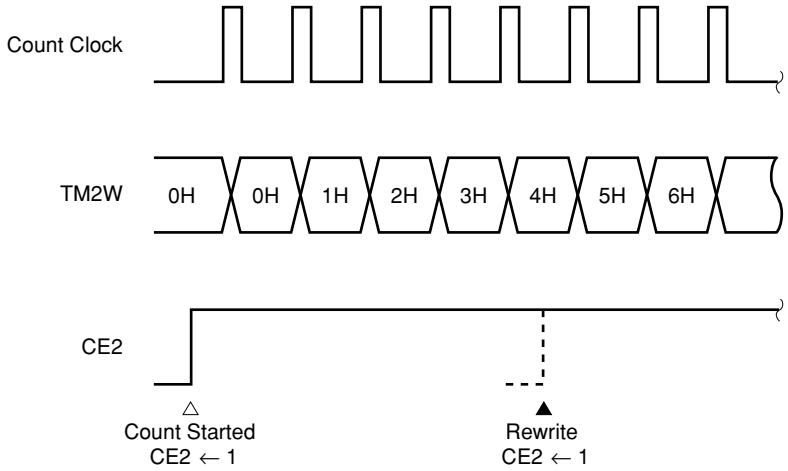


Figure 10-8 Basic Operation in 16-Bit Operating Mode (BW2 = 1)

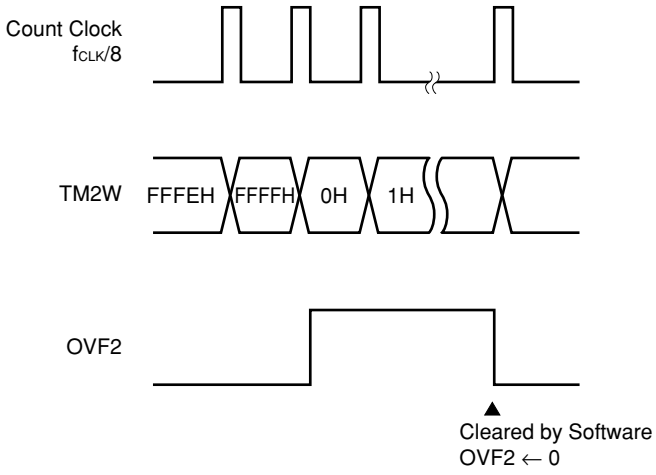
(a) Count started → count disabled → count started



(b) When "1" is written to the CE2 bit again after the count starts



(c) Operation when TM2W = FFFFH



10.4.2 Clear Operation

(1) Clear operation after match with compare register and capture operation

Timer register 2 (TM2) can be cleared automatically after a match with the compare register (CR2n: n = 0, 1) and a capture operation. When a clearance source arises, TM2 is cleared to 0H on the next count clock. Therefore, even if a clearance source arises, the value at the point at which the clearance source arose is retained until the next count clock arrives.

Figure 10-9 TM2 Clearance by Match With Compare Register (CR20/CR21)

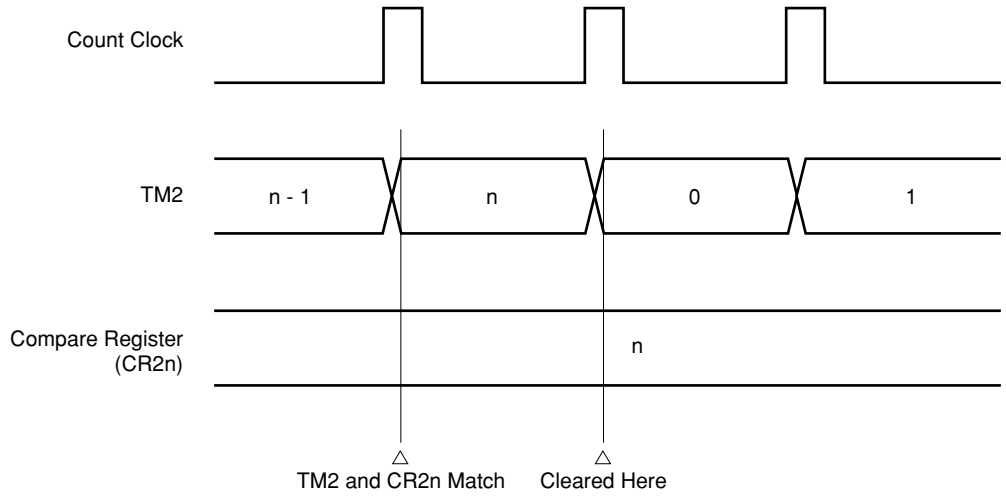
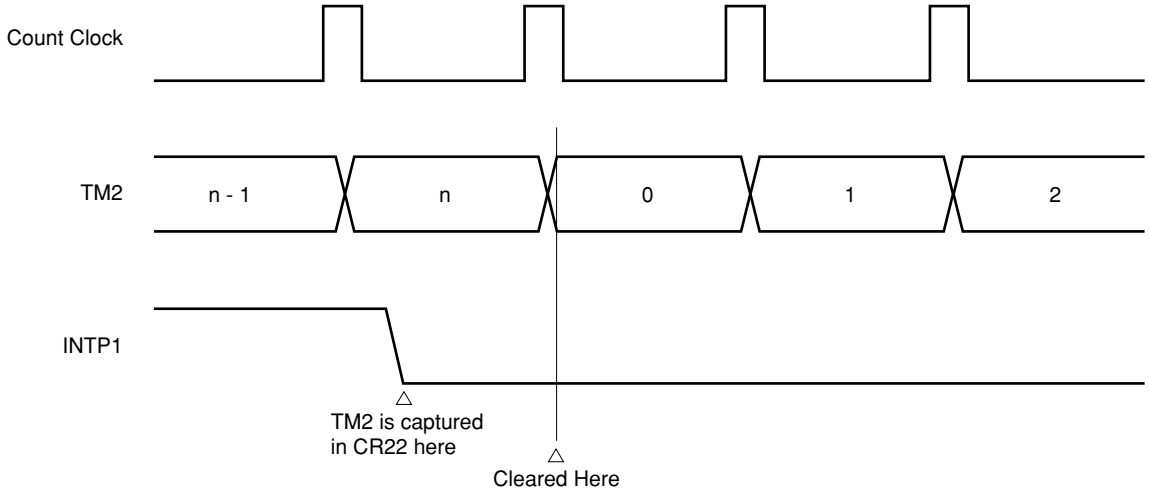


Figure 10-10 TM2 Clearance after Capture Operation

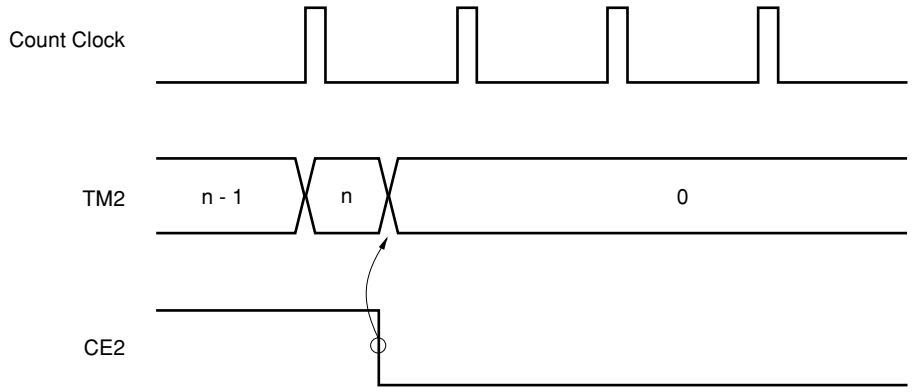


(2) Clear operation by CE2 bit of timer control register 2 (TMC2)

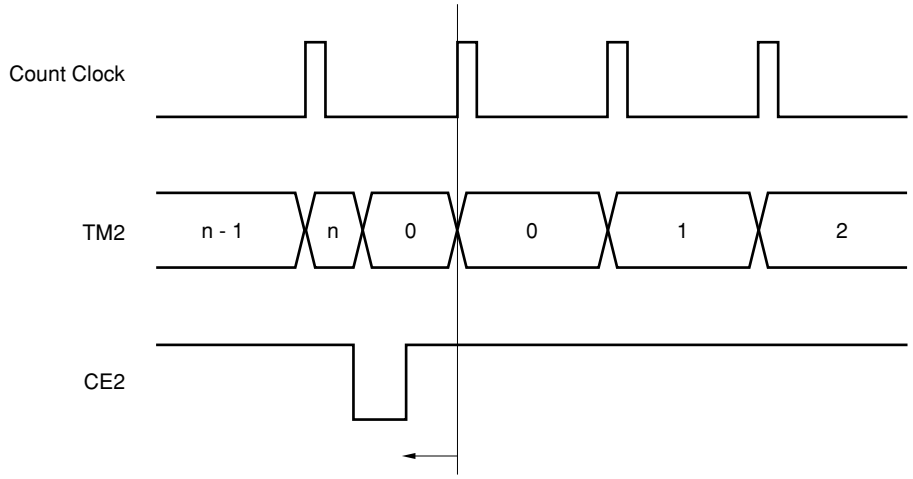
Timer register 2 (TM2) is also cleared when the CE2 bit of the TMC1 is cleared (to 0) by software. The clear operation is performed immediately after clearance (to 0) of the CE2 bit.

Figure 10-11 Clear Operation When CE2 Bit is Cleared (to 0)

(a) Basic operation

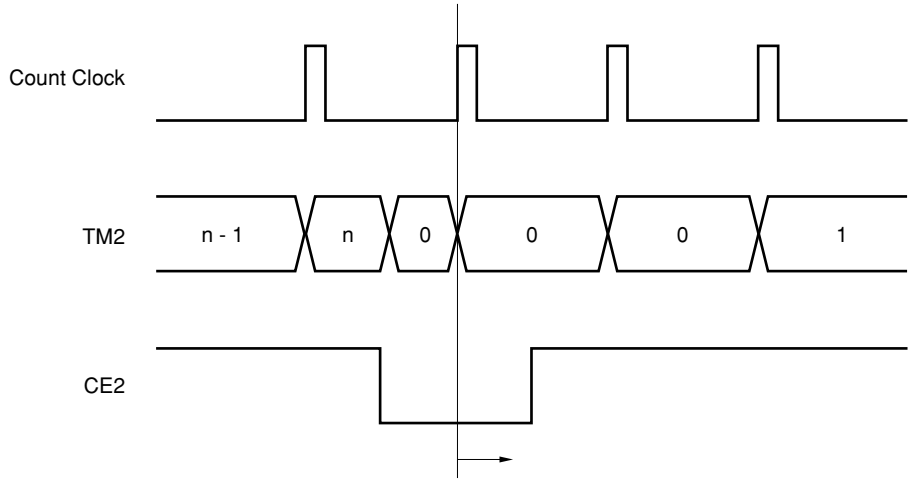


(b) Restart before count clock is input after clearance



If the CE2 bit is set (to 1) before this count clock, this count clock starts counting from 0.

(c) Restart after count clock is input after clearance



If the CE2 bit is set (to 1) from this count clock onward, the count starts from 0 on the count clock after the CE2 bit is set (to 1).

10.5 EXTERNAL EVENT COUNTER FUNCTION

Timer/counter 2 can count clock pulses input from external interrupt request input pin (INTP2/CI).

No special selection method is needed for the external event counter operating mode. When the timer register 2 (TM2) count clock is specified as external clock input by the setting of the high-order 4 bits of prescaler mode register 1 (PRM1), TM2 operates as an external event counter.

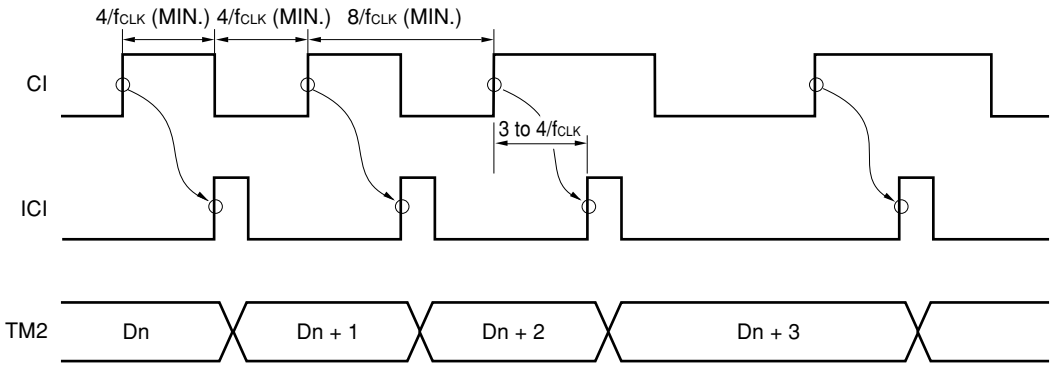
The maximum frequency of external clock pulses that can be counted by TM2 as the external event counter is 2.00 MHz (f_{CLK} = 16 MHz) irrespective of whether only one edge or both edges are counted on INTP2/CI input.

The pulse width of INTP2/CI input must be at least 4 system clocks (0.25 μs: f_{CLK} = 16 MHz) for both the high level and low level. If the pulse width is shorter than this, the pulse may not be counted.

The timer/counter 2 external event count timing is shown in Figure 10-12.

Figure 10-12 Timer/Counter 2 External Event Count Timing (1/2)

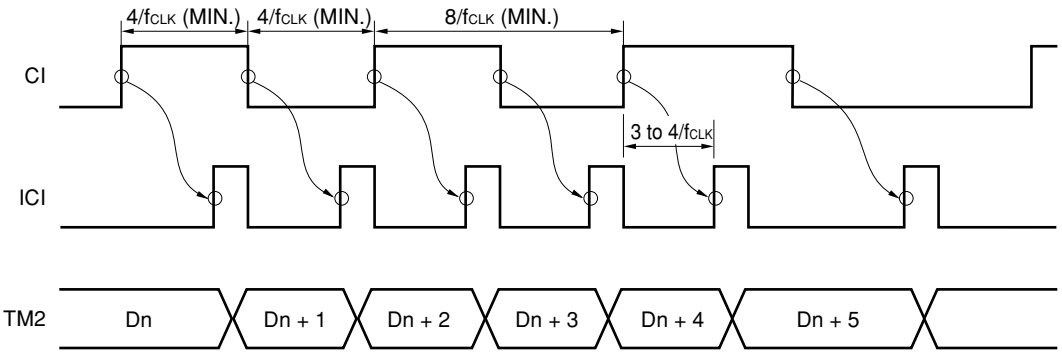
(1) Counting one edge (maximum frequency = f_{CLK}/8)



Remark ICI: CI input signal after passing through edge detection circuit

Figure 10-12 Timer/Counter 2 External Event Count Timing (2/2)

(2) Counting both edges (maximum frequency = $f_{CLK}/8$)



Remark ICI: CI input signal after passing through edge detection circuit

The TM2 count operation is controlled by the CE2 bit of the timer control register 1 (TMC1) in the same way as with the basic operation.

When the CE2 bit is set (to 1) by software, the contents of TM2 are set to 0H and the up-count operation is started on the initial count clock.

When the CE2 bit is cleared (to 0) by software during a TM2 count operation, the contents of TM2 are set to 0H immediately and the stopped state is entered. The TM2 count operation is not affected if the CE2 bit is set (to 1) by software again when it is already set (to 1).

Caution When timer/counter 2 is used as an external event counter, it is not possible to distinguish between the case where there is no valid edge input at all and the case where there is a single valid edge input using timer register 2 (TM2) alone (see Figure 10-13), since the contents of TM2 are 0 in both cases. If it is necessary to make this distinction, the INTP2 interrupt request flag should be used (the INTP2 pin and CI pin have a dual function, and both functions can be used at the same time). An example is shown in Figure 10-14.

Figure 10-13 Example of the Case Where the External Event Counter Does Not Distinguish Between One Valid Edge Input and No Valid Edge Input

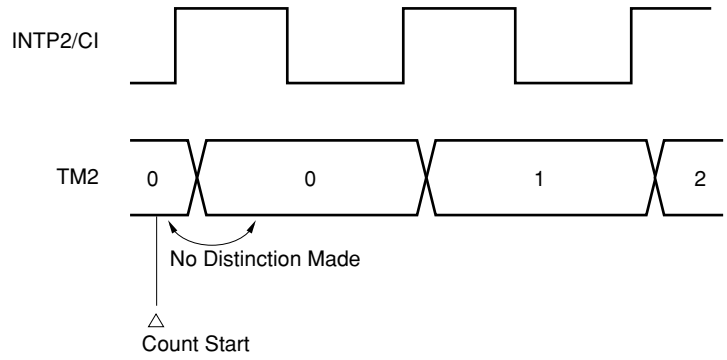
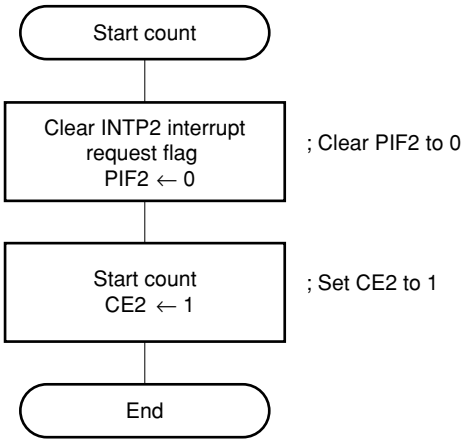
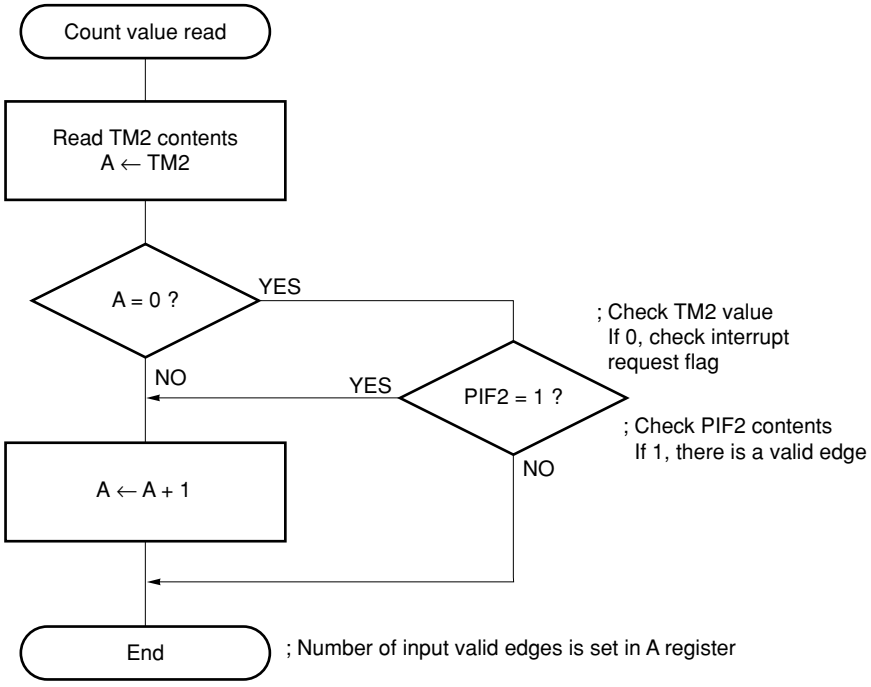


Figure 10-14 Methods of Enabling the External Event Counter to Distinguish No Valid Edge Input

(a) Processing when count is started



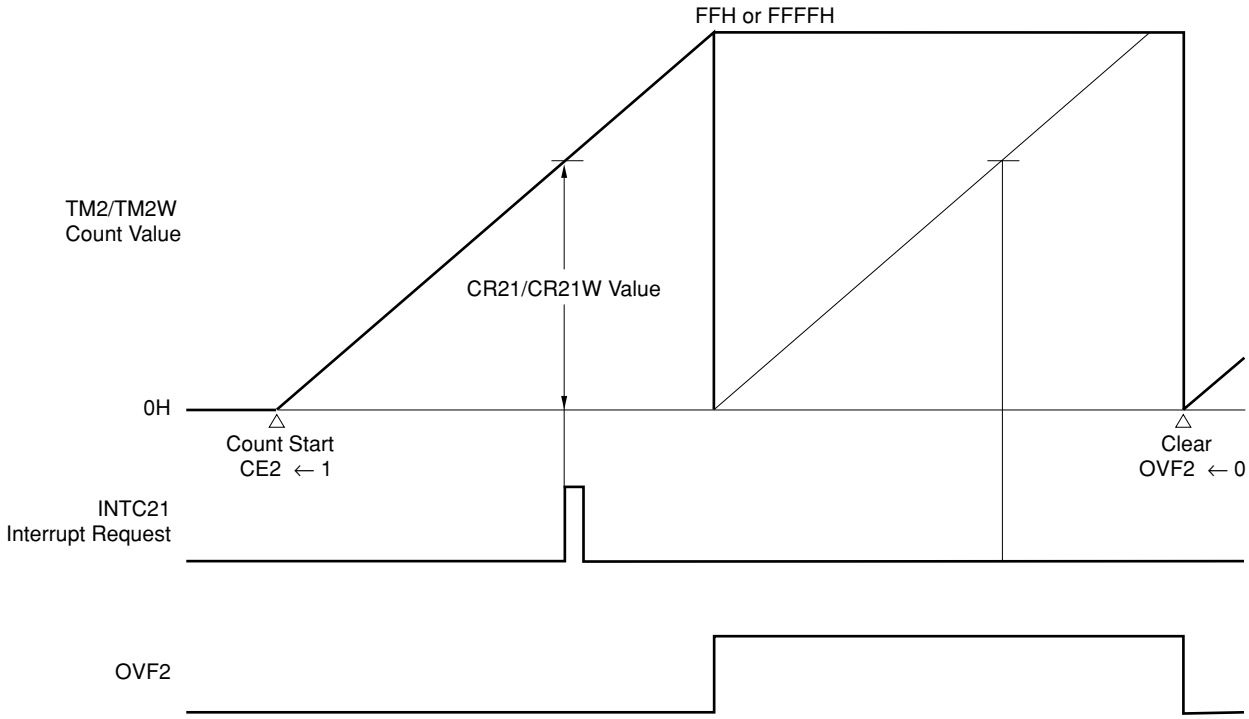
(b) Processing when count value is read



10.6 ONE-SHOT TIMER FUNCTION

Timer/counter 2 has an operating mode in which it stops automatically when a full count value is reached (FFH/FFFFH) as a result of counting by timer register 2 (TM2/TM2W).

Figure 10-15 One-Shot Timer Operation



As shown in Figure 10-15, the respective one-shot interrupt is generated when the value (0H to FFH/FFFFH) set beforehand in the CR20, CR21/CR21W or CR21W and the timer register 2 (TM2/TM2W) value match.

The one-shot timer operating mode is specified by setting (to 1) bit 5 (CMD2) of timer control register 1 (TMC1) by software.

The TM2/TM2W count operation is controlled by the CE2 bit of the TMC1 as with the basic operation.

When the CE2 bit is set (to 1) by software, the contents of TM2/TM2W are set to 0H and the up-count operation is started on the initial count clock.

When the contents of TM2/TM2W reach FFH/FFFFH (full count) as a result of the up-count operation, bit 6 (OVF2) of the TMC1 are set (to 1), and TM2/TM2W stops with the count at FFH/FFFFH.

The one-shot timer operation is started again from the count-stopped state by clearing (to 0) the OVF2 bit by software. When the OVF2 bit is cleared (to 0), the contents of TM2/TM2W become 0H and the up-count operation is restarted on the next count clock.

If the CE2 bit is cleared (to 0) by software during a TM2/TM2W count operation, the contents of TM2/TM2W are set to 0H immediately and the stopped state is entered. The TM2/TM2W count operation is not affected if the CE2 bit is set (to 1) by software again when it is already set (to 1).

10.7 COMPARE REGISTER, CAPTURE/COMPARE REGISTER, AND CAPTURE REGISTER OPERATION

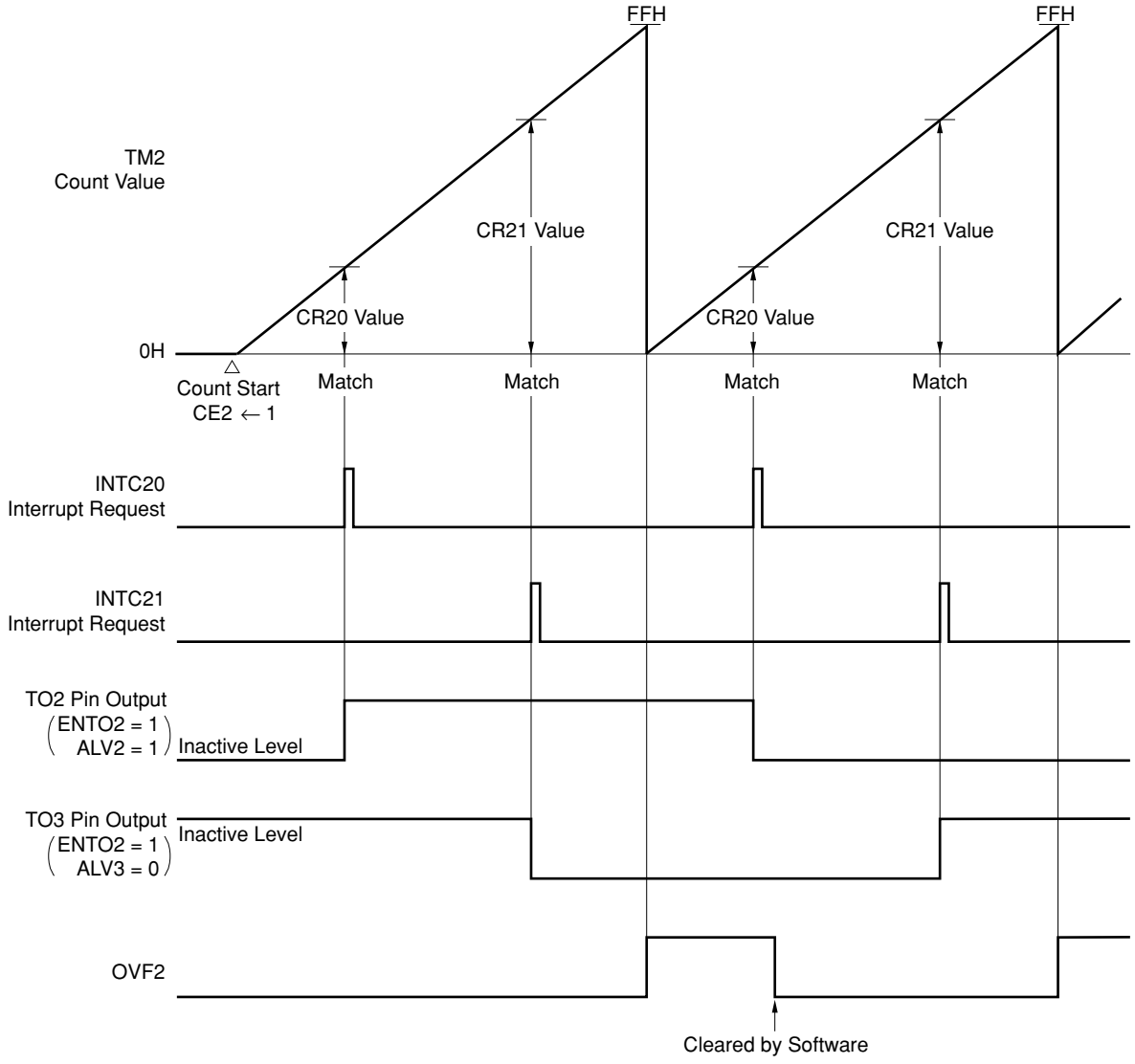
10.7.1 Compare Operations

Timer/counter 2 performs compare operations in which the value set in the compare register (CR20) and the capture/compare register (CR21) specified for compare operation is compared with the timer register 2 (TM2) count value.

If the count value of TM2 matches the preset value of the CR20, and CR21 when a compare operation is performed, as the result of the count operation, a match signal is sent to the output control circuit, and an interrupt request signal (INTC20 or INTC21) is generated at the same time.

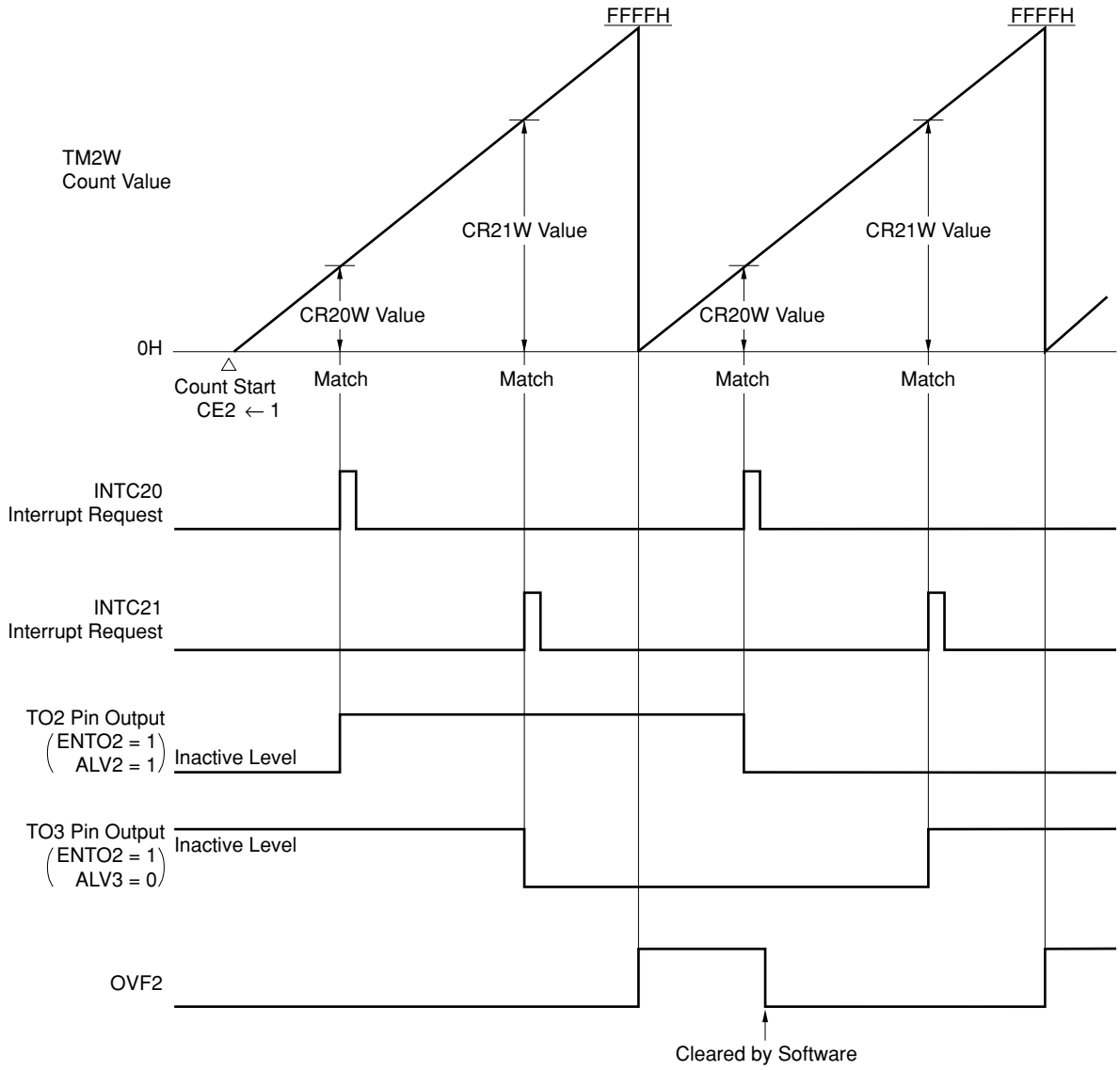
After a match with the CR20 or CR21 value, the TM2 contents can be cleared, and the timer functions as an interval timer that repeatedly counts up to the value set in the CR20 or CR21.

Figure 10-16 Compare Operation in 8-Bit Operating Mode



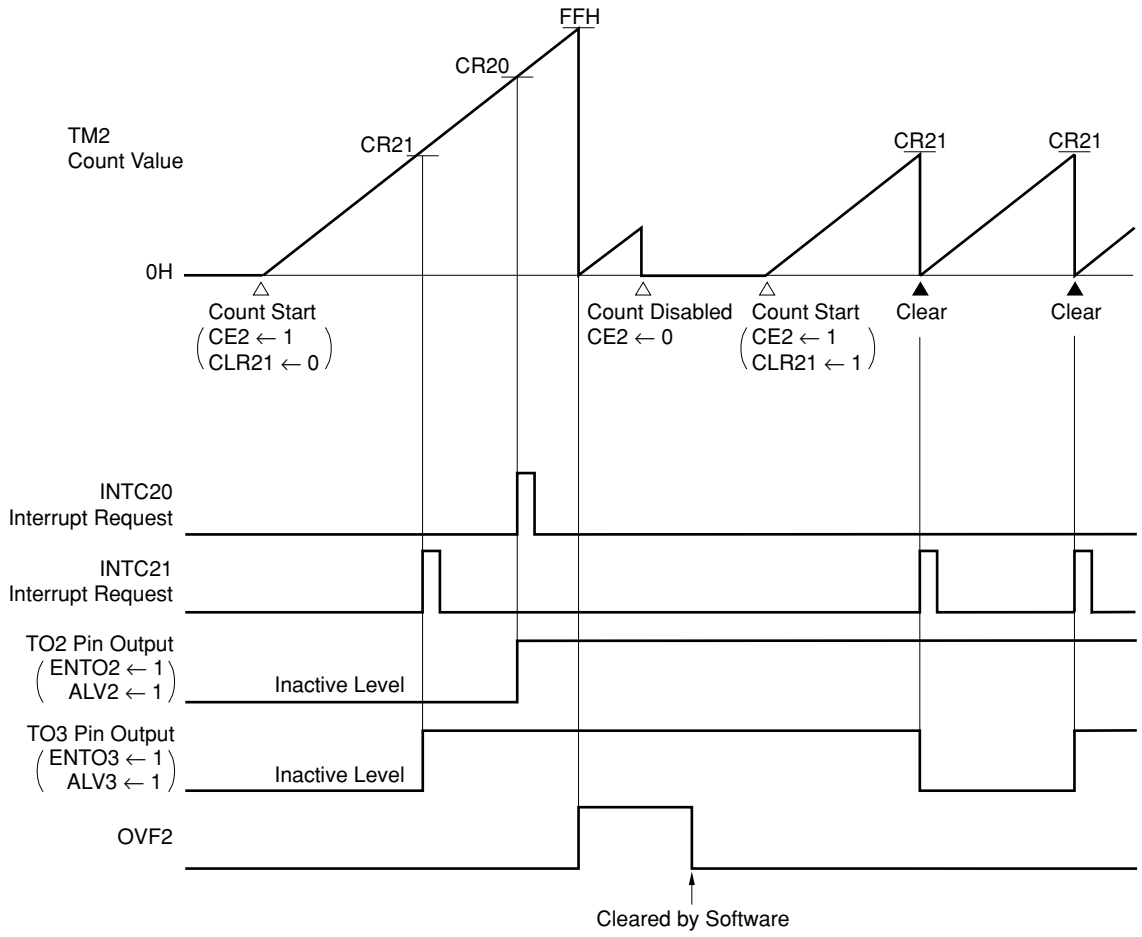
Remark CLR21 = 0, CLR22 = 0, BW2 = 0

Figure 10-17 Compare Operation in 16-Bit Operating Mode



Remark CLR21 = 0, CLR22 = 0, BW2 = 1

Figure 10-18 TM2 Clearance after Match Detection



Remark CLR22 = 0

10.7.2 Capture Operations

Timer/counter 2 performs capture operations in which the timer register 2 (TM2) count value is fetched into the capture register in synchronization with an external trigger, and retained there.

A valid edge detected from the input of the external interrupt request input pins (INTP1/INTP2) is used as the external trigger (capture trigger). The count value of TM2 in the process of being counted in synchronization with the capture trigger is fetched into the capture register (CR22) in synchronization with INTP1, or into the capture/compare register (CR21) when a capture operation is specified in synchronization with INTP2, and is retained there.

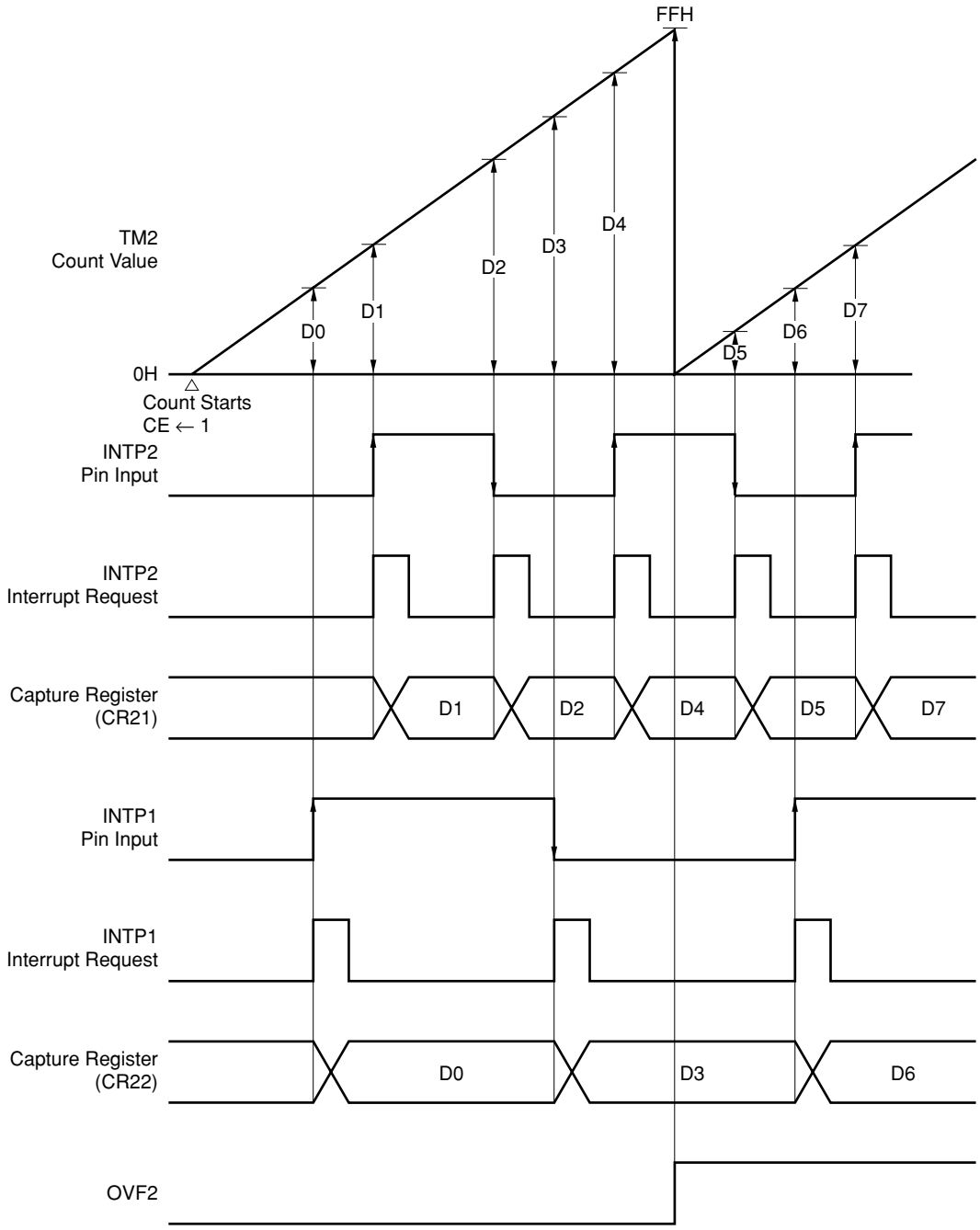
The contents of CR21 and CR22 are retained until the next capture triggers corresponding to CR21 and CR22 are generated.

The capture trigger valid edge is set by means of external interrupt mode register 0 (INTM0). If both rising and falling edges are set as capture triggers, the width of pulses input from off-chip can be measured, and if a capture trigger is generated by a single edge, the input pulse cycle can be measured.

See **Figure 21-1** in **CHAPTER 21 EDGE DETECTION FUNCTION** for details of the INTM0 format.

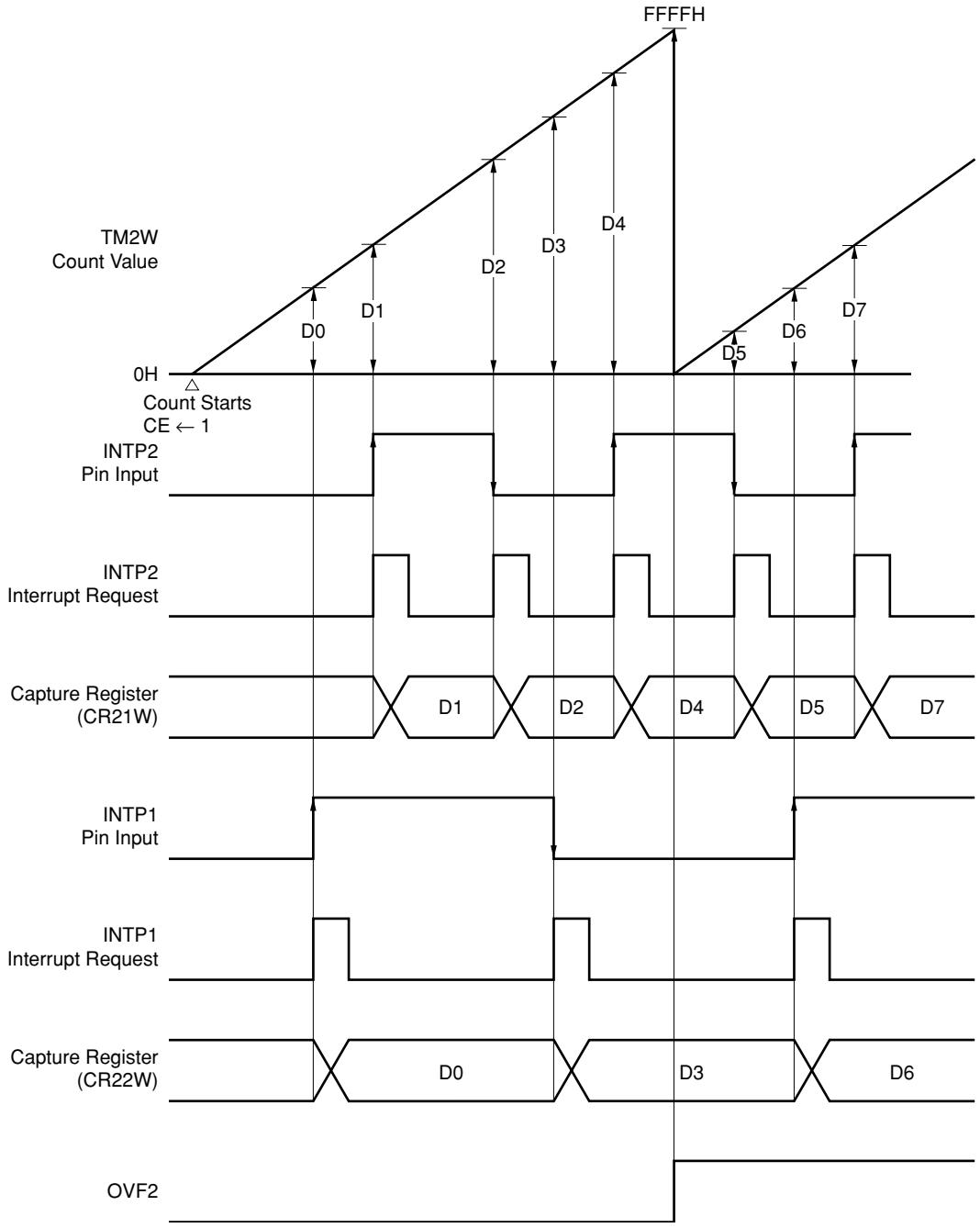
When CR21 is used as a capture register, TM2 can be cleared as soon as the contents of TM2 have been captured by capture trigger to CR21 or CR22.

Figure 10-19 Capture Operation in 8-Bit Operating Mode



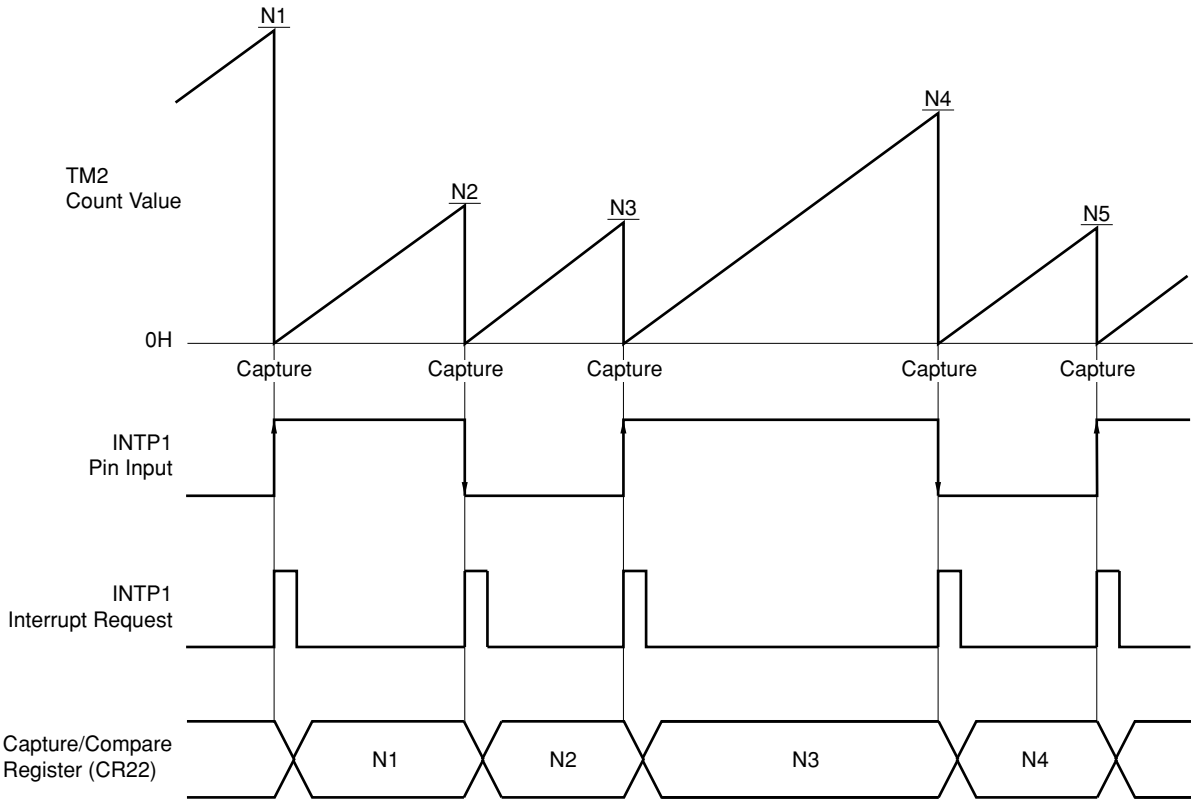
Remark Dn: TM2 count value (n = 0, 1, 2, ...)
CM21 = 1, CLR21 = 0, CLR22 = 0, BW2 = 0

Figure 10-20 Capture Operation in 16-Bit Operating Mode



Remark Dn: TM2W count value (n = 0, 1, 2, ...)
CM21 = 1, CLR21 = 0, CLR22 = 0, BW2 = 0

Figure 10-21 TM2 Clearance after Capture Operation



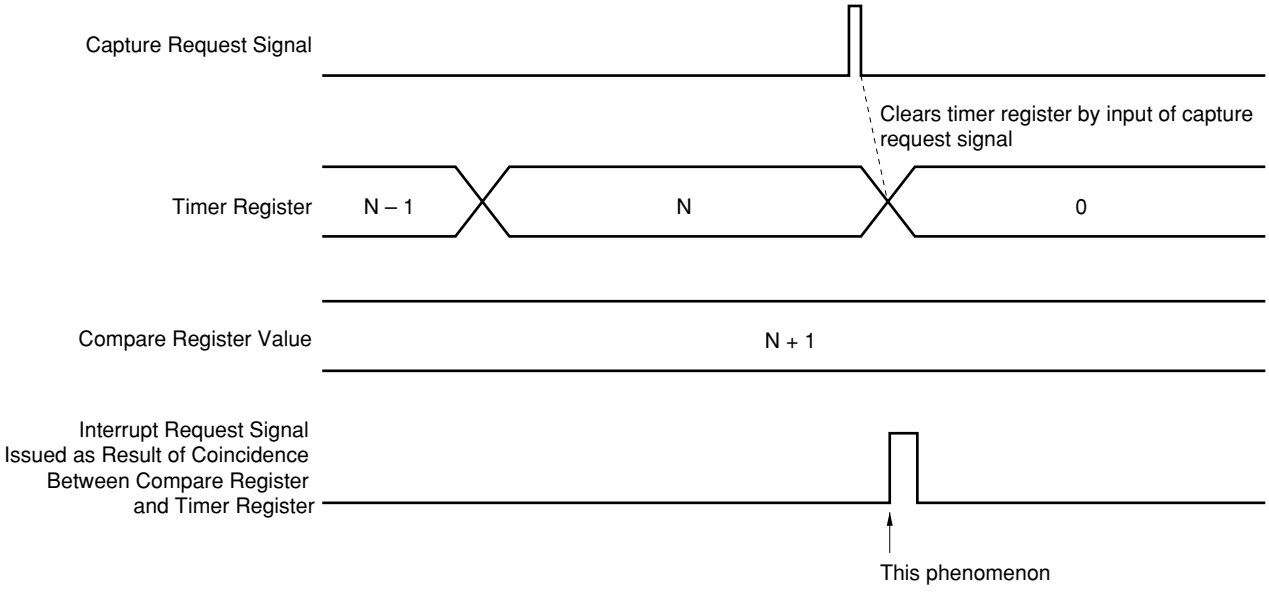
Remark CLR21 = 0, CLR22 = 1

Caution Even if an attempt is made to clear the timer register by inputting the capture request signal when the capture function of the timer is used, the timer register momentarily counts up immediately before it is cleared. Consequently, if a value greater than the value of the timer register by 1 is set to the compare register when the capture request signal is input, the values of the compare register and timer register coincide, and an unnecessary interrupt will be generated (refer to Figure 10-22). Therefore, take the following operation into consideration when creating a program.

<Operation>

Because the timer register is cleared at the next count if the capture request signal is generated when the value of timer register is "N" when the value "N + 1" is set to the compare register, no interrupt request is generated by the compare register. Actually, however, the timer register momentarily counts "N + 1" when the timer register is cleared. As a result, the values of the timer register and compare register coincide, and an interrupt request signal is generated by the compare register.

Figure 10-22 Example of Generation of Unnecessary Interrupt Request by Compare Register



10.8 BASIC OPERATION OF OUTPUT CONTROL CIRCUIT

The output control circuit controls the timer output pins (TO2/TO3) level by means of match signals from the compare register (CR22). The operation of the output control circuit is determined by the timer output control register (TOC) and capture/compare control register 2 (CRC2) (see **Table 10-6**). When TO2/TO3 signal is output to a pin, the relevant pin must be in control mode in the port 3 mode register (PMC3).

Table 10-6 Timer Output (TO2/TO3) Operations

TOC				CRC2				TMC1	TO3	TO2
ENTO3	ALV3	ENTO2	ALV2	MOD1,	MOD1,	CLR22	CLR21	CMD2		
0	0/1	0	0/1	×	×	×	×	×	High/low level fixed	High/low level fixed
0	0/1	1	0/1	0	0	× Note	×	×	High/low level fixed	Toggle output (active-low/high)
1	0/1	0	0/1	0	0	× Note	×	×	Toggle output (active-low/high)	High/low level fixed
1	0/1	1	0/1	0	0	× Note	×	×	Toggle output (active-low/high)	Toggle output (active-low/high)
0	0/1	1	0/1	0	1	0	0	0	High/low level fixed	PWM output (active-high/low)
1	0/1	0	0/1	0	1	0	0	0	Toggle output (active-low/high)	High/low level fixed
1	0/1	1	0/1	0	1	0	0	0	Toggle output (active-low/high)	PWM output (active-high/low)
0	0/1	1	0/1	1	0	0	0	0	High/low level fixed	PWM output (active-high/low)
1	0/1	0	0/1	1	0	0	0	0	PWM output (active-high/low)	High/low level fixed
1	0/1	1	0/1	1	0	0	0	0	PWM output (active-high/low)	PWM output (active-high/low)
0	0/1	1	0/1	1	1	0	1	0	High/low level fixed	PPG output (active-high/low)
1	0/1	0	0/1	1	1	0	1	0	Toggle output (active-low/high)	High/low level fixed
1	0/1	1	0/1	1	1	0	1	0	Toggle output (active-low/high)	PPG output (active-high/low)

Note CLR22 is normally set to 0 in this case.

Remarks 1. 0/1 in the ALVn (n = 2, 3) columns correspond to the items on the left and right of the slash ("/") in the TOn (n = 2, 3) columns respectively.

2. "×" indicates 0 or 1.

3. Combinations not shown in this table are prohibited to use in that combination.

10.8.1 Basic Operation

Setting (to 1) the ENTOn (n = 2, 3) bit of the timer output control register (TOC) enables timer output (TOn: n = 2, 3) to be varied at a timing in accordance with the settings of MOD0, MOD1, and CLR21 bits of capture/compare control register 2 (CRC2).

Clearing (to 0) ENTOn sets the TOn to a fixed level. The fixed level is determined by the ALVn (n = 2/3) bit of the TOC. The level is high when ALVn is 0, and low when 1.

10.8.2 Toggle Output

Toggle output is an operating mode in which the output level is inverted each time the compare register (CR20/CR21) value coincides with the timer register 2 (TM2) value. The output level of timer output (TO2) is inverted by a match between CR20 and TM2, and the output level of timer output (TO3) is inverted by a match between CR21 and TM2.

When timer/counter 2 is stopped by clearing (to 0) the CE2 bit of the timer control register 1 (TMC1), the inactive level ($\overline{\text{ALVn}}$: n = 0, 1) is output.

Figure 10-23 Toggle Output Operation

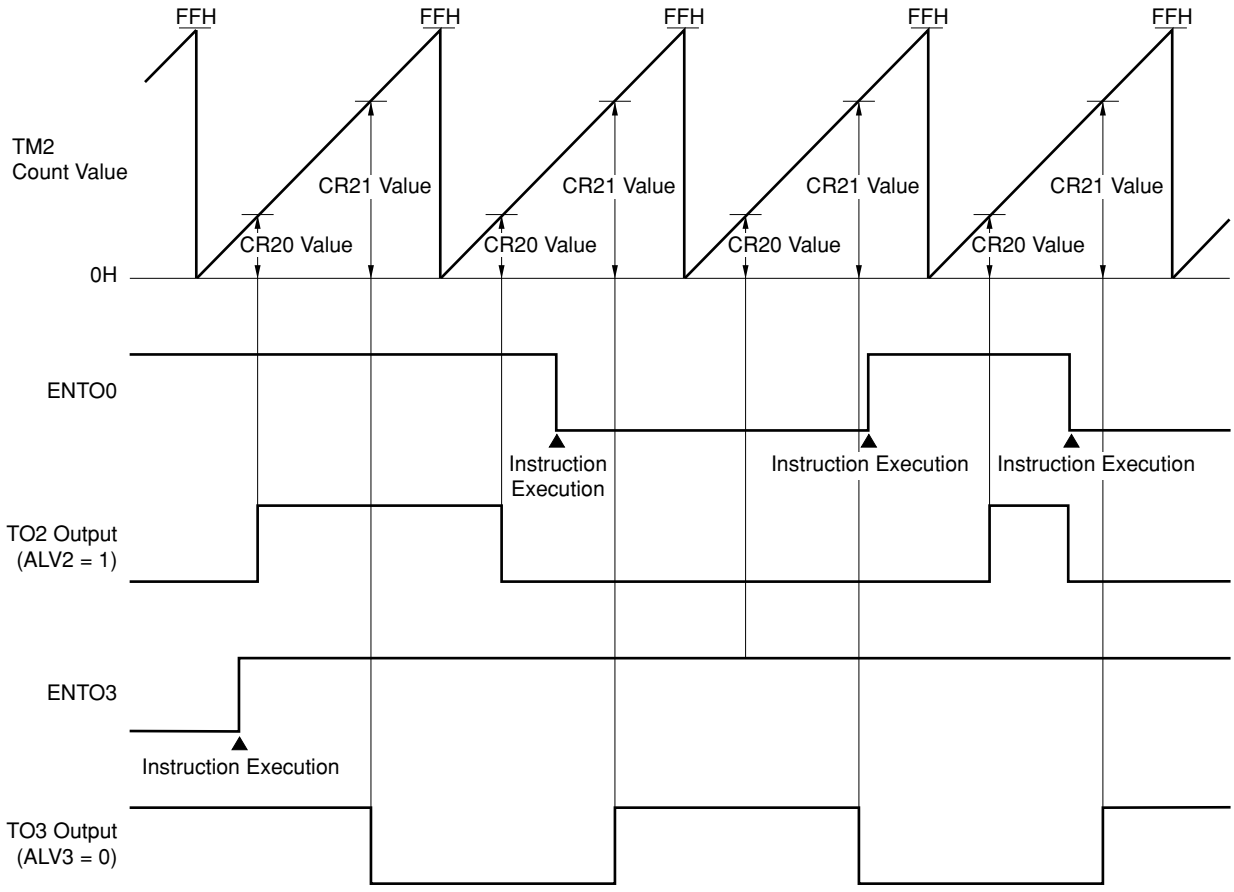


Table 10-7 TO2/TO3 Toggle Output (f_{XX} = 32 MHz)

Count Clock	Minimum Pulse Width	Maximum Pulse Width
$f_{XX}/8$	$8/f_{XX}$ (0.25 μs)	$2^{16} \times 8/f_{XX}$ (16.40 ms)
$f_{XX}/16$	$16/f_{XX}$ (0.50 μs)	$2^{16} \times 16/f_{XX}$ (32.80 ms)
$f_{XX}/32$	$32/f_{XX}$ (1.00 μs)	$2^{16} \times 32/f_{XX}$ (65.50 ms)
$f_{XX}/64$	$64/f_{XX}$ (2.00 μs)	$2^{16} \times 64/f_{XX}$ (131 ms)
$f_{XX}/128$	$128/f_{XX}$ (4.00 μs)	$2^{16} \times 128/f_{XX}$ (262 ms)
$f_{XX}/256$	$256/f_{XX}$ (8.00 μs)	$2^{16} \times 256/f_{XX}$ (524 ms)
$f_{XX}/512$	$512/f_{XX}$ (16.00 μs)	$2^{16} \times 512/f_{XX}$ (1.05 s)
$f_{XX}/1024$	$1,024/f_{XX}$ (32.00 μs)	$2^{16} \times 1,024/f_{XX}$ (2.10 s)
$f_{XX}/2048$	$2,048/f_{XX}$ (64.00 μs)	$2^{16} \times 2,048/f_{XX}$ (4.19 s)

10.8.3 PWM Output

(1) Basic operation of PWM output

In this mode, a PWM signal with the period in which timer register 2 (TM2) reaches a full count used as one cycle is output. The timer output (TO2) pulse width is determined by the value of compare register (CR20), and the timer output (TO3) pulse width is determined by the value of compare register (CR21). When this function is used, the CLR21 bit and CLR22 bit of capture/compare control register 2 (CRC2) and the CMD2 bit of timer control register 1 (TMC1) must be set to 0.

The pulse cycle and pulse width are as shown below.

(a) BW2 = 0

- PWM cycle = 256 × x/fxx
- PWM pulse width = CR2n × x/fxx **Note**; x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

  Note 0 cannot be set in the CR2n.

- $\text{Duty} = \frac{\text{PWM pulse width}}{\text{PWM}} = \frac{\text{CR2n}}{256}$

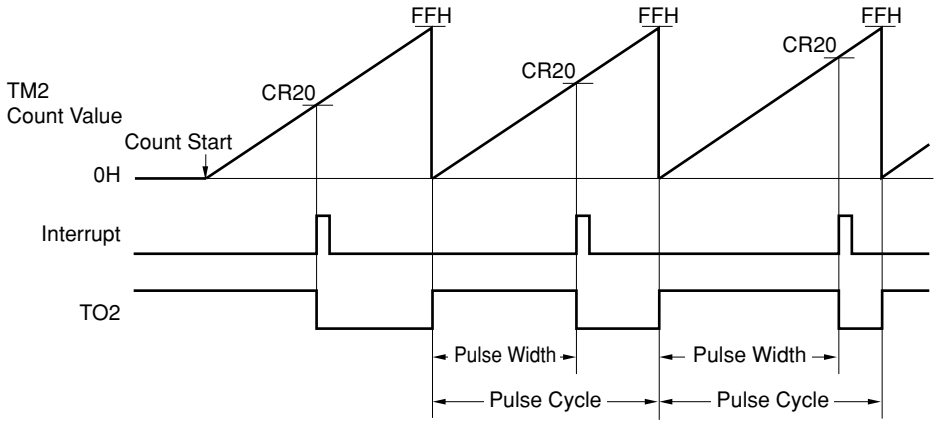
(b) BW2 = 1

- PWM cycle = 65,536 × x/fxx
- PWM pulse width = CR2n × x/fxx **Note**; x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

  Note 0 cannot be set in the CR2n.

- $\text{Duty} = \frac{\text{PWM pulse width}}{\text{PWM cycle}} = \frac{\text{CR2n}}{65{,}536}$

Figure 10-24 PWM Pulse Output (BW2 = 0)

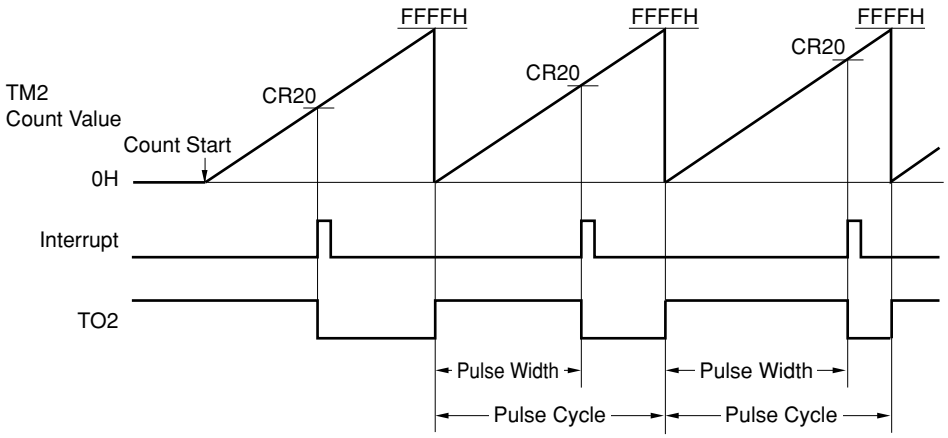


Remark ALV2 = 0

Table 10-8 TO2/TO3 PWM Cycle (f_{XX} = 32 MHz, BW2 = 0)

Count Clock	Minimum Pulse Width [μs]	PWM Cycle [ms]	PWM Frequency [Hz]
$f_{XX}/8$	0.25	0.06	15,625
$f_{XX}/16$	0.50	0.13	7,813
$f_{XX}/32$	1.00	0.26	3,906
$f_{XX}/64$	2.00	0.51	1,953
$f_{XX}/128$	4.00	1.02	977
$f_{XX}/256$	8.00	2.05	488
$f_{XX}/512$	16.00	4.10	244
$f_{XX}/1,024$	32.00	8.19	122
$f_{XX}/2,048$	64.00	16.40	61

Figure 10-25 PWM Pulse Output (BW2 = 1)



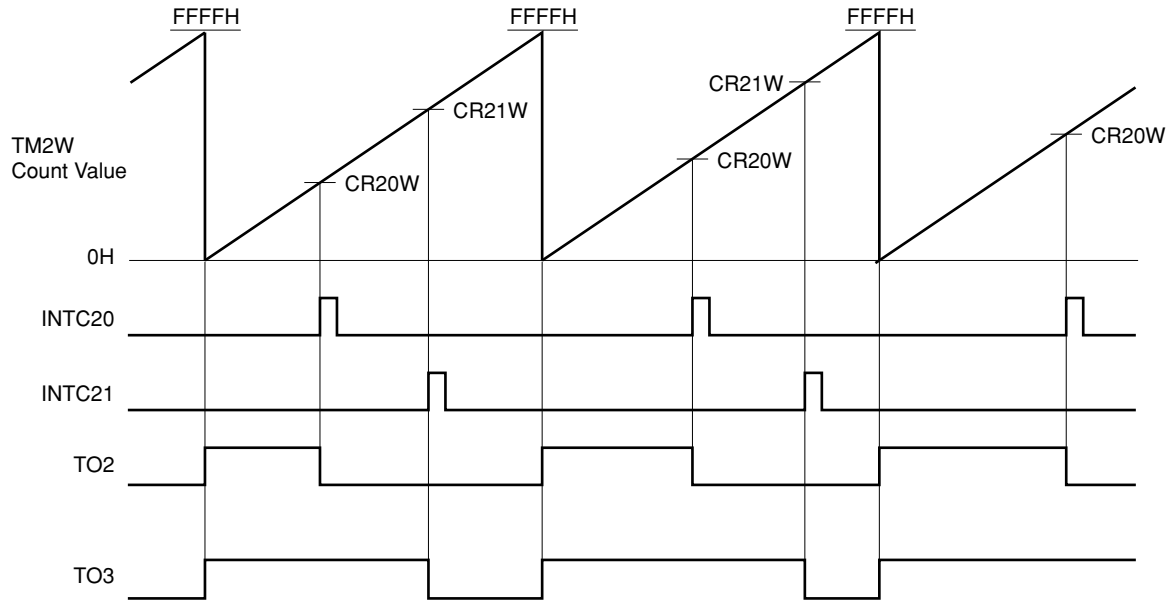
Remark ALV2 = 0

Table 10-9 TO2/TO3 PWM Cycle (f_{XX} = 32 MHz, BW2 = 1)

Count Clock	Minimum Pulse Width [μs]	PWM Cycle [s]	PWM Frequency [Hz]
$f_{XX}/8$	0.25	0.02	61.0
$f_{XX}/16$	0.50	0.03	30.5
$f_{XX}/32$	1.00	0.07	15.3
$f_{XX}/64$	2.00	0.13	7.6
$f_{XX}/128$	4.00	0.26	3.8
$f_{XX}/256$	8.00	0.52	1.9
$f_{XX}/512$	16.00	1.05	1.0
$f_{XX}/1,024$	32.00	2.10	0.5
$f_{XX}/2,048$	64.00	4.19	0.2

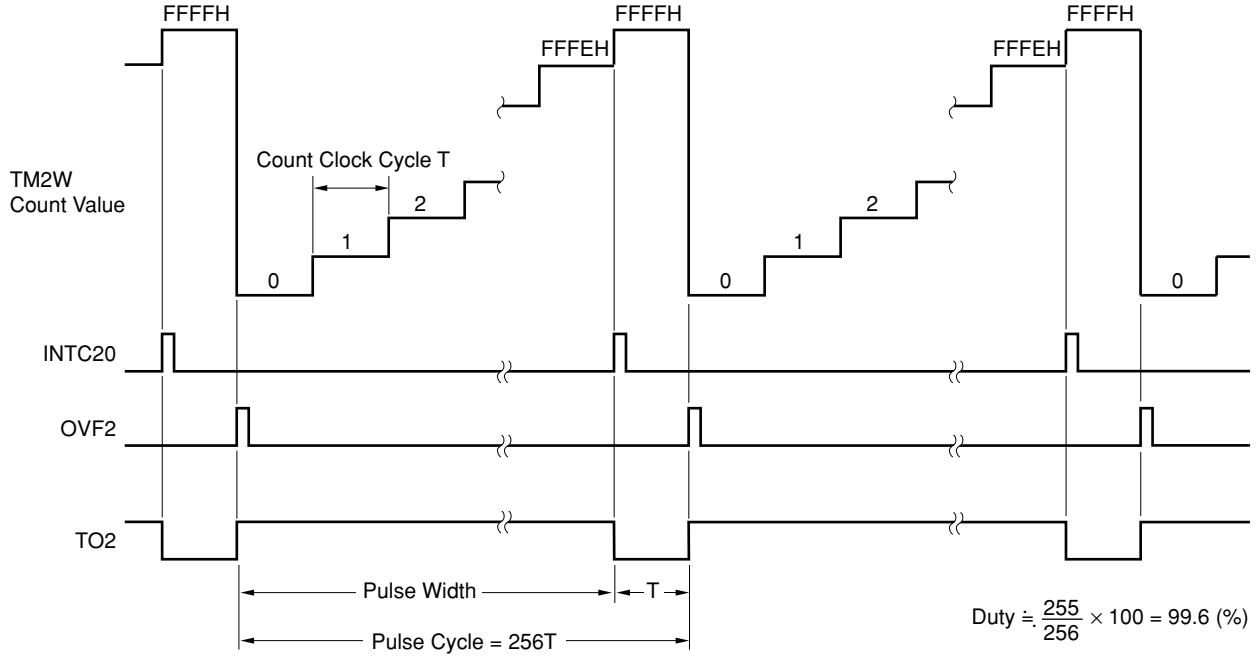
Figure 10-26 shows an example of 2-channel PWM output, and Figure 10-27 shows the case where FFFFH is set in the CR20W.

Figure 10-26 Example of PWM Output Using TM2W



Remark ALV2 = 0, ALV3 = 0

Figure 10-27 Example of PWM Output When CR20W = FFFFH



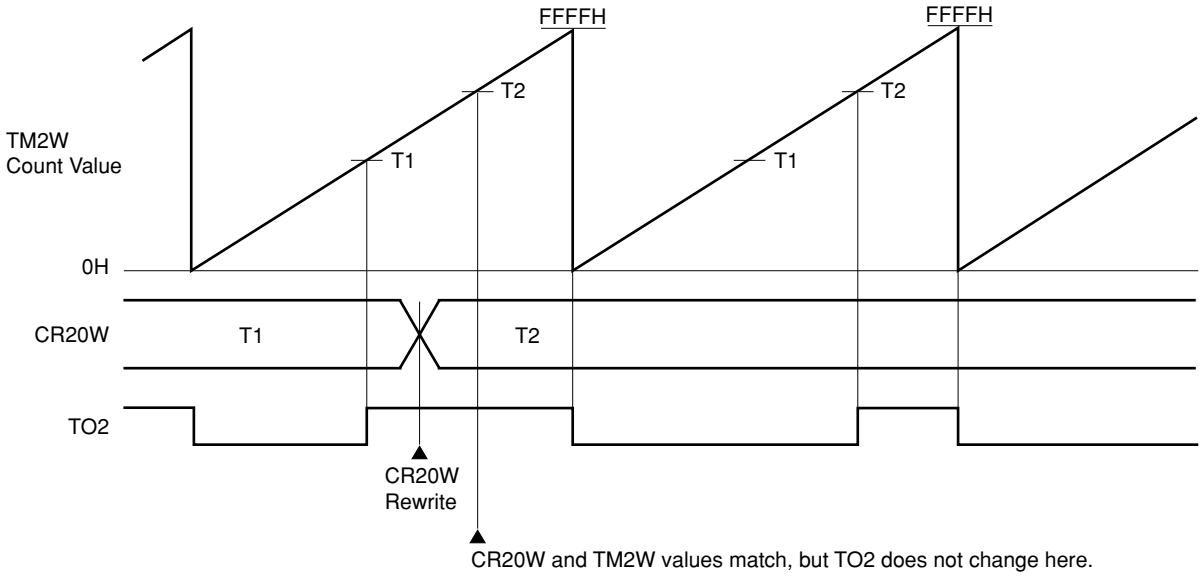
Remarks 1. ALV2 = 0

2. T = x/fxx (x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048)

(2) Rewriting compare registers (CR20, CR21)

The output level of the timer output (TOn + 2: n + 2 = 2, 3) is not inverted even if the CR2n (n = 0, 1) value matches the timer register 2 (TM2) value more than once during one PWM output cycle.

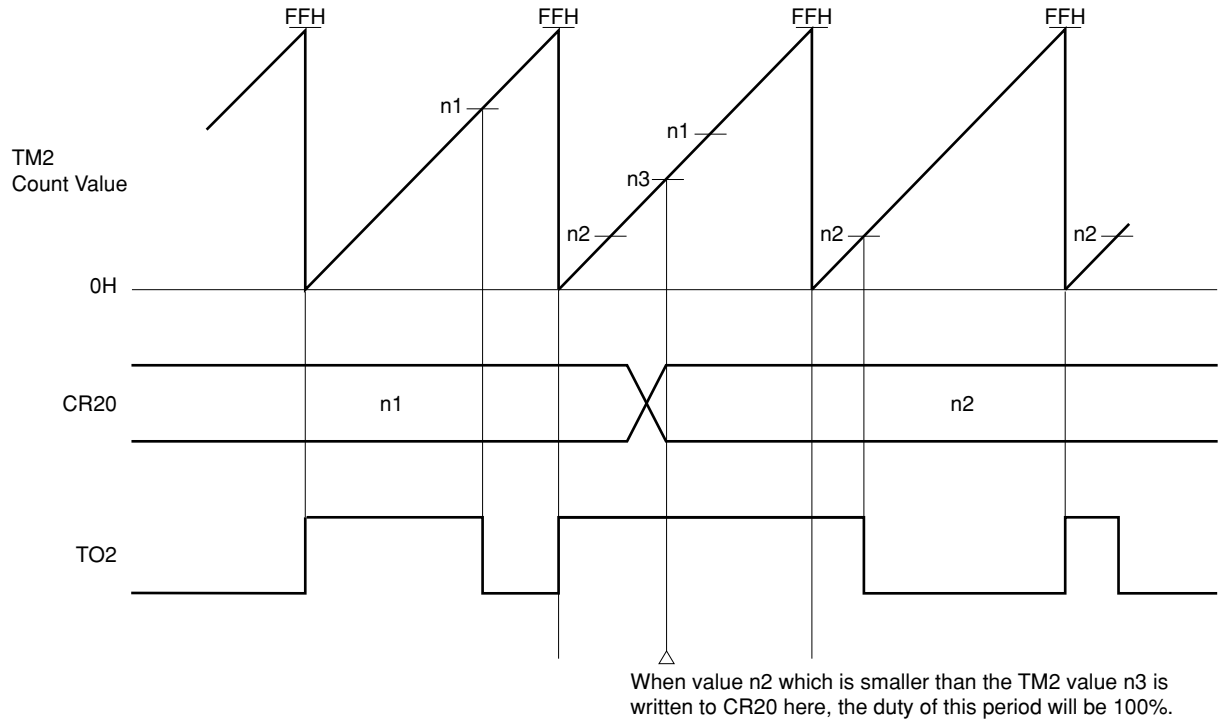
Figure 10-28 Example of Compare Register (CR20W) Rewrite



Remark ALV2 = 1

If a value smaller than that of the TM2 is set as the CR2n value, a 100% duty PWM signal will be output. CR2n rewriting should be performed by the interrupt due to a match between TM2 and the CR2n on which the rewrite is performed.

Figure 10-29 Example of 100% Duty With PWM Output

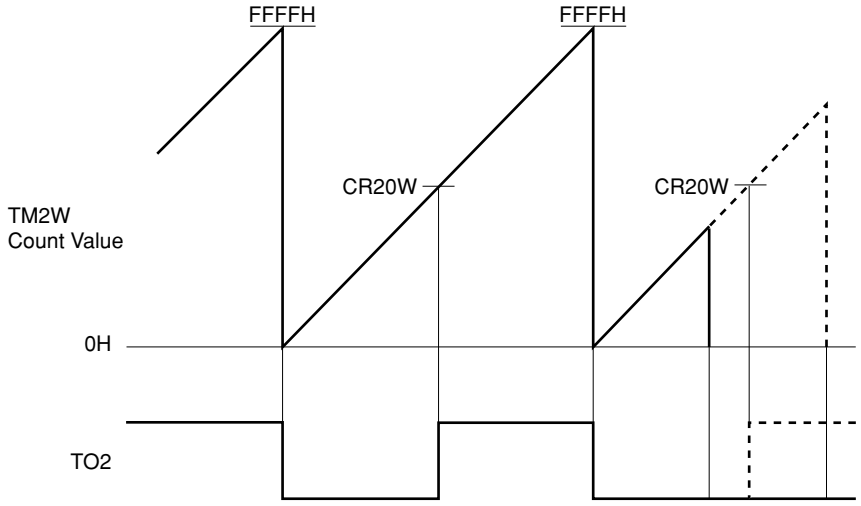


Remark ALV2 = 0

(3) Stopping PWM output

If timer/counter 2 is stopped by clearing (to 0) the CE2 bit of the timer control register 1 (TMC1) during PWM signal output, the active level is output.

Figure 10-30 When Timer/Counter 2 is Stopped During PWM Signal Output



Remark ALV2 = 1

Caution The output level of the TOn (n = 2/3) pin when timer output is disabled (ENTOn = 0: n = 2/3) is the inverse of the value set in ALVn (n = 2/3) bits. Caution is therefore required as the active level is output when timer output is disabled when the PWM output function has been selected.

10.8.4 PPG Output

(1) Basic operation of PPG output

This function outputs a square-wave with the time determined by compare register CR21 value as one cycle, and the time determined by compare register CR20 value as the pulse width. The PWM output PWM cycle is made variable. This signal can only be output from timer output (TO2).

When this function is used, it is necessary to set the CLR21 bit of capture/compare control register 2 (CRC2) to 1 and the CLR22 bit to 0, and to set the CMD2 bit of timer control register 1 (TMC1) to 0.

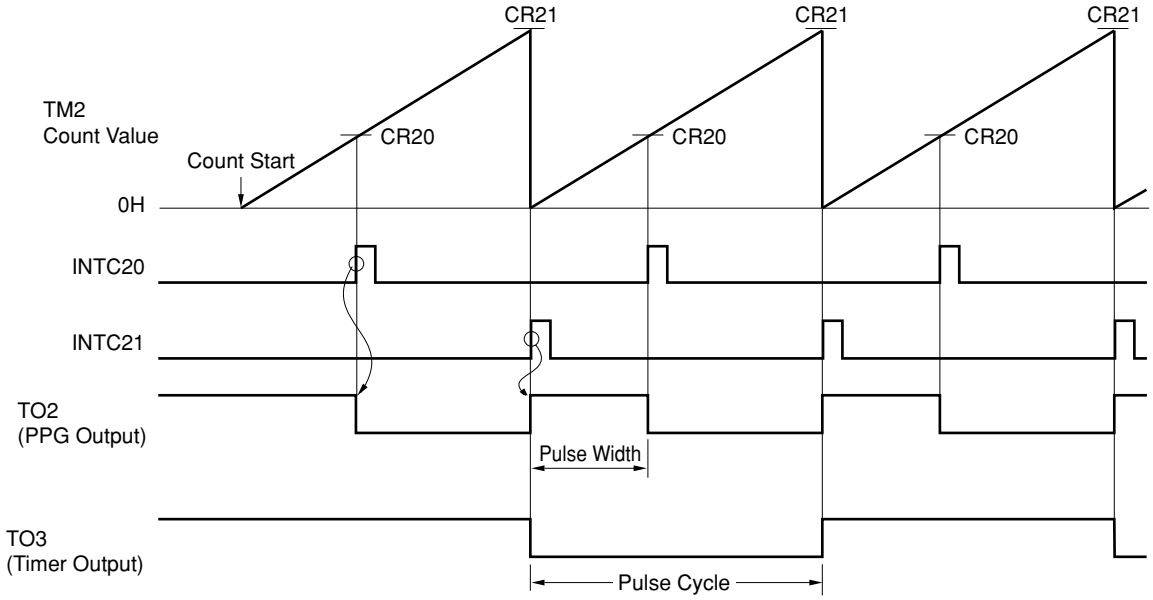
The pulse cycle and pulse width are as shown below.

- PPG cycle = (CR21 + 1) × x/fxx; x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048
- PPG pulse width = CR20 × x/fxx where 1 ≤ CR20 ≤ CR21 **Note**
- $\text{Duty} = \frac{\text{PPG pulse width}}{\text{PPG cycle}} = \frac{\text{CR20}}{\text{CR21} + 1}$ **Note**

Note Neither the CR20 nor the CR21 can be cleared to “0”.

Figure 10-31 shows an example of PPG output using timer register 2 (TM2), Figure 10-32 shows an example of the case where CR20 = CR21.

Figure 10-31 Example of PPG Output Using TM2

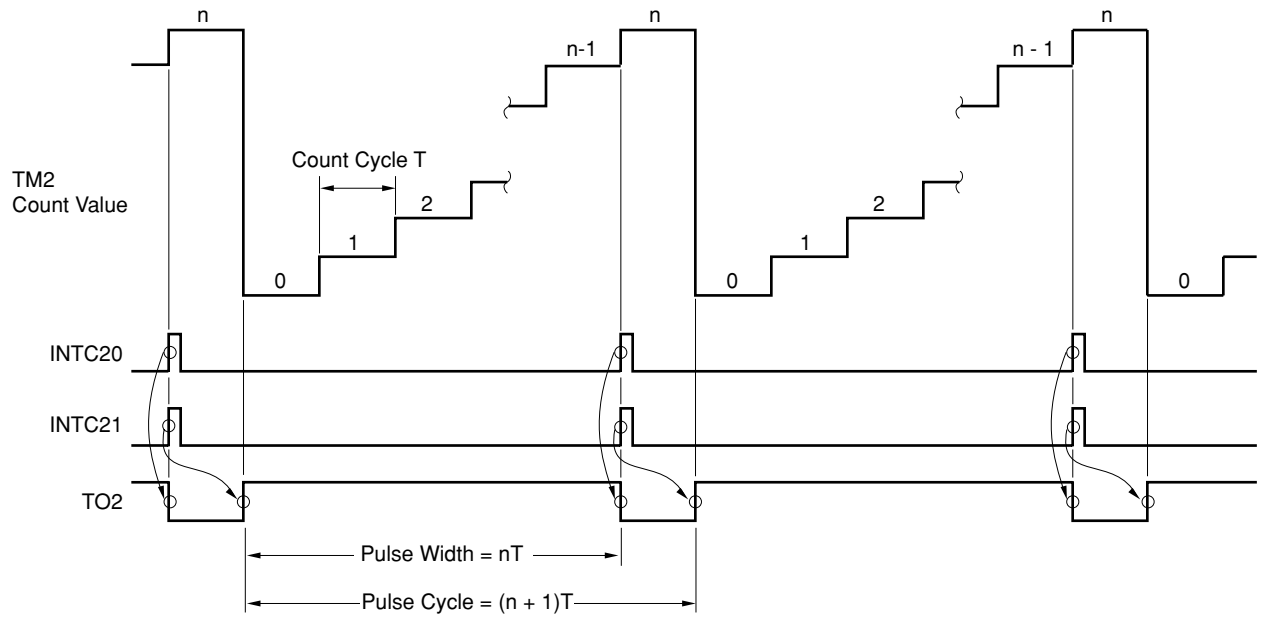


Remark ALV2 = 0, ALV3 = 0

Table 10-10 TO2 PPG Output (f_{XX} = 32 MHz)

Count Clock	Minimum Pulse Width	PPG Cycle	PPG Frequency
$f_{XX}/8$	0.25 μs	0.50 μs to 16.40 ms	2,000 kHz to 61.0 Hz
$f_{XX}/16$	0.50 μs	1.00 μs to 32.80 ms	1,000 kHz to 30.5 Hz
$f_{XX}/32$	1.00 μs	2.00 μs to 65.50 ms	500 kHz to 15.3 Hz
$f_{XX}/64$	2.00 μs	4.00 μs to 131 ms	250 kHz to 7.6 Hz
$f_{XX}/128$	4.00 μs	8.00 μs to 262 ms	125 kHz to 3.8 Hz
$f_{XX}/256$	8.00 μs	16.00 μs to 524 ms	62.5 kHz to 1.9 Hz
$f_{XX}/512$	16.00 μs	32.00 μs to 1.05 s	31.3 kHz to 1.0 Hz
$f_{XX}/1,024$	32.00 μs	64.00 μs to 2.10 s	15.6 kHz to 0.5 Hz
$f_{XX}/2,048$	64.00 μs	128.00 μs to 4.19 s	7.8 kHz to 0.2 Hz

Figure 10-32 Example of PPG Output When CR20 = CR21



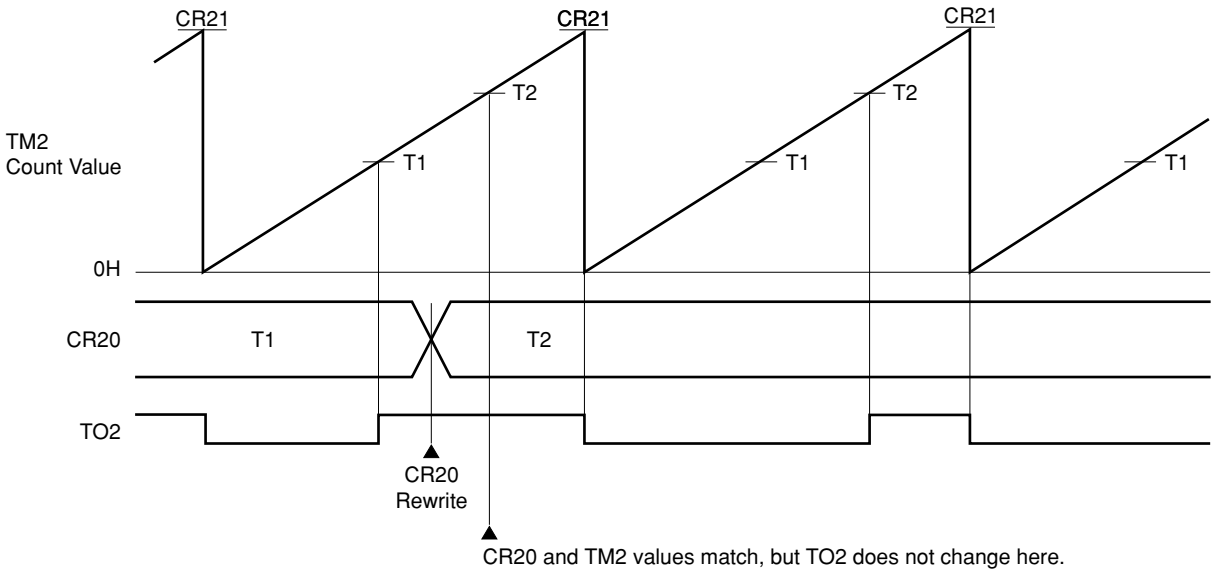
Remark ALV2 = 0

T = x/fxx (x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048)

(2) Rewriting compare register (CR20)

The output level of the timer output (TO2) is not changed even if the CR20 value matches the timer register 2 (TM2) value more than once during one PPG output cycle.

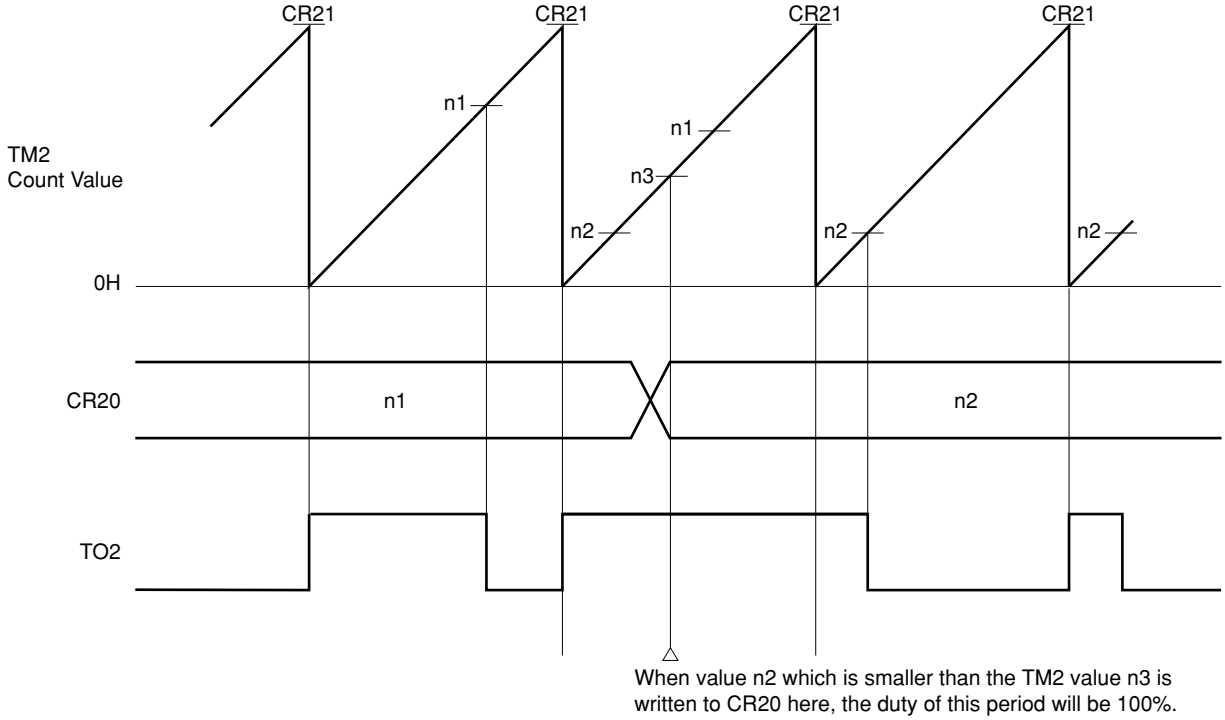
Figure 10-33 Example of Compare Register Rewrite



Remark ALV2 = 1

If a value equal to or less than the TM2 value is written to CR20 before the CR20 and TM2 match, the duty of that PPG cycle will be 100%. CR20 rewriting should be performed by the interrupt due to a match between TM2 and CR20.

Figure 10-34 Example of 100% Duty With PPG Output



Remark ALV2 = 0

Caution If the PPG cycle is extremely short as compared with the time required to acknowledge an interrupt, the value of CR20 cannot be rewritten by interrupt processing that is performed on match between TM2 and CR20. Use another method (for example, to poll the interrupt request flags by software with all the interrupts masked).

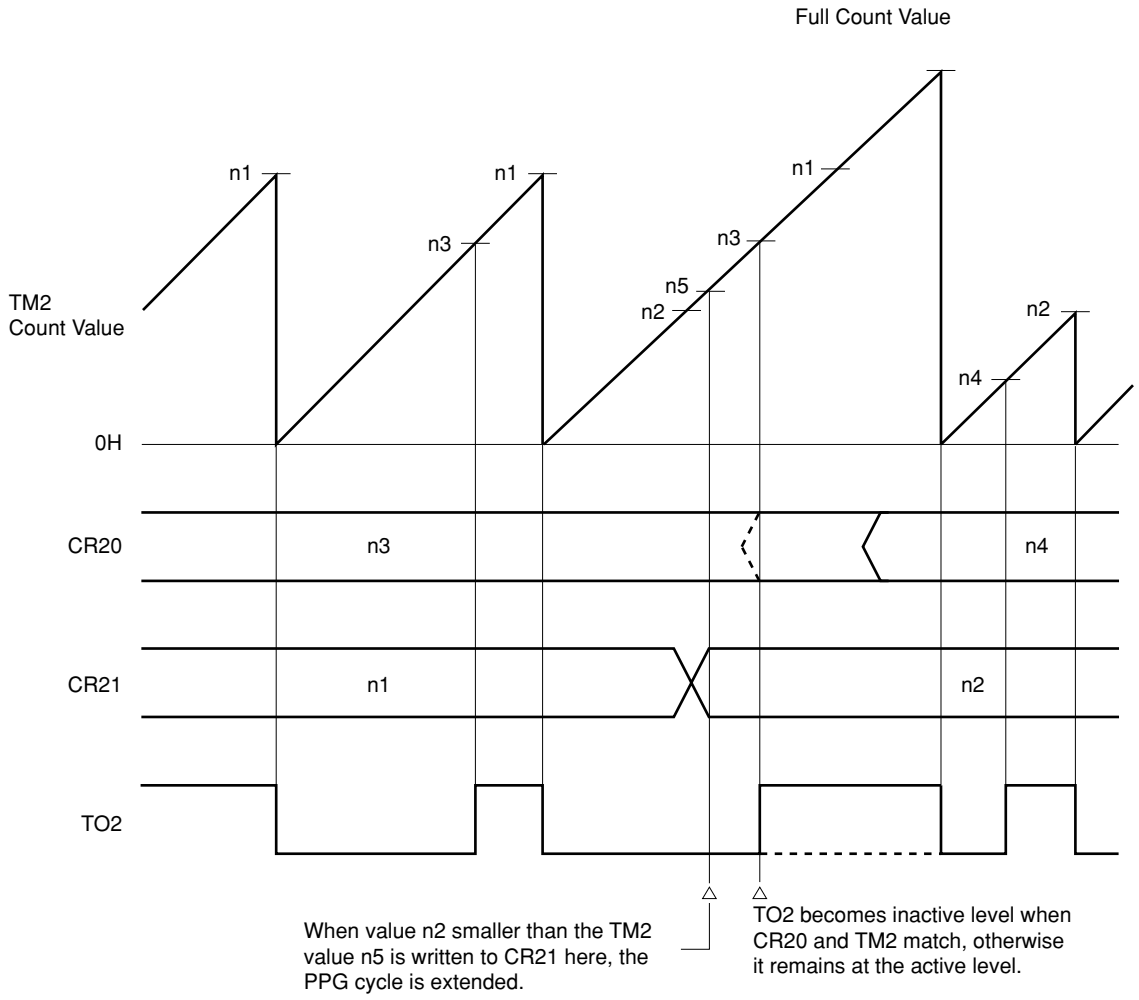
(3) Rewriting compare register (CR21)

If the current value of the CR21 is changed to a smaller value, and the CR21 value is made smaller than the register 2 (TM2) value, the PPG cycle at that time will be extended to the time equivalent to a full-count by TM2. If CR21 is rewritten after the compare register (CR20) and TM2 match, the output level at this time will be the inactive level until TM2 overflows and becomes 0, and will then return to normal PPG output.

If CR21 is rewritten before CR20 and TM2 match, the active level will be output until CR20 and TM2 match. If CR20 and TM2 match before TM2 overflows and becomes 0, the inactive level is output at that point. When TM2 overflows and becomes 0, the active level will be output, and normal PPG output will be restored.

CR21 rewriting should be performed by the interrupt due to a match between TM2 and CR21, etc.

Figure 10-35 Example of Extended PPG Output Cycle



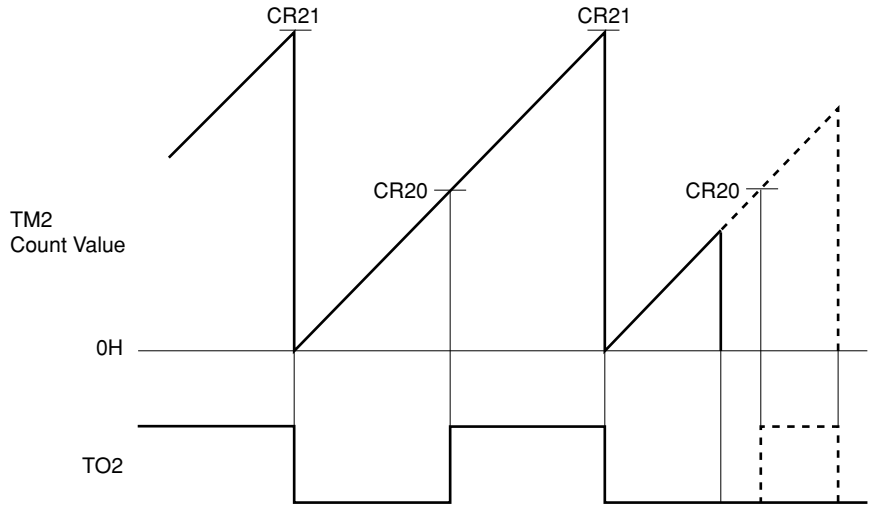
Remark ALV2 = 1

Caution If the PPG cycle is extremely short as compared with the time required to acknowledge an interrupt, the value of CR2n cannot be rewritten by interrupt processing that is performed on match between timer register 2 (TM2) and compare register (CR2n: n = 0, 1). Use another method (for example, to poll the interrupt request flags by software with all the interrupts masked).

(4) Stopping PPG output

If timer/counter 2 is stopped by clearing (to 0) the CE2 bit of the timer control register 1 (TMC1) during PPG signal output, the active level is output irrespective of the output level at the time timer/counter 2 was stopped.

Figure 10-36 When Timer/Counter 2 is Stopped During PPG Signal Output



Caution The output level of the TOn (n = 2/3) pin when timer output is disabled (ENTOn = 0: n = 2/3) is the inverse value of the value set in ALVn (n = 2/3) bits. Caution is therefore required as the active level is output when timer output is disabled when the PPG output function has been selected.

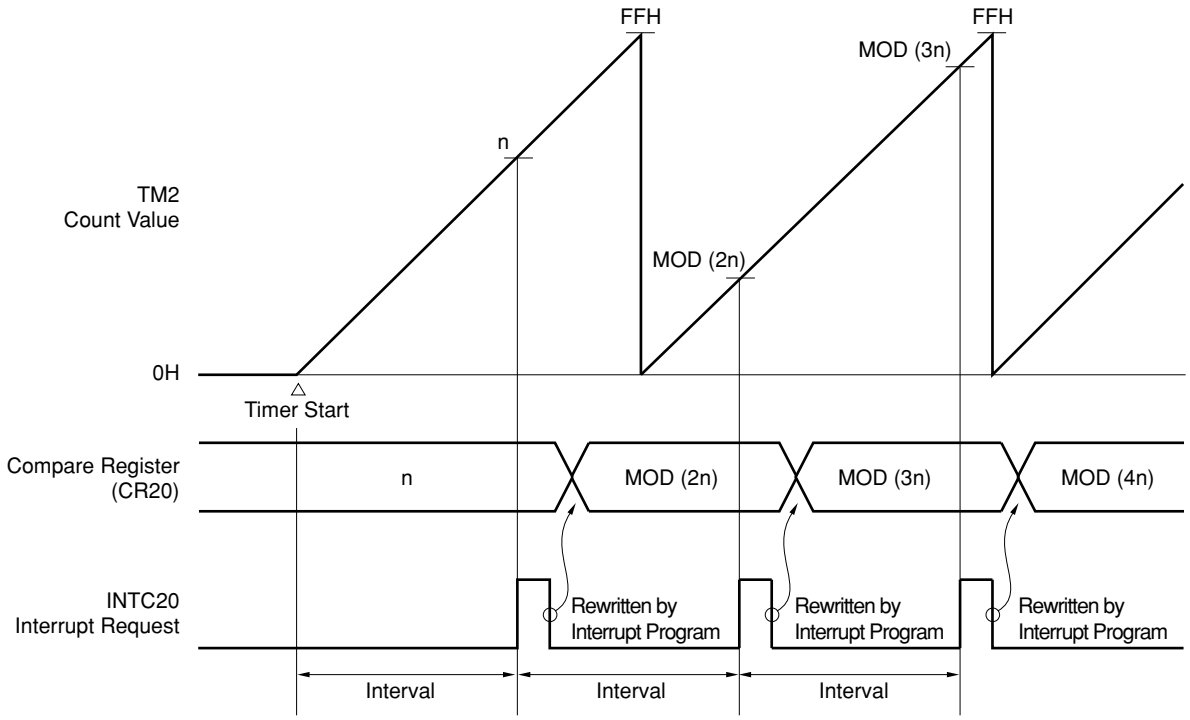
10.9 EXAMPLES OF USE

10.9.1 Operation as Interval Timer (1)

When timer register 2 (TM2) is made free-running and a fixed value is added to the compare register (CR2n: n = 0, 1) in the interrupt service routine, TM2 operates as an interval timer with the added fixed value as the cycle (see **Figure 10-37**).

The control register settings are shown in Figure 10-38, the setting procedure in Figure 10-39, and the processing in the interrupt service routine in Figure 10-40.

Figure 10-37 Interval Timer Operation (1) Timing

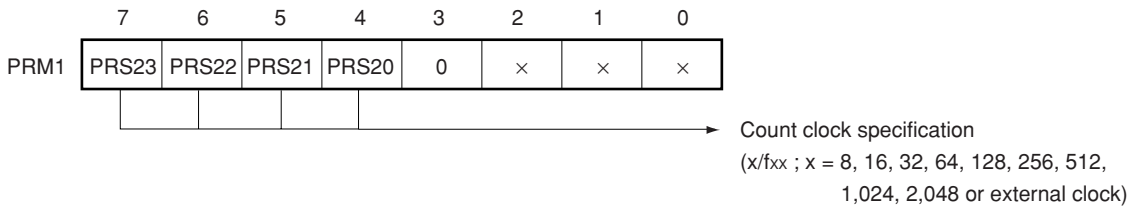


Remark Interval = n × x/fxx

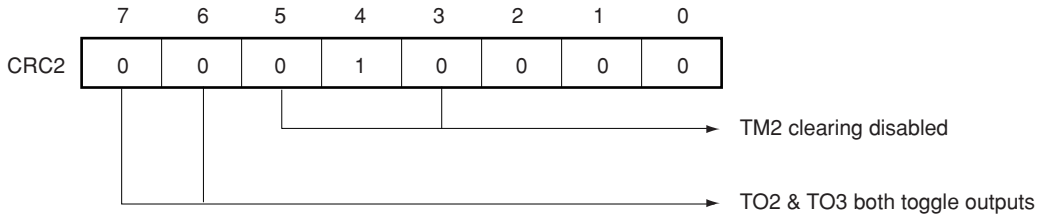
1 ≤ n ≤ FFH, x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

Figure 10-38 Control Register Settings for Interval Timer Operation (1)

(a) Prescaler mode register 1 (PRM1)



(b) Capture/compare control register 2 (CRC2)



(c) Timer control register 1 (TMC1)

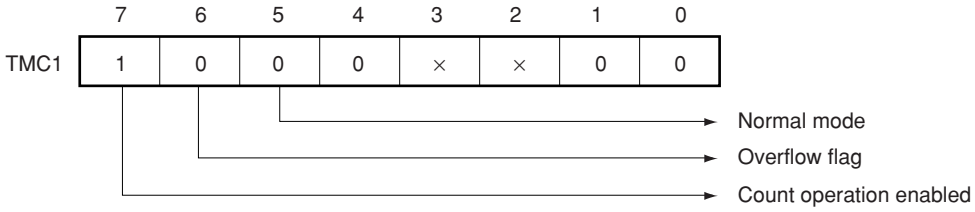


Figure 10-39 Interval Timer Operation (1) Setting Procedure

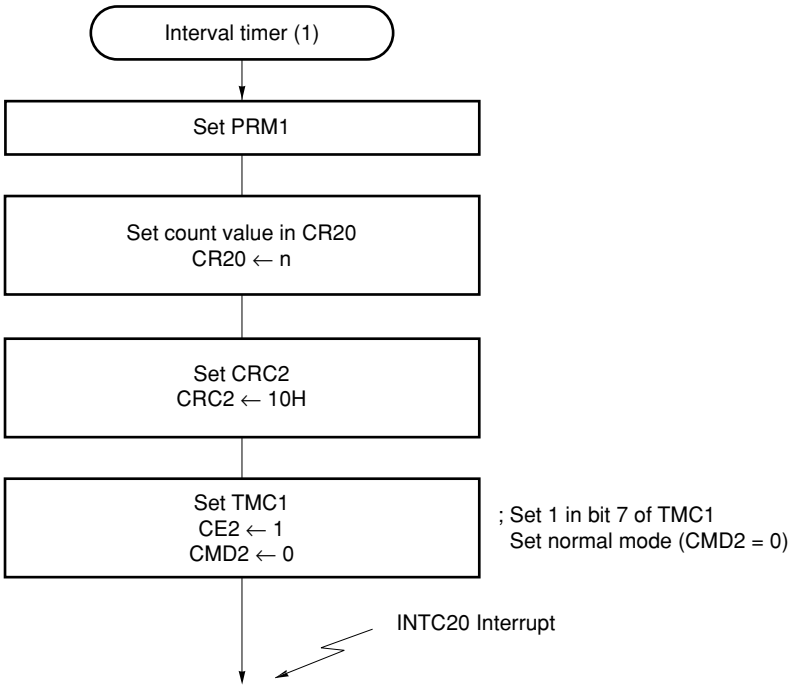
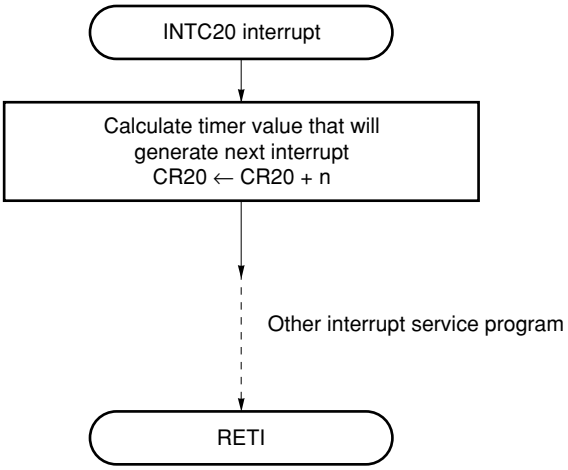


Figure 10-40 Interval Timer Operation (1) Interrupt Request Servicing

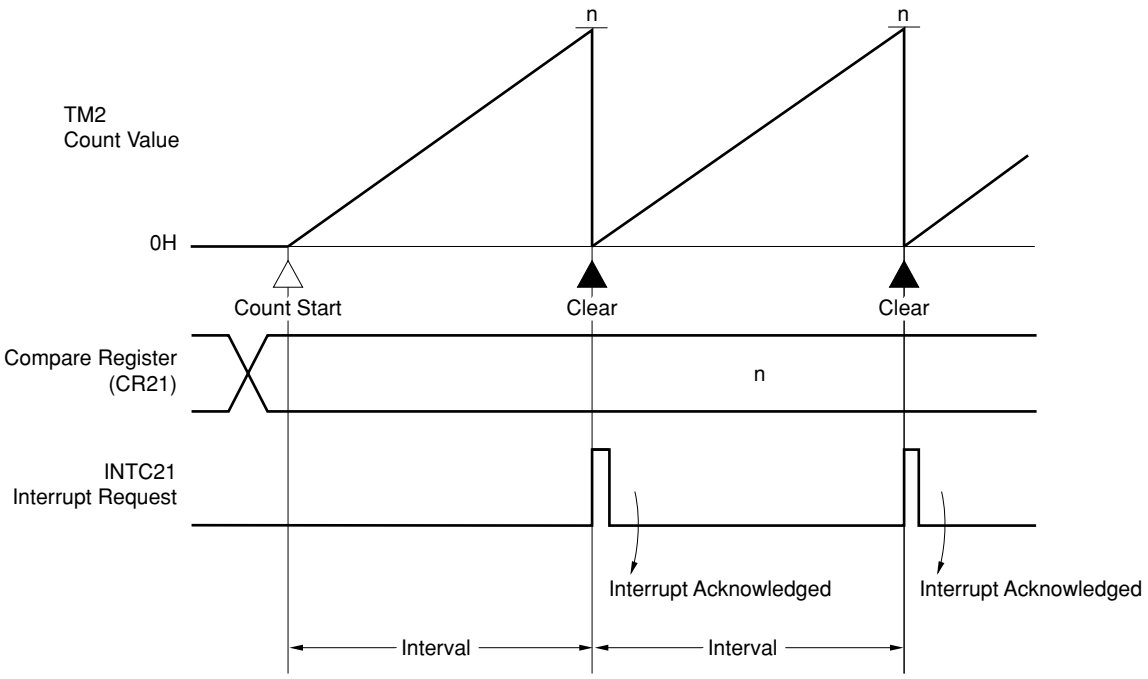


10.9.2 Operation as Interval Timer (2)

TM2 operates as an interval timer that generates interrupts repeatedly with the preset count time as the interval (see **Figure 10-41**).

The control register settings are shown in Figure 10-42, and the setting procedure in Figure 10-43.

Figure 10-41 Interval Timer Operation (2) Timing

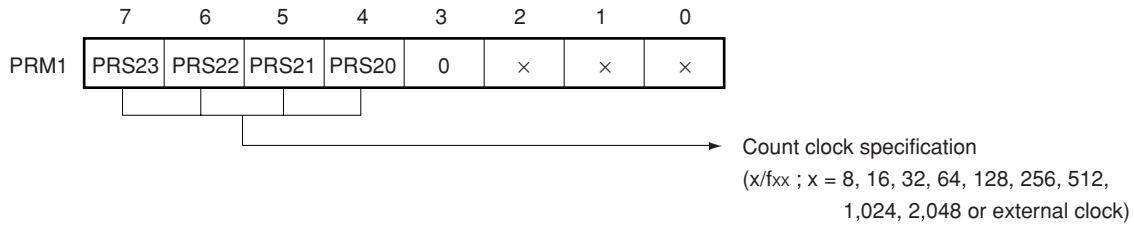


Remark Interval = (n + 1) × x/fxx

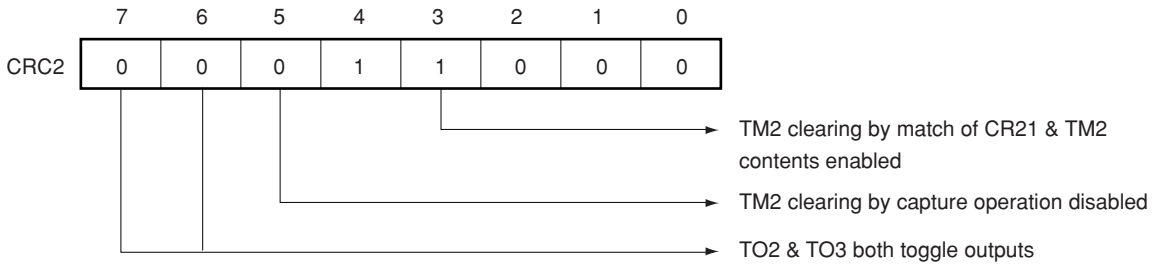
0 ≤ n ≤ FFH, x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

Figure 10-42 Control Register Settings for Interval Timer Operation (2)

(a) Prescaler mode register 1 (PRM1)



(b) Capture/compare control register 2 (CRC2)



(c) Timer control register 1 (TMC1)

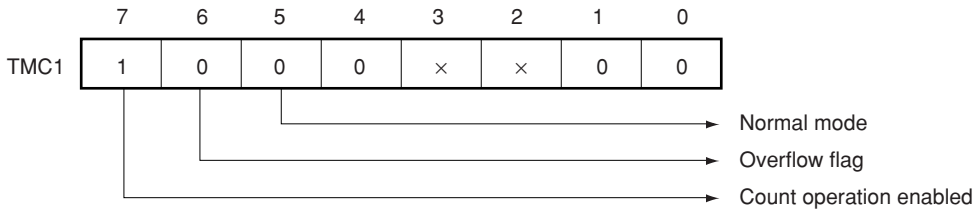
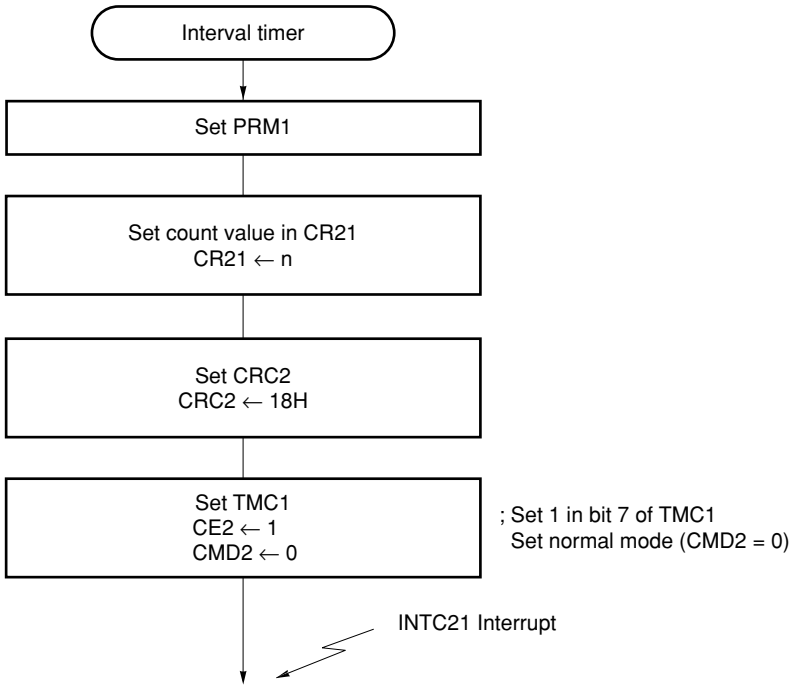


Figure 10-43 Interval Timer Operation (2) Setting Procedure



10.9.3 Pulse Width Measurement Operation

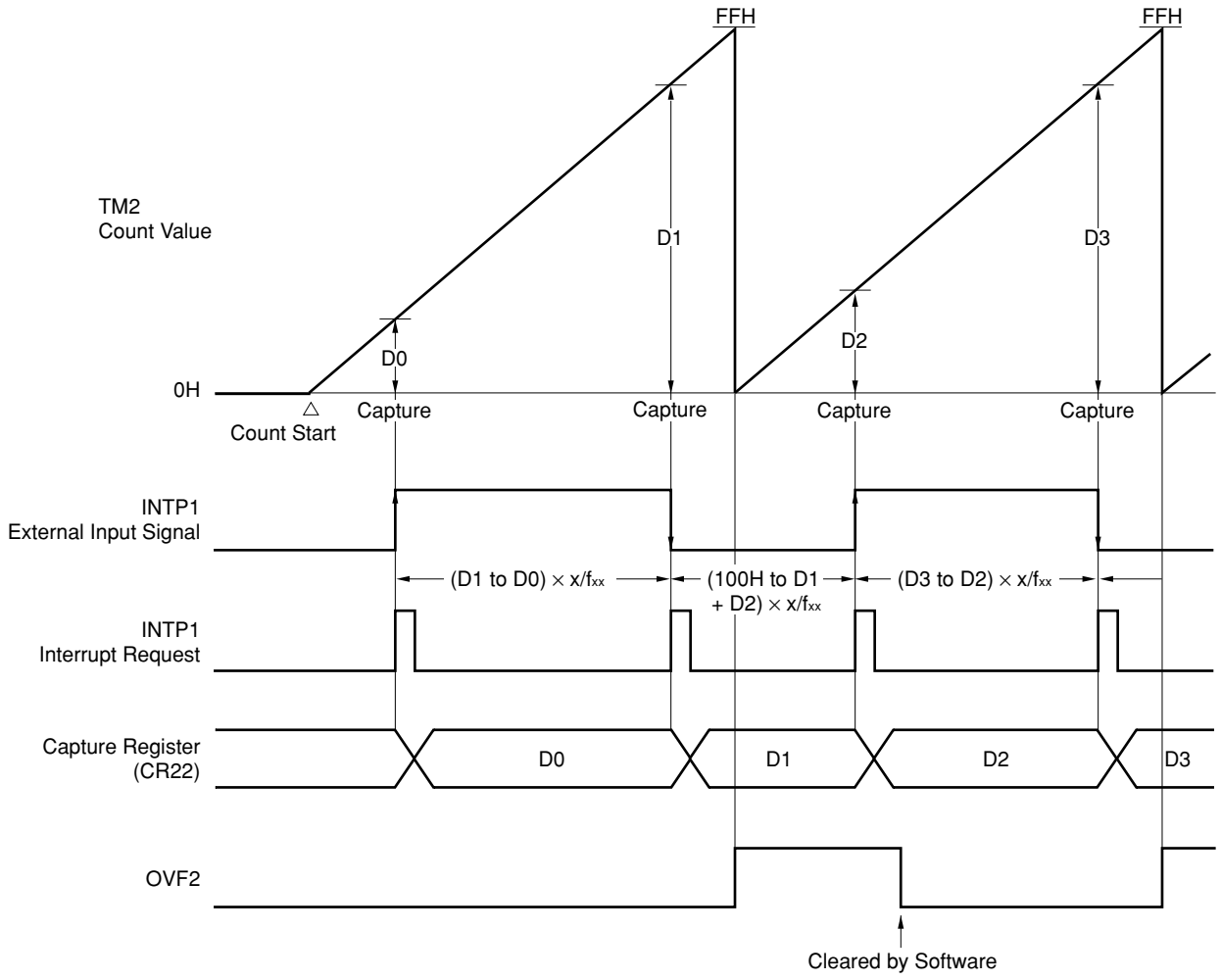
In pulse width measurement, the high-level or low-level width of external pulses input to the external interrupt request input pin (INTP1) pin are measured.

Both the high-level and low-level widths of pulses input to the INTP1 pin must be at least 3 system clocks (0.19 μs: f_{CLK} = 16 MHz); if shorter than this, the valid edge will not be detected and a capture operation will not be performed.

As shown in Figure 10-44, the timer register 2 (TM2) value being counted is fetched into the capture register (CR22) in synchronization with a valid edge (specified as both rising and falling edges) in the INTP1 pin input, and held there. The pulse width is obtained from the product of the difference value between the TM2 count value (D_n) fetched into and held in the CR22 on detection of the nth valid edge and the count value (D_{n-1}) fetched and held on detection of n - 1th valid edge, and the number of n - 1th count clocks (x/f_{xx}; x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048).

The control register settings are shown in Figure 10-45, and the setting procedure in Figure 10-46.

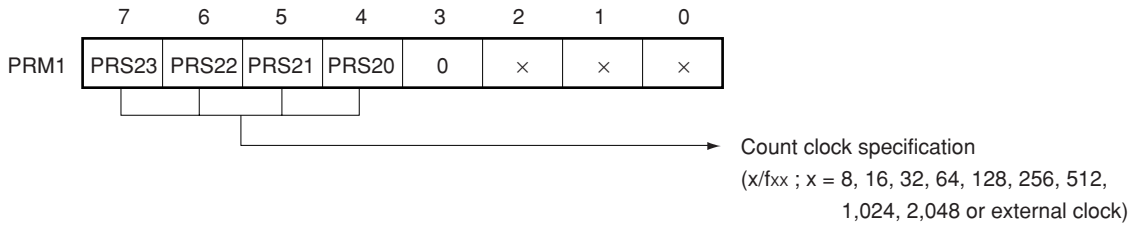
Figure 10-44 Pulse Width Measurement Timing



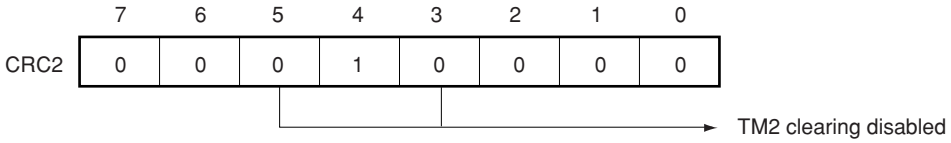
Remark Dn: TM2 count value (n = 0, 1, 2, ...)
x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

Figure 10-45 Control Register Settings for Pulse Width Measurement

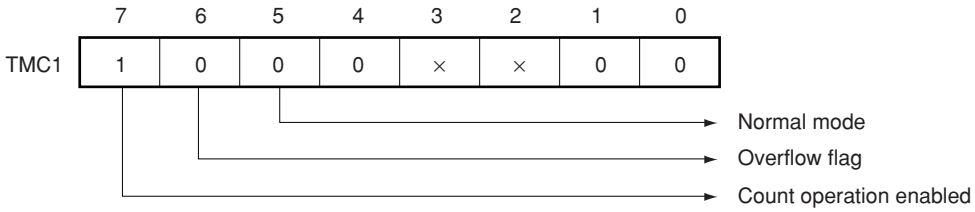
(a) Prescaler mode register 1 (PRM1)



(b) Capture/compare control register 2 (CRC2)



(c) Timer control register 1 (TMC1)



(d) External interrupt mode register 0 (INTM0)

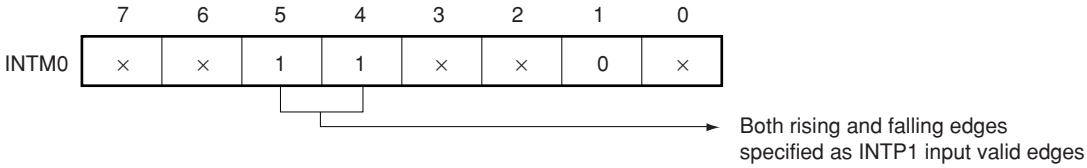


Figure 10-46 Pulse Width Measurement Setting Procedure

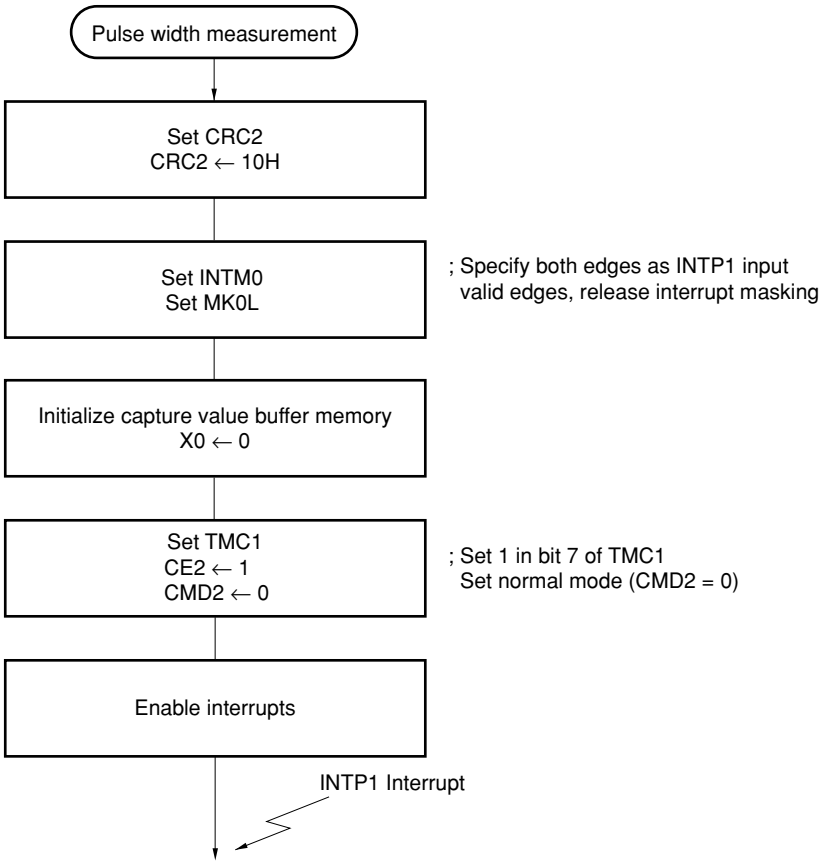
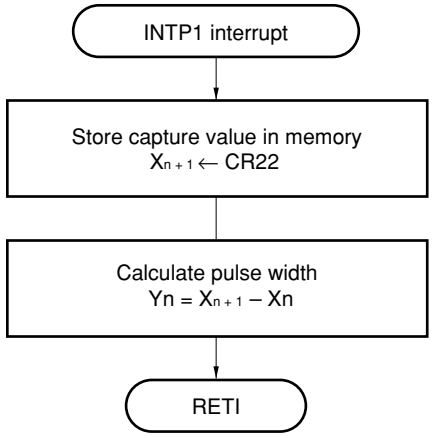


Figure 10-47 Interrupt Request Servicing that Calculates Pulse Width



10.9.4 Operation as PWM Output

In PWM output, pulses with the duty ratio determined by the value set in the compare register (CR2n: n = 0, 1) are output (see Figure 10-48).

This PWM output duty ratio can be varied in the range 1/256 to 255/256 in 1/256 units.

The control register settings are shown in Figure 10-49, the setting procedure in Figure 10-50, and the procedure for varying the duty in Figure 10-51.

Figure 10-48 Example of Timer/Counter 2 PWM Signal Output

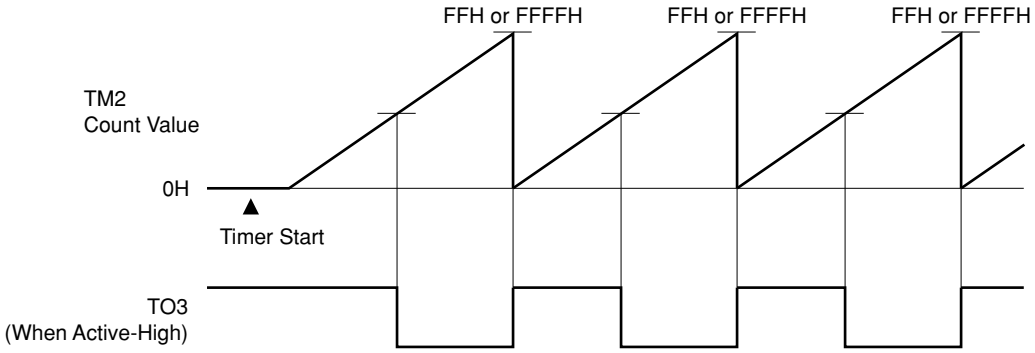
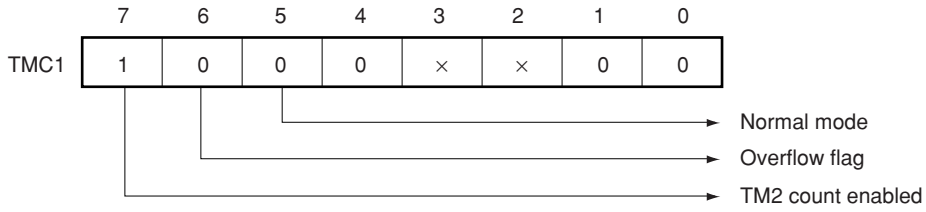
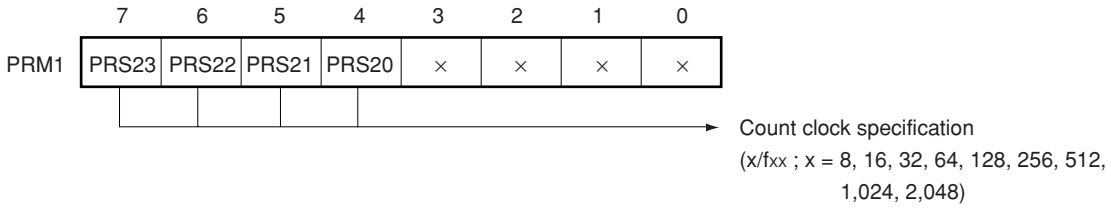


Figure 10-49 Control Register Settings for PWM Output Operation

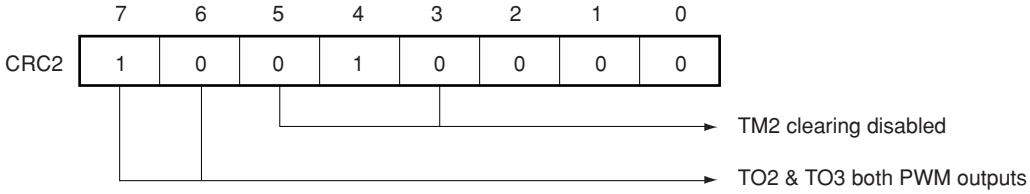
(a) Timer control register 1 (TMC1)



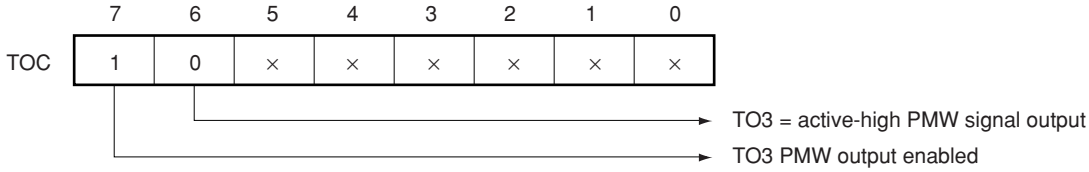
(b) Prescaler mode register 1 (PRM1)



(c) Capture/compare control register 2 (CRC2)



(d) Timer output control register (TOC)



(e) Port 3 mode control register (PMC3)

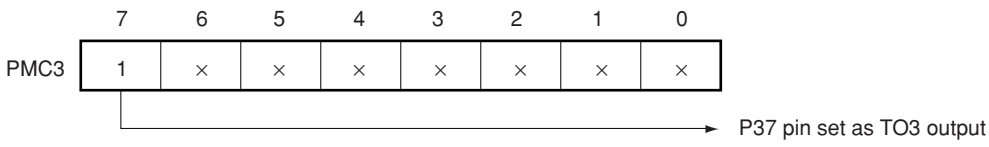


Figure 10-50 PWM Output Setting Procedure

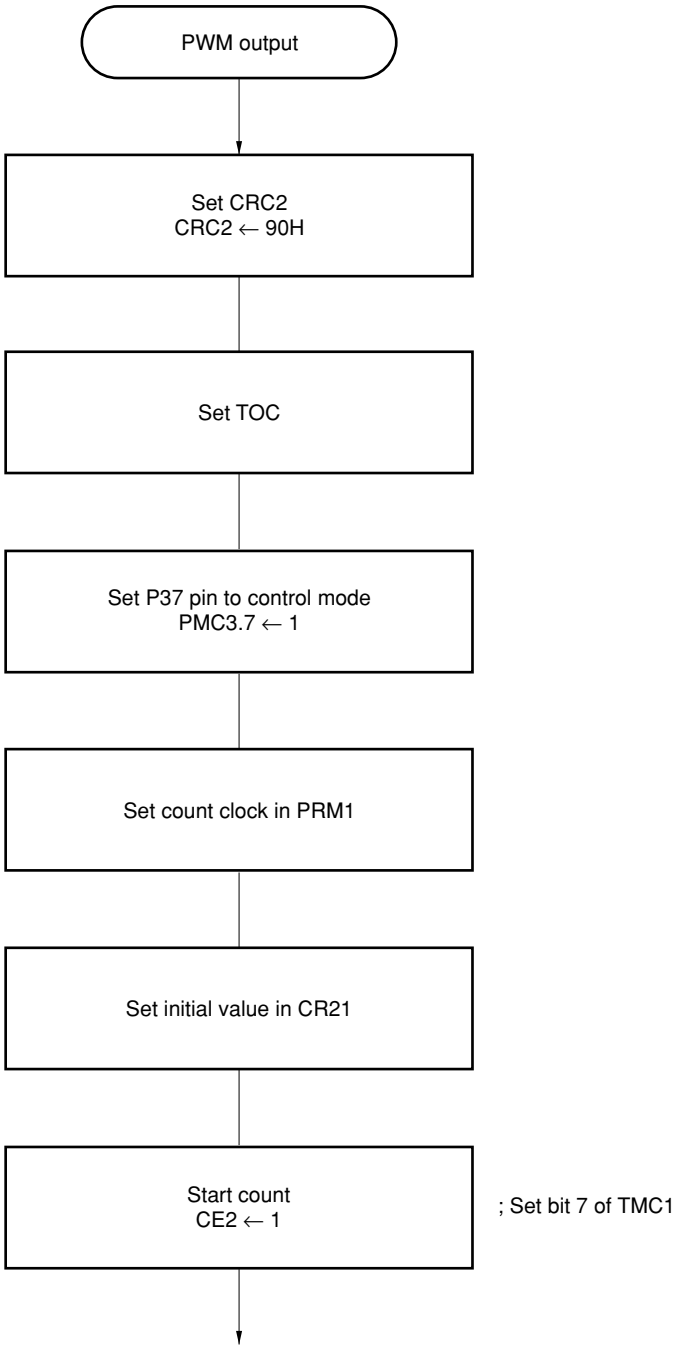
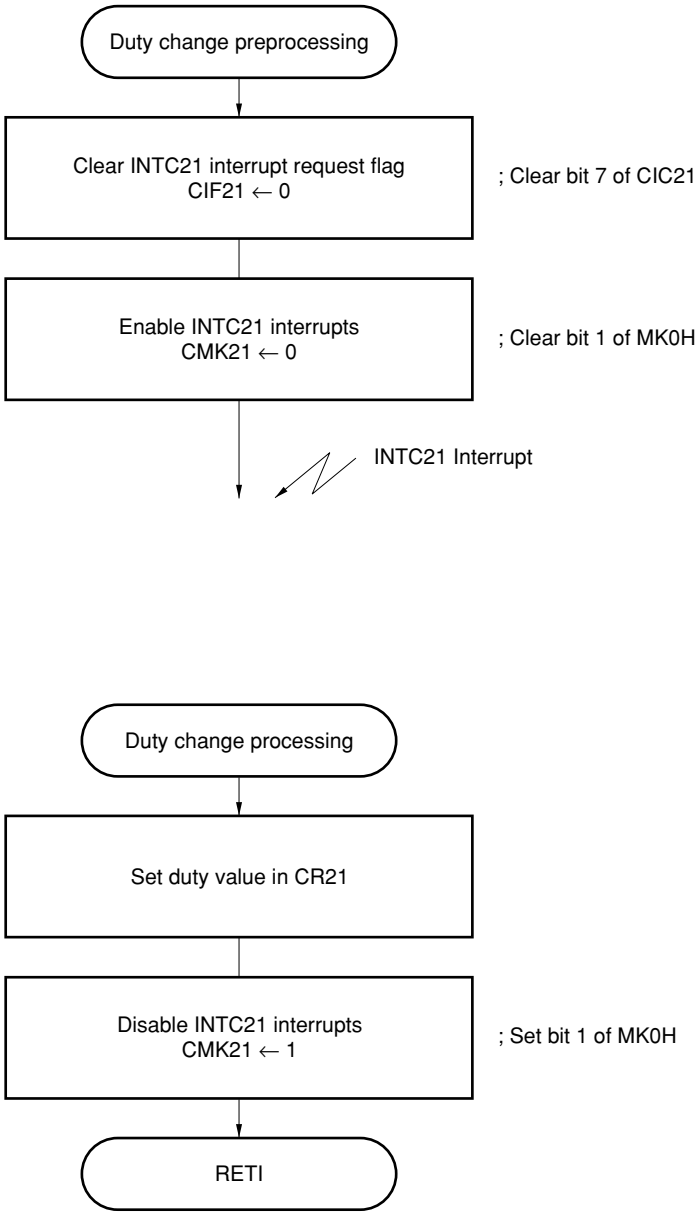


Figure 10-51 Changing PWM Output Duty



10.9.5 Operation as PPG Output

In PPG output, pulses with the cycle and duty ratio determined by the value set in the compare register (CR2n: n = 0, 1) are output (see **Figure 10-52**).

The control register settings are shown in Figure 10-53, the setting procedure in Figure 10-54, and the procedure for varying the duty in Figure 10-55.

Figure 10-52 Example of Timer/Counter 2 PPG Signal Output

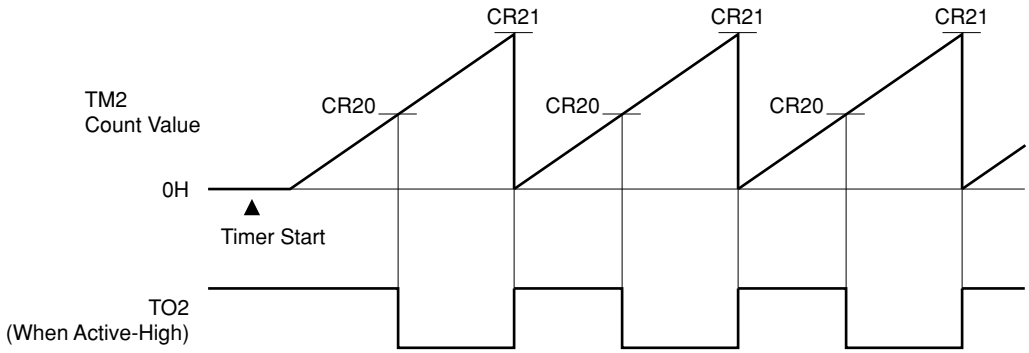
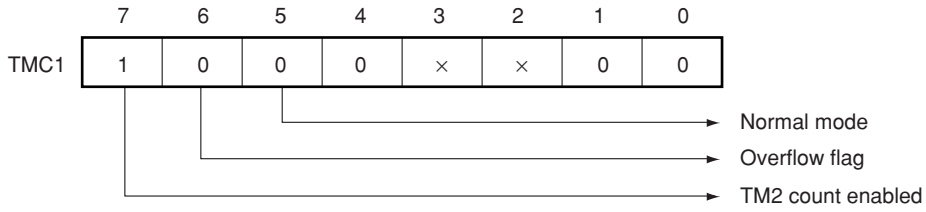
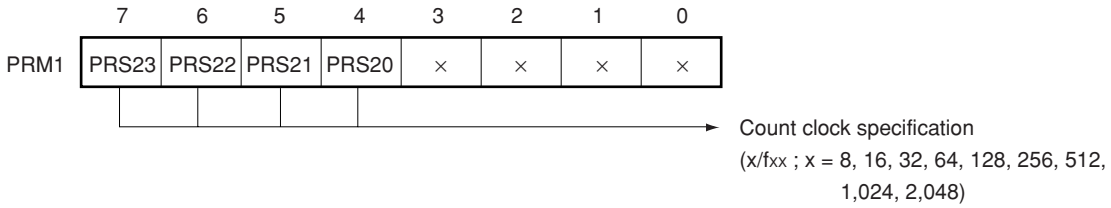


Figure 10-53 Control Register Settings for PPG Output Operation

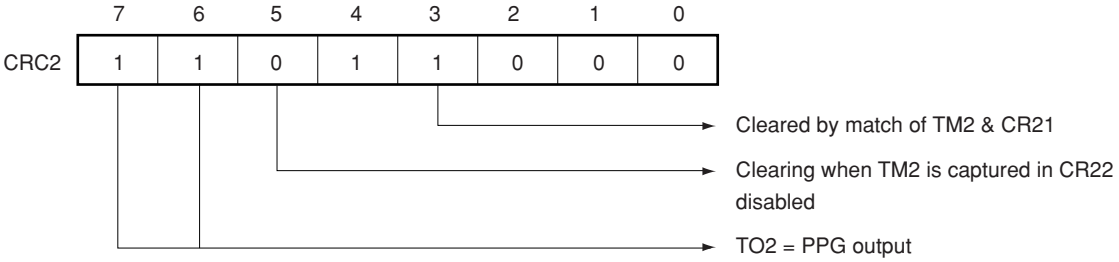
(a) Timer control register 1 (TMC1)



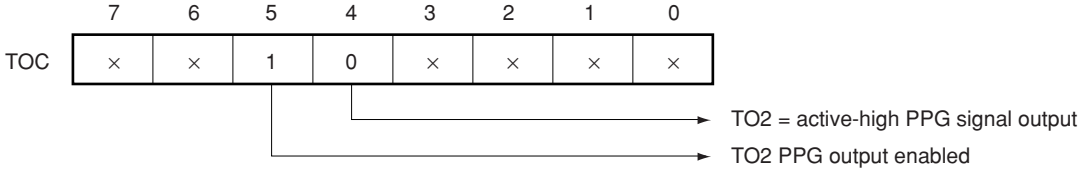
(b) Prescaler mode register 1 (PRM1)



(c) Capture/compare control register 2 (CRC2)



(d) Timer output control register (TOC)



(e) Port 3 mode control register (PMC3)

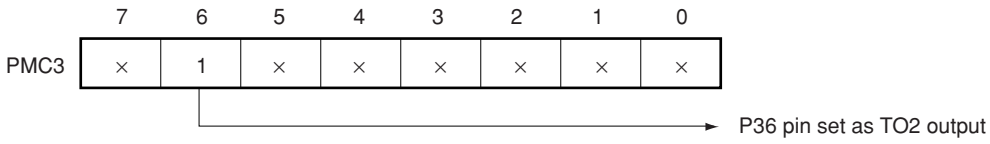


Figure 10-54 PPG Output Setting Procedure

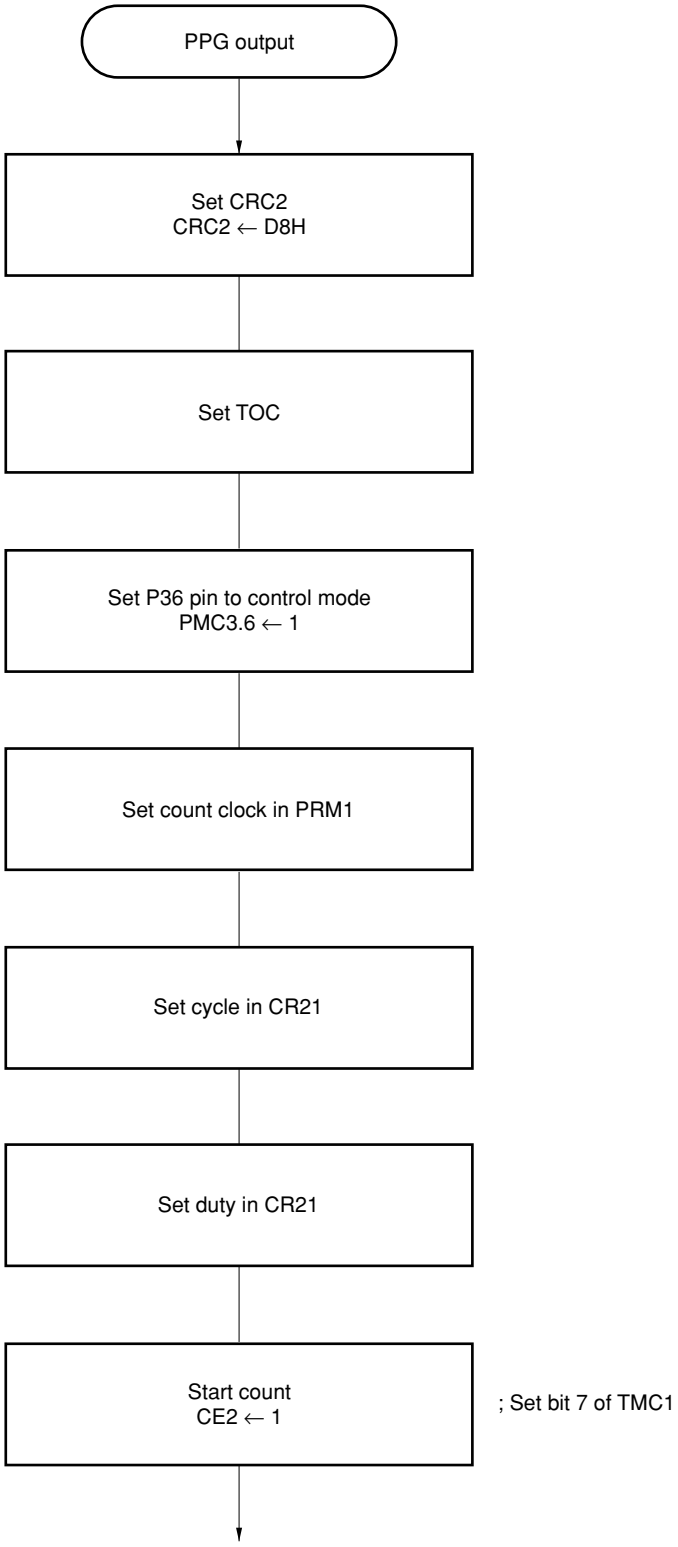
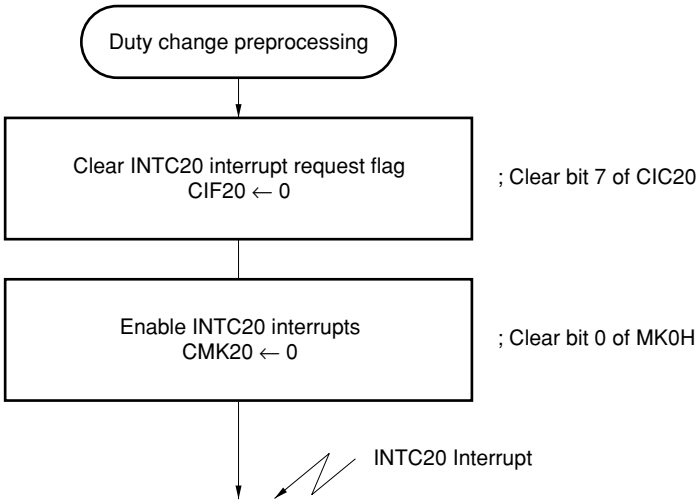
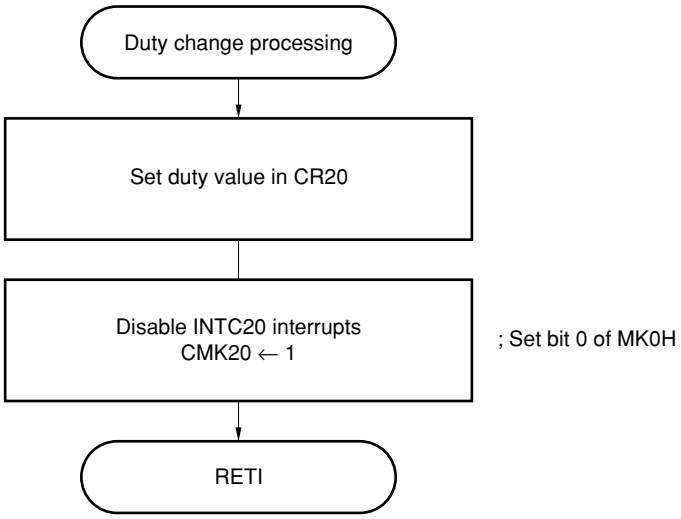


Figure 10-55 Changing PPG Output Duty



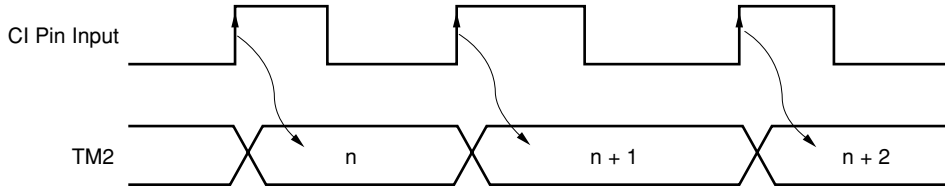


10.9.6 Operation as External Event Counter

An external event counter counts clock pulses (CI pin input pulses) input from off-chip.

As shown in Figure 10-56, the value of timer register 2 (TM2) is incremented in synchronization with a CI pin input valid edge (specified as rising edge only).

Figure 10-56 External Event Counter Operation (Single Edge)

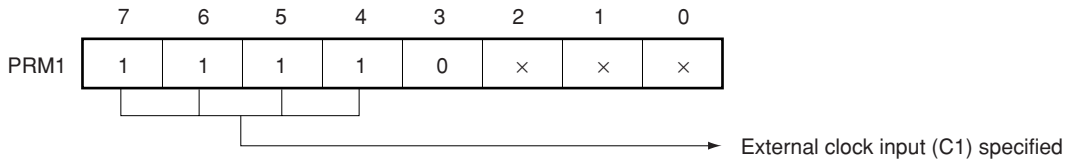


Remark The TM2 value is one less than the number of input clock pulses.

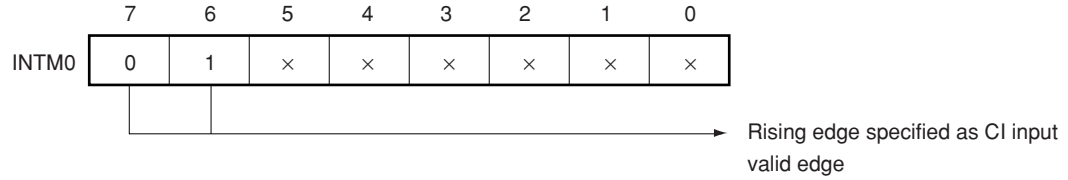
The control register settings when TM2 operates as an external event counter are shown in Figure 10-57, and the setting procedure in Figure 10-58.

Figure 10-57 Control Register Settings for External Event Counter Operation

(a) Prescaler mode register 1 (PRM1)



(b) External interrupt mode register 0 (INTM0)



(c) Timer control register 1 (TMC1)

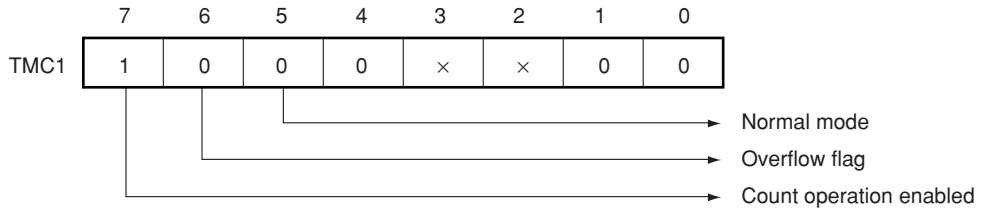
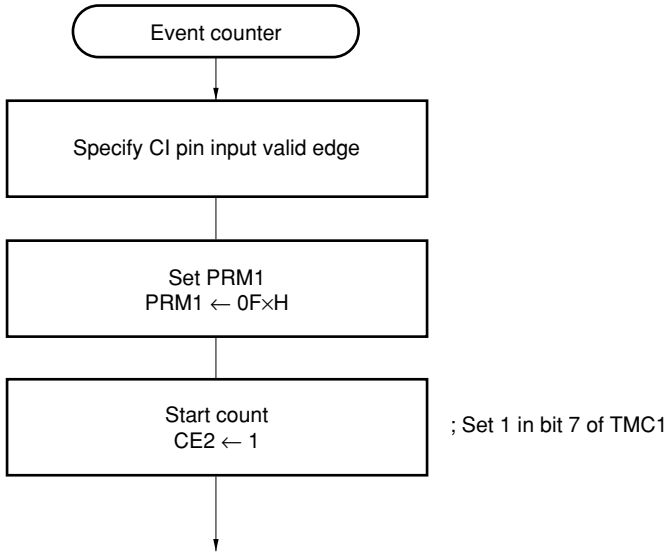


Figure 10-58 External Event Counter Operation Setting Procedure



10.9.7 Operation as One-Shot Timer

After timer register 2 (TM2) is started, it operates as a one-shot pulse that generates a single interrupt after the preset count time (see **Figure 10-59**).

The second and subsequent one-shot timer operations can be started by clearing the OVF2 bit of timer control register 1 (TMC1).

The control register settings are shown in Figure 10-60, the setting procedure in Figure 10-61, and the procedure for starting the one-shot timer from the second time onward in Figure 10-62.

Figure 10-59 One-Shot Timer Operation

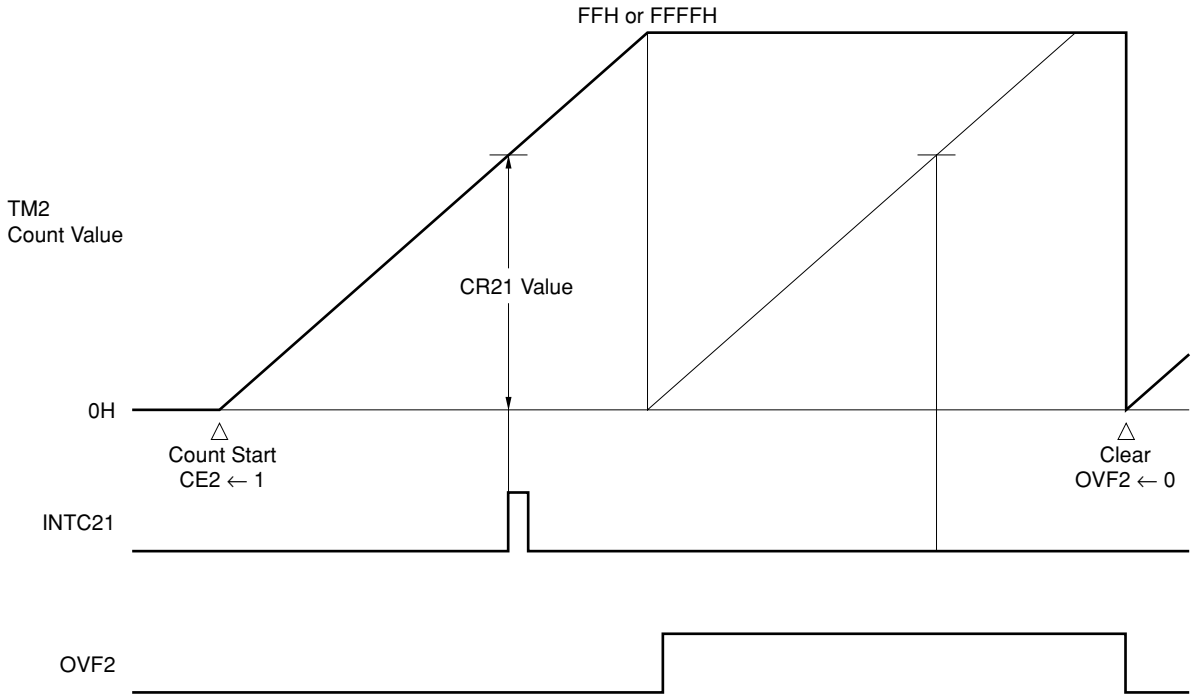
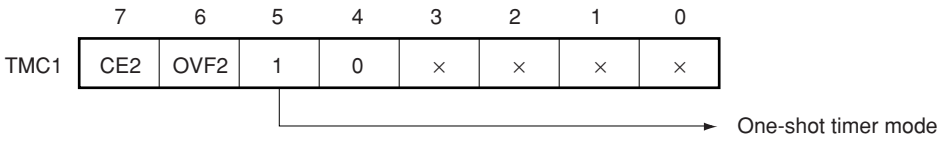
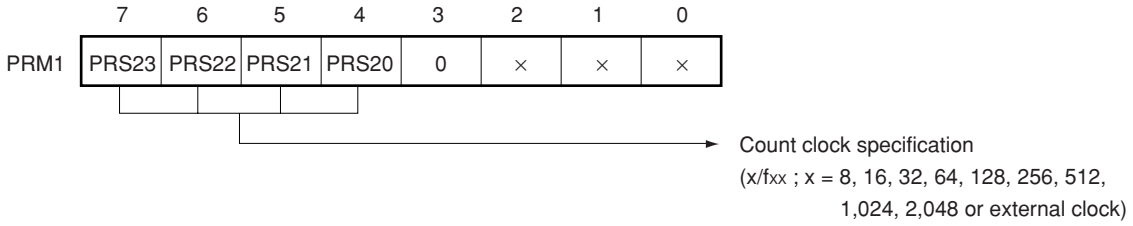


Figure 10-60 Control Register Settings for One-Shot Timer Operation

(a) Timer control register 1 (TMC1)



(b) Prescaler mode register 1 (PRM1)



(c) Capture/compare control register 2 (CRC2)

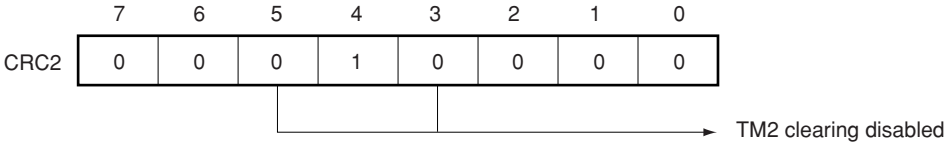


Figure 10-61 One-Shot Timer Operation Setting Procedure

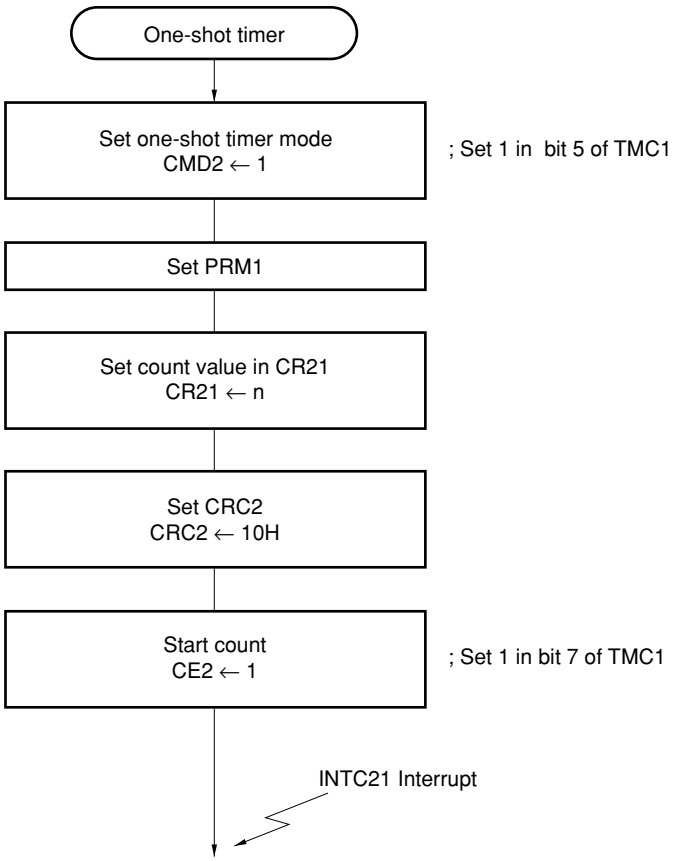
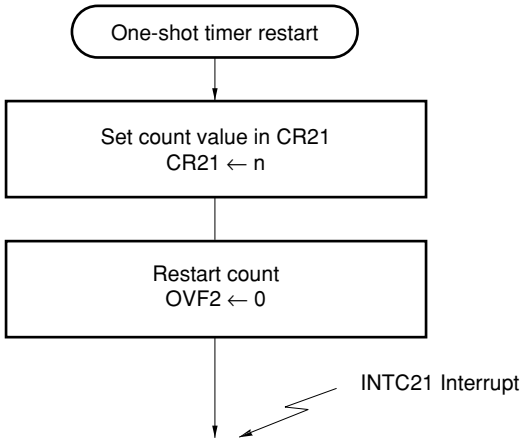


Figure 10-62 One-Shot Timer Operation Start Procedure from Second Time Onward



10.10 CAUTIONS

(1) While timer/counter 2 is operating (while the CE2 bit of the timer control register 1 (TMC1) is set), malfunctioning may occur if the contents of the following registers are rewritten. This is because it is undefined which takes precedence, the change in the hardware functions due to rewriting the register, or the change in the status because of the function before rewriting.
Therefore, be sure to stop the counter operation for the sake of safety before rewriting the contents of the following registers.

- Prescaler mode register 1 (PRM1)
- Capture/compare control register 2 (CRC2)
- Timer output control register (TOC)
- CMD2 bit of timer control register 1 (TMC)

(2) If the contents of the compare register (CR2n: n = 0, 1) match with those of TM2 when an instruction that stops timer register 2 (TM2) operation is executed, the counting operation of TM2 stops, but an interrupt request is generated. In order not to generate the interrupt when stopping the operation of TM2, mask the interrupt in advance by using the interrupt mask register before stopping TM2.

Example

Program that may generate interrupt request

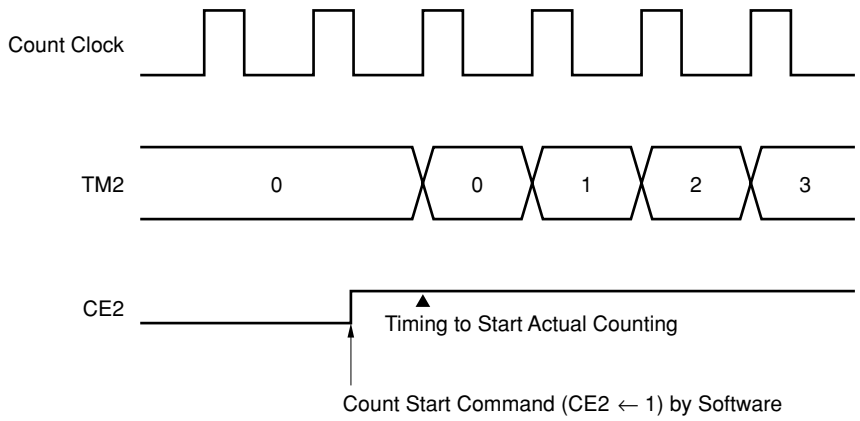
```
        ⋮
CLR1  CE2          ← Interrupt request
OR    MK0H, #03H     from timer/counter 2
        ⋮            occurs between
                     these instructions
```

Program that does not generate interrupt request

```
        ⋮
OR    MK0H, #03H   ← Disables interrupt from timer/
CLR1  CE2            counter 2
CLR1  CIF20        ← Clears interrupt request flag for timer/
CLR1  CIF21          counter 2
        ⋮
```

(3) Up to 1 count clock is required after an operation to start timer/counter 2 (CE2 ← 1) has been performed before timer/counter 2 actually starts (refer to **Figure 10-63**).
For example, when using timer/counter 2 as an interval timer, the first interval time is delayed by up to 1 clock. The second and those that follow are at the specified interval.

Figure 10-63 Operation When Counting is Started



(4) While an instruction that writes data to the compare register (CR2n: n = 0 or 1) is executed, coincidence between CR2n, to which the data is to be written, and timer register 2 (TM2) is not detected. For example, if the contents of CR2n do not change before and after the writing, the interrupt request is not generated even if the value of TM2 coincides with the value of CR2n, nor does the timer output (TOn + 2: n + 2 = 2, 3) change.
Write data to CR2n when timer/counter 2 is executing counting operation in the manner that the contents of TM2 do not match the value of CR2n before and after writing (e.g., immediately after an interrupt request has been generated because TM2 and CR2n have matched).

(5) Match between timer register 2 (TM2) and compare register (CR2n: n = 0, 1) is detected only when TM2 is incremented. Therefore, the interrupt request is not generated and timer output (TOn + 2: n + 2 = 2, 3) does not change even if the same value as TM2 is written to CR2n.

(6) During PPG output, if the PPG cycle is extremely short as compared with the time required to acknowledge an interrupt, the value of the compare register (CR2n: n = 0, 1) cannot be rewritten by interrupt processing that is performed on match between timer register (TM2) and compare register (CR2n). Use another method (for example, to poll the interrupt request flags by software with all the interrupts masked).

(7) The output level of the TOn (n = 2, 3) when the timer output is disabled (ENTOn = 0: n = 2, 3) is the reverse value of the value set to the ALVn (n = 2, 3) bit. Note, therefore, that an active level is output when the timer output is disabled with the PWM output function or PPG output function selected.

(8) If the value of the timer register is read under the condition indicated by "×" in Table 10-11, the read value may be illegal. Do not read the timer register under condition "×".

Table 10-11 Limits of Reading Timer Register

(√: Can be read, ×: Must not be read)

Timer Count Clock \ f_{CLK}	$f_{XX}/2$	$f_{XX}/4$	$f_{XX}/8$	$f_{XX}/16$
$f_{XX}/8$	√	√	×	×
$f_{XX}/16$	√	√	√	×
f_{XX}/n	√	√	√	√

Remarks 1. f_{XX}: Oscillation frequency
2. f_{CLK}: Internal system clock frequency
3. n = 32, 64, 128, 256, 512, 1,024, 2,048

(9) When using timer/counter 2 as an external event counter, the status where no valid edge is input cannot be distinguished from the status where only one valid edge has been input, by using timer register 2 (TM2) alone (refer to **Figure 10-64**), because the contents of TM2 are 0 in both the cases. To make a distinction, use the interrupt request flag of INTP2, as shown in Figure 10-65 (the INTP2 pin is multiplexed with the CI pin and both the functions can be used at the same time).

Figure 10-64 Example Where Whether One or No Valid Edge Has been Input Cannot Be Distinguished with External Event Counter

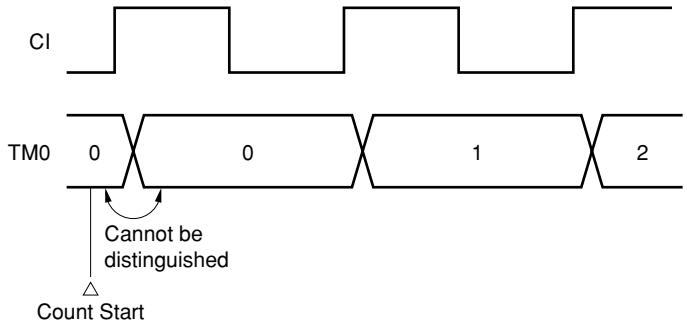
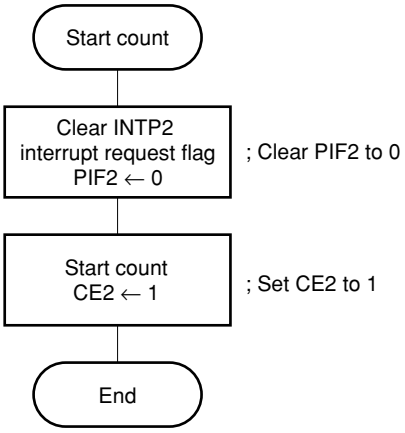
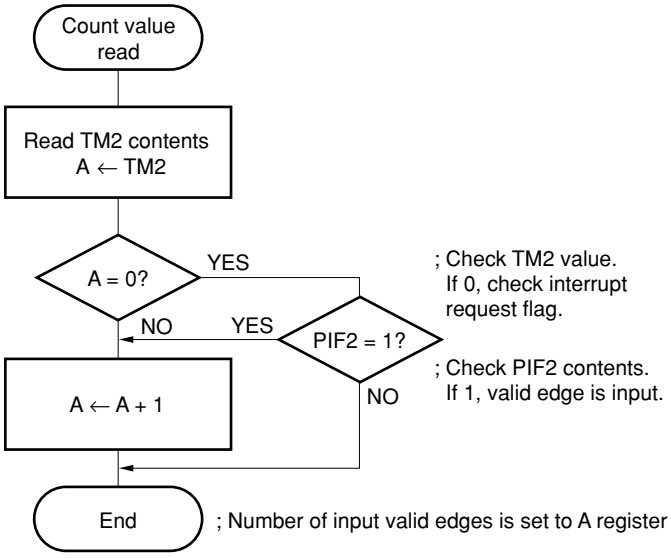


Figure 10-65 To Distinguish Whether One or No Valid Edge Has Been Input with External Event Counter

(a) Processing on starting counting



(b) Processing on reading count value

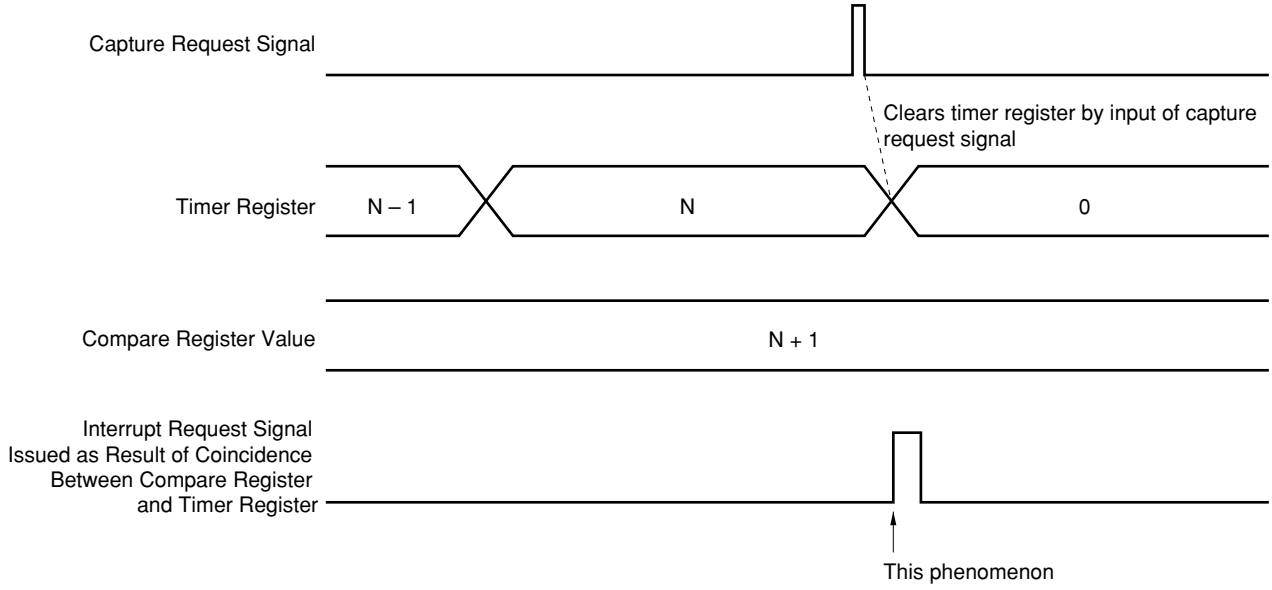


(10) Even if an attempt is mode to clear the timer register by inputting the capture request signal when the capture function of the timer is used, the timer register momentarily counts up immediately before it is cleared. Consequently, if a value greater than the value of the timer register by 1 is set to the compare register when the capture request signal is input, the values of the compare register and timer register coincide, and an unnecessary interrupt will be generated (refer to Figure 10-66). Therefore, take the following operation into consideration when creating a program.

<Operation>

Because the timer register is cleared at the next count if the capture request signal is generated when the value of timer register is "N" when the value "N + 1" is set to the compare register, no interrupt request is generated by the compare register. Actually, however, the timer register momentarily counts "N + 1" when the timer register is cleared. As a result, the values of the timer register and compare register coincide, and an interrupt request signal is generated by the compare register.

Figure 10-66 Example of Generation of Unnecessary Interrupt Request by Compare Register

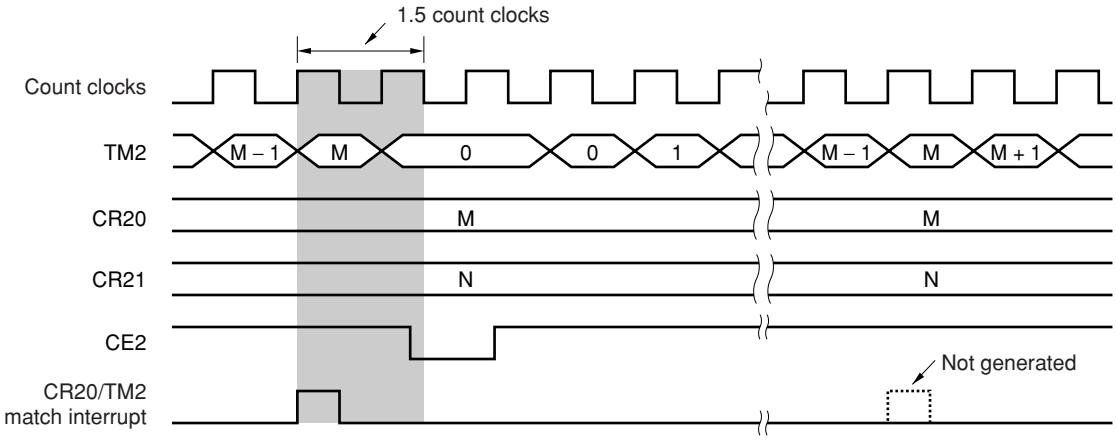


★ (11) If the count operation of TM2 stops at the timing at which compare register (CR20) and timer register 2 (TM2) match, the CR20/TM2 match interrupt may not be generated after timer/counter 2 is next started.

If the TM2 count operation is stopped within 1.5 count clocks after a match between CR20 and TM2, the first match interrupt after timer/counter 2 is next started will not be generated. The second and subsequent interrupts operate normally. Note that the timer output is unaffected by this bug.

This bug occurs because the timer interrupt controller inadvertently masks interrupts if timer/counter 2 is stopped in the period indicated by the shaded area in the figure below.

The interrupt controller is initialized by an overflow of timer/counter 2 or a match between CR21 and TM2.



Remark M < N

Do not stop timer/counter 2 within 1.5 count clocks after a match between CR20 and TM2.

Disable all interrupt requests (including macro servicing), read the value of the timer to be stopped, and wait until at least 1.5 count clocks have elapsed after a match between CR20 and TM2 before stopping timer/counter 2.

CHAPTER 11 TIMER 3

11.1 FUNCTION

Timer 3 is a 16- or 8-bit timer.

In addition to its function as an interval timer, it can be used as a counter for clocked serial interface (CSI) clock generation.

The interval timer generates internal interrupts at pre-set intervals. The interval setting range is shown in Table 11.1.

Table 11-1 Timer 3 Intervals

Minimum Interval	Maximum Interval	Resolution
8/fxx (0.25 μs)	$2^{16} \times 8$/fxx (16.40 ms)	8/fxx (0.25 ms)
16/fxx (0.50 μs)	$2^{16} \times 16$/fxx (32.80 ms)	16/fxx (0.50 ms)
32/fxx (1.00 μs)	$2^{16} \times 32$/fxx (65.50 ms)	32/fxx (1.00 ms)
64/fxx (2.00 μs)	$2^{16} \times 64$/fxx (131 ms)	64/fxx (2.00 ms)
128/fxx (4.00 μs)	$2^{16} \times 128$/fxx (262 ms)	128/fxx (4.00 ms)
256/fxx (8.00 μs)	$2^{16} \times 256$/fxx (524 ms)	256/fxx (8.00 ms)
512/fxx (16.00 μs)	$2^{16} \times 512$/fxx (1.05 s)	512/fxx (16.00 ms)
1,024/fxx (32.00 μs)	$2^{16} \times 1{,}024$/fxx (2.10 s)	1,024/fxx (32.00 ms)
2,048/fxx (64.00 μs)	$2^{16} \times 2{,}048$/fxx (4.19 s)	2,048/fxx (64.00 ms)

( ): When fxx = 32 MHz

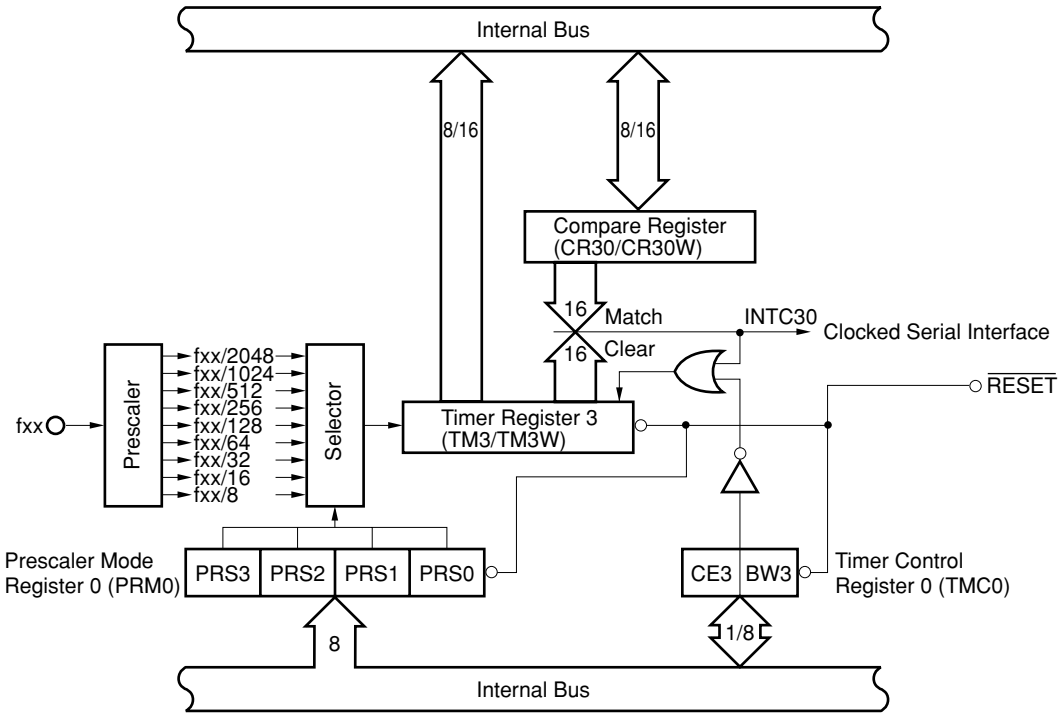
11.2 CONFIGURATION

Timer 3 consists of the following registers:

- Timer register (TM3/TM3W) × 1
- Compare register (CR30/CR30W) × 1

The block diagram of timer 3 is shown in Figure 11-1.

Figure 11-1 Timer 3 Block Diagram



(1) Timer register 3 (TM3/TM3W)

TM3/TM3W are timer registers that count up using the count clock specified by the high-order 4 bits of prescaler mode register 0 (PRM0).

The count operation is stopped or enabled by the timer control register 0 (TMC0). In addition, an 8-bit mode (TM3) or 16-bit mode (TM3W) can be selected.

TM3 can be read only with an 8/16-bit manipulation instruction.

When $\overline{\text{RESET}}$ is input, TM3 is cleared to 00H and the count is stopped.

Caution If the value of the timer register is read under the condition indicated by "×" in Table 11-2, the read value may be illegal. Do not read the timer register under condition "×".

Table 11-2 Limits of Reading Timer Register

(√: Can be read, ×: Must not be read)

Timer Count Clock \ f_{CLK}	$f_{XX}/2$	$f_{XX}/4$	$f_{XX}/8$	$f_{XX}/16$
$f_{XX}/8$	√	√	×	×
$f_{XX}/16$	√	√	√	×
f_{XX}/n	√	√	√	√

Remarks 1. f_{XX}: Oscillation frequency

2. f_{CLK}: Internal system clock frequency

3. n = 32, 64, 128, 256, 512, 1,024, 2,048

(2) Compare register (CR30/CR30W)

CR30/CR30W are 8/16-bit registers that hold the value that determines the interval timer frequency.

If the CR30/CR30W contents match the contents of TM3/TM3W, the contents of TM3/TM3W are cleared automatically and an interrupt request (INTC30) is generated.

This compare register operates as CR30 in the 8-bit mode and CR30W in the 16-bit mode.

The CR30 register can be read or written to with an 8/16-bit manipulation instruction. The contents of CR30 are undefined after $\overline{\text{RESET}}$ input.

(3) Prescaler

The prescaler generates the count clock from the internal system clock. The clock generated by the prescaler is selected by the selector, and is used as the count clock by the timer to perform count operations.

(4) Selector

The selector selects a signal resulting from dividing the internal clock or the edge detected by the edge detection circuit as the count clock of timer register 3 (TM3/TM3W).

11.3 TIMER 3 CONTROL REGISTERS

(1) Timer control register 0 (TMC0)

TMC0 controls the timer 3, TM3/TM3W, count operation by the high-order 4 bits (the low-order 4 bits control the count operation of timer/counter 0, TM0).

TMC0 can be read or written to with an 8-bit manipulation instruction. The format of the TMC0 is shown in Figure 11-2.

$\overline{\text{RESET}}$ input clears TMC0 to 00H.

Figure 11-2 Timer Control Register 0 (TMC0) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
TMC0	CE3	0	0	BW3	CE0	OVF0	0	0	0FF5DH	00H	R/W

Controls count operation of timer/counter 0, TM0 (see **Figure 8-2**).

BW3	Timer 3 Bit Length Specification
0	8-bit operating mode
1	16-bit operating mode

CE3	TM3/TM3W Count Operation Control
0	Count operation stopped with count cleared
1	Count operation enabled

(2) Prescaler mode register 0 (PRM0)

PRM0 specifies the count clock to timer/counter 3 TM3/TM3W by the high-order 4 bits (the low-order 4 bits specify the count clock to timer/counter 0, TM0).

PRM0 can be read and written with an 8-bit manipulation instruction. The format of the PRM0 is shown in Figure 11-3.

$\overline{\text{RESET}}$ input clears PRM0 to 11H.

Figure 11-3 Prescaler Mode Register 0 (PRM0) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PRM0	PRS3	PRS2	PRS1	PRS0	PRS03	PRS02	PRS01	PRS00	0FF5CH	11H	R/W

Specifies count clock to timer/counter 0 (TM0), TM0
(see **Figure 8-3**).

(f_{XX} = 32 MHz)

PRS3	PRS2	PRS1	PRS0	Timer 3 TM3/TM3W Count Clock Specification	
				Count Clock [Hz] Specification	Resolution [μs]
0	0	0	0	Setting prohibited	–
0	0	0	1	f_{XX}/8	0.25
0	0	1	0	f_{XX}/16	0.50
0	0	1	1	f_{XX}/32	1.00
0	1	0	0	f_{XX}/64	2.00
0	1	0	1	f_{XX}/128	4.00
0	1	1	0	f_{XX}/256	8.00
0	1	1	1	f_{XX}/512	16.00
1	0	0	0	f_{XX}/1,024	32.00
1	0	0	1	f_{XX}/2,048	64.00
Other than the above				Setting prohibited	

11.4 TIMER REGISTER 3 (TM3) OPERATION

11.4.1 Basic Operation

Timer 3 can operate in an 8-bit or 16-bit mode. These operation modes are selected by bit 4 (BW3) of timer control register 0 (TMC0) **Note**.

In the timer 3 count operation, an up-count is performed using the count clock specified by the high-order 4 bits of prescaler mode register 0 (PRM0).

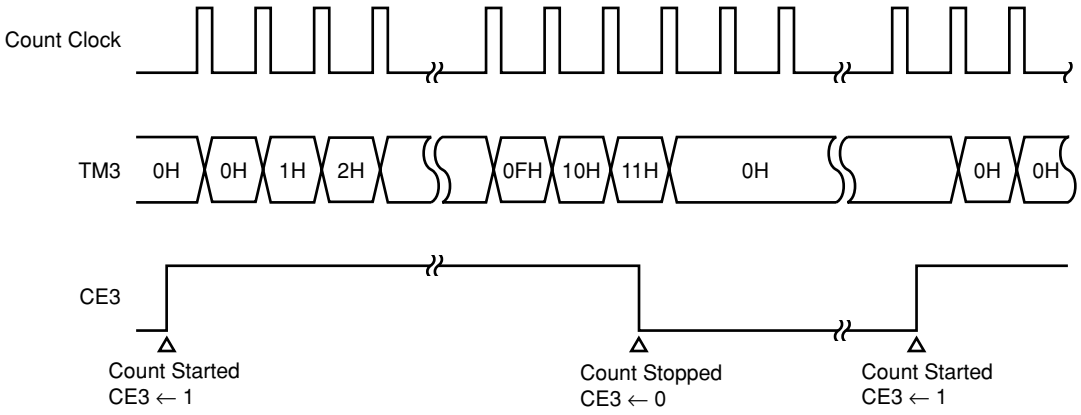
When $\overline{\text{RESET}}$ is input, TM3 is cleared to 0000H, and the count operation is stopped.

Count operation enabling/disabling is controlled by bit 7 (CE3) of timer control register 0 (TMC0) (the high-order 4 bits of TMC0 control timer 3 operation). When the CE3 bit is set (to 1) by software, the contents of TM3 are immediately cleared on the first count clock, and then the up-count operation is performed. When the CE3 bit is cleared (to 0), TM3 becomes 0H immediately, and match signal generation is stopped. If the CE3 bit is set (to 1) again when it is already set (to 1), TM3 continues the count operation without being cleared.

Note Unless there functional differences are found, the register names in the 8-bit mode are used. In the 16-bit mode, the register names TM3 and CR30 are TM3W and CR30W, respectively.

Figure 11-4 Basic Operation in 8-Bit Operating Mode (BW3 = 0)

(a) Count started → count stopped → count started



(b) When "1" is written to the CE3 bit again after the count starts

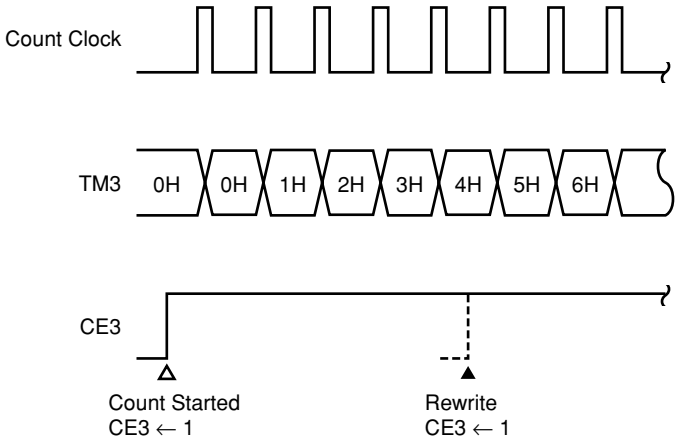
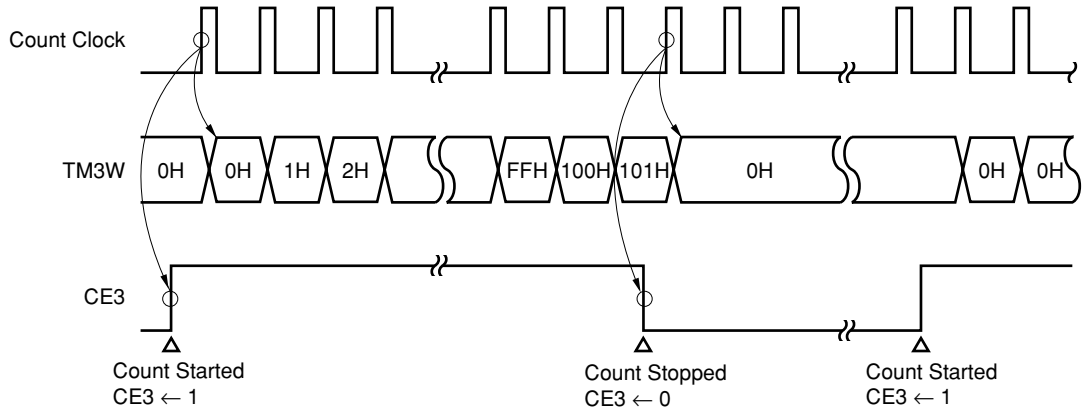
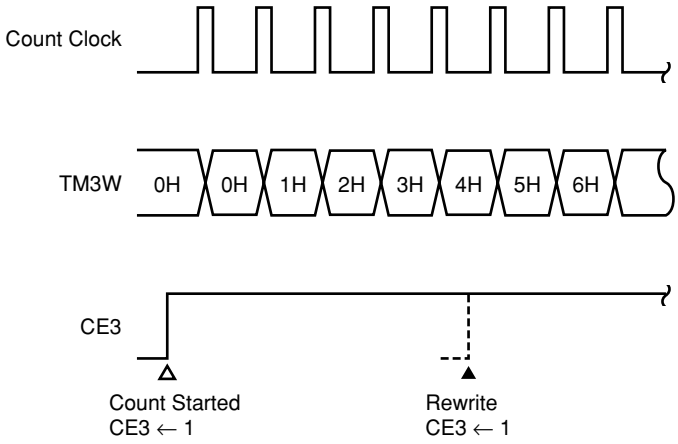


Figure 11-5 Basic Operation in 16-Bit Operating Mode (BW3 = 1)

(a) Count started → count stopped → count started



(b) When "1" is written to the CE3 bit again after the count starts

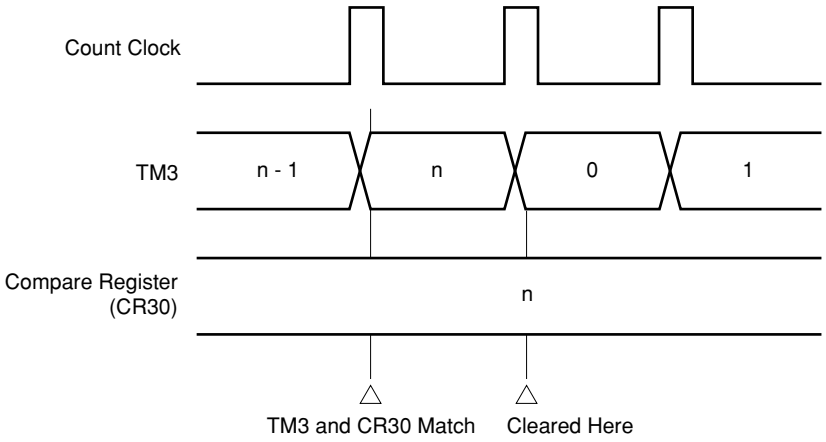


11.4.2 Clear Operation

(1) Clear operation by match with compare register (CR30)

16-bit timer 3 (TM3) is cleared automatically after a match with the compare register (CR30). When a clearance source arises, TM3 is cleared to 0H on the next count clock. Therefore, even if a clearance source arises, the value at the point at which the clearance source arose is retained until the next count clock arrives.

Figure 11-6 TM3 Clearance by Match with Compare Register (CR30)

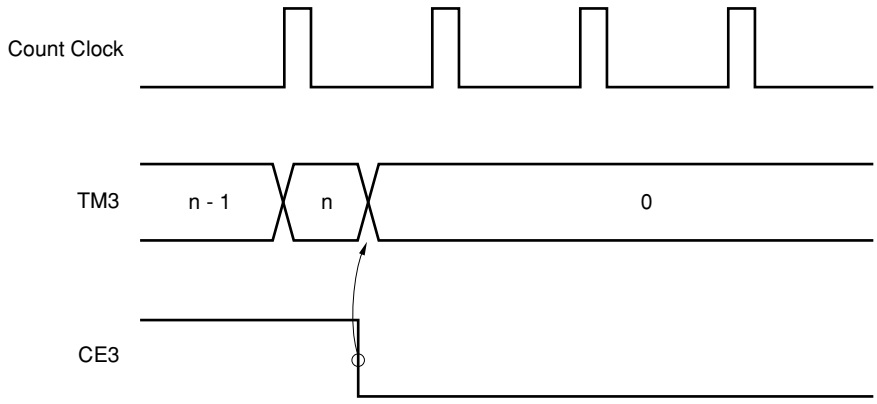


(2) Clear operation by CE3 bit of timer/control register 0 (TMC0)

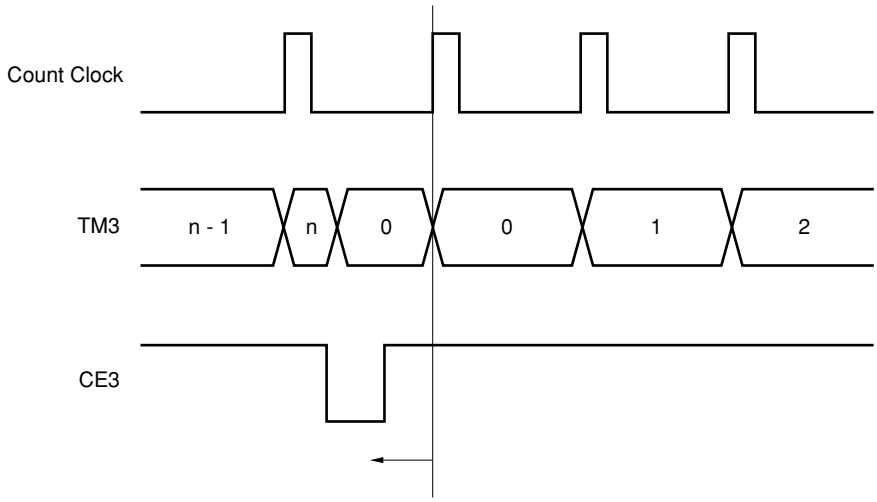
Timer register 3 (TM3) is also cleared when the CE3 bit of TMC0 is cleared (to 0) by software. The clear operation is performed following clearance (to 0) of the CE3 bit in the same way.

Figure 11-7 Clear Operation When CE3 Bit is Cleared (to 0)

(a) Basic operation

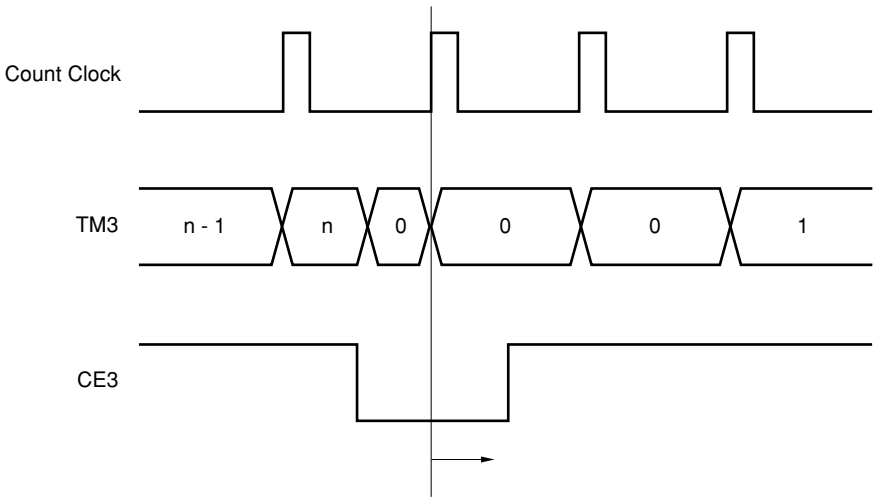


(b) Restart before count clock is input after clearance



If the CE3 bit is set (to 1) before this count clock, the count starts from 0 on this count clock

(c) Restart when count clock is input after clearance



If the CE3 bit is set (to 1) from this count clock onward, the count starts from 0 on the count clock after the CE3 bit is set (to 1).

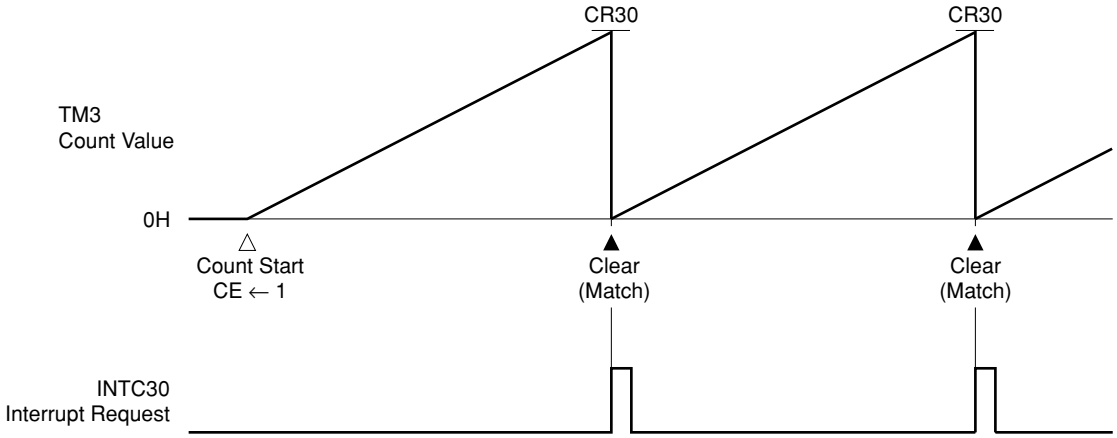
11.5 COMPARE REGISTER OPERATION

Timer 3 performs compare operations in which the value set in the compare register (CR30) is compared with the timer register 3 (TM3) count value.

If the count value of TM3 matches the preset CR30 value as the result of the count operation, an interrupt request (INTC30) is generated.

After a match, the TM3 contents are cleared automatically, and therefore TM3 functions as an interval timer that repeatedly counts up to the value set in the CR30.

Figure 11-8 Compare Operation



11.6 EXAMPLE OF USE

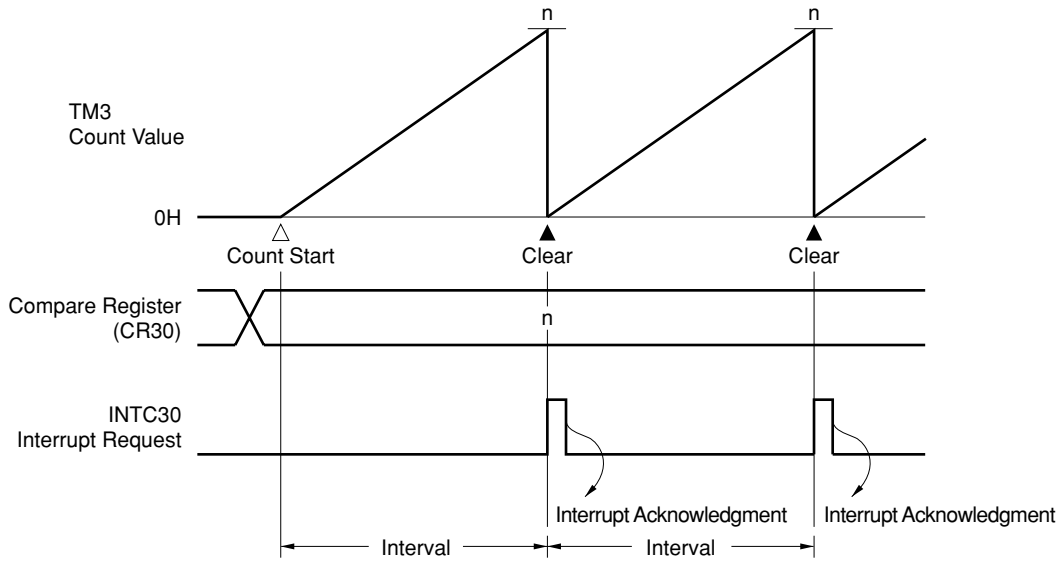
Operation as interval timer:

TM3 operates as an interval timer that generates interrupts repeatedly with the pre-set count time as the interval (see **Figure 11-9**). TM3 can also be used for baud rate generation.

This interval timer can count up to a maximum of 16.40 ms at the minimum resolution of 0.25 μs, and up to 4.19 s at the maximum resolution of 64.00 μs (internal system clock f_{XX} = 32 MHz).

The control register settings are shown in Figure 11-10, and the setting procedure in Figure 11-11.

Figure 11-9 Interval Timer Operation Timing



Remark Interval = (n + 1) × x/fxx

0 ≤ n ≤ FFH, x = 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048

Figure 11-10 Control Register Settings for Interval Timer Operation

Prescaler mode register 0 (PRM0)

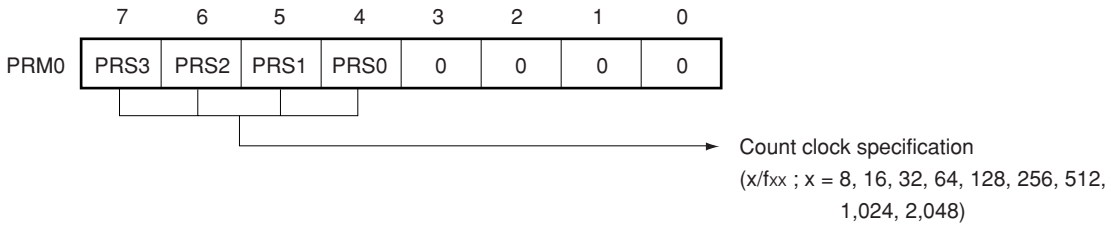
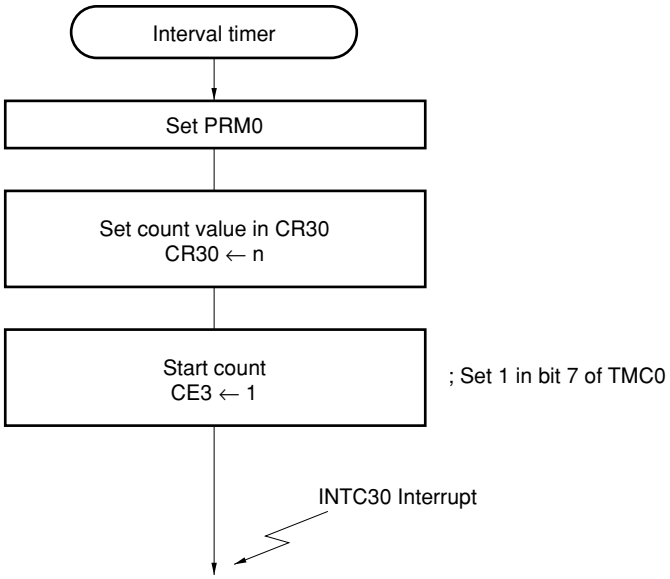


Figure 11-11 Interval Timer Operation Setting Procedure



11.7 CAUTIONS

(1) There is a possibility of malfunction if the contents of prescaler mode register 0 (PRM0) are rewritten while the timer 3 is running (when the CE3 bit of the timer control register 0 (TMC0) is set). The malfunction occurs as there is no defined order of priority in the event of contention between the timings at which the hardware function changes due to a register rewrite and the status changes in the function prior to the rewrite.
When the contents of PRM0 are rewritten, counter operations must be stopped first to ensure stability.

(2) If the compare register (CR30) and TM3 contents match when an instruction that stops timer register 3 (TM3) operation is executed, the TM3 count operation stops, but an interrupt request is generated.
If you do not want an interrupt to be generated when TM3 operation is stopped, interrupts should be masked by means of interrupt the mask register before stopping the TM3.

Example

Program in which an interrupt request may be generated

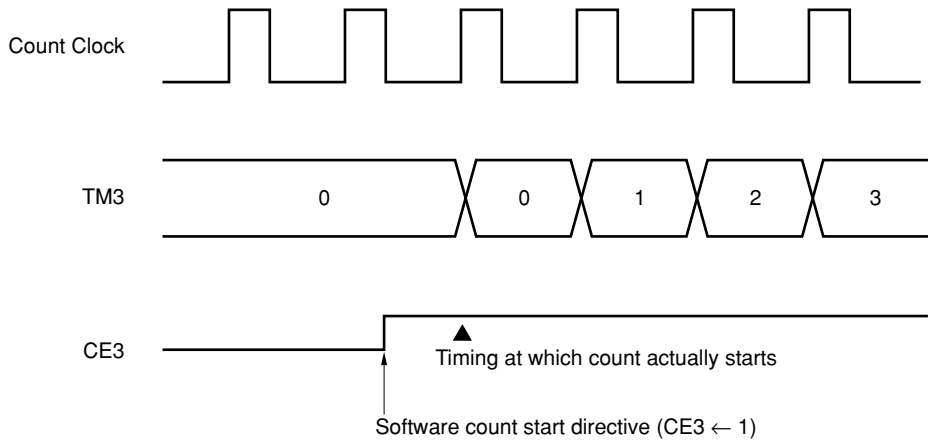
```
        ⋮
CLR1  CE3
SET1  CMK30
        ⋮                ← Interrupt request generated
        ⋮                  by timer 3 here
```

Program in which an interrupt request is not generated

```
        ⋮
SET1  CMK30
CLR1  CE3          ← Disables interrupts from timer 3
CLR1  CIF30
        ⋮          ← Clears timer 3 interrupt request flag
```

(3) There is a delay of up to one count clock between the operation that starts a timer 3 (CE3 ← 1) and the actual start of the timer/counter (see **Figure 11-23**).
For example, if a timer/counter is used as an interval timer, the first interval will be extended by up to one clock. The second and subsequent intervals will be as specified.

Figure 11-12 Operation When Count Starts



(4) While an instruction that writes data to the compare register (CR30) is executed, match between CR30, to which the data is to be written, and timer register 3 (TM3) is not detected.
Write data to CR30 when timer 3 is executing counting operation so that the contents of TM3 do not match the value of CR30 before and after writing (e.g., immediately after an interrupt request has been generated because TM3 and CR30 have matched).

(5) Match between timer register 3 (TM3) and compare register (CR30) is detected only when TM3 is incremented. Therefore, the interrupt request is not generated even if the same value as TM3 is written to CR30.

(6) If the value of the timer register is read under the condition indicated by "×" in Table 11-3, the read value may be illegal. Do not read the timer register under condition "×".

Table 11-3 Limits of Reading Timer Register

(√: Can be read, ×: Must not be read)

f_{CLK} / Timer Count Clock	$f_{XX}/2$	$f_{XX}/4$	$f_{XX}/8$	$f_{XX}/16$
$f_{XX}/8$	√	√	×	×
$f_{XX}/16$	√	√	√	×
f_{XX}/n	√	√	√	√

Remarks 1. f_{XX}: Oscillation frequency
2. f_{CLK}: Internal system clock frequency
3. n = 32, 64, 128, 256, 512, 1,024, 2,048

CHAPTER 12 WATCHDOG TIMER FUNCTION

The watchdog timer is a timer that detects inadvertent program loops.

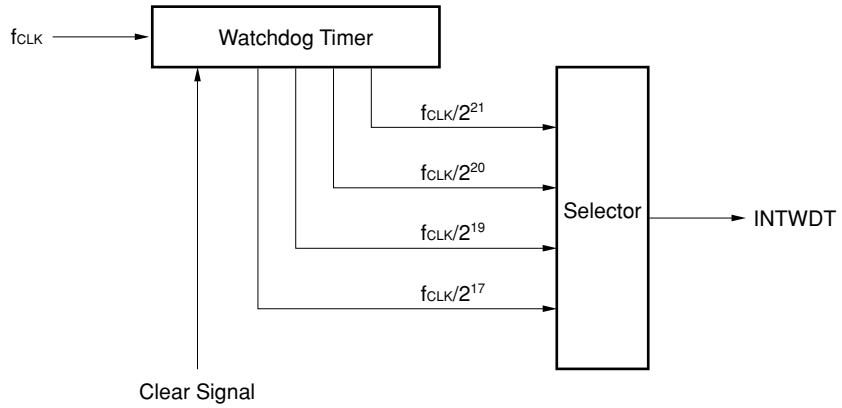
Watchdog timer interrupts are used to detect system or program errors. For this purpose, instructions that clear the watchdog timer (start the count) within a given period are inserted at various places in a program.

If an instruction that clears the watchdog timer is not executed within the set time and the watchdog timer overflows, a watchdog timer interrupt (INTWDT) is generated and a program error is reported.

12.1 CONFIGURATION

The watchdog timer block diagram is shown in Figure 12-1.

Figure 12-1 Watchdog Timer Block Diagram



12.2 WATCHDOG TIMER MODE REGISTER (WDM)

The WDM is an 8-bit register that controls the watchdog timer operation.

To prevent erroneous clearing of the watchdog timer by an inadvertent program loop, writing can only be performed by a dedicated instruction. This dedicated instruction, MOV WDM, #byte, has a special code configuration (4 bytes), and a write is not performed unless the 3rd and 4th bytes of the operation code are mutual complements.

If the 3rd and 4th bytes of the operation code are not complements, a write is not performed and an operand error interrupt is generated. In this case, the return address saved in the stack area is the address of the instruction that was the source of the error, and thus the address that was the source of the error can be identified from the return address saved in the stack area.

If recovery from an operand error is simply performed by means of an RETB instruction, an endless loop will result.

As an operand error interrupt is only generated in the event of an inadvertent program loop (with the NEC Electronics assembler, RA78K4, only the correct dedicated instruction is generated when MOV WDM, #byte is written), system initialization should be performed by the program.

Other write instructions (MOV WDM, A, AND WDM, #byte, SET1 WDM.7, etc.) are ignored and do not perform any operation. That is, a write is not performed to the WDM, and an interrupt such as an operand error interrupt is not generated.

After a system reset ($\overline{\text{RESET}}$ input), once the watchdog timer has been started (by setting (to 1) the RUN bit), the WDM contents cannot be changed. The watchdog timer can only be stopped by a reset, but can be cleared at any time with a dedicated instruction.

The WDM can be read at any time by a data transfer instruction.

$\overline{\text{RESET}}$ input clears the WDM to 00H.

The WDM format is shown in Figure 12-2.

Figure 12-2 Watchdog Timer Mode Register (WDM) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
WDM	RUN	0	0	PRC	0	WDI2	WDI1	0	FFC2H	00H	R/W

WDI2	WDI2	Overflow Time [ms] f_{CLK} = 16 MHz
0	0	$2^{17}/f_{CLK}$ (8.19)
0	1	$2^{19}/f_{CLK}$ (32.77)
1	0	$2^{20}/f_{CLK}$ (65.54)
1	1	$2^{21}/f_{CLK}$ (131.07)

Remark f_{CLK}: Internal System Clock Frequency

PRC	Watchdog Timer Interrupt Request Priority Specification
0	Watchdog timer interrupt request < NMI pin input interrupt request
1	Watchdog timer interrupt request > NMI pin input interrupt request

RUN	Watchdog Timer Operation Specification
0	Watchdog timer stopped
1	Clear watchdog timer and start count

Cautions 1. The watchdog timer mode register (WDM) can only be written to with a dedicated instruction (MOV WDM, #byte).

2. The same value should be written each time in writes to the WDM to set (to 1) the RUN bit. The contents written the first time cannot be changed even if a different value is written.

3. Once the RUN bit has been set (to 1), it cannot be reset (to 0) by software.

12.3 OPERATION

12.3.1 Count Operation

The watchdog timer is cleared, and the count started, by setting (to 1) the RUN bit of the watchdog timer mode register (WDM). When overflow time specified by the WDM2 and WDM1 bits of WDM has elapsed after the RUN bit has been set (to 1), a non-maskable interrupt (INTWDT) is generated.

If the RUN bit is set (to 1) again before the overflow time elapses, the watchdog timer is cleared and the count operation is started again.

12.3.2 Interrupt Priorities

The watchdog timer interrupt (INTWDT) is a non-maskable interrupt. Other non-maskable interrupts are interrupts from the NMI pin (NMI). The order of acknowledgment when an INTWDT interrupt and NMI interrupt are generated simultaneously can be specified by the setting of bit 4 of the watchdog timer mode register (WDM).

Even if INTWDT is generated while the NMI processing program is executed when NMI acknowledgement is specified to take precedence, INTWDT is not acknowledged until completion of execution of the NMI processing program.

12.4 CAUTIONS

12.4.1 General Cautions on Use of Watchdog Timer

(1) The watchdog timer is one means of detecting inadvertent program loops, but it cannot detect all inadvertent program loops. Therefore, in equipment that requires a high level of reliability, you should not rely on the on-chip watchdog timer alone, but should use external circuitry for early detection of inadvertent program loops, to enable processing to be performed that will restore the normal state or establish a stable state and then stop the operation.

(2) The watchdog timer cannot detect inadvertent program loops in the following cases.

<1> If watchdog timer clearance is performed in the timer interrupt service program
<2> If cases where an interrupt request or macro service is held pending (see **22.9**) occur consecutively
<3> If the watchdog timer is cleared periodically when inadvertent program looping is due to an error in the program logic (if each module of the program functions normally but the overall program does not)
<4> If the watchdog timer is periodically cleared by a group of instructions executed when an inadvertent program loop occurs
<5> If the STOP mode, HALT mode, or IDLE mode is entered as the result of an inadvertent program loop
<6> If watchdog timer runaway also occurs in the event of CPU runaway due to external noise

In cases <1>, <2> and <3> the program can be amended to allow detection to be performed.
In case <4>, the watchdog timer can only be cleared by a 4-byte dedicated instruction. Similarly, in case <5>, the STOP mode, HALT mode, or IDLE mode cannot be set unless a 4-byte dedicated instruction is used. For state <2> to be entered as the result of an inadvertent program loop, 3 or more consecutive bytes of data must comprise a specific pattern (e.g. BT PSWL.bit, $$, etc.). Therefore, the establishment of state <2> as the result of <4>, <5> or an inadvertent program loop is likely to be extremely rare.

12.4.2 Cautions on μPD784038 Subseries Watchdog Timer

(1) The watchdog timer mode register (WDM) can only be written to with a dedicated instruction (MOV WDM, #byte).

(2) The same value should be written each time in writes to the watchdog timer mode register (WDM) to set (to 1) the RUN bit. The contents written the first time cannot be changed even if a different value is written.

(3) Once the RUN bit has been set (to 1), it cannot be reset (to 0) by software.

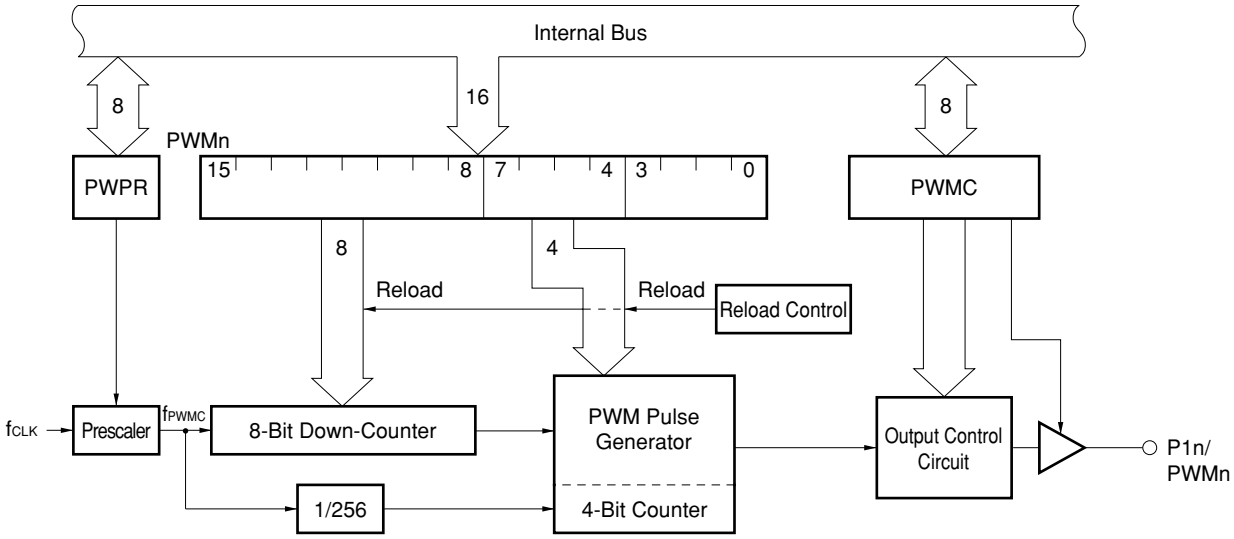
CHAPTER 13 PWM OUTPUT UNIT

The μPD784038 incorporates two 12-bit resolution PWM (pulse width modulation) output circuit channels. The active level of the PWM output pulses can be selected as high or low. The PWM output ports have a dual function as pins P10 and P11.

13.1 PWM OUTPUT UNIT CONFIGURATION

The PWM output unit configuration is shown in Figure 13-1.

Figure 13-1 PWM Output Unit Configuration



Remark n = 0, 1

(1) 8-bit down-counter

Generates the basic PWM signal timing.

(2) PWM pulse generator (including 4-bit counter)

Controls addition of extra pulses and generates the PWM pulses to be output.

(3) Reload control

Controls 8-bit down counter and 4-bit count modulo value reloading.

(4) Output control circuit

Controls the active level of the PWM signal.

(5) Prescaler

Scales f_{CLK}, and generates the reference clock.

13.2 PWM OUTPUT UNIT CONTROL REGISTERS

13.2.1 PWM Control Register (PWMC)

The PWMC is an 8-bit register that controls the operating status of the PWM output pins (PWMn: n = 0, 1).

The PWMC can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. Its format is shown in Figure 13-2.

When $\overline{\text{RESET}}$ is input, PWMC is set to 05H, the PWMn pin is set to port mode, and the input state (output high impedance) is set.

Figure 13-2 PWM Control Register (PWMC) Format

	7	6	5	4	③	2	①	0	Address	After Reset	R/W
PWMC	SYN1	0	SYN0	0	EN1	ALV1	EN0	ALV0	0FF70H	05H	R/W

(n = 0, 1)

ALVn	PWMn Pin PWM Active Level Specification
0	Active-low
1	Active-high

ENn	PWMn Pin PWM Output Control
0	Output disabled. The pin level is determined by the contents of th PM1, P1 and PU0 (port mode).
1	PWM output enabled

SYNn	PWM Pulse Width Rewrite Cycle Specification
0	Rewritten every 16 PWM cycles ($2^{12}/f_{PWMC}$)
1	Rewritten every PWM cycle ($2^{8}/f_{PWMC}$)

13.2.2 PWM Prescaler Register (PWPR)

The PWPR is an 8-bit register that selects the PWM output circuit operating clock (f_{PWMC}).

The PWPR can be read or written to with an 8-bit manipulation instruction. Its format is shown in Figure 13-3.

When $\overline{RESET}$ is input, PWPR is cleared to 00H, and f_{CLK} is selected as f_{PWMC} for both channels.

Figure 13-3 PWM Prescaler Register (PWPR) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PWPR	0	PWP12	PWP11	PWP10	0	PWP02	PWP01	PWP00	0FF71H	00H	R/W

(n = 0, 1)

PWPn2	PWPn1	PWPn0	PWMn Operating Clock (f_{PWMC})	PWMn Repetition Frequency (f_{CLK} = 16 MHz)
0	0	0	f_{CLK}	f_{CLK}/256 (62.5 kHz)
0	0	1		
0	1	0	f_{CLK}/2	f_{CLK}/512 (31.3 kHz)
0	1	1	f_{CLK}/3	f_{CLK}/768 (20.8 kHz)
1	0	0	f_{CLK}/4	f_{CLK}/1,024 (15.6 kHz)
Other than the above			Setting prohibited	

13.2.3 PWM Modulo Registers (PWM0, PWM1)

The PWM modulo register 16-bit register (PWMn: n = 0, 1) is a 16-bit register that determines the PWM pulse width. Reads/writes by a 16-bit manipulation instruction only are possible for data setting.

The contents of bits 4 to 15 of the PWMn determines the 12-bit PWM pulse width (12-bit resolution). Bits 3 to 0 have no meaning, and PWM output is not affected whether 1 or 0 is written to these bits.

When $\overline{RESET}$ is input, the PWMn content are undefined, and therefore data must be set by the program before PWM output is enabled.

Caution A value between 0000H and 00FFH should not be set in the PWM modulo registers (PWMn: n = 0, 1). A value between 0100H and FFFFH should be set in the PWMn registers. Outputtable PWM signal duty values are 17/4,096 to 4,096/4,096.

13.3 PWM OUTPUT UNIT OPERATION

13.3.1 Basic PWM Output Operation

The PWM pulse output duty is determined by the value set in bits 4 to 15 of the PWM modulo register (PWMn: n = 0, 1) as shown below.

$$\text{PWM pulse output duty} = \frac{(\text{Value of PWMn bits 4 to 15})^{\text{Note}} + 1}{4{,}096}$$

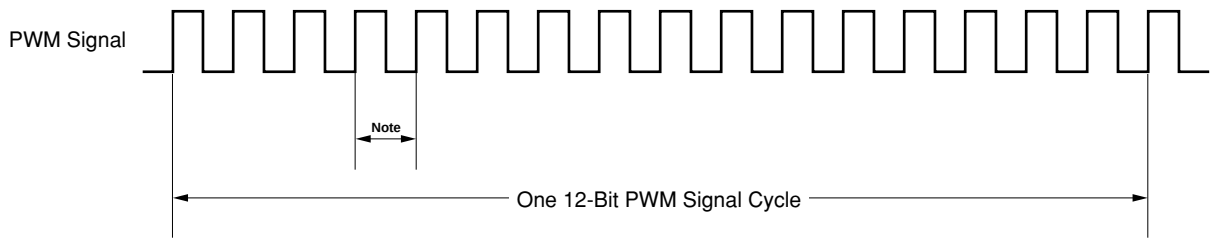
Note 16 ≤ (Value of PWMn bits 4 to 15) ≤ 4,095

The PWM pulse output repetition frequency is the frequency obtained by division-by-256 of the PWM clock f_{CLK}/1 to f_{CLK}/4 set by the PWM prescaler register (PWPR) (= f_{PWMC}/256), and the minimum pulse width is 1/f_{PWMC}.

In PWM pulse output, 12-bit resolution is achieved by repeating output of a f_{PWMC}/256 repetition frequency 8-bit resolution PWM signal 16 times.

The addition of extra pulses (1/f_{PWMC}) to the 8-bit resolution PWM pulses determined by bits 8 to 15 of the PWMn every cycle is controlled in accordance with the value of bits 4 to 7 of the PWMn to implement a PWM pulse signal once every 16 cycles.

Figure 13-4 Basic PWM Output Operation



Note 8-bit resolution per PWM pulse cycle

13.3.2 PWM Pulse Output Enabling/Disabling

When PWM pulses are output, the ENn (n = 0, 1) bits of the PMC register are set (to 1) after data is set in the PWM prescaler register (PWPR) and PWM modulo register (PWMn: n = 0, 1). As a result, PWM pulses with the active level specified by ALVn (n = 0, 1) bit of the PWM control register (PWMC) are output from the PWM output pin.

When the ENn bits of the PWMC are cleared (to 0), the PWM output unit immediately stops the PWM output operation and the PWM output pins are set to the state specified by the PM1, P1 and PUO registers.

That is, when PM1n (n = 0, 1) in the port 1 mode register (PM1) is 0, the output state is set and the contents of P1n (n = 0, 1) are specified. When PM1n = 1 (n = 0, 1) the input state is set, when PUO1 in the pull-up resistor option register (PUO) is 1 the high level is set by the on-chip pull-up resistor, and when PUO1 = 0 the output high-impedance state is set.

13.3.3 PWM Pulse Active Level Specification

The ALVn (n = 0, 1) bit of the PWM control register (PWMC) specify the active level of PWM pulses output from the PWM output pins.

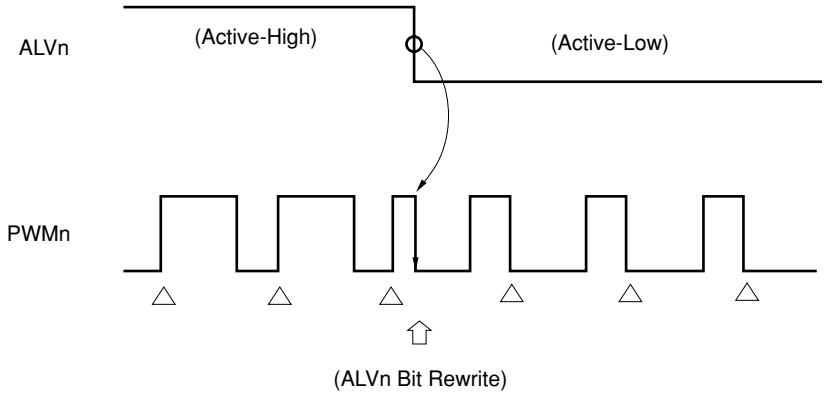
When ALVn bit is set (to 1), active-high level pulses are output, and when cleared (to 0), active-low level pulses are output.

When ALVn bit is rewritten, the PWM active level changes immediately. PWM output active level setting and pin states are shown in Figure 13-5.

Figure 13-5 shows the case where ALVn bit is switched when the ENn (n = 0/1) bit of the PWMC is set (to 1) and PWM output is enabled.

The pin state does not change if ALVn is rewritten when ENn bit is in the cleared (to 0) state.

Figure 13-5 PWM Output Active Level Setting



Remark ENn = 1 (n = 0, 1)

13.3.4 PWM Pulse Width Rewrite Cycle Specification

The start of PWM output and pulse width changes are performed in synchronization either with every 16 PWM pulse cycles ($2^{12}/f_{PWMC}$) or with every PWM pulse cycle ($2^{8}/f_{PWMC}$). This PWM pulse width rewrite cycle specification is performed by means of the SYNn bits of the PWM control register (PWMC).

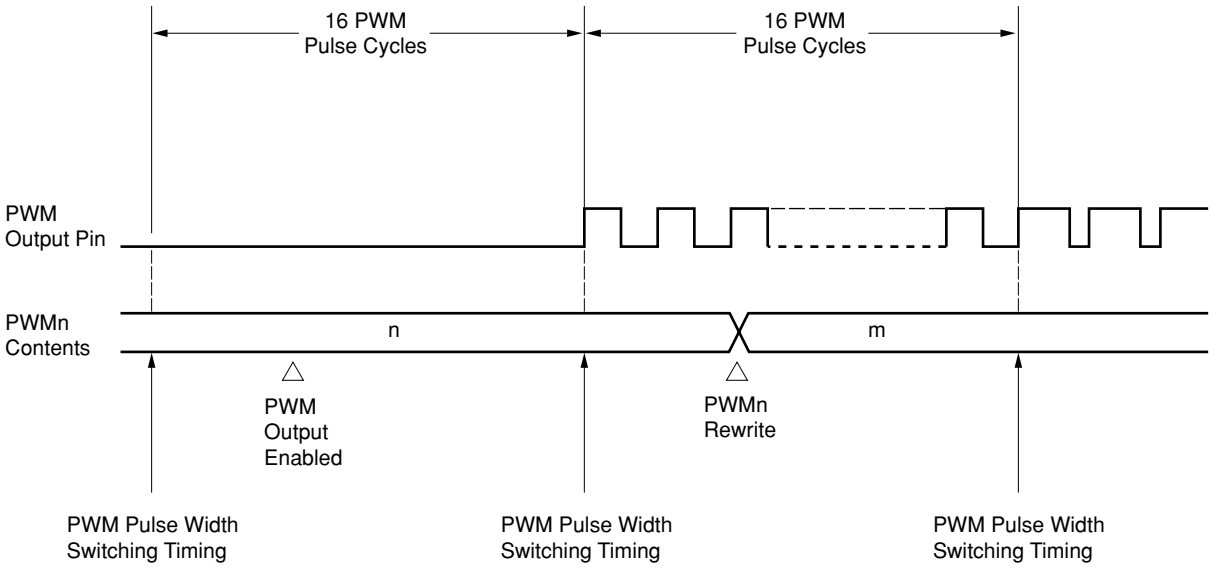
When the SYNn bit is cleared (to 0), a pulse width change is performed every 16 PWM pulse cycles ($2^{12}/f_{PWMC}$). It therefore takes a maximum of 2^{12} clocks (256 μs when f_{PWMC} = 16 MHz) until a pulse of a width corresponding to the data written in the PWM modulo register (PWMn: n = 0, 1) is output. An example of the PWM output timing at this time is shown in Figure 13-6.

When the SYNn bit is set (to 1), on the other hand, a pulse width change is performed every PWM pulse cycle ($2^{8}/f_{PWMC}$). In this case, it takes a maximum of 2^{8} clocks (16 μs when f_{PWMC} = 16 MHz) until a pulse of a width corresponding to the data written in the PWMn in is output.

However, caution is required since, if the PWM pulse rewrite cycle is specified as every $2^{8}/f_{PWMC}$, (if the SYNn bit is set (to 1)), the obtained PWM pulse precision is between 8 bits and 12 bits, and is lower than when the PWM pulse rewrite cycle is specified as $2^{12}/f_{PWMC}$.

An example of the PWM output timing when the rewrite timing is $2^{8}/f_{PWMC}$ is shown in Figure 13-7.

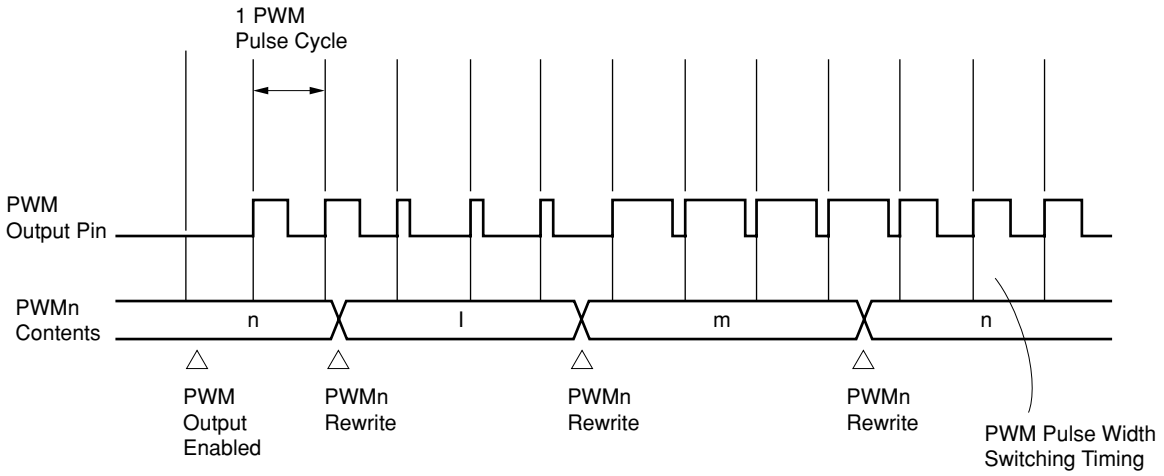
Figure 13-6 PWM Output Timing Example 1 (PWM Pulse Width Rewrite Cycle = $2^{12}/f_{PWMC}$)



Cautions 1. Pulse width rewriting is performed every PWM pulse cycle.

2. The PWM pulse precision is 12 bits.

Figure 13-7 PWM Output Timing Example 2 (PWM Pulse Width Rewrite Cycle = $2^8/f_{PWMC}$)



Cautions 1. Pulse width rewriting is performed every PWM pulse cycle.

2. The PWM pulse precision is between 8 and 12 bits.

Remark l, m, and n mean the PWMn contents.

13.4 CAUTION

A value between 0000H and 00FFH should not be set in the PWM modulo registers (PWMn: n = 0, 1). A value between 0100H and FFFFH should be set in the PWMn. Outputtable PWM signal duty values are 17/4096 to 4096/4096.

CHAPTER 14 A/D CONVERTER

The μPD784038 incorporates an analog/digital (A/D) converter with 8 multiplexed analog inputs (ANI0 to ANI7).

The successive approximation conversion method is used, and the conversion result is held in the 8-bit A/D conversion result register (ADCR). This allows fast, high-precision conversion to be performed (conversion time of 7.5 μs when f_{CLK} = 16 MHz and high-speed conversion is used).

There are two modes for starting A/D conversion, as follows:

- Hardware start : Conversion started by trigger input (INTP5).
- Software start : Conversion started in accordance with A/D converter mode register (ADM) bit setting.

After start-up, there are two operating modes, as follows:

- Scan mode : Multiple analog inputs are selected in order, and conversion data is obtained from all pins.
- Select mode: One pin is used as the analog input, and conversion values are obtained in succession.

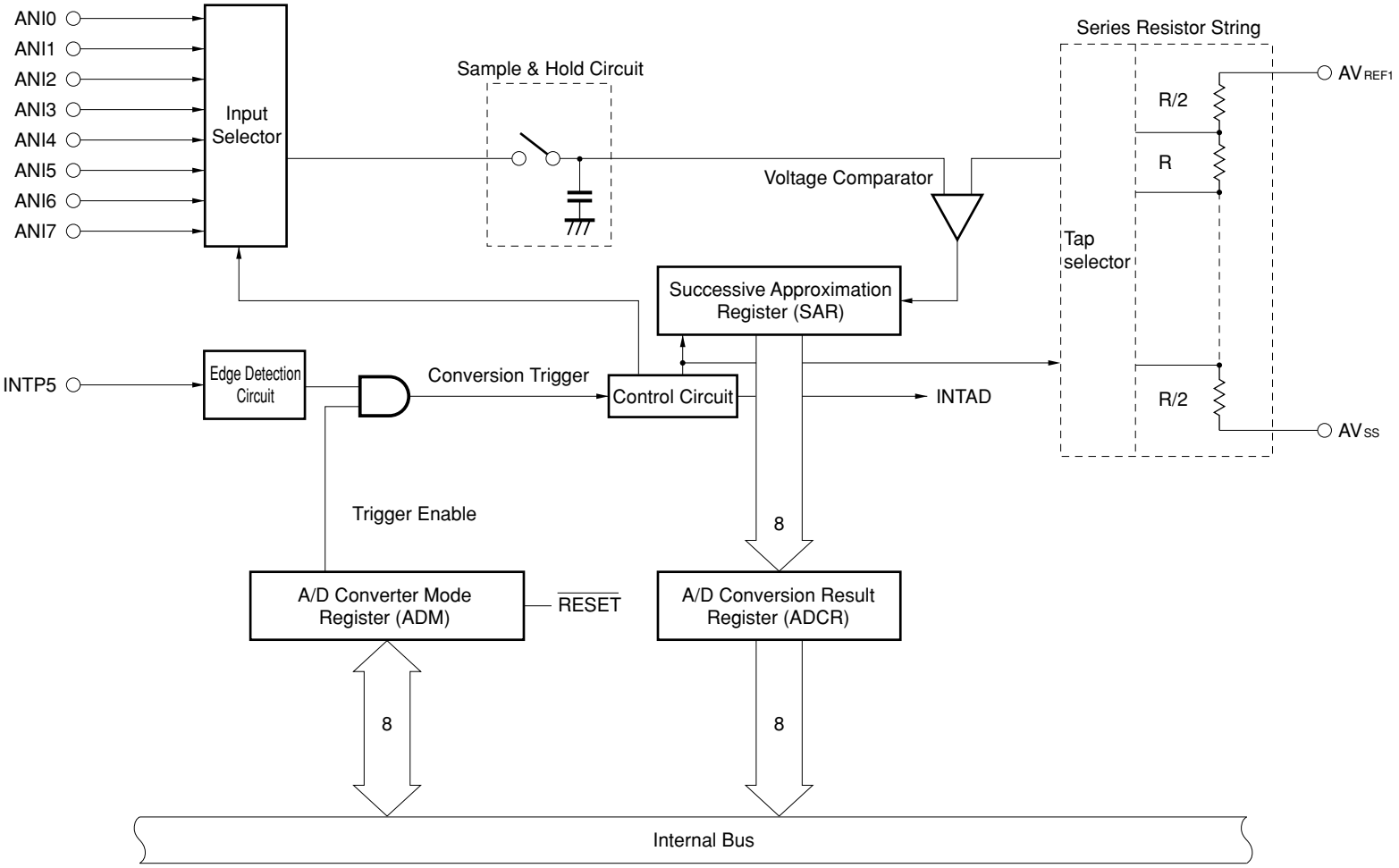
Stoppage of all the above modes and conversion operations is specified by the ADM register.

When the conversion result is transferred to the ADCR, an INTAD interrupt request is generated. This allows conversion values to be transferred to memory in succession by means of macro service.

14.1 CONFIGURATION

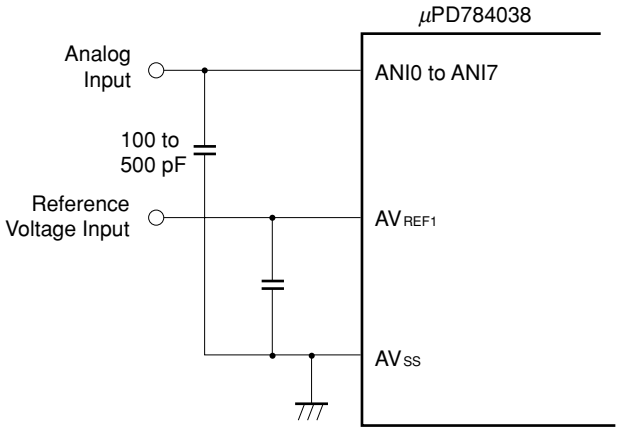
The A/D converter configuration is shown in Figure 14-1.

Figure 14-1 A/D Converter Block Diagram



Cautions 1. A capacitor should be connected between the analog input pins (ANI0 to ANI7) and AV_{SS} and between the reference voltage input pin (AV_{REF}) and AV_{SS} to prevent malfunction due to noise. Be sure to connect the capacitor as closely to ANI0 through ANI7 and AV_{REF1} as possible.

Figure 14-2 Example of Capacitor Connection on A/D Converter Pins



2. A voltage outside the range AV_{SS} to AV_{REF1} should not be applied to pins used as A/D converter input pins. See 14.5 CAUTIONS for details.

(1) Input circuit

The input circuit selects the analog input in accordance with the specification of the A/D converter mode register (ADM), and sends the analog input to the sample & hold circuit according to the operating mode,

(2) Sample & hold circuit

The sample & hold circuit samples the analog inputs arriving sequentially one by one and holds the analog input in the process of A/D conversion.

(3) Voltage comparator

The voltage comparator determines the voltage difference between the analog input and the series resistor string value tap.

(4) Series resistor string

The series resistor string is used to generate voltages that match the analog inputs.

The series resistor string is connected between the A/D converter reference voltage pin (AV_{REF1}) and the A/D converter GND pin (AV_{SS}). To provide 256 equal voltage steps between the two pins, it is made up of 255 equal resistors and two resistors with half that resistance value.

The series resistor string voltage tap is selected by a tap selector controlled by the SAR successive approximation register.

(5) SAR: Successive Approximation Register

The SAR is an 8-bit register in which the data for which the series resistor string voltage tap value matches the analog input voltage value is set bit by bit starting from the most significant bit (MSB).

When data has been set up to the least significant bit (LSB) of the SAR (when A/D conversion is completed), the SAR contents (conversion result) are stored in the A/D conversion result register (ADCR).

(6) ADCR: A/D Conversion Result Register

The ADCR is an 8-bit register that holds the A/D conversion result. The conversion result is loaded into this register from the successive approximation register (SAR) each time A/D conversion finishes.

The contents of this register approximation are undefined when $\overline{\text{RESET}}$ is input.

(7) Edge detection circuit

The edge detection circuit detects a valid edge from the interrupt request input pin (INTP5) input, and generates an external interrupt request signal (INTP5) and A/D conversion operation external trigger.

The INTP5 pin input valid edge is specified by external interrupt mode register 1 (INTM1) (see **Figure 21-2**). External trigger enabling/disabling is set by means of the A/D converter mode register (ADM) (see **14.2 A/D Converter Mode Register (ADM)**).

14.2 A/D CONVERTER MODE REGISTER (ADM)

ADM is an 8-bit register that controls A/D converter operations.

The ADM register can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. Its format is shown in Figure 14-3.

Bit 0 (MS) controls the operating mode.

Bits 1, 2 and 3 (ANI0, 1, 2) select the analog inputs for A/D conversion.

Bit 5 (SCMD) controls the A/D conversion operation in scan mode.

Bit 6 (TRG) enables external synchronization of the A/D conversion operation. If the TRG bit is set (to 1) when the CS bit is set (to 1), the conversion operation is initialized with each input of a valid edge as an external trigger to the INTP5 pin. When the TRG bit is cleared (to 0), the conversion operation is performed without regard to the INTP5 pin.

Bit 7 (CS) controls the A/D conversion operation. When the CS bit is set (to 1) the conversion operation is started, and when cleared (to 0), all conversion operations are stopped even if conversion is in progress. In this case, the A/D conversion result register (ADCR) is not updated and an INTAD interrupt request is not generated. Also, the power supply to the voltage comparator is stopped, and the A/D converter power consumption is reduced.

$\overline{\text{RESET}}$ input clears the ADM register to 00H.

Caution When the STOP mode or IDLE mode is used, the power consumption should be reduced by clearing (to 0) the CS bit before entering the STOP or IDLE mode. If the CS bit remains set (to 1), the conversion operation will be stopped by entering the STOP or IDLE mode, but the power supply to the voltage comparator will not be stopped, and therefore the A/D converter power consumption will not be reduced.

Figure 14-3 A/D Converter Mode Register (ADM) Format

	⑦	6	5	4	3	2	1	0	Address	After Reset	R/W
ADM	CS	TRG	SCMD	FR	ANIS2	ANIS1	ANIS0	MS	0FF68H	00H	R/W

ANIS2	ANIS1	ANIS0	MS	A/D Conversion Operating Mode Setting	
0	0	0	0	Scan mode (0/1)	ANI0 input scanned
0	0	1	0		Input ANI0 & ANI1 scanned
0	1	0	0		Input ANI0 to ANI2 scanned
0	1	1	0		Input ANI0 to ANI3 scanned
1	0	0	0		Input ANI0 to ANI4 scanned
1	0	1	0		Input ANI0 & ANI5 scanned
1	1	0	0		Input ANI0 to ANI6 scanned
1	1	1	0		Input ANI0 to ANI7 scanned
0	0	0	1	Select mode	ANI0 input selected
0	0	1	1		ANI1 input selected
0	1	0	1		ANI2 input selected
0	1	1	1		ANI3 input selected
1	0	0	1		ANI4 input selected
1	0	1	1		ANI5 input selected
1	1	0	1		ANI6 input selected
1	1	1	1		ANI7 input selected

FR	Conversion Speed Control (f_{CLK} = 16 MHz)	
0	242/f_{CLK} (15.125 μs)	Low-speed conversion
1	120/f_{CLK} (7.5 μs)	High-speed conversion

★

SCMD	MS	Scan Mode Selection
0	0	Scan mode 0 (no delay control)
1	0	Scan mode 1 (delay control)
0	1	Select mode
1	1	Setting prohibited

TRG	External Trigger Control
0	External trigger disabled
1	External trigger enabled

CS	A/D Conversion Operation Control
0	Stop A/D conversion operation
1	Start A/D conversion operation

Caution Once the A/D converter starts operating, conversion operations are performed repeatedly until the CS bit of the A/D converter mode register (ADM) is cleared (to 0). Therefore, a superfluous interrupt may be generated if ADM setting is performed after interrupt-related registers, etc., when A/D converter mode conversion, etc., is performed. The result of this superfluous interrupt is that the conversion result storage address appears to have been shifted when the scan mode is used. Also, when the select mode is used, the first conversion result appears to have been an abnormal value, such as the conversion result for the other channel. It is therefore recommended that A/D converter mode conversion be carried out using the following procedure.

<1> Write to the ADM (CS bit must be set (to 1))
<2> Interrupt request flag (ADIF) clearance (to 0)
<3> Interrupt mask flag or interrupt service mode flag setting

Operations <1> to <3> should not be divided by an interrupt or macro service. When scan mode 0 (no delay control) is used, in particular, you should ensure that the time between <1> and <2> is less than the time taken by one A/D conversion operation.

Alternatively, the following procedure is recommended.

<1> Stop the A/D conversion operation by clearing (to 0) the CS bit of the ADM.
<2> Interrupt request flag (ADIF) clearance (to 0).
<3> Interrupt mask flag or interrupt service mode flag setting
<4> Write to the ADM

14.3 OPERATION

14.3.1 Basic A/D Converter Operation

(1) A/D Conversion Operation procedure

A/D conversion is performed by means of the following procedure:

(a) Analog pin selection and operating mode specification are set with the A/D converter mode register (ADM).
(b) Bit 7 (CS) of the ADM is set (to 1), and A/D conversion is started.
(c) When conversion starts, the MSB (bit 7) of the successive approximation register (SAR) is set (to 1) automatically.
(d) When bit 7 of the SAR is set (to 1), the tap selector sets the series resistor string voltage tap to

$\frac{225}{512}$ AV_{REF1} ($\doteqdot$ 1/2 AV_{REF1}).

(e) The voltage difference between the series resistor string voltage tap and the analog input is determined by the voltage comparator. If the analog input is greater than (1/2) AV_{REF1}, the MSB of the SAR remains set (to 1), and if it is less than (1/2) AV_{REF1}, the MSB is cleared (to 0).
(f) Next, bit 6 of the SAR is set (to 1) automatically, and the next comparison is performed. Here, the series resistor string voltage tap is selected according to the value of bit 7 for which the result has already been set, as shown below.

- Bit 7 = 1 ..... $\frac{383}{512}$ $AV_{REF1} \doteqdot \frac{3}{4}$ AV_{REF1}
- Bit 7 = 0 ..... $\frac{127}{512}$ $AV_{REF1} \doteqdot \frac{1}{4}$ AV_{REF1}

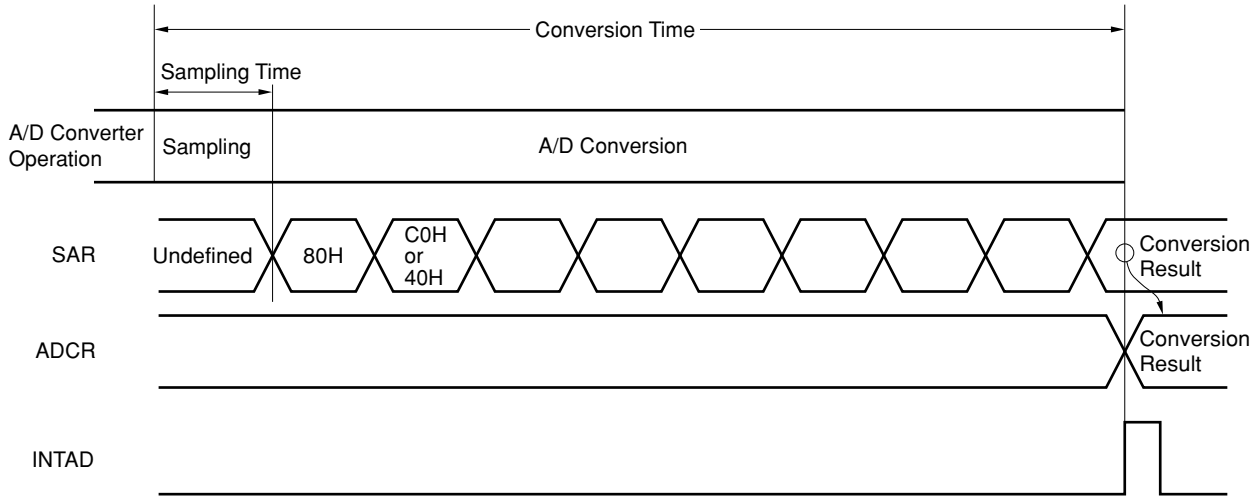
This voltage tap is compared with the analog input voltage, and bit 6 of the SAR is manipulated as follows according to the result:

- Analog input voltage ≥ voltage tap: Bit 6 = 1
- Analog input voltage < voltage tap: Bit 6 = 0

(g) The same kind of comparison is continued up to the LSB (bit 0) of the SAR (binary search method).

(h) When comparison of the 8 bits is completed, a valid digital result is left in the SAR, and that value is transferred to the A/D conversion result register (ADCR) and latched.
An A/D conversion operation end interrupt request (INTAD) can be generated at the same time.

Figure 14-4 Basic A/D Converter Operation



A/D conversion operations are performed successively until the CS bit is cleared (to 0) by software. If a write operation is performed on the ADM during an A/D conversion operation, the conversion operation is initialized, and if the CS bit is set (to 1), conversion will be started from the beginning.

The contents of the ADCR are undefined after $\overline{\text{RESET}}$ input.

(2) Input voltage and conversion result

The relationship between the analog input voltage input to an analog input pin (ANI0 to ANI7) and the A/D conversion result (value stored in ADCR) is shown by the following expression:

$$ADCR = INT\left(\frac{V_{IN}}{AV_{REF1}} \times 256 + 0.5\right)$$

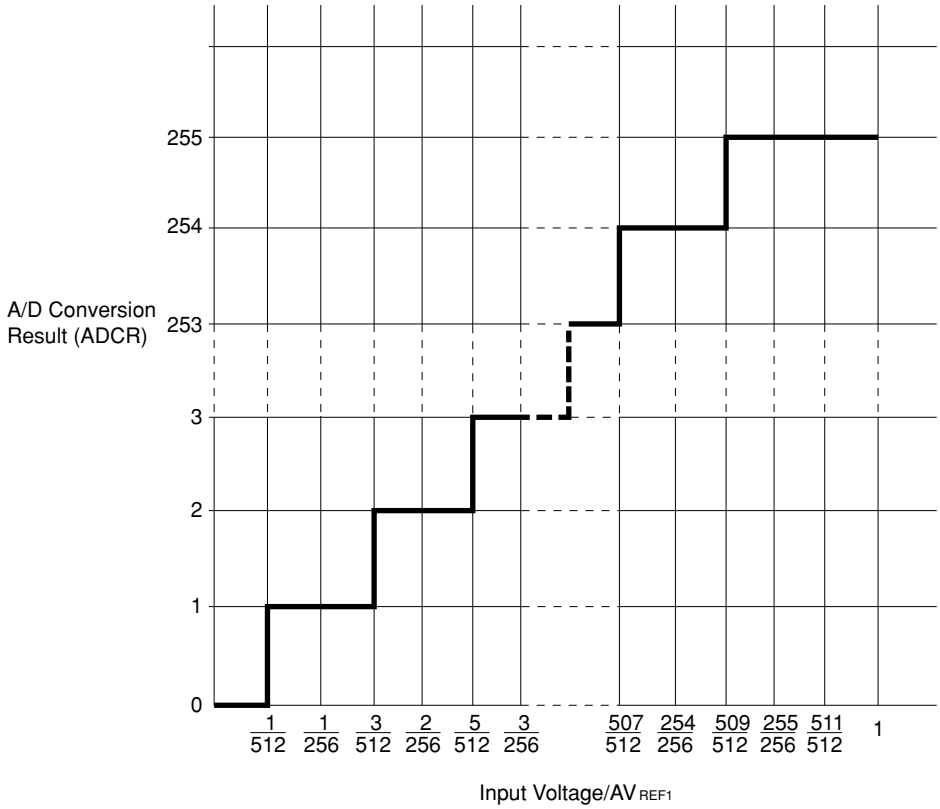
or

$$(ADCR - 0.5) \times \frac{AV_{REF1}}{256} \leq V_{IN} < (ADCR + 0.5) \times \frac{AV_{REF1}}{256}$$

Remark INT( ) : Function that returns the integer part of the value in ( )
V_{IN} : Analog input voltage
AV_{REF1} : AV_{REF1} pin voltage
ADCR : ADCR value

Figure 14-5 shows the relationship between the analog input voltage and the A/D conversion result in graphic form.

Figure 14-5 Relationship Between Analog Input Voltage and A/D Conversion Result



(3) A/D conversion time

The A/D conversion time is determined by the system clock frequency (f_{CLK}) and the FR bit of the A/D converter mode register (ADM).

The A/D conversion time includes the entire time required for one A/D conversion operation, and the sampling time is also included in the A/D conversion time.

These values are shown in Table 14-2.

Table 14-1 A/D Conversion Time

System Clock (f_{CLK}) Range	FR Bit	Conversion Time	Sampling Time
0.25 MHz ≤ f_{CLK} ≤ 16 MHz	0	180/f_{CLK} (11.25 μs to 90 μs)	36/f_{CLK} (2.25 μs to 18 μs)
0.25 MHz ≤ f_{CLK} ≤ 16 MHz	1	120/f_{CLK} (7.5 μs to 60 μs)	24/f_{CLK} (1.5 μs to 12 μs)

(4) A/D converter operating modes

There are two A/D converter operating modes, scan mode and select mode. These modes are selected according to the setting of bit 0 (MS) of the A/D converter mode register (ADM). In addition, scan mode 0 or 1 can be selected by bit 5 (SCMD) of the ADM.

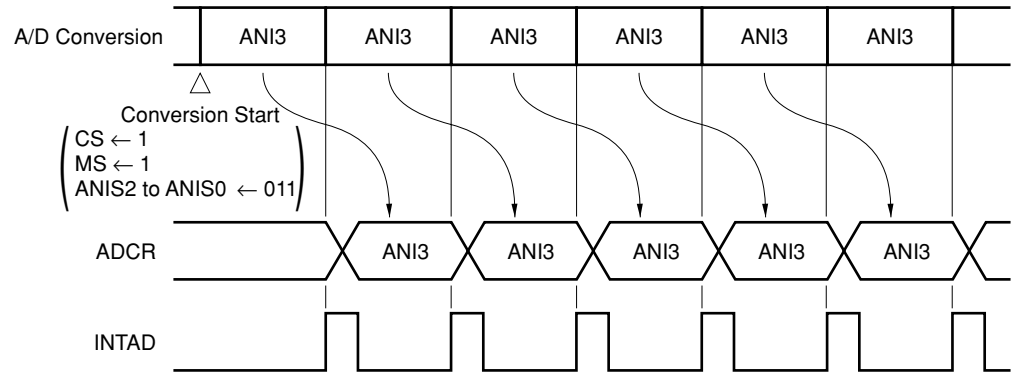
Operation in either mode continues until the ADM is rewritten.

14.3.2 Select Mode

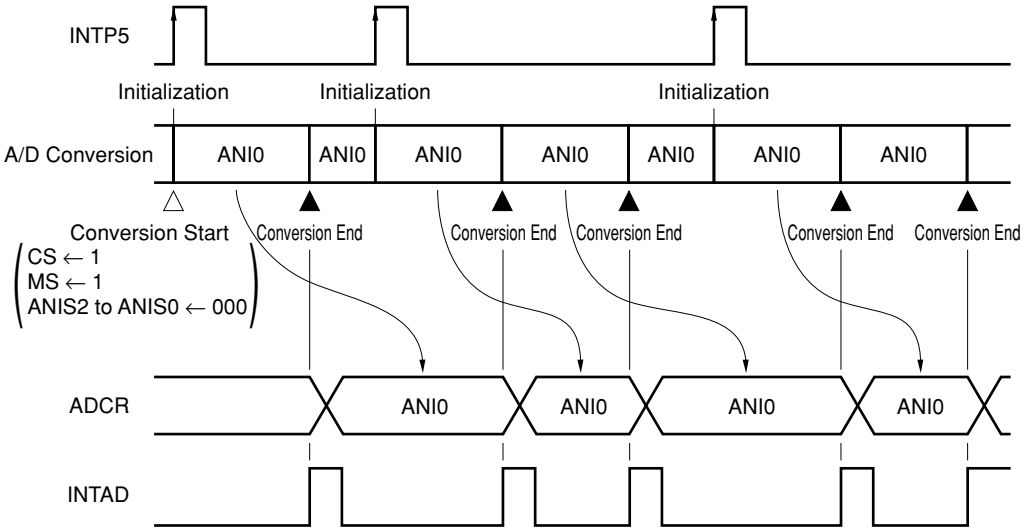
One analog input is specified by bits 1 to 3 (ANIS0 to ANIS2) of the A/D converter mode register (ADM), and A/D conversion of the specified analog input pin is started. The conversion result is stored in the A/D conversion result register (ADCR). An A/D conversion end interrupt request (INTAD) is generated at the end of each conversion operation.

Figure 14-6 Select Mode Operation Timing

(a) TRG bit ← 0



(b) TRG bit ← 1



14.3.3 Scan Mode

Two scan modes, 1 and 0, are available. In scan mode 0, delay control that takes delay in reading the A/D conversion result by the CPU into consideration can be performed. In scan mode 1, no delay control is performed but the A/D conversion interval is fixed.

Generally, use of scan mode 1 is recommended.

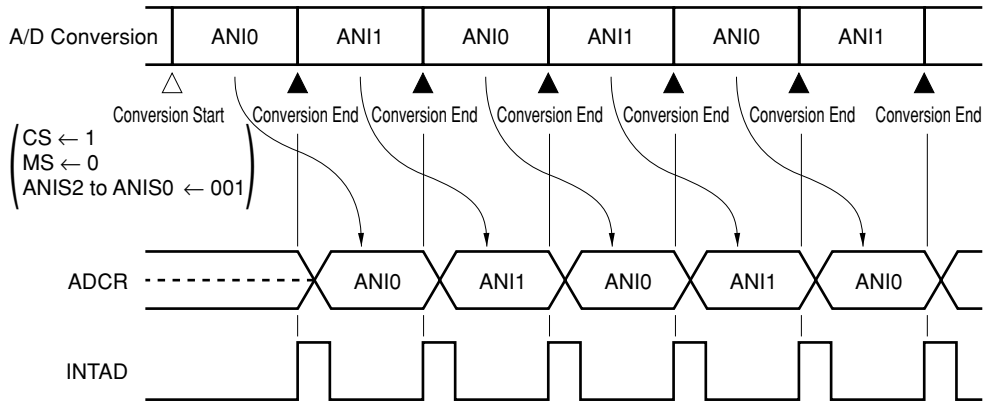
(1) Scan mode 0 (bit 5 (SCMD) of A/D converter mode register (ADM) = 0)

Input from the analog input pins specified by bits 1 to 3 (ANIS0 to ANIS2) of the ADM is selected and converted in order.

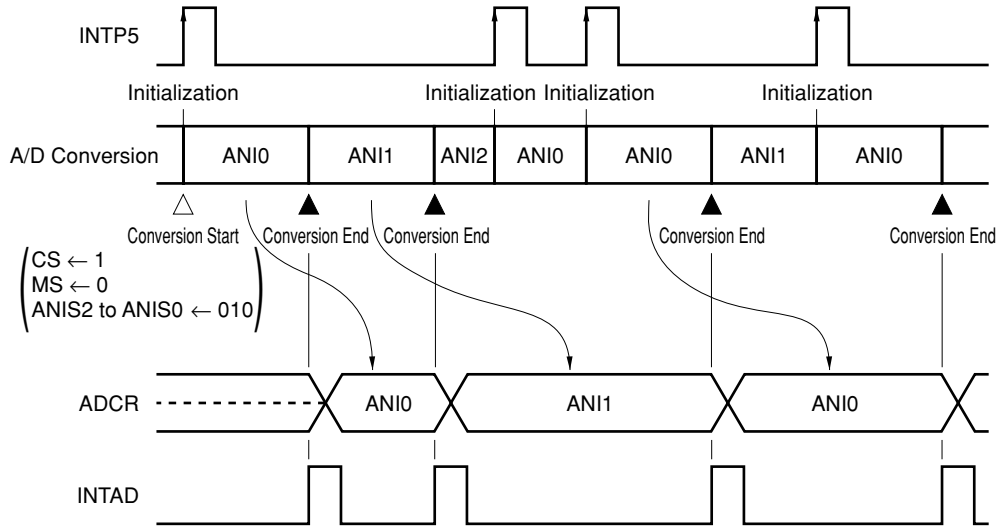
For example, if ANIS2 to ANIS0 of the ADM = 001, ANI0 and ANI1 will be scanned repeatedly (ANI0 → ANI1 → ANI0 → ANI1 → ...). In the scan mode, at the end of the conversion operation for each input the conversion value is stored in the A/D conversion result register (ADCR) and an A/D conversion end interrupt request (INTAD) is generated.

Figure 14-7 Scan Mode 0 Operation Timing

(a) TRG bit ← 0



(b) TRG bit ← 1

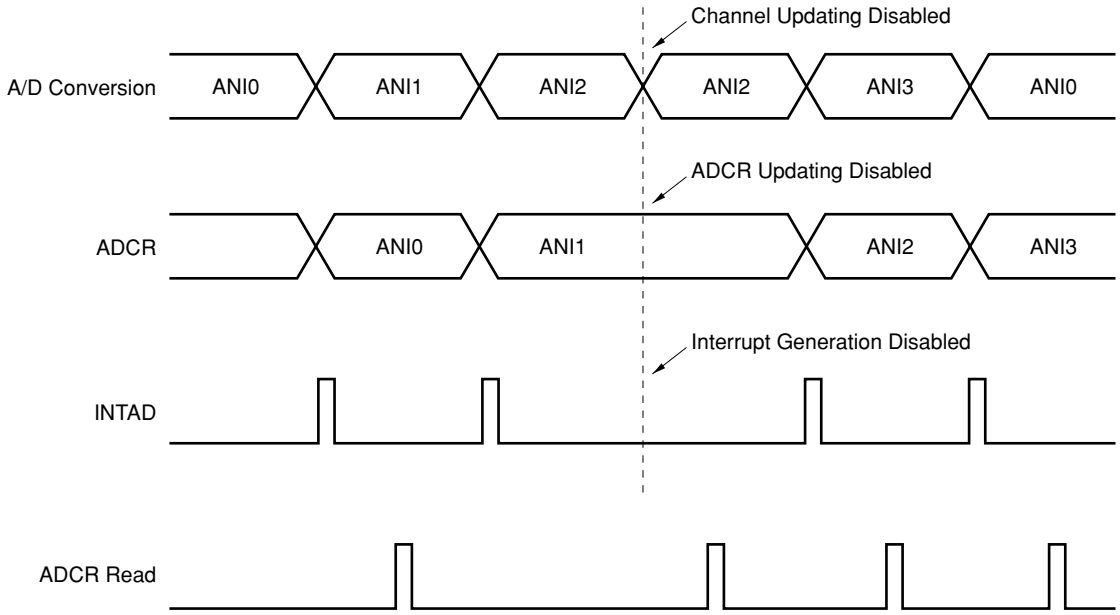


(2) Scan mode 1 (bit 5 (SCMD) of A/D converter mode register (ADM) = 1)

When bit 5 of the ADM is set (to 1), the analog input pins specified by bits 1 to 3 (ANIS0 to ANIS2) are selected, and subjected to conversion, in order. If an A/D conversion result register (ADCR) read is not performed by the CPU by the end of the next A/D conversion after A/D conversion end (INTAD) generation, conversion is restarted without performing INTAD generation, ADCR updating or channel updating (see **Figure 14-8**).

If an ADCR read is performed by the CPU before the end of the next A/D conversion, the same operation as in scan mode 0 is performed.

Figure 14-8 Scan Mode 1 Operation Timing



14.3.4 A/D Conversion Operation Start by Software

An A/D conversion operation start by software is performed by writing a value to the A/D converter mode register (ADM) that sets the TRG bit of the ADM register to 0 and the CS bit to 1.

If a value is written to the ADM during an A/D conversion operation (CS bit = 1) such that the TRG bit is set to 0 and the CS bit to 1 again, the A/D conversion operation being performed at that time is suspended, and A/D conversion is started immediately in accordance with the written value.

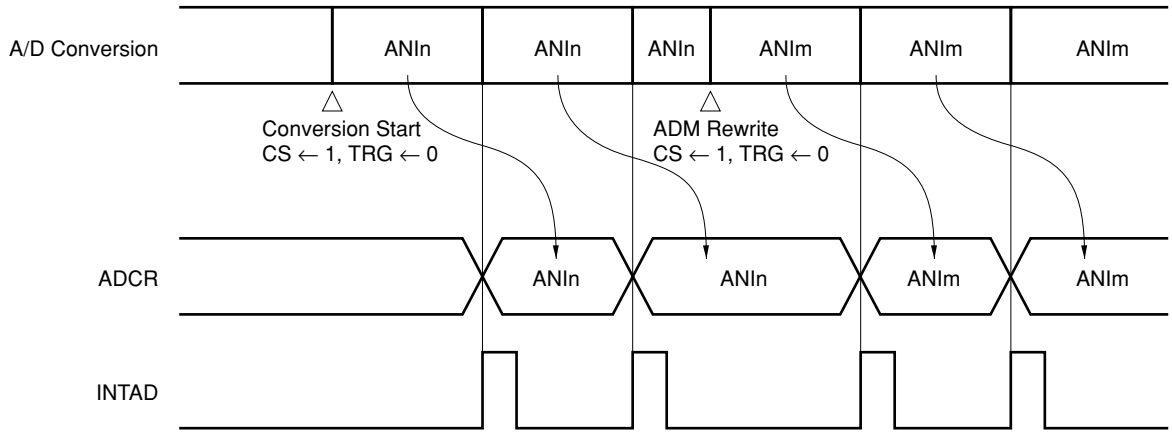
Once A/D conversion operation is started, as soon as one A/D conversion operation ends the next A/D conversion operation is started in accordance with the operating mode set by the ADM, and conversion operations continue repeatedly until an instruction that writes to the ADM is executed.

When A/D conversion operation is started by software (TRG bit = 0), INTP5 pin (P26 pin) input does not affect the A/D conversion operation.

(1) Select mode A/D conversion operation

An A/D conversion operation is started on the analog input pin set by the A/D converter mode register (ADM). As soon as the A/D conversion operation ends, another A/D conversion operation is performed on the same analog input pin. An A/D conversion end interrupt request (INTAD) is generated at the end of each A/D conversion operation.

Figure 14-9 Software Start Select Mode A/D Conversion Operation

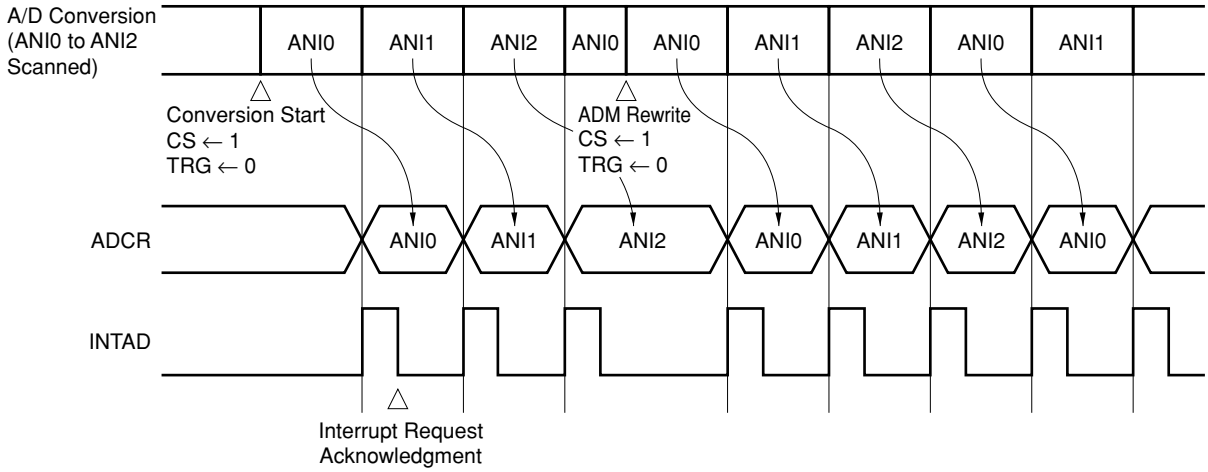


Remark n = 0, 1, …, 7
m = 0, 1, …, 7

(2) Scan mode A/D conversion operation

When conversion operation is started, an A/D conversion operation is started on the ANI0 pin input. When the A/D conversion operation ends, an A/D conversion operation is started on the next analog input pin. An A/D conversion end interrupt request (INTAD) is generated at the end of each A/D conversion operation.

Figure 14-10 Software Start Scan Mode A/D Conversion Operation



14.3.5 A/D Conversion Operation Start by Hardware

An A/D conversion operation start by hardware is made possible by setting both the TRG bit and the CS bit of the A/D converter mode register (ADM) to 1. When the TRG bit and the CS bit of the ADM are both set to 1, external signals are placed in the standby state, and an A/D conversion operation is started when a valid edge is input to the INTP5 pin (P26 pin).

If another valid edge is input to the INTP5 pin after the A/D conversion operation has been started by a valid edge input to the INTP5 pin, the A/D conversion operation being performed at that time is suspended, and A/D conversion is performed from the beginning in accordance with the contents set in the ADM.

If a value is written to the ADM during an A/D conversion operation (CS bit = 1) such that the TRG bit and CS bit are both set to 1 again, the A/D conversion operation being performed at that time is suspended (the standby state is also suspended), and a standby state is entered in which the A/D converter waits for input of a valid edge to the INTP5 pin in the A/D conversion operation mode in accordance with the written value, and a conversion operation is started when a valid edge is input.

Use of this function allows A/D conversion operations to be synchronized with external signals. Once A/D conversion operation is started, as soon as one A/D conversion operation ends the next A/D conversion operation is started in accordance with the operating mode set by the ADM (the A/D converter does not wait for INTP5 pin input), and conversion operations continue repeatedly until an instruction that writes to the ADM is executed, or a valid edge is input to the INTP5 pin.

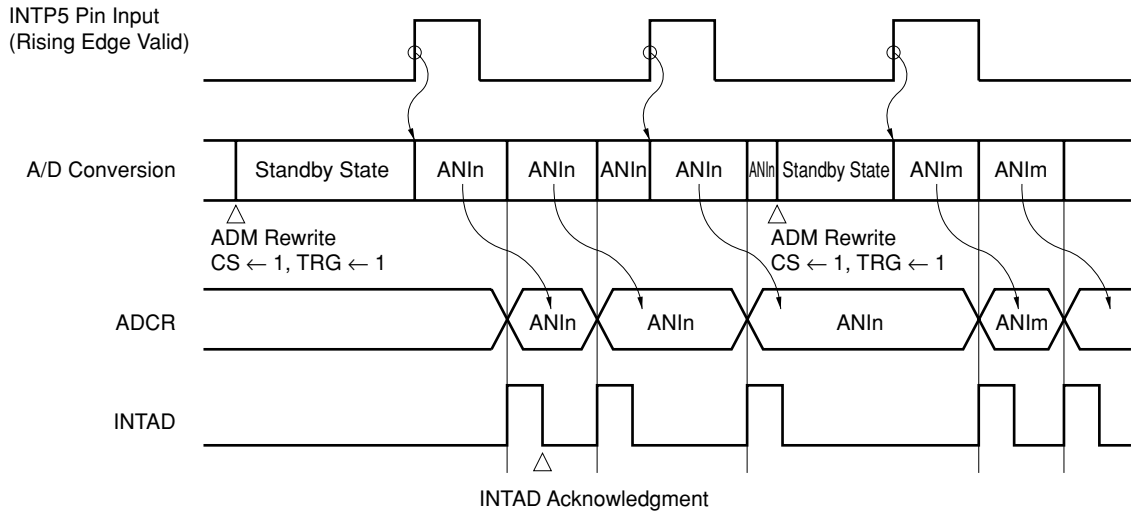
Caution Approximately 10 μs is required from the time a valid edge is input to the INTP5 pin until the A/D conversion operation is actually started. This delay must be taken into account in the design stage. See CHAPTER 21 EDGE DETECTION FUNCTION for details of the edge detection function.

(1) Select mode A/D conversion operation

An A/D conversion operation is started on the analog input pin set by the A/D converter mode register (ADM). As soon as the A/D conversion operation ends, another A/D conversion operation is performed on the same analog input pin. An A/D conversion end interrupt request (INTAD) is generated at the end of each A/D conversion operation.

If a valid edge is input to the INTP5 pin during an A/D conversion operation, the A/D conversion operation being performed at that time is suspended, and a new A/D conversion operation is started.

Figure 14-11 Hardware Start Select Mode A/D Conversion Operation



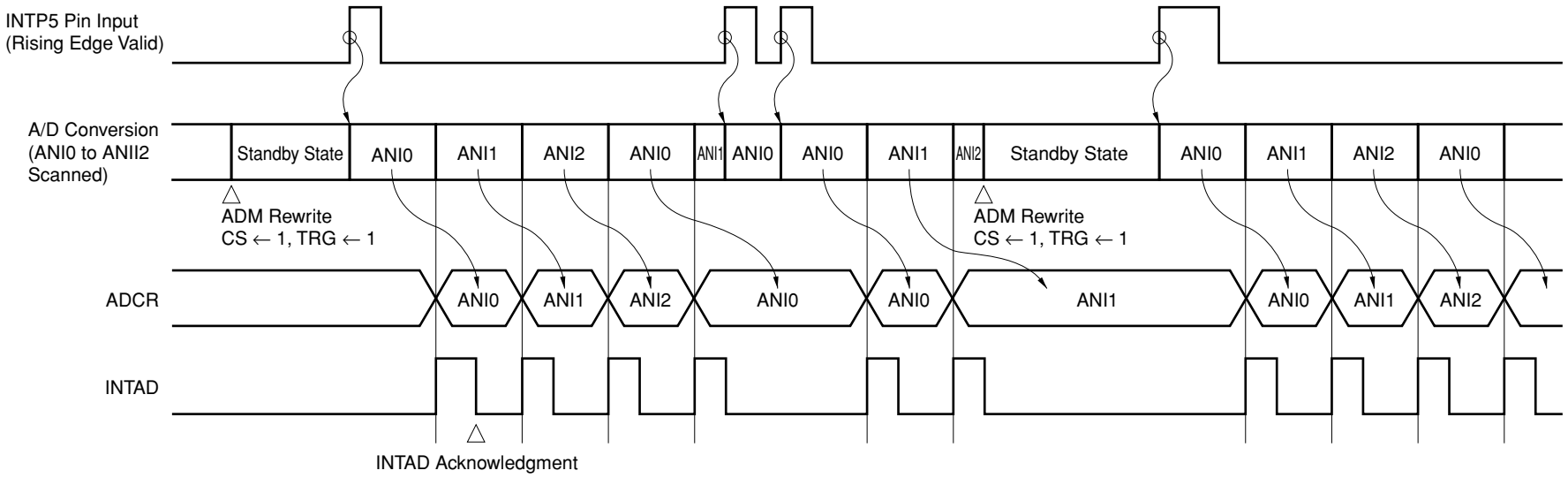
Remark n = 0, 1, …, 7
m = 0, 1, …, 7

(2) Scan mode A/D conversion operation

When conversion operation is started, an A/D conversion operation is started on the ANI0 pin input. When the A/D conversion operation ends, an A/D conversion operation is started on the next analog input pin. An A/D conversion end interrupt request (INTAD) is generated at the end of each A/D conversion operation.

If a valid edge is input to the INTP5 pin during an A/D conversion operation, the A/D conversion operation being performed at that time is suspended, and a new A/D conversion operation is started on the ANI0 pin input.

Figure 14-12 Hardware Start Scan Mode A/D Conversion Operation



14.4 EXTERNAL CIRCUIT OF A/D CONVERTER

The A/D converter is provided with a sample & hold circuit to stabilize its conversion operation. This sample & hold circuit outputs sampling noise during sampling immediately after an A/D conversion channel has been changed.

To absorb this sampling noise, an external capacitor must be connected. If the impedance of the signal source is high, an error may occur in the conversion result due to the sampling noise. Especially when the scan mode is used, the impedance of the signal source must be kept low because the channel whose signal is to be converted changes one after another.

One way to absorb the sampling noise is to increase the capacitance of the capacitor. However, if the capacitance is increased too much, the sampling noise is accumulated. Therefore, the most effective way is to reduce the resistance component.

14.5 CAUTIONS

(1) Range of voltages applied to analog input pins

The following must be noted concerning A/D converter analog input pins ANI0 to ANI7 (P70 to P77).

- A voltage outside the range AV_{SS} to AV_{REF1} should not be applied to pins subject to A/D conversion during an A/D conversion operation.

If this restriction is not observed, the μPD784038 may be damaged.

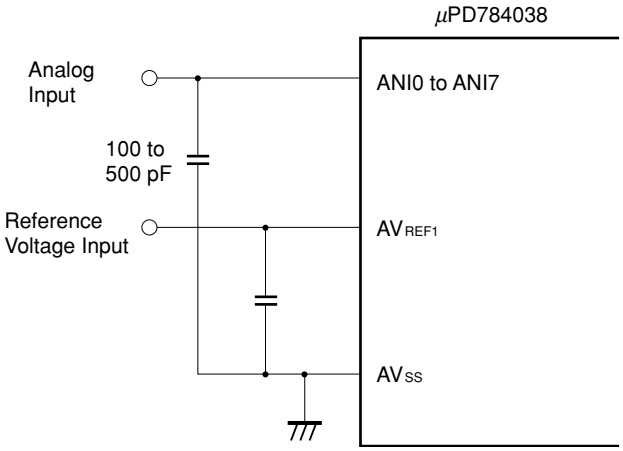
(2) Hardware start A/D conversion

Approximately 10 μs is required from the time a valid edge is input to the INTP5 pin until the A/D conversion operation is actually started. This delay must be taken into account in the design stage. See **CHAPTER 21 EDGE DETECTION FUNCTION** for details of the edge detection function.

(3) Connecting capacitor to analog input pins

A capacitor should be connected between the analog input pins (ANI0 to ANI7) and AVSS and between the reference voltage input pin (AV_{REF1}) and AV_{SS} to prevent malfunction due to noise.

Figure 14-13 Example of Capacitor Connection on A/D Converter Pins



(4) When the STOP mode or IDLE mode is used, the power consumption should be reduced by clearing (to 0) the CS bit before entering the STOP or IDLE mode. If the CS bit remains set (to 1), the conversion operation will be stopped by entering the STOP or IDLE mode, but the power supply to the voltage comparator will not be stopped, and therefore the A/D converter power consumption will not be reduced.

(5) Once the A/D converter starts operating, conversion operations are performed repeatedly until the CS bit of the A/D converter mode (ADM) is cleared (to 0). Therefore, a superfluous interrupt may be generated if ADM setting is performed after interrupt-related registers, etc., are set when A/D converter mode conversion, etc., is performed. The result of this superfluous interrupt is that the conversion result storage address appears to have been shifted when the scan mode is used. Also, when the select mode is used, the first conversion result appears to have been an abnormal value, such as the conversion result for the other channel. It is therefore recommended that A/D converter mode conversion be carried out using the following procedure.

<1> Write to the ADM (CS bit must be set (to 1))
<2> Interrupt request flag (ADIF) clearance (to 0)
<3> Interrupt mask flag or interrupt service mode flag setting

Operations <1> to <3> should not be divided by an interrupt or macro service. When scan mode 0 (no delay control) is used, in particular, you should ensure that the time between <1> and <2> is less than the time taken by one A/D conversion operation.

Alternatively, the following procedure is recommended.

<1> Stop the A/D conversion operation by clearing (to 0) the CS bit of the ADM.
<2> Interrupt request flag (ADIF) clearance (to 0).
<3> Interrupt mask flag or interrupt service mode flag setting
<4> Write to the ADM

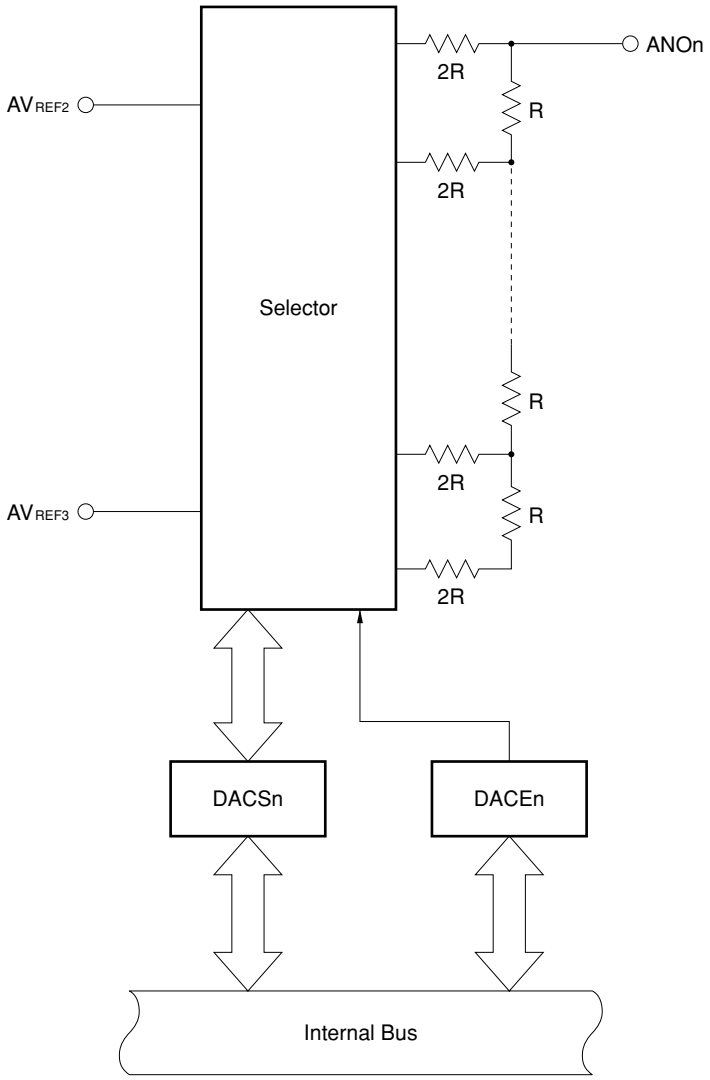
CHAPTER 15 D/A CONVERTER

The μPD784038 incorporates an 8-bit resolution voltage output type digital/analog (D/A) converter, which uses the R-2R resistor ladder type.

15.1 CONFIGURATION

The D/A converter block diagram is shown in Figure 15-1.

Figure 15-1 D/A Converter Block Diagram



Remark n = 0, 1

- **D/A conversion value setting registers (DACS0, DACS1)**

  These registers are used to set the voltage values to be output to the ANOn pins (n = 0, 1). The voltage value output to the ANOn pin is given by the following expression:

$$ANOn = \frac{AV_{REF2} - AV_{REF3}}{256} \times DACSn + AV_{REF3} \text{ [V]}$$

  $\overline{\text{RESET}}$ input initializes these registers to 00H.

15.2 D/A CONVERTER MODE REGISTER (DAM)

DAM is an 8-bit register that controls D/A converter operations. The DAM register can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. DMA format is shown in Figure 15-2. $\overline{\text{RESET}}$ input sets the DAM register to 03H, enabling D/A conversion output for both channels.

Figure 15-2 D/A Converter Mode Register (DAM) Format

	7	6	5	4	3	2	(1)	(0)	Address	After Reset	R/W
DAM	0	0	0	0	0	0	DACE1	DACE0	0FF62H	03H	R/W

DACE1	DACE0	D/A Converter Operation
0	0	Standby mode ANO0 & ANO1 pins both output high impedance
0	1	ANO0 pin output enabled ANO1 pin output high impedance
1	0	ANO0 pin output high impedance ANO1 pin output enabled
1	1	ANO0 & ANO1 pins both output enabled

15.3 D/A CONVERTER OPERATION

15.3.1 Basic Operation

When the value to be output is written to the D/A conversion value setting register (DACSn, n = 0, 1) while the D/A conversion enable bit (DACEn, n = 0, 1) of the D/A converter mode register (DAM) is set (to 1), an analog voltage corresponding to the value written is output from the ANOn pin (n = 0, 1). The output voltage is retained until the next value is written to the DACSn.

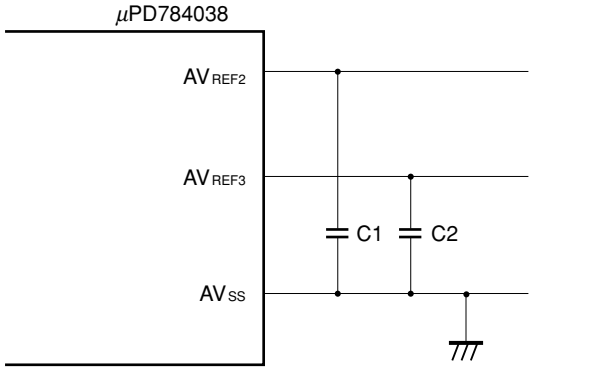
The voltage output from the ANOn pin is determined by the following expression:

$$ANOn = \frac{AV_{REF2} - AV_{REF3}}{256} \times DACSn + AV_{REF3} \text{ [V]}$$

While the $\overline{\text{RESET}}$ input is low, ANOn is in the output high impedance state, and the DACSn is initialized to 00H. After $\overline{\text{RESET}}$ release, the same level as the AVREF3 pin is output from the ANOn pin.

Connect capacitors between the reference voltage input pins (AV_{REF2} and AV_{REF3}) and AV_{SS} to stabilize the operation of the D/A converter.

Figure 15-3 Example of Connecting Capacitors to Reference Voltage Input Pins of D/A Converter



C1 = C2 = 100 to 500 pF

15.3.2 D/A Converter Standby Operation

When the D/A conversion enable bit (DACEn, n = 0, 1) of the D/A converter mode register (DAM) is cleared (to 0), the ANOn pin (n = 0, 1) is set to the output high impedance state.

When both DACEn bits are cleared (to 0), the D/A converter enters standby mode, enabling the power consumption to be reduced.

Clear both DACEn bits (to 0) when it is especially required to reduce the power consumption such as in STOP mode.

15.4 CAUTIONS

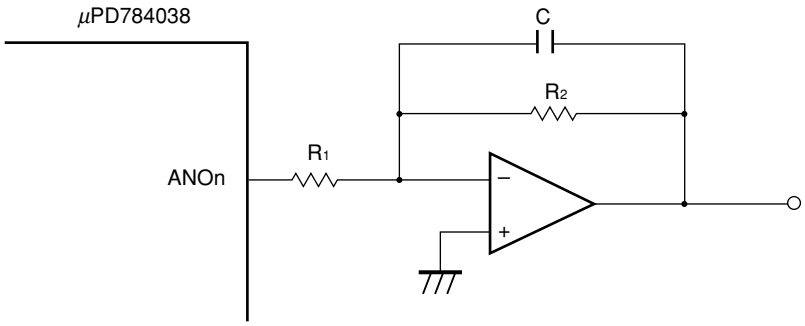
(1) As the D/A converter output impedance is high, a current cannot be taken from the ANOn pin (n = 0, 1). If the load input impedance is low, a buffer amplifier should be inserted between the load and the ANOn pin. Also, the wiring to the buffer amp and load should be kept as short as possible (since the output impedance is high). If the wiring is long, measures such as enclosure with a ground pattern should be taken.

(2) As the D/A converter output voltage varies in steps, the signal output by the D/A converter should generally be passed through a low-pass filter before use.

(3) The D/A converter incorporated in the μPD784038 is in the output high impedance state while $\overline{\text{RESET}}$ is low. The design should therefore make provision for high impedance input in the load side circuitry.

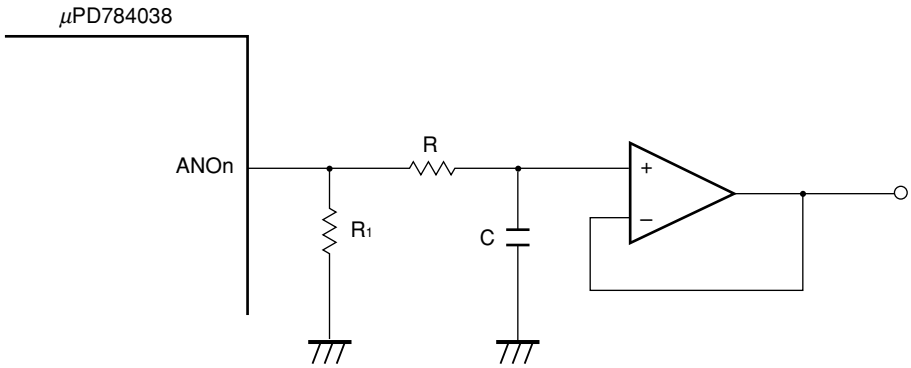
Figure 15-4 Example of Buffer Amp Insertion

(a) Inverting amp



• Buffer amplifier input impedance = R_1

(b) Voltage follower



• Buffer amplifier input impedance = R_1
• If R_1 were omitted, the output would be undefined when $\overline{\text{RESET}}$ is low.

(4) Since the D/A converter output is at the same level as the AV_{REF3} pin after reset release, the design should allow for AV_{REF3} pin level output after reset release.

CHAPTER 16 OUTLINE OF SERIAL INTERFACE

The μPD784038 Subseries is provided with three independent serial interface channels. Therefore, communication with an external system and local communication within the system can be simultaneously executed by using these three channels.

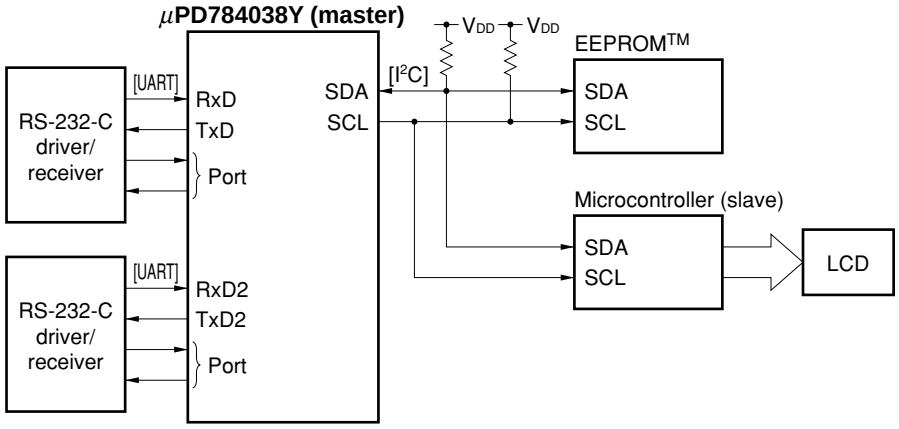
- Asynchronous serial interface (UART)/3-wire serial I/O (IOE) × 2 channels
  → Refer to **CHAPTER 17**.

- Clocked serial interface (CSI) × 1 channel
  - 3-wire serial I/O mode (MSB/LSB first)
    → Refer to **CHAPTER 18**.
  - 2-wire serial I/O mode (MSB first)
    → Refer to **CHAPTER 18**.
  - I^2C bus mode (MSB first) (μPD784038Y Subseries only)
    → Refer to **CHAPTER 19**.

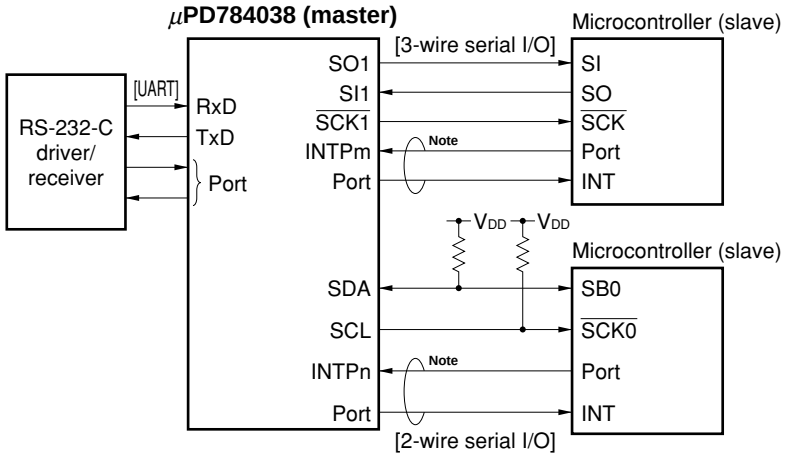
Figure 16-1 shows an example of the serial interface.

Figure 16-1 Example of Serial Interface

(1) UART + I²C



(2) UART + 3-wire serial I/O + 2-wire serial I/O



Note Handshake line

CHAPTER 17 ASYNCHRONOUS SERIAL INTERFACE/3-WIRE SERIAL I/O

The μPD784038 incorporates two serial interface channels for which asynchronous serial interface (UART) mode or 3-wire serial I/O (IOE) mode can be selected.

The two UART/IOE channels have completely identical functions. In this chapter, therefore, unless stated otherwise, UART/IOE1 will be described as representative of both UART/IOEs. When used as UART2/IOE2, the UART/IOE1 register names, bit names and pin names should be read as their UART2/IOE2 equivalents as shown in Table 17-1.

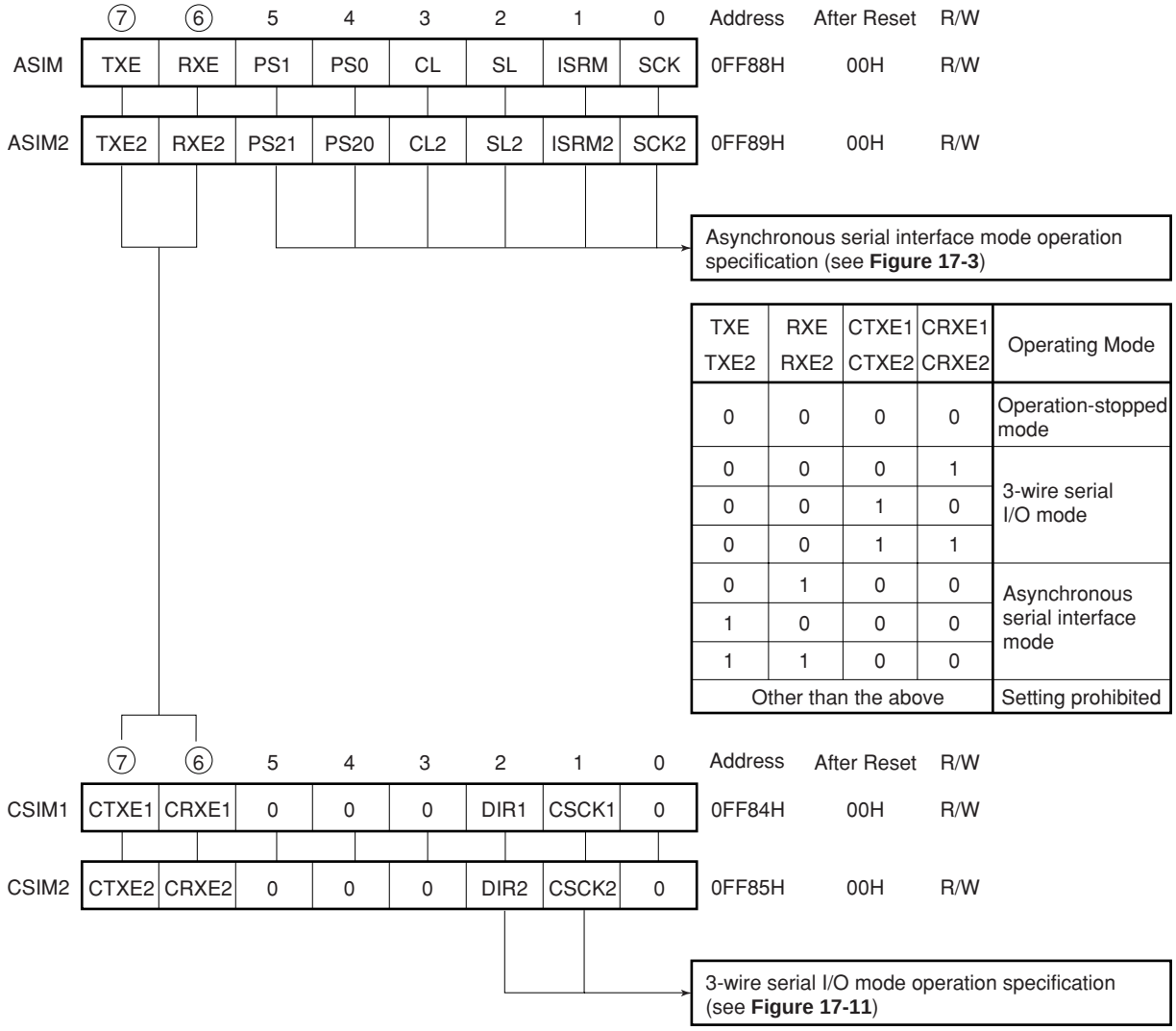
Table 17-1 Differences Between UART/IOE1 and UART2/IOE2 Names

Item	UART/IOE1	UART2/IOE2
Pin names	P25/ASCK/$\overline{SCK1}$, P30/RxD/SI1, P31/TxD/SO1	P12/ASCK2/$\overline{SCK2}$, P13/RxD2/SI2, P14/TxD2/SO2
Asynchronous serial interface mode register	ASIM	ASIM2
Asynchronous serial interface mode register bit names	TXE, RXE, PS1, PS0, CL, SL, ISRM, SCK	TXE2, RXE2, PS21, PS20, CL2, SL2, ISRM2, SCK2
Asynchronous serial interface status register	ASIS	ASIS2
Asynchronous serial interface status register bit names	PE, FE, OVE	PE2, FE2, OVE2
Clocked serial interface mode register	CSIM1	CSIM2
Clocked serial interface mode register bit names	CTXE1, CRXE1, DIR1, CSCK1	CTXE2, CRXE2, DIR2, CSCK2
Baud rate generator control register	BRGC	BRGC2
Baud rate generator control register bit names	TPS0 to TPS3, MDL0 to MDL3	TPS20 to TPS23, MDL20 to MDL23
Interrupt request names	INTSR/ITCSI1, INTSER, INTST	INTSR2/INTCSI2, INTSER2, INTST2
Interrupt control registers and bit names used in this chapter	SRIC, CSIIC1, SERIC, STIC, SRIF, CSIIF1, SERIF, STIF	SRIC2, CSIIC2, SERIC2, STIC2, SRIF2, CSIIF2, SERIF2, STIF2

17.1 SWITCHING BETWEEN ASYNCHRONOUS SERIAL INTERFACE MODE AND 3-WIRE SERIAL I/O MODE

The asynchronous serial interface mode and 3-wire serial I/O mode cannot be used simultaneously. Switching between these modes is performed in accordance with the settings of the asynchronous serial interface mode register (ASIM/ASIM2) and the clocked serial interface mode register (CSIM1/CSIM2) as shown in Figure 17-1.

Figure 17-1 Switching Between Asynchronous Serial Interface Mode and 3-Wire Serial I/O Mode



17.2 ASYNCHRONOUS SERIAL INTERFACE MODE

A UART (Universal Asynchronous Receiver Transmitter) is incorporated as the asynchronous serial interface. With this method, one byte of data is transmitted following a start bit, and full-duplex operation is possible.

A baud rate generator is incorporated, enabling communication to be performed at any of a wide range of baud rates.

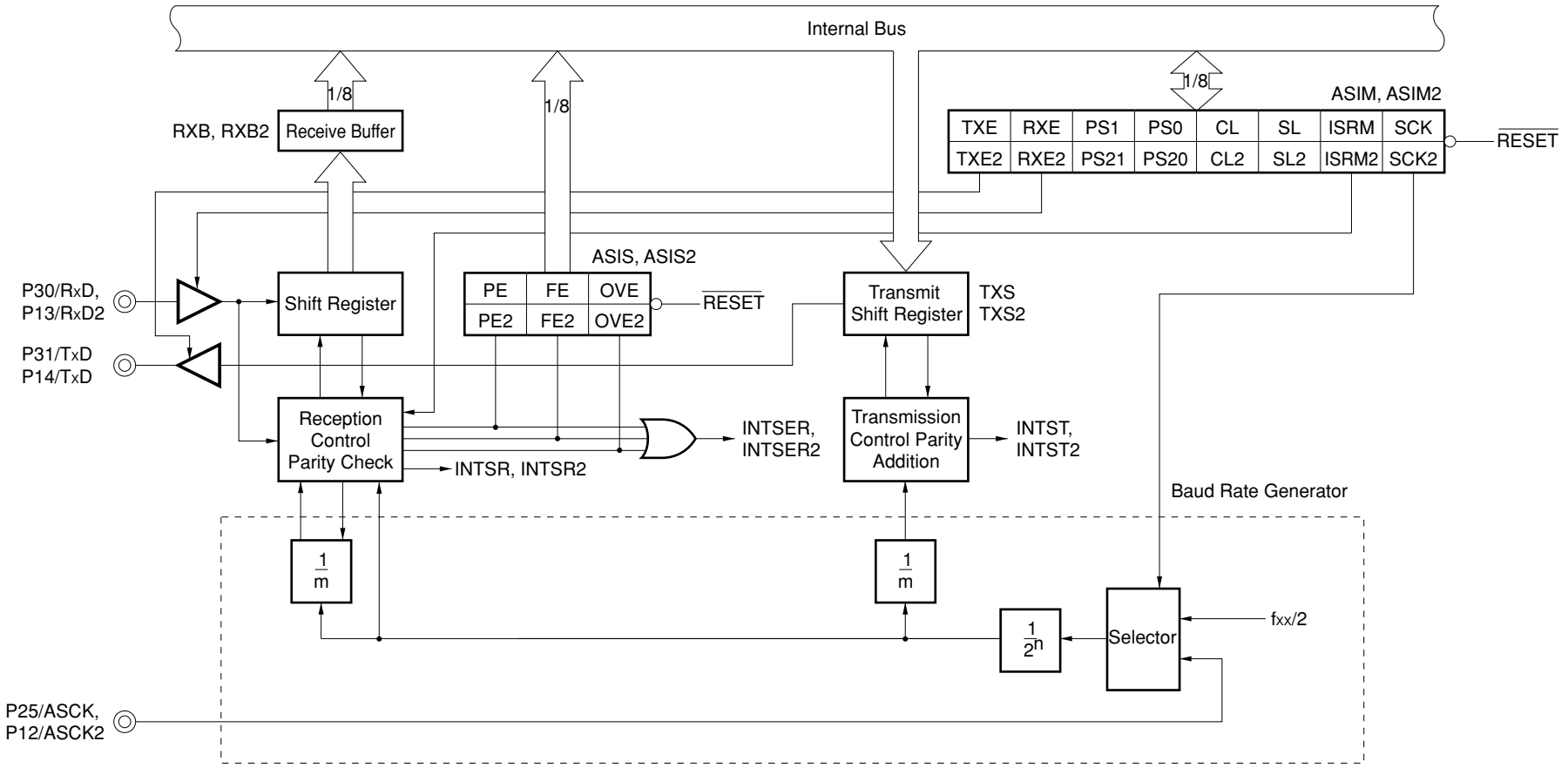
Also, the baud rate can be defined by scaling the clock input to the ASCK pin.

17.2.1 Configuration in Asynchronous Serial Interface Mode

The block diagram of the asynchronous serial interface is described in Figure 17-2.

See **17.4 Baud Rate Generator** for details of the baud rate generator.

Figure 17-2 Asynchronous Serial Interface Block Diagram



(1) Receive buffer (RXB/RXB2)

This is the register that holds the receive data. Each time one byte of data is received, the receive data is transferred from the shift register.

If a 7-bit data length is specified, receive data is transferred to bits 0 to 6 of RXB/RXB2, and the MSB of RXB/RXB2 is always “0”.

RXB/RXB2 can be read only with an 8-bit manipulation instruction. The contents of RXB/RXB2 are undefined after $\overline{\text{RESET}}$ input.

(2) Transmit shift register (TXS/TXS2)

This is the register in which the data to be transmitted is set. Data written to the TXS/TXS2 is transmitted as serial data.

If a 7-bit data length is specified, bits 0 to 6 of the data written in the TXS/TXS2 are treated as transmit data. A transmit operation starts when a write to the TXS/TXS2 is performed. The TXS/TXS2 cannot be written to during a transmit operation.

TXS/TXS2 can be written to only with an 8-bit manipulation instruction. The contents of TXS/TXS2 are undefined after $\overline{\text{RESET}}$ input.

(3) Shift register

This is the shift register that converts the serial data input to the RxD pin to parallel data. When one byte of data is received, the receive data is transferred to the receive buffer.

The shift register cannot be manipulated directly by the CPU.

(4) Reception control parity check

Receive operations are controlled in accordance with the contents set in the asynchronous serial interface mode register (ASIM/ASIM2). In addition, parity error and other error checks are performed during receive operations, and if an error is detected, a value is set in the asynchronous serial interface status register (ASIS/ASIS2) according to the type of error.

(5) Transmission control parity addition

Transmission operation is controlled by appending a start bit, parity bit, and stop bit to the data written to the transmit shift registers (TXS and TXS2) in accordance with the contents set to the asynchronous serial interface mode registers (ASIM and ASIM2).

(6) Selector

Selects the baud rate clock source.

17.2.2 Asynchronous Serial Interface Control Registers

(1) Asynchronous serial interface mode register (ASIM), Asynchronous serial interface mode register 2 (ASIM2)

The ASIM and ASIM2 are 8-bit registers that specify the UART mode operation.

These registers can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. The format of ASIM and ASIM is shown in Figure 17-3.

$\overline{\text{RESET}}$ input clears these registers to 00H.

Figure 17-3 Format of Asynchronous Serial Interface Mode Register (ASIM) and Asynchronous Serial Interface Mode Register 2 (ASIM2)

	⑦	⑥	5	4	3	2	1	0	Address	After Reset	R/W
ASIM	TXE	RXE	PS1	PS0	CL	SL	ISRM	SCK	0FF88H	00H	R/W
ASIM2	TXE2	RXE2	PS21	PS20	CL2	SL2	ISRM2	SCK2	0FF89H	00H	R/W

SCK SCK2	Specification of Input Clock to Baud Rate Generator
0	External clock input (ASCK, ASCK2)
1	Internal clock (fxx/2)

ISRM ISRM2	Specification of Enabling/Disabling of Reception Completion Interrupt Generation in Case of Receive Error Note
0	Enabled
1	Disabled

SL SL2	Stop Bit Length Specification (Transmission Only)
0	1 bit
1	2 bits

CL CL2	Data Character Length Specification
0	7 bits
1	8 bits

PS1 PS21	PS0 PS20	Parity Bit Specification
0	0	No parity
0	1	Transmission = 0 parity addition Reception = Parity error not generated
1	0	Odd parity
1	1	Even parity

TXE TXE2	RXE RXE2	Transmit/Receive Operation
0	0	Transmission/reception disabled, or 3-wire serial I/O mode
0	1	Reception enabled
1	0	Transmission enabled
1	1	Transmission/reception enabled

Note To disable the reception completion interrupt on occurrence of a reception error, insert wait cycles of two clocks that serve as the reference of the baud rate clock after occurrence of the reception error and before the receive buffers (RXB and RXB2) are read. Otherwise, the reception completion interrupt occurs even through the interrupt is disabled. The time equivalent to the above two clocks can be calculated by the following expression;

$$\text{Wait time} = \frac{2^{n+3}}{fxx}$$

Remark fxx: Oscillation frequency

n: Value of n when 12-bit prescaler is selected by baud rate generator control register (BRGC, BRGC2) (n = 0 to 11).

Caution An asynchronous serial interface mode register (ASIM/ASIM2) rewrite should not be performed during a transmit operation. If an ASIM/ASIM2 register rewrite is performed during a transmit operation, subsequent transmit operations may not be possible (normal operation is restored by $\overline{\text{RESET}}$ input). Software can determine whether transmission is in progress by using a transmission completion interrupt (INTST/INTST2) or the interrupt request flag (STIF/STIF2) set by INTST/INTST2.

(2) Asynchronous serial interface status register (ASIS) Asynchronous serial interface status register 2 (ASIS2)

The ASIS and ASIS2 contain flags that indicate the error contents when a receive error occurs. Flags are set (to 1) when a receive error occurs, and cleared (to 0) when data is read from the receive buffer (RXB/RXB2). If the next data is received before RXB/RXB2 is read, the overrun error flag (OVE/OVE2) is set (to 1), and the other error flags are cleared (to 0) (if there is an error in the next data, the corresponding error flag is set (to 1)).

These registers can be read only with an 8-bit manipulation instruction or bit manipulation instruction. The format of ASIS and ASIS2 is shown in Figure 17-4.

$\overline{\text{RESET}}$ input clears these registers to 00H.

Figure 17-4 Format of Asynchronous Serial Interface Status Register (ASIS) and Asynchronous Serial Interface Status Register 2 (ASIS2)

Cautions 1. The receive buffer (RXB/RXB2) must be read even if there is a receive error. If RXB/RXB2 is not read, an overrun error will occur when the next data is received, and the receive error state will continue indefinitely.

2. To disable the reception completion interrupt on occurrence of a reception error, insert wait cycles of two clocks that serve as the reference of the baud rate clock after occurrence of the reception error and before the receive buffers (RXB and RXB2) are read. Otherwise, the reception completion interrupt occurs even through the interrupt is disabled. The time equivalent to the above two clocks can be calculated by the following expression;

$$\text{Wait time} = \frac{2^{n+3}}{fxx}$$

Remark fxx: Oscillation frequency

n: Value of n when 12-bit prescaler is selected by baud rate generator control register (BRGC, BRGC2) (n = 0 to 11).

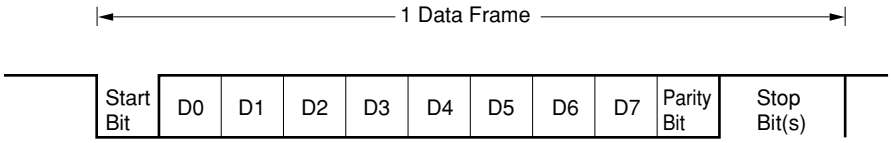
17.2.3 Data Format

Serial data transmission/reception is performed in full-duplex asynchronous mode.

The transmit/receive data format is shown in Figure 17-5. One data frame is made up of a start bit, character bits, parity bit, and stop bit(s).

Character bit length specification, parity selection and stop bit length specification for one data frame are performed by means of the asynchronous serial interface mode register (ASIM).

Figure 17-5 Asynchronous Serial Interface Transmit/Receive Data Format



- Start bit ····· 1 bit
- Character bits ····· 7 bits/8 bits
- Parity bit ····· Even parity/odd parity/0 parity/no parity
- Stop bit s ····· 1 bit/2 bits

The serial transfer rate is selected in accordance with the asynchronous serial interface mode register and baud rate generator settings. If a serial data receive error occurs, the nature of the receive error can be determined by reading the asynchronous serial interface status register (ASIS) status.

17.2.4 Parity Types and Operations

The parity bit is used to detect a bit error in the communication data. Normally, the same kind of parity bit is used on the transmission side and the reception side. With even parity and odd parity, 1 bit (odd number) errors can be detected. With 0 parity and no parity, errors cannot be detected.

- Even parity
  If the number of bits with a value of “1” in the transmit data is odd, the parity bit is set to “1”, and if the number of “1” bits is even, the parity bit is set to “0”. Control is thus performed to make the number of “1” bits in the transmit data plus the parity bit an even number. In reception, the number of “1” bits in the receive data plus the parity bit is counted, and if this number is odd, a parity error is generated.

- Odd parity
  Conversely to the case of even parity, control is performed to make the number of “1“ bits in the transmit data plus the parity bit an odd number.
  In reception, a parity error is generated if the number of “1” bits in the receive data plus the parity bit is even.

- 0 parity
  In transmission, the parity bit is set to “0” irrespective of the receive data.
  In reception, parity bit detection is not performed. Therefore, no parity error is generated irrespective of whether the parity bit is “0” or “1”.

- No parity
  In transmission, a parity bit is not added.
  In reception, reception is performed on the assumption that there is no parity bit. Since there is no parity bit, no parity error is generated.

17.2.5 Transmission

The μPD784038's asynchronous serial interface is set to the transmission enabled state when the TXE bit of the asynchronous serial interface mode register (ASIM) is set (to 1). A transmit operation is started by writing transmit data to the transmit shift register (TXS) when transmission is enabled. The start bit, parity bit and stop bit(s) are added automatically.

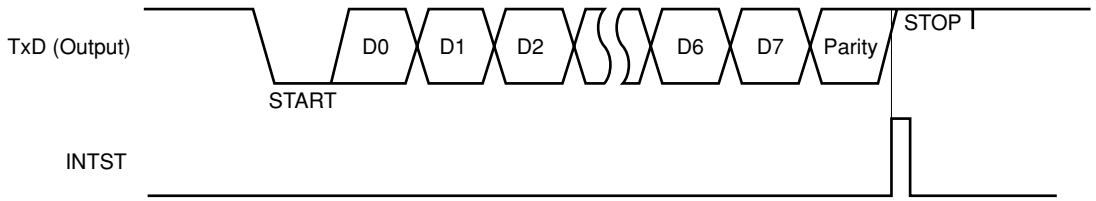
When a transmit operation is started, the data in the TXS is shifted out, and a transmission completion interrupt (INTST) is generated when the TXS is empty.

If no more data is written to the TXS, the transmit operation is discontinued.

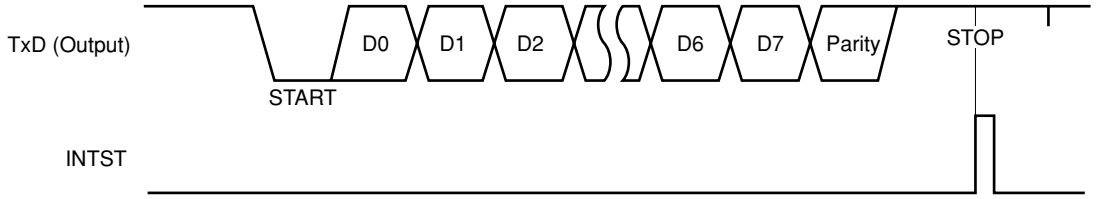
If the TXE bit is cleared (to 0) during a transmit operation, the transmit operation is discontinued immediately.

Figure 17-6 Asynchronous Serial Interface Transmission Completion Interrupt Timing

(a) Stop bit length: 1



(b) Stop bit length: 2



Cautions 1. After $\overline{\text{RESET}}$ input the transmit shift register (TXS) is emptied but a transmission completion interrupt is not generated. A transmit operation can be started by writing transmit data to the TXS.

2. An asynchronous serial interface mode register (ASIM) rewrite should not be performed during a transmit operation. If an ASIM rewrite is performed during a transmit operation, subsequent transmit operations may not be possible (normal operation is restored by $\overline{\text{RESET}}$ input). Software can determine whether transmission is in progress by using a transmission completion interrupt (INTST) or the interrupt request flag (STIF) set by INTST.

17.2.6 Reception

When the RXE bit of the asynchronous serial interface mode register (ASIM) is set (to 1), receive operations are enabled and sampling of the RxD input pin is performed.

RxD input pin sampling is performed using the serial clock (divide-by-m counter input clock) specified by ASIM and band rate generator control register (BRGC).

When the RxD pin input is driven low, the divide-by-m counter starts counting and a data sampling start timing signal is output on the m'th count. If the RxD pin input is low when sampled again by this start timing signal, the input is recognized as a start bit, the divide-by-m counter is initialized and the count is started, and data sampling is performed. When the character data, parity bit and stop bit are detected following the start bit, reception of one data frame ends.

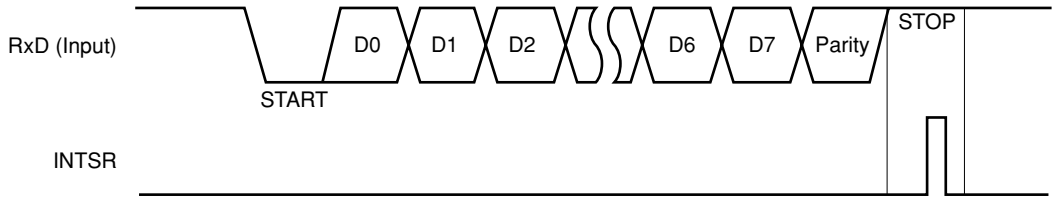
When reception of one data frame ends, the receive data in the shift register is transferred to the receive buffer, RXB, and a reception completion interrupt (INTSR) is generated.

If an error occurs, the receive data in which the error occurred is still transferred to RXB. If bit 1 (ISRM) of the ASIM was cleared (to 0) when the error occurred,

INTSR is generated. If the ISRM was set (to 1), INTSR is not generated.

If the RXE bit is cleared (to 0) during a receive operation, the receive operation is stopped immediately. In this case the contents of RXB and ASIS are not changed, and no INTSR or INTSER interrupt is generated.

Figure 17-7 Asynchronous Serial Interface Reception Completion Interrupt Timing



Cautions 1. The receive buffer (RXB) must be read even if there is a receive error. If RXB is not read, an overrun error will occur when the next data is received, and the receive error state will continue indefinitely.

2. To disable the reception completion interrupt on occurrence of a reception error, insert wait cycles of two clocks that serve as the reference of the baud rate clock after occurrence of the reception error and before the receive buffers (RXB and RXB2) are read. Otherwise, the reception completion interrupt occurs even through the interrupt is disabled. The time equivalent to the above two clocks can be calculated by the following expression;

$$\text{Wait time} = \frac{2^{n+3}}{f_{XX}}$$

Remark f_{XX}: Oscillation frequency

n: Value of n when 12-bit prescaler is selected by baud rate generator control register (BRGC, BRGC2) (n = 0 to 11).

17.2.7 Receive Errors

Three kinds of errors can occur in a receive operation: parity errors, framing errors and overrun errors. As the result of data reception, an error flag is raised in the asynchronous serial interface status register (ASIS) and a receive error interrupt (INTSER) is generated. Receive error causes are shown in Table 17-2.

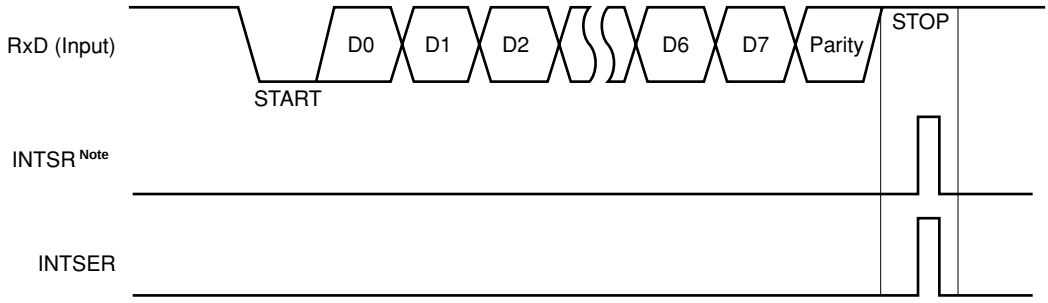
It is possible to detect the occurrence of any of the above errors during reception by reading the contents of the ASIS (see **Figures 17-4** and **17-8**).

The contents of the ASIS register are cleared (to 0) by reading the receive buffer (RXB) or by reception of the next data (if there is an error in the next data, the corresponding error flag is set).

Table 17-2 Receive Error Causes

Receive Error	Cause
Parity error	Transmit data parity specification and receive data parity do not match
Framing error	Stop bit not detected
Overrun error	Reception of next data completed before data is read from receive buffer

Figure 17-8 Receive Error Timing



Note If a receive error occurs while the ISRM bit is set (to 1), INTSR is not generated.

Remark In the μPD784038, a break signal cannot be detected by hardware. As a break signal is a low-level signal of two characters or more, a break signal may be judged to have been input if software detects the occurrence of two consecutive framing errors in which the receive data was 00H. The chance occurrence of two consecutive framing errors can be distinguished from a break signal by having the RxD pin level read by software (confirmation is possible by setting “1” in bit 0 of the port 3 mode register (PM3) and reading port 3 (P3)) and confirming that it is “0”.

Cautions 1. The contents of the asynchronous serial interface status register (ASIS) are cleared (to 0) by reading the receive buffer (RXB) or by reception of the next data. If you want to find the details of an error, therefore, ASIS must be read before reading RXB.

2. The RXB must be read even if there is a receive error. If RXB is not read, an overrun error will occur when the next data is received, and the receive error state will continue indefinitely.

3. To disable the reception completion interrupt on occurrence of a reception error, insert wait cycles of two clocks that serve as the reference of the baud rate clock after occurrence of the reception error and before the receive buffers (RXB and RXB2) are read. Otherwise, the reception completion interrupt occurs even through the interrupt is disabled. The time equivalent to the above two clocks can be calculated by the following expression;

$$\text{Wait time} = \frac{2^{n+3}}{fxx}$$

Remark fxx: Oscillation frequency

n: Value of n when 12-bit prescaler is selected by baud rate generator control register (BRGC, BRGC2) (n = 0 to 11).

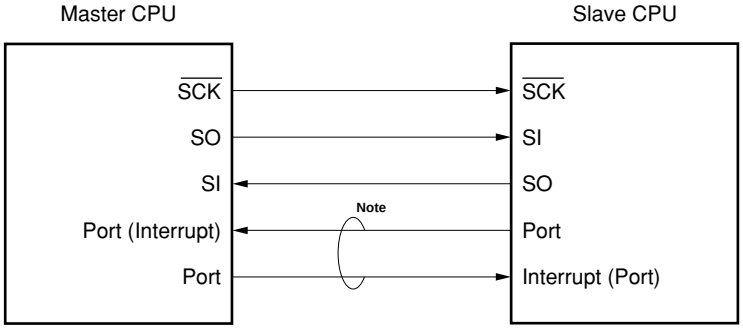
17.3 3-WIRE SERIAL I/O MODE

The 3-wire serial I/O mode is used to communicate with devices that incorporate a conventional clocked serial interface.

Basically, communication is performed using three lines: the serial clock (SCK), serial data output (SO), and serial data input (SI). Generally, a handshake line is necessary for checking the communication status.

Figure 17-9 Example of 3-Wire Serial I/O System Configuration

3-wire serial I/O ↔ 3-wire serial I/O

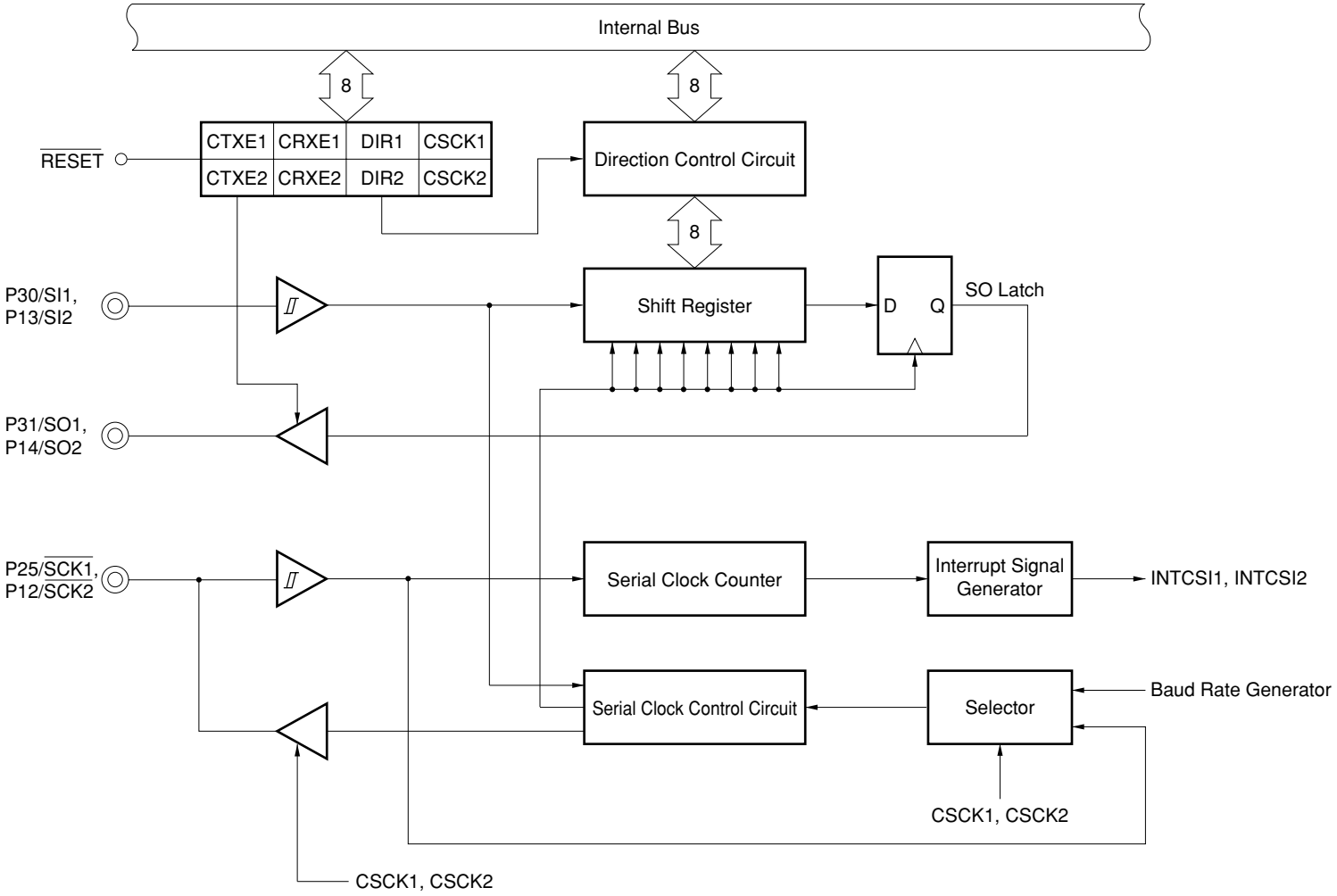


Note Handshaking lines

17.3.1 Configuration in 3-Wire Serial I/O Mode

The block diagram in the 3-wire serial I/O mode is shown in Figure 17-10.

Figure 17-10 3-Wire Serial I/O Mode Block Diagram



(1) Shift register (SIO1/SIO2)

The SIO1 and SIO2 converts 8-bit serial data to 8-bit parallel data, and vice versa. SIO1/SIO2 is used for both transmission and reception.

Actual transmit/receive operations are controlled by writing to/reading from SIO1/SIO2.

Reading/writing can be performed with an 8-bit manipulation instruction.

The contents of SIO1/SIO2 are undefined after $\overline{\text{RESET}}$ input.

(2) SO latch

The SO latch holds the SO1/SO2 pin output level.

(3) Serial clock selector

Selects the serial clock to be used.

(4) Serial clock counter

Counts the serial clocks output or input in a transmit/receive operation, and checks that 8-bit data transmission/reception has been performed.

(5) Interrupt signal generator

Generates an interrupt request when 8 serial clocks have been counted by the serial clock counter.

(6) Serial clock control circuit

Controls the supply of the serial clock to the shift register, and also controls the clock output to the $\overline{\text{SCK1}}$/$\overline{\text{SCK2}}$ pins when the internal clock is used.

(7) Direction control circuit

Switches between MSB-first and LSB-first modes.

17.3.2 Clocked Serial Interface Mode Registers (CSIM1, CSIM2)

The CSIM1 and CSIM2 are 8-bit registers that specify operations in the 3-wire serial I/O mode.

These registers can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. The CSIM1 and CSIM2 format is shown in Figure 17-11.

$\overline{\text{RESET}}$ input clears these registers to 00H.

Figure 17-11 Format of Clocked Serial Interface Mode Register 1 (CSIM1) and Clocked Serial Interface Mode Register 2 (CSIM2)

	⑦	⑥	5	4	3	2	1	0	Address	After Reset	R/W
CSIM1	CTXE1	CRXE1	0	0	0	DIR1	CSCK1	0	0FF84H	00H	R/W
CSIM2	CTXE2	CRXE2	0	0	0	DIR2	CSCK2	0	0FF85H	00H	R/W

(n = 1, 2)

CSCKn	Serial Clock Selection Bit	
	Source Clock	In case of $\overline{\text{SCKn}}$ (CTXEn, CRXEn = 1)
0	External input clock to $\overline{\text{SCKn}}$ pin	Input
1	Baud rate generator output	CMOS output

DIRn	Operating Mode Specification (Transfer Bit Order)
0	MSB-first
1	LSB-first

CTXEn	CRXEn	Transmit/Receive Operation
0	0	Transmission /reception disabled, or asynchronous serial interface mode
0	1	Reception enabled
1	0	Transmission enabled
1	1	Transmission/reception enabled

Caution Specify whether data is transferred with MSB or LSB first before writing the SIO. Even if the specification is made after writing the ISO, the byte order of the data already stored in the SIO cannot be changed.

17.3.3 Basic Operation Timing

In the 3-wire serial I/O mode, data transmission/reception is performed in 8-bit units. Data is transmitted/received bit by bit in MSB-first or LSB-first order in synchronization with the serial clock.

MSB/LSB switching is specified by the DIRn bit of the clock serial interface mode register (CSIMn).

Transmit data is output in synchronization with the fall of $\overline{SCKn}$, and receive data is sampled on the rise of $\overline{SCKn}$.

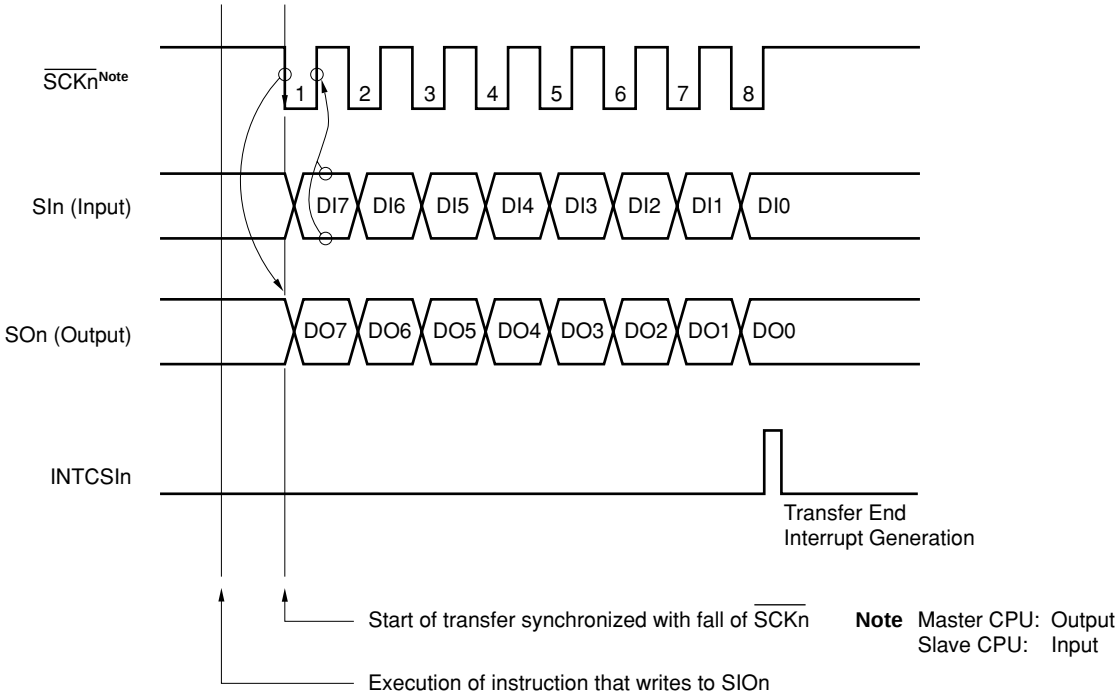
An interrupt request (INTCSIn) is generated on the 8th rise of $\overline{SCKn}$.

When the internal clock is used as $\overline{SCKn}$, $\overline{SCKn}$ output is stopped on the 8th rise of $\overline{SCKn}$ and $\overline{SCKn}$ remains high until the next data transmit or receive operation is started.

3-wire serial I/O mode timing is shown in Figure 17-12.

Figure 17-12 3-Wire Serial I/O Mode Timing (1/2)

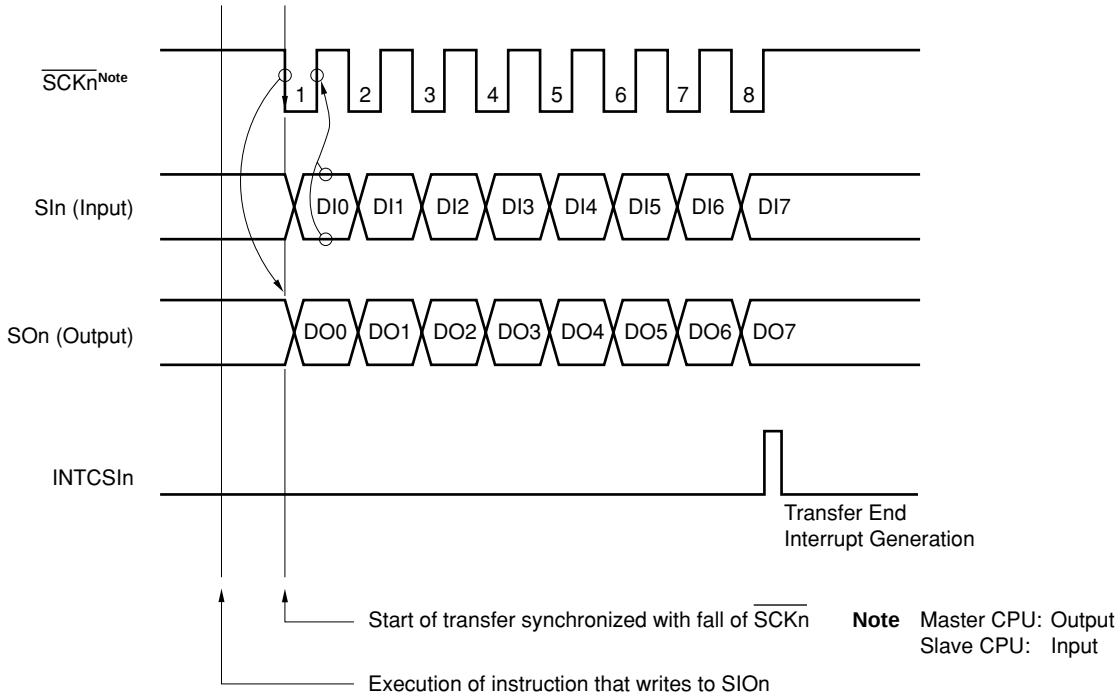
(a) MSB-first



Remark n = 1 or 2

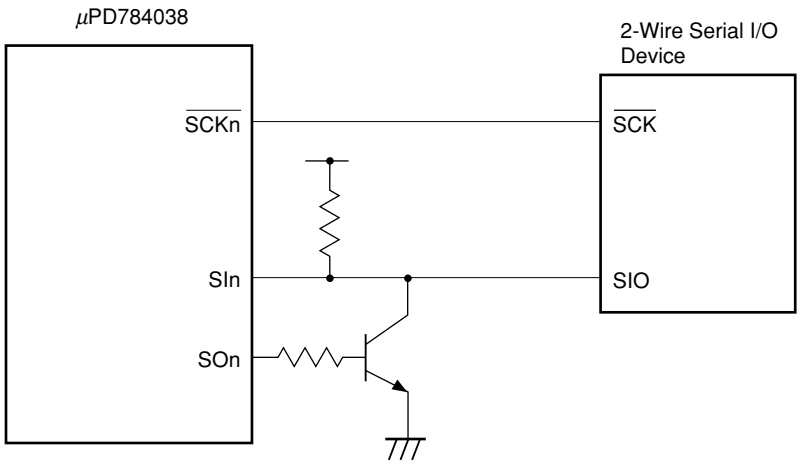
Figure 17-12 3-Wire Serial I/O Mode Timing (2/2)

(b) LSB-first



Remark If the μPD784038 is connected to a 2-wire serial I/O device, a buffer should be connected to the SOn pin as shown in Figure 17-13. In the example shown in Figure 17-13, the output level is inverted by the buffer, and therefore the inverse of the data to be output should be written to SIOn (n = 1 or 2).
In addition, non-connection of the internal pull-up resistor should be specified for the P31/SO1 or P14/SO2 pin.

Figure 17-13 Example of Connection to 2-Wire Serial I/O



17.3.4 Operation When Transmission Only is Enabled

A transmit operation is performed when the CTXEn bit of clocked serial interface mode register (CSIMn) is set (to 1). The transmit operation starts when a write to the shift register (SIOn) is performed while the CTXEn bit is set (to 1).

When the CTXEn bit is cleared (to 0), the SOn pin is in the output high level.

(1) When the internal clock is selected as the serial clock

When transmission starts, the serial clock is output from the $\overline{\text{SCKn}}$ pin and data is output in sequence from SIOn to the SOn pin in synchronization with the fall of the serial clock, and SIn pin signals are shifted into SIOn in synchronization with the rise of the serial clock.

There is a delay of up to one $\overline{\text{SCKn}}$ clock cycle between the start of transmission and the first fall of $\overline{\text{SCKn}}$.

If transmission is disabled during the transmit operation (by clearing (to 0) the CTXEn bit), $\overline{\text{SCKn}}$ clock output is stopped and the transmit operation is discontinued on the next rise of $\overline{\text{SCKn}}$. In this case an interrupt request (INTCSIn) is not generated, and the SOn pin becomes output high level.

(2) When an external clock is selected as the serial clock

When transmission starts, data is output in sequence from SIOn to the SOn pin in synchronization with the fall of the serial clock input to the $\overline{\text{SCKn}}$ pin after the start of transmission, and SIn pin signals are shifted into SIOn in synchronization with the rise of the $\overline{\text{SCKn}}$ pin input. If transmission has not started, shift operations are not performed and the SOn pin output level does not change even if the serial clock is input to the $\overline{\text{SCKn}}$ pin.

If transmission is disabled during the transmit operation (by clearing (to 0) the CTXEn bit), the transmit operation is discontinued and subsequent $\overline{\text{SCKn}}$ input is ignored. In this case an interrupt request (INTCSIn) is not generated, and the SOn pin becomes output high level.

Remark n = 1 or 2

17.3.5 Operation When Reception Only is Enabled

A receive operation is performed when the CRXEn bit of the clocked serial interface mode register (CSIMn) is set (to 1). The receive operation starts when the CRXEn changes from “0” to “1”, or when a read from shift register (SIOn) is performed.

(1) When the internal clock is selected as the serial clock

When reception starts, the serial clock is output from the $\overline{\text{SCKn}}$ pin and the SIn pin data is fetched in sequence into shift register (SIOn) in synchronization with the rise of the serial clock.

There is a delay of up to one $\overline{\text{SCKn}}$ clock cycle between the start of reception and the first fall of $\overline{\text{SCKn}}$.

If reception is disabled during the receive operation (by clearing (to 0) the CRXEn bit), $\overline{\text{SCKn}}$ clock output is stopped and the receive operation is discontinued on the next rise of $\overline{\text{SCKn}}$. In this case an interrupt request (INTCSIn) is not generated, and the contents of the SIOn are undefined.

(2) When an external clock is selected as the serial clock

When reception starts, the SIn pin data is fetched into shift register (SIOn) in synchronization with the rise of the serial clock input to the $\overline{\text{SCKn}}$ pin after the start of reception. If reception has not started, shift operations are not performed even if the serial clock is input to the $\overline{\text{SCKn}}$ pin.

If reception is disabled during the receive operation (by clearing (to 0) the CRXEn bit), the receive operation is discontinued and subsequent $\overline{\text{SCKn}}$ input is ignored. In this case an interrupt request (INTCSIn) is not generated.

Remark n = 1 or 2

17.3.6 Operation When Transmission/Reception is Enabled

When the CTXEn bit and CRXEn bit of the clocked serial interface mode register (CSIMn) register are both set (to 1), a transmit operation and receive operation can be performed simultaneously (transmit/receive operation). The transmit/receive operation is started when the CRXEn bit is changed from "0" to "1", or by performing a write to shift register (SIOn).

When a transmit/receive operation is started for the first time, the CRXEn bit always changes from "0" to "1", and there is thus a possibility that the transmit/receive operation will start immediately, and undefined data will be output. The first transmit data should therefore be written to SIOn beforehand when both transmission and reception are disabled (when the CTXEn bit and CRXEn bit are both cleared (to 0)), before enabling transmission/reception. However, specify whether data is transferred with MSB or LSB first before writing the SIOn. Even if the specification is made after writing the SIOn, the byte order of the data already stored in the SIOn cannot be changed.

When transmission/reception is disabled (CTXEn = CRXEn = 0), the SOn pin is in the output high level.

(1) When the internal clock is selected as the serial clock

When transmission/reception starts, the serial clock is output from the $\overline{\text{SCK}}$n pin, data is output in sequence from shift register (SIOn) to the (SOn) pin in synchronization with the fall of the serial clock, and SIn pin data is shifted in order into SIOn in synchronization with the rise of the serial clock.

There is a delay of up to one $\overline{\text{SCK}}$n clock cycle between the start of transmission and the first fall of $\overline{\text{SCK}}$n.

If either transmission or reception is disabled during the transmit/receive operation, only the disabled operation is discontinued. If transmission only is disabled, the SOn pin becomes output high level. If reception only is disabled, the contents of the SIOn will be undefined.

If transmission and reception are disabled simultaneously, $\overline{\text{SCK}}$n clock output is stopped and the transmit and receive operations are discontinued on the next rise of $\overline{\text{SCK}}$n. When transmission and reception are disabled simultaneously, the contents of SIOn are undefined, an interrupt request (INTCSIn) is not generated, and the SOn pin becomes output high level.

(2) When an external clock is selected as the serial clock

When transmission/reception starts, data is output in sequence from shift register (SIOn) to the SOn pin in synchronization with the fall of the serial clock input to the $\overline{\text{SCK}}$n pin after the start of transmission/reception, and SIn pin data is shifted in order into SIOn in synchronization with the rise of the serial clock. If transmission/reception has not started, the SIOn shift operations are not performed and the SOn pin output level does not change even if the serial clock is input to the $\overline{\text{SCK}}$n pin.

If either transmission or reception is disabled during the transmit/receive operation, only the disabled operation is discontinued. If transmission only is disabled, the SOn pin becomes output high level. If reception only is disabled, the contents of the SIOn will be undefined.

If transmission and reception are disabled simultaneously, the transmit and receive operations are discontinued and subsequent $\overline{\text{SCK}}$n input is ignored. When transmission and reception are disabled simultaneously, the contents of SIOn are undefined, an interrupt request (INTCSIn) is not generated, and the SOn pin becomes output high level.

Remark n = 1 or 2

17.3.7 Corrective Action in Case of Slippage of Serial Clock and Shift Operations

When an external clock is selected as the serial clock, there may be slippage between the number of serial clocks and shift operations due to noise, etc. In this case, since the serial clock counter is initialized by disabling both transmit operations and receive operations (by clearing (to 0) the CTXEn bit and CRXEn bit), synchronization of the shift operations and the serial clock can be restored by using the first serial clock input after reception or transmission is next enabled as the first clock.

Remark n = 1 or 2

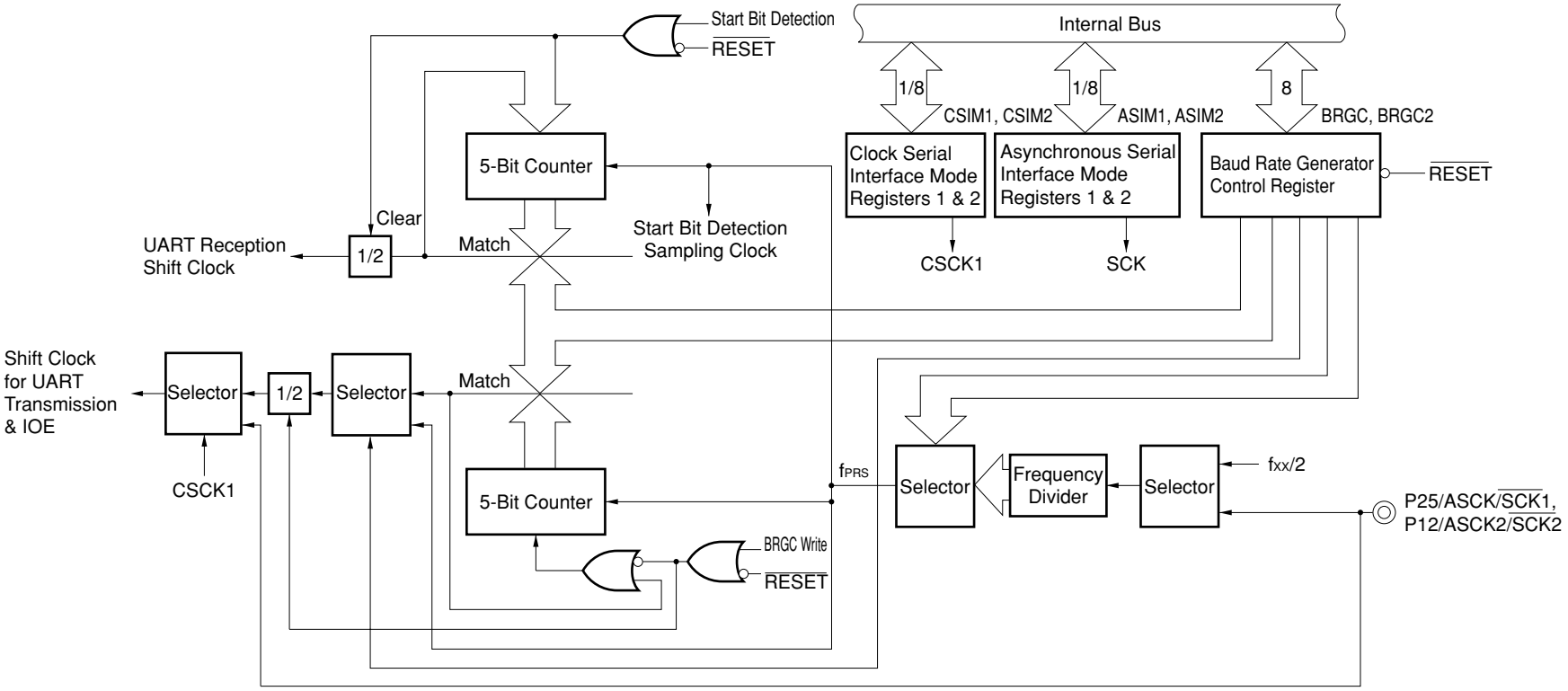
17.4 BAUD RATE GENERATOR

The baud rate generator is the circuit that generates the UART/IOE serial clock. Two independent circuits are incorporated, one for each serial interface.

17.4.1 Baud Rate Generator Configuration

The baud rate generator block diagram is shown in Figure 17-14.

Figure 17-14 Baud Rate Generator Block Diagram



(1) 5-bit counter

Counter that counts the clock (f_{PRS}) by which the output from the frequency divider is selected. Generates a signal with the frequency selected by the low-order 4 bits of the baud rate generator control registers (BRGC/BRGC2).

(2) Frequency divider

Scales the internal clock ($f_{XX}/2$) or, in asynchronous serial interface mode, a clock that is twice the external baud rate input (ASCK/ASCK2), and selects fPRS with the next-stage selector.

(3) Both-edge detection circuit

Detects both edges of the ASCK/ASCK2 pin input signal and generates a signal with a frequency twice that of the ASCK/ASCK2 input clock.

17.4.2 Baud Rate Generator Control Register (BRGC, BRGC2)

The BRGC and BRGC2 are 8-bit registers that set the baud rate clock in asynchronous serial interface mode or the shift clock in 3-wire serial I/O mode.

These registers can be read/written with an 8-bit manipulation instruction. The BRGC and BRGC2 format is shown in Figure 17-15.

$\overline{\text{RESET}}$ input clears the BRGC register to 00H.

Caution When a baud rate generator control register (BRGC, BRGC2) write instruction is executed, the 5-bit counter and 1/2 frequency divider operations are reset. Consequently, if a write to the BRGC and BRGC2 is performed during communication, the generated baud rate clock may be disrupted, preventing normal communication from continuing. The BRGC and BRGC2 should therefore not be written to during communication.

Figure 17-15 Baud Rate Generator Control Register (BRGC) Format and Baud Rate Generator Control Register 2 (BRGC2) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
BRGC	TPS3	TPS2	TPS1	TPS0	MDL3	MDL2	MDL1	MDL0	0FF90H	00H	R/W
BRGC2	TPS23	TPS22	TPS21	TPS20	MDL23	MDL22	MDL21	MDL20	0FF91H	00H	R/W

f_{PRS}: Prescaler output selection clock

MDL3 / MDL23	MDL2 / MDL22	MDL1 / MDL21	MDL0 / MDL20	k	Baud Rate Generator Input Clock Note 1
0	0	0	0	0	$f_{PRS}/16$
0	0	0	1	1	$f_{PRS}/17$
0	0	1	0	2	$f_{PRS}/18$
0	0	1	1	3	$f_{PRS}/19$
0	1	0	0	4	$f_{PRS}/20$
0	1	0	1	5	$f_{PRS}/21$
0	1	1	0	6	$f_{PRS}/22$
0	1	1	1	7	$f_{PRS}/23$
1	0	0	0	8	$f_{PRS}/24$
1	0	0	1	9	$f_{PRS}/25$
1	0	1	0	10	$f_{PRS}/26$
1	0	1	1	11	$f_{PRS}/27$
1	1	0	0	12	$f_{PRS}/28$
1	1	0	1	13	$f_{PRS}/29$
1	1	1	0	14	$f_{PRS}/30$
1	1	1	1	15	f_{PRS} Note 2

Notes 1. Only $f_{PRS}/16$ can be selected when ASCK/ASCK2 input is used.
2. Can only be used in 3-wire serial I/O mode.

f_{XX}: Oscillator frequency or external clock input

TPS3 / TPS23	TPS2 / TPS22	TPS1 / TPS21	TPS0 / TPS20	n	12-Bit Prescaler Tap Selection (f_{PRS})
0	0	0	0	0	$f_{XX}/4$, $f_{ASCK}/2^{Note}$
0	0	0	1	1	$f_{XX}/8$, $f_{ASCK}/4$
0	0	1	0	2	$f_{XX}/16$, $f_{ASCK}/8$
0	0	1	1	3	$f_{XX}/32$, $f_{ASCK}/16$
0	1	0	0	4	$f_{XX}/64$, $f_{ASCK}/32$
0	1	0	1	5	$f_{XX}/128$, $f_{ASCK}/64$
0	1	1	0	6	$f_{XX}/256$, $f_{ASCK}/128$
0	1	1	1	7	$f_{XX}/512$, $f_{ASCK}/256$
1	0	0	0	8	$f_{XX}/1,024$, $f_{ASCK}/512$
1	0	0	1	9	$f_{XX}/2,048$, $f_{ASCK}/1,024$
1	0	1	0	10	$f_{XX}/4,096$, $f_{ASCK}/2,048$
1	0	1	1	11	$f_{XX}/8,192$, $f_{ASCK}/4,096$
Other than the above				Setting prohibited	

Note Can not be selected when the value set in bits MDL3 to MDL0 or MDL23 to MDL20, k = 15.

17.4.3 Baud Rate Generator Operation

The baud rate generator only operates when UART/IOE transmit/receive operations are enabled. The generated baud rate clock is a signal scaled from the internal clock ($f_{XX}/2$) or a signal scaled from the clock input from the external baud rate input (ASCK) pin.

Caution If a write to the baud rate generator control register (BRGC) is performed during communication, the generated baud rate clock may be disrupted, preventing normal communication from continuing. The BRGC should therefore not be written to during communication.

(1) Baud rate clock generation in UART mode

(a) Using internal clock ($f_{XX}/2$)

This function is selected by setting (to 1) bit 0 (SCK) of the asynchronous serial interface mode register (ASIM). The internal clock ($f_{XX}/2$) is scaled by the frequency divider, this signal (f_{PRS}) is scaled by the 5-bit counter, and the signal further divided by 2 is used as the baud rate. The baud rate is given by the following expression:

$$\text{(Baud rate)} = \frac{f_{XX}}{(k+16)\cdot 2^{n+3}}$$

f_{XX} : Oscillator frequency or external clock input
k : Value set in bits MDL3 to MDL0 of BRGC (k = 0 to 14)
n : Value set in bits TPS3 to TPS0 of BRGC (n = 0 to 11)

(b) Using external baud rate input

This function is selected by clearing (to 0) bit 0 (SCK) of the asynchronous serial interface mode register (ASIM). When this function is used, bits MDL3 to MDL0 of the baud rate generator control register (BRGC) must all be cleared (to 0) (k = 0).

When this function is used with UART2, it is necessary to set (to 1) bit 2 of the port 3 mode control register (PMC3) and set the P12 pin to control mode.

The ASCK pin input clock is scaled by the frequency divider, and the signal obtained by dividing this signal by 32 (f_{PRS}) (division by 16 and division by 2) is used as the baud rate. The baud rate is given by the following expression:

$$\text{(Baud rate)} = \frac{f_{ASCK}}{2^{n+6}}$$

f_{ASCK} : ASCK pin input clock frequency
n : Value set in bits TPS3 to TPS0 of BRGC (n = 0 to 11)

When this function is used, a number of baud rates can be generated by one external input clock.

(2) Serial clock generation in 3-wire serial I/O mode

Selected when the CSCK1 bit of the clocked serial interface mode register (CSIM1) is set (to 1) and $\overline{SCK1}$ is output.

(a) Normal mode

The internal clock (fxx/2) is scaled by the frequency divider, this signal (f_{PRS}) is scaled by the 5-bit counter, and the signal further divided by 2 is used as the serial clock. The serial clock is given by the following expression:

$$(\text{Serial clock}) = \frac{f_{XX}}{(k+16)\cdot 2^{n+3}}$$

fxx : Oscillator frequency or external clock input
k : Value set in bits MDL3 to MDL0 of BRGC (k = 0 to 14)
n : Value set in bits TPS3 to TPS0 of BRGC (n = 0 to 11)

(b) High-speed mode

When this function is used, bits MDL3 to MDL0 of the baud rate generator control register (BRGC) are all set (1) (k = 15).

The internal clock (fxx/2) is scaled by the frequency divider, and this signal (f_{PRS}) divided by 2 is used as the serial clock. The serial clock is given by the following expression:

$$(\text{Serial clock}) = \frac{f_{XX}}{2^{n+3}}$$

fxx : Oscillator frequency or external clock input
n : Value set in bits TPS3 to TPS0 of BRGC (n = 1 to 11)

17.4.4 Baud Rate Setting in Asynchronous Serial Interface Mode

There are two methods of setting the baud rate, as shown in Table 17-3.

This table shows the range of baud rates that can be generated, the baud rate calculation expression and selection method for each case.

Table 17-3 Baud Rate Setting Methods

Baud Rate Clock Source		Selection Method	Baud Rate Calculation Expression	Baud Rate Range
Baud rate generator	Internal system clock	SCK in ASIM = 1	$\frac{f_{XX}}{(k+16)\cdot 2^{n+3}}$	$\frac{f_{XX}}{491,520} - \frac{f_{XX}}{128}$
	ASCK input	SCK in ASIM = 0	$\frac{f_{ASCK}}{2^{n+6}}$	$\frac{f_{ASCK}}{131,072} - \frac{f_{ASCK}}{64}$ **Note**

Note Including f_{ASCK} input range: (0 – f_{XX}/256)

Remarks f_{XX} : Oscillator frequency or external clock input
k : Value set in bits MDL3 to MDL0 of BRGC (k = 0 to 14; see Figure 17-15)
n : Value set in bits TPS3 to TPS0 of BRGC (n = 0 to 11; see Figure 17-15)
f_{ASCK} : ASCK input clock frequency (0 – f_{XX}/4)

(1) Examples of settings when baud rate generator is used

Examples of baud rate generator control register (BRGC) settings when the baud rate generator is used are shown below.

When the baud rate generator is used, the SCK bit of the asynchronous serial interface mode register (ASIM) should be set (to 1).

Table 17-4 Examples of BRGC Settings When Baud Rate Generator is Used

Oscillator Frequency (f_{XX}) or External Clock (f_X)	32.0000 MHz		31.9488 MHz		25.0000 MHz		24.5760 MHz		12.0000 MHz		11.0592 MHz	
Baud Rate [bps]	BRGC Value	Error (%)	BRGC Value	Error (%)	BRGC Value	Error (%)	BRGC Value	Error (%)	BRGC Value	Error (%)	BRGC Value	Error (%)
75	BAH	0.16	BAH	0.00	B4H	1.73	B4H	0.00	A4H	2.34	A2H	0.00
110	B2H	1.36	B2H	1.52	ACH	0.92	ABH	1.01	9BH	1.36	99H	1.82
150	AAH	0.16	AAH	0.00	A4H	1.73	A4H	0.00	94H	2.34	92H	0.00
300	9AH	0.16	9AH	0.00	94H	1.73	94H	0.00	84H	2.34	82H	0.00
600	8AH	0.16	8AH	0.00	84H	1.73	84H	0.00	74H	2.34	72H	0.00
1,200	7AH	0.16	7AH	0.00	74H	1.73	74H	0.00	64H	2.34	62H	0.00
2,400	6AH	0.16	6AH	0.00	64H	1.73	64H	0.00	54H	2.34	52H	0.00
4,800	5AH	0.16	5AH	0.00	54H	1.73	54H	0.00	44H	2.34	42H	0.00
9,600	4AH	0.16	4AH	0.00	44H	1.73	44H	0.00	34H	2.34	32H	0.00
19,200	3AH	0.16	3AH	0.00	34H	1.73	34H	0.00	24H	2.34	22H	0.00
31,250	30H	0.00	30H	0.16	29H	0.00	29H	1.70	18H	0.00	16H	0.54
38,400	2AH	0.16	2AH	0.00	24H	1.73	24H	0.00	14H	2.34	12H	0.00
76,800	1AH	0.16	1AH	0.00	14H	1.73	14H	0.00	04H	2.34	02H	0.00
115,200	11H	2.12	11H	1.96	0BH	0.47	0BH	1.23	00H	18.62	00H	25.00
153,600	0AH	0.16	0AH	0.00	04H	1.73	04H	0.00	00H	38.96	00H	43.75

(2) Examples of settings when external baud rate input (ASCK) is used

Table 17-5 shows an example of setting when external baud rate input (ASCK) is used. When using the ASCK input, clear the SCK bit of the asynchronous serial interface mode register (ASIM) to 0, and set the corresponding pin in the control mode by using PMC3 or PMC1.

Table 17-5 Examples of Settings When External Baud Rate Input (ASCK) is Used

f_{ASCK} (ASCK Input Frequency)	153.6 kHz	4.9152 MHz
Baud Rate [bps]	BRGC Value	BRGC Value
75	50H	A0H
150	40H	90H
300	30H	80H
600	20H	70H
1,200	10H	60H
2,400	00H	50H
4,800	—	40H
9,600	—	30H
19,200	—	20H
38,400	—	10H
76,800	—	00H

17.5 CAUTIONS

(1) An asynchronous serial interface mode register (ASIM) rewrite should not be performed during a transmit operation. If an ASIM rewrite is performed during a transmit operation, subsequent transmit operations may not be possible (normal operation is restored by $\overline{\text{RESET}}$ input).
Software can determine whether transmission is in progress by using a transmission completion interrupt (INTST) or the interrupt request flag (STIF) set by INTST.

(2) After $\overline{\text{RESET}}$ input the transmit shift register (TXS) is emptied but a transmission completion interrupt is not generated. A transmit operation can be started by writing transmit data to the TXS.

(3) The receive buffer (RXB) must be read even if there is a receive error. If RXB is not read, an overrun error will occur when the next data is received, and the receive error state will continue indefinitely.

(4) To disable the reception completion interrupt on occurrence of a reception error, insert wait cycles of two clocks that serve as the reference of the baud rate clock after occurrence of the reception error and before the receive buffers (RXB and RXB2) are read. Otherwise, the reception completion interrupt occurs even through the interrupt is disabled. The time equivalent to the above two clocks can be calculated by the following expression;

$$\text{Wait time} = \frac{2^{n+3}}{f_{XX}}$$

Remark f_{XX}: Oscillation frequency
n: Value of n when 12-bit prescaler is selected by baud rate generator control register (BRGC, BRGC2) (n = 0 to 11).

(5) The contents of the asynchronous serial interface status register (ASIS) are cleared (to 0) by reading the receive buffer (RXB) or by reception of the next data. If you want to find the details of an error, therefore, ASIS must be read before reading RXB.

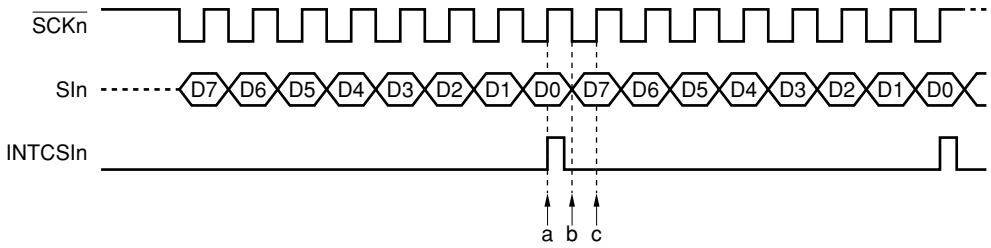
(6) The baud rate generator control register (BRGC) should not be written to during communication. If a write instruction is executed, the 5-bit counter and 1/2 frequency divider operations will be reset, and the generated baud rate clock may be disrupted, preventing normal communication from continuing.

(7) Specify whether data is transferred with MSB or LSB first before writing the SIO. Even if the specification is made after writing the SIO, the byte order of the data already stored in the SIO cannot be changed.

★ (8) When data is successively transmitted from the transmission side in 3-wire serial I/O mode, the second and subsequent receive data may be undefined under the following conditions <1> and <2>.

<1> Read from the shift register (SIOn) is not completed in the period from reception completion (**a** in the figure below) to the next fall of the serial clock ($\overline{\text{SCKn}}$) (**b** in the figure below) (n = 1 or 2)

<2> The reception enable bit is cleared in the period from reception completion (**a** in the figure below) to the next rise of the serial clock ($\overline{\text{SCKn}}$) (**c** in the figure below), and the reception enable flag cannot be set after the shift register (SIOn) is read (n = 1 or 2)



Implement the following workaround to prevent this bug.

- Read the shift register after reception completion (**a** in the figure above) by the next fall of the serial clock (**b** in the figure above).
- Clear the reception enable bit after reception completion (**a** in the figure above) by the next rise of the serial clock (**c** in the figure above), read the shift register, and set the reception enable flag.

Remark n = 1 or 2

CHAPTER 18 3-WIRE/2-WIRE SERIAL I/O MODE

18.1 FUNCTIONS

(1) 3-wire serial I/O mode (MSB/LSB first)

In this mode, 8-bit data are transferred by using three lines, a serial clock line ($\overline{SCK0}$) and two serial bus lines (SO0 and SI0). This mode is useful when connecting a peripheral I/O or display controller having the conventional clocked serial interface.

Generally, a handshake line is necessary for checking the communication status.

(2) 2-wire serial I/O mode (MSB first)

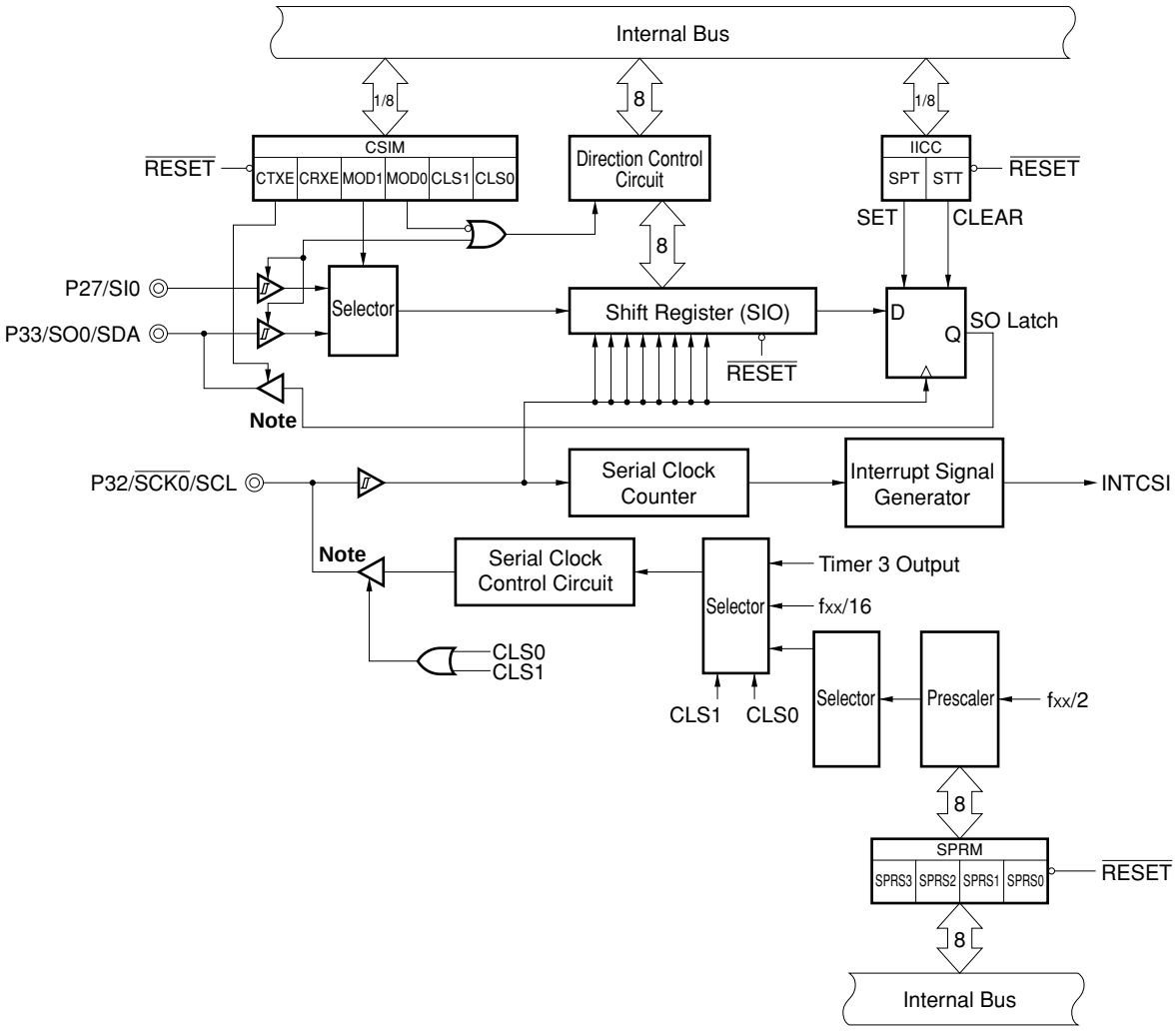
In this mode, 8-bit data are transferred by using two lines, a serial clock line (SCL) and a serial data bus line (SDA).

Generally, a handshake line is necessary for checking the communication status.

18.2 CONFIGURATION

Figure 18-1 shows the block diagram of the clocked serial interface (CSI) in the 3-wire/2-wire serial I/O mode.

Figure 18-1 Block Diagram of Clocked Serial Interface (in 3-wire/2-wire serial mode)



Note CMOS push-pull output : in 3-wire serial I/O mode
N-ch open-drain output : in 2-wire serial I/O mode

(1) Shift register (SIO)

The SIO converts 8-bit serial data to 8-bit parallel data, and vice versa. SIO is used for both transmission and reception. Actual transmit/receive operations are controlled by writing to/reading from SIO. SIO can be read or written to with an 8-bit manipulation instruction. The contents of SIO are undefined after $\overline{\text{RESET}}$ input.

(2) SO latch

The SO latch holds the SO0/SDA pin output level. This latch can also be directly controlled by software.

(3) Serial clock selector

Selects the serial clock to be used.

(4) Serial clock counter

Counts the serial clocks output or input in a transmit/receive operation, and checks that 8-bit data transmission/reception has been performed.

(5) Interrupt signal generator

A interrupt request is generated when 8 serial clocks have been counted by the serial clock counter.

(6) Serial clock control circuit

Controls the supply of the serial clock to the shift register (SIO), and also controls the clock output to the $\overline{\text{SCK0}}$ pin when the internal clock is used.

(7) Direction control circuit

Controls the transmit/receive data shift direction.

18.3 CONTROL REGISTERS

18.3.1 Clocked Serial Interface Mode Register (CSIM)

The CSIM is an 8-bit register that specifies the serial interface operating mode, serial clock, etc.

This register can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. The CSIM format is shown in Figure 18-2.

$\overline{\text{RESET}}$ input clears the CSIM register to 00H.

Figure 18-2 Clocked Serial Interface Mode Register (CSIM) Format

	⑦	⑥	5	4	3	2	1	0	Address	After Reset	R/W
CSIM	CTXE	CRXE	0	0	MOD1	MOD0	CLS1	CLS0	0FF82H	00H	R/W

CLS1	CLS0	Serial Clock Specification			$\overline{\text{SCK0}}$, SCL Pin
0	0	External clock	Slave		Input
0	0	Internal clock	Master	TM3/2	Output
1	0			SPRS	
1	1			fxx/16	

MOD1	MOD0	Operating Mode Selection Bit		
		Operating Mode	Transfer Direction	Pins Used
0	0	3-wire serial I/O mode	MSB-first	SO0, SI0, $\overline{\text{SCK0}}$
0	1	3-wire serial I/O mode	LSB-first	
1	0	2-wire serial I/O mode	MSB-first	SDA, SCL
1	1	Setting prohibited		

CRXE	Receive Operations
0	Disabled
1	Enabled

CTXE	Transmit Operations
0	Disabled
1	Enabled

Caution When changing from "CTXE = 0, CRXE = 1" to "CTXE = 1, CRXE = 0", and when changing from "CTXE = 1, CRXE = 0" to "CTXE = 0, CRXE = 1", ensure that this is not done with a single instruction, as this will result in malfunction of the serial clock counter, and the first communication after the change will finish in fewer than 8 bits. Instead, two instructions should be used as shown below.

Example To change "CTXE = 1, CRXE = 0" to "CTXE = 0, CRXE = 1"

```
CLR1 CTXE
SET1 CRXE
```

18.3.2 Prescaler Mode Register for Serial Clock (SPRM)

SPRM is an 8-bit register that specifies the division ratio of the serial clock when SPRS is specified by setting the CLS1 bit of the clocked serial interface mode register (CSIM) to 1 and clearing the CLS0 bit of CSIM to 0.

This register can be read or written to with an 8-bit manipulation instruction. Figure 18-3 shows the format of SPRM. $\overline{\text{RESET}}$ input sets this register to 04H.

Rewrite the contents of SPRM only when transmission/reception is disabled (CTXE = CRXE = 0).

Figure 18-3 Format of Prescaler Mode Register (SPRM) for Serial Clock

	7	6	5	4	3	2	1	0	Address	After reset	R/W
SPRM	0	0	0	0	SPRS3	SPRS2	SPRS1	SPRS0	0FF81H	04H	R/W

SPRS3	SPRS2	SPRS1	SPRS0	Specifies Serial Clock
0	0	0	0	$f_{XX}/16$
0	0	0	1	$f_{XX}/24$
0	0	1	0	$f_{XX}/32$
0	0	1	1	$f_{XX}/48$
0	1	0	0	$f_{XX}/64$
0	1	0	1	$f_{XX}/96$
0	1	1	0	$f_{XX}/128$
0	1	1	1	$f_{XX}/192$
1	0	0	0	$f_{XX}/256$
1	0	0	1	$f_{XX}/384$

18.3.3 I²C Bus Control Register (IICC)

IICC is an 8-bit register composed of bits which control the SO latch status.

IICC is read or written with 8-bit manipulation instructions and bit manipulation instructions. When a read is performed, IICC is read as “00”. The format of the IICC register is shown in Figure 18-4. The IICC register must not be written to during a transmit, receive, or transmit/receive operation.

$\overline{\text{RESET}}$ input clears SBIC to 00H.

Figure 18-4 Format of I²C Bus Control Register (IICC)

	7	6	5	4	3	2	①	⓪	Address	After reset	R/W
IICC	0	0	0	0	0	0	STT	SPT	0FF80H	00H	R/W

SPT	Operation
0	Not affected
1	Sets SO latch (1)

STT	Operation
0	Not affected
1	Clears SO latch (0)

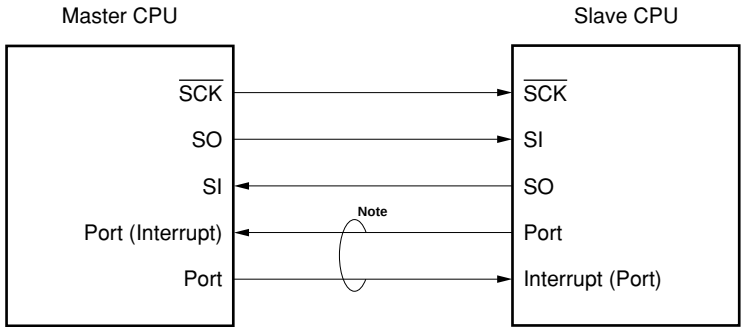
18.4 3-WIRE SERIAL I/O MODE

The 3-wire serial I/O mode is used to communicate with devices that incorporate a conventional clocked serial interface.

Basically, communication is performed using three lines: the serial clock ($\overline{\text{SCK0}}$), serial data output (SO0), and serial data input (SI0). Generally, a handshake line is necessary for checking the communication status.

Figure 18-5 Example of 3-Wire Serial I/O System Configuration

3-wire serial I/O ↔ 3-wire serial I/O



Note Handshaking lines

18.4.1 Basic Operation Timing

In the 3-wire serial I/O mode, data transmission/ reception is performed in 8-bit units. Data is transmitted/received bit by bit in MSB-first or LSB-first order in synchronization with the serial clock.

MSB first/LSB first switching is specified by the MOD 0 bit of the clocked serial interface mode register (CSIM).

Transmit data is output in synchronization with the fall of $\overline{SCK0}$, and receive data is sampled on the rise of $\overline{SCK0}$.

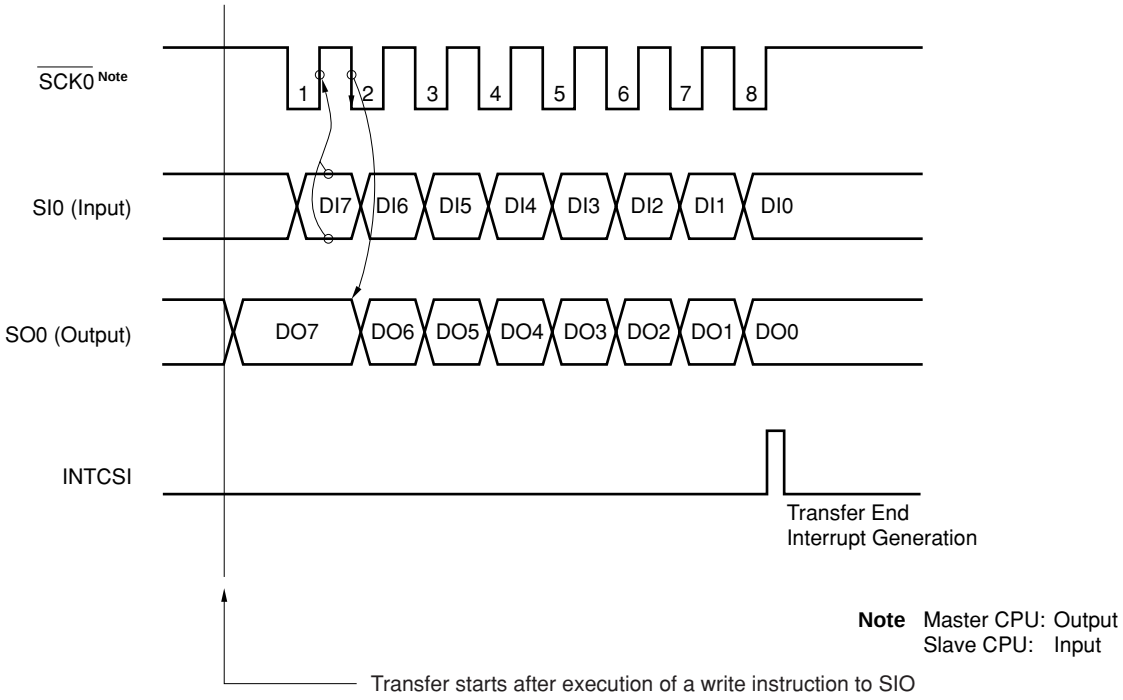
An interrupt request (INTCSI) is generated on the 8th rise of $\overline{SCK0}$.

When the internal clock is used as $\overline{SCK0}$, $\overline{SCK0}$ output is stopped on the 8th rise of $\overline{SCK0}$ and $\overline{SCK0}$ remains high until the next data transmit or receive operation is started.

3-wire serial I/O mode timing is shown in Figure 18-6.

★ **Figure 18-6 3-Wire Serial I/O Mode Timing (1/2)**

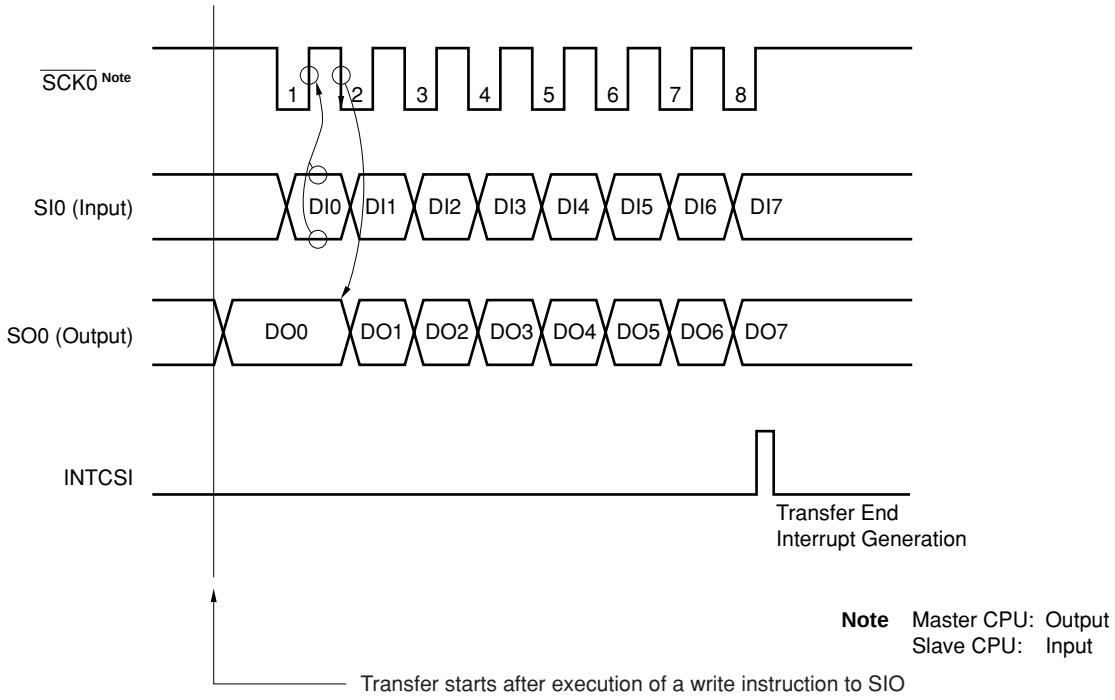
(a) MSB-first



Note Master CPU: Output
Slave CPU: Input

★ **Figure 18-6 3-Wire Serial I/O Mode Timing (2/2)**

(b) LSB-first



In the 3-wire serial I/O mode, the SO0 pin functions as a CMOS push-pull output.

18.4.2 Operation When Transmission Only is Enabled

A transmit operation is performed when the CTXE bit of the clocked serial interface mode register (CSIM) is set (to 1). The transmit operation starts when a write to the shift register (SIO) is performed while the CTXE1 bit is set (to 1).

When the CTXE bit is cleared (to 0), the SO0 pin is in the output high impedance state.

(1) When the internal clock is selected as the serial clock

When transmission starts, the serial clock is output from the $\overline{SCK0}$ pin and data is output in sequence from SIO to the SO0 pin in synchronization with the fall of the serial clock, and SI0 pin signals are shifted into SIO in synchronization with the rise of the serial clock.

There is a delay of up to one $\overline{SCK0}$ clock cycle between the start of transmission and the first fall of $\overline{SCK0}$.

If transmission is disabled during the transmit operation (by clearing (to 0) the CTXE bit), $\overline{SCK0}$ clock output is stopped and the transmit operation is discontinued on the next rise of $\overline{SCK0}$. In this case an interrupt request (INTCSI) is not generated, and the SO0 pin becomes output high impedance.

(2) When an external clock is selected as the serial clock

When transmission starts, data is output in sequence from SIO to the SO0 pin in synchronization with the fall of the serial clock input to the $\overline{SCK0}$ pin after the start of transmission, and SI0 pin signals are shifted into SIO in synchronization with the rise of the $\overline{SCK0}$ pin input. If transmission has not started, shift operations are not performed and the SO0 pin output level does not change even if the serial clock is input to the $\overline{SCK0}$ pin.

If transmission is disabled during the transmit operation (by clearing (to 0) the CTXE bit), the transmit operation is discontinued and subsequent $\overline{SCK0}$ input is ignored. In this case an interrupt request (INTCSI) is not generated, and the SO0 pin becomes output high impedance.

18.4.3 Operation When Reception Only is Enabled

A receive operation is performed when the CRXE bit of the clocked serial interface mode register (CSIM) is set (to 1). The receive operation starts when the CRXE changes from “0” to “1”, or when a read from shift register (SIO) is performed.

(1) When the internal clock is selected as the serial clock

When reception starts, the serial clock is output from the $\overline{SCK0}$ pin and the SI0 pin data is fetched in sequence into shift register (SIO) in synchronization with the rise of the serial clock.

There is a delay of up to one $\overline{SCK0}$ clock cycle between the start of reception and the first fall of $\overline{SCK0}$.

If reception is disabled during the receive operation (by clearing (to 0) the CRXE bit), $\overline{SCK0}$ clock output is stopped and the receive operation is discontinued on the next rise of $\overline{SCK0}$. In this case an interrupt request (INTCSI) is not generated, and the contents of the SIO register will be undefined.

(2) When an external clock is selected as the serial clock

When reception starts, the SI0 pin data is fetched into shift register (SIO) in synchronization with the rise of the serial clock input to the $\overline{SCK0}$ pin after the start of reception. If reception has not started, shift operations are not performed even if the serial clock is input to the $\overline{SCK0}$ pin.

If reception is disabled during the receive operation (by clearing (to 0) the CRXE bit), the receive operation is discontinued and subsequent $\overline{SCK0}$ input is ignored. In this case an interrupt request (INTCSI) is not generated.

18.4.4 Operation When Transmission/Reception is Enabled

When the CTXE bit and CRXE bit of the clocked serial interface mode register (CSIM) are both set (to 1), a transmit operation and receive operation can be performed simultaneously (transmit/receive operation). The transmit/receive operation is started when the CRXE bit is changed from "0" to "1", or by performing a write to shift register (SIO).

When a transmit operation is started for the first time, the CRXE bit always changes from "0" to "1", and there is thus a possibility that the transmit/receive operation will start immediately, and undefined data will be output. The first transmit data should therefore be written to SIO beforehand when both transmission and reception are disabled (when the CTXE bit and CRXE bit are both cleared (to 0)), before enabling transmission/reception.

When transmission/reception is disabled (CTXE = CRXE = 0), the SO0 pin is in the output high impedance state.

(1) When the internal clock is selected as the serial clock

When transmission/reception starts, the serial clock is output from the $\overline{\text{SCK0}}$ pin, data is output in sequence from shift register (SIO) to the SO0 pin in synchronization with the fall of the serial clock, and SI0 pin data is shifted in order into SIO in synchronization with the rise of the serial clock.

There is a delay of up to one $\overline{\text{SCK0}}$ clock cycle between the start of transmission and the first fall of $\overline{\text{SCK0}}$.

If either transmission or reception is disabled during the transmit/receive operation, only the disabled operation is discontinued. If transmission only is disabled, the SO0 pin becomes output high impedance. If reception only is disabled, the contents of the SIO register will be undefined.

If transmission and reception are disabled simultaneously, $\overline{\text{SCK0}}$ clock output is stopped and the transmit and receive operations are discontinued on the next rise of $\overline{\text{SCK0}}$. When transmission and reception are disabled simultaneously, the contents of SIO are undefined, an interrupt request (INTCSI) is not generated, and the SO0 pin becomes output high impedance.

(2) When an external clock is selected as the serial clock

When transmission/reception starts, data is output in sequence from shift register (SIO) to the SO0 pin in synchronization with the fall of the serial clock input to the $\overline{\text{SCK0}}$ pin after the start of transmission/reception, and SI0 pin data is shifted in order into SIO in synchronization with the rise of the serial clock. If transmission/reception has not started, shift operations are not performed and the SO0 pin output level does not change even if the serial clock is input to the $\overline{\text{SCK0}}$ pin.

If either transmission or reception is disabled during the transmit/receive operation, only the disabled operation is discontinued. If transmission only is disabled, the SO0 pin becomes output high impedance. If reception only is disabled, the contents of the SIO register will be undefined.

If transmission and reception are disabled simultaneously, the transmit and receive operations are discontinued and subsequent $\overline{\text{SCK0}}$ input is ignored. When transmission and reception are disabled simultaneously, the contents of SIO are undefined, an interrupt request (INTCSI) is not generated, and the SO0 pin becomes output high impedance.

18.4.5 Corrective Action in Case of Slippage of Serial Clock and Shift Operations

When an external clock is selected as the serial clock, there may be slippage between the number of serial clocks and shift operations due to noise, etc. In this case, since the serial clock counter is initialized by disabling both transmit operations and receive operations (by clearing (to 0) the CTXE bit and CRXE bit), synchronization of the shift operations and the serial clock can be restored by using the first serial clock input after reception or transmission is next enabled as the first clock.

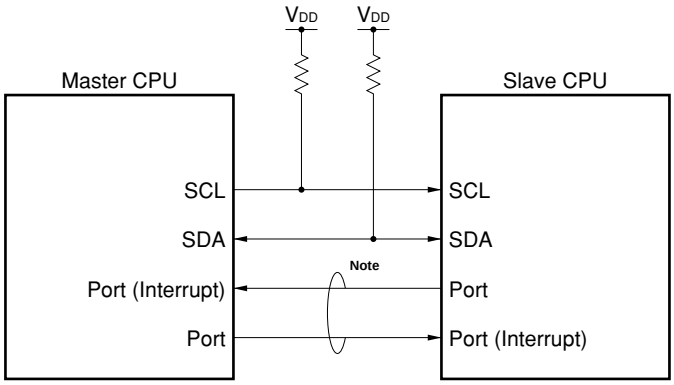
18.5 2-WIRE SERIAL I/O MODE

The 2-wire serial I/O mode an support any communication format depending on the program.

Basically, communication is performed by using two lines, a serial clock line (SCL) and a serial data input/output line (SDA). Generally, a handshake line is necessary for checking the communication status.

In the 2-wire serial I/O mode, both the SCL and SDA pins serves as N-ch open-drain output pins in the output mode. Therefore, connect external pull-up resistors to these pins.

Figure 18-7 Example of Configuration of 2-Wire Serial I/O System



Note Handshake line

18.5.1 Basic Operation Timing

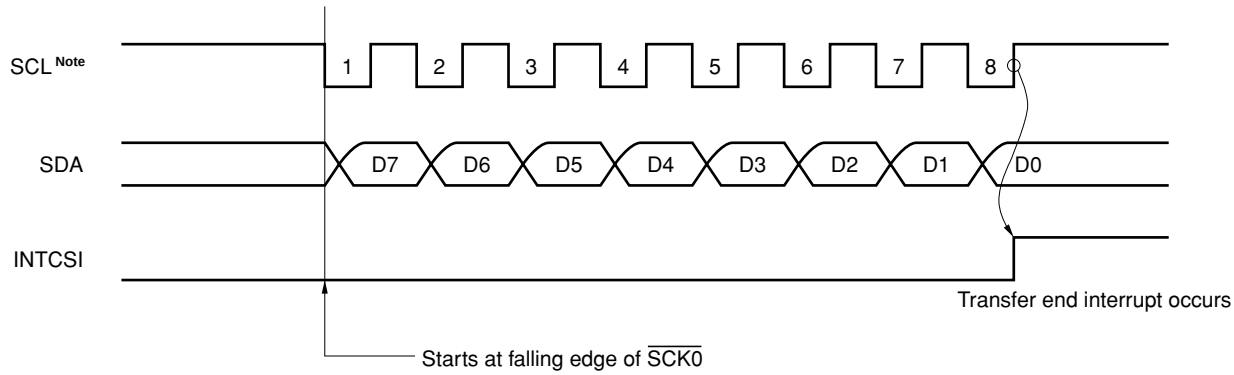
In the 2-wire serial I/O mode, data are transferred/received in 8-bit units. Data are transferred/received in synchronization with the serial clock in 1-bit units with the MSB first.

Transmit data is output at the falling edge of SCL. Receive data is sampled at the rising edge of SCL.

An interrupt request (INTCSI) is generated at the eighth rising edge of SCL.

When SCL is used as the internal clock, output of SCL is stopped at the eighth rising edge of SCL and SCL is kept high until transfer or reception of the next data is started.

Figure 18-8 Timing in 2-Wire Serial I/O Mode



Note Master CPU: output
Slave CPU: input

The pin specified as the serial data bus of the SDA pin serves as an N-ch open-drain I/O pin and must be externally pulled up by resistor.

Because SDA outputs the status of the SO latch, the output status of the SDA pin can be manipulated by setting the SPT and STT bits. However, do not set these bits during serial transfer.

When SCL is used as the internal clock (when used as the master CPU), the SCL pin serves as an N-ch open-drain output pin and must be externally pulled up by resistor.

18.5.2 Operation When Transmission Only is Enabled

A transmit operation is performed when the CTXE bit of the clocked serial interface mode register (CSIM) is set (to 1). The transmit operation starts when a write to the shift register (SIO) is performed while the CTXE1 bit is set (to 1).

When the CTXE bit is cleared (to 0), the SDA pin is in the output high impedance state.

(1) When the internal clock is selected as the serial clock

When transmission starts, the serial clock is output from the SCL pin and data is output in sequence from SIO to the SDA pin in synchronization with the fall of the serial clock.

There is a delay of up to one SCL clock cycle between the start of transmission and the first fall of SCL.

If transmission is disabled during the transmit operation (by clearing (to 0) the CTXE bit), SCL clock output is stopped and the transmit operation is discontinued on the next rise of SCL. In this case an interrupt request (INTCSI) is not generated, and the SDA pin becomes output high impedance.

(2) When an external clock is selected as the serial clock

When transmission starts, data is output in sequence from SIO to the SDA pin in synchronization with the fall of the serial clock input to the SCL pin after the start of transmission. If transmission has not started, shift operations are not performed and the SDA pin output level does not change even if the serial clock is input to the SCL pin.

If transmission is disabled during the transmit operation (by clearing (to 0) the CTXE bit), the transmit operation is discontinued and subsequent SCL input is ignored. In this case an interrupt request (INTCSI) is not generated, and the SDA pin becomes output high impedance.

(3) Detecting transmit error

Because the status of the serial data (SDA) being transmitted is also input to the SIO of the device that is sending the data in the 2-wire serial I/O mode, the data of the SIO before and after transmission can be compared and it can be judged, if the two data are different, that a transmit error has occurred.

18.5.3 Operation When Reception Only is Enabled

A receive operation is performed when the CRXE bit of the clocked serial interface mode register (CSIM) is set (to 1). The receive operation starts when the CRXE changes from “0” to “1”, or when a read from shift register (SIO) is performed.

(1) When the internal clock is selected as the serial clock

When reception starts, the serial clock is output from the SCL pin and the SDA pin data is fetched in sequence into shift register (SIO) in synchronization with the rise of the serial clock.

There is a delay of up to one SCL clock cycle between the start of reception and the first fall of SCL.

If reception is disabled during the receive operation (by clearing (to 0) the CRXE bit), SCL clock output is stopped and the receive operation is discontinued on the next rise of SCL. In this case an interrupt request (INTCSI) is not generated, and the contents of the SIO register will be undefined.

(2) When an external clock is selected as the serial clock

When reception starts, the SDA pin data is fetched into shift register (SIO) in synchronization with the rise of the serial clock input to the SCL pin after the start of reception. If reception has not started, shift operations are not performed even if the serial clock is input to the SCL pin.

If reception is disabled during the receive operation (by clearing (to 0) the CRXE bit), the receive operation is discontinued and subsequent SCL input is ignored. In this case an interrupt request (INTCSI) is not generated.

18.5.4 Operation When Transmission/Reception is Enabled

When the CTXE bit and CRXE bit of the clocked serial interface mode register (CSIM) are both set (to 1), a transmit operation and receive operation can be performed simultaneously (transmit/receive operation). The transmit/receive operation is started when the CRXE bit is changed from “0” to “1”, or by performing a write to shift register (SIO).

When a transmit operation is started for the first time, the CRXE bit always changes from “0” to “1”, and there is thus a possibility that the transmit/receive operation will start immediately, and undefined data will be output. The first transmit data should therefore be written to SIO beforehand when both transmission and reception are disabled (when the CTXE bit and CRXE bit are both cleared (to 0)), before enabling transmission/reception.

When transmission/reception is disabled (CTXE = CRXE = 0), the SDA pin is in the output high impedance state.

(1) When the internal clock is selected as the serial clock

When transmission/reception starts, the serial clock is output from the SCL pin, data is output in sequence from shift register (SIO) to the SDA pin in synchronization with the fall of the serial clock, and SDA pin data is shifted in order into SIO in synchronization with the rise of the serial clock.

There is a delay of up to one SCL clock cycle between the start of transmission and the first fall of SCL.

If either transmission or reception is disabled during the transmit/receive operation, only the disabled operation is discontinued. If transmission only is disabled, the SDA pin becomes output high impedance. If reception only is disabled, the contents of the SIO register will be undefined.

If transmission and reception are disabled simultaneously, SCL clock output is stopped and the transmit and receive operations are discontinued on the next rise of SCL. When transmission and reception are disabled simultaneously, the contents of SIO are undefined, an interrupt request (INTCSI) is not generated, and the SDA pin becomes output high impedance.

(2) When an external clock is selected as the serial clock

When transmission/reception starts, data is output in sequence from shift register (SIO) to the SDA pin in synchronization with the fall of the serial clock input to the SCL pin after the start of transmission/reception, and SDA pin data is shifted in order into SIO in synchronization with the rise of the serial clock. If transmission/reception has not started, shift operations are not performed and the SDA pin output level does not change even if the serial clock is input to the SCL pin.

If either transmission or reception is disabled during the transmit/receive operation, only the disabled operation is discontinued. If transmission only is disabled, the SDA pin becomes output high impedance. If reception only is disabled, the contents of the SIO register will be undefined.

If transmission and reception are disabled simultaneously, the transmit and receive operations are discontinued and subsequent SCL input is ignored. When transmission and reception are disabled simultaneously, the contents of SIO are undefined, an interrupt request (INTCSI) is not generated, and the SDA pin becomes output high impedance.

(3) Detecting transmit error

Because the status of the serial data (SDA) being transmitted is also input to the SIO of the device that is sending the data in the 2-wire serial I/O mode, the data of the SIO before and after transmission can be compared and it can be judged, if the two data are different, that a transmit error has occurred.

18.5.5 Corrective Action in Case of Slippage of Serial Clock and Shift Operations

When an external clock is selected as the serial clock, there may be slippage between the number of serial clocks and shift operations due to noise, etc. In this case, since the serial clock counter is initialized by disabling both transmit operations and receive operations (by clearing (to 0) the CTXE bit and CRXE bit), synchronization of the shift operations and the serial clock can be restored by using the first serial clock input after reception or transmission is next enabled as the first clock.

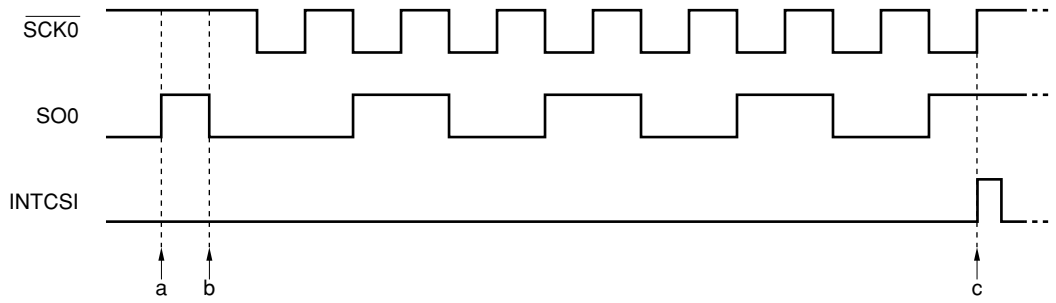
18.6 CAUTIONS

(1) When changing from “CTXE = 0, CRXE = 1” to “CTXE = 1, CRXE = 0”, and when changing from “CTXE = 1, CRXE = 0” to “CTXE = 0, CRXE = 1”, ensure that this is not done with a single instruction, as this will result in malfunction of the serial clock counter, and the first communication after the change will finish in fewer than 8 bits. Instead, two instructions should be used as shown below.

Example To change “CTXE = 1, CRXE = 0” to “CTXE = 0, CRXE = 1”

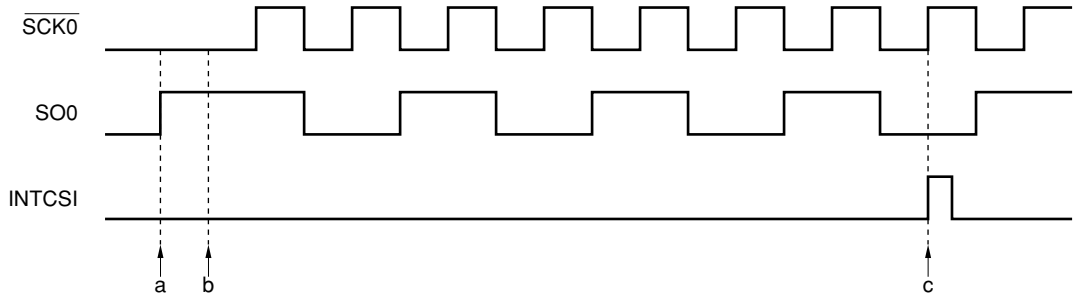
```
CLR1 CTXE
SET1 CRXE
```

★ (2) In the μPD784038, 784038Y Subseries, data is output when data is written to the shift register.



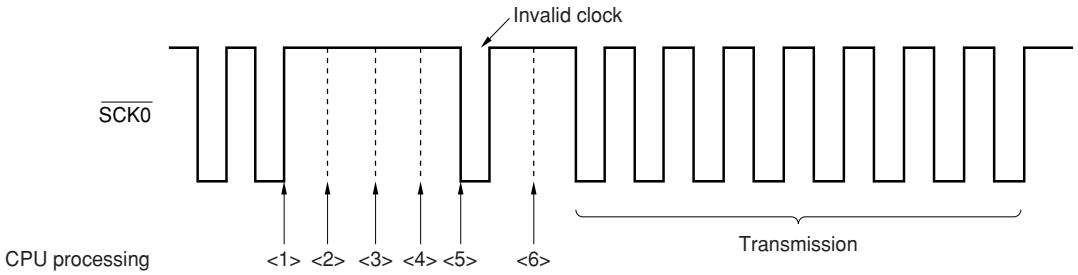
a: Setting the transmission enable bit (CTXE ← 1)
b: Data (55H) written to shift register (SIO ← 55H)
c: Generation of transfer completion interrupt request

★ (3) In the μPD784038, 784038Y Subseries, the serial clock counter is incremented by one when the shift register is written after transmission is enabled and if the $\overline{\text{SCK0}}$ pin is low level. Therefore, if transmission is started while the external clock is selected as the serial clock and the $\overline{\text{SCK0}}$ pin is low level, a transmission completion interrupt request is generated at the 7th rising of the serial clock.



a: Setting the transmission enable bit (CTXE ← 1)
b: Data (AAH) written to shift register (SIO ← AAH)
c: Generation of transfer completion interrupt request

★ (4) When master transmission and slave reception are executed alternately in 3-wire serial I/O mode, an invalid serial clock may be output from the $\overline{SCK0}$ pin. (Not applicable to $\overline{SCK1}$ and $\overline{SCK2}$ pins)
Set $\overline{SCK0}$ to high-level output port mode in the period in which the invalid clock (1 system clock max.) indicated in attachment 2 may be generated when alternately executing master transmission and slave reception.



<1> End of slave reception
<2> SIO read
<3> Reception disabled (CRXE = 0)
<4> Transmit clock selection
<5> Master transmission enabled (CTXE = 1)
<6> SIO write

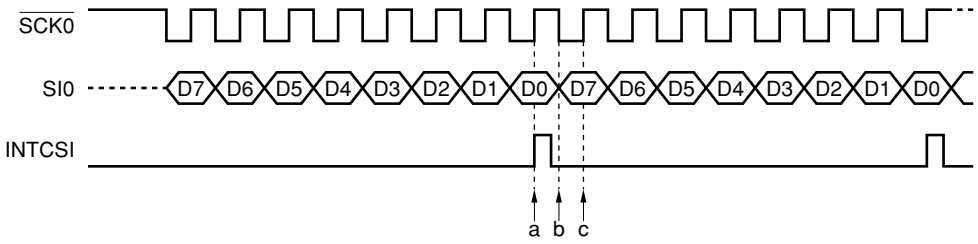
- Preventive program example (when TM3/2 is selected as the internal clock)

```
SET1    P3.2        ; P32 = 1
CLR1    PMC3.2      ; SCK0 pin: I/O port mode
CLR1    CRXE        ; Reception disabled
SET1    CSIM.0      ; Slave → Master
SET1    CTXE        ; Transmission enabled
  :                 ; Wait for at least 1 serial clock
SET1    PMC3.2      ; SCK0 pin: SCK0 I/O mode
MOV     SIO,A       ; Write to SIO register
```

★ (5) When data is successively transmitted from the transmission side in 3-wire serial I/O mode, the second and subsequent receive data may be undefined under the following conditions <1> and <2>.

<1> Read from the shift register (SIO) is not completed in the period from reception completion (**a** in the figure below) to the next fall of the serial clock ($\overline{SCK0}$) (**b** in the figure below)

<2> The reception enable bit is cleared in the period from reception completion (**a** in the figure below) to the next rise of the serial clock ($\overline{SCK0}$) (**c** in the figure below), and the reception enable flag cannot be set after the shift register (SIO) is read



Implement the following workaround to prevent this bug.

- Read the shift register after reception completion (**a** in the figure above) by the next fall of the serial clock (**b** in the figure above).
- Clear the reception enable bit after reception completion (**a** in the figure above) by the next rise of the serial clock (**c** in the figure above), read the shift register, and set the reception enable flag.

CHAPTER 19 I²C BUS MODE (μPD784038Y SUBSERIES ONLY)

19.1 OUTLINE OF FUNCTIONS

- **I²C (INTER IC) bus mode (MSB First)**

  The I²C bus mode is an interface for communicating with devices that conform with the I²C bus format. It allows 8-bit data transfer to several devices using 2 lines: a serial clock line (SCL) and a serial data bus (SDA).

  In the I²C bus mode, the master can output start conditions, data, and stop conditions to a slave on the serial data bus. A slave automatically detects the data received by means of hardware. This function can simplify the I²C bus control part in application programs.

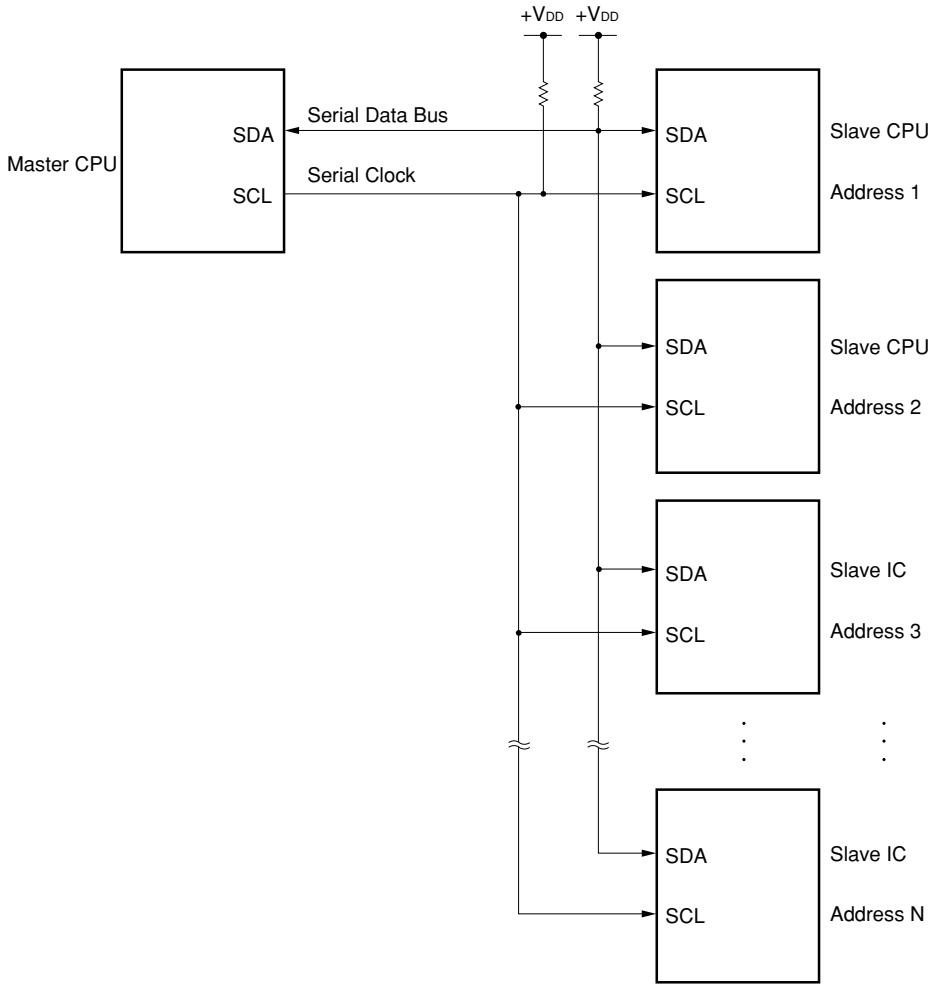
  The conventional serial I/O method being limited to a data transfer function, a lot of ports and wiring are required in order to discriminate chip select signals and command/data and recognize busy states when the serial bus is configured with several devices connected. In addition, performing these controls with software places a considerable load on software.

  In the I²C bus mode, the serial bus can be configured with two signal lines, a serial clock line (SCL) and a serial data bus (SDA). Therefore, use of this mode is effective to reduce the number of microcontroller ports, wiring, and complicated routing in the circuit board.

  The I²C bus mode is used for performing single master and slave operations through the I²C bus.

  For further information, refer to **19.4 I²C Bus Mode Functions**

Figure 19-1 Example of Serial Bus Configuration Using I²C Bus



19.2 CONFIGURATION

The block diagram of the clock-synchronous serial interface (CSI) in the I²C bus mode is shown in Figure 19-2.

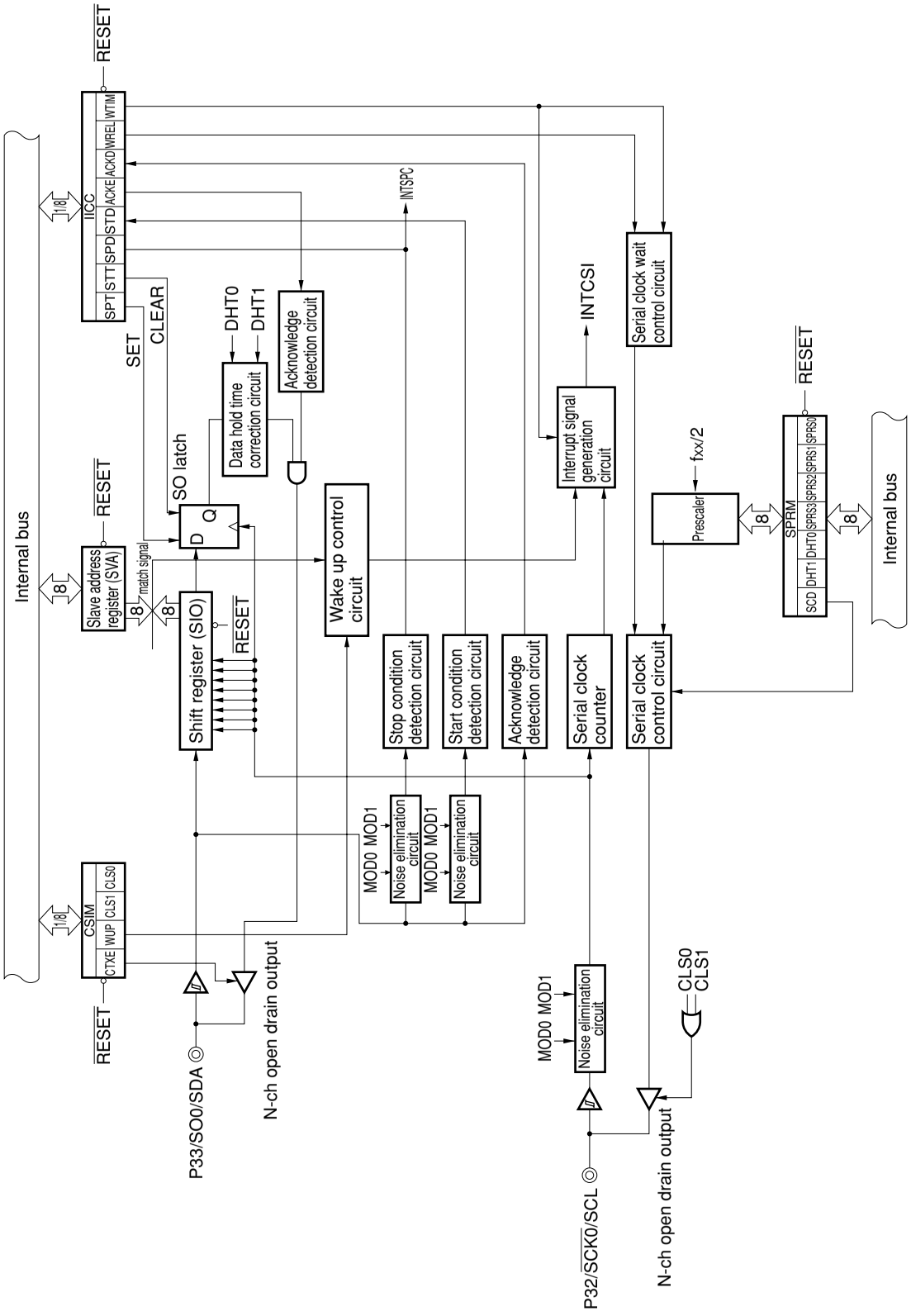


Figure 19-2 Block Diagram of Clock-Synchronous Serial Interface (In I²C Bus Mode)

(1) Shift register (SIO)

The SIO register converts 8-bit serial data to 8-bit parallel data or vice versa. It is used both for transmission and reception.

Actual transmission and reception is controlled by writing/reading to/from the SIO register.

Reading and writing is performed with 8-bit manipulation instructions. $\overline{\text{RESET}}$ input causes the contents of this register to become undefined.

(2) Slave address register (SVA)

The slave address register is used to set the address of this microcomputer when it is used as a slave.

This register can also be used to check the direction of transmission.

(3) SO latch

The SO latch serves to retain the SDA pin output level.

It can be controlled by software.

(4) Wake up control circuit

The wake up control circuit is used when the microcomputer is employed as a slave to control whether to always generate an interrupt or generate an interrupt only when the address set to the slave address register (SVA) matches the reception address.

(5) Serial clock selector

The serial clock selector selects the serial clock to be used.

(6) Serial clock counter

The serial clock counter counts the serial clocks output during transmission and reception to check whether transmission or reception has been performed.

(7) Interrupt signal generator

The interrupt signal generator controls the generation of interrupt request signal.

Interrupt requests are generated with the timing shown in Table 19-1 based on the setting of bit 7 (WTIM) in the I²C bus control register (IICC) and bit 5 (WUP) in the clock synchronous serial interface mode register (CSIM).

(8) Serial clock control circuit

This circuit controls the serial clock supplied to the shift register (SIO). In addition, it controls the clock output to the SCL pin if the internal clock is used.

(9) Serial clock wait control circuit

The serial clock wait control circuit controls wait timing.

(10) Acknowledge output circuit, Stop condition detection circuit, Start condition detection circuit, Acknowledge detection circuit

These circuits output and detect various control signals.

(11) Data hold time correction circuit

This circuit generates the data hold time at the falling edge of the serial clock.

This circuit is controlled by setting the data hold time specification bit (DHT0, DHT1) in the prescaler mode register (SPRM) for serial clock with the external oscillation frequency.

19.3 CONTROL REGISTER

19.3.1 Clocked Serial Interface Mode Register (CSIM)

CSIM is an 8-bit register used to specify the serial interface operation mode, serial clock, wakeup function, and so on. It is read/written with 8-bit manipulation instructions. The format of the CSIM is shown in Figure 19-3.

$\overline{\text{RESET}}$ input clears the contents of this register to 00H.

Figure 19-3 Clocked Serial Interface Mode Register (CSIM) Format

	⑦	6	⑤	4	3	2	1	0	Address	After Reset	R/W
CSIM	CTXE	0	WUP	0	1	1	CLS1	CLS0	0FF82H	00H	R/W

CLS1	CLS0	Serial Clock Specification			$\overline{\text{SCK0}}$, SCL pins
0	0	External clock	Slave		Input
1	0	Internal clock	Master	SPRS	Output
Other than above		Setting prohibited			

WUP	Wakeup Function Control
0	An interrupt request is generated every time address and data transfer is completed.
1	An interrupt request (INTCSI) is generated upon reception of this microcontroller's address, and then an interrupt request (INTCSI) is generated every time data transfer is completed.

CTXE	Reception/Transmission
0	Disabled
1	Enabled

19.3.2 I²C Bus Control Register (IICC)

IICC is an 8-bit register consisting of a bit that controls the serial bus status.

Reading and writing is performed with 8-bit manipulation instructions. During read operation, the write-only bit is “0”. Figure 19-4 shows the format of IICC. Do not write data to IICC during transmission, reception, and transmission/reception.

$\overline{\text{RESET}}$ input clears the contents of this register to 00H.

Figure 19-4 I²C Bus Control Register (IICC) Format (1/2)

	⑦	⑥	⑤	④	③	②	①	⓪	Address	After Reset	R/W
IICC	WTIM	WREL	ACKD	ACKE	STD	SPD	STT	SPT	0FF80H	00H	R/W

Wait Timing Setting Bit (R/W)

This bit controls the interrupt generation timing and wait timing control during data reception. Rewrite this bit only when transmission/reception is prohibited (CTXE = 0).

WTIM	0	8-clock wait: Interrupt request (INTCSI) is generated at eighth falling edge of SCL. • When used as master : After 8 clocks are output, changes SCL output to low level and waits. • When used as slave : After 8 clocks are input, changes SCL pin to low level and generates a wait request.
	1	9-clock wait: Interrupt request (INTSCI) is generated at ninth falling edge of SCL. • When used as master : After 9 clocks are output, changes SCL output to low level and waits. • When used as slave : After 9 clocks are input, changes SCL pin to low level and generates a wait request.

Wait Cancellation Trigger Bit (W)

WREL	Wait status is canceled (SCL is set to high level) when WREL = 1.

Acknowledge Detection Flag (R)

ACKD	Clear Conditions (ACKD = 0)	Set Conditions (ACKD = 1)
	① Upon detection of acknowledge signal ($\overline{ACK}$), at wait cancellation (WREL = 1 or S10 write or SPT = 1) ② CTXE = 0 ③ At reset input	Upon detection of acknowledge signal ($\overline{ACK}$) (When SDA is low level at ninth rising edge of SCL)

Acknowledge Output Enable (R/W)

This bit enables output of the acknowledge signal upon reception of data.

ACKE	0	Disables automatic output of acknowledge signal. Used during transmission, or when 8-clock wait is selected. However, when WUP = 1 during address reception, operation is as follows. • Upon reception of microcontroller address : Automatically outputs acknowledge signal in synchronization with ninth falling edge. • Upon reception of other than microcontroller address : Does not automatically output acknowledge signal.
	1	When 8-clock wait is selected : By making ACKE = 1 before performing wait control, an acknowledge signal is output in synchronization with the eighth falling edge of SCL. When 9-clock wait is selected : By making ACKE = 1 beforehand, an acknowledge signal is automatically output in synchronization with the eighth falling edge of SCL. However, when WUP = 1 during address reception, operation is as follows. • Upon reception of microcontroller address : Automatically outputs acknowledge signal. • Upon reception of other than microcontroller address : Does not automatically output acknowledge signal.

Following the eighth falling edge when 8-clock wait is selected, if the ACKE bit is changed from 0 to 1, $\overline{ACK}$ is output with the timing set in ACKE.

Figure 19-4 I²C Bus Control Register (IICC) Format (2/2)

	⑦	⑥	⑤	④	③	②	①	⓪	Address	After Reset	R/W
IICC	WTIM	WREL	ACKD	ACKE	STD	SPD	STT	SPT	0FF80H	00H	R/W

Start Condition Detection Flag (R)

STD	Clear Conditions (STD = 0)	Set Conditions (STD = 1)
	① 1 Upon wait cancellation following detection of start condition (WREL = 1 or SI0 write Note or SPT-1) ② CTXE = 0 ③ At reset input **Note** Except during address write when microcontroller is used as master.	• When WUP = 0 : Upon detection of start condition • When WUP = 1 : Upon detection of microcontroller address

Stop Condition Detection Flag (R)

SPD	Clear Conditions (SPD = 0)	Set Conditions (SPD = 1)
	① 1 Upon detection of start condition ② CTXE = 0 ③ At reset input	Upon detection of stop condition

Start Condition Trigger Bit (W)

STT	By making STT = 1 when SCL and SDA are high level Note 1, the S0 latch is cleared (to 0). After the S0 latch is cleared, SCL becomes low level, and the STT bit is automatically cleared (to 0). **Notes 1.** The level of SCL can be checked by using the P32/SCL pin as an input pin (PM32 = 1) and reading SCL Note 2. The level of SDA can be checked by using the P33/SDA pin as an input pin (PM33 = 1) and reading SDA Note 2. **2.** SCL and SDA are defined as reserved words when using an NEC Electronics assembler, and as sfr variables using the #pragma sfr command in C compiler. **Cautions 1. Even when STT is set (to 1) when SCL and SDA are low level, the start condition is not output (after STT is set to 1, SCL the start condition is not output even when SCL becomes high level.** **2. After STT is set (to 1), be sure to write address to SIO after executing one or more instructions with NOP or the like.**

Stop Condition Trigger Bit (W)

SPT	By making SPT = 1, the SO latch is cleared (to 0), and the SCL becomes high level. After SCL becomes high level, the SO latch is set (to 1). After the SO latch is set, the SPT bit is automatically cleared (to 0).

19.3.3 Prescaler Mode System for Serial Clock (SPRM)

SPRM is an 8-bit register used to specify the serial clock and duty when the data hold time in relation to the falling edge of SCL and the serial clock are specified for the internal clock (CLS1 bit = 1, CLS0 bit 0).

This register is read/written with 8-bit manipulation instructions. Figure 19-5 shows the format of SPRM.

$\overline{\text{RESET}}$ input sets the contents of this register to 04H.

Rewrite SPRM only when transmission/reception is disabled (CTXE = 0).

Figure 19-5 Prescaler Mode Register for Serial Clock (SPRM) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SPRM	SCD	DHT1	DHT0	0	SPRS3	SPRS2	SPRS1	SPRS0	0FF81H	04H	R/W

SPRS3	SPRS2	SPRS1	SPRS0	Serial Clock Specification
0	0	0	0	fxx/16
0	0	0	1	fxx/24
0	0	1	0	fxx/32
0	0	1	1	fxx/48
0	1	0	0	fxx/64
0	1	0	1	fxx/96
0	1	1	0	fxx/128
0	1	1	1	fxx/192
1	0	0	0	fxx/256
1	0	0	1	fxx/384

DHT1	DHT0	Data Hold Time Specification	
0	0	16 MHz < fxx ≤ 32 MHz	10/fxx to 12/fxx
0	1	8 MHz < fxx ≤ 16 MHz	5/fxx to 7/fxx
1	×	4 MHz < fxx ≤ 8 MHz	3/fxx to 5/fxx

SCD	Serial Clock Duty Specification
0	For standard mode
1	For high-speed mode

19.3.4 Slave Address Register (SVA)

SVA is an 8-bit register used to specify the microcomputer's address when it is used as a slave device.

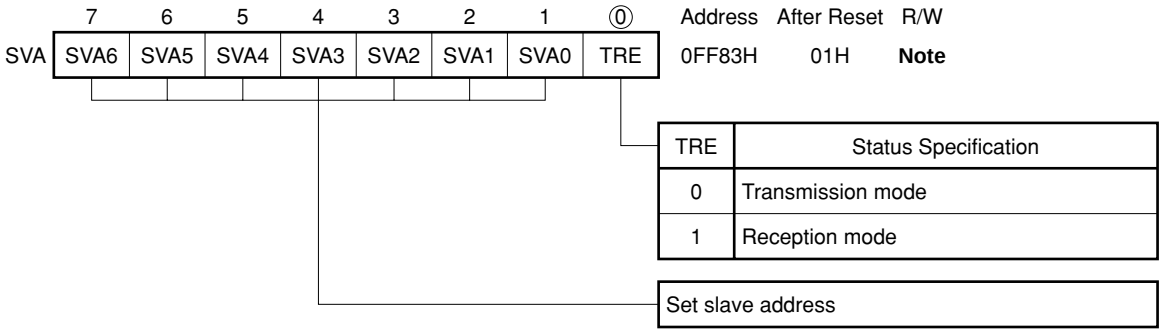
Bit 0 of SVA (TRE bit) can be used to check whether transmission or reception is performed.

Bits 1 to 7 are read/written with an 8-bit manipulation instruction. Bit 0 can only be read, using an 8-bit manipulation instruction and a bit manipulation instruction.

Figure 19-5 shows the format of SVA.

$\overline{\text{RESET}}$ input set the contents of this register to 01H.

Figure 19-6 Slave Address Register (SVA) Format



Note Bit 0: Only Read (R) is possible.
Bits 1 to 7: Read/Write (R/W) are possible.

19.4 I²C BUS MODE FUNCTION

19.4.1 Pin Configuration

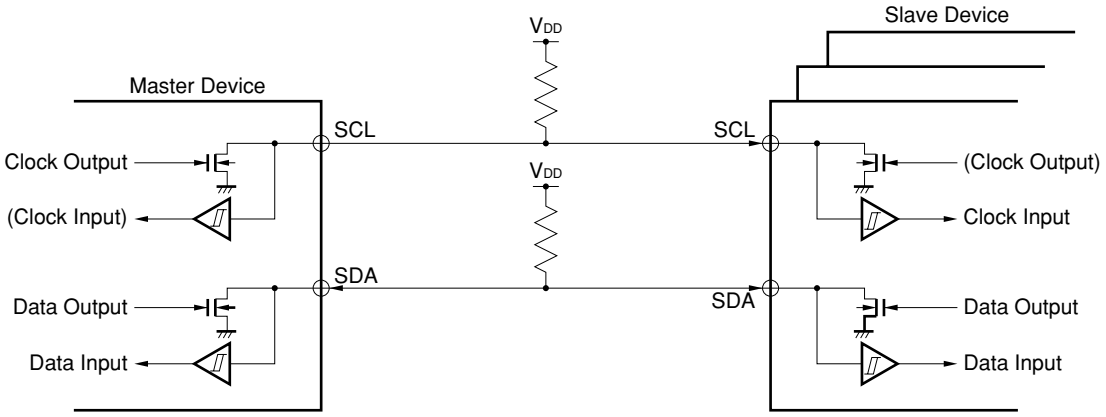
The serial clock pin (SCL) and serial data bus pin (SDA) are configured as follows:

(1) SCL ····· Pin that inputs/outputs serial clock
- Master : N-ch open-drain output
- Slave : Schmitt input

(2) SDA ····· Serial data input/output dual pin N-ch open-drain output and Schmitt input for both master and slave.

Because both the serial clock and serial data bus are N-ch open drain output, they must be connected to external pull-up resistors.

Figure 19-7 Pin Configuration



19.4.2 Functions

The following function is available in the I²C bus mode of μPD784038Y.

(1) Automatic identification of serial data

The “start condition”, “data” and “stop condition” on the serial data bus are automatically detected.

(2) Chip select by address

The master can select specific slave device from those connected to the I²C bus by transmitting a slave address and communicate with that slave.

(3) Wake-up function

When a slave operates, it generates an interrupt only when the address it has received from the master coincides with the value of the slave address register (SVA). Therefore the slave on the I²C bus other than the one selected by the master can operate independently of the serial communication.

(4) Acknowledge signal ($\overline{\text{ACK}}$) control function

The acknowledge signal that is used to check whether serial communication has been correctly executed can be controlled during the master and slave operations.

(5) Wait signal ($\overline{\text{WAIT}}$) control function

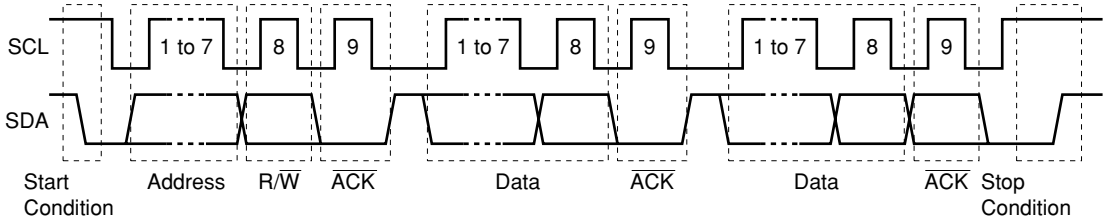
A slave device can control the wait signal that indicates the busy status of the slave.

19.5 DEFINITION AND CONTROL METHOD OF THE I²C BUS

The following describes the serial data communication format of the I²C bus and the meanings of the signals used.

Figure 19-8 shows the transfer timing of the “start condition”, “data”, and “stop condition” output to the I²C serial data bus.

Figure 19-8 Serial Data Transfer Timing on I²C Bus



The start condition, slave address, and stop condition are output by the master.

The acknowledge signal ($\overline{ACK}$) is output by either the master or slave (usually, this signal is output by the side that receives 8-bit data).

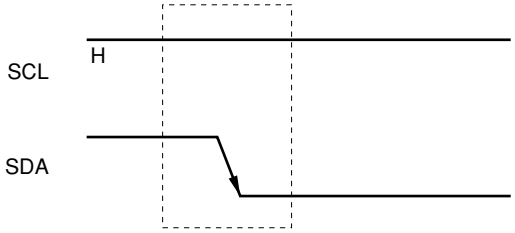
The serial clock (SCL) is continuously output by the master. However, the slave can extend the SCL low level period, thus a wait can be inserted.

19.5.1 Start Condition

The start condition is output to the serial data bus when SDA pin goes low while the SCL pin is high.

Therefore, the start condition of the SCL and SDA pins is a signal output by the master when the master starts serial transfer to a slave.

Figure 19-9 Start Condition



The start condition is output by making STT = 1 when SCL is being high. When the start condition is executed, STD is set (STD = 1).

After STT is set (to 1), be sure to write address to SIO after executing one or more instructions with NOP or the like.

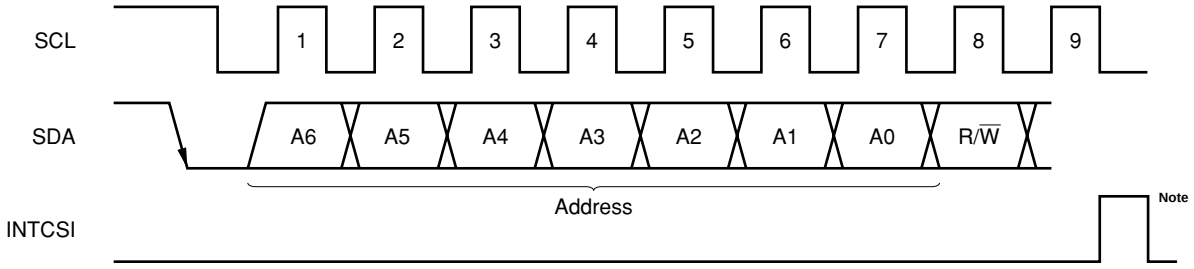
19.5.2 Addresses

The 7-bit data following the start condition is defined to be an address.

An address is 7 bit of data output by the master to select a specific slave from those connected to the bus line. Therefore, all the slaves on the bus line must have a different address.

A slave detects the start condition by hardware and checks whether the 7-bit data output by the master coincides with the value of the slave address register (SVA) of the slave. If the 7-bit data coincides with the value of the slave address register of the slave, is selected. After that, communication takes place between the master and this slave, until the master transmits a start or stop condition.

Figure 19-10 Address



Note In slave, when WUP = 1, if an address other than own address is received, INTCSI does not occur.

The slave address and the transfer direction explained in section **19.5.3 Transfer Direction Specification** are written to ISO simultaneously, and then an address is output. The received address is also written into ISO with specification of transfer direction.

Slave address, however, is assigned to the higher 7 bits of SIO.

After STT is set (to 1), be sure to write address to SIO after executing one or more instructions with NOP or the like

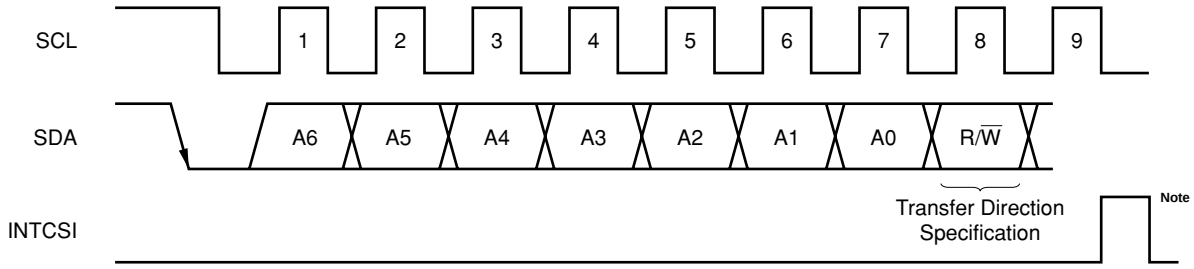
19.5.3 Transfer Direction Specification

The master transmits the 1-bit data to specify the transfer direction following 7-bit address.

The transfer direction specification bit 0 indicates data transmission from the master to the slave.

On the other hand, the transfer specification bit 1 indicates data reception from the slave to the master.

Figure 19-11 Transfer Direction Specification



Note INTCSI is not generated if other than own address is received during WUP = 1 in slave operation.

Transfer direction specification bit is output by writing into SIO with address simultaneously.

In addition, the received direction is written not only into SIO with address but also into TRE bit (bit 0) in the slave address register (SVA) simultaneously.

The transfer direction is assigned to the lowest order bit in the SIO.

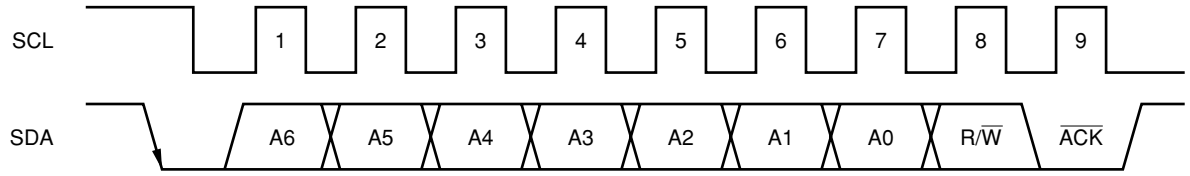
After the STT is set (to 1), at least one instruction should be executed using NOP, etc. before writing the transfer direction to the SIO.

19.5.4 Acknowledge Signal ($\overline{ACK}$)

The acknowledge signal is used to confirm that serial data has been received at transmission and reception sides.

The reception side returns the acknowledge signal each time it has received 8 bits of data. However, do not return $\overline{ACK}$ on receiving the last data when a start condition or stop condition is to be issued while the master receives data (refer to Figure 19-16). The transmission side checks whether the reception side has returned the acknowledge signal after it has transmitted 8-bit data. When the acknowledge signal has been returned, it is assumed that the 8-bit data has been correctly received, and the next processing is performed. If a slave does not return the acknowledge signal, it is not received the data correctly. Consequently, the master outputs a stop condition to abort transmission.

Figure 19-12 Acknowledge Signal



Remark When 8 clock wait : The acknowledge signal is output synchronized with the falling edge of the eighth clock of SCL by setting ACKE = 1 before the wait release.
When 9 clock wait : The acknowledge signal is output synchronized with the falling edge of the eighth clock of SCL by setting ACKE = 1 beforehand.

The acknowledge signal is output synchronized with the falling edge of the 8th clock of SCL by setting ACKE (to 1). In WUP = 1, however, acknowledge is automatically output synchronized with the falling edge of the 8th clock of SCL regardless of ACKE value when receiving own address and the acknowledged signal is not output when receiving other than own address.

- When 8-clock wait : The acknowledge signal is output synchronized with the falling edge of the 8th clock of SCL by setting ACKE = 1 before the wait release.
- When 9-clock wait : The acknowledge signal is output synchronized with the falling edge of the 8th clock of SCL by setting ACKE = 1 beforehand.

The following operate when address reception in WUP = 1.

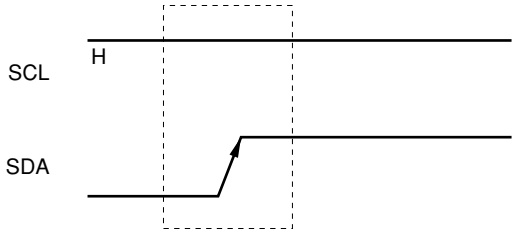
- Upon reception of relevant microcontroller address : Automatic output of acknowledge signal is performed.
- Upon reception of other than relevant microcontroller address : Automatic output of acknowledge signal is not performed.

19.5.5 Stop Condition

The stop condition is set when the SDA pin goes high while the SCL pin is high.

The stop condition is output by the master to the slave when serial transfer has been completed.

Figure 19-13 Stop Condition



The Stop condition is generated by setting SPT (to 1).

And when detecting the stop condition, SPD is set (to 1) and INTSPC is generated.

19.5.6 Wait Signal ($\overline{\text{WAIT}}$)

The wait signal is output by a slave to the master to indicate that the slave is waiting to send/receive data (wait status).

The slave informs the master that it is the wait status by making SCL pin low. When the slave is released from the wait status, the master can start the next transfer.

Figure 19-14 Wait Signal (1/2)

(1) When Eight Clocks Wait for Master and Slave
(master: transmission, Slave: reception, ACKE = 0)

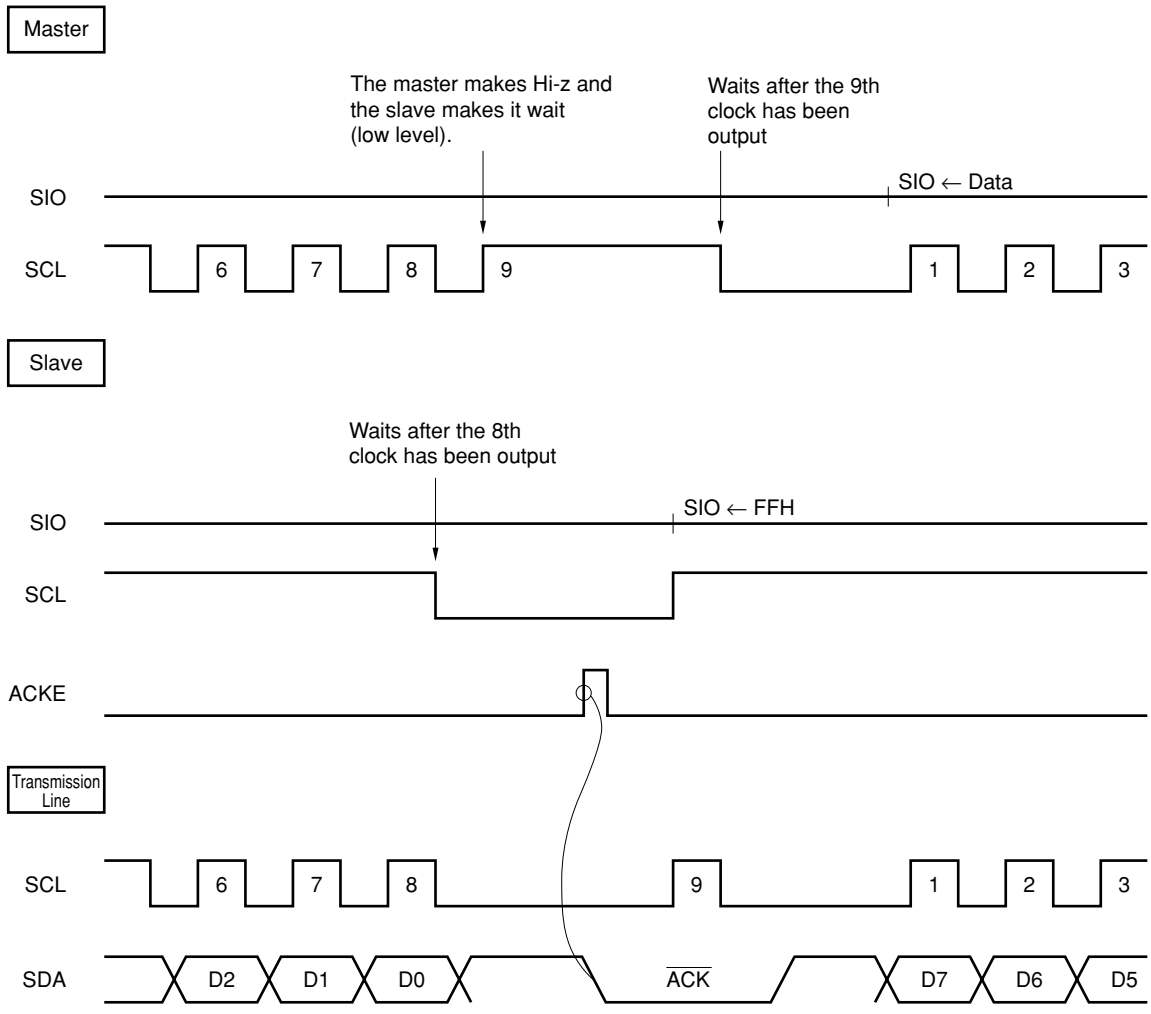
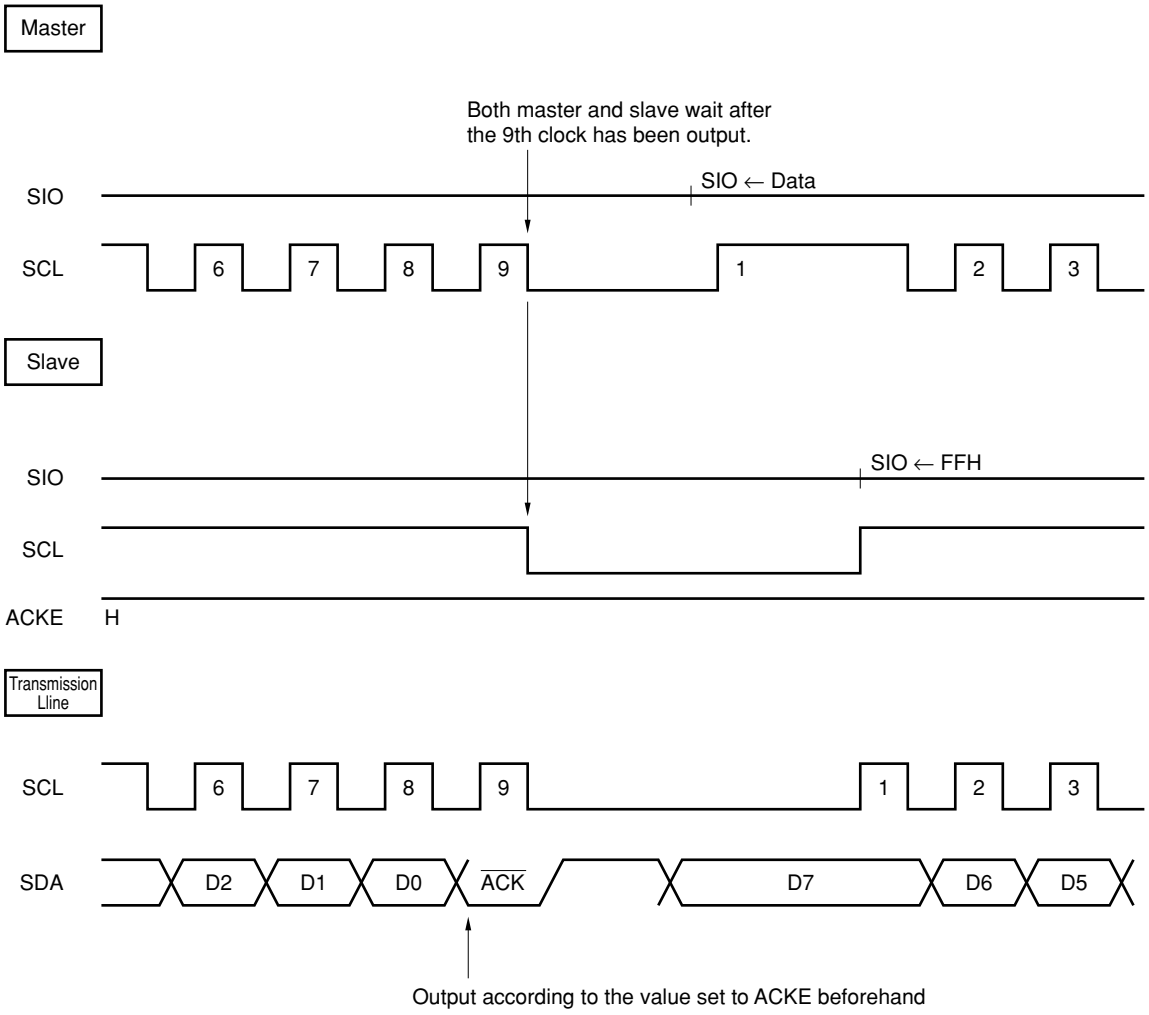


Figure 19-14 Wait Signal (2/2)

(2) When nine clocks wait for master and slave
(Master: Transmission, Slave: reception, ACKE = 1)



The wait is automatically generated when values are set to both WUP and WTIM.

The wait, however, is released when WREL = 1 is made, and address is written to SIO, or CTXE is cleared (to 0).

19.5.7 Interrupt Request (INTCSI) Generation Timing and Wait Control

The interrupt request is generated when the combination of the WUP bit of the clock synchronous serial interface (CSIM) and the WTIM bit of the I²C bus control register (IICC) are correspond with the timings shown in Figure 19-1 and also wait is controlled by the same manner.

Table 19-1 INTCSI Generation Timing and Wait Control

WUP	WTIM	Slave Operation			Master Operation		
		Address	Data Reception	Data Transmission	Address	Data Reception	Data Transmission
0	0	8	8	9	9	8	9
0	1	8	9	9	9	9	9
1	0	9 Note 1, 2	8 Note 2	9 Note 2	9	8	9
1	1	9 Note 1, 2	9 Note 2	9 Note 2	9	9	9

Notes 1. Only when the INTCSI signal and wait in slave operation in WUP = 1 and the address which has been set in the slave address register (SVA) matched, an interrupt request or wait are generated at the falling edge of the 9th clock. At this time, even if ACKE is set, $\overline{ACK}$ is output.

2. When WUP = 1, neither INTCSI nor wait in is generated if the SVA and received address have not matched.

Remark The numbers in the table refer to the number of serial clocks. Both the interrupt request and wait control synchronize with the falling edge of the serial clock.

(1) When address transmission and reception

- At slave operation : The interrupt timing and wait timing are determined by the WUP regardless of WTIM bit.
- At master operation : The interrupt timing and wait timing are generated at the falling edge of the 9th clock regardless of the WUP bit and WTIM bit.

(2) When data reception

- At master/slave operation : The interrupt timing and wait timing are determined by the WTIM bit regardless of WUM bit.

(3) When data transmission

- At master/slave operation : The interrupt timing and wait timing are generated at the falling edge of the 9th clock regardless of the WUP bit and WTIM bit.

(4) Wait release method

Three wait release methods are described as below.

- WREL = 1 in the I²C bus control register (IICC).
- Write operation of serial shift register (SIO) CTXE = 0 in clock synchronous serial interface mode register
- When 8 clock wait (WTIM = 0) is selected, $\overline{ACK}$ output level should be determined before wait release.

19.5.8 Interrupt Request Generation Timing

INTSPC is generated at the detection of the stop condition.

Processing to wait the generation of the next start condition is required in the INTSPC interrupt routine.

This is applied when used as a slave or WUP = 0.

19.5.9 Detection Method of Address Match

In the I²C bus mode, the master transmits the slave address, as a consequence, the specific slave device can be selected. The address match detection can be performed automatically with the hardware. If the self-addressed setting assigned to the slave address register in the wake up function specification bit (WUP) = 1, INTCSI interrupt request is generated only when the slave address transmitted from the master matches the address set to SVA. To identify the slave reception data as the address, the value of the start bit condition bit (STD) should be verified.

Caution When WUP = 0, the INTCSI interrupt request is generated, even though the self-addressed setting set to the slave address register (SVA) dose not match the data (address) received after the start condition.

19.5.10 Error Detection

In the I²C bus mode, the state of the serial bus (SDA) during transmission is transferred to the serial shift register of the device during transmission.

Thus, the transmission error is detected by comparing the data before the transmission with the data after the transmission. In this case, the transmission error occurred if the two data are different each other.

19.6 TIMING CHART

In the I²C bus mode, the master outputs the address to the serial bus, then, a target slave device is chosen among several slave devices.

The master transmits the TRE bit, which indicates the data transmission direction next to the slave address, and then, the transmission to the slave starts.

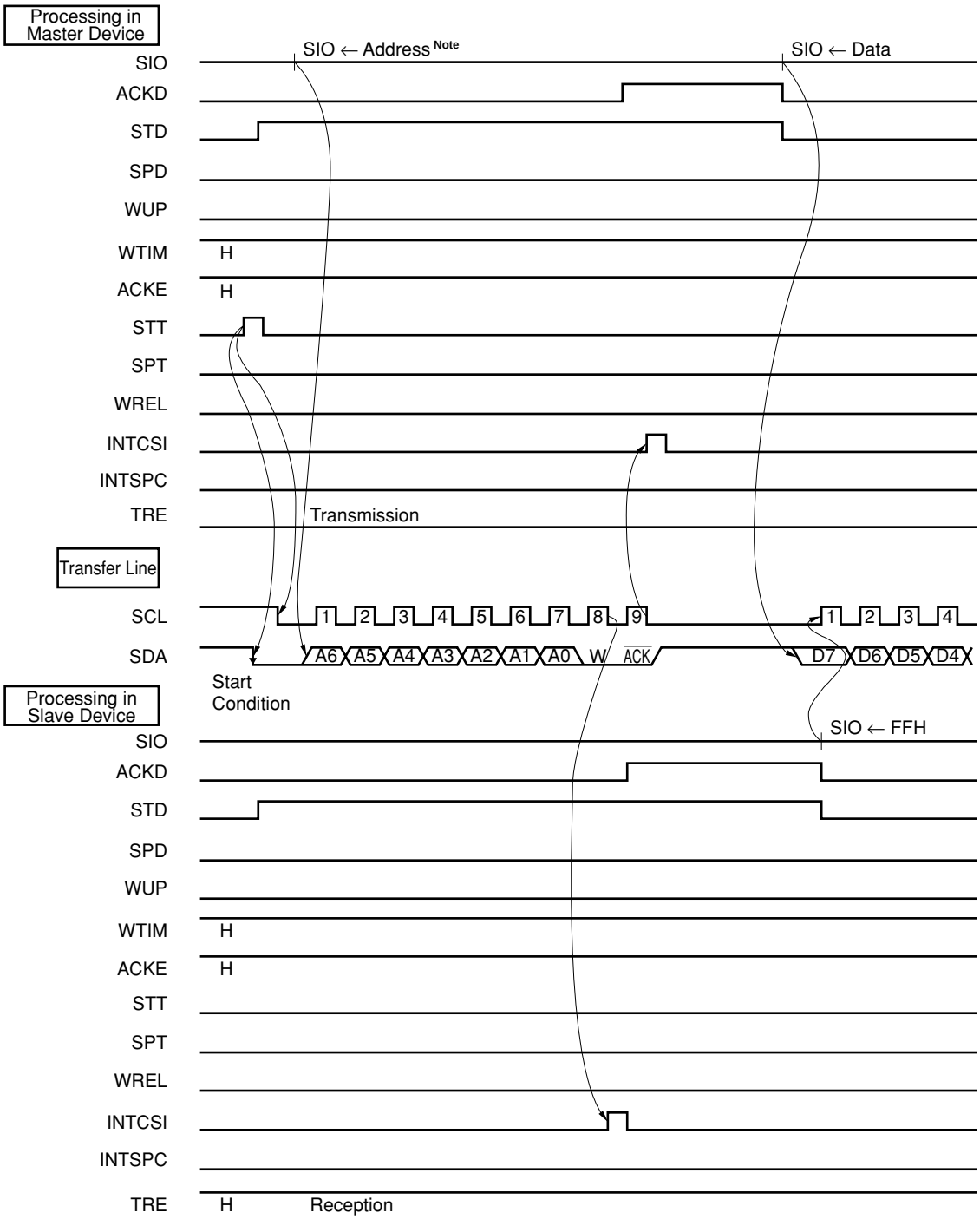
Figures 19-15 and 19-16 indicate the timing chart of the data transmission.

First, the shift operation of the shift register (SIO) is performed synchronizing with the falling edge of serial clock, next, the transmission data are sent to SO0 latch, finally, they are output from SDA pin as MSB first.

On the other hand, the data input to the SDA pin triggered by SCL rising edge are held into the SIO.

Figure 19-15 Example of Communication from Master to Slave (with 9-clock wait selected for both master and slave. Slave: WUP = 0) (1/3)

(1) Start condition = address



Note After the STT is set (to 1), at least one instruction such as NOP etc., should be executed before writing the address to the SIO.

Figure 19-15 Example of Communication from Master to Slave
(with 9-clock wait both selected for master and slave. Slave: WUP = 0) (2/3)

(2) Data

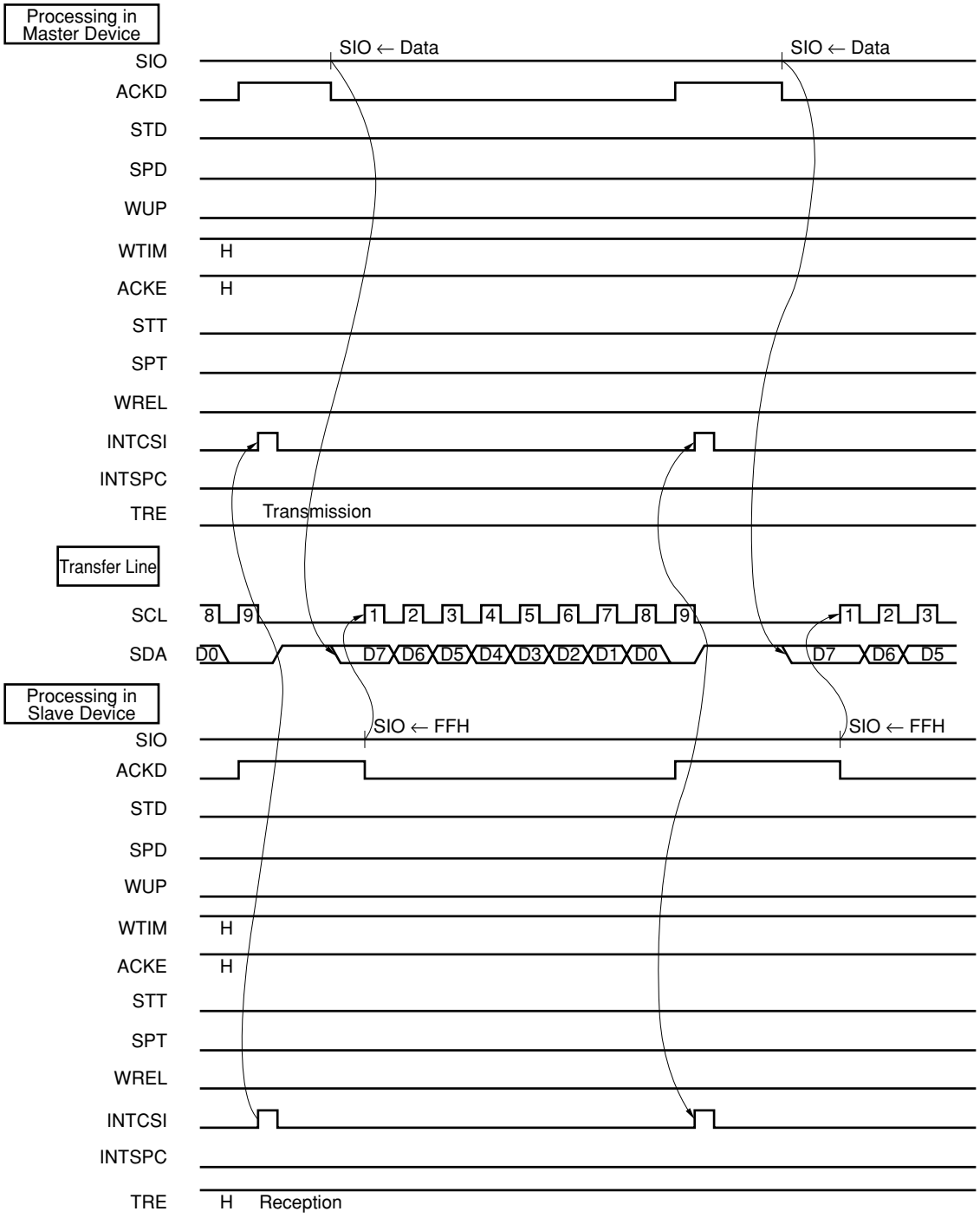


Figure 19-15 Example of Communication from Master to Slave
(with 9-clock wait both selected for master and slave. Slave: WUP = 0) (3/3)

(3) Stop condition

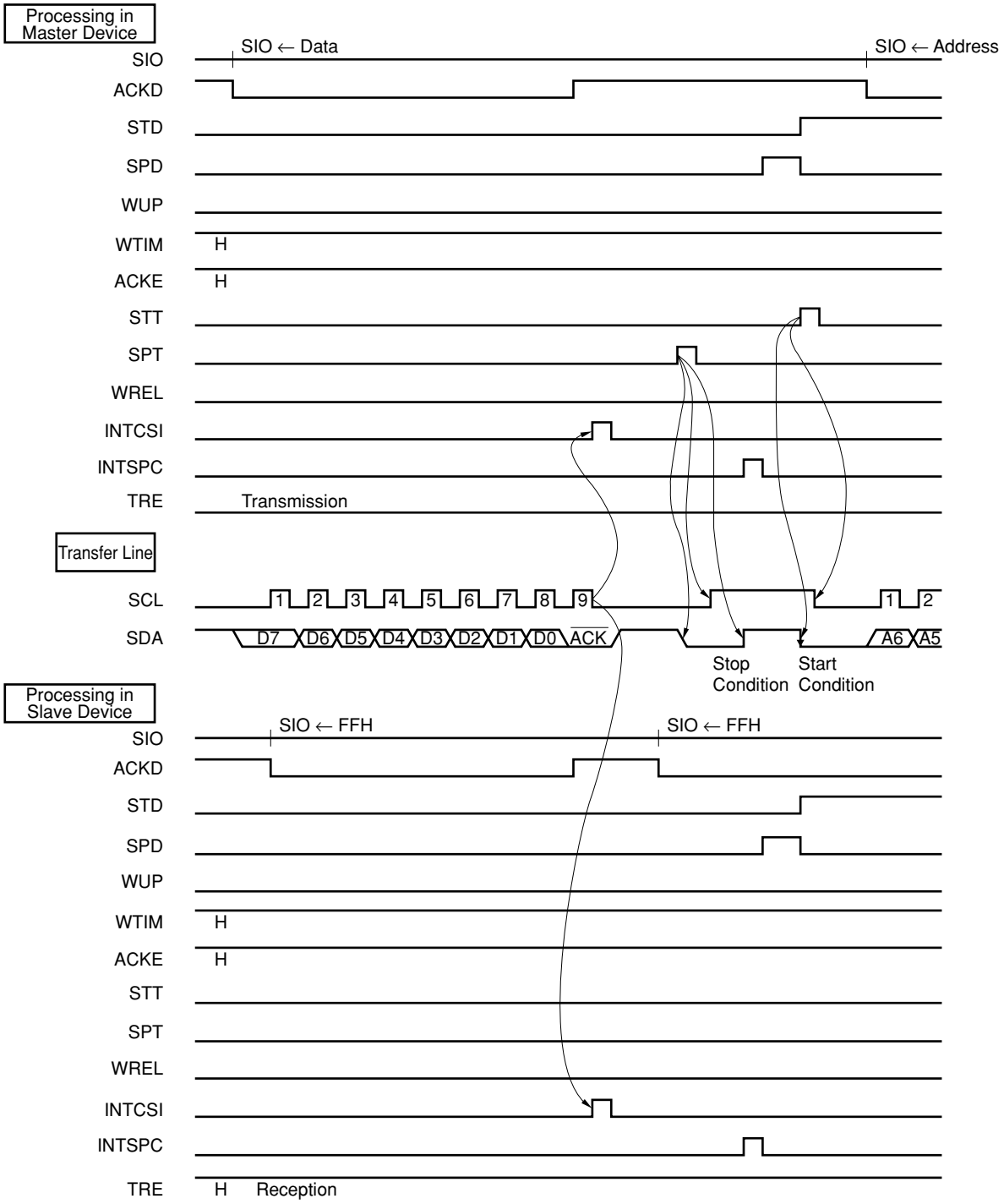
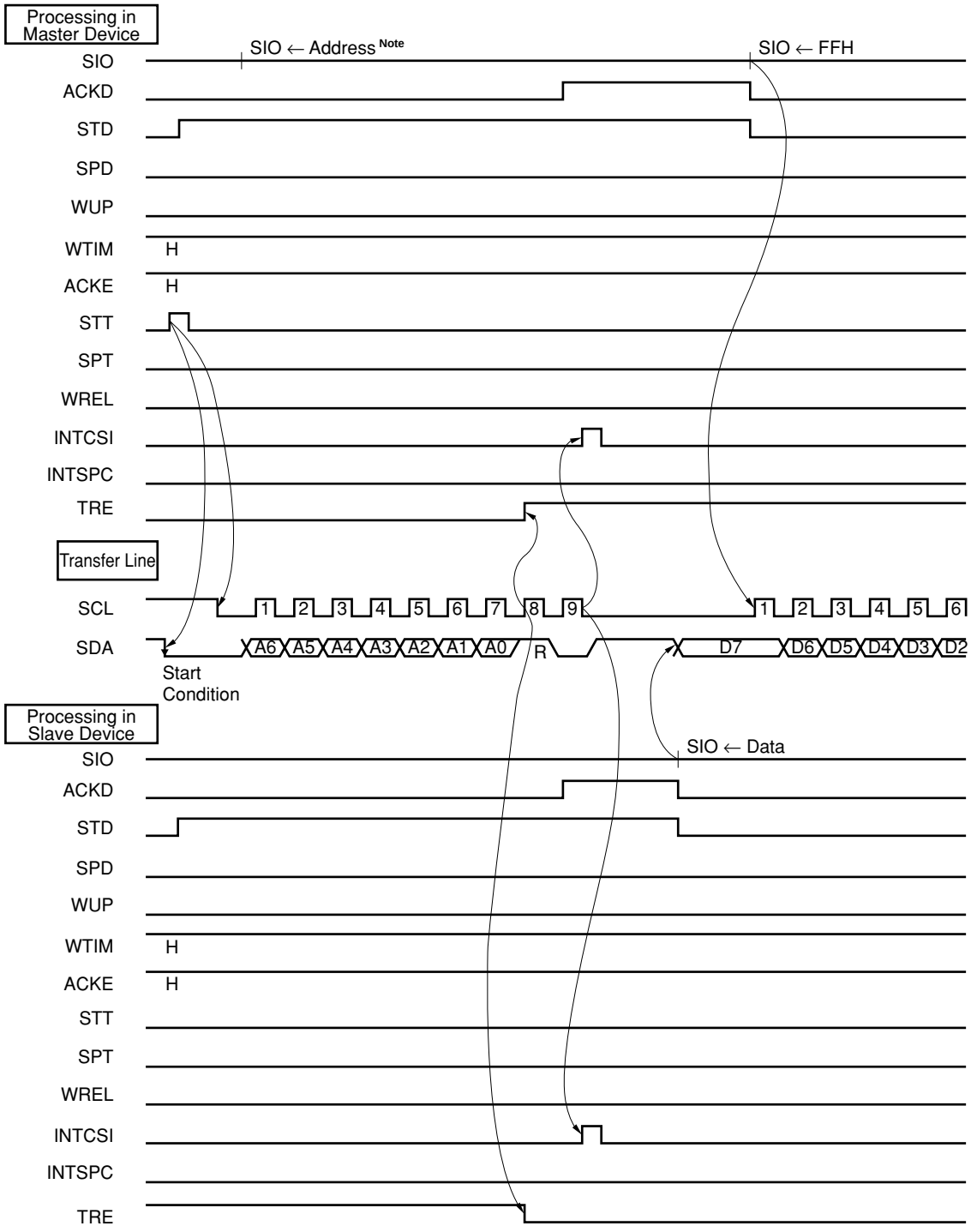


Figure 19-16 Example of Communication from Slave to Master (When selecting the 9th clock wait both master and slave) (1/3)

(1) Start condition = address



Note After the STT is set (to 1), at least one instruction such as NOP, etc., should be executed before writing the address to the SIO.

Figure 19-16 Example of Communication from Slave to Master (When selecting the 9th clock wait both master and slave) (2/3)

(2) Data

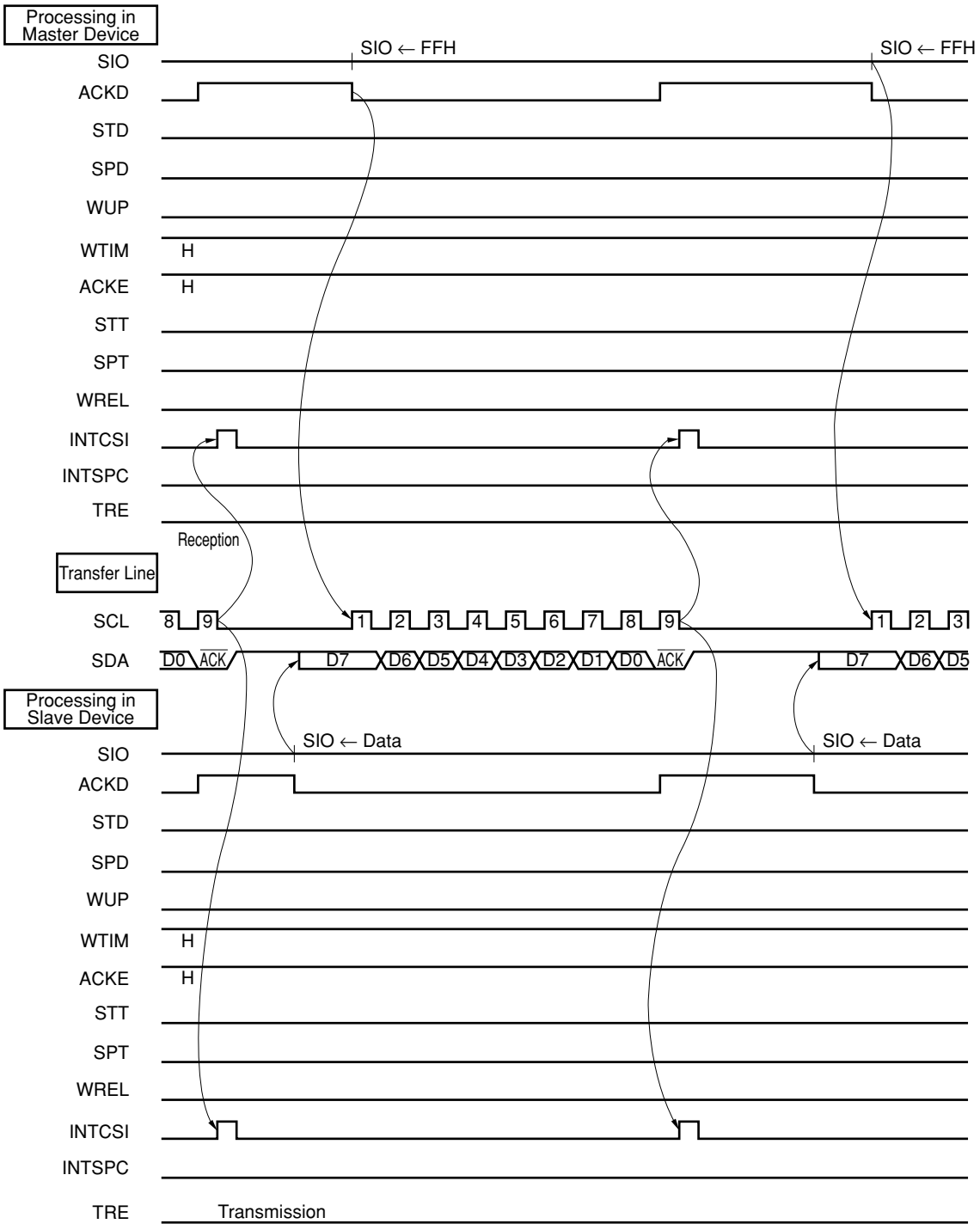
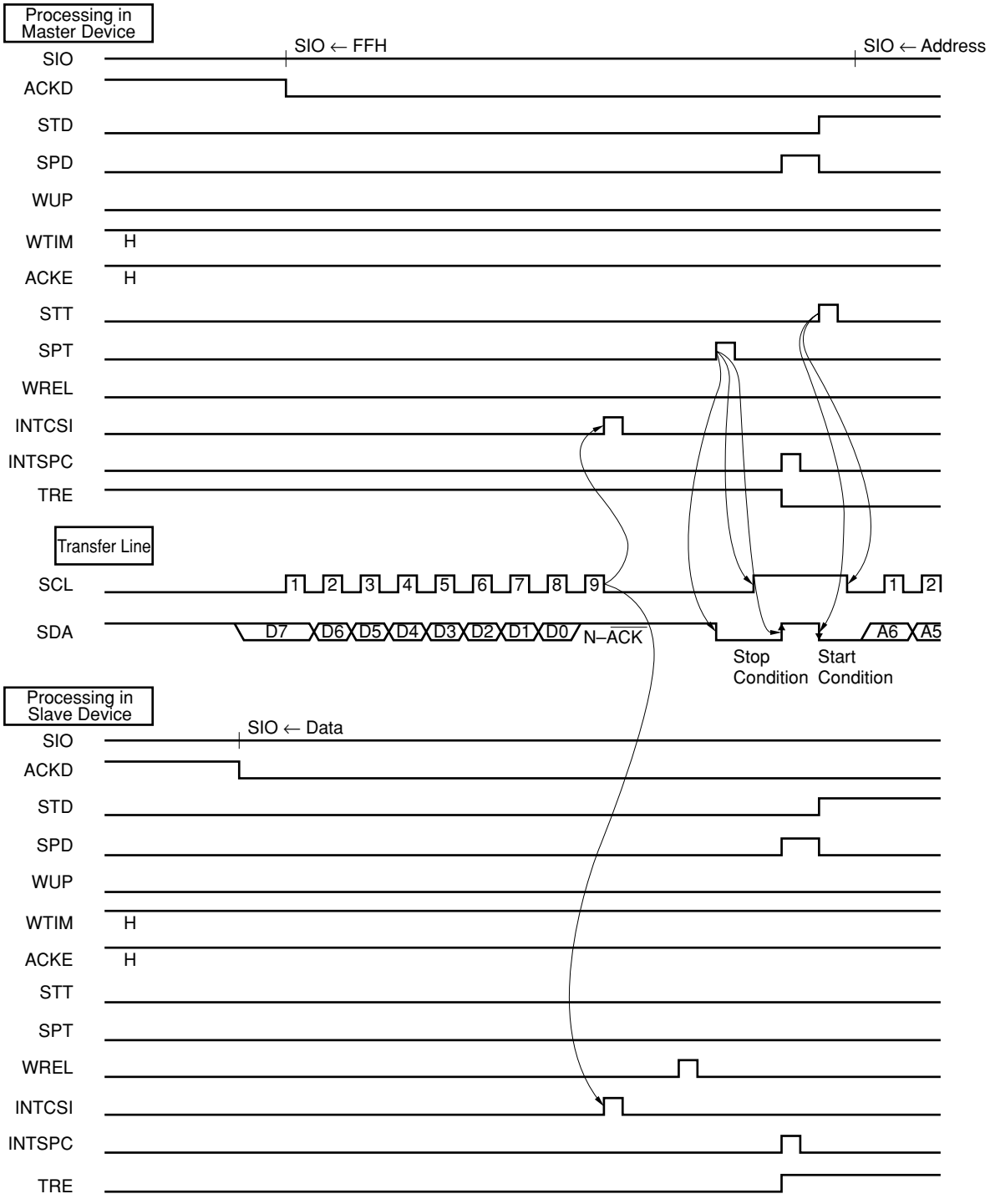


Figure 19-16 Example of Communication from Slave to Master (When selecting the 9th clock wait both master and slave) (3/3)

(3) Stop Condition



19.7 SIGNAL AND FLAGS

Table 19-2 lists the relationship between kinds of signals and flags in I²C bus mode.

Table 19-2 Relationship between Signals and Flags

Signal	Outputting Device	Definition	Conditions for Outputting	Effect to Flag	Meaning of Signal
Start condition	Master	At falling edge of SDA when SCL is in high level	Sets STT.	Sets STD, Clear SPD.	Transmitting the address to next, and indicates start of serial communication.
Stop condition	Master	At falling edge of SDA when SCL is in high level	Sets SPT.	Sets both SPD and SPCIF, Clears both ACKD and STD.	Indicates end of the serial communication.
Acknowledge signal ($\overline{ACK}$)	Master/Slave (receiver)	After the reception has been completed, the low level signal of SDA output when 9th clock of SCL is staying in high level.	ACKE = 1	Sets ACKD.	Indicates 1 byte reception has ended.
Wait ($\overline{WAIT}$)	Master/Slave	Low level signal output to SCL	Depends on the value of WTIM bit		Indicates that serial communication is in disable state.
Serial clock	Master	Synchronous clock for various signals output	Executes to write data to SIO when CTXE = 1 (start instruction of serial transmission) Note 1	Sets CSIIF.Note 2	Synchronous signal of serial communication
Address (A6 to A0)	Master	7-bit data synchronizing with SCL after start condition output			Indicates address value to specify slave on the serial bus
Transfer direction (R/W)	Master	1-bit data synchronizing with SCL after address is output.			Perform either data transmission or reception
Data (D7 to D0)	Master/Slave	8-bit data synchronizing with SCL which is not immediately after entering start condition			Indicates actual data for communicating

Notes 1. When it is wait state, serial transmission is started after the wait state has been released.

2. For further details of timing for generation of a interrupt request, refer to Table 19-1 INTCSI Generation Timing and Wait Control.

CHAPTER 20 CLOCK OUTPUT FUNCTION

The μPD784038 has a clock function that outputs a signal scaled from the system clock.

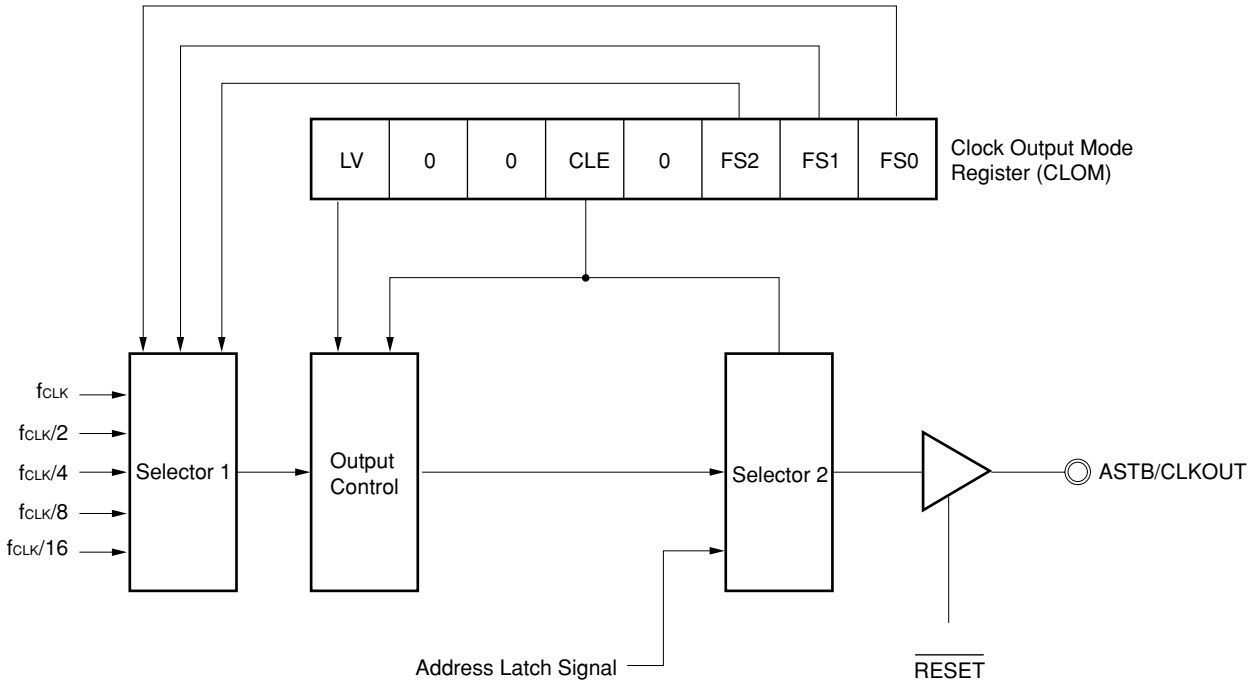
The clock output function can output the system clock directly, or a 1/2, 1/4, 1/8 or 1/16 system clock signal. In addition, it can be used as a 1-bit output port. The output pin has a dual function as the ASTB pin.

Caution This function cannot be used with the μPD784031, and when the external memory extension mode is used.

20.1 CONFIGURATION

The clock output function configuration is shown in Figure 20-1.

Figure 20-1 Clock Output Function Configuration



(1) Clock output mode register (CLOM)

Register that controls the operation of the clock output function.

(2) Selector 1

Selector that selects the frequency of the clock to be output.

(3) Output control

Controls the output signal in accordance with the contents of the clock output mode register (CLOM).

(4) Selector 2

Selects either the ASTB signal or the CLKOUT signal as the signal to be output to the ASTB/CLKOUT pin.

(5) ASTB/CLKOUT pin

Pin that outputs the signal selected by selector 2. While the $\overline{\text{RESET}}$ input is low, the ASTB/CLO pin is in the Hi-Z state, and when the $\overline{\text{RESET}}$ input becomes high it outputs a low-level signal, and then outputs a signal according to the set function.

20.2 CLOCK OUTPUT MODE REGISTER (CLOM)

The CLOM controls the clock output function.

CLOM can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction.

The CLOM format is shown in Figure 20-2.

$\overline{\text{RESET}}$ input clears the CLOM register to 00H.

Figure 20-2 Clock Output Mode Register (CLOM) Format

	⑦	6	5	④	3	2	1	0	Address	After Reset	R/W
CLOM	LV	0	0	CLE	0	FS2	FS1	FS0	0FFC6H	00H	R/W

FS2	FS1	FS0	Frequency Selection
0	0	0	f_{CLK} Note
0	0	1	$f_{CLK}/2$
0	1	0	$f_{CLK}/4$
0	1	1	$f_{CLK}/8$
1	0	0	$f_{CLK}/16$

Note Outputs the system clock
Duty ≠ 50 %

CLE	Clock Output Control
0	Outputs LV bit contents
1	Outputs clock selected by bits FS2 to FS0

LV	Output Level Control
0	Outputs low level
1	Outputs high level

Cautions 1. With the μPD784031, and when the external memory extension mode is used, the clock output mode register (CLOM) should be set to 00H (value after $\overline{\text{RESET}}$ release).

2. The other bits (FS0 to FS2 and LV) must not be changed while the CLE bit is set (to 1).

3. The other bits (FS0 to FS2 and LV) must not be changed at the same time when the CLE bit is changed.

20.3 OPERATION

20.3.1 Clock Output

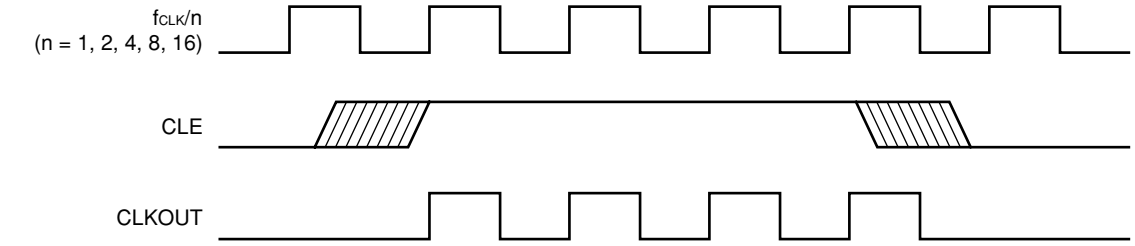
A signal with the clock output frequency selected by bits FS0 to FS2 is selected by selector 1 and output.

The output signal has the same level as the LV bit when the CLE bit is cleared (to 0), and is output from the clock signal immediately after the CLE bit is set (to 1).

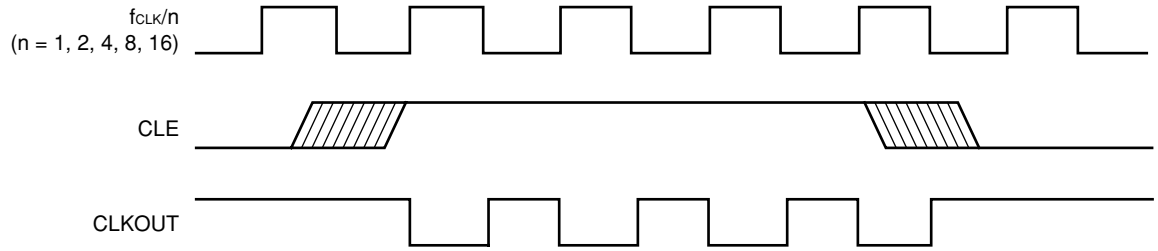
When the CLE bit is cleared (to 0), the contents of the LV bit are output in synchronization with the clock signal, and further output operations are stopped.

Figure 20-3 Clock Output Operation Timing

(a) LV = 0



(b) LV = 1



Setting of bits FS0 to FS2 and the LV bit should only be performed when CLE = 0 (bits FS0 to FS2 and the LV bit should not be changed within the same instruction that changes the CLE bit contents).

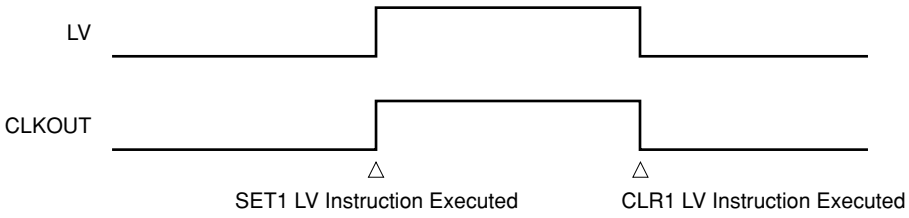
<Operation Example>

```
MOV CLOM, #82H ; CLKOUT pin: high level, clock output: fCLK/4
SET1 CLE       ; Starts clock output
  ⋮
  ⋮
CLR1 CLE       ; Stops clock output, CLKOUT pin: high level
```

20.3.2 One-Bit Output Port

When the CLE bit is cleared (to 0), the contents of the LV bit are output from the CLKOUT pin. The CLKOUT pin changes as soon as the contents of the LV bit change.

Figure 20-4 One-Bit Output Port Operation



20.3.3 Operation in Standby Mode

(1) HALT mode

The state prior to setting of the HALT mode is maintained. That is, if, during clock output, clock output has been performed continuously, and clock output has been disabled, the LV bit contents set before the HALT mode setting are output unchanged.

(2) STOP mode and IDLE mode

Clock output must be disabled before setting the STOP mode or IDLE mode (this must be done by software). The CLKOUT pin level output is the level before the STOP mode or IDLE mode was set (the contents of the LV bit).

20.4 CAUTIONS

(1) This function cannot be used with the μPD784031, and when the external memory extension mode is used.

(2) With the μPD784031, and when the external memory extension mode is used, the clock output mode register (CLOM) should be set to 00H (value after $\overline{\text{RESET}}$ release).

(3) The other bits (FS0 to FS2 and LV) must not be changed while the CLE bit is set (to 1).

(4) The other bits (FS0 to FS2 and LV) must not be changed at the same time when the CLE bit is changed.

CHAPTER 21 EDGE DETECTION FUNCTION

P20 to P26 have an edge detection function that allows a rising edge/falling edge to be set programmably, and the detected edge is sent to internal hardware. The relation between pins P20 to P26 and the use of the detected edge is shown in Table 21-1.

Table 21-1 Pins P20 to P26 and Use of Detected Edge

Pin	Use	Detected Edge Specification Register
P20	NMI, standby circuit control	INTM0
P21	INTP0, timer/counter 1 capture signal timer/counter 1 count clock signal Real-time output port trigger signal	
P22	INTP1, timer/counter 2 CR22 capture signal	
P23	INTP2, CI (timer/counter 2 count clock signal), timer/counter 2 CR21 capture signal	
P24	INTP3, timer/counter 0 capture signal timer/counter 0 count clock signal	INTM1
P25	INTP4, standby circuit control	
P26	INTP5, A/D converter conversion start signal, standby circuit control	

The edge detection function operates at all times except in STOP mode and IDLE mode (although the edge detection function for pins P20, P25 and P26 also operates in STOP mode and IDLE mode).

For the P21/INTP0 pin, the noise elimination time when edge detection is performed can be selected by software.

21.1 EDGE DETECTION FUNCTION CONTROL REGISTERS

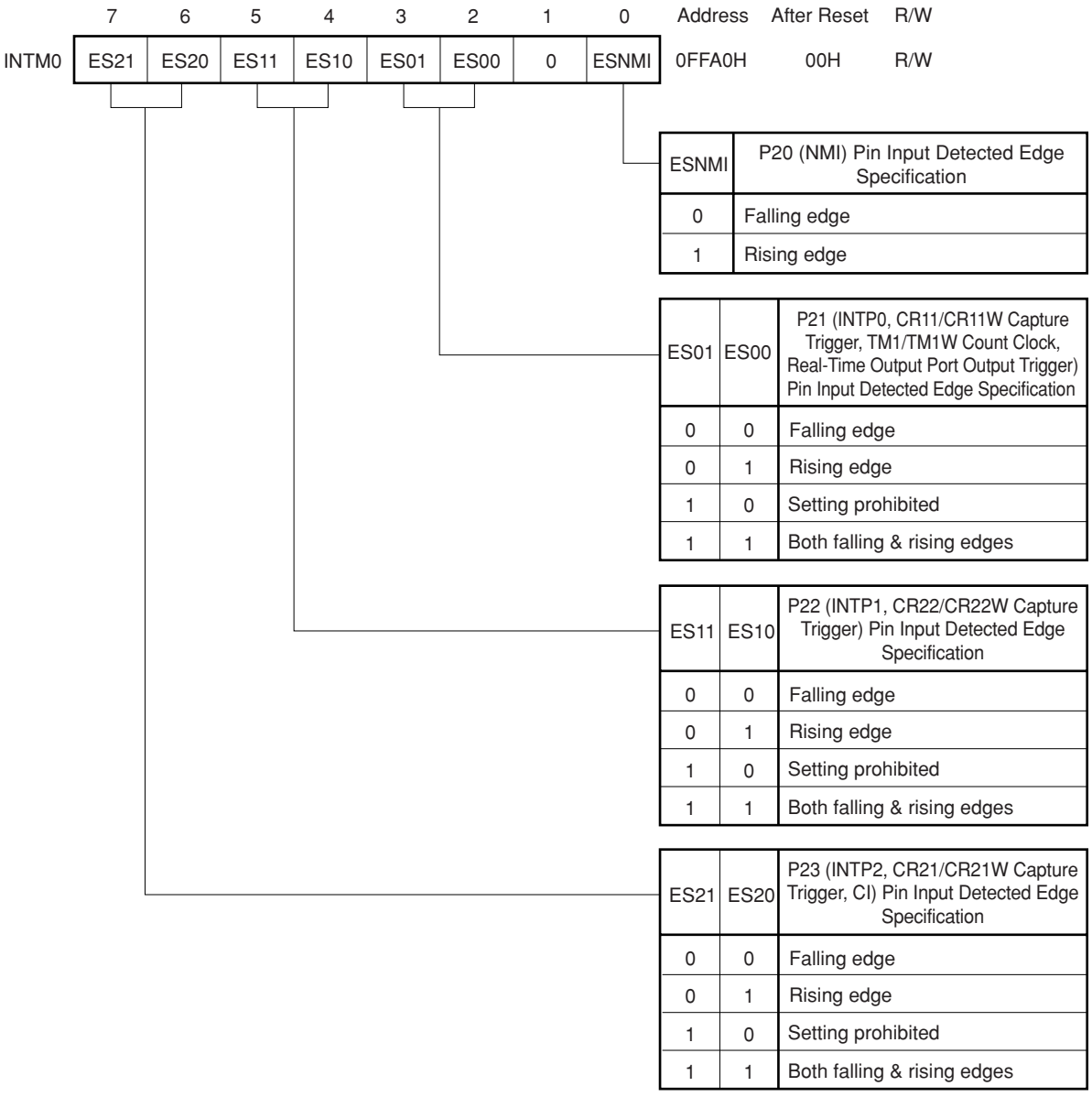
21.1.1 External Interrupt Mode Registers (INTM0, INTM1)

The INTMn (n = 0, 1) specify the valid edge to be detected on pins P20 to P26. The INTM0 specifies the valid edge for pins P20 to P23, and the INTM1 specifies the valid edge for pins P24 to P26.

The INTMn can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. The format of INTM0 and INTM1 are shown in Figures 21-1 and 21-2 respectively.

$\overline{\text{RESET}}$ input clears these registers to 00H.

Figure 21-1 External Interrupt Mode Register 0 (INTM0) Format



	7	6	5	4	3	2	1	0	Address	After Reset	R/W
INTM0	ES21	ES20	ES11	ES10	ES01	ES00	0	ESNMI	0FFA0H	00H	R/W

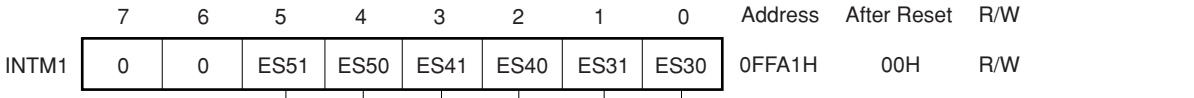
ESNMI	P20 (NMI) Pin Input Detected Edge Specification
0	Falling edge
1	Rising edge

ES01	ES00	P21 (INTP0, CR11/CR11W Capture Trigger, TM1/TM1W Count Clock, Real-Time Output Port Output Trigger) Pin Input Detected Edge Specification
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling & rising edges

ES11	ES10	P22 (INTP1, CR22/CR22W Capture Trigger) Pin Input Detected Edge Specification
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling & rising edges

ES21	ES20	P23 (INTP2, CR21/CR21W Capture Trigger, CI) Pin Input Detected Edge Specification
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling & rising edges

Figure 21-2 External Interrupt Mode Register 1 (INTM1) Format



	7	6	5	4	3	2	1	0	Address	After Reset	R/W
INTM1	0	0	ES51	ES50	ES41	ES40	ES31	ES30	0FFA1H	00H	R/W

ES31	ES30	P24 (INTP3, CR02 Capture Trigger, TM0 Count Clock) Pin Input Detected Edge Specification
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling & rising edges

ES41	ES40	P25 (INTP4) Pin Input Detected Edge Specification
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling & rising edges

ES51	ES50	P26 (INTP5, A/D Conversion Start Signal) Pin Input Detected Edge Specification
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both falling & rising edges

Caution Valid edge detection cannot be performed when the valid edge is changed by a write to the external interrupt mode register (INTMn: n = 0, 1). Also, if an edge is input during a change of the valid edge, that edge may or may not be judged to be a valid edge.

21.1.2 Sampling Clock Selection Register (SCS0)

The SCS0 specifies the sampling clock (f_{SMP}) for digital noise elimination performed on pin P21.

The SCS0 can be read or written to with an 8-bit manipulation instruction. The format of SCS0 is shown in Figure 21-3.

$\overline{\text{RESET}}$ input clears the SCS0 register to 00H.

Figure 21-3 Sampling Clock Selection Register (SCS0) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
SCS0	0	0	0	0	0	0	SCS01	SCS00	0FFA4H	00H	R/W

[f_{XX} = 32 MHz
f_{CLK} = 16 MHz]

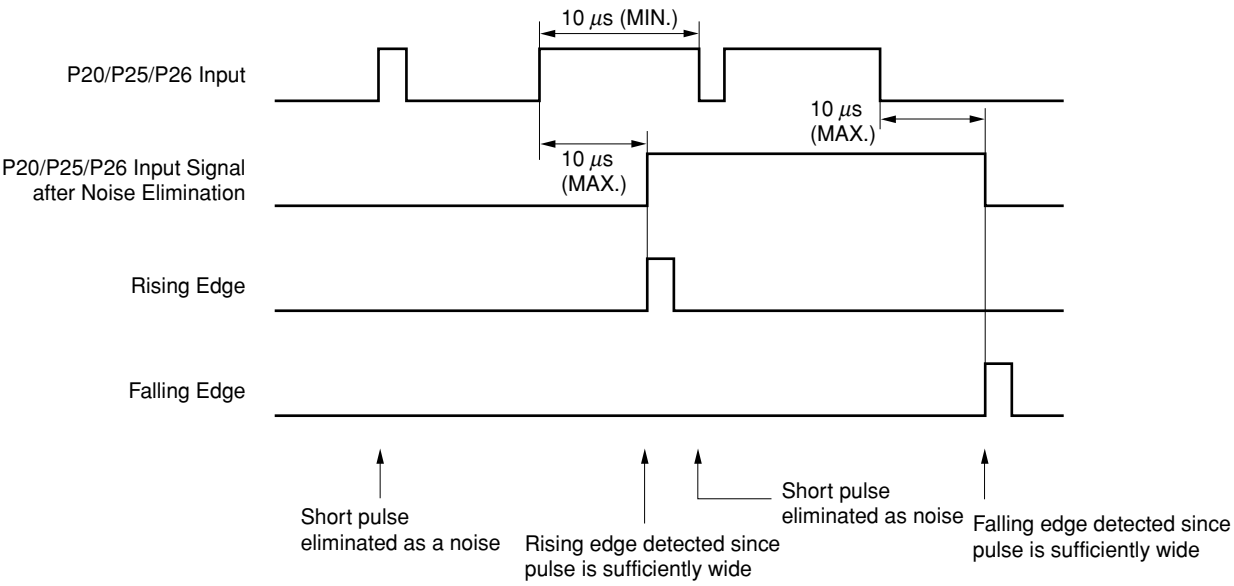
SCS01	SCS00	Sampling Clock (f_{SMP})	Pulse Width Eliminated as Noise	Minimum Pulse Width Recognized as Signal
0	0	f_{CLK}	$3/f_{CLK}$ (188 ns)	$4/f_{CLK}$ (250 ns)
0	1	$f_{XX}/64$	$192/f_{XX}$ (6.0 μs)	$256/f_{XX}$ (8.0 μs)
1	0	$f_{XX}/128$	$384/f_{XX}$ (12.0 μs)	$512/f_{XX}$ (16.0 μs)
1	1	$f_{XX}/256$	$768/f_{XX}$ (24.0 μs)	$1,024/f_{XX}$ (32.0 μs)

21.2 EDGE DETECTION FOR PINS P20, P25 AND P26

On pins P20, P25 and P26, noise elimination is performed by means of analog delay before edge detection. Therefore, an edge cannot be detected unless the pulse width is a given time (10 μs) or longer.

★ The width of the pulse eliminated as noise varies depending on the characteristics and ambient temperature of the device used. It is recommended to input a pulse with a width of 10 μs or more to prevent the pulse from being eliminated as noise.

Figure 21-4 Edge Detection for Pins P20, P25 and P26



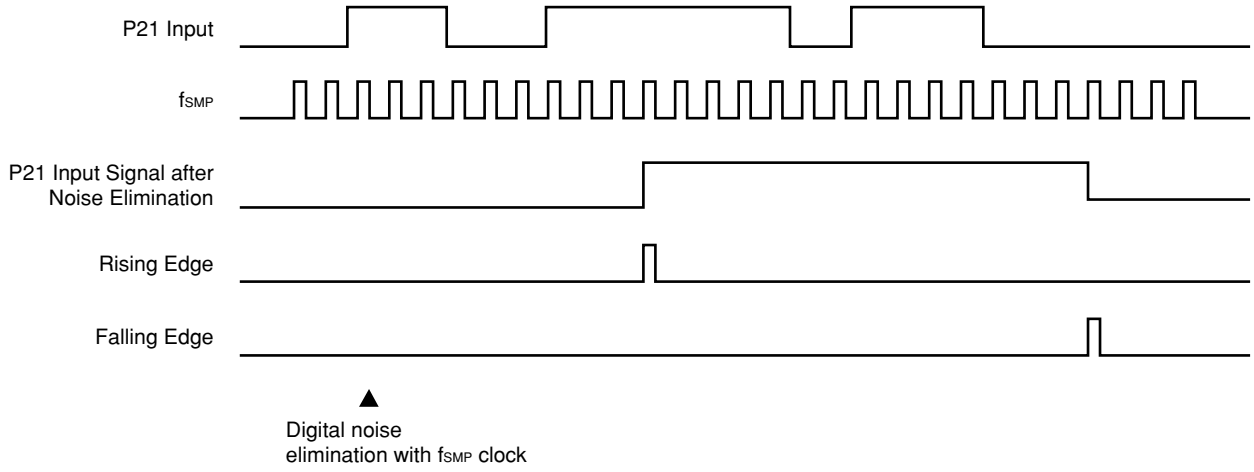
Caution Since analog delay noise elimination is performed on pins P20, P25 and P26, an edge is detected up to 10 μs after it is actually input. Also, unlike pins P21 to P24, the delay before an edge is detected is not a specific value, because of differences in the characteristics of various devices.

21.3 EDGE DETECTION FOR PIN P21

In P21 edge detection, digital noise elimination is performed using the clock (f_{SMP}) specified by the sampling clock selection register (SCS0). In digital noise elimination, input is sampled using the f_{SMP} clock, and if the input level is not the same at least four times in succession (if it is the same only three or fewer times in succession), it is eliminated as noise. Therefore, the level must be maintained for at least 4 f_{SMP} clock cycles in order to be recognized as a valid edge.

Remark When the pulse width of a signal with a comparatively long pulse width and a lot of noise, such as an infrared remote count reception signal, is measured, or when a signal is input in which oscillation occurs when an edge occurs, as with switch input chattering, for instance, it is better to set the sampling clock to low speed with the sampling clock selection register (SCS0). If the sampling clock is high-speed, there will be a reaction to the short-pulse noise components as well, and the program will frequently have to judge whether the input is noise or a signal. However, by slowing down the sampling clock, reaction to short pulse width noise is eliminated and thus the program does not have to make judgments so frequently, and can thus be simplified.

Figure 21-5 P21 Pin Edge Detection



Cautions 1. Since digital noise elimination is performed with the f_{SMP} clock, there is a delay of 3 to 4 f_{SMP} clocks between input of an edge to the pin and the point at which the edge is actually detected.

2. If the input pulse width is 3 to 4 f_{SMP} clocks, it is uncertain whether a valid edge will be detected. Therefore, to ensure reliable operation, the level should be held for at least 4 clocks.

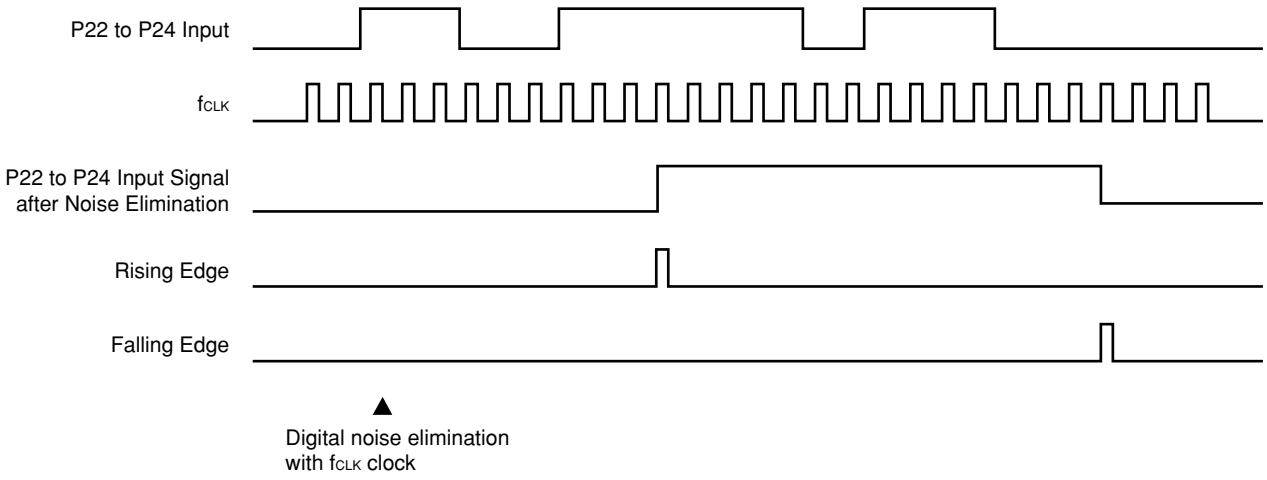
3. If noise input to the pin is synchronized with the f_{SMP} clock in the μPD784038, it may not be recognized as noise. If there is a possibility of such noise being input, noise should be eliminated by adding a filter to the input pin.

21.4 EDGE DETECTION FOR PINS P22 TO P24

Edge detection for pins P22 to P24 is performed after digital noise elimination by means of clock sampling. Unlike the P21 pin, f_{CLK} is used as the sampling clock.

In digital noise elimination, input is sampled using the f_{CLK} clock, and if the input level is not the same at least four times in succession (if it is the same only three or fewer times in succession), it is eliminated as noise. Therefore, the level must be maintained for at least 4 f_{CLK} clock cycles (0.25 μs: f_{CLK} = 16 MHz, f_{CLK} = 1/2 f_{XX}, f_{XX} = 32 MHz) in order to be recognized as a valid edge.

Figure 21-6 Edge Detection for Pins P22 to P24



Cautions 1. Since digital noise elimination is performed with the f_{CLK} clock, there is a delay of 3 to 4 f_{CLK} clocks between input of an edge to the pin and the point at which the edge is actually detected.

2. If the input pulse width is 3 to 4 f_{CLK} clocks, it is uncertain whether a valid edge will be detected. Therefore, to ensure reliable operation, the level should be held for at least 4 clocks.

3. If noise input to a pin is synchronized with the f_{CLK} clock in the μPD784038, it may not be recognized as noise. If there is a possibility of such noise being input, noise should be eliminated by adding a filter to the input pins.

21.5 CAUTIONS

(1) Valid edge detection cannot be performed when the valid edge is changed by a write to the external interrupt mode register (INTMn: n = 0, 1). Also, if an edge is input during a change of the valid edge, that edge may or may not be judged to be a valid edge.

(2) Since analog delay noise elimination is performed on pins P20, P25 and P26, an edge is detected up to 10 μs after it is actually input. Also, unlike pins P21 to P24, the delay before an edge is detected is not a specific value, because of differences in the characteristics of various devices.

(3) Since digital noise elimination is performed on the P21 pin with the f_{SMP} clock, there is a delay of 3 to 4 f_{SMP} clocks between input of an edge to the pin and the point at which the edge is actually detected.

(4) If the input pulse width on the P21 pin is 3 to 4 f_{SMP} clocks, it is uncertain whether a valid edge will be detected. Therefore, to ensure reliable operation, the level should be held for at least 4 clocks.

(5) If noise input of the P21 pin is synchronized with the f_{SMP} clock in the μPD784038, it may not be recognized as noise. If there is a possibility of such noise being input, noise should be eliminated by adding a filter to the input pins.

(6) Since digital noise elimination is performed on pins P22 to P24 with the f_{CLK} clock, there is a delay of 3 to 4 f_{CLK} clocks between input of an edge to the pin and the point at which the edge is actually detected.

(7) If the input pulse width on pins P22 to P24 is 3 to 4 f_{CLK} clocks, it is uncertain whether a valid edge will be detected. Therefore, to ensure reliable operation, the level should be held for at least 4 clocks.

(8) If noise input to pins P22 to P24 is synchronized with the f_{CLK} clock in the μPD784038, it may not be recognized as noise. If there is a possibility of such noise being input, noise should be eliminated by adding a filter to the input pins.

CHAPTER 22 INTERRUPT FUNCTIONS

The μPD784038 is provided with three interrupt request service modes (see **Table 22-1**). These three service modes can be set as required in the program. However interrupt service by macro service can only be selected for interrupt request sources provided with the macro service processing mode shown in Table 22-2. Context switching cannot be selected for non-maskable interrupts or operand error interrupts.

Multiple-interrupt control using 4 priority levels can easily be performed for maskable vectored interrupts.

Table 22-1 Interrupt Request Service Modes

Interrupt Request Service Mode	Servicing Performed	PC & PSW Contents	Service
Vectored interrupts	Software	Saving to & restoration from stack	Executed by branching to service program at address Note specified by vector table
Context switching		Saving to & restoration from fixed area in register bank	Executed by automatic switching to register bank specified by vector table and branching to service program at address Note specified by fixed area in register bank
Macro service	Hardware (firmware)	Retained	Execution of pre-set service such as data transfers between memory and I/O

Note The start addresses of all interrupt service programs must be in the base area. If the body of a service program cannot be located in the base area, a branch instruction to the service program should be written in the base area.

22.1 INTERRUPT REQUEST SOURCES

The μPD784038 has the 25 interrupt request sources shown in Table 22-2, with a vector table allocated to each.

Table 22-2 Interrupt Request Sources (1/2)

Type of Interrupt Request	Default Priority	Interrupt Request Generating Source	Generating Unit	Interrupt Control Register Name	Context Switching	Macro Service	Macro Service Control Word Address	Vector Table Address
Software	None	BRK instruction execution	—	—	Not	Not possible	— possible	3EH
		BRKCS instruction execution	—	—	Possible	Not	—	—
Operand error	None	Invalid operand in MOV STBC, #byte instruction or MOV WDM, #byte instruction, and LOCATION instruction	—	—	Not possible	Not possible	—	3CH
Non-maskable	None	NMI (pin input edge detection)	Edge detection	—	Not possible	Not possible	—	2H
		INTWDT (watchdog timer overflow)	Watchdog timer	—	Not possible	Not possible	—	4H

Table 22-2 Interrupt Request Sources (2/2)

Type of Interrupt Request	Default Priority	Interrupt Request Generating Source	Generating Unit	Interrupt Control Register Name	Context Switching	Macro Service	Macro Service Control Word Address	Vector Table Address
Maskable	0	INTP0 (pin input edge detection)	Edge detection	PIC0	Possible	Possible	0FE06H	6H
	1	INTP1 (pin input edge detection)		PIC1			0FE08H	8H
	2	INTP2 (pin input edge detection)		PIC2			0FE0AH	0AH
	3	INTP3 (pin input edge detection)		PIC3			0FE0CH	0CH
	4	INTC00 (TM0-CR00 match signal generation)	Timer/ counter 0	CIC00			0FE0EH	0EH
	5	INTC01 (TM0-CR01 match signal generation)		CIC01			0FE10H	10H
	6	INTC10 (TM1-CR10 or TM1W-CR10W match signal generation)	Timer/ counter 1	CIC10			0FE12H	12H
	7	INTC11 (TM1-CR11 or TM1W-CR11W match signal generation)		CIC11			0FE14H	14H
	8	INTC20 (TM2-CR20 or TM2W-CR20W match signal generation)	Timer/ counter 2	CIC20			0FE16H	16H
	9	INTC21 (TM2-CR21 or TM2W-CR22W match signal generation)		CIC21			0FE18H	18H
	10	INTC30 (TM3-CR30 or TM3W-CR30W match signal generation)	Timer 3	CIC30			0FE1AH	1AH
	11	INTP4 (pin input edge detection)	Edge	PIC4			0FE1CH	1CH
	12	INTP5 (pin input edge detection)	detection	PIC5			0FE1EH	1EH
	13	INTAD (A/D conversion end)	A/D converter	ADIC			0FE20H	20H
	14	INTSER (asynchronous serial interface receive error)	Asynchro-nous	SERIC		Not possible	0FE22H	22H
	15	INTSR (asynchronous serial interface reception end)	serial interface/	SRIC		Possible	0FE24H	24H
		INTCSI1 (clocked serial interface transfer end)	clocked serial	CSIIC1				
	16	INTST (asynchronous serial interface transmission end)	interface 1	STIC			0FE26H	26H
	17	INTCSI (clocked serial interface transfer end)	Clocked serial interface	CSIIC			0FE28H	28H
	18	INTSER2 (asynchronous serial interface 2 receive error)	Asynchro-nous	SERIC2		Not possible	0FE2AH	2AH
	19	INTSR2 (asynchronous serial interface 2 reception end)	serial interface 2/	SRIC2		Possible	0FE2CH	2CH
		INTCSI2 (clocked serial interface 2 transfer end)	clocked serial	CSIIC2				
	20	INTST2 (asynchronous serial interface 2 transmission end)	interface 2	STIC2			0FE2EH	2EH
	21 **Note**	INTSPC (I^2C bus stop condition interrupt)	Clocked serial interface	SPCIC			0FE30H	30H

Note μPD784038Y Subseries only

Remarks 1. The default priority is a fixed number. This indicates the order of priority when interrupt requests specified as having the same priority are generated simultaneously,

2. The INTSR and INTCSI1 interrupts are generated by the same hardware (they cannot both be used simultaneously). Therefore, although the same hardware is used for the interrupts, two names are provided, for use in each of the two modes. The same applies to INTSR2 and INTCSI2.

22.1.1 Software Interrupts

Interrupts by software consist of the BRK instruction which generates a vectored interrupt and the BRKCS instruction which performs context switching.

Software interrupts are acknowledged even in the interrupt disabled state, and are not subject to priority control.

22.1.2 Operand Error Interrupts

These interrupts are generated if there is an illegal operand in an MOV STBC, #byte instruction or MOV WDMC, #byte instruction, and LOCATION instruction.

Operand error interrupts are acknowledged even in the interrupt disabled state, and are not subject to priority control.

22.1.3 Non-Maskable Interrupts

A non-maskable interrupt is generated by NMI pin input or the watchdog timer.

Non-maskable interrupts are acknowledged unconditionally **Note**, even in the interrupt disabled state. They are not subject to interrupt priority control, and are of higher priority that any other interrupt.

Note Except during execution of the service program for the same non-maskable interrupt, and during execution of the service program for a higher-priority non-maskable interrupt

22.1.4 Maskable Interrupts

A maskable interrupt is one subject to masking control according to the setting of an interrupt mask flag. In addition, acknowledgment enabling/disabling can be specified for all maskable interrupts by means of the IE flag in the program status word (PSW).

In addition to normal vectored interruption, maskable interrupts can be acknowledged by context switching and macro service (though some interrupts cannot use macro service: see **Table 22-2**).

The priority order for maskable interrupt requests when interrupt requests of the same priority are generated simultaneously is predetermined (default priority) as shown in Table 22-2. Also, multiprocessing control can be performed with interrupt priorities divided into 4 levels. However, macro service requests are acknowledged without regard to priority control or the IE flag.

22.2 INTERRUPT SERVICE MODES

There are three μPD784038 interrupt service modes, as follows:

- Vectored interrupt service
- Macro service
- Context switching

22.2.1 Vectored Interrupt Service

When an interrupt is acknowledged, the program counter (PC) and program status word (PSW) are automatically saved to the stack, a branch is made to the address indicated by the data stored in the vector table, and the interrupt service routine is executed.

22.2.2 Macro Service

When an interrupt is acknowledged, CPU execution is temporarily suspended and a data transfer is performed by hardware. Since macro service is performed without the intermediation of the CPU, it is not necessary to save or restore CPU statuses such as the program counter (PC) and program status word (PSW) contents. This is therefore very effective in improving the CPU service time (See **22.8 Macro Service Function**).

22.2.3 Context Switching

When an interrupt is acknowledged, the prescribed register bank is selected by hardware, a branch is made to a preset vector address in the register bank, and at the same time the current program counter (PC) and program status word (PSW) are saved in the register bank (see **22.4.2 BRKCS Instruction Software Interrupt (Software Context Switching) Acknowledgment Operation** and **22.7.2 Context Switching**).

Remark "Context" refers to the CPU registers that can be accessed by a program while that program is being executed. These registers include general registers, the program counter (PC), program status word (PSW), and stack pointer (SP).

22.3 INTERRUPT SERVICE CONTROL REGISTERS

μPD784038 interrupt service is controlled for each interrupt request by various control registers that perform interrupt service specification. The interrupt control registers are listed in Table 22-3.

Table 22-3 Control Registers

Register Name	Symbol	Function
Interrupt control registers	PIC0 PIC1 PIC2 PIC3 CIC00 CIC01 CIC10 CIC11 CIC20 CIC21 CIC30 PIC4 PIC5 ADIC SERIC SRIC CSIIC1 STIC CSIIC SERIC2 SRIC2 CSIIC2 STIC2 SPCIC **Note**	Registers that perform each interrupt request generation recording, mask control, vectored interrupt service or macro service specification, context switching function enabling/disabling, and priority specification.
Interrupt mask registers	MK0 MK1L	Maskable interrupt request mask control Linked to mask control flags in interrupt control registers Word accesses or byte accesses possible
In-service priority register	ISPR	Records priority of interrupt request currently being acknowledged
Interrupt mode control register	IMC	Controls nesting of maskable interrupts for which lowest priority level (level 3) is specified
Watchdog timer mode register	WDM	Specifies priority of interrupts due to NMI pin input and interrupts due to watchdog timer overflow
Program status word	PSW	Specifies enabling/disabling of maskable interrupt acknowledgment

Note μPD784038Y Subseries only

An interrupt control register is allocated to each interrupt source. The flags of each register perform control of the contents corresponding to the relevant bit position in the register. The interrupt control register flag names corresponding to each interrupt request signal are shown in Table 22-4.

Table 22-4 Interrupt Control Register Flags Corresponding to Interrupt Sources

Default Priority	Interrupt Request Signal		Interrupt Control Registers				
			Interrupt Request Flag	Interrupt Mask Flag	Macro Service Enable Flag	Priority Specification Flag	Context Switching Enable Flag
0	INTP0	PIC0	PIF0	PMK0	PISM0	PPR00	PCSE0 PPR01
1	INTP1	PIC1	PIF1	PMK1	PISM1	PPR10	PCSE1 PPR11
2	INTP2	PIC2	PIF2	PMK2	PISM2	PPR20	PCSE2 PPR21
3	INTP3	PIC3	PIF3	PMK3	PISM3	PPR30	PCSE3 PPR31
4	INTC00	CIC00	CIF00	CMK00	CISM00	CPR000	CCSE00 CPR001
5	INTC01	CIC01	CIF01	CMK01	CISM01	CPR010	CCSE01 CPR011
6	INTC10	CIC10	CIF10	CMK10	CISM10	CPR100	CCSE10 CPR101
7	INTC11	CIC11	CIF11	CMK11	CISM11	CPR110	CCSE11 CPR111
8	INTC20	CIC20	CIF20	CMK20	CISM20	CPR200	CCSE20 CPR201
9	INTC21	CIC21	CIF21	CMK21	CISM21	CPR210	CCSE21 CPR211
10	INTC30	CIC30	CIF30	CMK30	CISM30	CPR300	CCSE30 CPR301
11	INTP4	PIC4	PIF4	PMK4	PISM4	PPR40	PCSE4 PPR41
12	INTP5	PIC5	PIF5	PMK5	PISM5	PPR50	PCSE5 PPR51
13	INTAD	ADIC	ADIF	ADMK	ADISM	ADPR0	ADCSE ADPR1
14	INTSER	SERIC	SERIF	SERMK	—	SERPR0	SERCSE SERPR1
15	INTSR	SRIC	SRIF	SRMK	SRISM	SRPR0	SRCSE SRPR1
	INTCSI1	CSIIC1	CSIIF1	CSIMK1	CSIISM1	CSIPR10	CSICSE1 CSIPR11
16	INTST	STIC	STIF	STMK	STISM	STPR0	STCSE STPR1
17	INTCSI	CSIIC	CSIIF	CSIMK	CSIISM	CSIPR0	CSICSE CSIPR1
18	INTSER2	SERIC2	SERIF2	SERMK2	—	SERPR20	SERCSE2 SERPR21
19	INTSR2	SRIC2	SRIF2	SRMK2	SRISM2	SRPR20	SRCSE2 SRPR21
	INTCSI2	CSIIC2	CSIIF2	CSIMK2	CSIISM2	CSIPR20	CSICSE2 CSIPR21
20	INTST2	STIC2	STIF2	STMK2	STISM2	STPR20	STCSE2 SERPR21
21 **Note**	INTSPC	SPCIC	SPCIF	SPCMK	SPCISM	SPCPR0	SPCCSE SPCPR1

Note μPD784038Y Subseries only

22.3.1 Interrupt Control Registers

An interrupt control register is allocated to each interrupt source, and performs priority control, mask control, etc. for the corresponding interrupt request. The interrupt control register format is shown in Figure 22-1.

(1) Priority specification flags (××PR1/××PR0)

The priority specification flags specify the priority on an individual interrupt source basis for the 21 (22 types for the μPD784038Y Subseries) maskable interrupts.

Up to 4 priority levels can be specified, and a number of interrupt sources can be specified at the same level. Among maskable interrupt sources, level 0 is the highest priority.

If multiple interrupt requests are generated simultaneously among interrupt source of the same priority level, they are acknowledged in default priority order.

These flags can be manipulated bit-wise by software.

$\overline{\text{RESET}}$ input sets all bits to “1”.

(2) Context switching enable flag (××CSE)

The context switching enable flag specifies that a maskable interrupt request is to be serviced by context switching. In context switching, the register bank specified beforehand is selected by hardware, a branch is made to a vector address stored beforehand in the register bank, and at the same time the current contents of the program counter (PC) and program status word (PSW) are saved in the register bank.

Context switching is suitable for real-time processing, since execution of interrupt servicing can be started faster than with normal vectored interrupt servicing.

This flag can be manipulated bit-wise by software.

$\overline{\text{RESET}}$ input sets all bits to “0”.

(3) Macro service enable flag (××ISM)

The macro service enable flag specifies whether an interrupt request corresponding to that flag is to be handled by vectored interruption or context switching, or by macro service.

When macro service processing is selected, at the end of the macro service (when the macro service counter reaches 0) the macro service enable flag is automatically cleared (to 0) by hardware (vectored interrupt service/context switching service).

This flag can be manipulated bit-wise by software.

$\overline{\text{RESET}}$ input sets all bits to “0”.

(4) Interrupt mask flag (××MK)

An interrupt mask flag specifies enabling/disabling of vectored interrupt servicing and macro service processing for the interrupt request corresponding to that flag.

The interrupt mask contents are not changed by the start of interrupt service, etc., and are the same as the interrupt mask register contents (see **22.3.2 Interrupt Mask Registers (MK0/MK1L)**).

Macro service processing requests are also subject to mask control, and macro service requests can also be masked with this flag.

This flag can be manipulated by software.

$\overline{\text{RESET}}$ input sets all bits to “1”.

(5) Interrupt request flag (××IF)

An interrupt request flag is set (to 1) by generation of the interrupt request that corresponds to that flag. When the interrupt is acknowledged, the flag is automatically cleared (to 0) by hardware.

This flag can be manipulated by software.

$\overline{\text{RESET}}$ input sets all bits to “0”.

Figure 22-1 Interrupt Control Registers (××ICn) (1/3)

	⑦	⑥	⑤	④	3	2	1	0	Address	After Reset	R/W
PIC0	PIF0	PMK0	PISM0	PCSE0	0	0	PPR01	PPR00	0FFE0H	43H	R/W
PIC1	PIF1	PMK1	PISM1	PCSE1	0	0	PPR11	PPR10	0FFE1H	43H	R/W
PIC2	PIF2	PMK2	PISM2	PCSE2	0	0	PPR21	PPR20	0FFE2H	43H	R/W
PIC3	PIF3	PMK3	PISM3	PCSE3	0	0	PPR31	PPR30	0FFE3H	43H	R/W
CIC00	CIF00	CMK00	CISM00	CCSE00	0	0	CPR001	CPR000	0FFE4H	43H	R/W
CIC01	CIF01	CMK01	CISM01	CCSE01	0	0	CPR011	CPR010	0FFE5H	43H	R/W
CIC10	CIF10	CMK10	CISM10	CCSE10	0	0	CPR101	CPR100	0FFE6H	43H	R/W
CIC11	CIF11	CMK11	CISM11	CCSE11	0	0	CPR111	CPR110	0FFE7H	43H	R/W

××PRn1 (Bit 1)	××PRn0 (Bit 0)	Interrupt Request Priority Specification
0	0	Priority 0 (highest priority)
0	1	Priority 1
1	0	Priority 2
1	1	Priority 3

××CSEn (Bit 4)	Context Switching Servicing Specification
0	Serviced by vectored interruption
1	Serviced by context switching

××ISMn (Bit 5)	Interrupt Service Mode Specification
0	Vectored interruption servicing/ context switching servicing
1	Macro servicing

××MKn (Bit 6)	Interrupt Service Enabling/Disabling
0	Interrupt service enabled
1	Interrupt service disabled

××IFn (Bit 7)	Interrupt Request Generation Presence/Absence
0	No interrupt request (interrupt signal not being generated)
1	Interrupt request state (interrupt signal being generated)

Figure 22-1 Interrupt Control Registers (××ICn) (2/3)

	⑦	⑥	⑤	④	3	2	1	0	Address	After Reset	R/W
CIC20	CIF20	CMK20	CISM20	CCSE20	0	0	CPR201	CPR200	0FFE8H	43H	R/W
CIC21	CIF21	CMK21	CISM21	CCSE21	0	0	CPR211	CPR210	0FFE9H	43H	R/W
CIC30	CIF30	CMK30	CISM30	CCSE30	0	0	CPR301	CPR300	0FFEAH	43H	R/W
PIC4	PIF4	PMK4	PISM4	PCSE4	0	0	PPR41	PPR40	0FFEBH	43H	R/W
PIC5	PIF5	PMK5	PISM5	PCSE5	0	0	PPR51	PPR50	0FFECH	43H	R/W
ADIC	ADIF	ADMK	ADISM	ADCSE	0	0	ADPR1	ADPR0	0FFEDH	43H	R/W
SERIC	SERIF	SERMK	0	SERCSE	0	0	SERPR1	SERPR0	0FFEEH	43H	R/W
SRIC	SRIF	SRMK	SRISM	SRCSE	0	0	SRPR1	SRPR0	0FFEFH	43H	R/W

××PRn1 (Bit 1)	××PRn0 (Bit 0)	Interrupt Request Priority Specification
0	0	Priority 0 (highest priority)
0	1	Priority 1
1	0	Priority 2
1	1	Priority 3

××CSEn (Bit 4)	Context Switching Servicing Specification
0	Serviced by vectored interruption
1	Serviced by context switching

××ISMn (Bit 5)	Interrupt Service Mode Specification
0	Vectored interruption servicing/ context switching servicing
1	Macro servicing

××MKn (Bit 6)	Interrupt Service Enabling/Disabling
0	Interrupt service enabled
1	Interrupt service disabled

××IFn (Bit 7)	Interrupt Request Generation Presence/Absence
0	No interrupt request (interrupt signal not being generated)
1	Interrupt request state (interrupt signal being generated)

Figure 22-1 Interrupt Control Registers (××ICn) (3/3)

	⑦	⑥	⑤	④	3	2	1	0	Address	After Reset	R/W
CSIIC1	CSIIF1	CSIMK1	CSIISM1	CSICSE1	0	0	CSIPR11	CSIPR10	0FFEFH	43H	R/W
STIC	STIF	STMK	STISM	STCSE	0	0	STPR1	STPR0	0FFF0H	43H	R/W
CSIIC	CSIIF	CSIMK	CSIISM	CSICSE	0	0	CSIPR1	CSIPR0	0FFF1H	43H	R/W
SERIC2	SERIF2	SERMK2	0	SERCSE2	0	0	SERPR21	SERPR20	0FFF2H	43H	R/W
SRIC2	SRIF2	SRMK2	SRISM2	SRCSE2	0	0	SRPR21	SRPR20	0FFF3H	43H	R/W
CSIIC2	CSIIF2	CSIMK2	CSIISM2	CSICSE2	0	0	CSIPR21	CSIPR20	0FFF3H	43H	R/W
STIC2	STIF2	STMK2	STISM2	STCSE2	0	0	STPR21	STPR20	0FFF4H	43H	R/W
SPCIC Note	SPCIF	SPCMK	SPCISM	SPCCSE	0	0	SPCPR1	SPCPR0	0FFF5H	43H	R/W

××PRn1 (Bit 1)	××PRn0 (Bit 0)	Interrupt Request Priority Specification
0	0	Priority 0 (highest priority)
0	1	Priority 1
1	0	Priority 2
1	1	Priority 3

××CSEn (Bit 4)	Context Switching Servicing Specification
0	Serviced by vectored interruption
1	Serviced by context switching

××ISMn (Bit 5)	Interrupt Service Mode Specification
0	Vectored interruption servicing/ context switching servicing
1	Macro servicing

××MKn (Bit 6)	Interrupt Service Enabling/Disabling
0	Interrupt service enabled
1	Interrupt service disabled

××IFn (Bit 7)	Interrupt Request Generation Presence/Absence
0	No interrupt request (interrupt signal not being generated)
1	Interrupt request state (interrupt signal being generated)

Note μPD784038Y Subseries only

22.3.2 Interrupt Mask Registers (MK0/MK1L)

The MK0 and MK1L are composed of interrupt mask flags. MK0 is a 16-bit register which can be manipulated as 8-bit units, MK0L and MK0H, as well as being manipulated as a 16-bit unit.

MK1L is an 8-bit register that can be manipulated as an 8-bit unit. In addition, each bit of the MK0 and MK1L can be manipulated individually with a bit manipulation instruction. Each interrupt mask flag controls enabling/disabling of the corresponding interrupt request.

When an interrupt mask flag is set (to 1), acknowledgment of the corresponding interrupt request is disabled.

When an interrupt mask flag is cleared (to 0), the corresponding interrupt request can be acknowledged as a vectored interrupt or macro service request.

Each interrupt mask flag in the MK0 and MK1L is the same flag as the interrupt mask flag in the interrupt control register. The MK0 and MK1L are provided for en bloc control of interrupt masking.

After $\overline{\text{RESET}}$ input, the MK0 is set to FFFFH, the MK1L is set to FFH, and all maskable interrupts are disabled.

Figure 22-2 Interrupt Mask Register (MK0, MK1L) Format

(1) Byte Accesses

	⑦	⑥	⑤	④	③	②	①	⓪	Address	After Reset	R/W
MK0L	CMK11	CMK10	CMK01	CMK00	PMK3	PMK2	PMK1	PMK0	0FFACH	FFH	R/W
MK0H	CSIMK1 SRMK	SERMK	ADMK	PMK5	PMK4	CMK30	CMK21	CMK20	0FFADH	FFH	R/W
MK1L	1	1	1 Note	STMK2	CSIMK2 SRMK2	SERMK2	CSIMK	STMK	0FFAEH	FFH	R/W

MK	Interrupt Request Enabling/Disabling Specification
0	Interrupt service enabled
1	Interrupt service disabled

Note SPCMK when the μPD784038Y Subseries is used

(2) Word Accesses

	⑮	⑭	⑬	⑫	⑪	⑩	⑨	⑧	Address	After Reset	R/W
MK0	CSIMK1 SRMK	SERMK	ADMK	PMK5	PMK4	CMK30	CMK21	CMK20	0FFACH	FFFFH	R/W

	⑦	⑥	⑤	④	③	②	①	⓪
	CMK11	CMK10	CMK01	CMK00	PMK3	PMK2	PMK1	PMK0

MK	Interrupt Request Enabling/Disabling Specification
0	Interrupt service enabled
1	Interrupt service disabled

22.3.3 In-Service Priority Register (ISPR)

The ISPR shows the priority level of the maskable interrupt currently being serviced and the non-maskable interrupt being serviced. When a maskable interrupt request is acknowledged, the bit corresponding to the priority of that interrupt request is set (to 1), and remains set until the service program ends. When a non-maskable interrupt is acknowledged, the bit corresponding to the priority of that non-maskable interrupt is set (to 1), and remains set until the service program ends.

When an RETI instruction or RETCS instruction is executed, the bit, among those set (to 1) in the ISPR, that corresponds to the highest-priority interrupt request is automatically cleared (to 0) by hardware.

The contents of the ISPR are not changed by execution of an RETB or RETCSB instruction.

$\overline{\text{RESET}}$ input clears the ISPR register to 00H.

Figure 22-3 In-Service Priority Register (ISPR) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
ISPR	NMIS	WDTS	0	0	ISPR3	ISPR2	ISPR1	ISPR0	0FFA8H	00H	R

(n = 0 to 3)

ISPRn	Priority Level
0	Priority n interrupt not being acknowledged
1	Priority n interrupt being acknowledged

WDTS	Watchdog Timer Interrupt Service State
0	Watchdog timer interrupt not being acknowledged
1	Watchdog timer interrupt being acknowledged

NMIS	NMI Service State
0	NMI interrupt not being acknowledged
1	NMI interrupt being acknowledged

Caution In-service priority register (ISPR) is a read-only register. There is a risk of malfunction if a write is performed on this register.

22.3.4 Interrupt Mode Control Register (IMC)

The IMC contains the PRSL flag. The PRSL flag specifies enabling/disabling of nesting of maskable interrupts for which the lowest priority level (level 3) is specified.

When the IMC is manipulated, the interrupt disabled state (DI state) should be set first to prevent malfunction.

The IMC can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction.

$\overline{\text{RESET}}$ input sets the IMC register to 80H.

Figure 22-4 Interrupt Mode Control Register (IMC) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
IMC	PRSL	0	0	0	0	0	0	0	0FFAAH	80H	R/W

PRSL	Control of Nesting Operations for Maskable Interrupts (Lowest Level)
0	Nesting between interrupts set as level 3 (lowest level) enabled
1	Nesting between interrupts set as level 3 (lowest level) disabled

22.3.5 Watchdog Timer Mode Register (WDM)

The PRC bit of the WDM specifies the priority of NMI pin input non-maskable interrupts and watchdog timer overflow non-maskable interrupts.

The WDM can be written to only by a dedicated instruction. This dedicated instruction, MOV WDM, #byte, has a special code configuration (4 bytes), and a write is not performed unless the 3rd and 4th bytes of the operation code are mutual 1's complements.

If the 3rd and 4th bytes of the operation code are not 1's complements, a write is not performed and an operand error interrupt is generated. In this case, the return address saved in the stack area is the address of the instruction that was the source of the error, and thus the address that was the source of the error can be identified from the return address saved in the stack area.

If recovery from an operand error is simply performed by means of an RETB instruction, an endless loop will result.

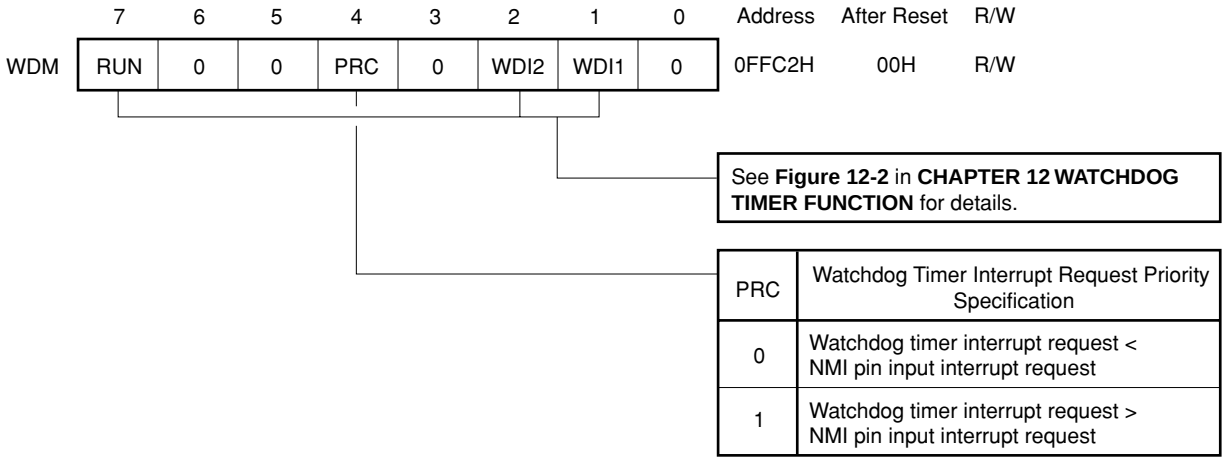
As an operand error interrupt is only generated in the event of an inadvertent program loop (with the NEC Electronics assembler, RA78K4, only the correct dedicated instruction is generated when MOV WDM, #byte is written), system initialization should be performed by the program.

Other write instructions ("MOV WDM, A", "AND WDM, #byte", "SET1 WDM.7", etc.) are ignored and do not perform any operation. That is, a write is not performed to the WDM, and an interrupt such as an operand error interrupt is not generated.

The WDM can be read at any time by a data transfer instruction.

$\overline{\text{RESET}}$ input clears the WDM register to 00H.

Figure 22-5 Watchdog Timer Mode Register (WDM) Format



Caution The watchdog timer mode register (WDM) can only be written to with a dedicated instruction (MOV WDM, #byte).

22.3.6 Program Status Word (PSW)

The PSW is a register that holds the current status regarding instruction execution results and interrupt requests. The IE flag that sets enabling/disabling of maskable interrupts is mapped in the low-order 8 bits of the PSW (PSWL).

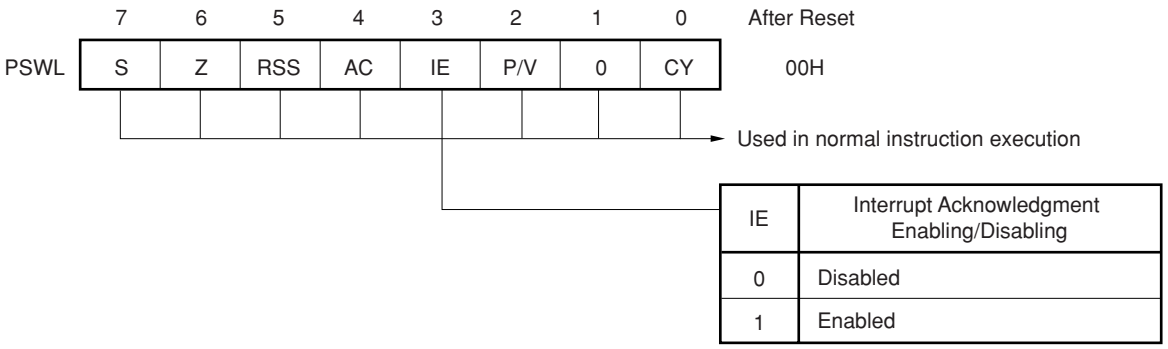
PSWL can be read or written to with an 8-bit manipulation instruction, and can also be manipulated with a bit manipulation instruction or dedicated instruction (EI/DI).

When a vectored interrupt is acknowledged or a BRK instruction is executed, PSWL is saved to the stack and the IE flag is cleared (to 0). PSWL is also saved to the stack by the PUSH PSW instruction, and is restored from the stack by the RETI, RETB and POP PSW instructions.

When context switching or a BRKCS instruction is executed, PSWL is saved to a fixed area in the register bank, and the IE flag is cleared (to 0). PSWL is restored from the fixed area in the register bank by an RETCSI or RETCSB instruction.

$\overline{\text{RESET}}$ input clears PSWL to 00H.

Figure 22-6 Program Status Word (PSWL) Format



IE	Interrupt Acknowledgment Enabling/Disabling
0	Disabled
1	Enabled

22.4 SOFTWARE INTERRUPT ACKNOWLEDGMENT OPERATIONS

A software interrupt is acknowledged in response to execution of a BRK or BRKCS instruction. Software interrupts cannot be disabled.

22.4.1 BRK Instruction Software Interrupt Acknowledgment Operation

When a BRK instruction is executed, the program status word (PSW), program counter (PC) are saved in that order to the stack, the IE flag is cleared (to 0), the vector table (003EH/003FH) contents are loaded into the low-order 16 bits of the PC, and 0000B into the high-order 4 bits, and a branch is performed (the start of the service program must be in the base area).

The RETB instruction must be used to return from a BRK instruction software interrupt.

Caution The RETI instruction must not be used to return from a BRK instruction software interrupt.

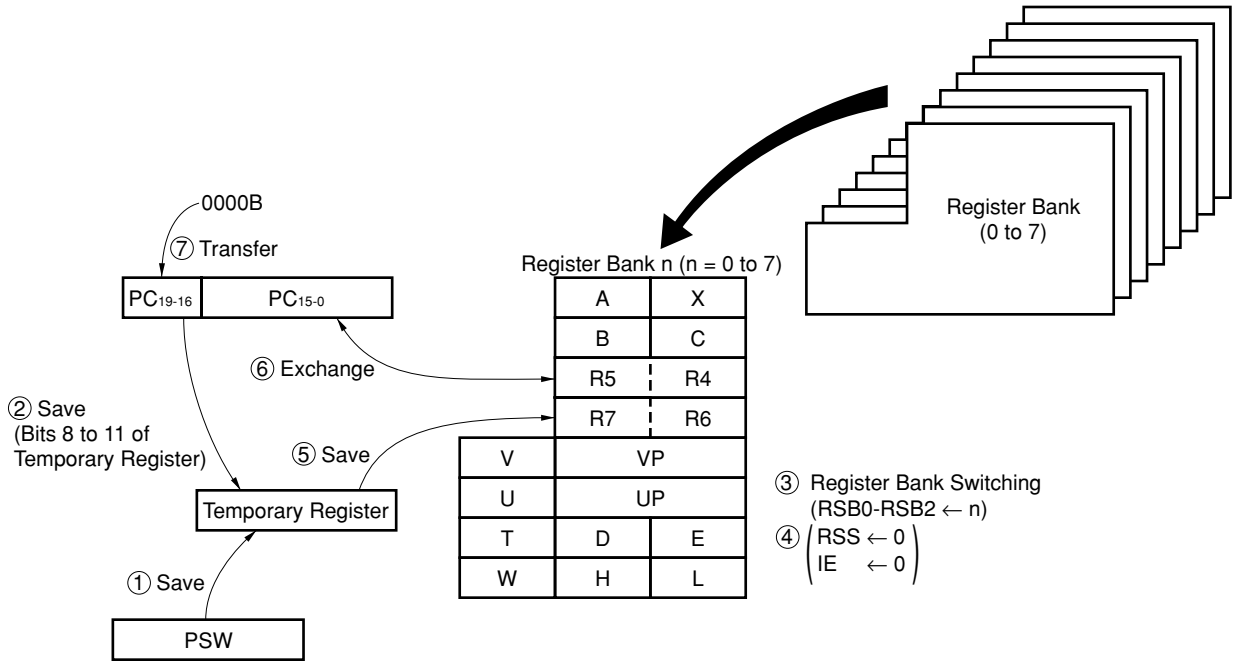
22.4.2 BRKCS Instruction Software Interrupt (Software Context Switching) Acknowledgment Operation

The context switching function can be initiated by executing a BRKCS instruction.

The register bank to be used after context switching is specified by the BRKCS instruction operand.

When a BRKCS instruction is executed, the program branches to the start address of the interrupt service program (which must be in the base area) stored beforehand in the specified register bank, and the contents of the program status word (PSW) and program counter (PC) are saved in the register bank.

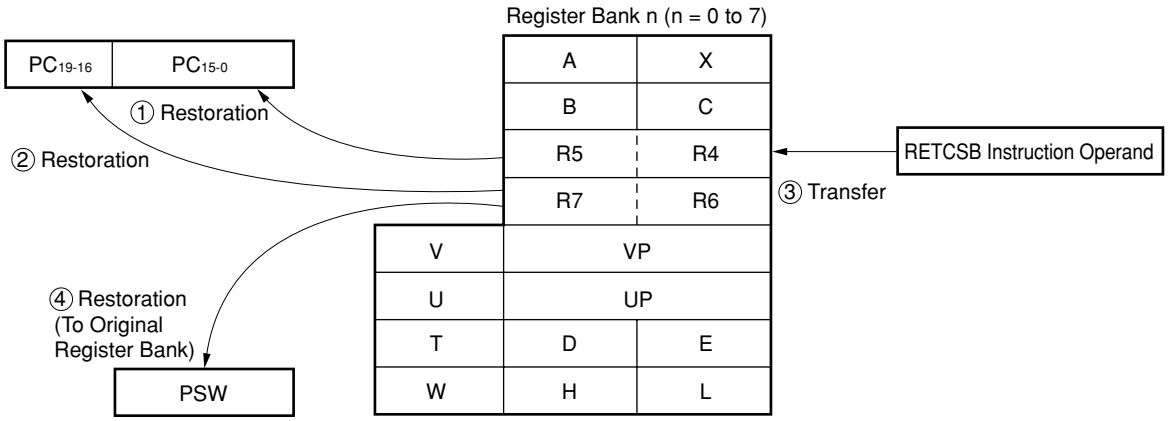
Figure 22-7 Context Switching Operation by Execution of a BRKCS Instruction



The RETCSB instruction is used to return from a software interrupt due to a BRKCS instruction. The RETCSB instruction must specify the start address of the interrupt service program for the next time context switching is performed by a BRKCS instruction. This interrupt service program start address must be in the base area.

Caution The RETCS instruction must not be used to return from a BRKCS instruction software interrupt.

Figure 22-8 Return from BRKCS Instruction Software Interrupt (RETCSB Instruction Operation)



22.5 OPERAND ERROR INTERRUPT ACKNOWLEDGMENT OPERATION

An operand error interrupt is generated when the data obtained by inverting all the bits of the 3rd byte of the operand of an MOV STBC, #byte instruction or LOCATION instruction or an MOV WDM,#byte instruction does not match the 4th byte of the operand. Operand error interrupts cannot be disabled.

When an operand error interrupt is generated, the program status word (PSW) and the start address of the instruction that caused the error are saved to the stack, the IE flag is cleared (to 0), the vector table value is loaded into the program counter (PC), and a branch is performed (within the base area only).

As the address saved to the stack is the start address of the instruction in which the error occurred, simply writing an RETB instruction at the end of the operand error interrupt service program will result in generation of another operand error interrupt. You should therefore either process the address in the stack or initialize the program by referring to **22.12 RESTORING INTERRUPT FUNCTION TO INITIAL STATE**.

22.6 NON-MASKABLE INTERRUPT ACKNOWLEDGMENT OPERATION

Non-maskable interrupts are acknowledged even in the interrupt disabled state. Non-maskable interrupts can be acknowledged at all times except during execution of the service program for an identical non-maskable interrupt or a non-maskable interrupt of higher priority.

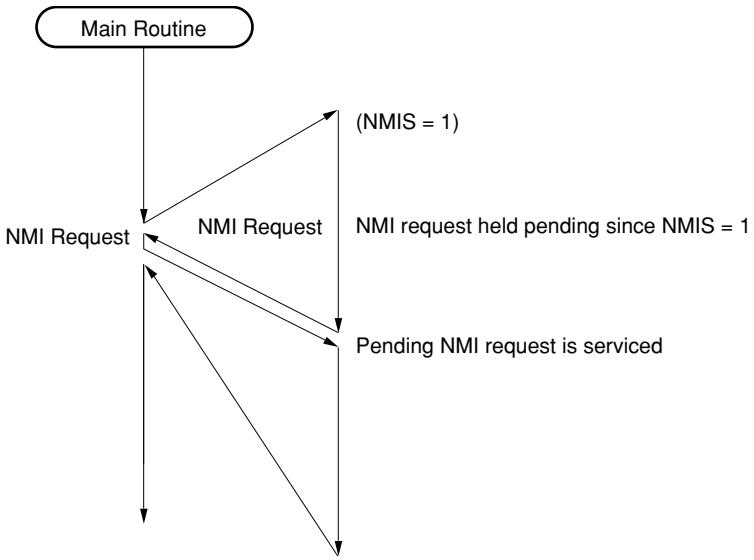
The relative priorities of non-maskable interrupts are set by the PRC bit of the watchdog timer mode register (WDM) (see **22.3.5 Watchdog Timer Mode register (WDM)**).

Except in the cases described in **22.9 WHEN INTERRUPT REQUESTS AND MACRO SERVICE ARE TEMPORARILY HELD PENDING**, a non-maskable interrupt request is acknowledged immediately. When a non-maskable interrupt request is acknowledged, the program status word (PSW) and program counter (PC) are saved in that order to the stack, the IE flag is cleared (to 0), the in-service priority register (ISPR) bit corresponding to the acknowledged non-maskable interrupt is set (to 1), the vector table contents are loaded into the PC, and a branch is performed. The ISPR bit that is set (to 1) is the NMIS bit in the case of a non-maskable interrupt due to edge input to the NMI pin, and the WDTS bit in the case of watchdog timer overflow.

When the non-maskable interrupt service program is executed, non-maskable interrupt requests of the same priority as the non-maskable interrupt currently being executed and non-maskable interrupts of lower priority than the non-maskable interrupt currently being executed are held pending. A pending non-maskable interrupt is acknowledge after completion of the non-maskable interrupt service program currently being executed (after execution of the RETI instruction). However, even if the same non-maskable interrupt request is generated more than once during execution of the non-maskable interrupt service program, only one non-maskable interrupt is acknowledged after completion of the non-maskable interrupt service program.

Figure 22-9 Non-Maskable Interrupt Request Acknowledgment Operations (1/2)

(a) When a new NMI request is generated during NMI service program execution



(b) When a watchdog timer interrupt request is generated during NMI service program execution (when the watchdog timer interrupt priority is higher (when PRC in the WDM = 1))

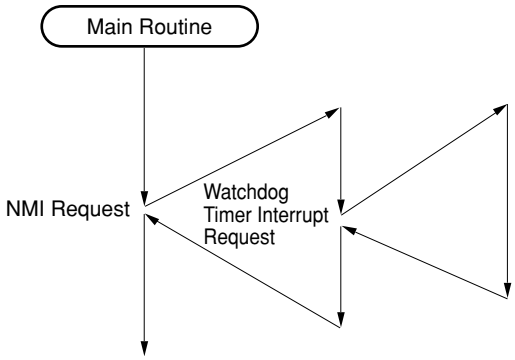
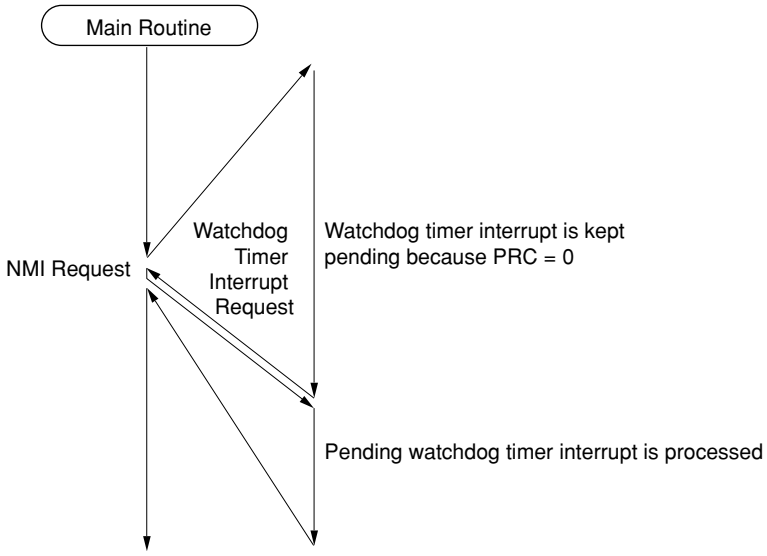
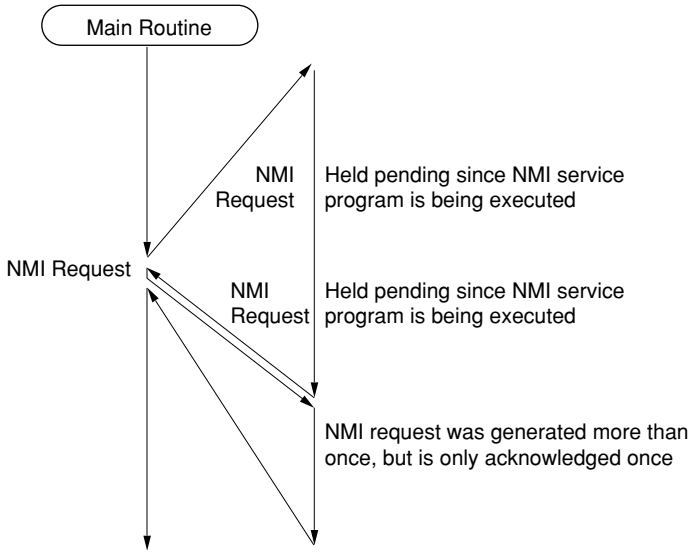


Figure 22-9 Non-Maskable Interrupt Request Acknowledgment Operations (2/2)

(c) When a watchdog timer interrupt request is generated during NMI service program execution (when the NMI interrupt priority is higher (when PRC in the WDM = 0))



(d) When an NMI request is generated twice during NMI service program execution



Cautions 1. Macro service requests are acknowledged and serviced even during execution of a non-maskable interrupt service program. If you do not want macro service processing to be performed during a non-maskable interrupt service program, you should manipulate the interrupt mask register in the non-maskable interrupt service program to prevent macro service generation.

2. The RETI instruction must be used to return from a non-maskable interrupt. Subsequent interrupt acknowledgment will not be performed normally if a different instruction is used. To resume program execution from the initial state after the non-maskable interrupt has been acknowledged, see 22.12 RESTORING INTERRUPT FUNCTION TO INITIAL STATE.

3. Non-maskable interrupts are always acknowledged, except during non-maskable interrupt service program execution (except when a high non-maskable interrupt request is generated during execution of a low-priority non-maskable interrupt service program) and for a certain period after execution of the special instructions shown in 22.9. Therefore, a non-maskable interrupt will be acknowledged even when the stack pointer (SP) value is undefined, in particular after reset release, etc. In this case, depending on the value of the SP, it may happen that the program counter (PC) and program status word (PSW) are written to the address of a write-inhibited special function register (SFR) (see Table 3.5 in 3.9 Special Function Registers (SFR)), and the CPU becomes deadlocked, or an unexpected signal is output from a pin, or the PC and PSW are written to an address in which RAM is not mounted, with the result that the return from the non-maskable interrupt service program is not performed normally and a software upsets occurs.
Therefore, the program following $\overline{\text{RESET}}$ release must be as shown below.

```
        CSEG AT 0
        DW   STRT
        CSEG BASE
STRT:
        LOCATION 0FH      ; or LOCATION 0H
        MOVG SP, #imm24
```

22.7 MASKABLE INTERRUPT ACKNOWLEDGMENT OPERATION

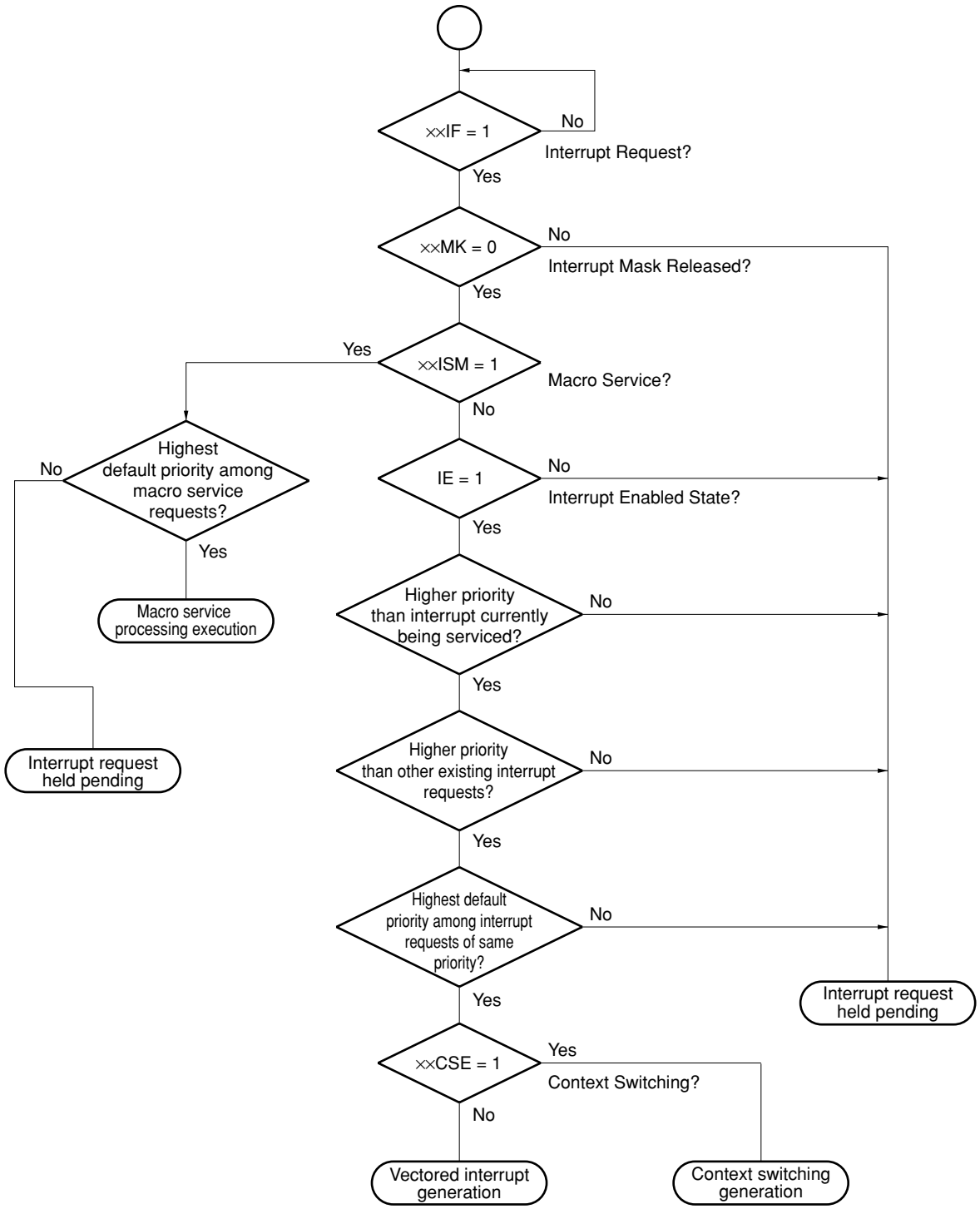
A maskable interrupt can be acknowledged when the interrupt request flag is set (to 1) and the mask flag for that interrupt is cleared (to 0). When servicing is performed by macro service, the interrupt is acknowledged and serviced by macro service immediately. In the case of vectored interrupt and context switching, an interrupt is acknowledged in the interrupt enabled state (when the IE flag is set (to 1)) if the priority of that interrupt is one for which acknowledgment is permitted.

If maskable interrupt requests are generated simultaneously, the interrupt for which the highest priority is specified by the priority specification flag is acknowledged. If the interrupts have the same priority specified, they are acknowledged in accordance with their default priorities.

A pending interrupt is acknowledged when a state in which it can be acknowledged is established.

The interrupt acknowledgment algorithm is shown in Figure 22-10.

Figure 22-10 Interrupt Acknowledgment Processing Algorithm



22.7.1 Vectored Interruption

When a vectored interruption maskable interrupt request is acknowledged, the program status word (PSW) and program counter (PC) are saved in that order to the stack, the IE flag is cleared (to 0) (the interrupt disabled state is set), and the in-service priority register (ISPR) bit corresponding to the priority of the acknowledged interrupt is set (to 1). Also, data in the vector table predetermined for each interrupt request is loaded into the PC, and a branch is performed. The return from a vectored interrupt is performed by means of the RETI instruction.

Caution When a maskable interrupt is acknowledged by vectored interruption, the RETI instruction must be used to return from the interrupt. Subsequent interrupt acknowledgment will not be performed normally if a different instruction is used.

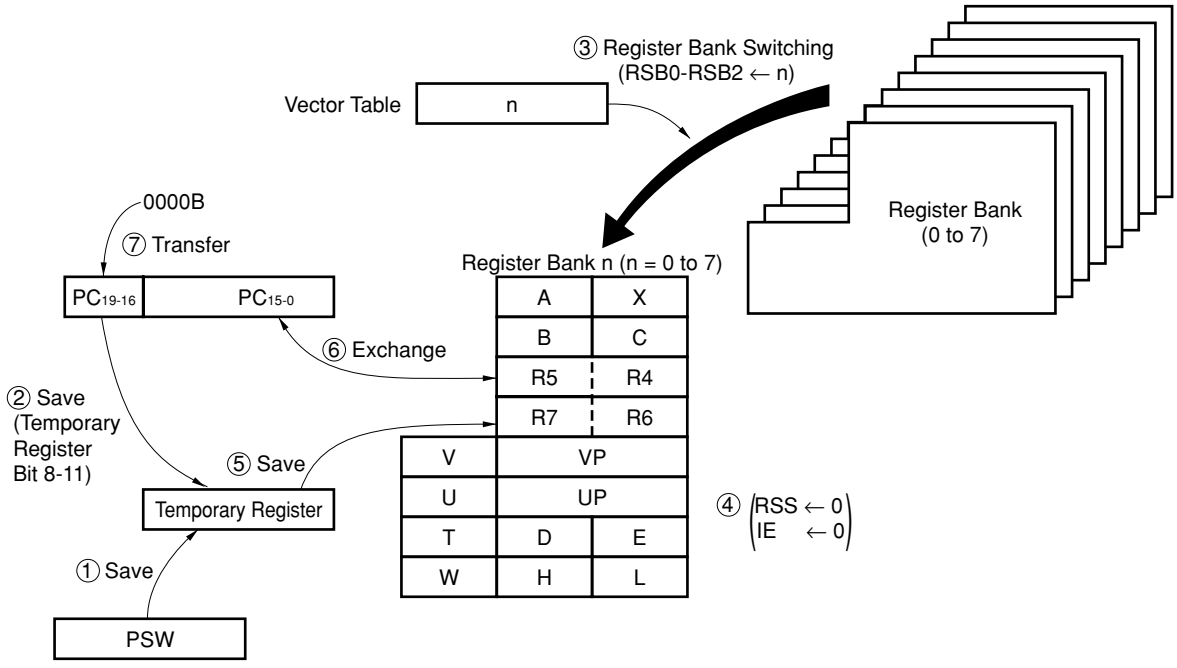
22.7.2 Context Switching

Initiation of the context switching function is enabled by setting (to 1) the context switching enable flag of the interrupt control register.

When an interrupt request for which the context switching function is enabled is acknowledged, the register bank specified by 3 bits of the lower address (even address) of the corresponding vector table address is selected.

The vector address stored beforehand in the selected register bank is transferred to the program counter (PC), and at the same time the contents of the PC and program status word (PSW) up to that time are saved in the register bank and a branch is made to the interrupt service program.

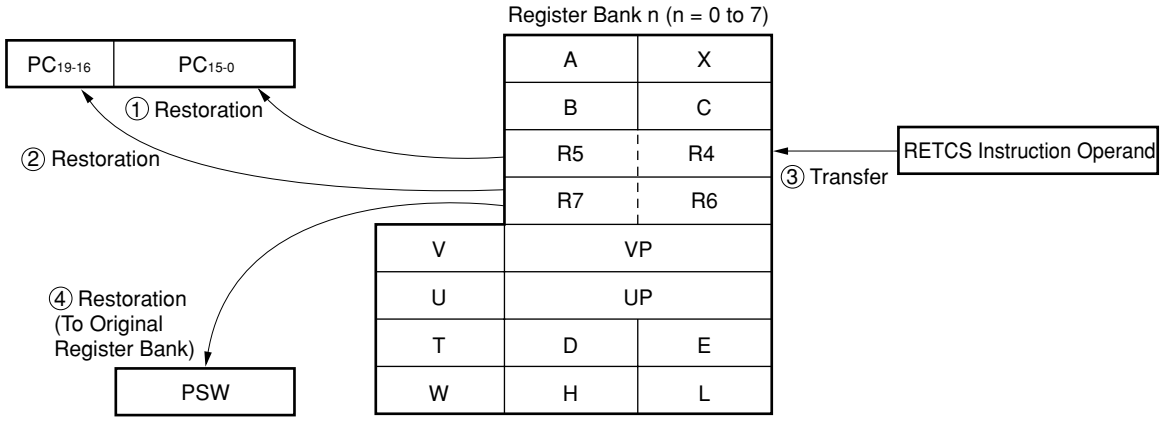
Figure 22-11 Context Switching Operation by Generation of an Interrupt Request



The RETCS instruction is used to return from an interrupt that uses the context switching function. The RETCS instruction must specify the start address of the interrupt service program to be executed when that interrupt is acknowledged next. This interrupt service program start address must be in the base area.

Caution The RETCS instruction must be used to return from an interrupt serviced by context switching. Subsequent interrupt acknowledgment will not be performed normally if a different instruction is used.

Figure 22-12 Return from Interrupt that Uses Context Switching by Means of RETCS Instruction



22.7.3 Maskable Interrupt Priority Levels

The μPD784038 performs multiple interrupt servicing in which an interrupt is acknowledged during servicing of another interrupt. Multiple interrupts can be controlled by priority levels.

There are two kinds of priority control, control by default priority and programmable priority control in accordance with the setting of the priority specification flag. In priority control by means of default priority, interrupt service is performed in accordance with the priority preassigned to each interrupt request (default priority) (see **Table 22-2**). In programmable priority control, interrupt requests are divided into four levels according to the setting of the priority specification flag. Interrupt requests for which multiple interruption is permitted are shown in Table 22-5.

Since the IE flag is cleared (to 0) automatically when an interrupt is acknowledged, when multiple interruption is used, the IE flag should be set (to 1) to enable interrupts by executing an EI instruction in the interrupt service program, etc.

Table 22-5 Multiple Interrupt Servicing

Priority of Interrupt Currently Being Acknowledged	ISPR Value	IE Flag in PSW	PRSL in IMC Register	Acknowledgeable Maskable Interrupts
No interrupt being acknowledged	00000000	0	×	• All macro service only
		1	×	• All maskable interrupts
3	00001000	0	×	• All macro service only
		1	0	• All maskable interrupts
		1	1	• All macro service • Maskable interrupts specified as priority 0/1/2
2	0000×100	0	×	• All macro service only
		1	×	• All macro service • Maskable interrupts specified as priority 0/1
1	0000××10	0	×	• All macro service only
		1	×	• All macro service • Maskable interrupts specified as priority 0
0	0000×××1	×	×	• All macro service only
Non-maskable interrupts	1000×××× 0100×××× 1100××××	×	×	• All macro service only

Figure 22-13 Examples of Servicing When Another Interrupt Request is Generated During Interrupt Service (1/3)

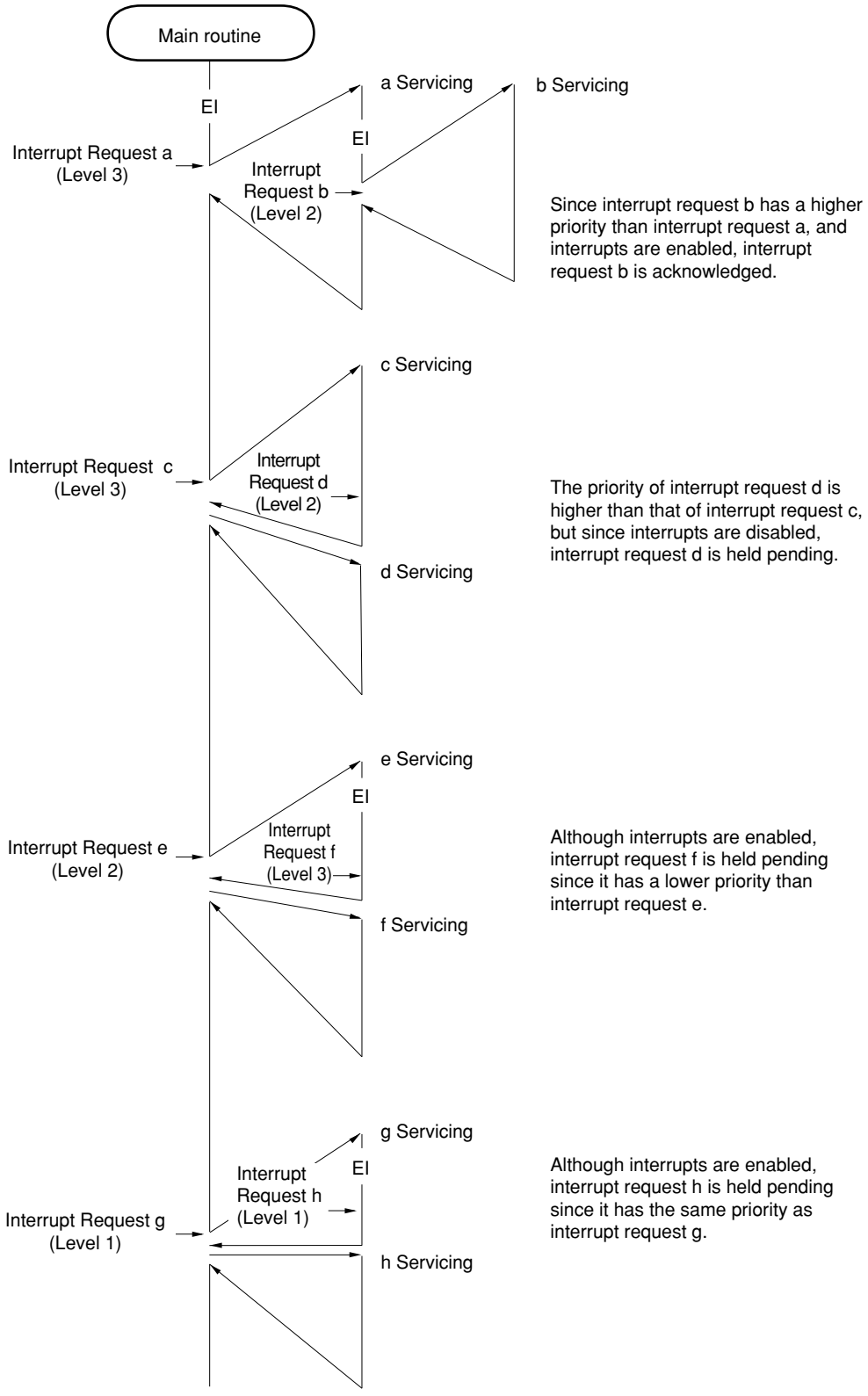
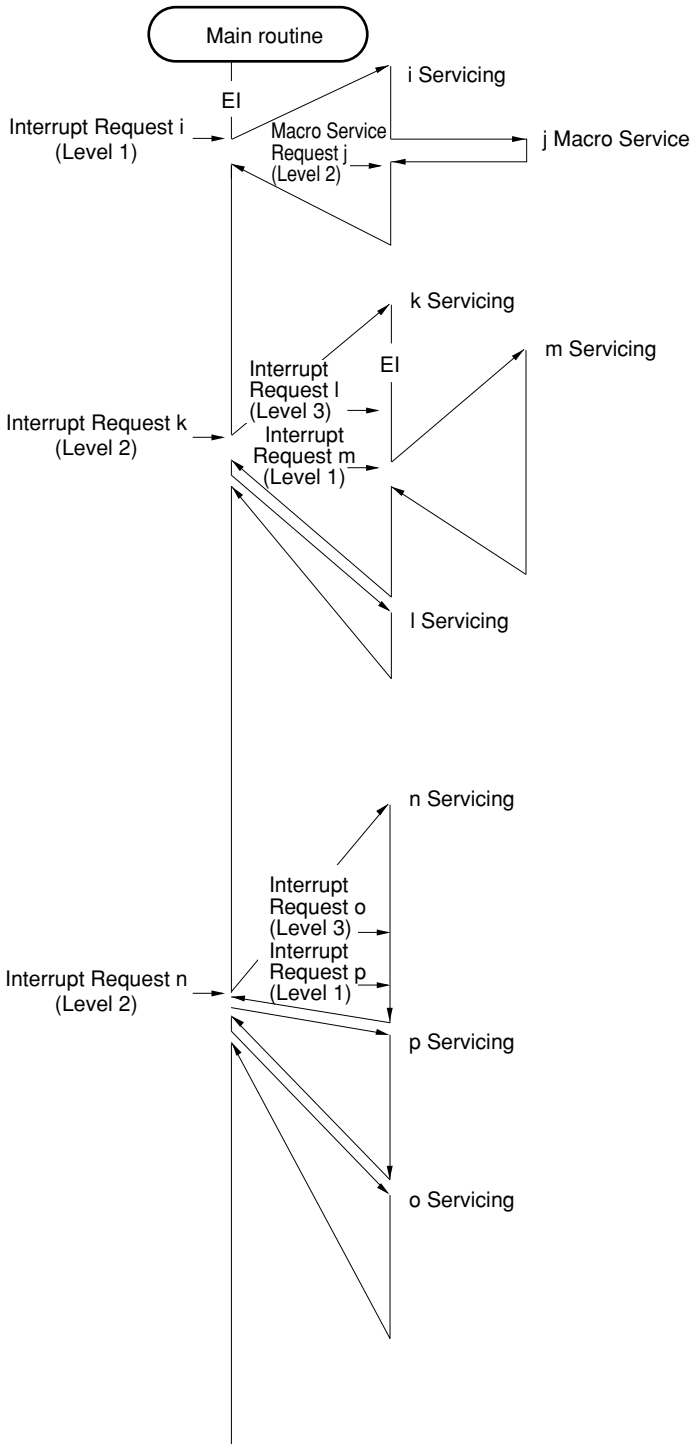


Figure 22-13 Examples of Servicing When Another Interrupt Request is Generated During Interrupt Service (2/3)

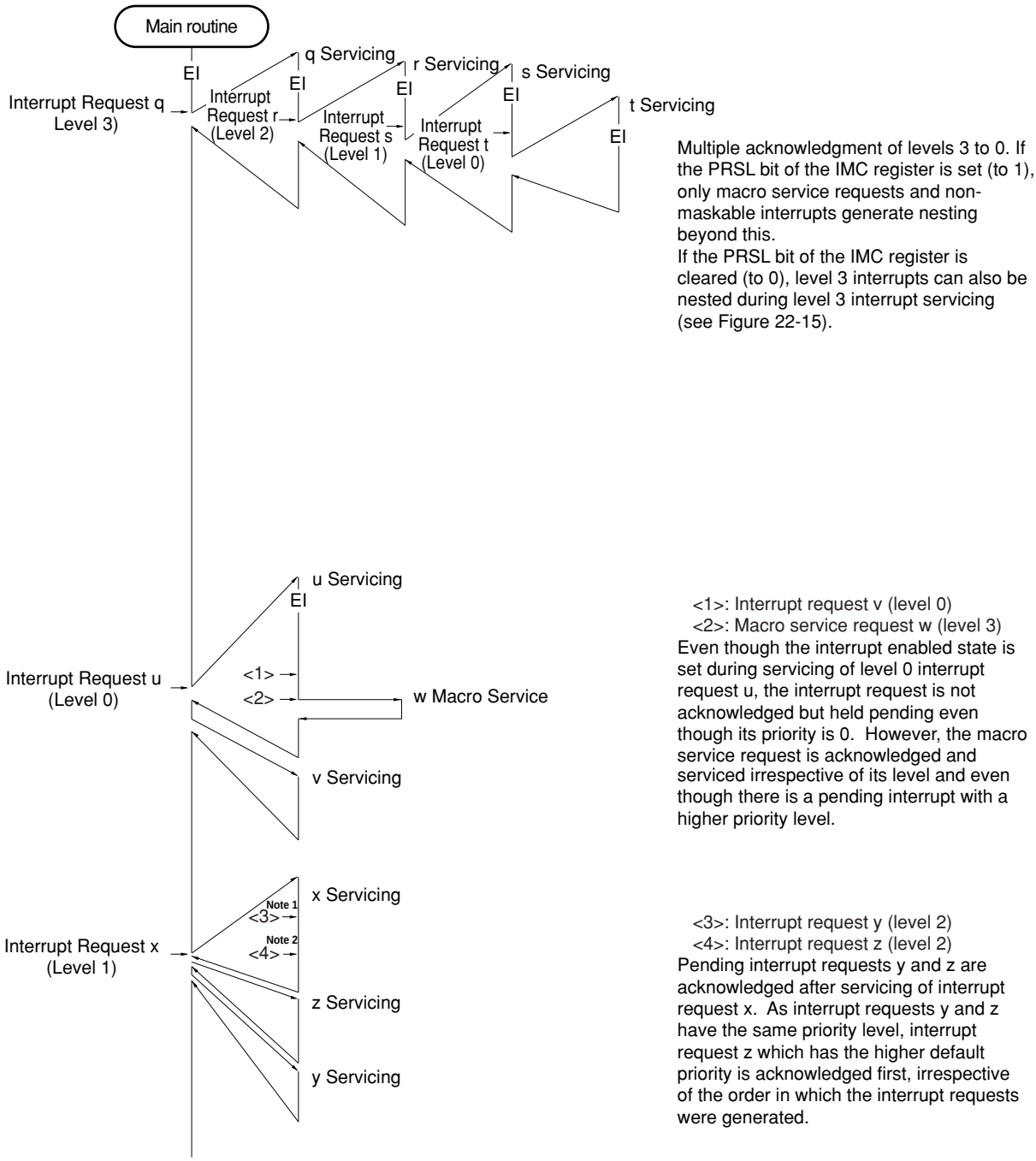


The macro service request is serviced irrespective of interrupt enabling/disabling and priority.

The interrupt request is held pending since it has a lower priority than interrupt request k. Interrupt request m generated after interrupt request l has a higher priority, and is therefore acknowledged first.

Since servicing of interrupt request n performed in the interrupt disabled state, interrupt requests o and p are held pending.
After interrupt request n servicing, the pending interrupt requests are acknowledged. Although interrupt request o was generated first, interrupt request p has a higher priority and is therefore acknowledged first.

Figure 22-13 Examples of Servicing When Another Interrupt Request is Generated During Interrupt Service (3/3)



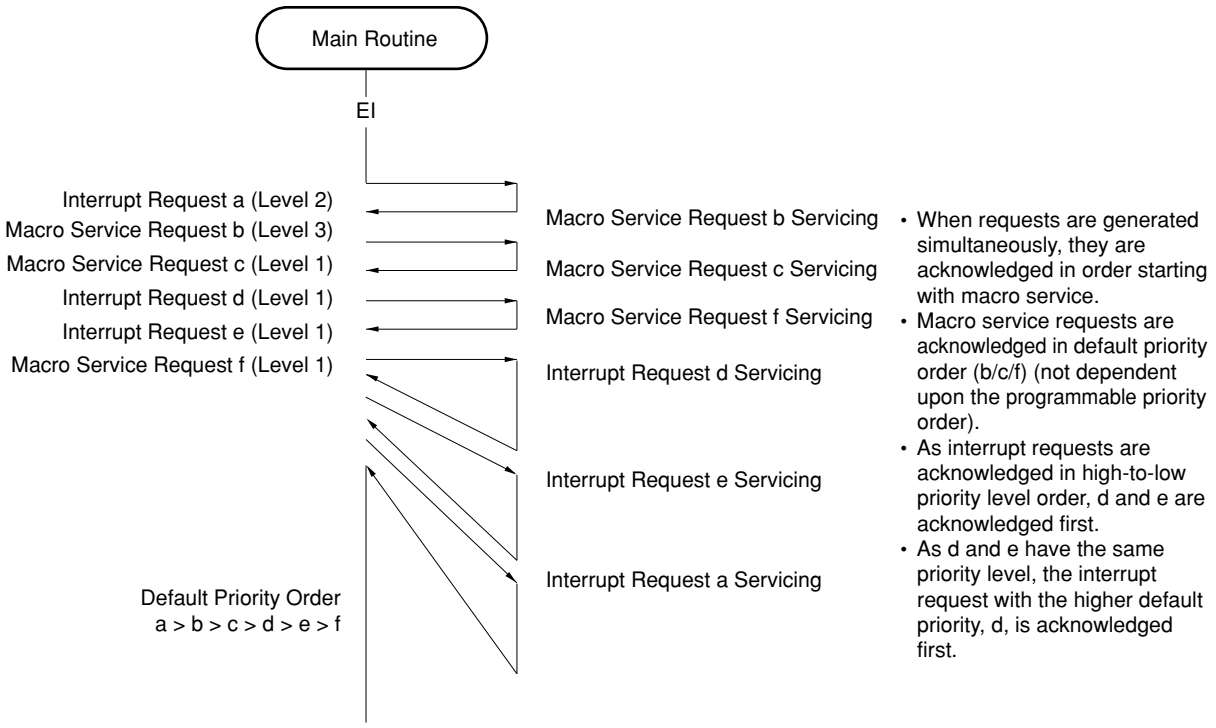
Notes 1. Low default priority

2. High default priority

Remarks 1. “a” to “z” in the figure are arbitrary names used to differentiate between the interrupt requests and macro service requests.

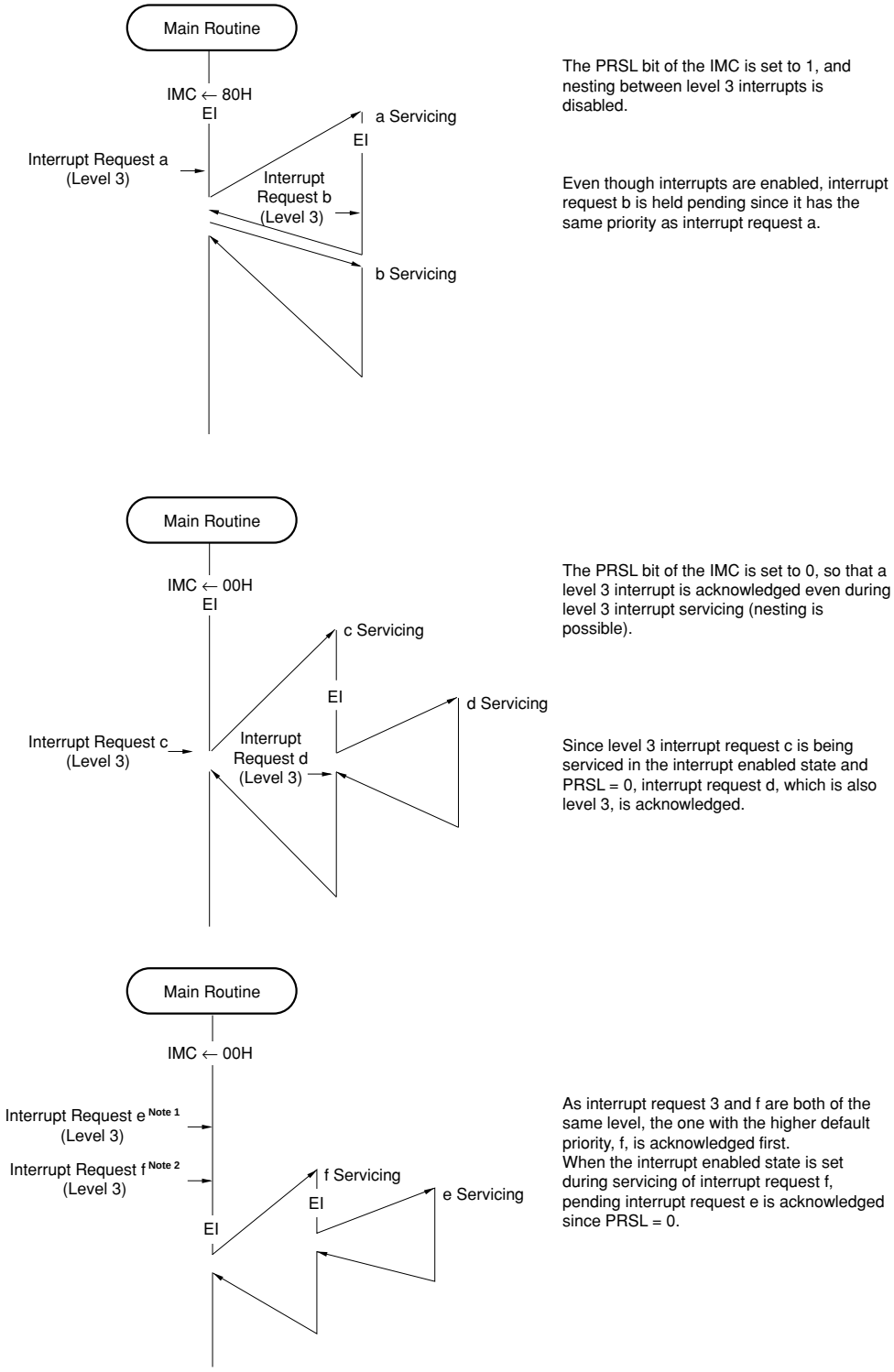
2. High/low default priorities in the figure indicate the relative priority levels of the two interrupt requests.

Figure 22-14 Examples of Servicing of Simultaneously Generated Interrupts



Remark “a” to “f” in the figure are arbitrary names used to differentiate between the interrupt requests and macro service requests.

Figure 22-15 Differences in Level 3 Interrupt Acknowledgment According to IMC Register Setting



Notes 1. Low default priority

2. High default priority

Remarks 1. “a” to “f” in the figure are arbitrary names used to differentiate between the interrupt requests and macro service requests.

2. High/low default priorities in the figure indicate the relative priority levels of the two interrupt requests.

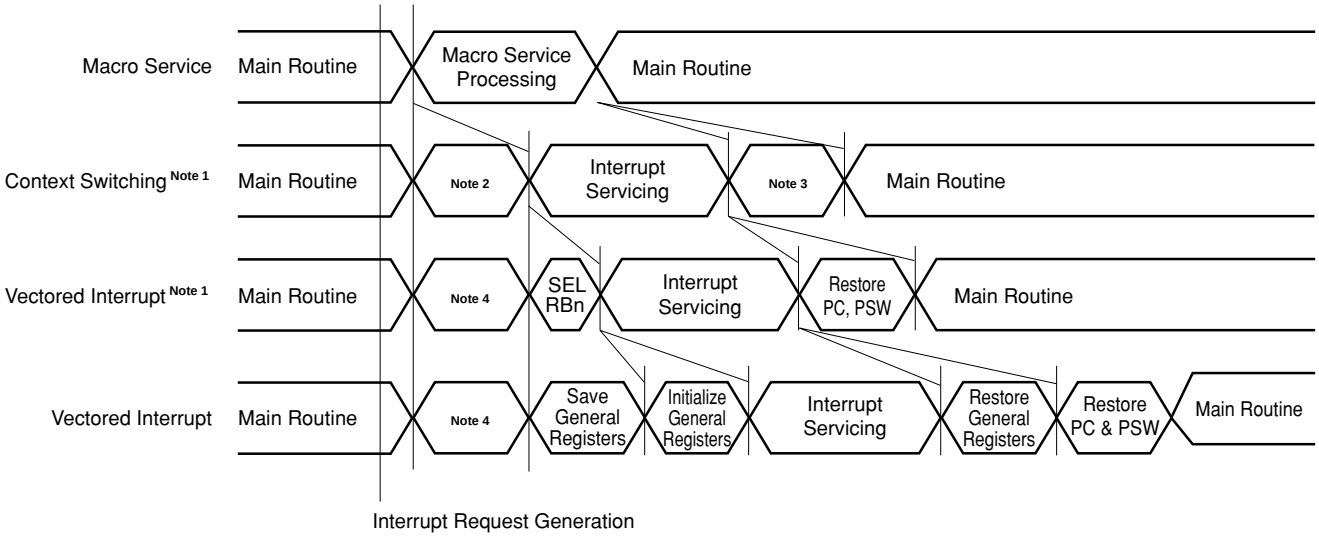
22.8 MACRO SERVICE FUNCTION

22.8.1 Outline of Macro Service Function

Macro service is one method of servicing interrupts. With a normal interrupt, the program counter (PC) and program status word (PSW) are saved, and the start address of the interrupt service program is loaded into the PC, but with macro service, different processing (mainly data transfers) is performed instead of this processing. This enables interrupt requests to be responded to quickly, and moreover, since transfer processing is faster than processing by a program, the processing time can also be reduced.

Also, since a vectored interrupt is generated after processing has been performed the specified number of times, another advantage is that vectored interrupt programs can be simplified.

Figure 22-16 Differences between Vectored Interrupt and Macro Service Processing



Notes 1. When register bank switching is used, and an initial value has been set in the register beforehand

2. Register bank switching by context switching, saving of PC and PSW

3. Register bank, PC and PSW restoration by context switching

4. PC and PSW saved to the stack, vector address loaded into PC

22.8.2 Types of Macro Service

Macro service can be used with the 19 kinds **Note** of interrupt shown in Table 22-6. There are four kinds of operation, which can be used to suit the application.

Note Twenty types with the μPD784038Y Subseries

Table 22-6 Interrupts for Which Macro Service Can be Used

Default Priority	Interrupt Request Generation Source	Generating Unit	Macro Service Control Word Address
0	INTP0 (pin input edge detection)	Edge detection	0FE06H
1	INTP1 (pin input edge detection)		0FE08H
2	INTP2 (pin input edge detection)		0FE0AH
3	INTP3 (pin input edge detection)		0FE0CH
4	INTC00 (TM0-CR00 match signal generation)	Timer/counter 0	0FE0EH
5	INTC01 (TM0-CR01 match signal generation)		0FE10H
6	INTC10 (TM1-CR10 or TM1W-CR10W match signal generation)	Timer/counter 1	0FE12H
7	INTC11 (TM1-CR11 or TM1W-CR11W match signal generation)		0FE14H
8	INTC20 (TM2-CR20 or TM2W-CR20W match signal generation)	Timer/counter 2	0FE16H
9	INTC21 (TM2-CR21 or TM2W-CR21W match signal generation)		0FE18H
10	INTC30 (TM3-CR30 or TM3W-CR30W match signal generation)	Timer 3	0FE1AH
11	INTP4 (pin input edge detection)	Edge detection	0FE1CH
12	INTP5 (pin input edge detection)		0FE1EH
13	INTAD (A/D conversion end)	A/D converter	0FE20H
14	INTSR (asynchronous serial interface reception end)	Asynchronous serial interface/ clocked serial interface 1	0FE24H
	INTCSI1 (clocked serial interface transfer end)		
15	INTST (asynchronous serial interface transmission end)		0FE26H
16	INTCSI (clocked serial interface transfer end)	Clocked serial interface	0FE28H
17	INTSR2 (asynchronous serial interface 2 reception end)	Asynchronous serial interface 2/ clocked serial interface 2	0FE2CH
	INTCSI2 (clocked serial interface 2 transfer end)		
18	INTST2 (asynchronous serial interface 2 transmission end)		0FE2EH
19 **Note**	INTSPC (I^2C bus stop condition interrupt)	Clocked serial interface	0FE30H

Note μPD784038Y Subseries only

Remarks 1. The default priority is a fixed number. This indicates the order of priority when macro service requests are generated simultaneously,

2. The INTSR and INTCSI1 interrupts are generated by the same hardware (they cannot both be used simultaneously). Therefore, although the same hardware is used for the interrupts, two names are provided, for use in each of the two modes. The same applies to INTSR2 and INTCSI2.

There are four kinds of macro service, as shown below.

(1) Type A

One byte or one word of data is transferred between a special function register (SFR) and memory each time an interrupt request is generated, and a vectored interrupt request is generated when the specified number of transfers have been performed.

Memory that can be used in the transfers is limited to internal RAM addresses 0FE00H to 0FEFFH when the LOCATION 0H instruction is executed, and addresses 0FFE00H to 0FFEFFH when the LOCATION 0FH instruction is executed.

The specification method is simple and is suitable for low-volume, high-speed data transfers.

(2) Type B

As with type A, one byte or one word of data is transferred between a special function register (SFR) and memory each time an interrupt request is generated, and a vectored interrupt request is generated when the specified number of transfers have been performed.

The SFR and memory to be used in the transfers is specified by the macro service channel (the entire 1-Mbyte memory space can be used).

This is a general version of type A, suitable for large volumes of transfer data.

(3) Type C

Data is transferred from memory to two special function registers (SFR) each time an interrupt request is generated, and a vectored interrupt request is generated when the specified number of transfers have been performed.

With type C macro service, not only are data transfers performed to two locations in response to a single interrupt request, but it is also possible to add output data ring control and a function that automatically adds data to a compare register. The entire 1-Mbyte memory space can be used.

Type C is mainly used with the INTC10 and INTC11 interrupts, and is used for stepping motor control, etc., by macro service, with P0L or P0H and CR10, CR10W, CR11 and CR11W used as the SFRs to which data is transferred.

(4) Counter mode

This mode is to decrement the macro service counter (MSC) when an interrupt occurs and is used to count the division operation of an interrupt and interrupt generation circuit.

When MSC is 0, a vector interrupt can be generated.

To restart the macro service, MSC must be set again.

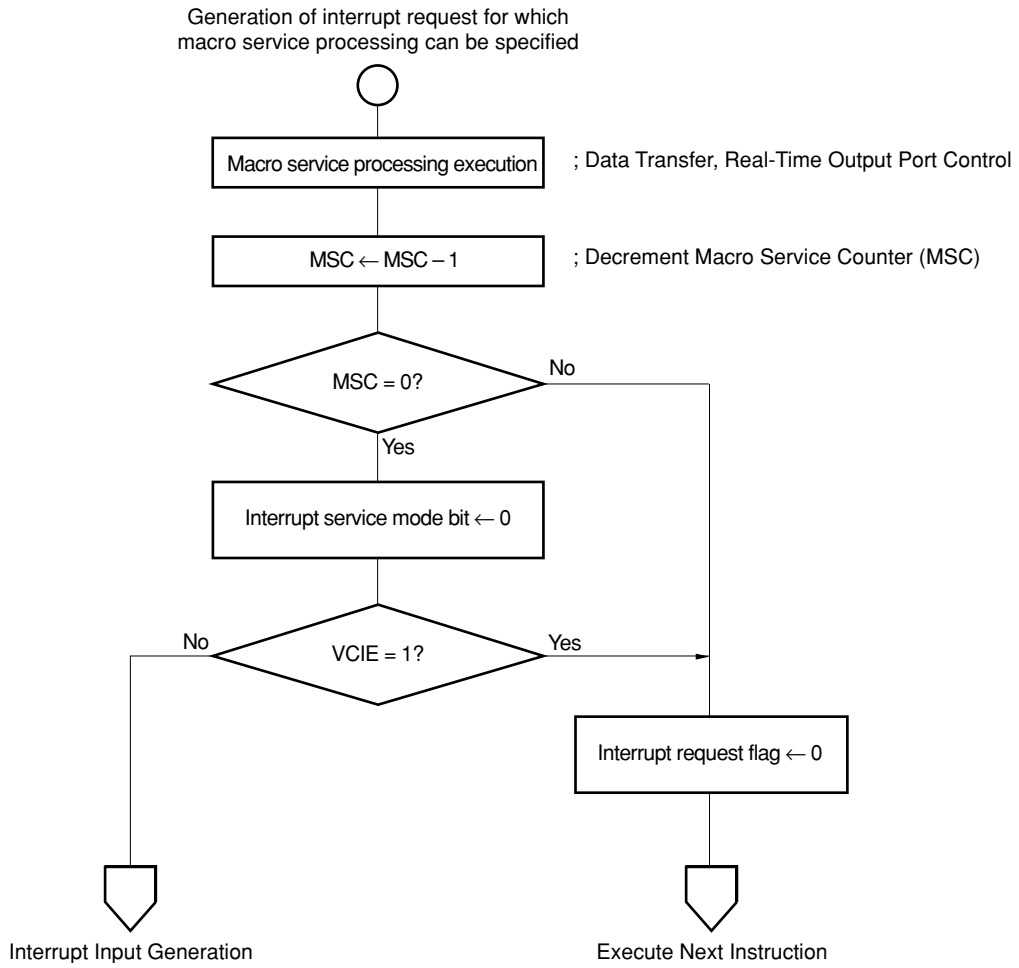
MSC is fixed to 16 bits and cannot be used as an 8-bit counter.

22.8.3 Basic Macro Service Operation

Interrupt requests for which the macro service processing generated by the algorithm shown in Figure 22-10 can be specified are basically serviced in the sequence shown in Figure 22-17.

Interrupt requests for which macro service processing can be specified are not affected by the status of the IE flag, but are disabled by setting (to 1) an interrupt mask flag in the interrupt mask register (MK0). Macro service processing can be executed in the interrupt disabled state and during execution of an interrupt service program.

Figure 22-17 Macro Service Processing Sequence



The macro service type and transfer direction are determined by the value set in the macro service control word mode register. Transfer processing is then performed using the macro service channel specified by the channel pointer according to the macro service type.

The macro service channel is memory which contains the macro service counter which records the number of transfers, the transfer destination and transfer source pointers, and data buffers, and can be located at any address in the range FE00H to FEFFH when the LOCATION 0H instruction is executed, or FFE00H to FFEFFH when the LOCATION 0FH instruction is executed.

22.8.4 Operation at End of Macro Service

In macro service, processing is performed the number of times specified during execution of another program. Macro service ends when the processing has been performed the specified number of times (when the macro service counter (MSC) reaches 0). Either of two operations may be performed at this point, as specified by the VCIE bit (bit 7) of the macro service mode register for each macro service.

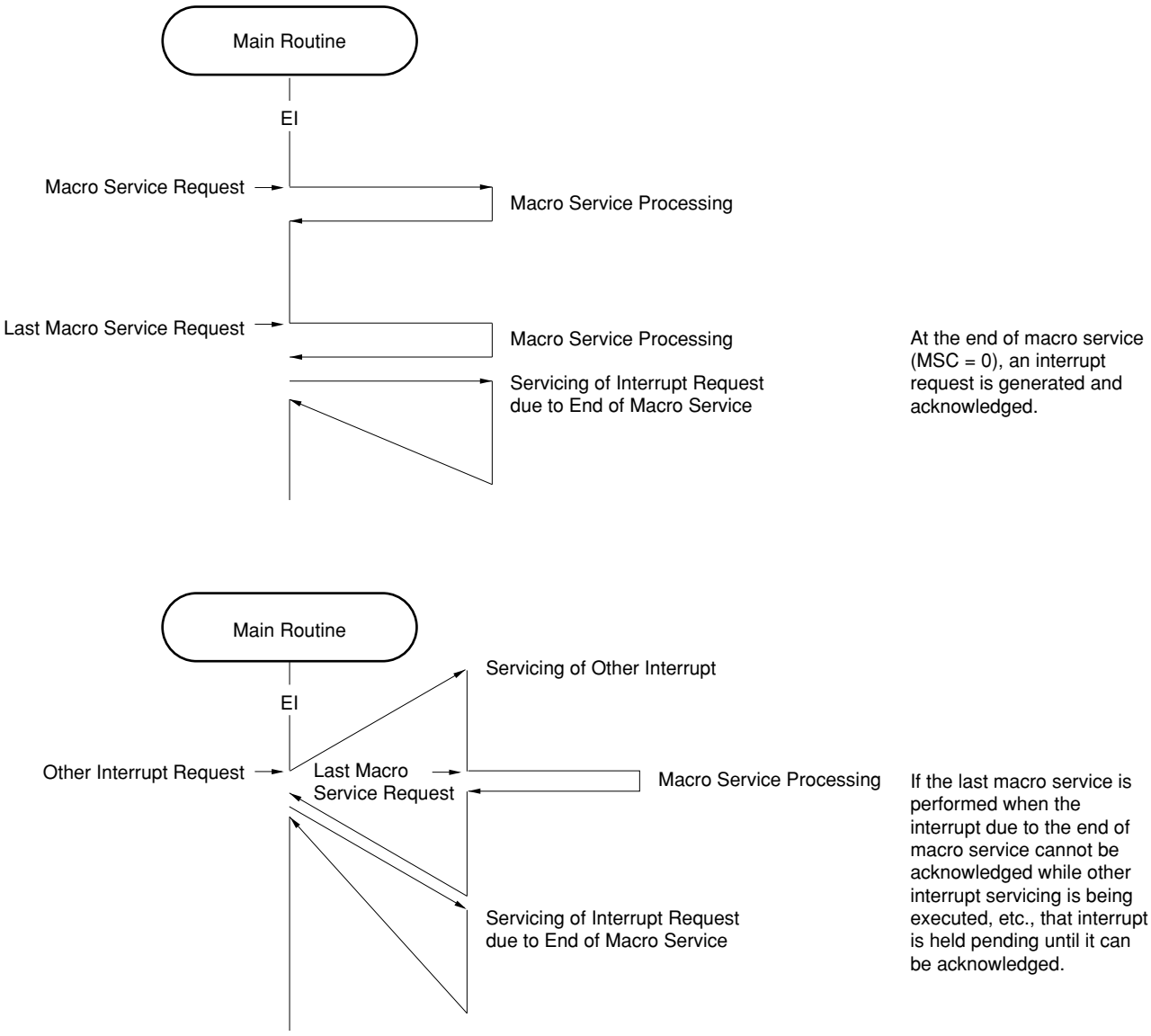
(1) When VCIE bit is 0

In this mode, an interrupt is generated as soon as the macro service ends. Figure 22-18 shows an example of macro service and interrupt acknowledgment operations when the VCIE bit is 0.

This mode is used when a series of operations end with the last macro service processing performed, for instance. It is mainly used in the following cases:

- Asynchronous serial interface receive data buffering (INTSR/INTSR2)
- A/D conversion result fetch (INTAD)
- Compare register update as the result of a match between a timer register and the compare register (INTC00/INTC01/INTC10/INTC11/INTC20/INTC21/INTC30)
- Timer/counter capture register read due to edge input to the INTPn pin (INTP0/INTP1/INTP2/INTP3)

Figure 22-18 Operation at End of Macro Service When VCIE = 0



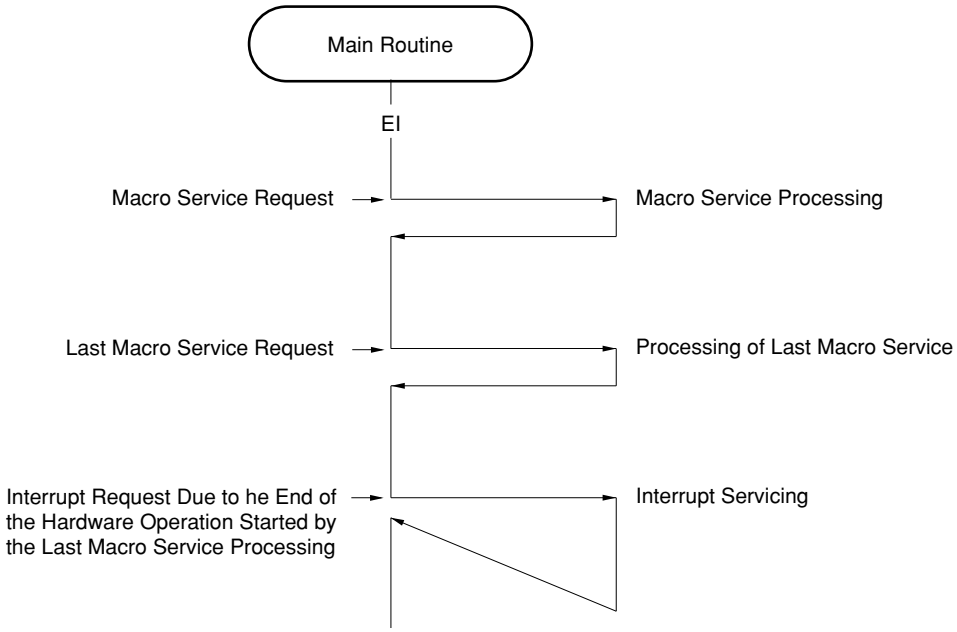
(2) When VCIE bit is 1

In this mode, an interrupt is not generated after macro service ends. Figure 22-19 shows an example of macro service and interrupt acknowledgment operations when the VCIE bit is 1.

This mode is used when the final operation is to be started by the last macro service processing performed, for instance. It is mainly used in the following cases:

- Clocked serial interface receive data transfers (INTCSI/INTCSI1/INTCSI2)
- Asynchronous serial interface data transfers (INTST/ INTST2)
- To stop a stepping motor in the case (INTC10/INTC11) of stepping motor control by means of macro service type C using the real-time output port and timer/counter.

Figure 22-19 Operation at End of Macro Service When VCIE = 1



22.8.5 Macro Service Control Registers

(1) Macro service control word

The μPD784038's macro service function is controlled by the macro service control mode register and macro service channel pointer. The macro service processing mode is set by means of the macro service mode register, and the macro service channel address is indicated by the macro service channel pointer.

The macro service mode register and macro service channel pointer are mapped onto the part of the internal RAM shown in Figure 22-20 for each macro service as the macro service control word.

When macro service processing is performed, the macro service mode register and channel pointer values corresponding to the interrupt requests for which macro service processing can be specified must be set beforehand.

Figure 22-20 Macro Service Control Word Format

Reserved Word	Address		Source
SPCHP	0FE31	Channel Pointer	INTSPC Note
SPMMD	0FE30	Mode Register	
STCHP2	0FE2F	Channel Pointer	INTST2
STMMD2	0FE2E	Mode Register	
SRCHP2/CSICHP2	0FE2D	Channel Pointer	INTSR2/INTCSI2
SRMMD2/CSIMMD2	0FE2C	Mode Register	
CSICHP	0FE29	Channel Pointer	INTCSI
CSIMMD	0FE28	Mode Register	
STCHP	0FE27	Channel Pointer	INTST
STMMD	0FE26	Mode Register	
SRCHP/CSICHP1	0FE25	Channel Pointer	INTSR/INTCSI1
SRMMD/CSIMMD1	0FE24	Mode Register	
ADCHP	0FE21	Channel Pointer	INTAD
ADMMD	0FE20	Mode Register	
PCHP5	0FE1F	Channel Pointer	INTP5
PMMD5	0FE1E	Mode Register	
PCHP4	0FE1D	Channel Pointer	INTP4
PMMD4	0FE1C	Mode Register	
CCHP30	0FE1B	Channel Pointer	INTC30
CMMD30	0FE1A	Mode Register	
CCHP21	0FE19	Channel Pointer	INTC21
CMMD21	0FE18	Mode Register	
CCHP20	0FE17	Channel Pointer	INTC20
CMMD20	0FE16	Mode Register	
CCHP11	0FE15	Channel Pointer	INTC11
CMMD11	0FE14	Mode Register	
CCHP10	0FE13	Channel Pointer	INTC10
CMMD10	0FE12	Mode Register	
CCHP01	0FE11	Channel Pointer	INTC01
CMMD01	0FE10	Mode Register	
CCHP00	0FE0F	Channel Pointer	INTC00
CMMD00	0FE0E	Mode Register	
PCHP3	0FE0D	Channel Pointer	INTP3
PMMD3	0FE0C	Mode Register	
PCHP2	0FE0B	Channel Pointer	INTP2
PMMD2	0FE0A	Mode Register	
PCHP1	0FE09	Channel Pointer	INTP1
PMMD1	0FE08	Mode Register	
PCHP0	0FE07	Channel Pointer	INTP0
PMMD0	0FE06	Mode Register	

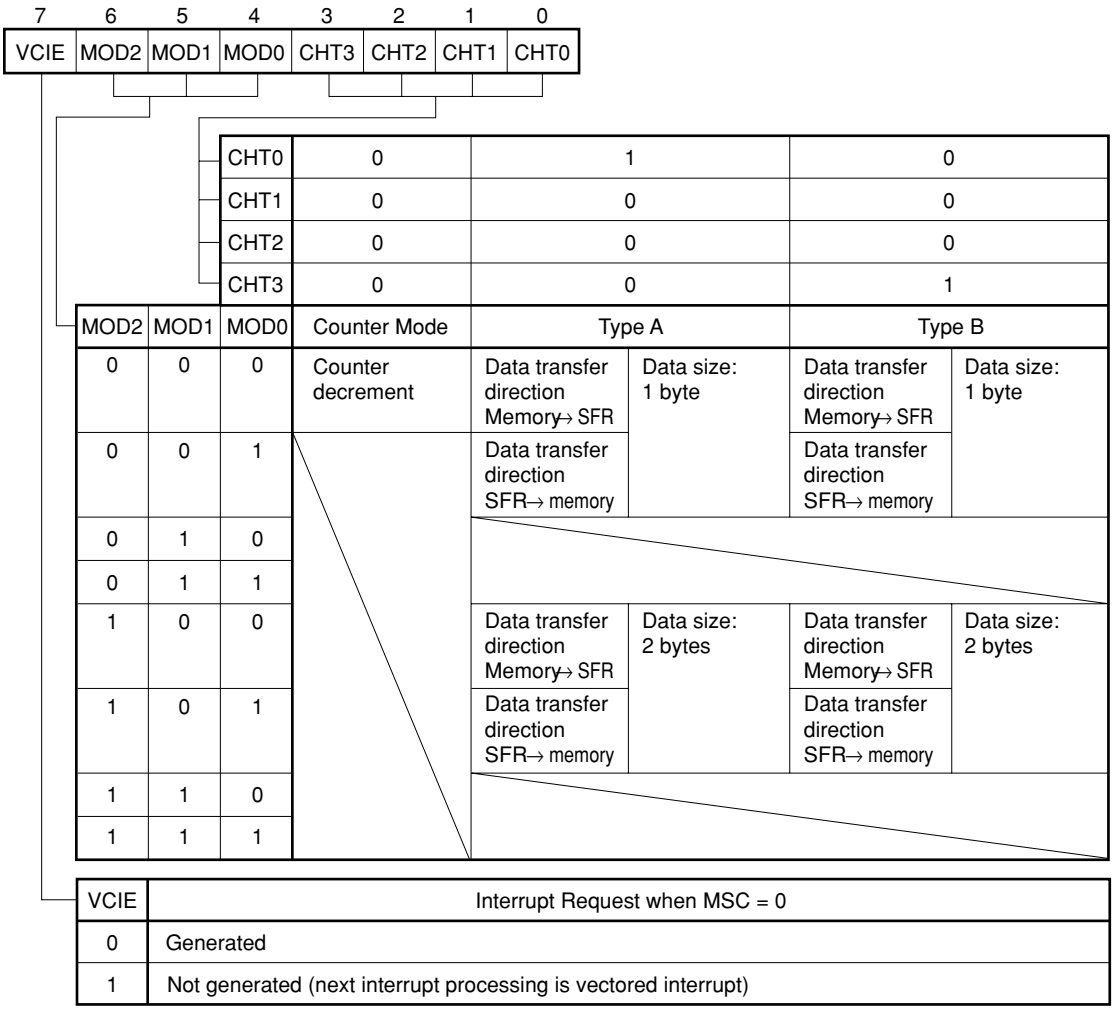
Note μPD784038Y Subseries only

(2) Macro service mode register

The macro service mode register is an 8-bit register that specifies the macro service operation. This register is written in internal RAM as part of the macro service control word (see **Figure 22-20**).

The format of the macro service mode register is shown in Figure 22-21.

Figure 22-21 Macro Service Mode Register Format (1/2)

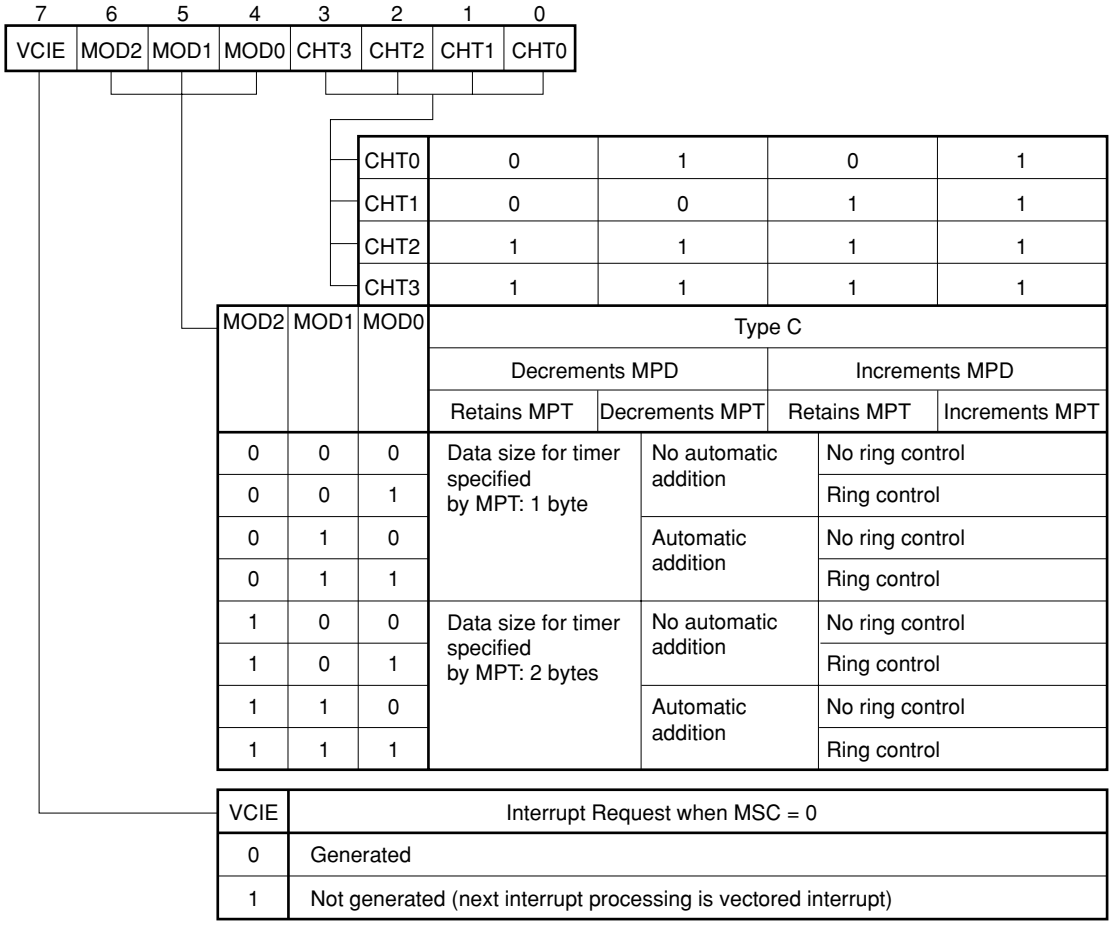


CHT0	0	1		0			
CHT1	0	0		0			
CHT2	0	0		0			
CHT3	0	0		1			

MOD2	MOD1	MOD0	Counter Mode	Type A		Type B	
0	0	0	Counter decrement	Data transfer direction Memory→ SFR	Data size: 1 byte	Data transfer direction Memory→ SFR	Data size: 1 byte
0	0	1		Data transfer direction SFR→ memory		Data transfer direction SFR→ memory	
0	1	0					
0	1	1					
1	0	0		Data transfer direction Memory→ SFR	Data size: 2 bytes	Data transfer direction Memory→ SFR	Data size: 2 bytes
1	0	1		Data transfer direction SFR→ memory		Data transfer direction SFR→ memory	
1	1	0					
1	1	1					

VCIE	Interrupt Request when MSC = 0
0	Generated
1	Not generated (next interrupt processing is vectored interrupt)

Figure 22-21 Macro Service Mode Register Format (2/2)



7	6	5	4	3	2	1	0
VCIE	MOD2	MOD1	MOD0	CHT3	CHT2	CHT1	CHT0

CHT0	0	1	0	1
CHT1	0	0	1	1
CHT2	1	1	1	1
CHT3	1	1	1	1

MOD2	MOD1	MOD0	Type C			
			Decrements MPD		Increments MPD	
			Retains MPT	Decrements MPT	Retains MPT	Increments MPT
0	0	0	Data size for timer specified by MPT: 1 byte	No automatic addition	No ring control	
0	0	1			Ring control	
0	1	0		Automatic addition	No ring control	
0	1	1			Ring control	
1	0	0	Data size for timer specified by MPT: 2 bytes	No automatic addition	No ring control	
1	0	1			Ring control	
1	1	0		Automatic addition	No ring control	
1	1	1			Ring control	

VCIE	Interrupt Request when MSC = 0
0	Generated
1	Not generated (next interrupt processing is vectored interrupt)

(3) Macro service channel pointer

The macro service channel pointer specifies the macro service channel address. The macro service channel can be located in the 256-byte space from FE00H to FEFFH when the LOCATION 0H instruction is executed, or FFE00H to FFEFFH when the LOCATION 0FH instruction is executed, and the high-order 16 bits of the address are fixed. Therefore, the low-order 8 bits of the data stored to the highest address of the macro service channel are set in the macro service channel pointer.

22.8.6 Macro Service Type A

(1) Operation

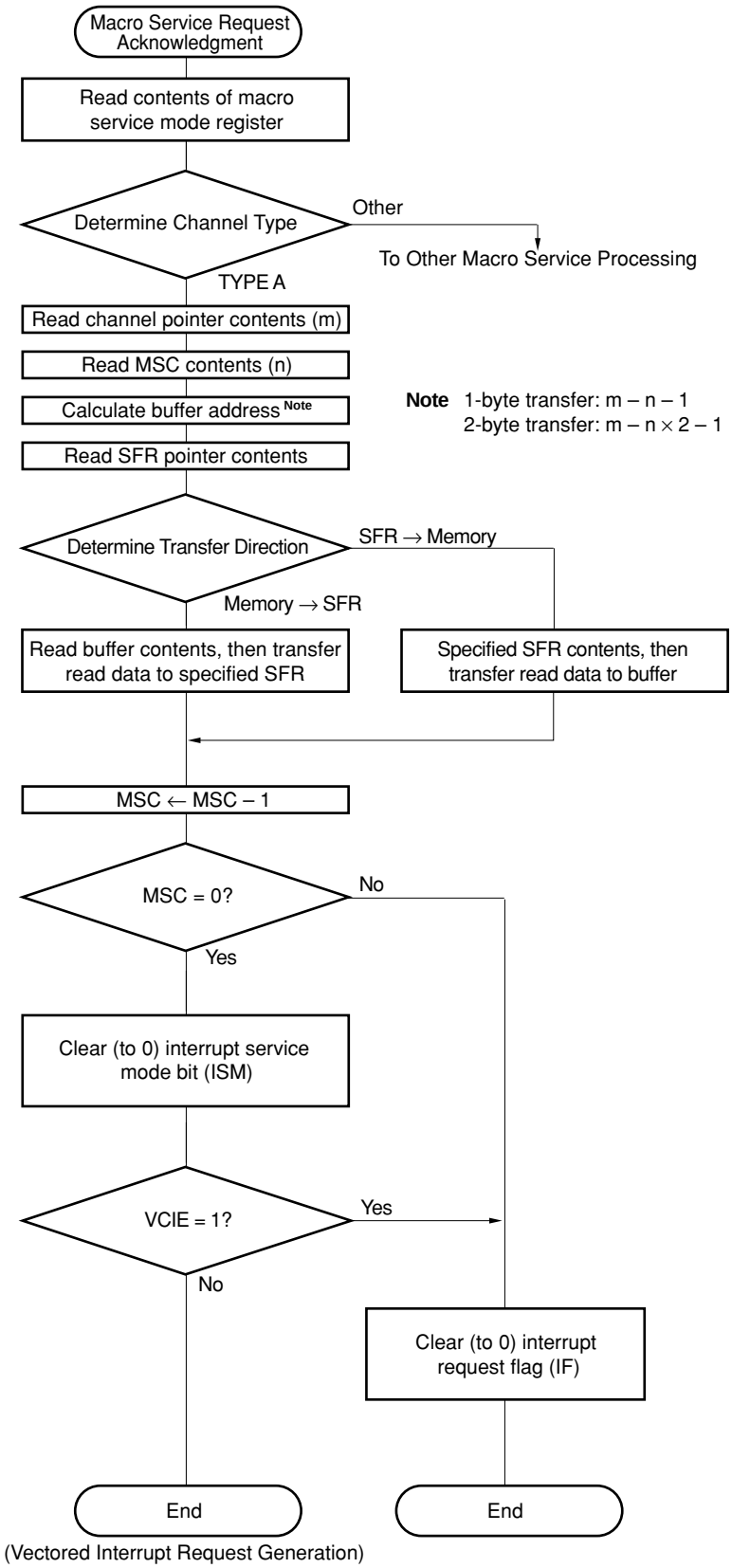
Data transfers are performed between buffer memory in the macro service channel and an SFR specified in the macro service channel.

With type A, the data transfer direction can be selected as memory-to-SFR or SFR-to-memory.

Data transfers are performed the number of times set beforehand in the macro service counter. One macro service processing transfers 8-bit or 16-bit data.

Type A macro service is useful when the amount of data to be transferred is small, as transfers can be performed at high speed.

Figure 22-22 Macro Service Data Transfer Processing Flow (Type A)



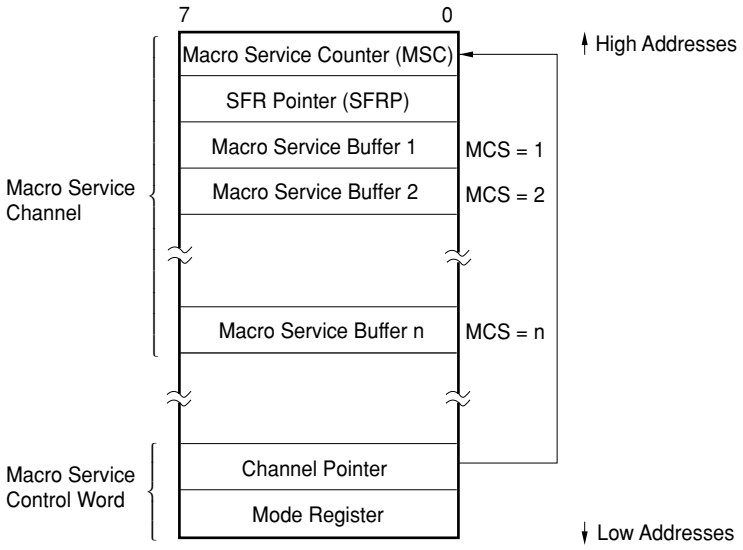
(2) Macro service channel configuration

The channel pointer and 8-bit macro service counter (MSC) indicate the buffer address in internal RAM (FE00H to FEFFH when the LOCATION 0H instruction is executed, or FFE00H to FFEFFH when the LOCATION 0FH instruction is executed) which is the transfer source or transfer destination (see **Figure 22-23**). In the channel pointer, the low-order 8 bits of the address are written to the macro service counter in the macro service channel.

The SFR involved with the access is specified by the SFR pointer (SFRP). The low-order 8 bits of the SFR address are written to the SFRP.

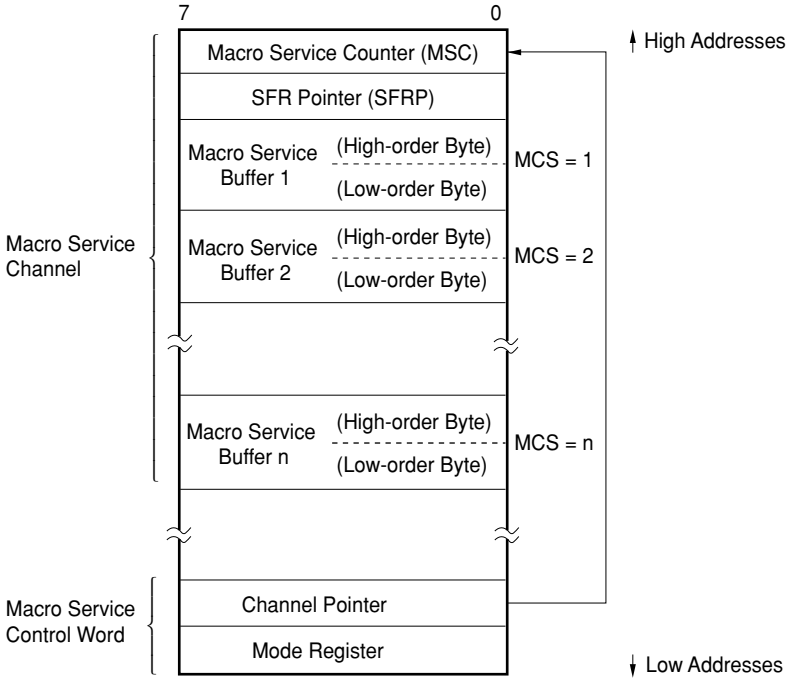
Figure 22-23 Type A Macro Service Channel

(a) 1-byte transfers



Macro service buffer address = (channel pointer) – (macro service counter) – 1

(b) 2-byte transfers

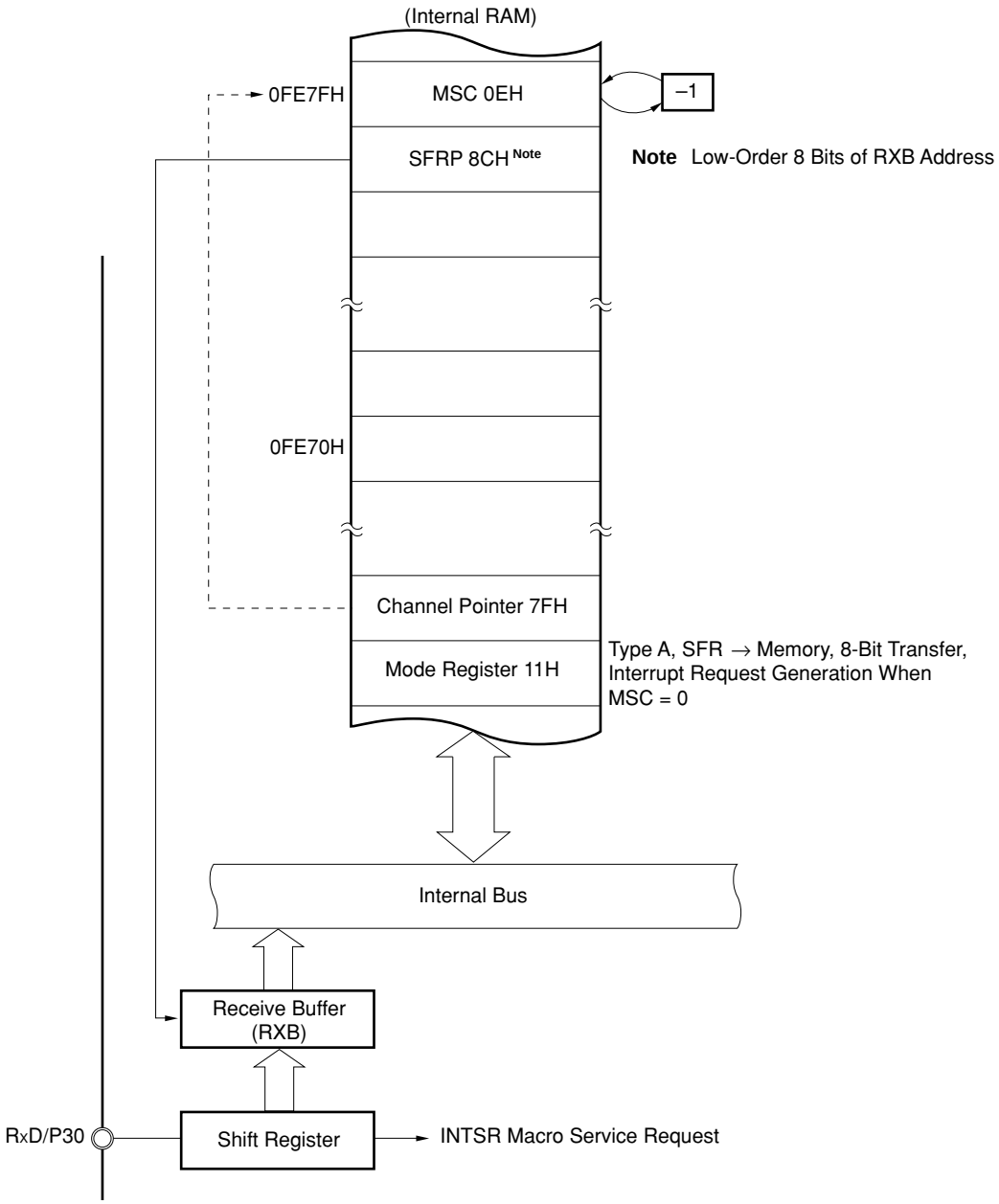


Macro service buffer address = (channel pointer) – (macro service counter) × 2 – 1

(3) Example of use of type A

An example is shown below in which data received via the asynchronous serial interface is transferred to a buffer area in on-chip RAM.

Figure 22-24 Asynchronous Serial Reception



Remark Addresses in the figure are the values when the LOCATION 0H instruction is executed.
When the LOCATION 0FH instruction is executed, 0F0000H should be added to the values in the figure.

22.8.7 Macro Service Type B

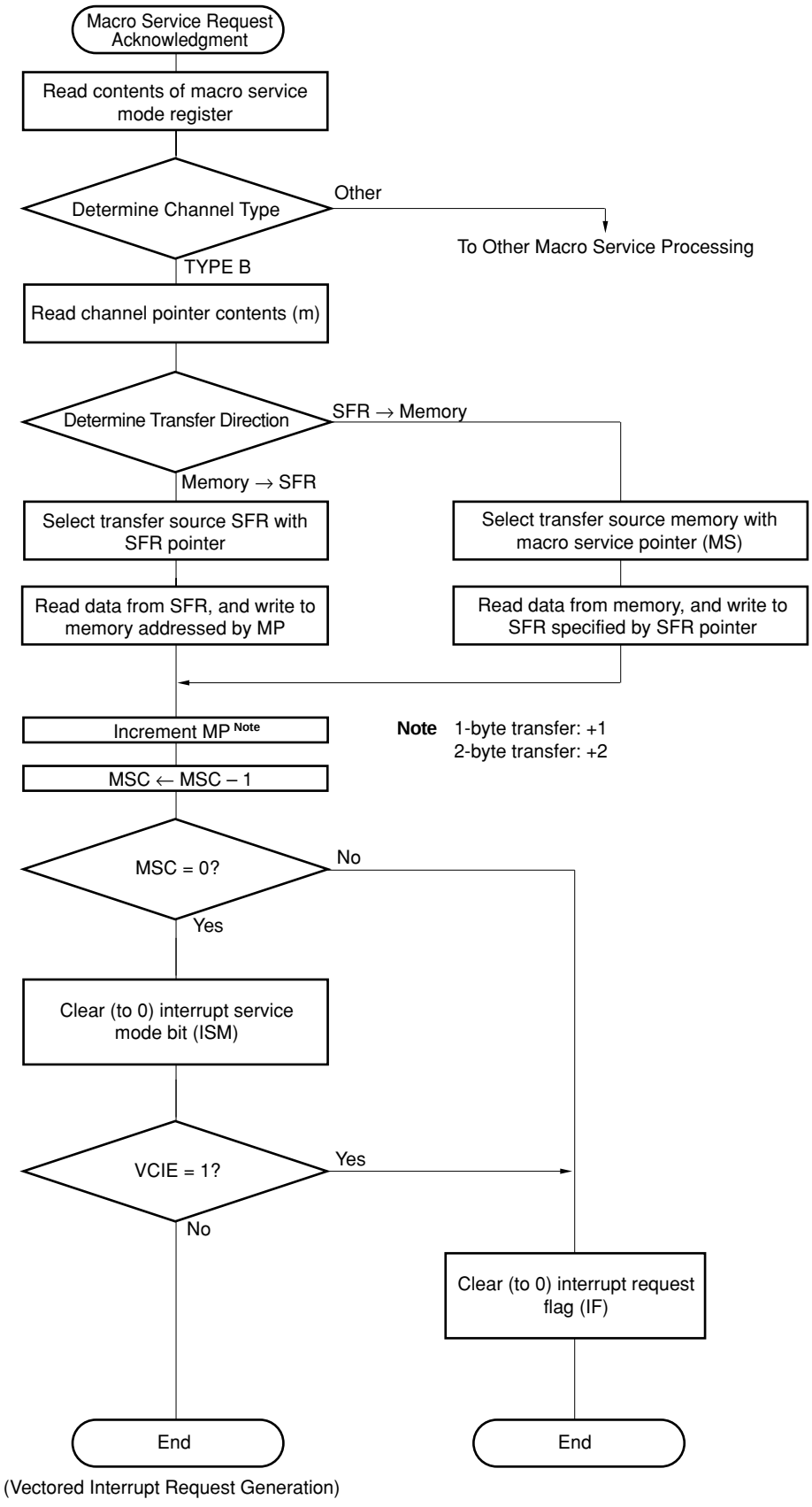
(1) Operation

Data transfers are performed between a data area in memory and an SFR specified by the macro service channel. With type B, the data transfer direction can be selected as memory-to-SFR or SFR-to-memory.

Data transfers are performed the number of times set beforehand in the macro service counter. One macro service processing transfers 8-bit or 16-bit data.

This type of macro service is macro service type A for general purposes and is ideal for processing a large amount of data because up to 64 Kbytes of data buffer area when 8-bit data is transferred or 1 Mbyte of data buffer area when 16-bit data is transferred can be set in any address space.

Figure 22-25 Macro Service Data Transfer Processing Flow (Type B)



(Vectored Interrupt Request Generation)

(2) Macro service channel configuration

The macro service pointer (MP) indicates the data buffer area in the 1-Mbyte memory space that is the transfer destination or transfer source.

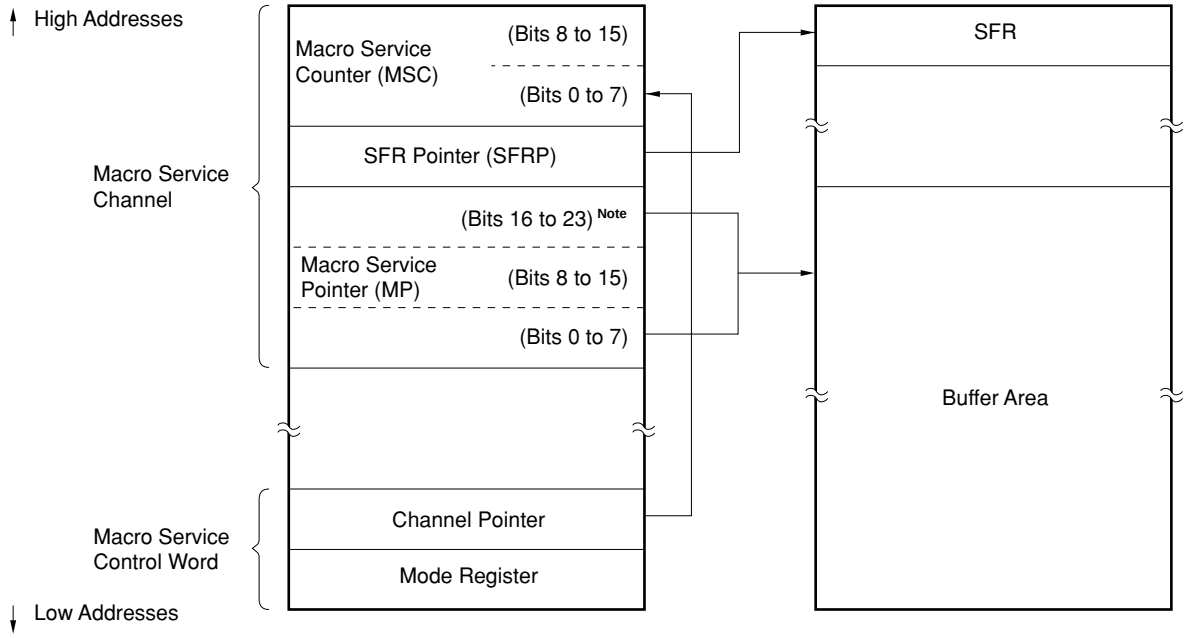
The low-order 8 bits of the SFR that is the transfer destination or transfer source is written to the SFR pointer (SFRP).

The macro service counter (MSC) is a 16-bit counter that specifies the number of data transfers.

The macro service channel that stores the MP, SFRP and MSC is located in internal RAM space addresses 0FE00H to 0FEFFH when the LOCATION 0H instruction is executed, or 0FFE00H to 0FFEFFH when the LOCATION 0FH instruction is executed.

The macro service channel is indicated by the channel pointer as shown in Figure 22-26. In the channel pointer, the low-order 8 bits of the address are written to the macro service counter in the macro service channel.

Figure 22-26 Type B Macro Service Channel



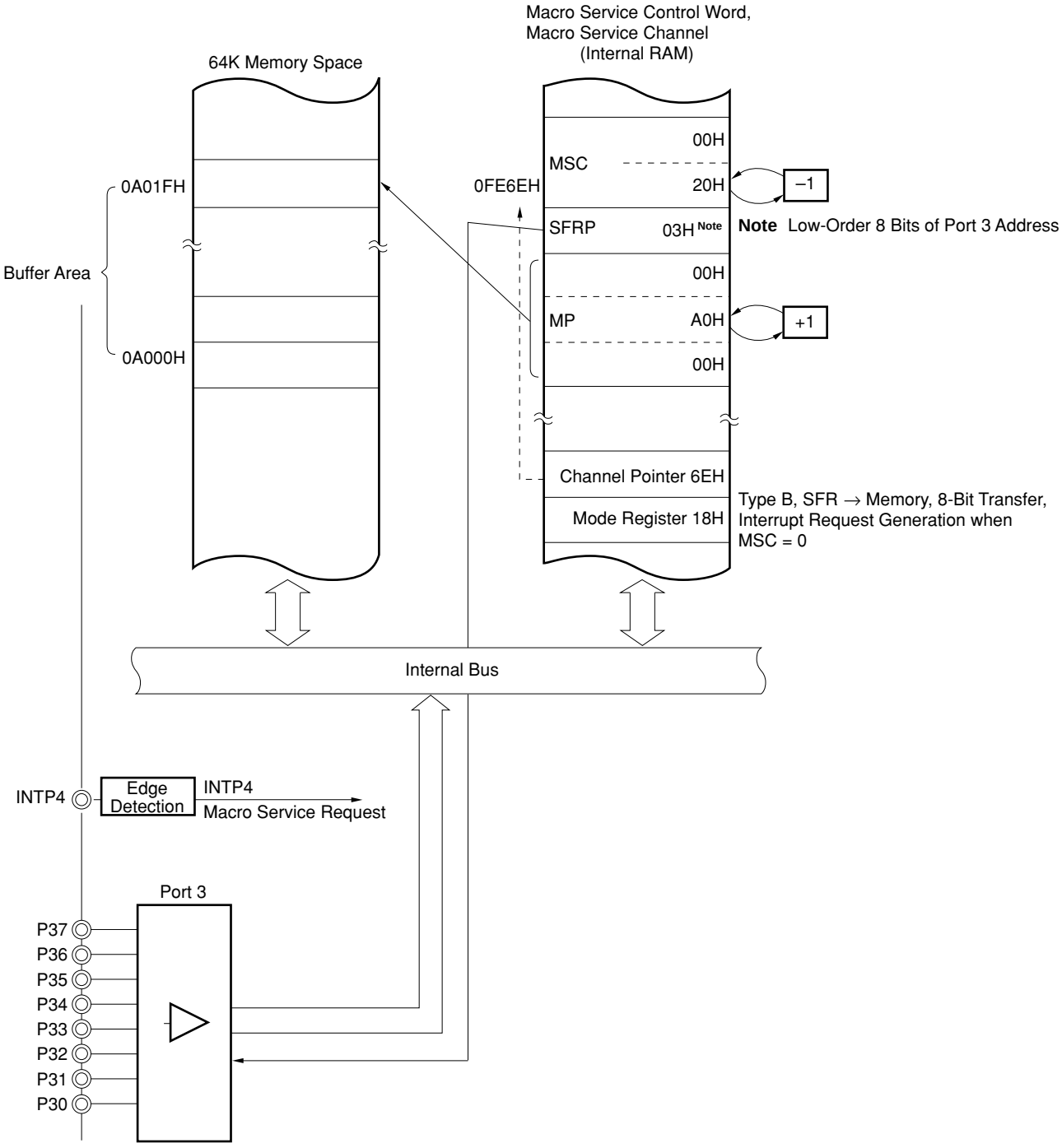
Macro service buffer address = macro service pointer

Note Bits 20 to 23 must be set to 0.

(3) Example of use of type B

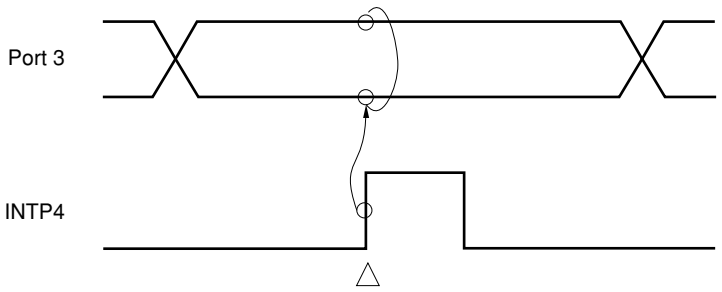
An example is shown below in which parallel data is input from port 3 in synchronization with an external signal. The INTP4 external interrupt pin is used for synchronization with the external signal.

Figure 22-27 Parallel Data Input Synchronized with External Interrupts



Remark Macro service channel addresses in the figure are the values when the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, 0F0000H should be added to the values in the figure.

Figure 22-28 Parallel Data Input Timing



22.8.8 Macro Service Type C

(1) Operation

In type C macro service, data in the memory specified by the macro service channel is transferred to two SFRs, for timer use and data use, specified by the macro service channel in response to a single interrupt request (the SFRs can be freely selected). An 8-bit or 16-bit timer SFR can be selected.

In addition to the basic data transfers described above, type C macro service, the following functions can be added to type C macro service to reduce the size of the buffer area and alleviate the burden on software.

These specifications are made by using the mode register of the macro service control word.

(a) Updating of timer macro service pointer

It is possible to choose whether the timer macro service pointer (MPT) is to be kept as it is or incremented/decremented. The MPT is incremented or decremented in the same direction as the macro service pointer (MPD) for data.

(b) Updating of data macro service pointer

It is possible to choose whether the data macro service pointer (MPD) is to be incremented or decremented.

(c) Automatic addition

The current compare register value is added to the data addressed by the timer macro service pointer (MPT), and the result is transferred to the compare register. If automatic addition is not specified, the data addressed by the MPT is simply transferred to the compare register.

(d) Ring control

An output data pattern of the length specified beforehand is automatically output repeatedly.

These specifications are made by the mode register in the macro service control word.

Figure 22-29 Macro Service Data Transfer Processing Flow (Type C) (1/2)

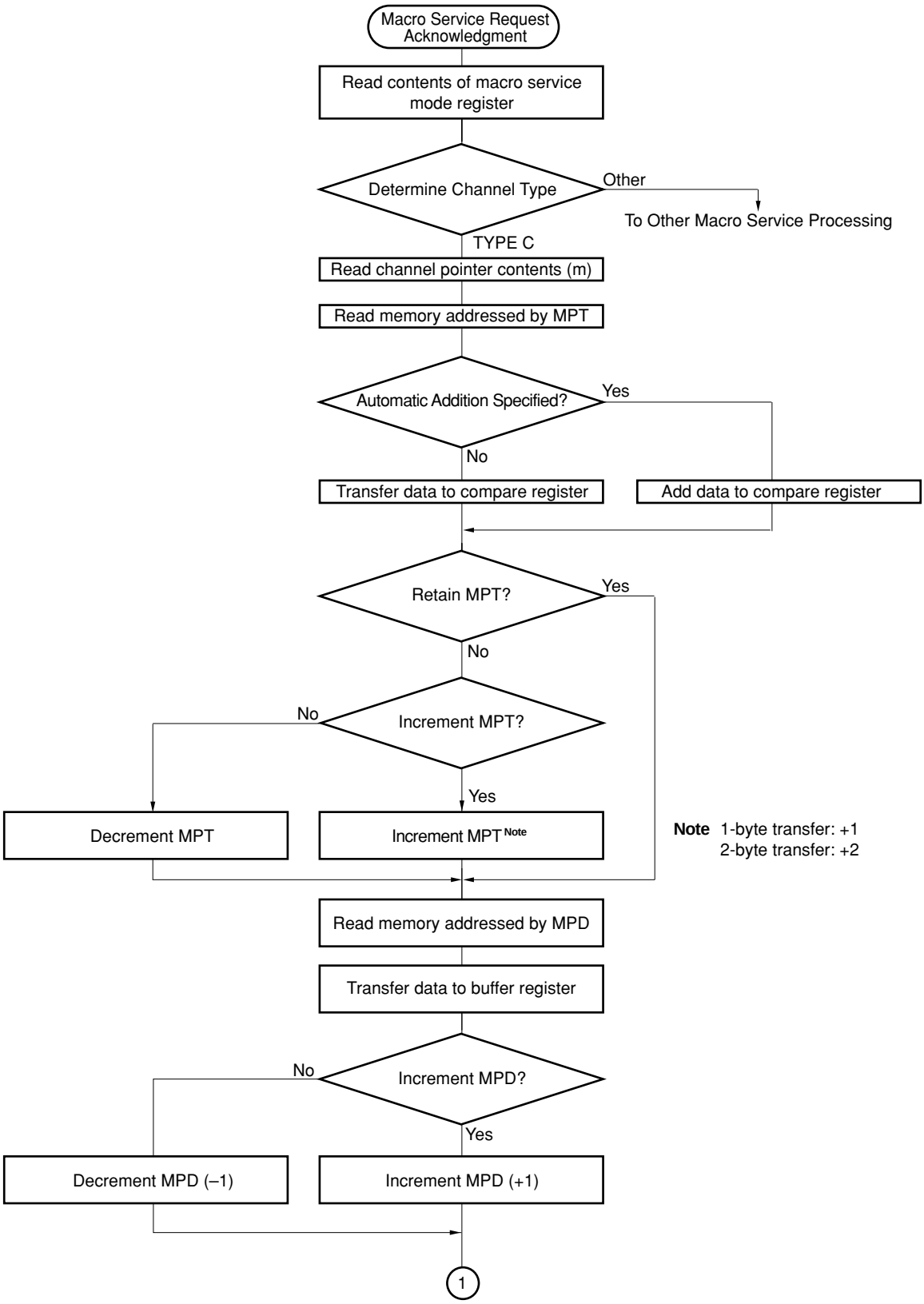
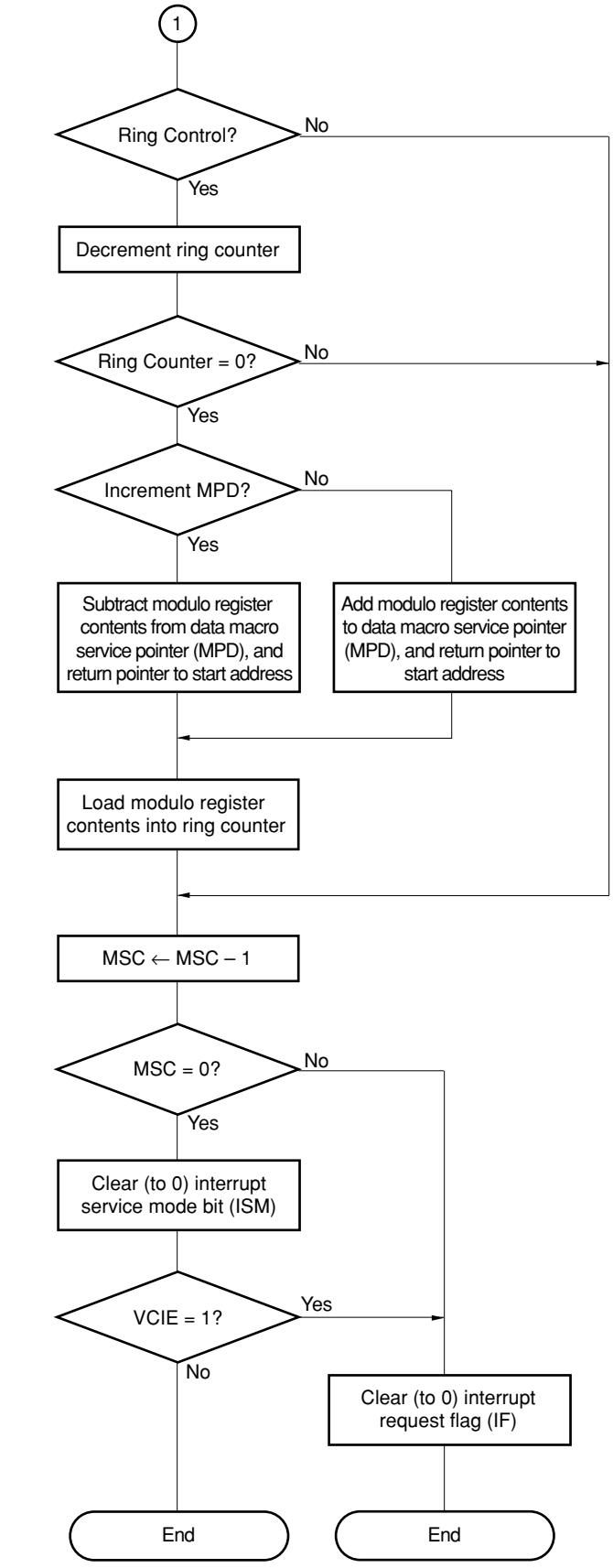


Figure 22-29 Macro Service Data Transfer Processing Flow (Type C) (2/2)



(Vectored Interrupt Request Generation)

(2) Macro service channel configuration

There are two kinds of type C macro service channel, as shown in Figure 22-30.

The timer macro service pointer (MPT) mainly indicates the data buffer area in the 1-Mbyte memory space to be transferred or added to the timer/counter compare register.

The data macro service pointer (MPD) indicates the data buffer area in the 1-Mbyte memory space to be transferred to the real-time output port.

The modulo register (MR) specifies the number of repeat patterns when ring control is used.

The ring counter (RC) holds the step in the pattern when ring control is used. When initialization is performed, the same value as in the MR is normally set in this counter.

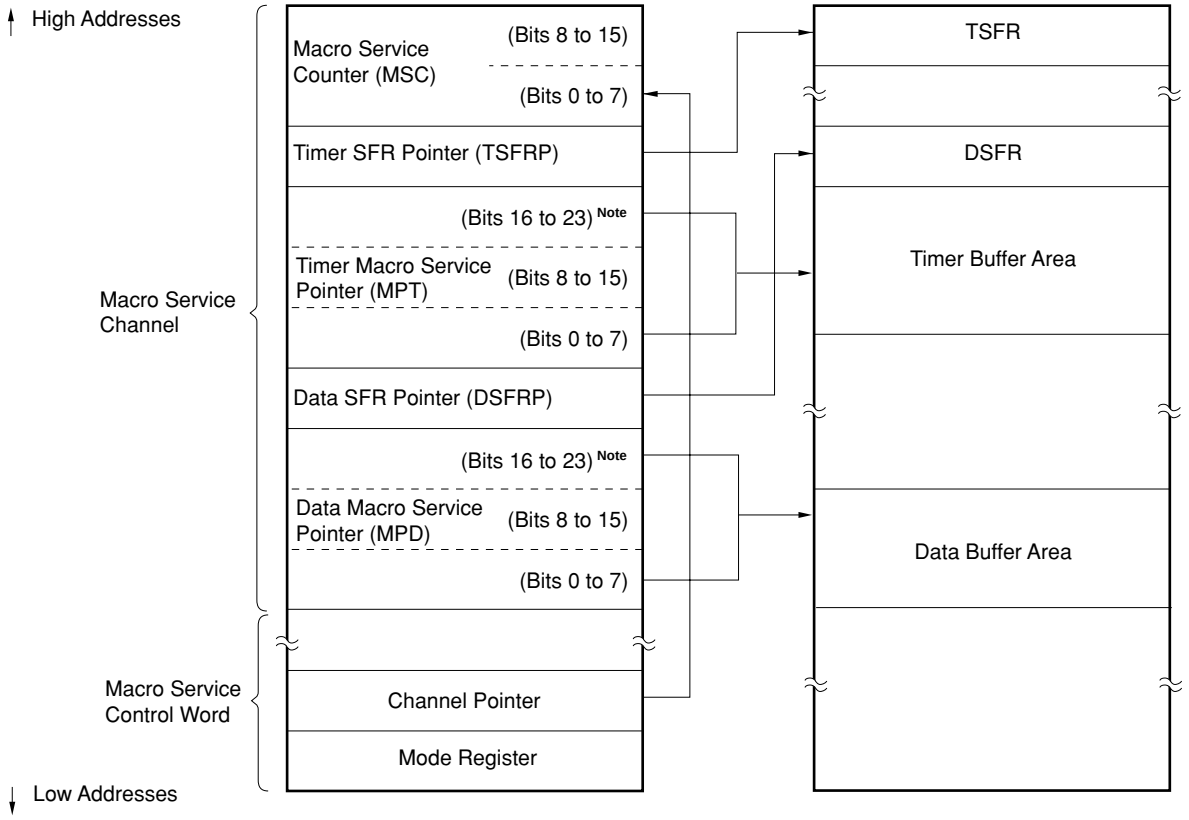
The macro service counter (MSC) is a 16-bit counter that specifies the number of data transfers.

The low-order 8 bits of the SFR that is the transfer destination is written to the timer SFR pointer (TSFRP) and data SFR pointer (DSFRP).

The macro service channel that stores these pointers and counters is located in internal RAM space addresses 0FE00H to 0FEFFH when the LOCATION 0H instruction is executed, or 0FFE00H to 0FFEFFH when the LOCATION 0FH instruction is executed. The macro service channel is indicated by the channel pointer as shown in Figure 22-30. In the channel pointer, the low-order 8 bits of the address are written to the macro service counter in the macro service channel.

Figure 22-30 Type C Macro Service Channel (1/2)

(a) No ring control

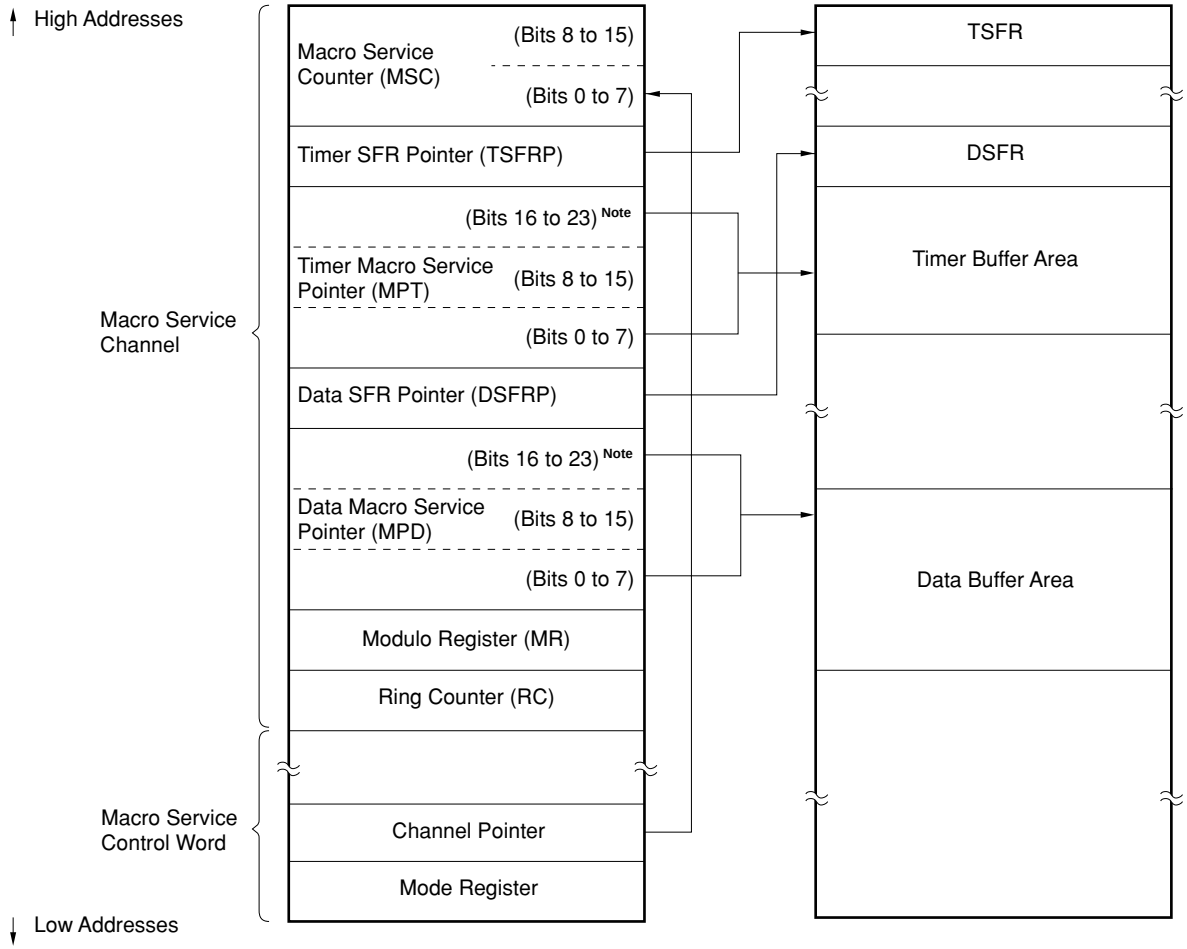


Macro service buffer address = macro service pointer

Note Bits 20 to 23 must be set to 0.

Figure 22-30 Type C Macro Service Channel (2/2)

(b) With ring control



Macro service buffer address = macro service pointer

Note Bits 20 to 23 must be set to 0.

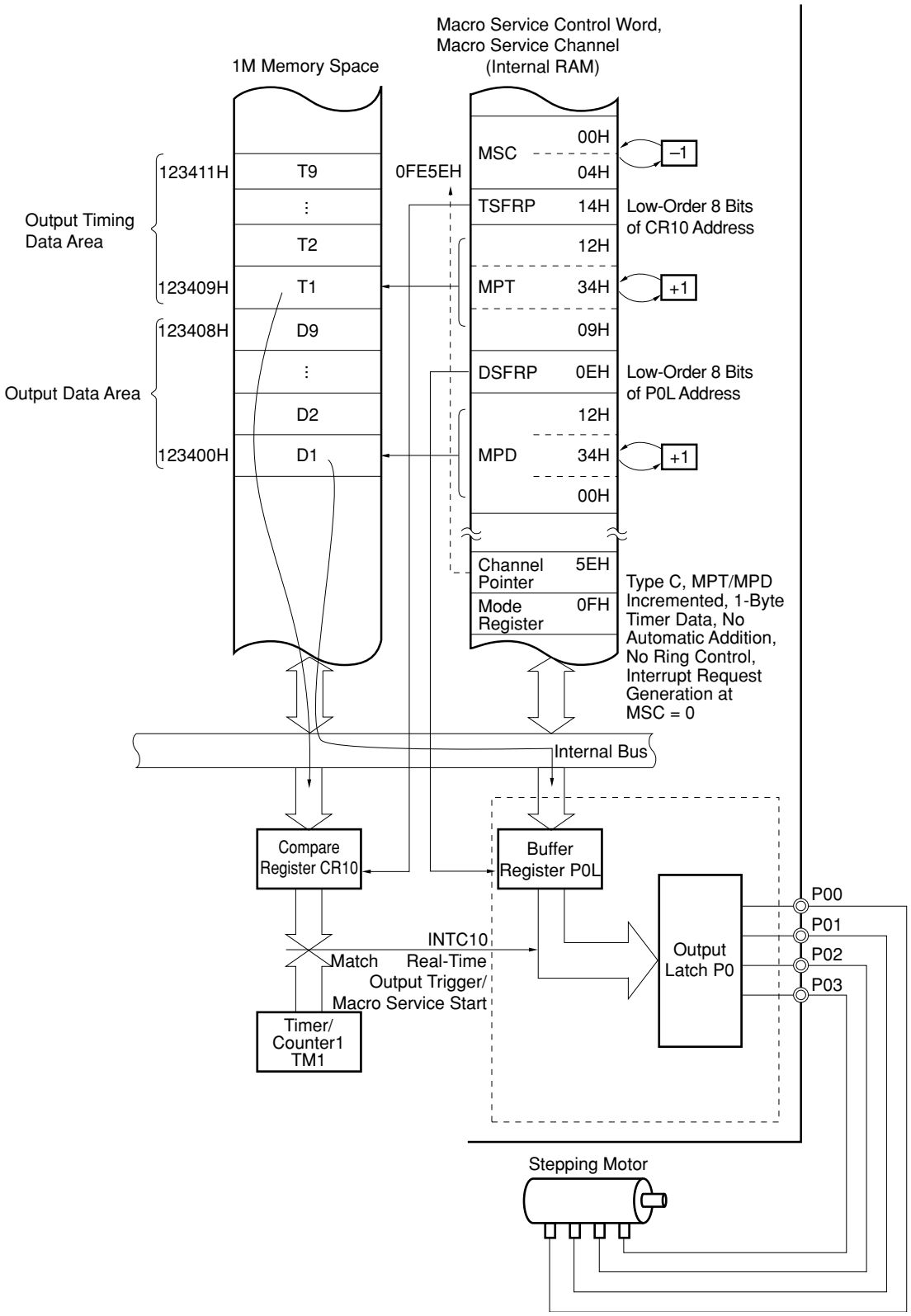
(3) Examples of use of type C

(a) Basic operation

An example is shown below in which the output pattern to the real-time output port and the output interval are directly controlled.

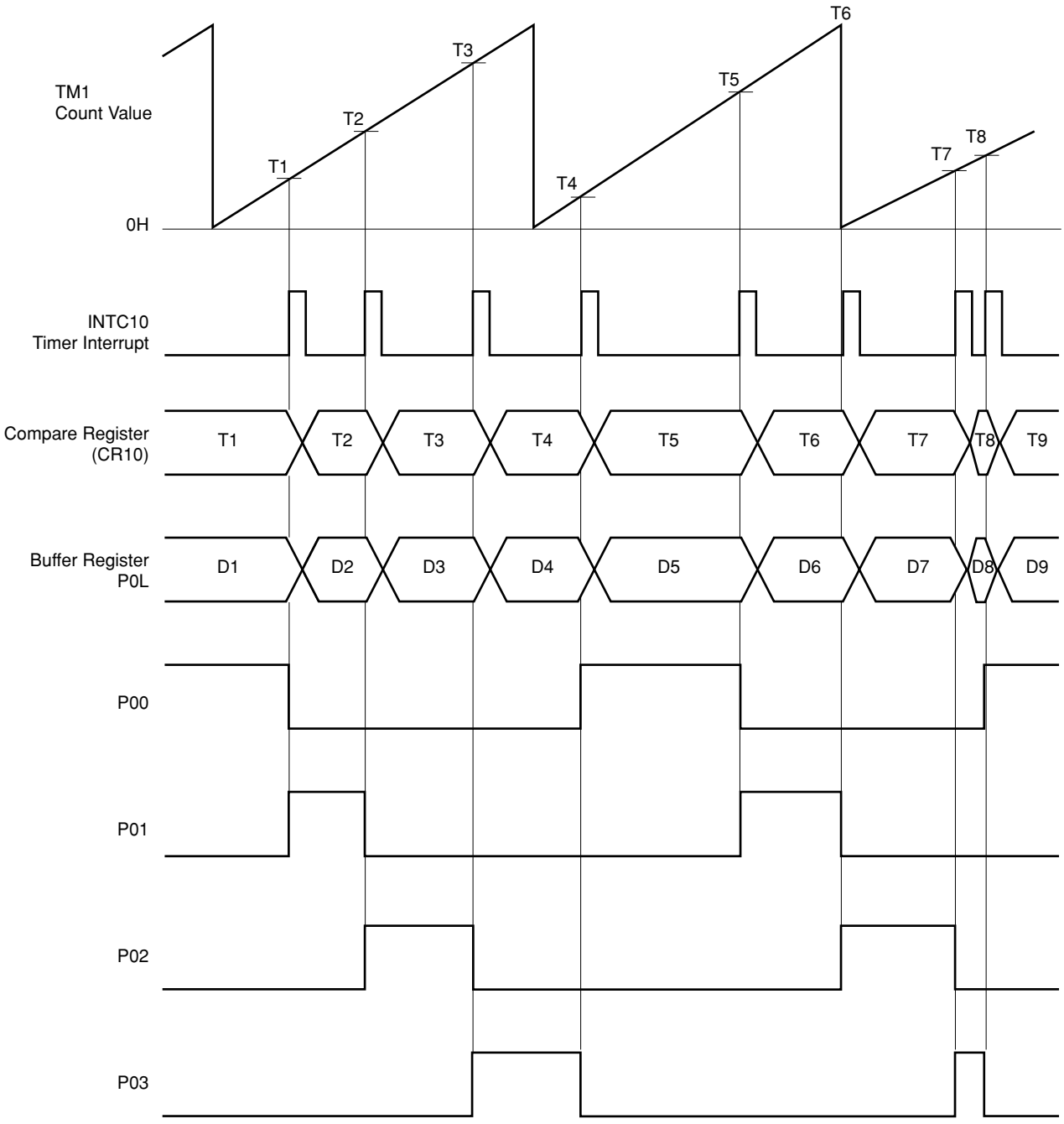
Update data is transferred from the two data storage areas set in the 1-Mbyte space beforehand to the real-time output function buffer register (P0L) and the compare register (CR10).

Figure 22-31 Stepping Motor Open Loop Control by Real-Time Output Port



Remark Internal RAM addresses in the figure are the values when the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, 0F0000H should be added to the values in the figure.

Figure 22-32 Data Transfer Control Timing



(b) Examples of use of automatic addition control and ring control

(i) Automatic addition control

The output timing data (Δt) specified by the macro service pointer (MPT) is added to the contents of the compare register, and the result is written back to the compare register.

Use of this automatic addition control eliminates the need to calculate the compare register setting value in the program each time.

(ii) Ring control

With ring control, the predetermined output patterns is prepared for one cycle only, and the one-cycle data patterns are output repeatedly in order in ring form.

When ring control is used, only the output patterns for one cycle need be prepared, allowing the size of the data ROM area to be reduced.

The macro service counter (MSC) is decremented each time a data transfer is performed.

With ring control, too, an interrupt request is generated when MSC = 0.

When controlling a stepping motor, for example, the output patterns will vary depending on the configuration of the stepping motor concerned, and the phase excitation method (single-phase excitation, two-phase excitation, etc.), but repeat patterns are used in all cases. Examples of single-phase excitation and 1-2-phase excitation of a 4-phase stepping motor are shown in Figures 22-33 and 22-34.

Figure 22-33 Single-Phase Excitation of 4-Phase Stepping Motor

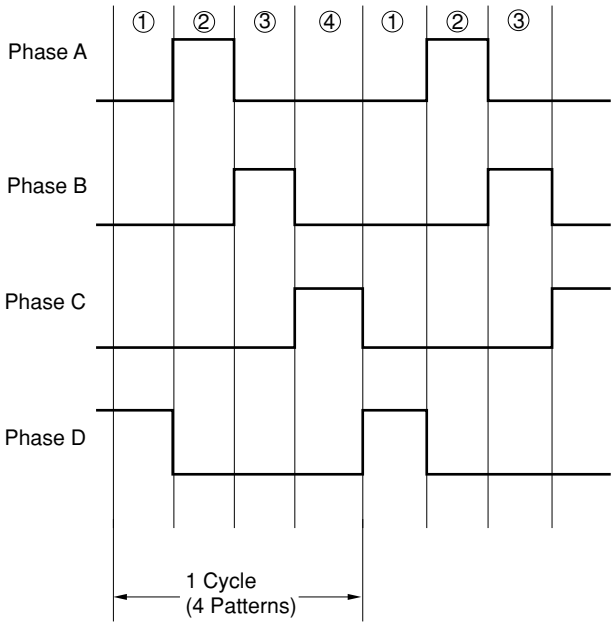


Figure 22-34 1-2-Phase Excitation of 4-Phase Stepping Motor

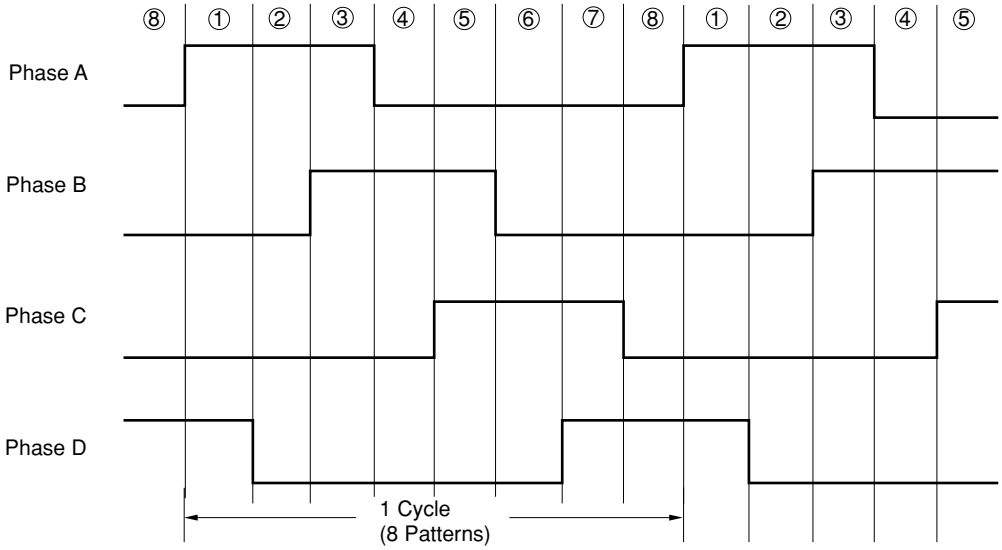
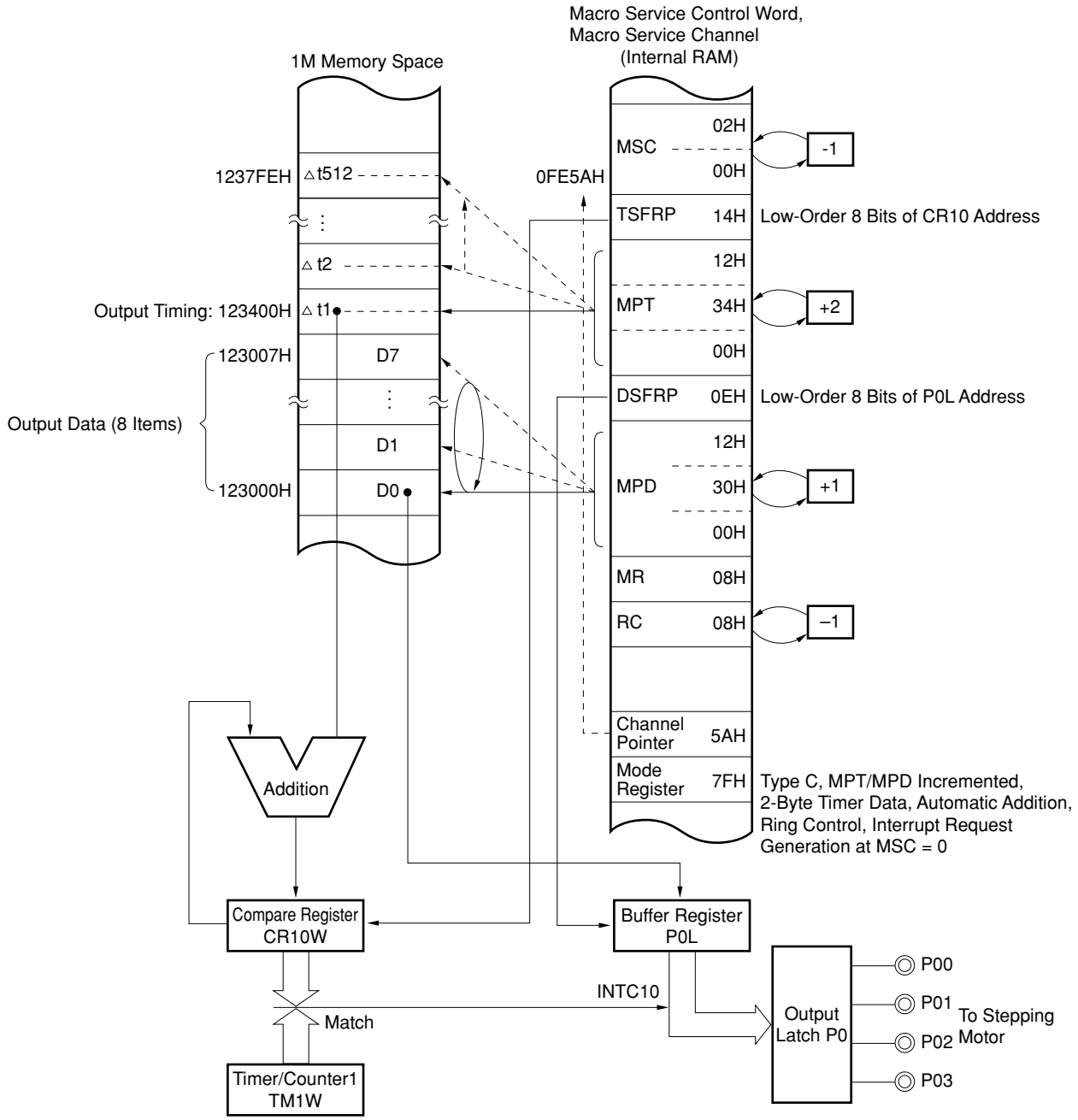


Figure 22-35 Automatic Addition Control + Ring Control Block Diagram 1 (When Output Timing Varies with 1-2-Phase Excitation)



Remark Internal RAM addresses in the figure are the values when the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, 0F0000H should be added to the values in the figure.

Figure 22-36 Automatic Addition Control + Ring Control Timing Diagram 1 (When Output Timing Varies with 1-2-Phase Excitation)

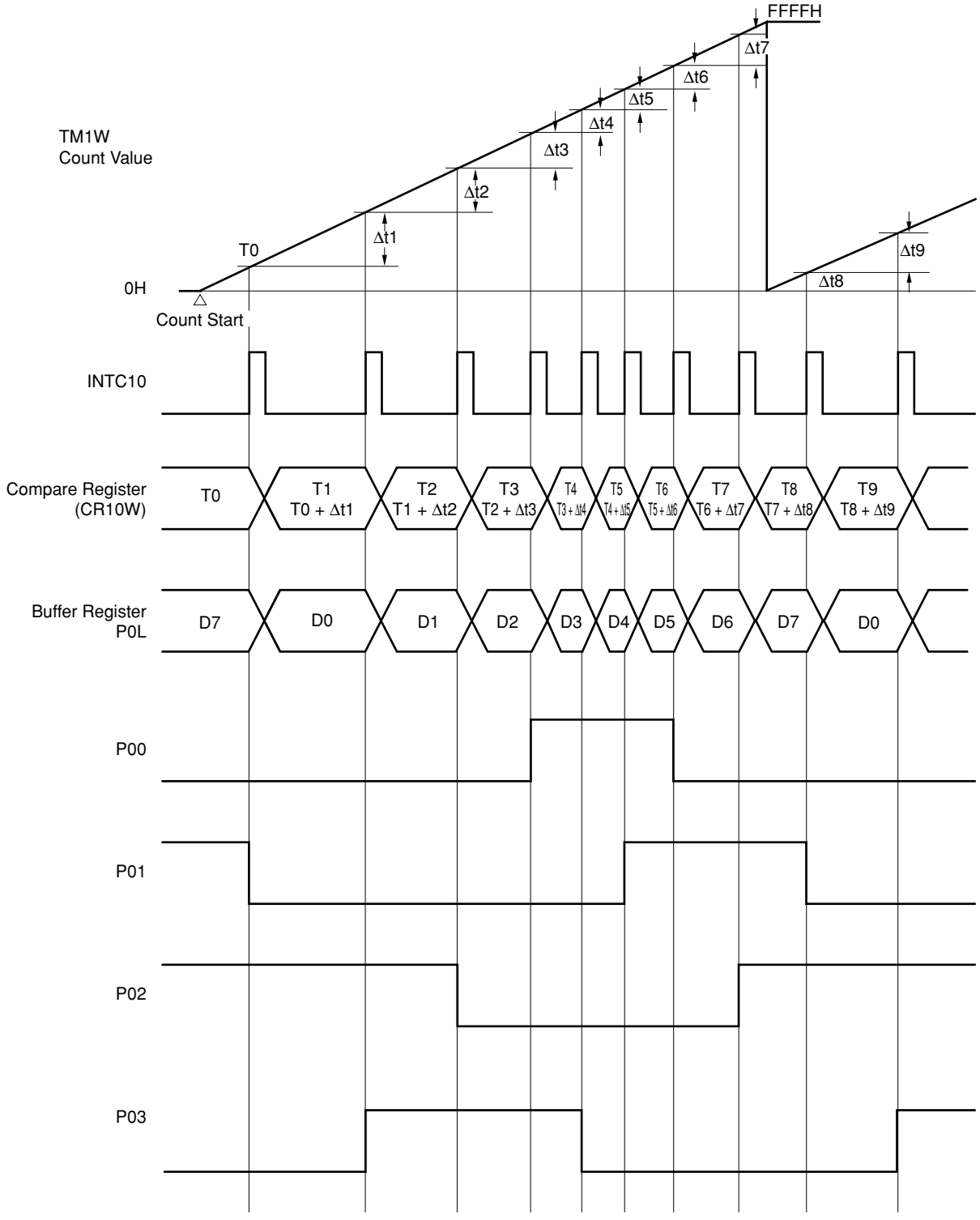
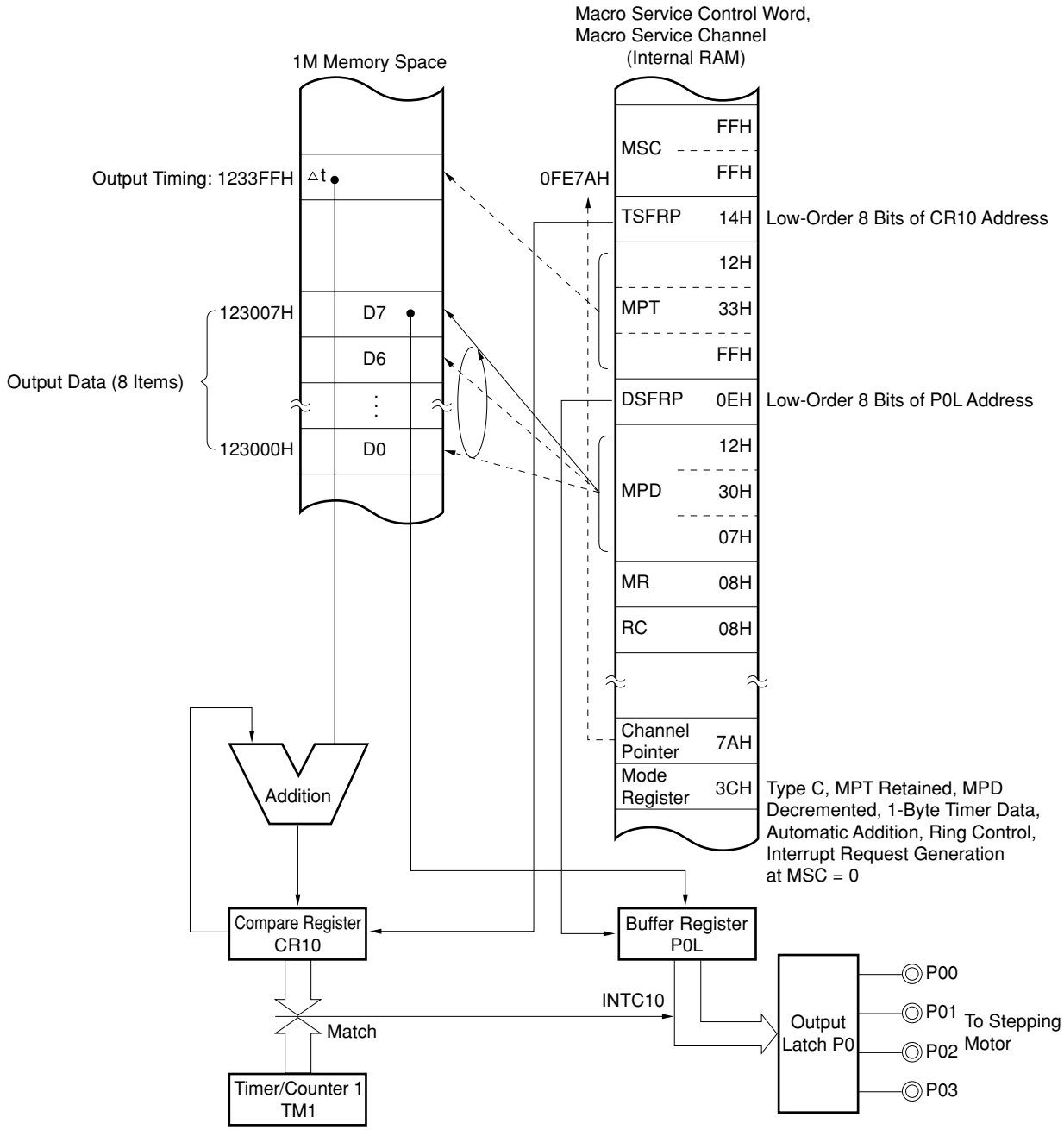
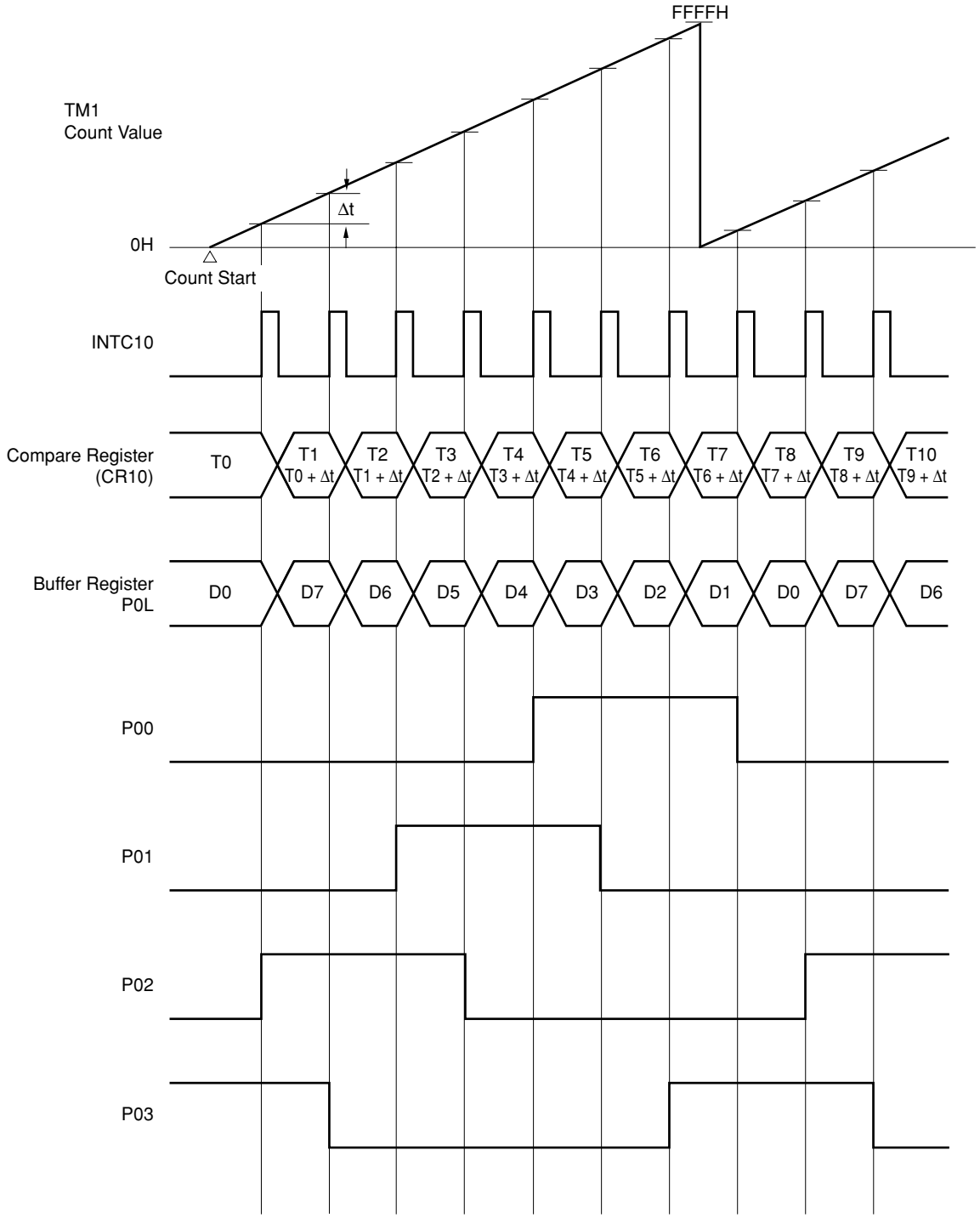


Figure 22-37 Automatic Addition Control + Ring Control Block Diagram 2 (1-2-Phase Excitation Constant-Velocity Operation)



Remark Internal RAM addresses in the figure are the values when the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, 0F0000H should be added to the values in the figure.

Figure 22-38 Automatic Addition Control + Ring Control Timing Diagram 2 (1-2-Phase Excitation Constant-Velocity Operation)



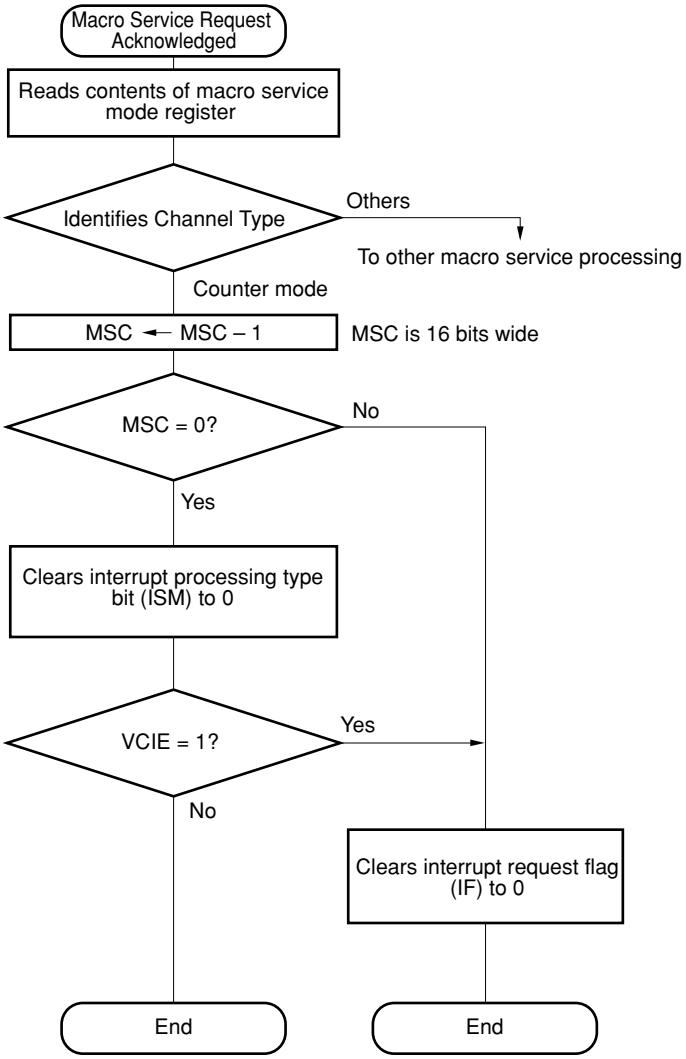
22.8.9 Counter Mode

(1) Operation

MSC is decremented the number of times set in advance to the macro service counter (MSC).

Because the number of times an interrupt occurs can be counted, this function can be used as an event counter where the interrupt generation cycle is long.

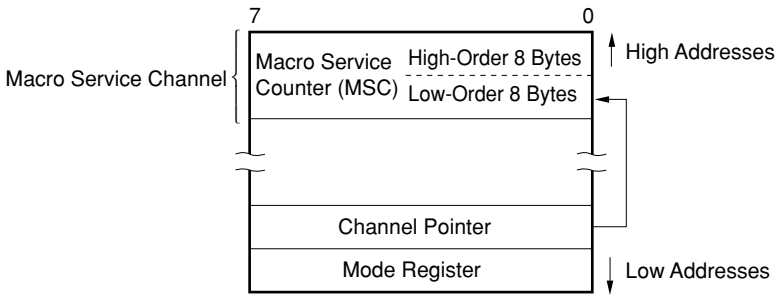
Figure 22-39 Macro Service Data Transfer Processing Flow (Counter Mode)



(2) Configuration of macro service channel

The macro service channel consists of only a 16-bit macro service counter. The low-order 8 bits of the address of the MSC are written to the channel pointer.

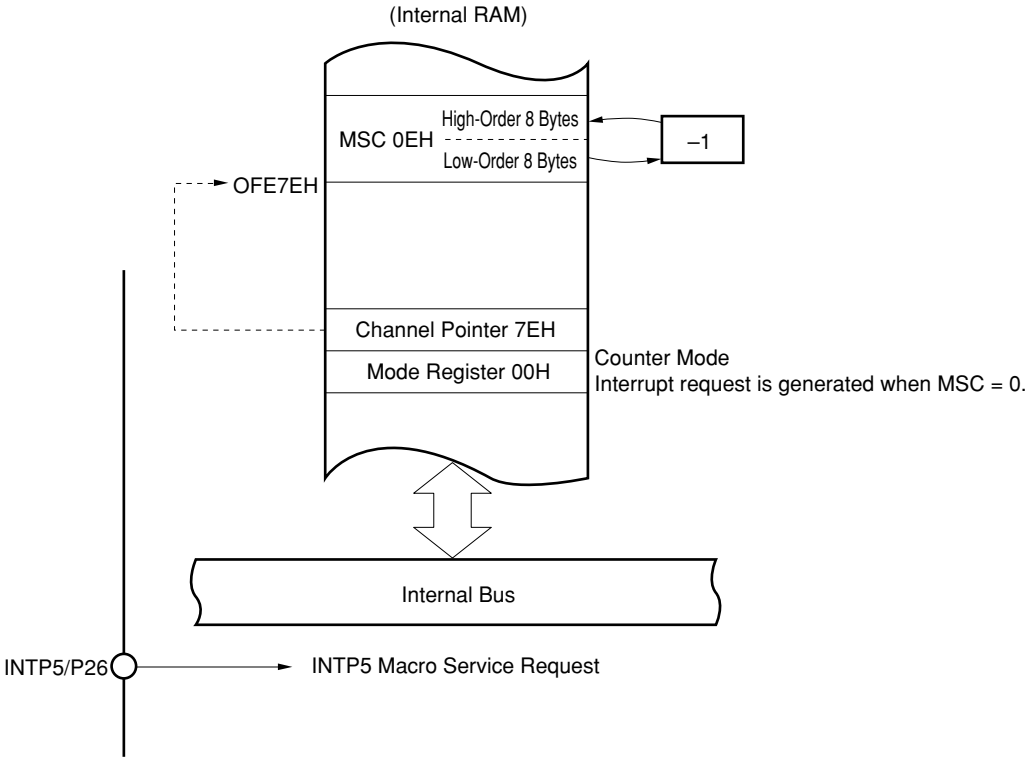
Figure 22-40 Counter Mode



(3) Example of using counter mode

Here is an example of counting the number of edges input to external interrupt pin INTP5.

Figure 22-41 Counting Number of Edges



Remark The internal RAM address in the figure above is the value when the LOCATION 0H instruction is executed. When the LOCATION 0FH instruction is executed, add 0F0000H to this value.

22.9 WHEN INTERRUPT REQUESTS AND MACRO SERVICE ARE TEMPORARILY HELD PENDING

When the following instructions are executed, interrupt acknowledgment and macro service processing is deferred for 8 system clock cycles. However, software interrupts are not deferred.

EI
DI
BRK
BRKCS
RETCS
RETCSB !addr16
RETI
RETB
LOCATION 0H or LOCATION 0FH
POP PSW
POPU post
MOV PSWL, A
MOV PSWL, #byte
MOVG SP, #imm24
Write instruction and bit manipulation instruction to an interrupt control register **Note**, or the MK0, MK1L, IMC or ISPR register (except BT and BF instructions)
PSW bit manipulation instruction
(Excluding the BT PSWL. bit, $addr20, BF PSWL. bit, $addr20, BT PSWH. bit, $addr20, BF PSWH. bit, $addr20, SET1 CY, NOT1 CY, and CLR1 CY instructions)

Note Interrupt control registers: PIC0, PIC1, PIC2, PIC3, PIC4, PIC5, CIC00, CIC01, CIC10, CIC11, CIC20, CIC21, CIC30, ADIC, SERIC, SRIC, CSIIC1, STIC, CSIIC, SERIC2, SRIC2, CSIIC2, STIC2, SPCIC (μPD784038Y Subseries only)

Cautions 1. When an interrupt related register is polled using a BF instruction, etc., the branch destination of that BR instruction, etc., should not be that instruction. If a program is written in which a branch is made to that instruction itself, all interrupts and macro service requests will be held pending until a condition whereby a branch is not made by that instruction arises.

Bad Example

```
        ⋮
LOOP :  BF   PIC0.7, $LOOP      All interrupts and macro service requests are held pending until
                                PIC0.7 is 1.
        ×××                     ← Interrupts and macro service requests are not serviced until
                                after execution of the instruction following the BF instruction.
```

Good Example (1)

```
        ⋮
LOOP :  NOP
        BF   PIC0.7, $LOOP      ← Interrupts and macro service requests are serviced after execu
        ⋮                       tion of the NOP instruction, so that interrupts are never held
                                pending for a long period.
```

Good Example (2)

```
        ⋮
LOOP :  BT   PIC0.7, $NEXT      Using a BTCLR instruction instead of a BT instruction has the
                                advantage that the flag is cleared (to 0) automatically.
        BR   $LOOP              ← Interrupts and macro service requests are serviced after execu-
NEXT :  ⋮                       tion of the BR instruction, so that interrupts are never held
                                pending for a long period.
```

2. For a similar reason, if problems are caused by a long pending period for interrupts and macro service when instructions to which the above applies are used in succession, a time at which interrupts and macro service requests can be acknowledged should be provided by inserting an NOP instruction, etc., in the series of instructions.

22.10 INSTRUCTIONS WHOSE EXECUTION IS TEMPORARILY SUSPENDED BY AN INTERRUPT OR MACRO SERVICE

Execution of the following instructions is temporarily suspended by an acknowledgeable interrupt request or macro service request, and the interrupt or macro service request is acknowledged. The suspended instruction is resumed after completion of the interrupt service program or macro service processing.

Temporarily suspended instructions:

MOVM, XCHM, MOVBK, XCHBK
CMPME, CMPMNE, CMPMC, CMPMNC
CMPBKE, CMPBKNE, CMPBKC, CMPBKNC
SACW

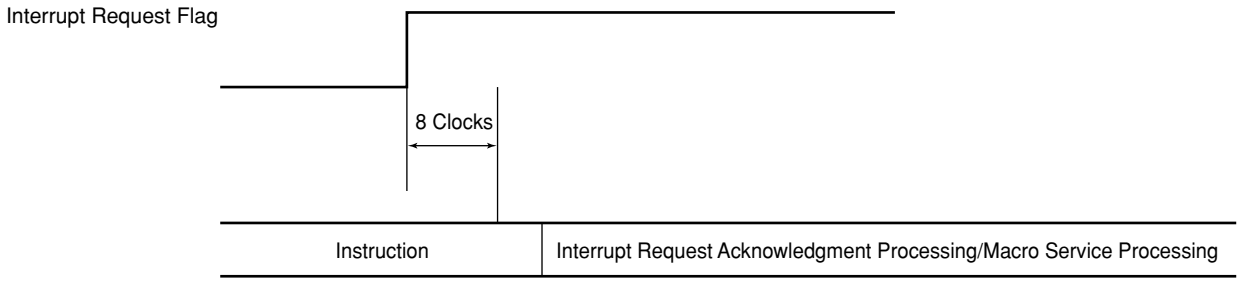
22.11 INTERRUPT AND MACRO SERVICE OPERATION TIMING

Interrupt requests are generated by hardware. The generated interrupt request sets (to 1) an interrupt request flag.

When the interrupt request flag is set (to 1), a time of 8 clocks (0.5 μs: f_{CLK} = 16 MHz) is taken to determine the priority, etc.

Following this, if acknowledgment of that interrupt or macro service is enabled, interrupt request acknowledgment processing is performed when the instruction being executed ends. If the instruction being executed is one which temporarily defers interrupts and macro service, the interrupt request is acknowledged after the following instruction (see **22.9 WHEN INTERRUPT REQUESTS AND MACRO SERVICE ARE TEMPORARILY HELD PENDING** for deferred instructions).

Figure 22-42 Interrupt Request Generation and Acknowledgment (Unit: Clock = 1/f_{CLK})



22.11.1 Interrupt Acknowledge Processing Time

The time shown in Table 22-7 is required to acknowledge an interrupt request. After the time shown in this table has elapsed, execution of the interrupt processing program is started.

Table 22-7 Interrupt Acknowledge Processing Time

(Unit: Clock = 1/f_{CLK})

Vector Table	IROM						EMEM					
Branch Destination	IROM, PRAM			EMEM			PRAM			EMEM		
Stack	IRAM	PRAM	EMEM	IRAM	PRAM	EMEM	IRAM	PRAM	EMEM	IRAM	PRAM	EMEM
Vectored Interrupts	26	29	37 + 4n	27	30	38 + 4n	30	33	41 + 4n	31	34	42 + 4n
Context Switching	22	–	–	23	–	–	22	–	–	23	–	–

Remarks 1. IROM : internal ROM (with high-speed fetch specified)
PRAM : peripheral RAM of internal RAM (only when LOCATION 0H instruction is executed in the case of branch destination)
IRAM : internal high-speed RAM
EMEM : internal ROM when external memory and high-speed fetch are not specified

2. n is the number of wait states per byte necessary for writing data to the stack (the number of wait states is the sum of the number of address wait states and the number of access wait states).

3. It the vector table is EMEM, and if wait states are inserted in reading the vector table, add 2 m to the value of the vectored interrupt in the above table, and add m to the value of context switching, where m is the number of wait states per byte necessary for reading the vector table.

4. It the branch destination is EMEM and if wait states are inserted in reading the instruction at the branch destination, add that number of wait states.

5. If the stack is occupied by PRAM and if the value of the stack pointer (SP) is odd, add 4 to the value in the above table.

6. The number of wait states is the sum of the number of address wait states and the number of access wait states.

22.11.2 Processing Time of Macro Service

Macro service processing time differs depending on the type of the macro service, as shown in Table 22-8.

Table 22-8 Macro Service Processing Time

(Units: Clock = 1/f_{CLK})

Processing Type of Macro Service			Data Area	
			IRAM	Others
Type A	SFR → memory	1 byte	24	–
		2 bytes	25	–
	Memory → SFR	1 byte	24	–
		2 bytes	26	–
Type B	SFR → memory		33	35
	Memory → SFR		34	36
Type C			49	53
Counter mode	MSC ≠ 0		17	–
	MSC = 0		25	–

Remarks 1. IRAM: internal high-speed RAM

2. In the following cases in the other data areas, add the number of clocks specified below.

- If the data size is 2 bytes with IROM or IRAM, and the data is located at an odd address: 4 clocks
- If the data size is 1 byte with EMEM: number of wait states for data access
- If the data size is 2 bytes with EMEM: 4 + 2n (where n is the number of wait states per byte)

3. If MSC = 0 with type A, B, or C, add 1 clock.

4. With type C, add the following value depending on the function to be used and the status at that time.

- Ring control: 4 clocks. Adds 7 more clocks if the ring counter is 0 during ring control.

22.12 RESTORING INTERRUPT FUNCTION TO INITIAL STATE

If an inadvertent program loop or system error is detected by means of an operand error interrupt, the watchdog timer, NMI pin input, etc., the entire system must be restored to its initial state. In the μPD784038, interrupt acknowledgment related priority control is performed by hardware. This interrupt acknowledgment related hardware must also be restored to its initial state, otherwise subsequent interrupt acknowledgment control may not be performed normally.

A method of initializing interrupt acknowledgment related hardware in the program is shown below. The only way of performing initialization by hardware is by $\overline{\text{RESET}}$ input.

Example

```
        MOVW    MK0, #0FFFFH    ;  Mask all maskable interrupts
        MOV     MK1L, #0FFH
IRESL   :
        CMP     ISPR, #0        ;  No interrupt service programs running?
        BZ      $NEXT
        MOVG    SP, #RETVAL     ;  Forcibly change SP location
        RETI                    ;  Forcibly terminate running interrupt service program, return
                                   address = IRESL
RETVAL  :
        DW      LOWW (IRESL)    ;  Stack data to return to IRESL with RETI instruction
        DB      0
        DB      HIGHW (IRESL)   ;  LOWW & HIGHW are assembler operators for calculating low-order
                                   16 bits & high-order 16 bits respectively of symbol NEXT
NEXT    :
```

- It is necessary to ensure that a non-maskable interrupt request is not generated via the NMI pin during execution of this program.
- After this, on-chip peripheral hardware initialization and interrupt control register initialization are performed.
- When interrupt control register initialization is performed, the interrupt request flags must be cleared (to 0).

22.13 CAUTIONS

(1) The in-service priority register (ISPR) is read-only. Writing to this register may result in malfunction.

(2) The watchdog timer mode register (WDM) can only be written to with a dedicated instruction (MOV WDM/#byte).

(3) The RETI instruction must not be used to return from a software interrupt caused by a BRK instruction. Use the RETB instruction.

(4) The RETCS instruction must not be used to return from a software interrupt caused by a BRKCS instruction. Use the RETCSB instruction.

(5) When a maskable interrupt is acknowledged by vectored interruption, the RETI instruction must be used to return from the interrupt. Subsequent interrupt related operations will not be performed normally if a different instruction is used.

(6) The RETCS instruction must be used to return from a context switching interrupt. Subsequent interrupt related operations will not be performed normally if a different instruction is used.

(7) Macro service requests are acknowledged and serviced even during execution of a non-maskable interrupt service program. If you do not want macro service processing to be performed during a non-maskable interrupt service program, you should manipulate the interrupt mask register in the non-maskable interrupt service program to prevent macro service generation.

(8) The RETI instruction must be used to return from a non-maskable interrupt. Subsequent interrupt acknowledgment will not be performed normally if a different instruction is used. To resume program execution from the initial state after the non-maskable interrupt has been acknowledged, see **22.12 RESTORING INTERRUPT FUNCTION TO INITIAL STATE**.

(9) Non-maskable interrupts are always acknowledged, except during non-maskable interrupt service program execution (except when a high non-maskable interrupt request is generated during execution of a low-priority non-maskable interrupt service program) and for a certain period after execution of the special instructions shown in **22.9**. Therefore, a non-maskable interrupt will be acknowledged even when the stack pointer (SP) value is undefined, in particular after reset release, etc. In this case, depending on the value of the SP, it may happen that the program counter (PC) and program status word (PSW) are written to the address of a write-inhibited special function register (SFR) (see **Table 3-5 in 3.9 SPECIAL FUNCTION REGISTERS (SFR)**), and the CPU becomes deadlocked, or the PC and PSW are written to an unexpected signal is output from a pin, or an address is which RAM is not mounted, with the result that the return from the non-maskable interrupt service program is not performed normally and a software upsets occurs. Therefore, the program following $\overline{\text{RESET}}$ release must be as follows.

```
        CSEG  AT 0
        DW    STRT
        CSEG  BASE
STRT:
        LOCATION 0FH        ; or LOCATION 0H
        MOVG  SP, #imm24
```

(10) When an interrupt related register is polled using a BF instruction, etc., the branch destination of that BR instruction, etc., should not be that instruction. If a program is written in which a branch is made to that instruction itself, all interrupts and macro service requests will be held pending until a condition whereby a branch is not made by that instruction arises.

Bad Example

```
        ⋮
LOOP:   BF   PIC0.7, $LOOP      All interrupts and macro service requests are held pending until PIC0.7 is
                                1.
        ×××                     ← Interrupts and macro service requests are not serviced until after execution
        ⋮                         of the instruction following the BF instruction.
```

Good Example (1)

```
        ⋮
LOOP:   NOP
        BF   PIC0.7, $LOOP      ← Interrupts and macro service requests are serviced after execution of the
        ×××                       NOP instruction, so that interrupts are never held pending for a long period.
        ⋮
```

Good Example (2)

```
        ⋮
LOOP:   BT   PIC0.7, $NEXT      Using a BTCLR instruction instead of a BT instruction has the advantage
        ×××                     that the flag is cleared (to 0) automatically.
        BR   $LOOP              ← Interrupts and macro service requests are serviced after execution of the
                                  BR instruction, so that interrupts are never held pending for a long period.
NEXT:   ⋮
```

(11) For a similar reason to that given in (10), if problems are caused by a long pending period for interrupts and macro service when instructions to which the above applies are used in succession, a time at which interrupts and macro service requests can be acknowledged should be provided by inserting an NOP instruction, etc., in the series of instructions.

CHAPTER 23 LOCAL BUS INTERFACE FUNCTION

The local bus interface function is provided for the connection of external memory (ROM and RAM) and I/Os.

External memory (ROM and RAM) and I/Os are accessed using the $\overline{RD}$, $\overline{WR}$, and ASTB pin signals, with pins AD0 to AD7 used as the multiplexed address/data bus and pins A8 to A19 as the address bus.

The basic bus interface timing is shown in Figures 23-7 and 23-8.

Also provided are a wait function for interfacing with low-speed memory, a refresh signal output function for refreshing pseudo-static RAM, and a bus hold function for connecting devices that have a bus master function, such as a DMA controller.

23.1 MEMORY EXTENSION FUNCTION

With the μPD784038, external memory and I/O extension can be performed by setting the memory extension mode register (MM).

The μPD784031 can access an external memory of 64 Kbytes from the initial status. The memory space that can be accessed can be extended by setting of MM.

23.1.1 Memory Extension Mode Register (MM)

The MM is an 8-bit register that performs external extension memory control, address wait number specification, and internal fetch cycle control.

The MM register can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. The MM format is shown in Figure 23-1.

$\overline{RESET}$ input sets the MM register to 20H.

Figure 23-1 Memory Extension Mode Register (MM) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MM	IFCH	0	AW	0	MM3	MM2	MM1	MM0	0FFC4H	20H	R/W

MM3	MM2	MM1	MM0	Mode	Port 4	Port 5	P60 to P63	P64/$\overline{RD}$	P65/$\overline{WR}$	ASTB/CLKOUT
0	0	0	0	Single-chip mode Note	Port	Port	Port	Port	Port	CLKOUT
0	0	1	1	256-byte extension mode Note	AD0 to AD7	Port	Port	$\overline{RD}$	$\overline{WR}$	ASTB
0	1	0	0	1-Kbyte extension mode Note	AD0 to AD7	A8, A9 Port	Port	$\overline{RD}$	$\overline{WR}$	
0	1	0	1	4-Kbyte extension mode Note	AD0 to AD7	A8 to A11 Port	Port	$\overline{RD}$	$\overline{WR}$	
0	1	1	0	16-Kbyte extension mode Note	AD0 to AD7	A8 to A13 Port	Port	$\overline{RD}$	$\overline{WR}$	
0	1	1	1	64-Kbyte extension mode	AD0 to AD7	A8 to A15	Port	$\overline{RD}$	$\overline{WR}$	
1	0	0	0	256-Kbyte extension mode	AD0 to AD7	A8 to A15	A16, A17 Port	$\overline{RD}$	$\overline{WR}$	
1	0	0	1	1-Mbyte extension mode	AD0 to AD7	A8 to A15	A16 to A19	$\overline{RD}$	$\overline{WR}$	
Other than the above				Setting prohibited						

Note With the μPD784031, these settings are the same as 64-Kbyte extension mode.

AW	Address Wait Specification
0	Disabled
1	Enabled

IFCH	Internal ROM Fetches
0	Fetch performed at same speed as external memory All wait control settings valid
1	High-speed fetches performed Wait control specification invalid

23.1.2 Memory Map with External Memory Extension

The memory map when memory extension is used is shown in Figures 23-2 to 23-5. External devices at the same addresses as the internal ROM area, internal RAM area and SFR area (excluding the external SFR area (0FFD0H to 0FFDFH)) cannot be accessed. If an access is made to these addresses, the memory or SFR in the μPD784038 has access priority and no ASTB signal, $\overline{RD}$ signal or $\overline{WR}$ signal is output (these pins remain at the inactive level). The address bus output level remains at the level output prior to this, and the address/data bus output becomes high-impedance.

Except in 1-Mbyte extension mode, the address output externally is output with the upper part of the address specified by the program masked.

Example 1:

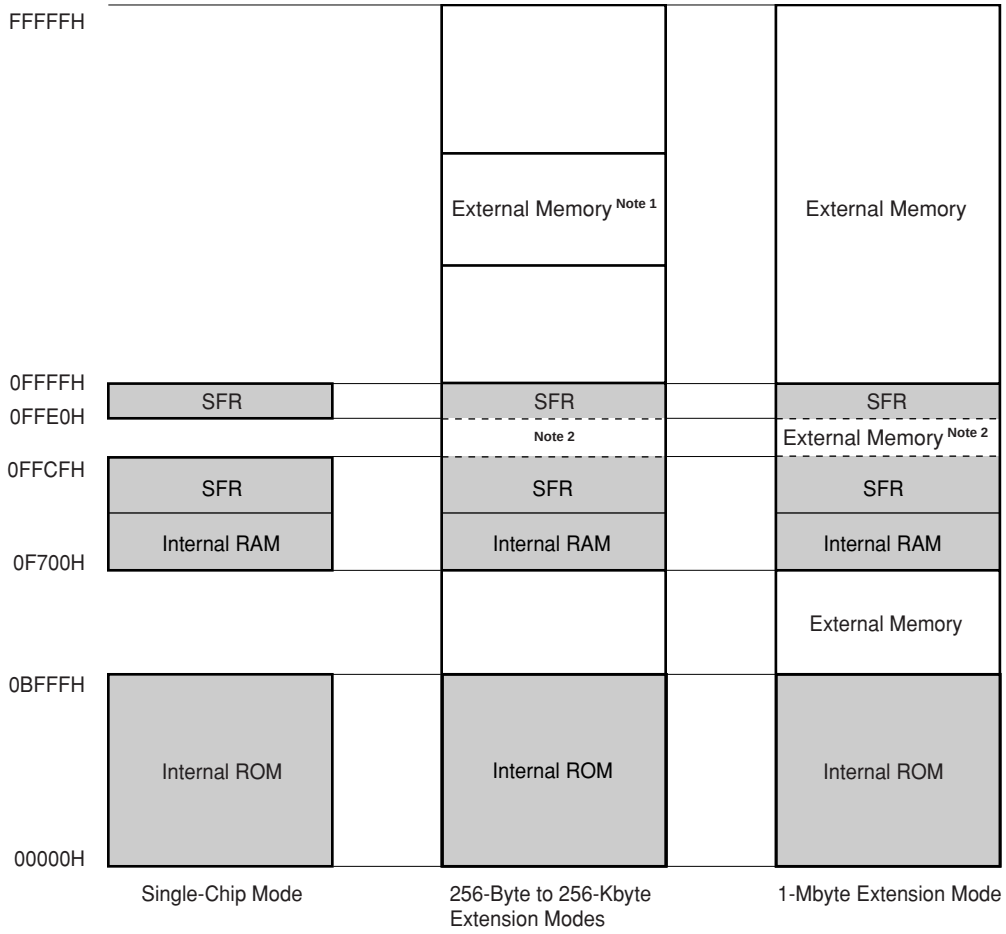
In 256-byte extension mode, when address 54321H is accessed by the program, the output address is 21H.

Example 2:

In 256-byte extension mode, when address 67821H is accessed by the program, the output address is 21H.

Figure 23-2 μPD784035 Memory Map (1/2)

(a) When LOCATION 0H instruction is executed

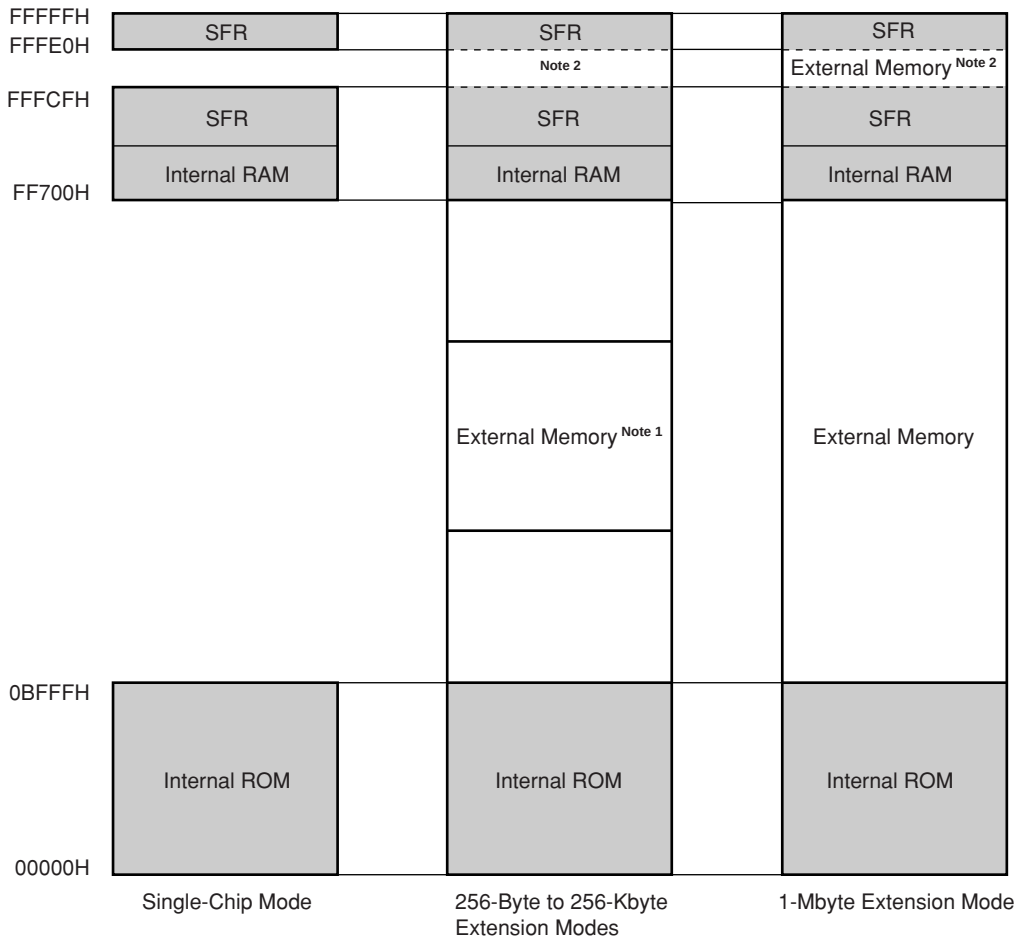


Notes 1. Any extension size area in unshaded part

2. External SFR area

Figure 23-2 μPD784035 Memory Map (2/2)

(b) When LOCATION 0FH instruction is executed

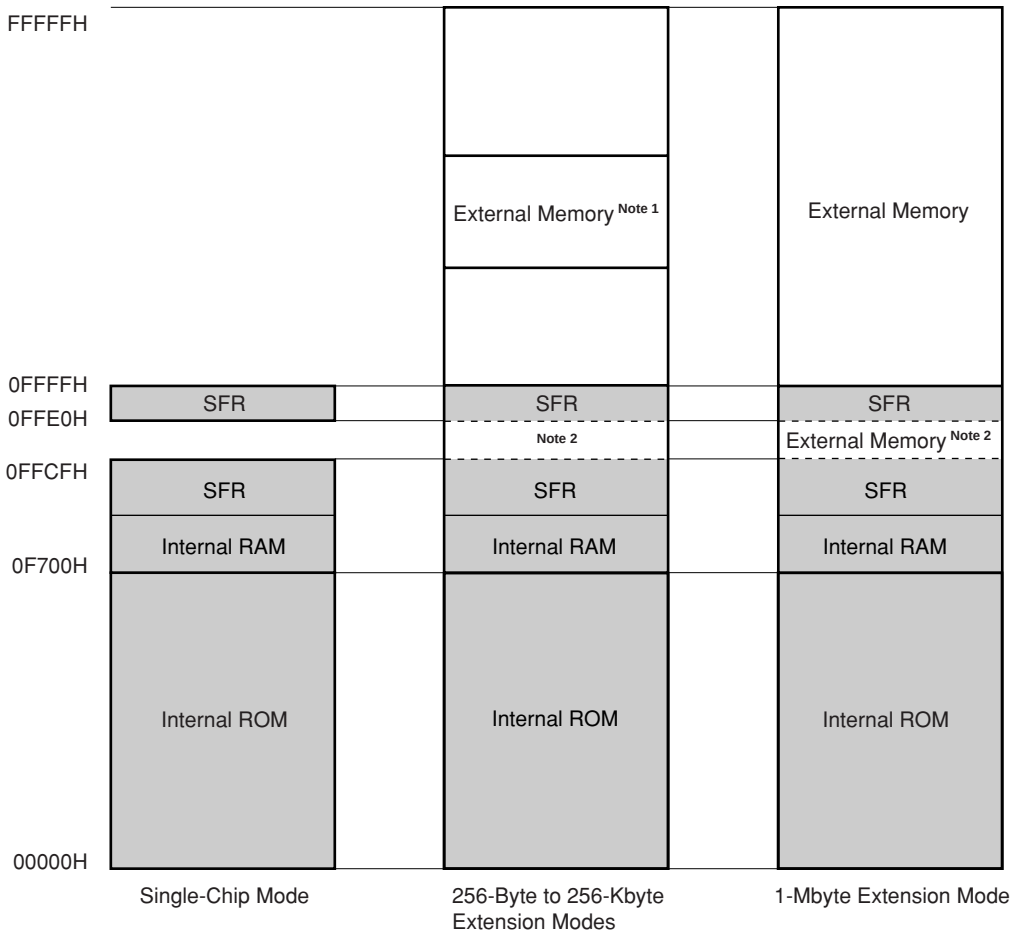


Notes 1. Any extension size area in unshaded part

2. External SFR area

Figure 23-3 μPD784036 Memory Map (1/2)

(a) When LOCATION 0H instruction is executed

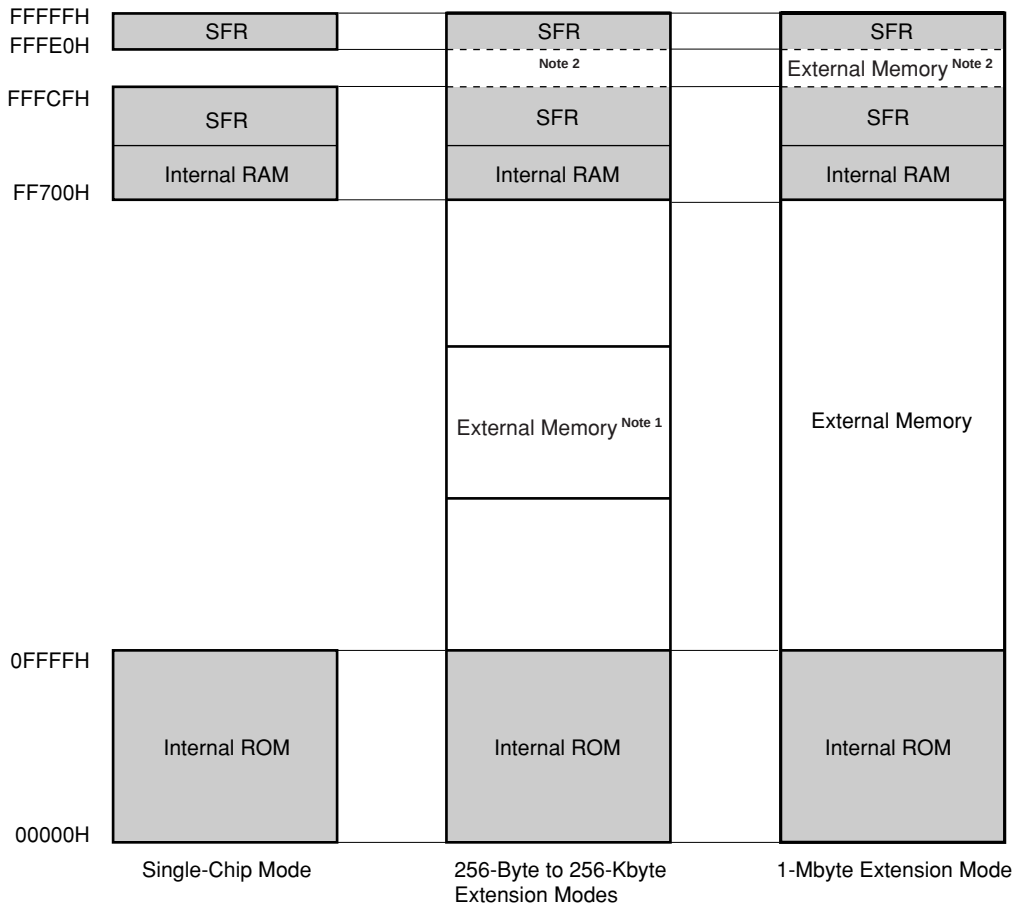


Notes 1. Any extension size area in unshaded part

2. External SFR area

Figure 23-3 μPD784036 Memory Map (2/2)

(b) When LOCATION 0FH instruction is executed

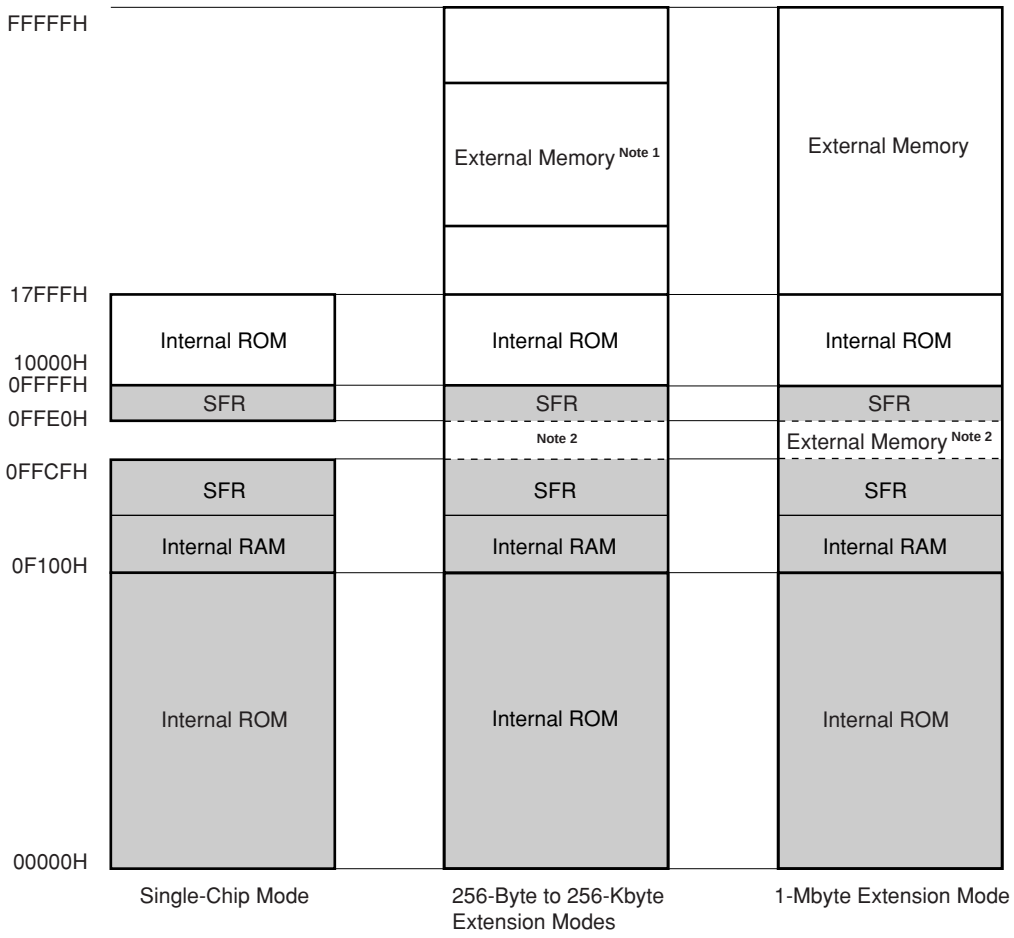


Notes 1. Any extension size area in unshaded part

2. External SFR area

Figure 23-4 μPD784037 Memory Map (1/2)

(a) When LOCATION 0H instruction is executed

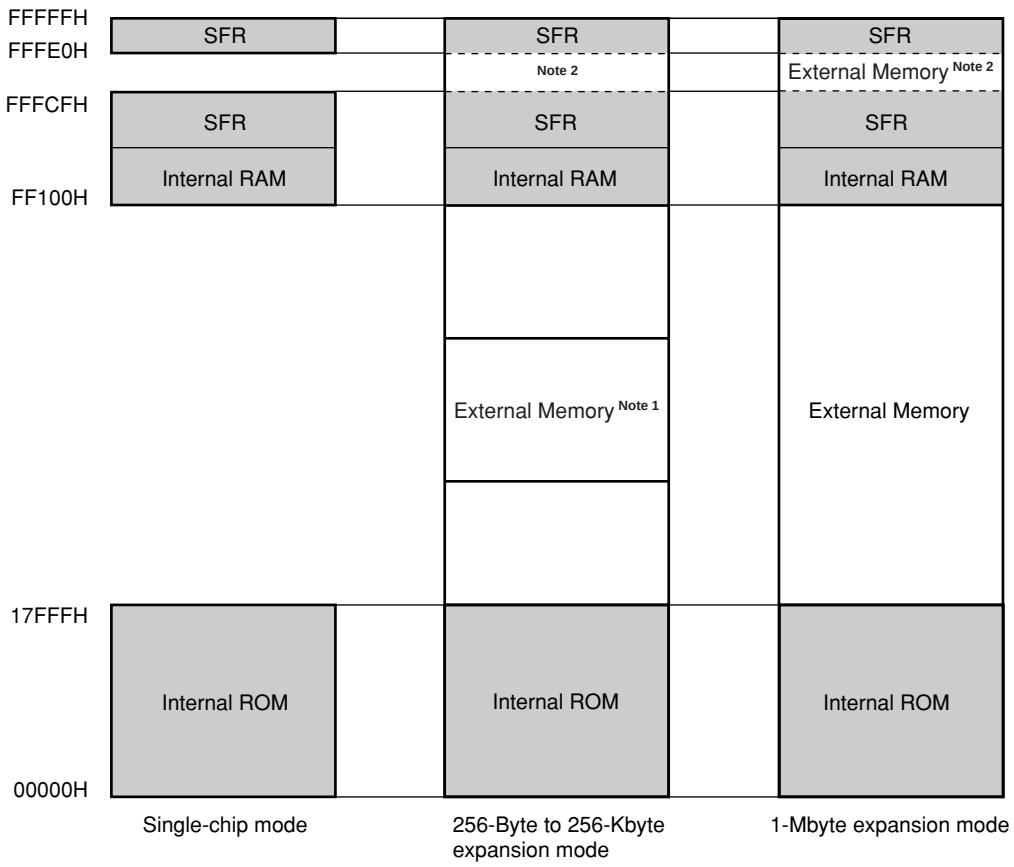


Notes 1. Any extension size area in unshaded part

2. External SFR area

Figure 23-4 μPD784037 Memory Map (2/2)

(b) When LOCATION 0FH instruction is executed

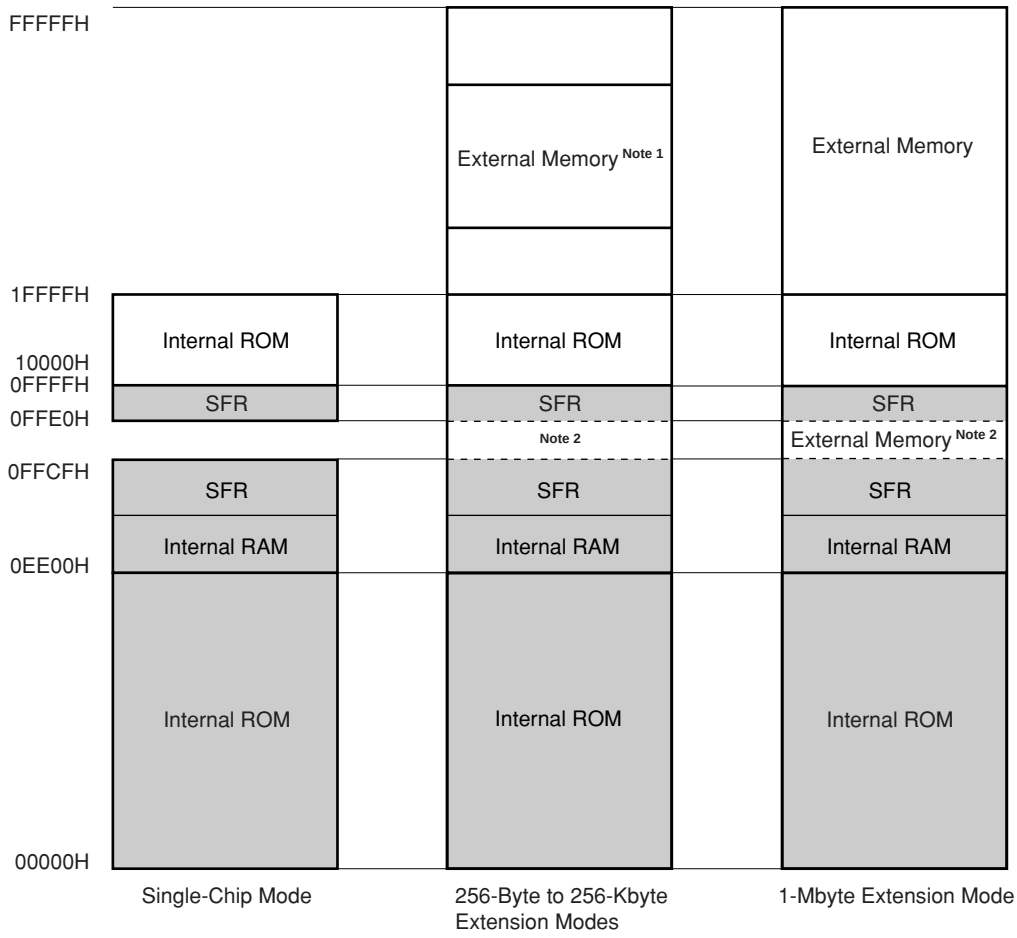


Notes 1. Any extension size area in unshaded part

2. External SFR area

Figure 23-5 μPD784038 Memory Map (1/2)

(a) When LOCATION 0H instruction is executed

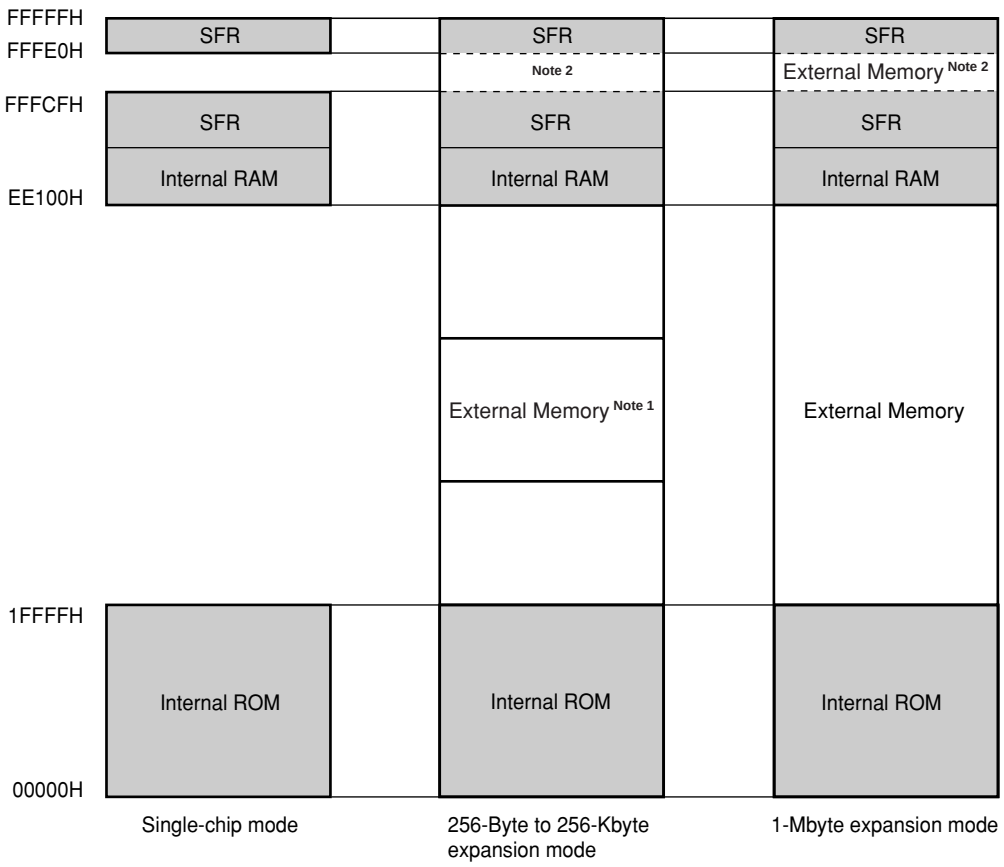


Notes 1. Any extension size area in unshaded part

2. External SFR area

Figure 23-5 μPD784038 Memory Map (2/2)

(b) When LOCATION 0FH instruction is executed

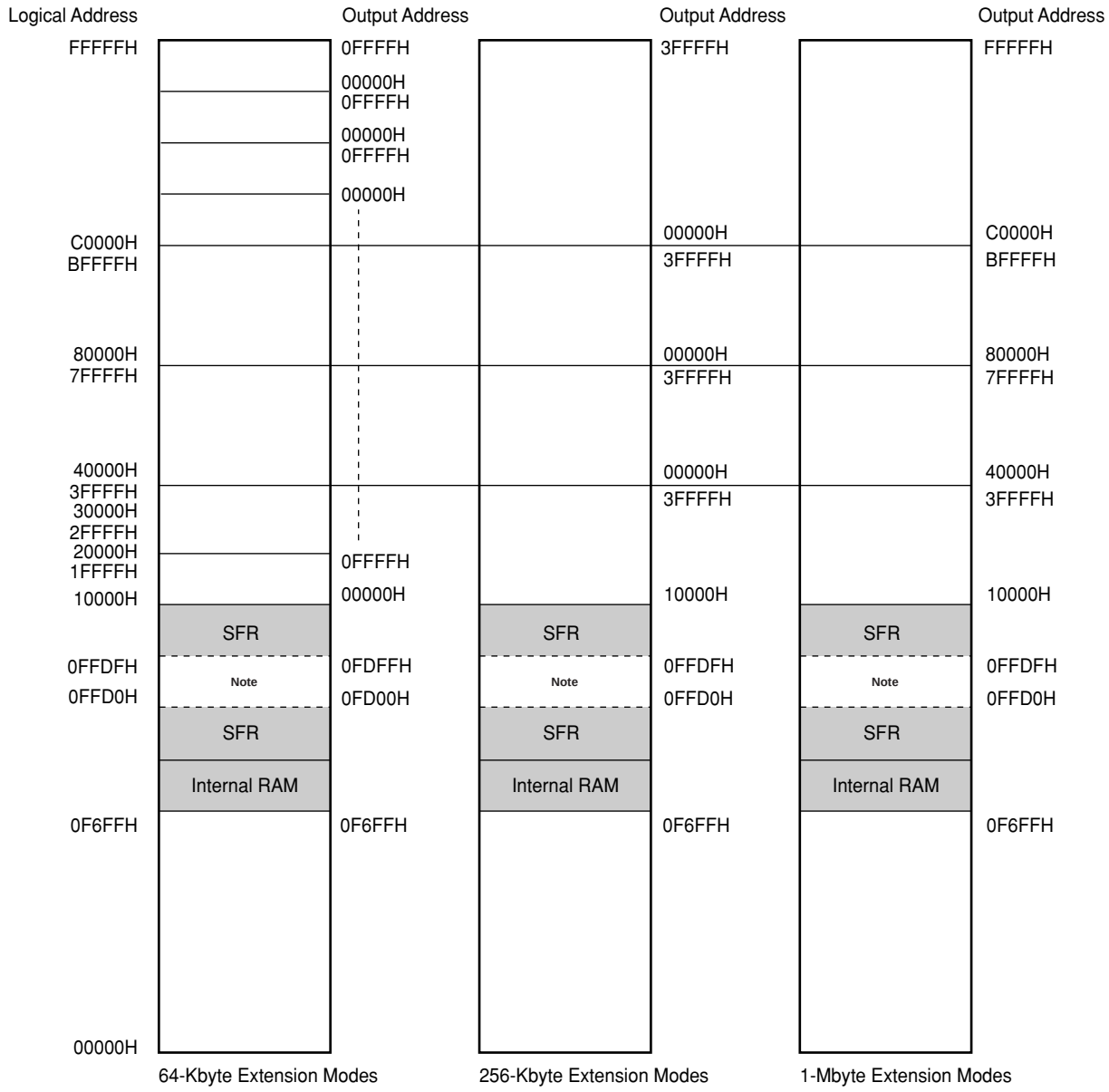


Notes 1. Any extension size area in unshaded part

2. External SFR area

Figure 23-6 μPD784031 Memory Map (1/2)

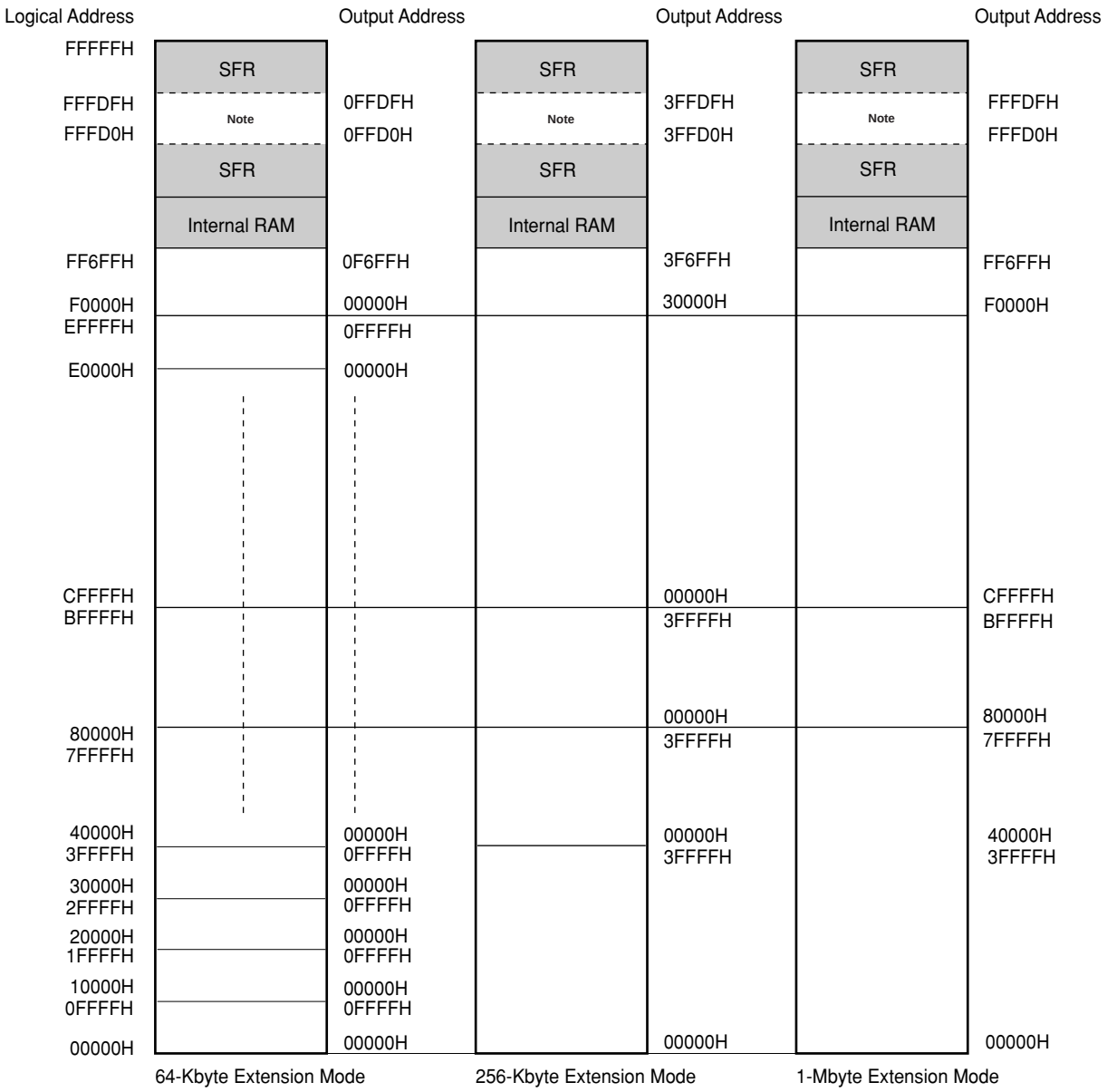
(a) When LOCATION 0H instruction is executed



Note External SFR area

Figure 23-6 μPD784031 Memory Map (2/2)

(b) When LOCATION 0FH instruction is executed



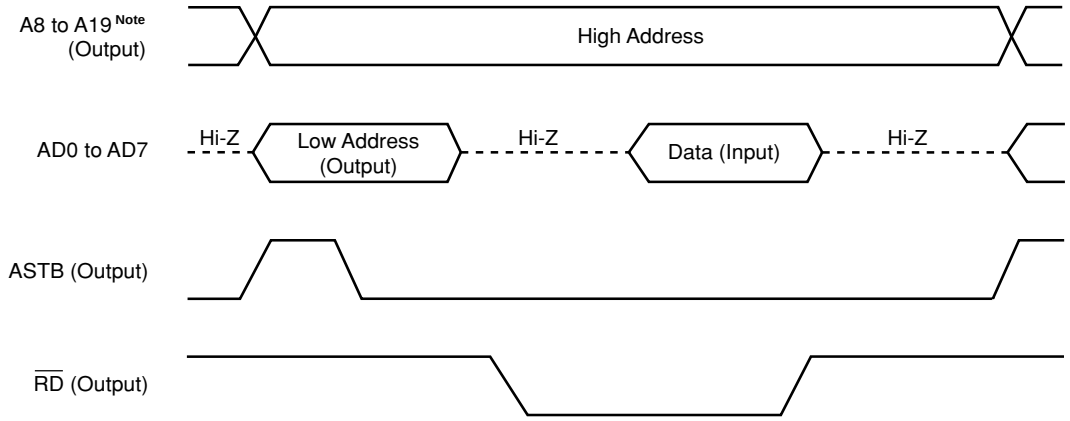
Note External SFR area

23.1.3 Basic Operation of Local Bus Interface

The local bus interface accesses external memory using ASTB, $\overline{RD}$, $\overline{WR}$, an address/data bus (AD0 to AD7) and address bus (A8 to A19). When the local bus interface is used, P64, P65 and port 4 automatically operate as $\overline{RD}$, $\overline{WR}$, and AD0 to AD7. With the μPD784031, these pins always operate only as $\overline{RD}$, $\overline{WR}$, and AD0 to AD7. On the address bus, only the pins that correspond to the extension memory size operate as address bus pins.

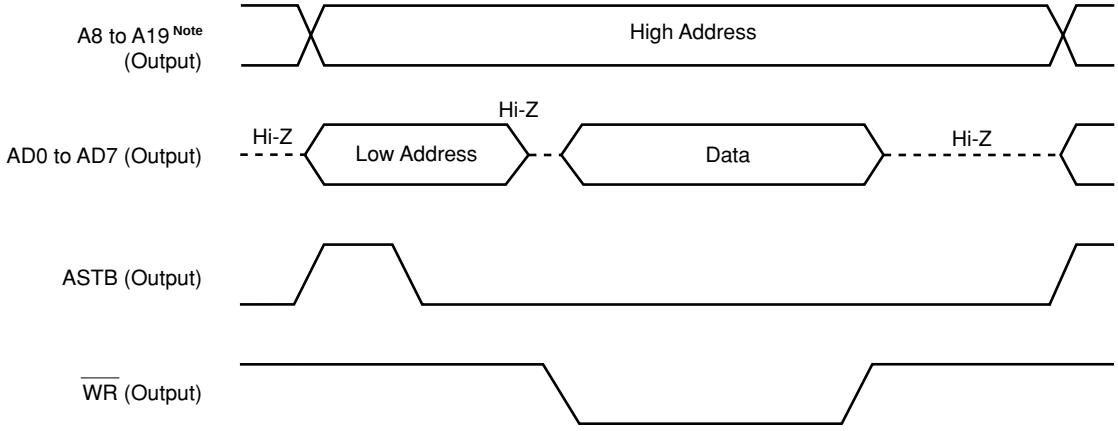
An outline of the memory access timing is shown in Figures 23-7 and 23-8.

Figure 23-7 Read Timing



Note The number of address bus pins used depends on the extension mode size.

Figure 23-8 Write Timing



Note The number of address bus pins used depends on the extension mode size.

23.2 WAIT FUNCTION

When a low-speed memory or I/O is connected externally to the μPD784038, waits can be inserted in the external memory access cycle.

There are two kinds of wait cycle, an address wait for securing the address decoding time, and an access wait for securing the access time.

23.2.1 Wait Function Control Registers

(1) Memory extension mode register (MM)

The IFCH bit of the MM performs wait control setting for internal ROM accesses, and the AW bit performs address wait setting.

The MM can be read or written to with an 8-bit manipulation instruction. The MM format is shown in Figure 23-9.

When $\overline{\text{RESET}}$ is input, the MM register is set to 20H, the same cycle as for external memory is used for internal ROM accesses, and the address wait function is validated.

Figure 23-9 Memory Extension Mode Register (MM) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
MM	IFCH	0	AW	0	MM3	MM2	MM1	MM0	0FFC4H	20H	R/W

Memory extension mode settings (see **23.1 MEMORY EXTENSION FUNCTION**)

AW	Address Wait Specification
0	Disabled
1	Enabled

IFCH	Internal ROM Fetches
0	Fetch performed at same speed as external memory All wait control settings valid
1	High-speed fetches performed Wait control specification invalid

(2) Programmable wait control registers (PWC1/PWC2)

The PWC1 and PWC2 specify the number of waits.

PWC1 is an 8-bit register that divides the space from 0 to FFFFH into four, and specifies wait control for each of these four spaces. PWC2 is a 16-bit register that divides the space from 10000H to FFFFH into four, and specifies wait control for each of these four spaces.

The PWC1 can be read or written to with an 8-bit manipulation instruction, and the PWC2 with a 16-bit manipulation instruction. The PWC1 and PWC2 formats are shown in Figure 23-9.

The high-order 8 bits of the PWC2 are fixed at AAH, and therefore ensure that the high-order 8 bits are set to AAH.

When $\overline{\text{RESET}}$ is input, the PWC1 is set to AAH, and the PWC2 to AAAAH, and 2-wait insertion is performed on the entire space.

Caution Do not set external wait to the internal ROM area. Otherwise, the CPU may be in the deadlock status which can be cleared only by reset input.

Figure 23-10 Programmable Wait Control Register (PWC1/PWC2) Format

(a) Programmable wait control register 1 (PWC1)

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PWC1	PW31	PW30	PW21	PW20	PW11	PW10	PW01	PW00	0FFC7H	AAH	R/W

(n = 0 to 3)

Addresses Subject to Wait		00C000H to 00FFFFH Note	008000H to 00BFFFH	004000H to 007FFFH	000000H to 003FFFH
PWn1	PWn0				
0	0	No access wait cycle inserted			
0	1	1 access wait cycle inserted			
1	0	2 access wait cycles inserted			
1	1	Access wait cycle inserted only for $\overline{\text{WAIT}}$ pin low-level input period			

Note Except part overlapping internal data area

(b) Programmable wait control register 2 (PWC2)

	15	14	13	12	11	10	9	8
PWC2	1	0	1	0	1	0	1	0

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
	PW71	PW70	PW61	PW60	PW51	PW50	PW41	PW40	0FFC8H	AAAAH	R/W

(n = 4 to 7)

Addresses Subject to Wait		080000H to 0FFFFFH Note	040000H to 07FFFFH	020000H to 03FFFFH	010000H to 01FFFFH
PWn1	PWn0				
0	0	No access wait cycle inserted			
0	1	1 access wait cycle inserted			
1	0	2 access wait cycles inserted			
1	1	Access wait cycle inserted only for $\overline{\text{WAIT}}$ pin low-level input period			

Note Except for part overlapping internal data area

Caution When the bus hold function is used, access wait control cannot be performed by means of the $\overline{\text{WAIT}}$ pin, and 0, 1 or 2 waits must be selected for the entire space.

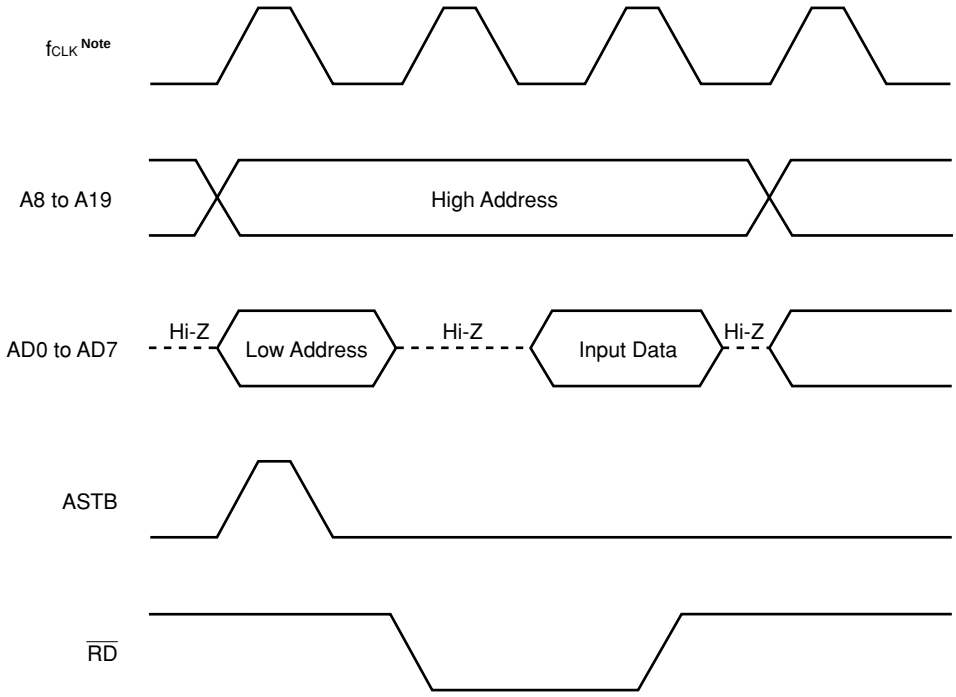
23.2.2 Address Waits

Address waits are used to secure the address decoding time. If the AW bit of the memory extension mode register (MM) is set (to 1), waits are inserted in every memory access **Note**. When an address wait is inserted, the high-level period of the ASTB signal is extended by one system clock cycle (62.5 ns: f_{CLK} = 16 MHz).

Note Except for the internal RAM, internal SFRs, and internal ROM during high-speed fetch.
If it is specified that the internal ROM is accessed in the same cycle as the external ROM, an address wait state is inserted even when the internal ROM is accessed.

Figure 23-11 Address Wait Function Read/Write Timing (1/3)

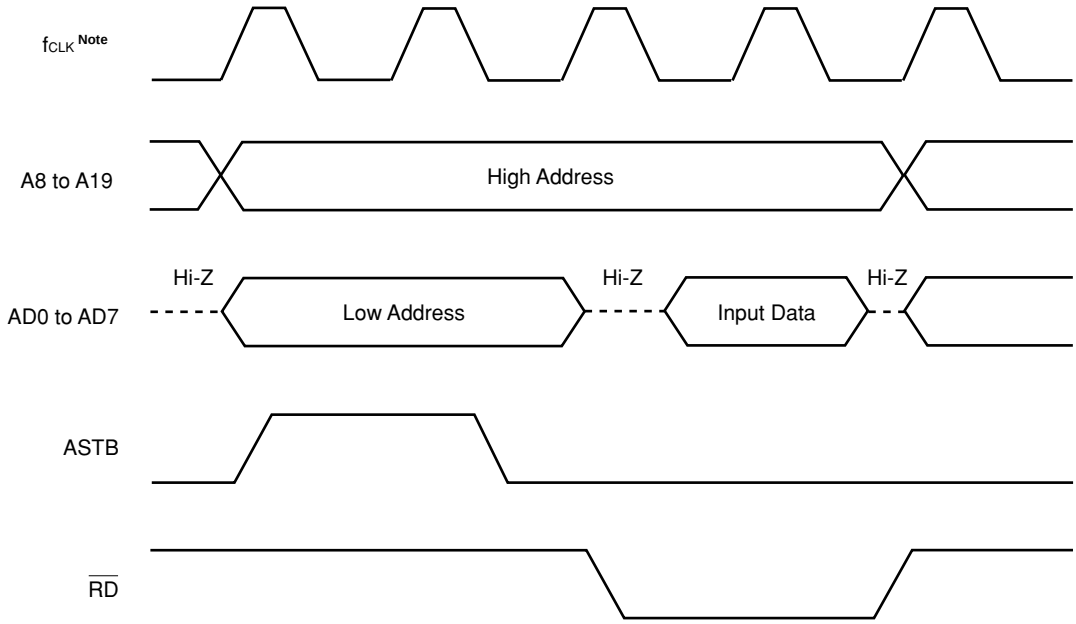
(a) Read timing with no address wait insertion



Note f_{CLK}: Internal system clock frequency. This signal is present inside the μPD784038 only.

Figure 23-11 Address Wait Function Read/Write Timing (2/3)

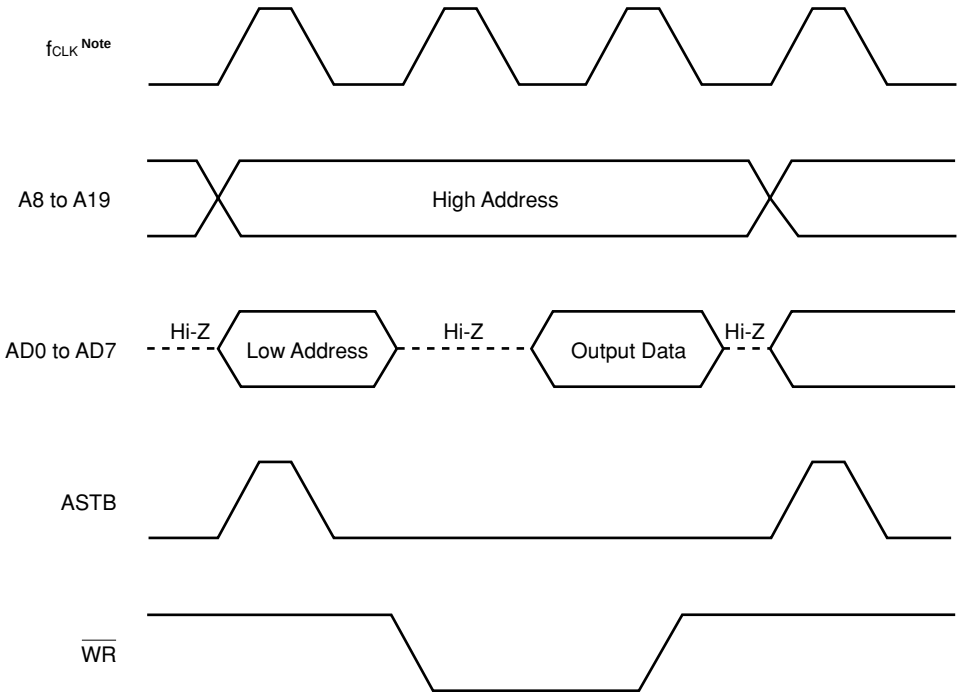
(b) Read timing with address wait insertion



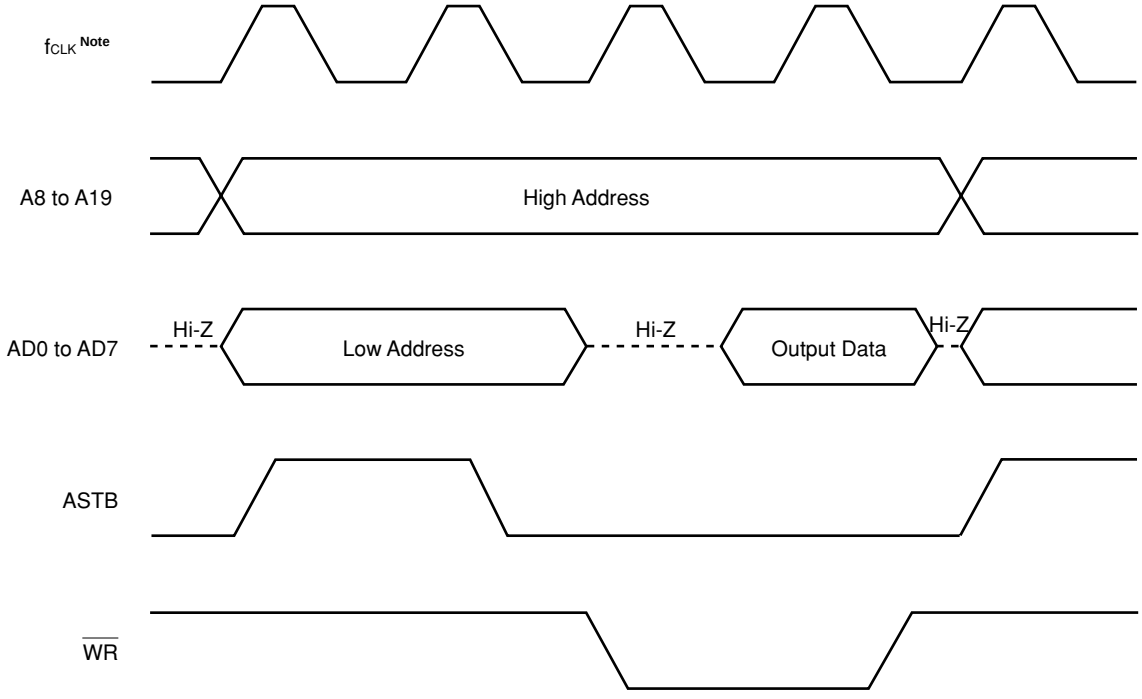
Note f_{CLK}: Internal system clock frequency. This signal is present inside the μPD784038 only.

Figure 23-11 Address Wait Function Read/Write Timing (3/3)

(c) Write timing with no address wait insertion



(d) Write timing with address wait insertion



Note f_{CLK}: Internal system clock frequency. This signal is present inside the μPD784038 only.

23.2.3 Access Waits

Access waits are inserted in the $\overline{RD}$ or $\overline{WR}$ signal low-level period, and extend the low-level period by 1/f_{CLK} (62.5 ns: f_{CLK} = 16 MHz) per cycle.

There are two wait insertion methods, using either the programmable wait function that automatically inserts the preset number of cycles, or the external wait function controlled by a wait signal from outside.

For wait cycle insertion control, the 1-Mbyte memory space is divided into eight as shown in Figure 23-12, and control is specified for each space by means of the programmable wait control registers (PWC1/PWC2). Waits are not inserted in accesses to internal ROM or internal RAM using high-speed fetches. In accesses to internal SFRs, waits are inserted at the necessary times regardless of this specification.

If access operations are specified as being performed in the same number of cycles as for external ROM, waits are inserted also in internal ROM accesses in accordance with the PWC1 settings.

If there is a space for which control by a wait signal from outside has been selected by means of the PWC1/PWC2, the P66 pin operates as the $\overline{WAIT}$ signal input pin. After $\overline{RESET}$ input, the P66 pin operates as a general-purpose input/output port.

Bus timing in the case of access wait insertion is shown in Figures 23-13 to 23-15.

Cautions 1. The external wait function cannot be used when the bus hold function is used.

2. Do not set external wait to the internal ROM area. Otherwise, the CPU may be in the deadlock status which can be cleared only by reset input.

Figure 23-12 Wait Control Spaces

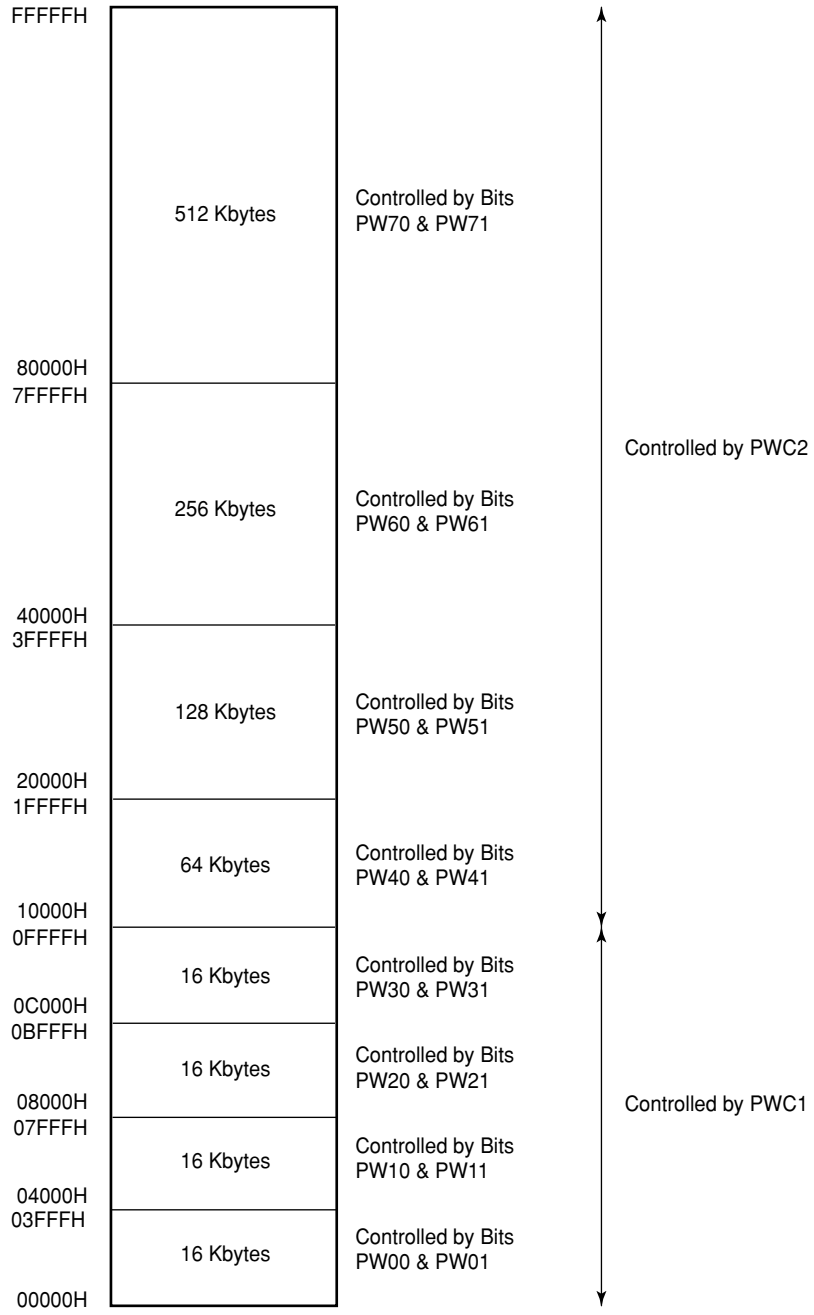
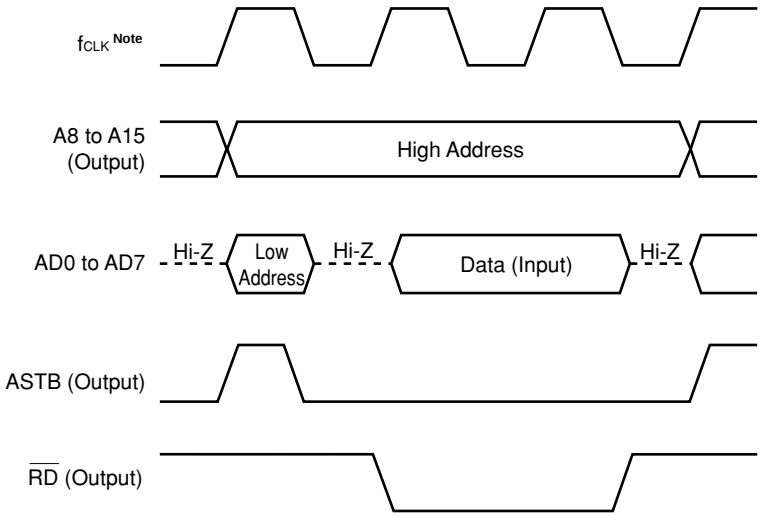
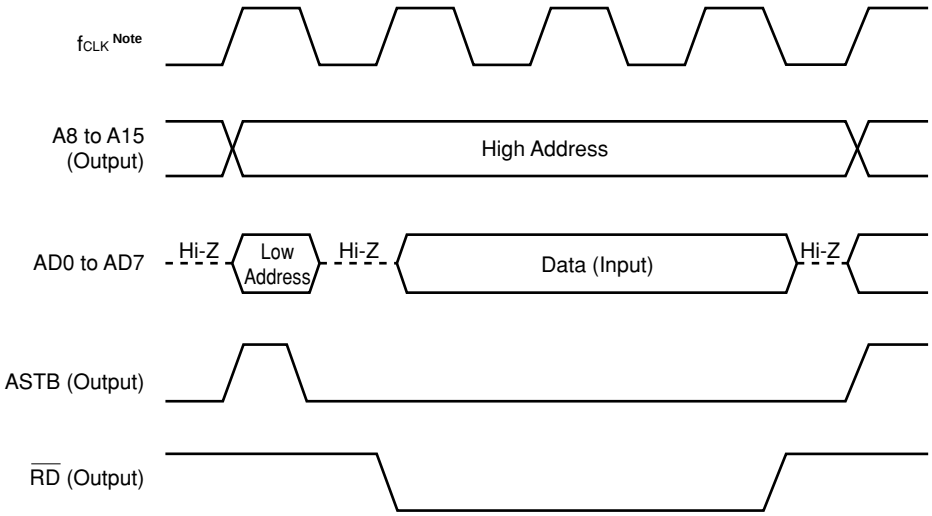


Figure 23-13 Access Wait Function Read Timing (1/2)

(a) 0 wait cycles set



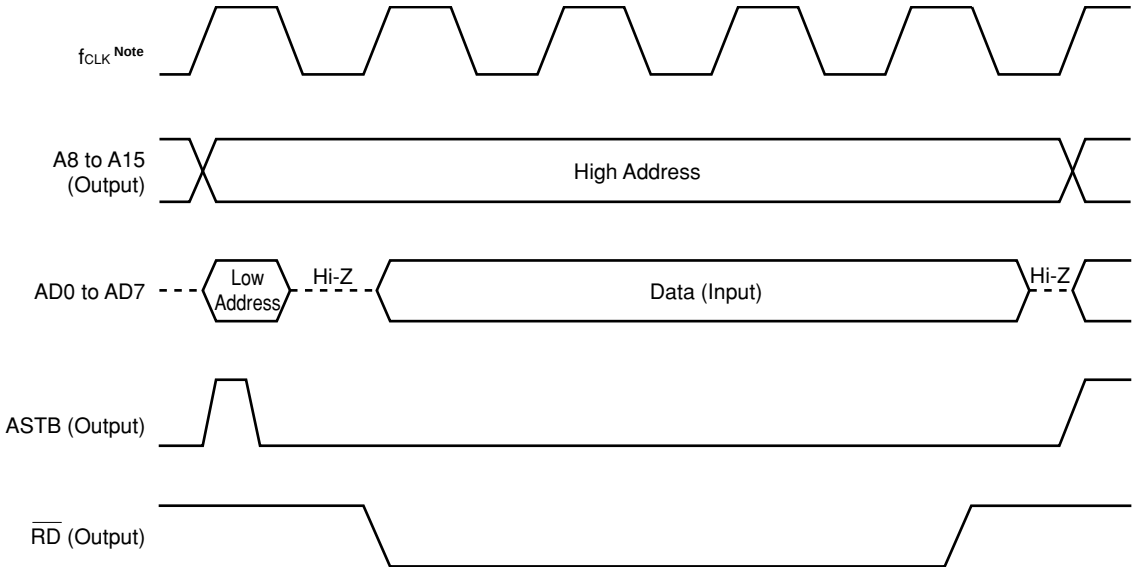
(b) 1 wait cycle set



Note f_{CLK}: Internal system clock frequency. This signal is only present inside the μPD784038.

Figure 23-13 Access Wait Function Read Timing (2/2)

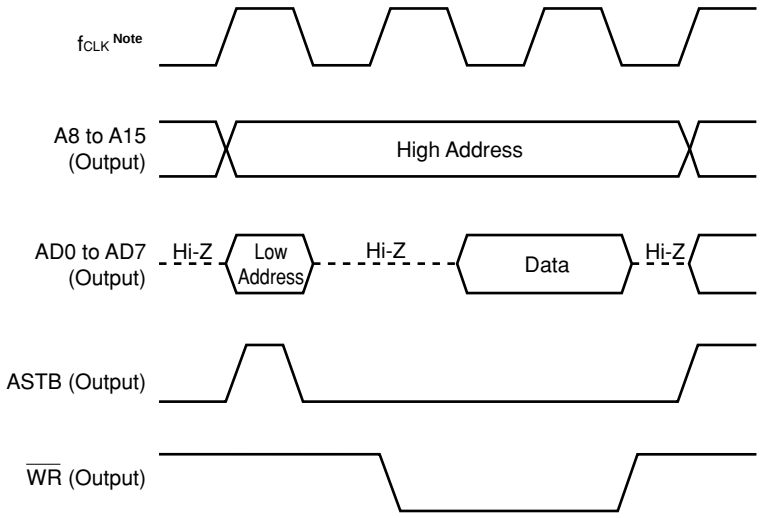
(c) 2 wait cycles set



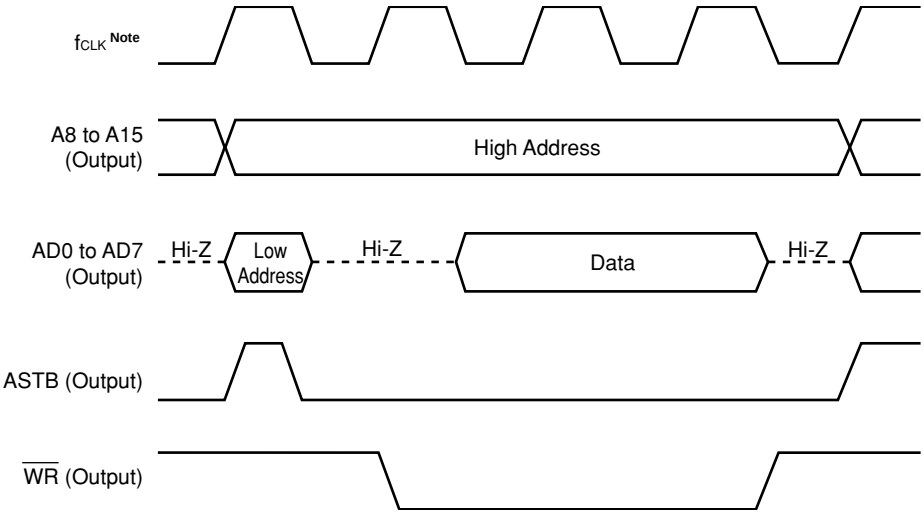
Note f_{CLK}: Internal system clock frequency. This signal is only present inside the μPD784038.

Figure 23-14 Access Wait Function Write Timing (1/2)

(a) 0 wait cycles set



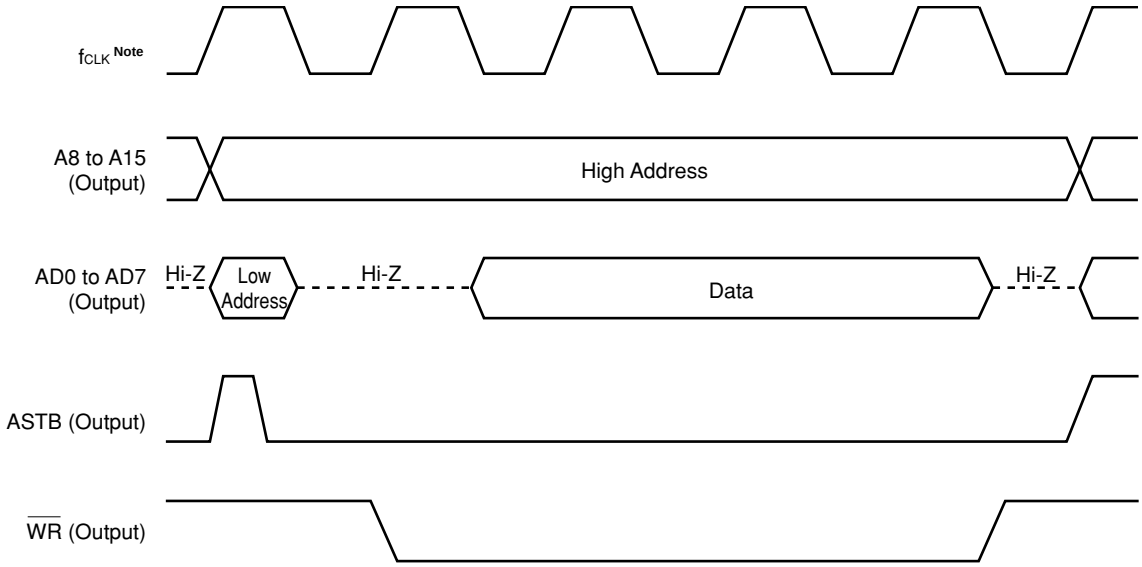
(b) 1 wait cycle set



Note f_{CLK}: Internal system clock frequency. This signal is only present inside the μPD784038.

Figure 23-14 Access Wait Function Write Timing (2/2)

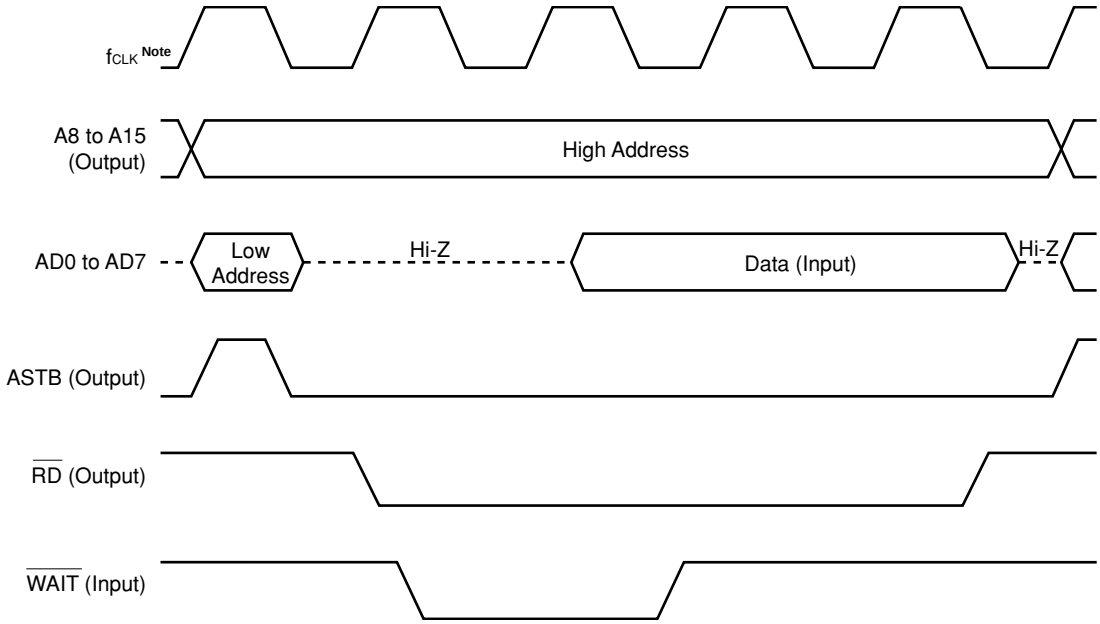
(c) 2 wait cycles set



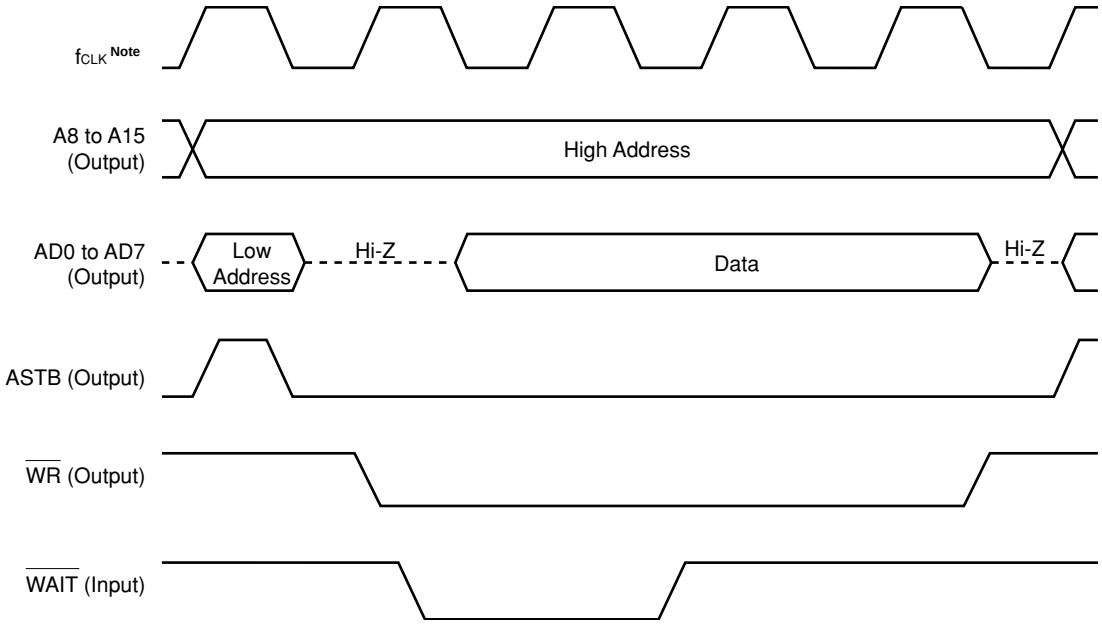
Note f_{CLK}: Internal system clock frequency. This signal is only present inside the μPD784038.

Figure 23-15 Timing with External Wait Signal

(a) Read timing



(b) Write timing



Note f_{CLK}: Internal system clock frequency. This signal is only present inside the μPD784038.

23.3 PSEUDO-STATIC RAM REFRESH FUNCTION

The μPD784038 incorporates a pseudo-static RAM refresh function for direct connection of pseudo-static RAM.

The pseudo-static RAM refresh function outputs refresh pulses at any desired intervals. The refresh pulse output interval is specified by the refresh mode register (RFM) setting.

The refresh area specification register (RFA) specifies the addresses on which refresh operations can be performed at the same time as memory access operations. This enables bus cycle insertions for refresh operations to be greatly decreased, thus minimizing the reduction in performance due to refresh operations.

The μPD784038 is provided with a function for supporting self-refresh operations that offers low power consumption by a pseudo-static RAM application system.

Caution The refresh function cannot be used when the bus hold function is used.

23.3.1 Control Registers

(1) Refresh mode register (RFM)

The RFM is an 8-bit register that controls the pseudo-static RAM refresh cycle and switching to self-refresh operations. The RFM register can be read or written to with an 8-bit manipulation instruction or bit manipulation instruction. RFM format is shown in Figure 23-16.

$\overline{\text{RESET}}$ input clears the RFM register to 00H and sets the $\overline{\text{REFRQ}}$ pin to port mode, so that it operates as the alternate-function P67 pin.

Figure 23-16 Refresh Mode Register (RFM) Format

	⑦	6	5	④	3	2	1	0
RFM	RFLV	0	0	RFEN	0	0	RFT1	RFT0

Address	After Reset	R/W
0FFCCH	00H	R/W

(f_{CLK} = 16 MHz)

RFT1	RFT0	Refresh Pulse Output Cycle Specification
0	0	$32/f_{CLK}$ Note (2 μs)
0	1	$64/f_{CLK}$ (4 μs)
1	0	$128/f_{CLK}$ (8 μs)
1	1	$256/f_{CLK}$ (16 μs)

Note f_{CLK}: Internal system clock frequency

RFLV	RFEN	$\overline{\text{REFRQ}}$ Pin Output Control
×	0	Port mode
0	1	Self-refresh operation ($\overline{\text{REFRQ}}$ low level)
1	1	Refresh pulse output enabled

Remark ×: 0 or 1

Caution The refresh function cannot be used when the bus hold function is used. In this case, ensure that refreshing is specified as disabled.

(2) Refresh area specification register (RFA)

The RFA is an 8-bit register that specifies the areas on which refresh operations can be performed at the same time as memory access operations.

The RFA register can be read or written to with an 8-bit manipulation instruction and bit manipulation instruction. RFA format is shown in Figure 23-17.

$\overline{\text{RESET}}$ input clears the RFA register to 00H.

Figure 23-17 Refresh Area Specification Register (RFA) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
RFA	RFA7	RFA6	RFA5	RFA4	RFA3	RFA2	RFA1	RFA0	0FFCDH	00H	R/W

(n = 0 to 7)

RFAn \ Refresh Specification Area	080000H to 0FFFFFH	040000H to 07FFFFH	020000H to 03FFFFH	010000H to 01FFFFH	00C000H to 00FFFFH	008000H to 00BFFFH	004000H to 007FFFH	000000H to 003FFFH
0	Refreshing performed at same time as memory access operations in corresponding block							
1	Refreshing performed exclusively from memory access operations in corresponding block							

23.3.2 Operations

(1) Pulse refresh operation

To support the pulse refresh cycles of pseudo-static RAM, refresh pulses are output from the $\overline{\text{REFRQ}}$ pin in synchronization with bus cycles.

The system clock frequency and bits 1 and 0 (RFT1/RFT0) of the refresh mode register (RFM) are adjusted so that 512 or more refresh pulses are generated in an 8-ms period.

Table 23-1 System Clock Frequency and Refresh Pulse Output Cycle When Pseudo-static RAM is Used

System Clock Frequency (f_{CLK}) MHz	Refresh Pulse Output Cycle Specification	RFT1	RFT0
$8.192 < f_{CLK} \leq 16$	$128/f_{CLK}$	1	0
$4.096 < f_{CLK} \leq 8.192$	$64/f_{CLK}$	0	1
$2.048 < f_{CLK} \leq 4.096$	$32/f_{CLK}$	0	0

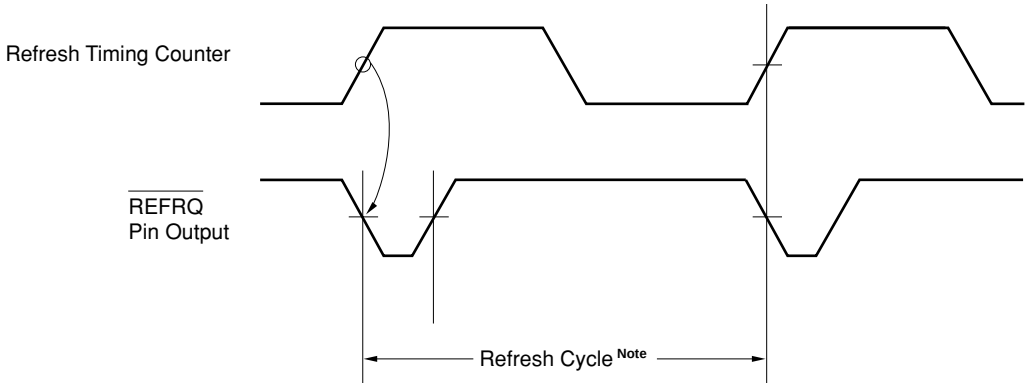
These pulse refresh operations are performed so that they do not overlap external memory access operations. During a refresh cycle, an external memory access cycle is held pending (ASTB, $\overline{\text{RD}}$, $\overline{\text{WR}}$, etc. are inactive), and during an external memory access cycle, a refresh cycle is held pending.

If there is no overlapping with an external memory access operation, the refresh cycle is performed without affecting CPU instruction execution.

(a) Internal memory accesses

In the case of internal memory accesses in which the external pseudo-static RAM is not accessed, also, refresh bus cycles are output at the intervals specified by the refresh mode (RFM) register so that the data stored in the pseudo-static RAM is retained. In this case, CPU instruction execution is not affected.

Figure 23-18 Pulse Refresh Operation in Internal Memory Access



Note Cycle specified by the RFT1 and RFT0 bits of the RFM

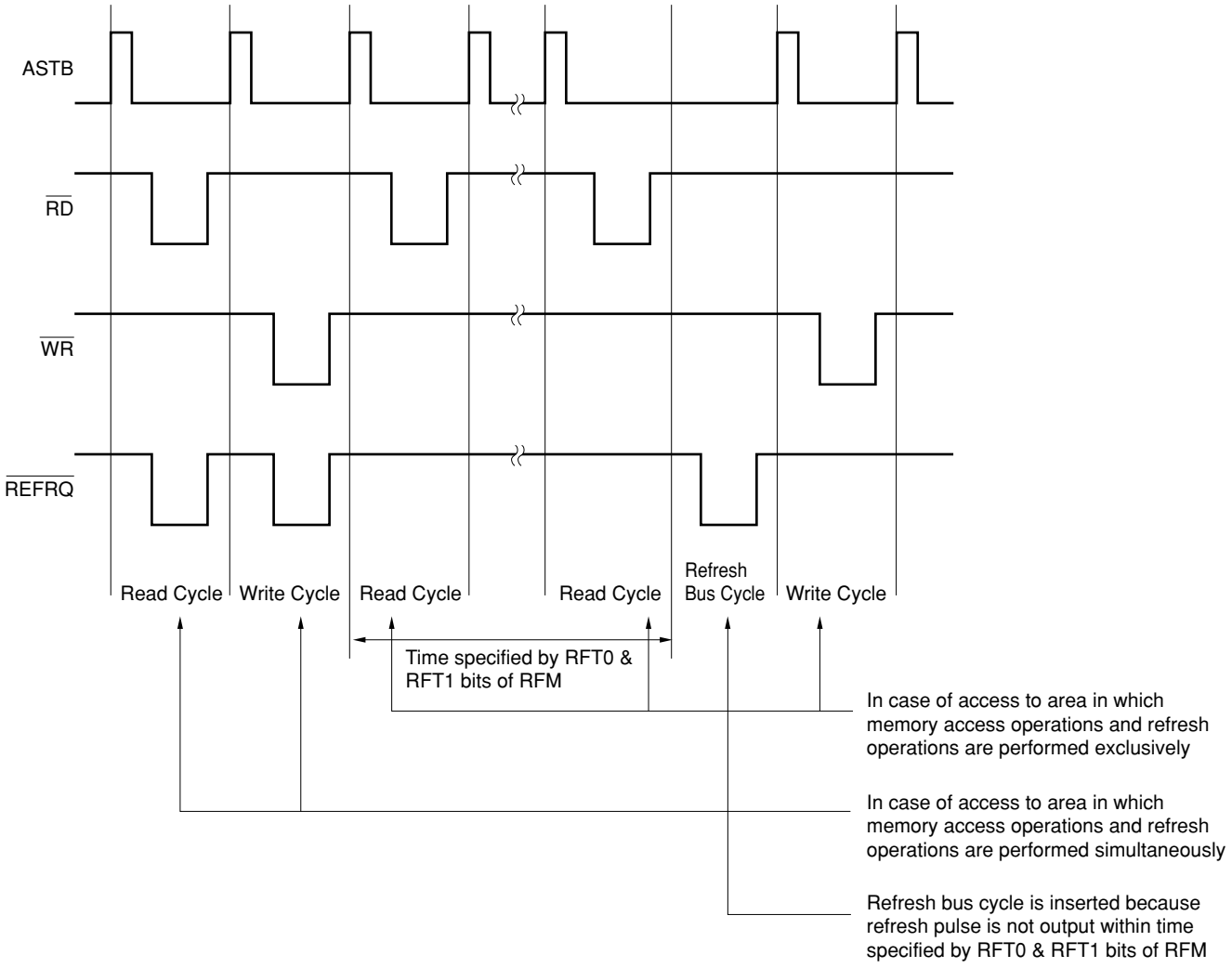
(b) External memory accesses

When an access is made to an address corresponding to a cleared (to 0) bit in the refresh area specification register (RFA), a refresh pulse is always output from the $\overline{\text{REFRQ}}$ pin at the same time as the $\overline{\text{RD}}$ signal or $\overline{\text{WR}}$ signal, irrespective of the cycle specified by the refresh mode register (RFM).

After refresh pulse output, accesses to internal memory or accesses to addresses corresponding to a set (to 1) bit in the RFA continue, and after the time specified by the RFT0 and RFT1 bits of the RFM has elapsed, a refresh bus cycle is generated so as not to overlap a memory access cycle, and a refresh pulse is output.

In this way, refreshing can be performed while memory that does not need refreshing, such as PROM, is being accessed, refresh bus cycle insertions can be reduced, and instruction execution can be performed efficiently.

Figure 23-19 Refresh Pulse Output Operation



(2) Self-refresh operation

This mode is used to retain the contents of pseudo-static RAM in standby mode.

(a) Self-refresh operating mode setting

When bit 4 (RFEN) of the refresh mode (RFM) register is set to “1”, and bit 7 (RFLV) to “0”, a low level is output from the $\overline{\text{REFRQ}}$ pin, and the self-refresh operating mode is specified for the pseudo-static RAM.

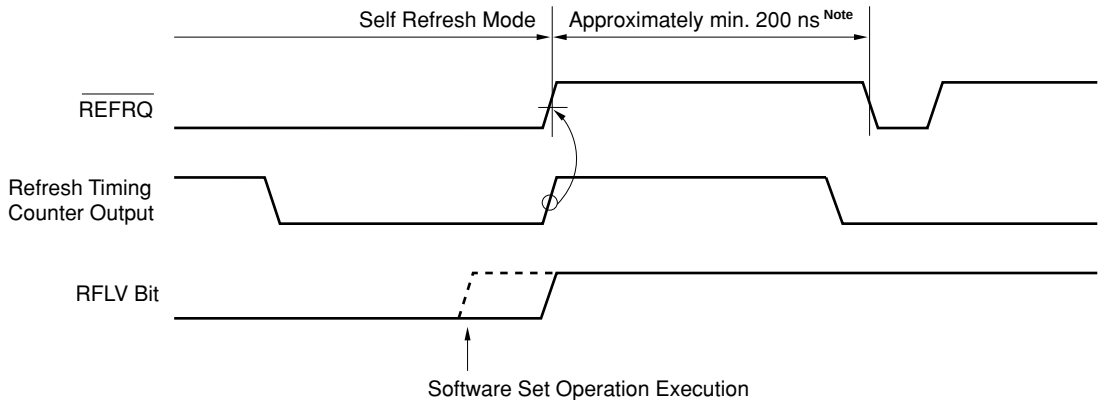
(b) Return from self-refresh operation

Refresh pulse output to the pseudo-static RAM is disabled approximately 200 ns **Note** after the $\overline{\text{REFRQ}}$ pin output level changes from low to high. Therefore, the μPD784038 arranges for refresh pulses not to be output during the disabled time by raising the $\overline{\text{REFRQ}}$ pin in synchronization with the refresh timing counter.

To enable this low-to-high transition of the $\overline{\text{REFRQ}}$ pin level to be recognized, the RFLV bit read level is set (to 1) when the $\overline{\text{REFRQ}}$ pin level changes from low to high.

Note This time varies according to the speed rank, etc. of the pseudo-static RAM.

Figure 23-20 Timing for Return from Self-Refresh Operation



Note Refreshing disabled time

23.4 BUS HOLD FUNCTION

The bus hold function is provided for the connection of a device that functions as the bus master, such as a DMA controller. In response to a request from the bus master device, all local bus interface pins are set to high impedance (except HLDAK), and local bus interface mastership is passed to that device.

The bus hold function cannot be used when the external wait function or refresh function is used.

23.4.1 Hold Mode Register (HLDM)

The HLDM is an 8-bit register that specifies enabling/disabling of the bus hold function. HLDM format is shown in Figure 23-21.

When $\overline{\text{RESET}}$ is input, the HLDM register is cleared to 00H, so that the bus hold function is disabled. The HLDRQ and HLDAK pins are set to port mode and operate as the P66 and P67 pins.

Figure 23-21 Hold Mode Register (HLDM) Format

	⑦	6	5	4	3	2	1	0	Address	After Reset	R/W
HLDM	HLDE	0	0	0	0	0	0	0	0FFC5H	00H	R/W

HLDE	Bus Hold Enabling/Disabling	P66	P67
0	Disabled	Port or $\overline{\text{WAIT}}$	Port or $\overline{\text{REFRQ}}$
1	Enabled	HLDRQ	HLDAK

Caution The bus hold function must be disabled when the external wait function or refresh function is used.

23.4.2 Operation

When the HLDE bit of the hold mode register (HLDM) is set (to 1), the bus hold function is enabled. When the bus hold function is enabled, pins P66 and P67 operate as the HLDRQ and HLDAK pins respectively. The HLDRQ pin becomes high-impedance, and the HLDAK pin outputs a low-level signal.

If a high-level signal is input to the HLDRQ pin when the bus hold function is enabled, at the end of the access operation being executed the address bus (A8 to A19), address/data bus (AD0 to AD7), $\overline{RD}$, $\overline{WR}$, and ASTB pins are all set to high-impedance, the HLDAK pin output level is driven high, and the hold mode is established. At this time, it is recommended to connect a pull-up resistor to the $\overline{RD}$ and $\overline{WR}$ pins and a pull-down resistor to the ASTB pin because the address bus, address/data bus, $\overline{RD}$, $\overline{WR}$, and ASTB pins go into a high-impedance state.

While the HLDAK pin is high (in the hold mode) the μPD784038 does not use the local bus interface, and therefore an external DMA controller, etc. is free to access the memory.

When the HLDRQ pin input level changes from high to low, the hold mode is released, the HLDAK pin level changes from high to low, and then the μPD784038 resumes use of the local bus.

A transition to the hold mode is performed between bus cycles, and the instruction being executed may be suspended.

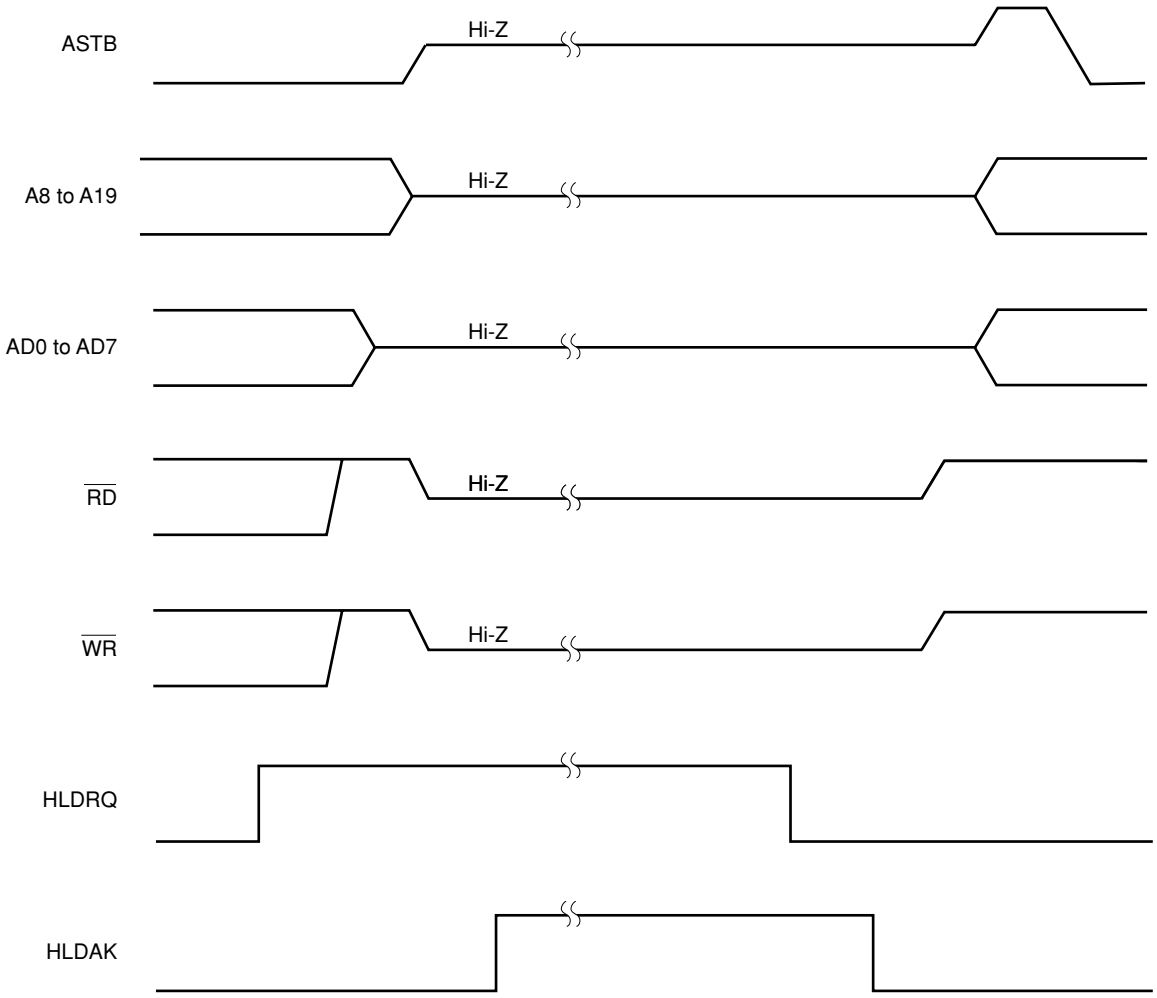
When a program is fetched from the internal memory, instructions can be executed until it comes to an instruction that uses the local bus interface.

Therefore, instruction execution is not stopped unless the external memory is accessed.

Also, if a transition to the hold mode is made during execution of an instruction that does not use the local bus interface when a program is fetched from the external memory, the μPD784038 continues execution of prefetched instructions until it comes to an instruction that uses the local bus interface, and suspends instruction execution when it comes to an instruction that uses the local bus interface, or when there are no more prefetched instructions. When the hold mode is released, execution of the suspended instruction is resumed from the point at which it was suspended.

When a program is fetched from the internal ROM or RAM, execution of instructions until it comes to an instruction that uses the local bus interface continues.

Figure 23-22 Hold Mode Timing



23.5 CAUTIONS

(1) When the bus hold function is used, the external wait function cannot be used (access wait control by means of the $\overline{\text{WAIT}}$ pin), and 0, 1 or 2 waits must be selected for the entire space.

(2) The refresh function cannot be used when the bus hold function is used. In this case, ensure that refreshing is specified as disabled.

(3) Do not set external wait to the internal ROM area. Otherwise, the CPU may be in the deadlock status which can be cleared only by reset input.

CHAPTER 24 STANDBY FUNCTION

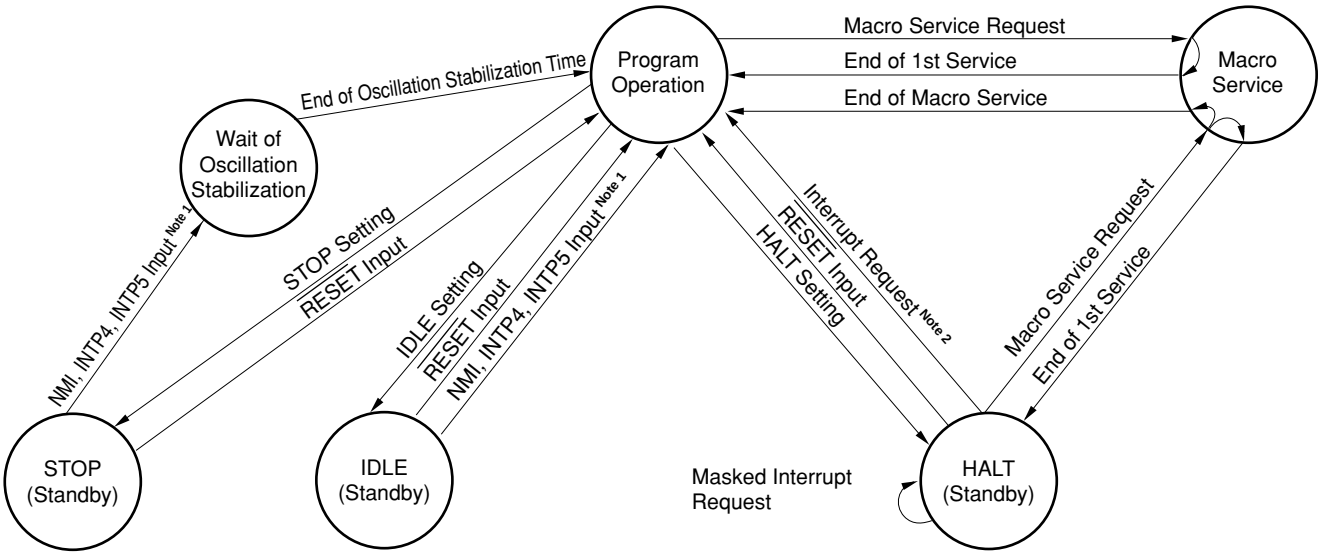
24.1 CONFIGURATION AND FUNCTION

The μPD784038 has a standby function that enables the system power consumption to be reduced. The standby function includes three modes as follows:

- HALT mode..... In this mode the CPU operating clock is stopped. Intermittent operation in combination with the normal operating mode enables the total system power consumption to be reduced.
- IDLE mode..... In this mode the oscillator continues operating while the entire remainder of the system is stopped. Normal program operation can be restored at a low power consumption close to that of the STOP mode and in a time equal to that of the HALT mode.
- STOP mode.....In this mode the oscillator is stopped and the entire system is stopped. Ultra-low power consumption can be achieved, consisting of leakage current only.

These modes are set by software. The standby mode (STOP/IDLE/HALT mode) transition diagram is shown in Figure 24-1, and the standby function block diagram in Figure 24-2.

Figure 24-1 Standby Mode Transition Diagram

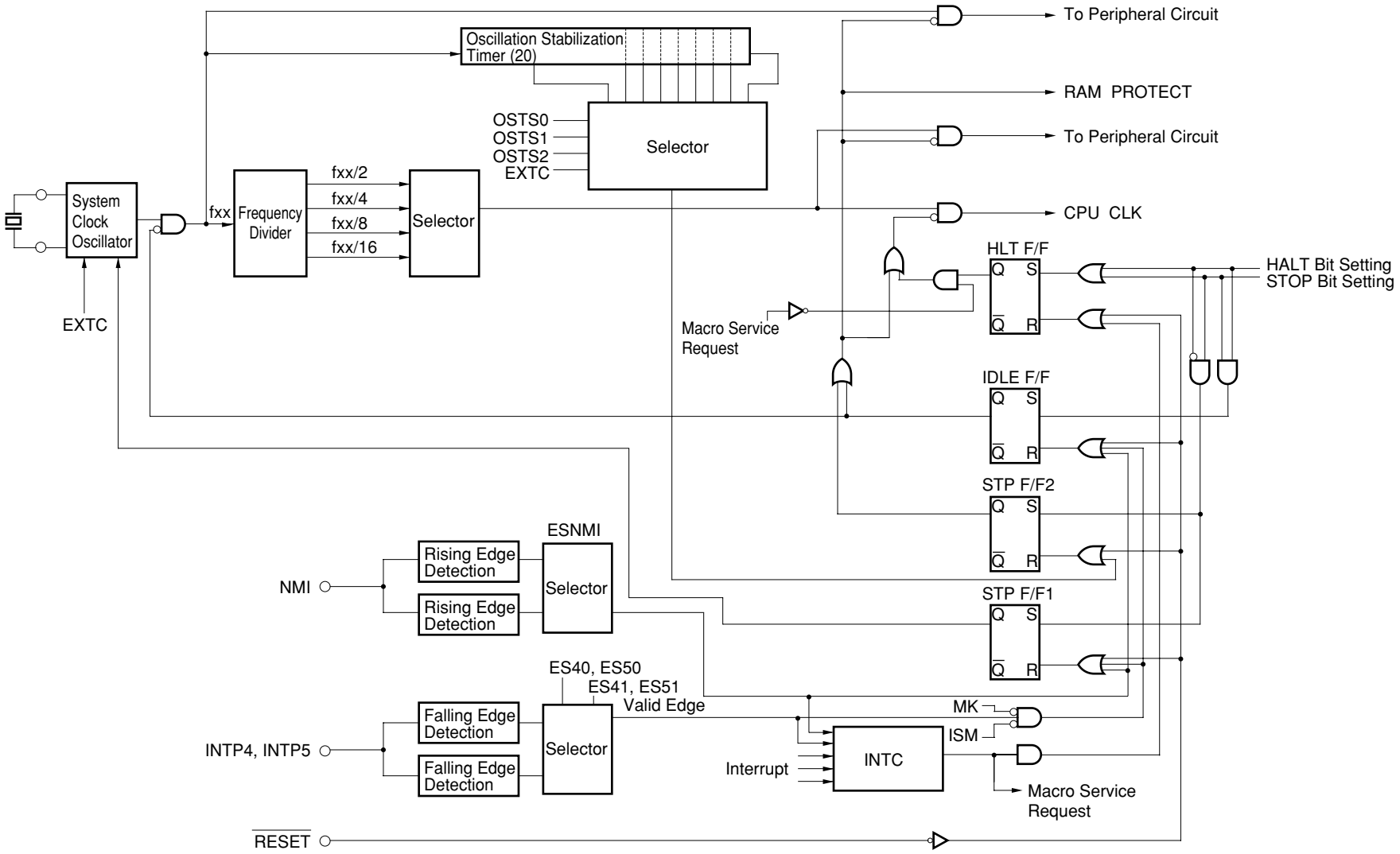


Notes 1. When INTP4 and INTP5 are not masked

2. Unmasked interrupt request only

Remark Only external input is valid as NMI. The watchdog timer must not be used to release the standby mode (STOP, HALT, or IDLE mode)

Figure 24-2 Standby Function Block Diagram



24.2 CONTROL REGISTERS

24.2.1 Standby Control Register (STBC)

The STBC is used to select the STOP mode setting and the internal system clock.

To prevent entry into standby mode due to an inadvertent program loop, the STBC register can only be written to with a dedicated instruction. This dedicated instruction, MOV STBC, #byte, has a special code configuration (4 bytes), and a write is only performed if the 3rd and 4th bytes of the operation code are mutual 1's complements.

If the 3rd and 4th bytes of the operation code are not mutual 1's complements, a write is not performed and an operand error interrupt is generated. In this case, the return address saved in the stack area is the address of the instruction that was the source of the error, and thus the address that was the source of the error can be identified from the return address saved in the stack area.

If recovery from an operand error is simply performed by means of an RETB instruction, an endless loop will result.

As an operand error interrupt is only generated in the event of an inadvertent program loop (with the NEC Electronics assembler, RA78K4, only the correct dedicated instruction is generated when MOV STBC, #byte is written), system initialization should be performed by the program.

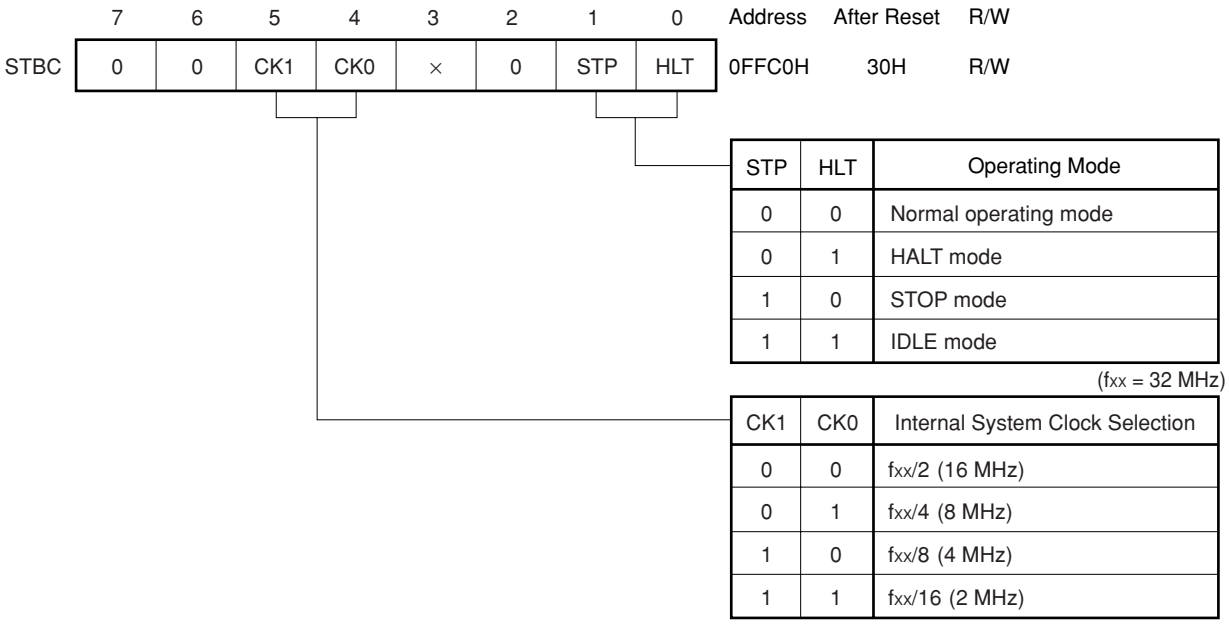
Other write instructions ("MOV STBC, A", "AND STBC, #byte", "SET1 STBC.7", etc.) are ignored and do not perform any operation. That is, a write is not performed to the STBC, and an interrupt such as an operand error interrupt is not generated.

The STBC can be read at any time by a data transfer instruction.

$\overline{\text{RESET}}$ input sets the STBC register to 30H.

The format of the STBC is shown in Figure 24-3.

Figure 24-3 Standby Control Register (STBC) Format



	7	6	5	4	3	2	1	0	Address	After Reset	R/W
STBC	0	0	CK1	CK0	×	0	STP	HLT	0FFC0H	30H	R/W

STP	HLT	Operating Mode
0	0	Normal operating mode
0	1	HALT mode
1	0	STOP mode
1	1	IDLE mode

(f_{XX} = 32 MHz)

CK1	CK0	Internal System Clock Selection
0	0	f_{XX}/2 (16 MHz)
0	1	f_{XX}/4 (8 MHz)
1	0	f_{XX}/8 (4 MHz)
1	1	f_{XX}/16 (2 MHz)

Cautions 1. If the STOP mode is used when using external clock input, the EXTC bit of the oscillation stabilization time specification register (OSTS) must be set (to 1) before setting STOP mode. If the STOP mode is used with the EXTC bit cleared (to 0) when using external clock input, the μPD784038 may suffer damage or reduced reliability.

When setting the EXTC bit of OSTS to 1, be sure to input a clock in phase reverse to that of the clock input to the X1 pin, to the X2 pin (refer to 4.3.1 Clock Generation Circuit).

2. Execute an NOP instruction three times after the standby instruction (after the standby mode has been released). Otherwise, the standby instruction cannot be executed if execution of the standby instruction and an interrupt request contend, and the interrupt is acknowledged after two or more instructions following the standby instruction have been executed. The instruction that is executed before acknowledging the interrupt is the one that is executed within up to 6 clocks after the standby instruction has been executed.

```
Example    MOV STBC, #byte
           NOP
           NOP
           NOP
           ⋮
```

24.2.2 Oscillation Stabilization Time Specification Register (OSTS)

The OSTS specifies the oscillator operation and the oscillation stabilization time when STOP mode is released. The EXTC bit of the OSTS specifies whether crystal/ceramic oscillation or an external clock is used. STOP mode can be set when external clock input is used only when the EXTC bit is set (to 1).

Bits OSTS0 to OSTS2 of the OSTS select the oscillation stabilization time when STOP mode is released. In general, an oscillation stabilization time of at least 40 ms should be selected when a crystal resonator is used, and at least 4 ms when a ceramic oscillator is used.

The time taken for oscillation stabilization is affected by the crystal resonator or ceramic resonator used, and the capacitance of the connected capacitor. Therefore, if you want to set a short oscillation stabilization time, you should consult the crystal resonator or ceramic resonator manufacturer.

The OSTS can be written to only with an 8-bit transfer instruction.

$\overline{\text{RESET}}$ input clears the OSTS register to 00H.

The format of the OSTS is shown in Figure 24-4.

Figure 24-4 Oscillation Stabilization Time Specification Register (OSTS) Format

	7	6	5	4	3	2	1	0	Address	After Reset	R/W
OSTS	EXTC	0	0	0	0	OSTS2	OSTS1	OSTS0	0FFCFH	00H	R/W

(fxx = 32 MHz)

EXTC	OSTS2	OSTS1	OSTS0	Oscillation Stabilization Time Selection Bits
0	0	0	0	2^{20}/fxx (32.80 ms)
0	0	0	1	2^{19}/fxx (16.40 ms)
0	0	1	0	2^{18}/fxx (8.19 ms)
0	0	1	1	2^{17}/fxx (4.10 ms)
0	1	0	0	2^{16}/fxx (2.05 ms)
0	1	0	1	2^{15}/fxx (1.02 ms)
0	1	1	0	2^{14}/fxx (512 μs)
0	1	1	1	2^{13}/fxx (256 μs)
1	×	×	×	512/fxx (16 μs)

EXTC	External Clock Selection
0	When crystal/ceramic oscillation is used
1	When external clock is used

Cautions 1. When crystal/ceramic oscillation is used, the EXTC bit of the oscillation stabilization time specification register (OSTS) must be cleared (to 0) before use. If the EXTC bit is set (to 1), oscillation will stop.

2. If the STOP mode is used when using external clock input, the EXTC bit must be set (to 1) before setting STOP mode. If the STOP mode is used with the EXTC bit cleared (to 0) the μPD784038 may suffer damage or reduced reliability.

When setting the EXTC bit of OSTS to 1, be sure to input a clock in phase reverse to that of the clock input to the X1 pin, to the X2 pin (refer to 4.3.1 Clock Generation Circuit).

24.3 HALT MODE

24.3.1 HALT Mode Setting and Operating States

The HALT mode is selected by setting (to 1) the HLT bit of the standby control (STBC) register.

The only writes that can be performed on the STBC are 8-bit data writes by means of a dedicated instruction. HALT mode setting is therefore performed by means of the “MOV STBC/#byte” instruction.

Write a NOP instruction three times after the instruction that sets the HALT mode (after releasing the HALT mode). Otherwise, two or more instructions may be executed before an interrupt is acknowledged. As a result, the execution sequence of the interrupt processing and instructions may be changed. To prevent troubles due to changes in the execution sequence, the above processing is necessary.

Caution If HALT mode setting is performed when a condition that releases HALT mode is in effect, HALT mode is not entered, and execution of the next instruction, or a branch to a vectored interrupt service program, is performed. To ensure that a definite HALT mode setting is made, interrupt requests should be cleared (to 0), etc. before entering HALT mode.

Table 24-1 Operating States in HALT Mode

Clock oscillator		Operating
Internal system clock		Operating
CPU		Operation stopped **Note**
I/O lines		Retain state prior to HALT mode setting
Peripheral functions		Continue operating
Internal RAM		Retained
Bus lines	AD0 to AD7	High-impedance
	A8 to A19	Retained
$\overline{RD}$, $\overline{WR}$ output		High level
ASTB output		Low level
$\overline{REFRQ}$ output		Continue operating
HLDRQ input		Continue operating (input)
HLDAK output		Continue operating

Note Macro service processing is executed.

24.3.2 HALT Mode Release

HALT mode can be released by the following three sources.

- Non-maskable interrupt request (NMI pin input only)
- Maskable interrupt request (vectored interrupt/context switching/macro service)
- $\overline{RESET}$ input

Release sources and an outline of operations after release are shown in Table 24-2. Figure 24-5 shows operations after HALT mode release.

Table 24-2 HALT Mode Release and Operations after Release

Release Source	MK Note 1	IE Note 2	State on Release	Operation after Release
$\overline{\text{RESET}}$ input	×	×	—	Normal reset operation
Non-maskable interrupt request (NMI pin input only, excluding watchdog timer) Note 6	×	×	• Non-maskable interrupt service program not being executed • Low-priority non-maskable interrupt service program being executed	Interrupt request acknowledgment
			• Service program for same request being executed • High-priority non-maskable interrupt service program being executed	Execution of instruction after MOV STBC/ #byte instruction (interrupt request that released HALT mode is held pending Note 3)
Maskable interrupt request (excluding macro service request)	0	1	• Interrupt service program not being executed • Low-priority maskable interrupt service program being executed • PRSL bit Note 4 cleared (to 0) during execution of priority level 3 interrupt service program	Interrupt request acknowledgment
			• Same-priority maskable interrupt service program being executed (If PRSL bit Note 4 is cleared (to 0), excluding execution of priority level 3 interrupt service program) • High-priority interrupt service program being executed	Execution of instruction after MOV STBC/ #byte instruction (interrupt request that released HALT mode is held pending Note 3)
	0	0	—	
	1	×	—	HALT mode maintained
Macro service request	0	×	—	Macro service processing execution End condition not established → HALT mode again End condition established → If VCIE Note 5 = 1: HALT mode again If VCIE Note 5 = 0: Same as release by maskable interrupt request
	1	×	—	HALT mode maintained

Notes 1. Interrupt mask bit in individual interrupt request source

2. Interrupt enable flag in program status word (PSW)

3. Pending interrupt requests are acknowledged when acknowledgment becomes possible.

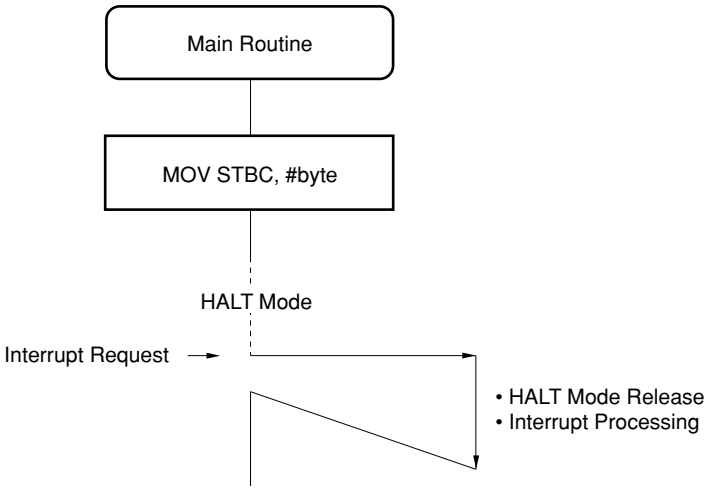
4. Bit in interrupt mode control register (IMC)

5. Bit in macro service mode register of macro service control word in individual macro service request source

6. The watchdog timer cannot be used to release the HALT mode.

Figure 24-5 Operation after HALT Mode Release (1/4)

(1) When interrupt generates after HALT mode has been set



(2) Reset after HALT mode has been set

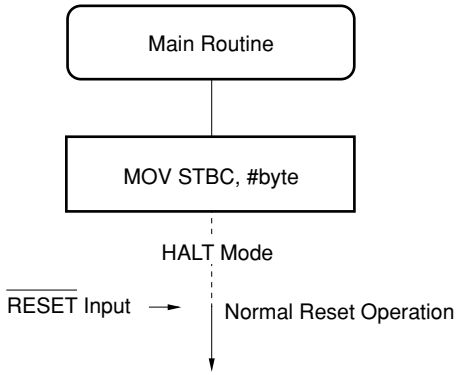
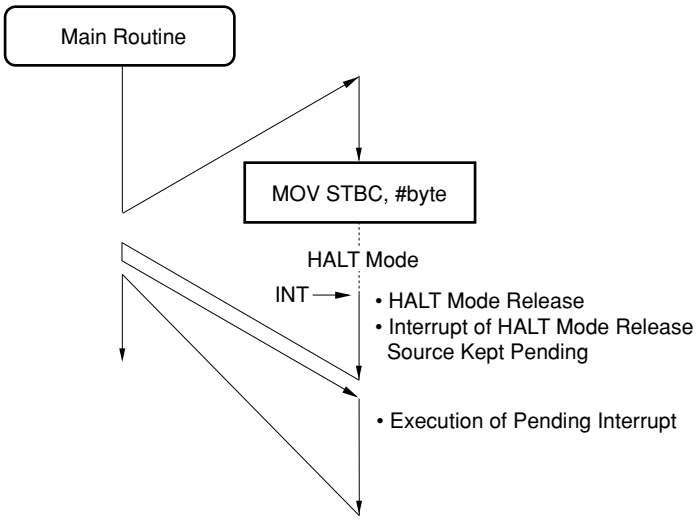


Figure 24-5 Operation after HALT Mode Release (2/4)

(3) When HALT mode is set while interrupt routine with priority higher than or same as that of interrupt of release source



(4) When HALT mode is set while interrupt routine with priority lower than that of interrupt of release source

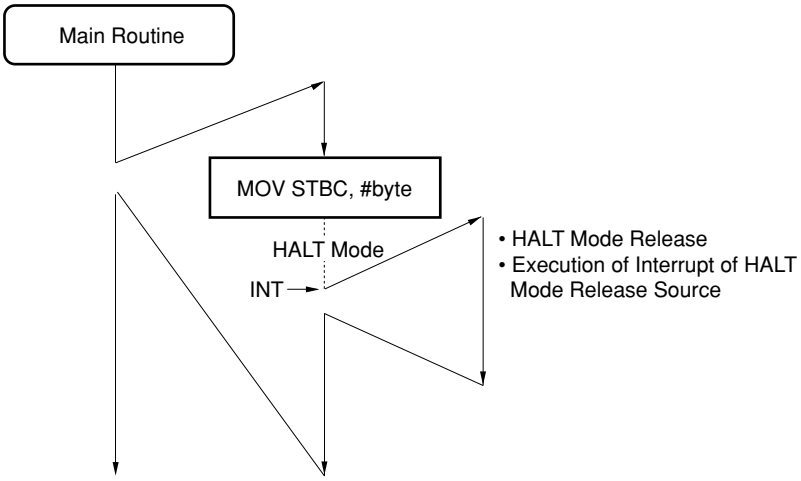
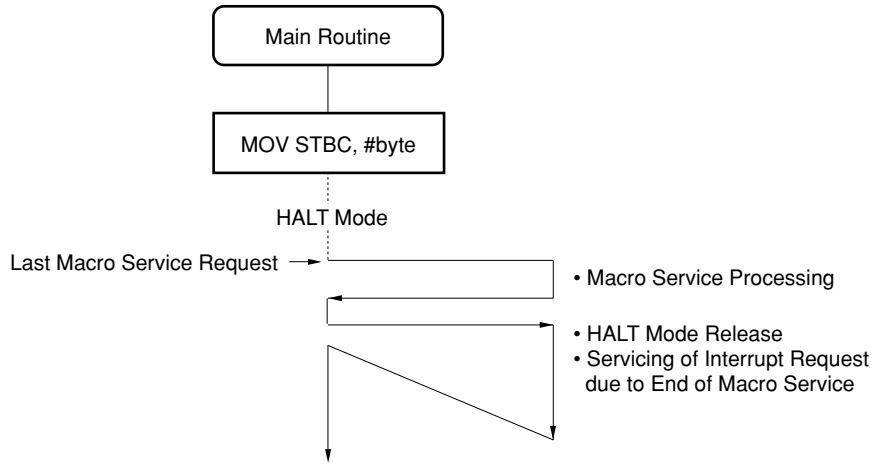


Figure 24-5 Operation after HALT Mode Release (3/4)

(5) When macro service request is generated in HALT mode

(a) When end condition of macro service is satisfied and interrupt request is generated immediately (VCIE = 0)



(b) When end condition of macro service is not satisfied, or if end condition of macro service is satisfied but interrupt request is not generated immediately (VCIE = 1)

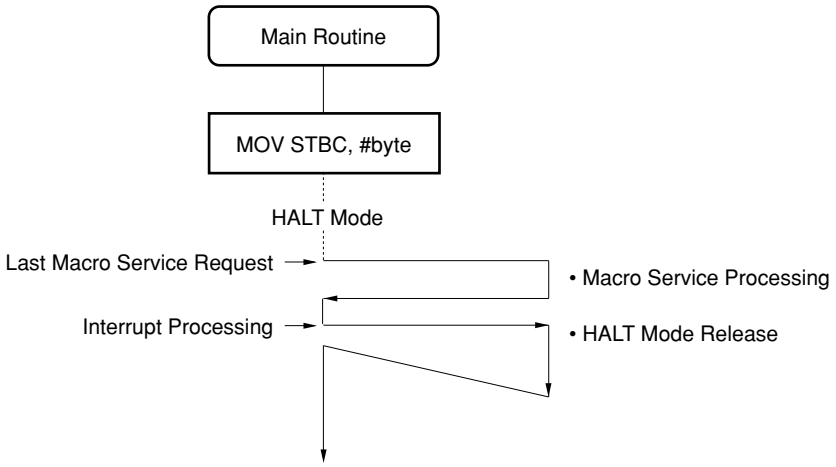
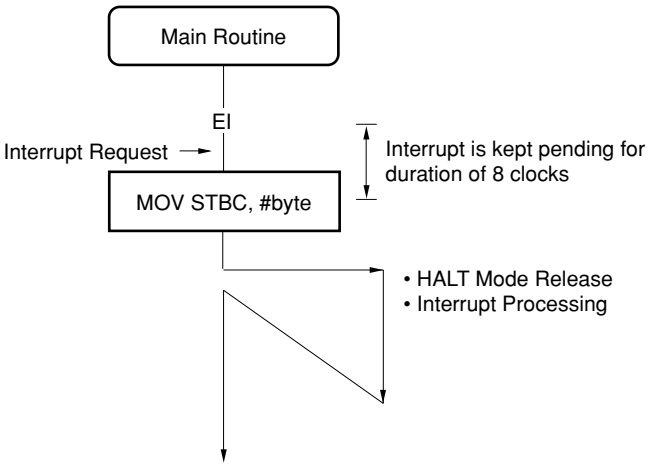
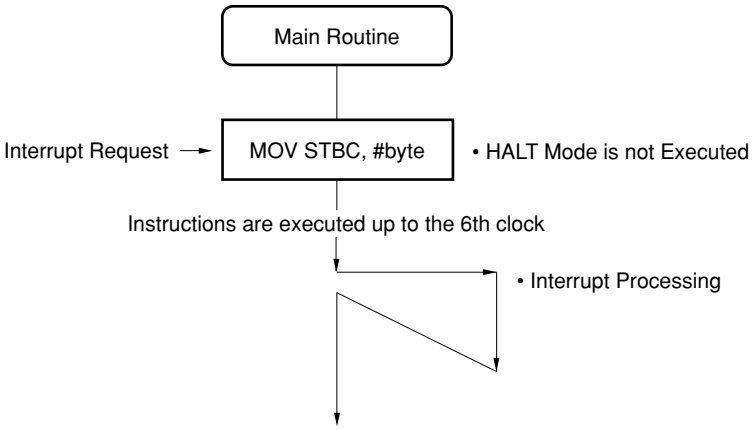


Figure 24-5 Operation after HALT Mode Release (4/4)

(6) When interrupt generates during execution of instruction that temporarily keeps interrupt pending, and if HALT mode is set while that interrupt is kept pending



(7) When HALT instruction and interrupt contend



(1) Release by non-maskable interrupt

When a non-maskable interrupt is generate, the μPD784038 is released from HALT mode irrespective of whether the interrupt acknowledgment enabled state (EI) or disabled state (DI) is in effect.

When the μPD784038 is released from HALT mode, if the non-maskable interrupt that released HALT mode can be acknowledged, acknowledgment of that non-maskable interrupt is performed and a branch is made to the service program. If the interrupt cannot be acknowledged, the instruction following the instruction that set the HALT mode (the MOV STBC/#byte instruction) is executed, and the non-maskable interrupt that released the HALT mode is acknowledged when acknowledgment becomes possible. See **22.6 NON-MASKABLE INTERRUPT ACKNOWLEDGMENT OPERATION** for details of non-maskable interrupt acknowledgment.

Caution The watchdog timer cannot be used to release the HALT mode.

(2) Release by maskable interrupt request

HALT mode release by a maskable interrupt request can only be performed by an interrupt for which the interrupt mask flag is 0.

When HALT mode is released, if an interrupt can be acknowledged when the interrupt request enable flag (IE) is set (to 1), a branch is made to the interrupt service program. If the interrupt cannot be acknowledged and if the IE flag is cleared (to 0), execution is resumed from the instruction following the instruction that set the HALT mode. See **22.7 MASKABLE INTERRUPT ACKNOWLEDGMENT OPERATION** for details of interrupt acknowledgment.

With macro service, HALT mode is released temporarily, service is performed once, then HALT mode is restored. When macro service has been performed the specified number of times, HALT mode is released if the VCIC bit in the macro service mode register of the macro service control word is cleared (to 0). The operation after release in this case is the same as for release by a maskable interrupt described earlier. If the VCIE bit is set (to 1), the HALT mode is entered again and is released by the next interrupt request.

Table 24-3 HALT Mode Release by Maskable Interrupt Request

Release Source	MK Note 1	IE Note 2	State on Release	Operation after Release
Maskable interrupt request (excluding macro service request)	0	1	• Interrupt service program not being executed • Low-priority maskable interrupt service program being executed • PRSL bit Note 4 cleared (to 0) during execution of priority level 3 interrupt service program	Interrupt request acknowledgment
			• Same-priority maskable interrupt service program being executed (If PRSL bit Note 4 is cleared (to 0), excluding execution of priority level 3 interrupt service program) • High-priority interrupt service program being executed	Execution of instruction after MOV STBC/ #byte instruction (interrupt request that released HALT mode is held pending Note 3)
	0	0	—	
	1	×	—	HALT mode maintained
Macro service request	0	×	—	Macro service processing execution End condition not established → HALT mode again End condition established → If VCIE Note 5 = 1: HALT mode again If VCIE Note 5 = 0: Same as release by maskable interrupt request
	1	×	—	HALT mode maintained

Notes 1. Interrupt mask bit in individual interrupt request source

2. Interrupt enable flag in program status word (PSW)

3. Pending interrupt requests are acknowledged when acknowledgment becomes possible.

4. Bit in interrupt mode control register (IMC)

5. Bit in macro service mode register of macro service control word in individual macro service request source

(3) Release by $\overline{\text{RESET}}$ input

The program is executed after branching to the reset vector address, as in a normal reset operation. However, internal RAM contents retain their value directly before HALT mode was set.

24.4 STOP MODE

24.4.1 STOP Mode Setting and Operating States

The STOP mode is selected by setting (to 1) the STP bit of the standby control register (STBC) register.

The only writes that can be performed on the STBC register are 8-bit data writes by means of a dedicated instruction. STOP mode setting is therefore performed by means of the “MOV STBC/#byte” instruction.

If interrupts are enabled (when the IE flag of PSW is set to 1), write a NOP instruction three times after the instruction that sets the STOP mode (after releasing the STOP mode). Otherwise, two or more instructions may be executed before an interrupt is acknowledged. As a result, the execution sequence of the interrupt processing and instructions may be changed. To prevent troubles due to changes in the execution sequence, the above processing is necessary.

Caution If the STOP mode is set when the condition to release the HALT mode is satisfied (refer to 24.3.2 HALT Mode Release), the STOP mode is not set, but the next instruction is executed or execution branches to a vectored interrupt service program. To accurately set the STOP mode, clear the interrupt request before setting the STOP mode.

Table 24-4 Operating States in STOP Mode

Clock oscillator		Oscillation stopped
Internal system clock		Stopped
CPU		Operation stopped
I/O lines		Retain state prior to STOP mode setting
Peripheral functions		All operation stopped **Note**
Internal RAM		Retained
Bus lines	AD0 to AD7	High-impedance
	A8 to A19	High-impedance
$\overline{RD}$, $\overline{WR}$ output		High-impedance
ASTB output		High-impedance
$\overline{REFRQ}$ output		Retained
HLDRQ input		High-impedance
HLDAK output		Low level

Note A/D converter operation is stopped, but if the CS bit of the A/D converter mode register (ADM) is set (to 1), the power consumption does not decrease. D/A converter operation is not stopped.

Cautions 1. When the STOP mode is used in a system that uses an external clock, the EXTC bit of the OSTS must be set (to 1). If STOP mode setting is performed in a system to which an external clock is input when the EXTC bit of the OSTS is cleared (to 0), the power consumption increases.
When setting the EXTC bit of OSTS to 1, be sure to input a clock in phase reverse to that of the clock input to the X1 pin, to the X2 pin (refer to 4.3.1 Clock Generation Circuit).
2. The CS bit of the A/D converter mode (ADM) register should be cleared (to 0).
3. D/A converter operation is not stopped simply by setting the STOP mode. In order to reduce the power consumption, the DACEn (n = 0, 1) bits of the D/A converter mode register (DAM) must both be cleared (to 0). When DACEn is cleared (to 0), the ANOn (n = 0, 1) pin output level becomes high-impedance.

24.4.2 STOP Mode Release

STOP mode is released by NMI input, INTP4 input, and $\overline{\text{RESET}}$ input.

Release sources and an outline of operations after release are shown in Table 24-5. Figure 24-6 shows operations after STOP mode release.

Table 24-5 STOP Mode Release and Operations after Release

Release Source	MK Note 1	ISM Note 2	IE Note 3	State after Release	Operation after Release
$\overline{\text{RESET}}$ input	×	×	×	—	Normal reset operation
NMI pin input	×	×	×	• Non-maskable interrupt service program not being executed • Low-priority non-maskable interrupt service program being executed	Interrupt request acknowledgment
				• NMI pin input service program being executed • High-priority non-maskable interrupt service program being executed	Execution of instruction after MOV STBC/#byte instruction (interrupt request that released STOP mode is held pending Note 4)
INTP4/INTP5 pin input	0	0	1	• Interrupt service program not being executed • Low-priority maskable interrupt service program being executed • PRSL bit Note 5 cleared (to 0) during execution of priority level 3 interrupt service program	Interrupt request acknowledgment
				• Same-priority maskable interrupt service program being executed (If PRSL bit Note 5 is cleared (to 0), excluding execution of priority level 3 interrupt service program) • High-priority interrupt service program being executed	Execution of instruction after MOV STBC/#byte instruction (interrupt request that released STOP mode is held pending Note 4)
	0	0	0	—	
	1	0	×	—	STOP mode maintained
	×	1	×		

Notes 1. Interrupt mask bit in individual interrupt request source

2. Macro service enable flag in individual interrupt request source

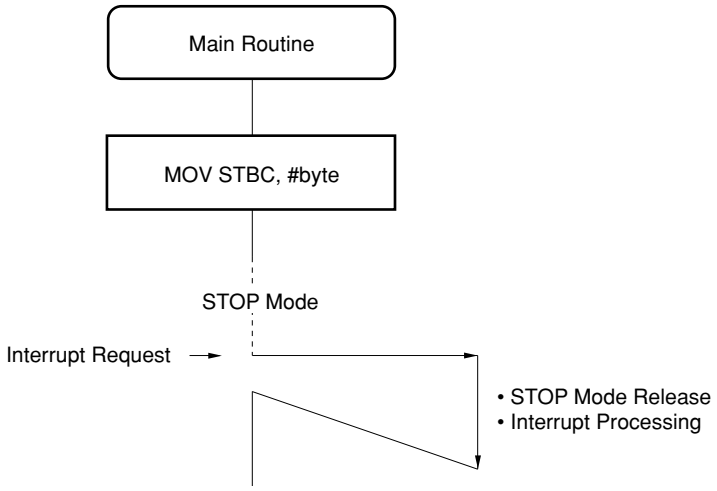
3. Interrupt enable flag in program status word (PSW)

4. Pending interrupt requests are acknowledged when acknowledgment becomes possible.

5. Bit in interrupt mode control register (IMC)

Figure 24-6 Operation after STOP Mode Release (1/2)

(1) When interrupt generates after STOP mode has been set



(2) Reset after STOP mode has been set

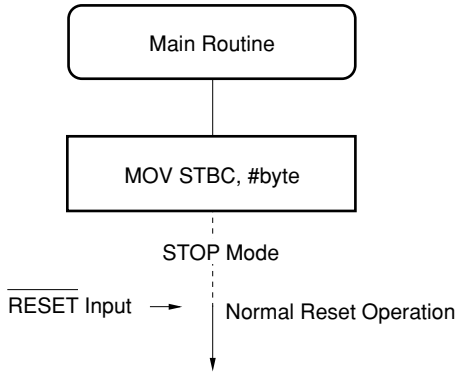
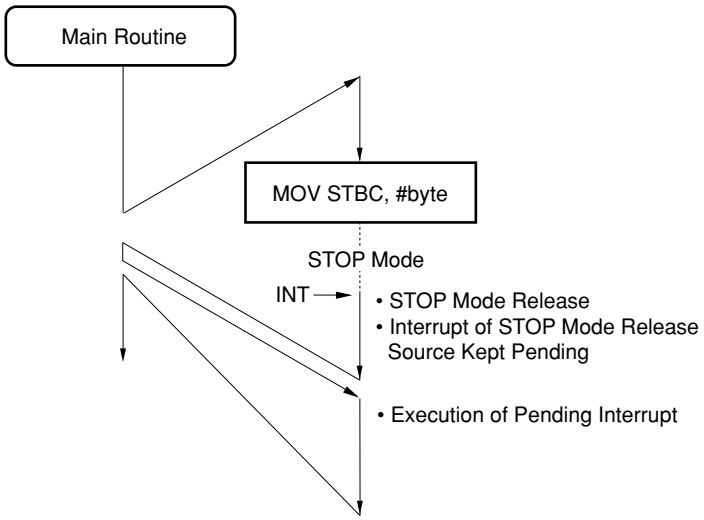
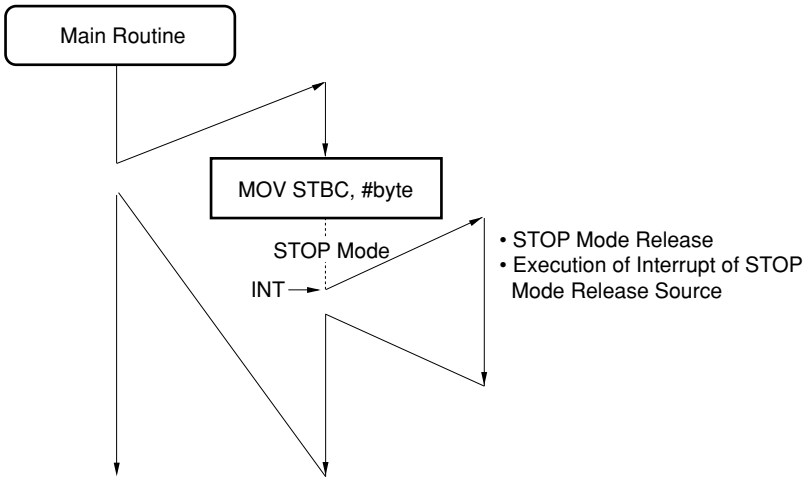


Figure 24-6 Operation after STOP Mode Release (2/2)

(3) When STOP mode is set while interrupt routine with priority higher than or same as that of interrupt of release source



(4) When STOP mode is set while interrupt routine with priority lower than that of interrupt of release source



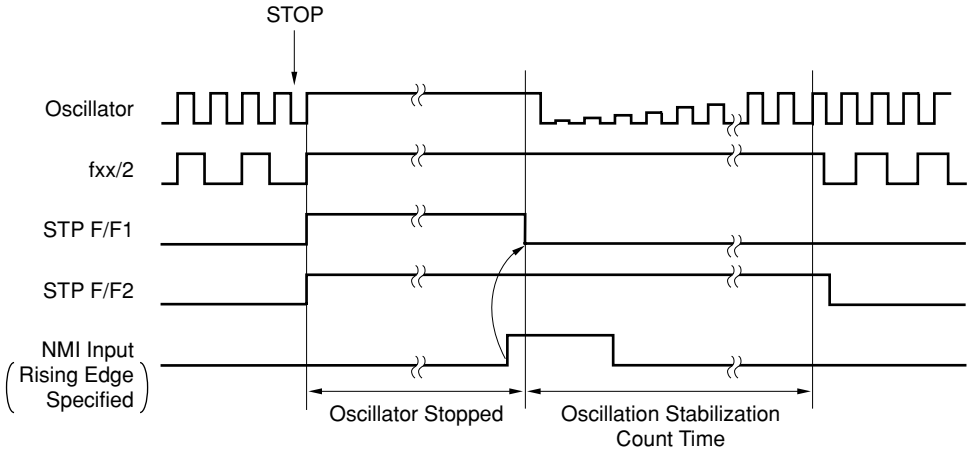
(1) STOP mode release by NMI input

The oscillator resumes oscillation when the valid edge specified by external interrupt mode register 0 (INTM0) is input to the NMI input. STOP mode is released after the oscillation stabilization time specified by the oscillation stabilization time specification register (OSTS) elapses.

When the μPD784038 is released from STOP mode, if a non-maskable interrupt by NMI pin input can be acknowledged, a branch is made to the NMI interrupt service program. If the interrupt cannot be acknowledged (if the STOP mode is set in an NMI interrupt service program, etc.), execution is resumed from the instruction following the instruction that set the STOP mode, and a branch is made to the NMI interrupt service program when acknowledgment becomes possible (by execution of an RETI instruction, etc.).

See **22.6 NON-MASKABLE INTERRUPT ACKNOWLEDGMENT OPERATION** for details of NMI interrupt acknowledgment.

Figure 24-7 STOP Mode Release by NMI Input



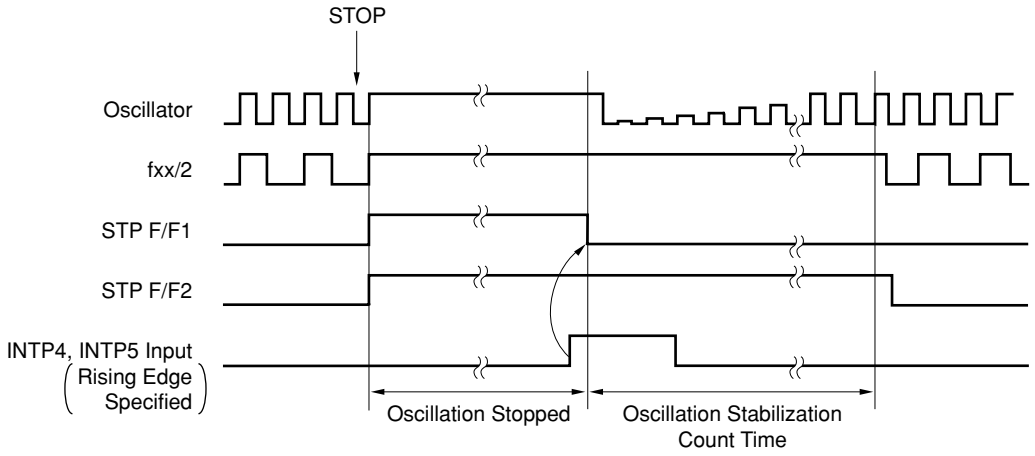
(2) STOP mode release by INTP4 or INTP5 input

When masking of interrupts by INTP4 and INTP5 input is released and macro service is disabled, the oscillator resumes oscillation when the valid edge specified by external interrupt mode register 1 (INTM1) is input to the INTP4 or INTP5 input. Following this, STOP mode is released after the oscillation stabilization time specified by the oscillation stabilization time specification register (OSTS) elapses.

When the μPD784038 is released from STOP mode, if an interrupt can be acknowledged when the interrupt enable flag (IE) is set (to 1), a branch is made to the interrupt service program. If the interrupt cannot be acknowledged and if the IE flag is cleared (to 0), execution is resumed from the instruction following the instruction that set the STOP mode.

See **22.7 MASKABLE INTERRUPT ACKNOWLEDGMENT OPERATION** for details of interrupt acknowledgment.

Figure 24-8 STOP Mode Release by INTP4/INTP5 Input



(3) STOP mode release by $\overline{\text{RESET}}$ input

When $\overline{\text{RESET}}$ input falls from high to low and the reset state is established, the oscillator resumes oscillation. The oscillation stabilization time should be secured while $\overline{\text{RESET}}$ is active. Thereafter, normal operation is started when $\overline{\text{RESET}}$ rises.

Unlike an ordinary reset operation, data memory retains its contents prior to STOP mode setting.

24.5 IDLE MODE

24.5.1 IDLE Mode Setting and Operating States

The IDLE mode is selected by setting (to 1) both the STP bit and the HLT bit of the standby control (STBC) register.

The only writes that can be performed on the STBC are 8-bit data writes by means of a dedicated instruction. IDLE mode setting is therefore performed by means of the ”MOV STBC/#byte” instruction.

Write a NOP instruction three times after the instruction that sets the IDLE mode (after releasing the IDLE mode). Otherwise, two or more instructions may be executed before an interrupt is acknowledged. As a result, the execution sequence of the interrupt processing and instructions may be changed. To prevent troubles due to changes in the execution sequence, the above processing is necessary.

Caution If the IDLE mode is set when the condition to release the HALT mode is satisfied (refer to 24.3.2 HALT Mode Release), the IDLE mode is not set, but the next instruction is executed or execution branches to a vectored interrupt service program. To accurately set the IDLE mode, clear the interrupt request before setting the IDLE mode.

Table 24-6 Operating States in IDLE Mode

Clock oscillator		Oscillation stopped
Internal system clock		Stopped
CPU		Operation stopped
I/O lines		Retain state prior to IDLE mode setting
Peripheral functions		All operation stopped **Note**
Internal RAM		Retained
Bus lines	AD0 to AD7	High-impedance
	A8 to A19	High-impedance
$\overline{RD}$, $\overline{WR}$ output		High-impedance
ASTB output		High-impedance
$\overline{REFRQ}$ output		Retained
HLDRQ input		High-impedance
HLDAK output		Low level

Note A/D converter operation is stopped, but if the CS bit of the A/D converter mode register (ADM) is set, the power consumption does not decrease. D/A converter operation is not stopped.

Cautions 1. The CS bit of the A/D converter mode (ADM) register should be reset.

2. D/A converter operation is not stopped simply by setting the IDLE mode. In order to reduce the power consumption, the DACEn (n = 0, 1) bits of the D/A converter mode register (DAM) must both be cleared (to 0). When DACEn is cleared (to 0), the ANOn (n = 0, 1) pin output level becomes high-impedance.

24.5.2 IDLE Mode Release

IDLE mode is released by NMI input, INTP4 input, INTP5 input, or $\overline{\text{RESET}}$ input.

Release source and an outline of operations after release are shown in Table 24-7. Figure 24-9 shows operations after IDLE mode release.

Table 24-7 IDLE Mode Release and Operations after Release

Release Source	MK Note 1	ISM Note 2	IE Note 3	State after Release	Operation after Release
$\overline{\text{RESET}}$ input	×	×	×	—	Normal reset operation
NMI pin input	×	×	×	• Non-maskable interrupt service program not being executed • Low-priority non-maskable interrupt service program being executed	Interrupt request acknowledgment
				• NMI pin input service program being executed • High-priority non-maskable interrupt service program being executed	Execution of instruction after MOV STBC/#byte instruction (interrupt request that released IDLE mode is held pending Note 4)
INTP4/INTP5 pin input	0	0	1	• Interrupt service program not being executed • Low-priority maskable interrupt service program being executed • PRSL bit Note 5 cleared (to 0) during execution of priority level 3 interrupt service program	Interrupt request acknowledgment
				• Same-priority maskable interrupt service program being executed (If PRSL bit Note 5 is cleared (to 0), excluding execution of priority level 3 interrupt service program) • High-priority interrupt service program being executed	Execution of instruction after MOV STBC/#byte instruction (interrupt request that released IDLE mode is held pending Note 4)
	0	0	0	—	
	1	0	×	—	IDLE mode maintained
	×	1	×		

Notes 1. Interrupt mask bit in individual interrupt request source

2. Macro service enable flag in individual interrupt request source

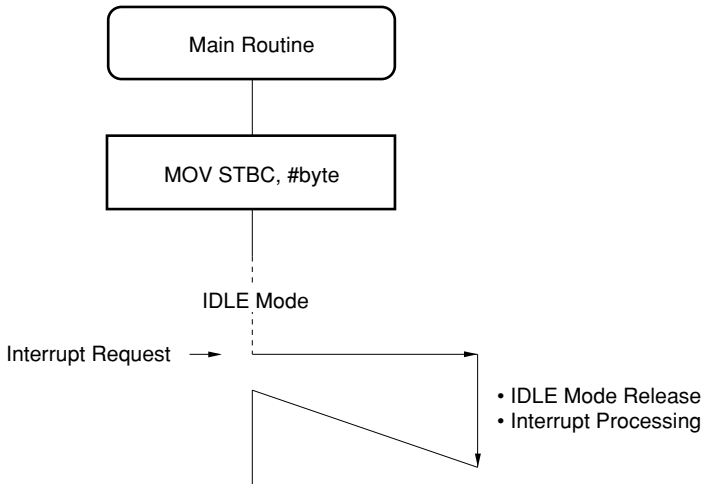
3. Interrupt enable flag in program status word (PSW)

4. Pending interrupt requests are acknowledged when acknowledgment becomes possible.

5. Bit in interrupt mode control register (IMC)

Figure 24-9 Operation after IDLE Mode Release (1/2)

(1) When interrupt generates after IDLE mode has been set



(2) Reset after IDLE mode has been set

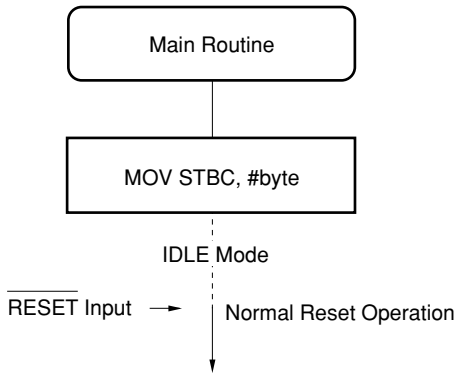
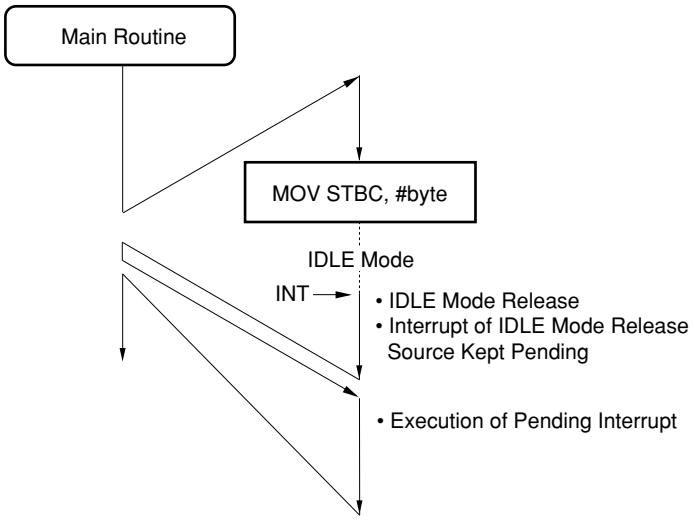
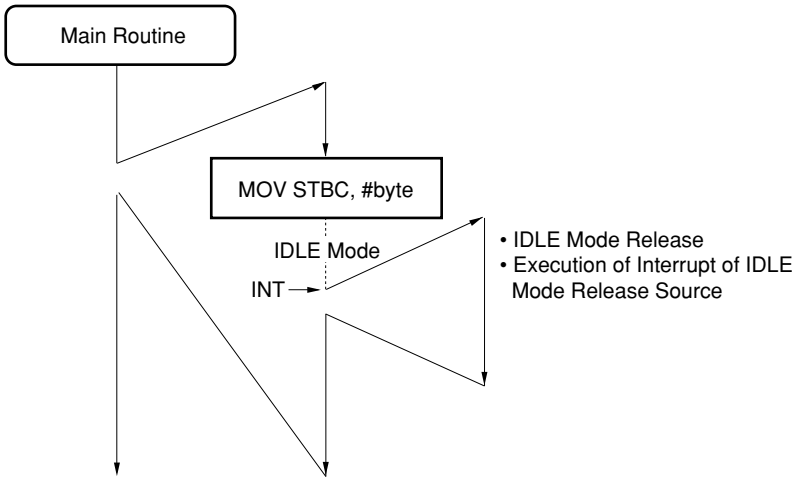


Figure 24-9 Operation after IDLE Mode Release (2/2)

(3) When IDLE mode is set while interrupt routine with priority higher than or same as that of interrupt of release source



(4) When IDLE mode is set while interrupt routine with priority lower than that of interrupt of release source



(1) IDLE mode release by NMI input

IDLE mode is released when the valid edge specified by external interrupt mode register 0 (INTM0) is input to the NMI input.

When the μPD784038 is released from IDLE mode, if a non-maskable interrupt by NMI pin input can be acknowledged, a branch is made to the NMI interrupt service program. If the interrupt cannot be acknowledged (if the IDLE mode is set in an NMI interrupt service program, etc.), execution is resumed from the instruction following the instruction that set the IDLE mode, and a branch is made to the NMI interrupt service program when acknowledgment becomes possible (by execution of an RETI instruction, etc.).

See **22.6 NON-MASKABLE INTERRUPT ACKNOWLEDGMENT OPERATION** for details of NMI interrupt acknowledgment.

(2) IDLE mode release by INTP4 or INTP5 input

When masking of interrupts by INTP4 and INTP5 input is released and macro service is disabled, IDLE mode is released when the valid edge specified by external interrupt mode register 1 (INTM1) is input to the INTP4 or INTP5 input.

When the μPD784038 is released from IDLE mode, if an interrupt can be acknowledged when the interrupt enable flag (IE) is set (to 1), a branch is made to the interrupt service program. If the interrupt cannot be acknowledged and if the IE flag is cleared (to 0), execution is resumed from the instruction following the instruction that set the IDLE mode.

See **22.7 MASKABLE INTERRUPT ACKNOWLEDGMENT OPERATION** for details of interrupt acknowledgment.

24.6 CHECK ITEMS WHEN STOP MODE/IDLE MODE IS USED

Check items required to reduce the power consumption when STOP mode/IDLE mode is used are shown below.

(1) Is the output level of each output pin appropriate?

The appropriate output level for each pin varies according to the next-stage circuit. You should select the output level that minimizes the power consumption.

- If high level is output when the input impedance of the next-stage circuit is low, a current will flow from the power supply to the port, resulting in an increased power consumption. This applies when the next-stage circuit is a CMOS IC, etc. When the power supply is off, the input impedance of a CMOS IC is low. In order to suppress the power consumption, or to prevent an adverse effect on the reliability of the CMOS IC, low level should be output. If a high level is output, latchup may result when power is turned on again.

- Depending on the next-stage circuit, inputting low level may increase the power consumption. In this case, high-level or high-impedance output should be used to reduce the power consumption.

- If the next-stage circuit is a CMOS IC, the power consumption of the CMOS IC may increase if the output is made high-impedance when power is supplied to it (the CMOS IC may also be overheated and damaged). In this case you should output an appropriate level, or pull the output high or low with a resistor.

The method of setting the output level depends on the port mode.

- When a port is in control mode, the output level is determined by the status of the on-chip hardware, and therefore the on-chip hardware status must be taken into consideration when setting the output level.

- In port mode, the output level can be set by writing to the port output latch and port mode register by software.

When a port is in control mode, its output level can be set easily by changing to port mode.

(2) Is the input pin level appropriate?

The voltage level input to each pin should be in the range between V_{SS} potential and V_{DD} potential. If a voltage outside this range is applied, the power consumption will increase and the reliability of the μPD784038 may be adversely affected.

Also ensure that an intermediate potential is not applied.

(3) Are pull-up resistors necessary?

An unnecessary pull-up resistor will increase the power consumption and cause a latchup of other devices. A mode should be specified in which pull-up resistors are used only for parts that require them.

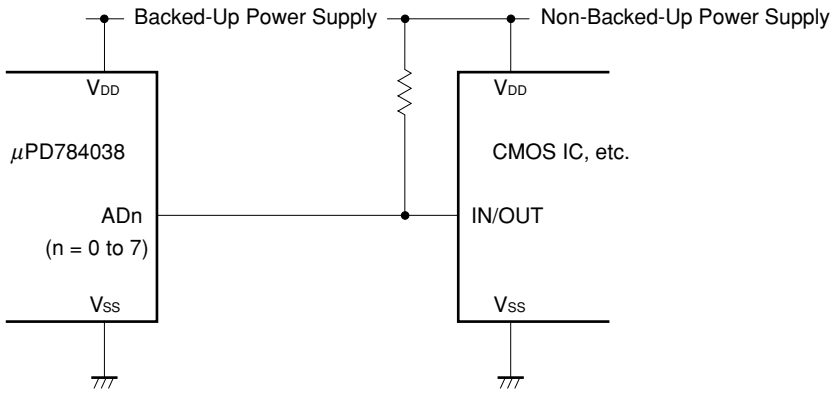
If there is a mixture of parts that do and do not require pull-up resistors, for parts that do, you should connect a pull-up resistor externally and specify a mode in which the on-chip pull-up resistor is not used.

(4) Is processing of the address bus, address/data bus, etc., appropriate?

In STOP mode and IDLE mode, the address bus, address/data bus, $\overline{RD}$ and $\overline{WR}$ pins become high-impedance. Normally, these pins are pulled high with a pull-up resistor. If this pull-up resistor is connected to the backed-up power supply, then if the input impedance of circuitry connected to the non-backed-up power supply is low, a current will flow through the pull-up resistor, and the power consumption will increase. Therefore, the pull-up resistor should be connected to the non-backed-up power supply side as shown in Figure 24-10.

Also, in STOP mode and IDLE mode the ASTB pin also becomes high impedance, and the $\overline{REFRQ}$/HLDAK pin adopts a fixed level. Countermeasures should be taken with reference to the points noted in (1).

Figure 24-10 Example of Address/Data Bus Processing



The voltage level input to the $\overline{WAIT}$/HLDRQ pin should be in the range between V_{SS} potential and V_{DD} potential. If a voltage outside this range is applied, the power consumption will increase and the reliability of the μPD784038 may be adversely affected.

★ **(5) A/D converter**

The current flowing to the AV_{DD}, AV_{REF1} pins can be reduced by clearing (to 0) the CS bit (bit 7) of the A/D converter mode register (ADM).

Make sure that the AV_{DD} pin is not at the same potential as the V_{DD} pin. Unless power is supplied to the AV_{DD} pin in the STOP mode, not only does the power consumption increase, but the reliability is also affected.

(6) D/A converter

In the STOP mode and IDLE mode the D/A converter still consumes a certain power. Clearing (to 0) the both DACEn (n = 0, 1) bits of the D/A converter mode register (DAM) sets the ANOn (n = 0/1) output to high impedance, enabling the power consumption to be reduced. (power consumption is not reduced if only one of the DACEn bits is cleared to 0).

The power consumption at resistor string can be eliminated by setting the voltage input to the AV_{REF2} pin to the same potential as AV_{REF3}. The ANOn output when the DACEn bit of the DAM is set (to 1) will be at the same potential as AV_{REF3}, and therefore the AV_{REF3} pin voltage should be set so as to minimize the power consumption of the next-stage circuit.

The power consumption of the μPD784038 can be minimized by clearing both the DACEn bits of DAM to 0. However, the output of the ANOn pin goes into a high-impedance state.

Also, a voltage should not be applied to the ANOn pins from off-chip, as this may result in an increase in the power consumption, and the μPD784038 may suffer damage or reduced reliability.

24.7 CAUTIONS

(1) If HALT/STOP/IDLE mode (standby mode hereafter) setting is performed when a condition that release HALT mode (refer to **24.3.2 HALT Mode Release**) is satisfied, standby mode is not entered, and execution of the next instruction, or a branch to a vectored interrupt service program, is performed. To ensure that a definite standby mode setting is made, interrupt requests should be cleared, etc. before entering standby mode.

(2) When crystal/ceramic oscillation is used, the EXTC bit must be cleared (to 0) before use. If the EXTC bit is set (to 1), oscillation will stop.

(3) When the STOP mode is used in a system that uses an external clock, the EXTC bit of the OSTS must be set (to 1). If STOP mode setting is performed in a system to which an external clock is input when the EXTC bit of the OSTS is cleared (to 0), the power consumption increases.
When setting the EXTC bit of OSTS to 1, be sure to input a clock in phase reverse to that of the clock input to the X1 pin, to the X2 pin (refer to **4.3.1 Clock Generation Circuit**).

(4) In STOP mode and IDLE mode, the CS bit of the A/D converter mode ADM register should be cleared (to 0).

(5) D/A converter operation is not stopped simply by setting the STOP mode or IDLE mode. In order to reduce the power consumption, the DACEn (n = 0, 1) bits of the D/A converter mode register (DAM) must both be cleared (to 0). When DACEn is cleared (to 0), the ANOn (n = 0, 1) pin output level becomes high-impedance.

(6) Execute an NOP instruction three times after the standby instruction (after the standby mode has been released). Otherwise, the standby instruction cannot be executed if execution of the standby instruction and an interrupt request contend, and the interrupt is acknowledged after two or more instructions following the standby instruction have been executed. The instruction that is executed before acknowledging the interrupt is the one that is executed within up to 6 clocks after the standby instruction has been executed.

```
Example   MOV STBC, #byte
          NOP
          NOP
          NOP
            ⋮
```

CHAPTER 25 RESET FUNCTION

25.1 RESET FUNCTION

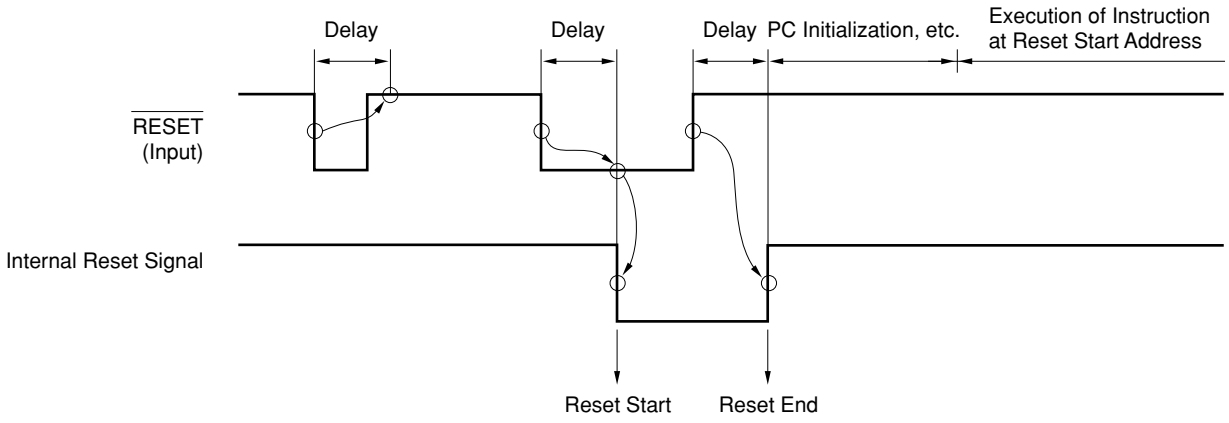
When low level is input to the $\overline{\text{RESET}}$ input pin, a system reset is affected, the various hardware units are set to the states shown in Table 25-2, and all pins except the power supply pins and the X1 and X2 pins are placed in the high-impedance state. Table 25-1 shows the pin statuses on reset and after reset release.

When the $\overline{\text{RESET}}$ input changes from low to high level, the reset state is released, the contents of address 00000H of the reset vector table are set in bits 0 to 7 of the program counter (PC), the contents of address 00001H in bits 8 to 15, and 0000B in bits 16 to 19, a branch is made, and program execution is started at the branch destination address. A reset start can therefore be performed from any address in the base area.

The contents of the various registers should be initialized as required in the program in the base area.

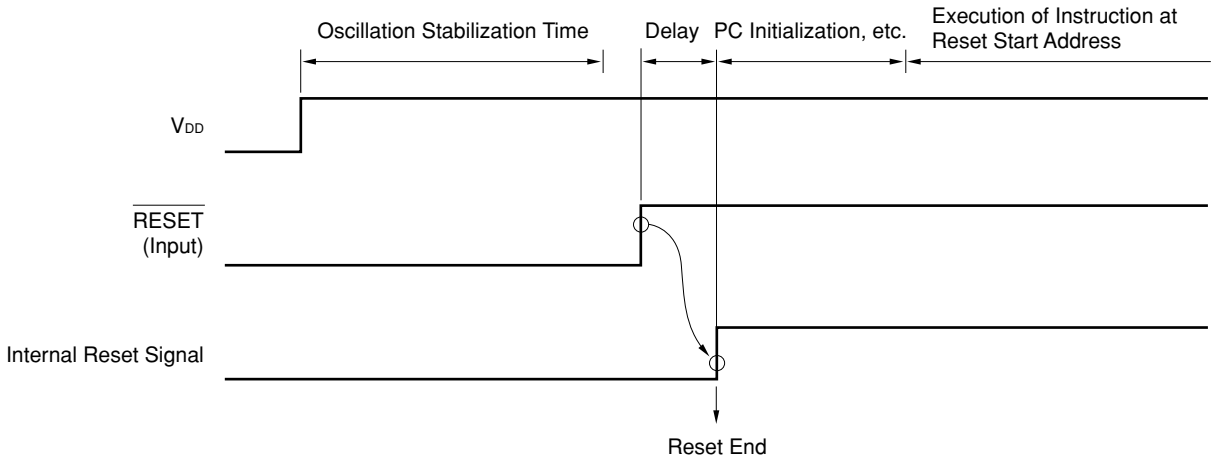
To prevent from malfunction due to noise, the $\overline{\text{RESET}}$ input pin incorporates an analog delay noise elimination circuit (see **Figure 25-1**).

Figure 25-1 Reset Signal Acknowledgment



In a reset operation upon powering on, the $\overline{\text{RESET}}$ signal must be kept active until the oscillation stabilization time has elapsed (approx. 40 ms, depending on the resonator used).

Figure 25-2 Power-On Reset Operation



Remark f_{CLK}: Internal system clock frequency

Table 25-1 Pin Statuses During Reset Input and After Reset Release

Pin Name	Input/Output	On Reset	Directly After Reset Release
P00 to P07	Input/output	Hi-Z	Hi-Z (input port mode)
P10/PWM0 to P17	Input/output	Hi-Z	Hi-Z (input port mode)
P20/NMI to P27/SI	Input	Hi-Z	Hi-Z (input port)
P30/RxD to P37/TO3	Input/output	Hi-Z	Hi-Z (input port mode)
P40/AD0 to P47/AD7	Input/output	Hi-Z	Hi-Z (input port mode) **Note 1**
P50/A8 to P57/A15	Input/output	Hi-Z	Hi-Z (input port mode) **Note 1**
P60/A16 to P63/A19 **Note 2**	Input/output	Hi-Z	Hi-Z (input port mode) **Note 1**
P64/$\overline{\text{RD}}$, P65/$\overline{\text{WR}}$	Input/output	Hi-Z	Hi-Z (input port mode) **Note 1**
P66/$\overline{\text{WAIT}}$, P67/$\overline{\text{REFRQ}}$	Input/output	Hi-Z	Hi-Z (input port mode)
P70/ANI0 to P77/ANI7	Input/output	Hi-Z	Hi-Z (input port mode)
ASTB/CLKOUT	Output	Hi-Z	0
ANO0, ANO1	Output	Hi-Z	Outputs AV_{REF3} pin input voltage

Notes 1. With the μPD784031, these pins function as the address/data bus pins, and output signal to fetch the reset vector address from address 0000H (refer to Figure 25-3 (a)).

2. With the μPD784031, these pins function only as the output port pins, and output 0 after reset release.

Table 25-2 Hardware States After Reset (1/2)

Hardware			State After Reset
Program counter (PC)			Set with contents of reset vector table (0000H/0001H).
Stack pointer (SP)			Undefined Note 1
Program status word (PSW)			02H
On-chip RAM	Data memory		Undefined Note 1
	General-purpose registers		
Ports	Ports 0, 1, 2, 3, 4, 5, 6 Note 2, 7		Undefined (high impedance)
Port mode registers	PM0, 1, 3, 4, 5, 6 Note 3, 7		FFH
Port mode control registers (PMC1, PMC3)			00H
Pull-up resistor option register (PUO)			00H
Real-time output port control register (RTPC)			00H
Timer/counter	Timer registers (TM0, TM1W, TM2W, TM3W)		0000H
	Compare registers (CR00, CR01, CR10LW, CR20W, CR30W)		Undefined
	Capture registers (CR02, CR12W, CR22W)		
	Capture/compare registers (CR11W, CR21W)		
	Timer control registers (TMC0, TMC1)		00H
	Timer output control register (TOC)		
	Capture/compare control registers	CRC0	10H
		CRC1, CRC2	00H
	Prescaler mode registers (PRM0, PRM1)		00H
	One-shot pulse output control register (OSPC)		00H
PWM	PWM control register (PWMC)		05H
	PWM prescaler register (PWPR)		00H
	PWM modulo registers (PWM0, PWM1)		Undefined
A/D converter	A/D converter mode register (ADM)		00H
	A/D conversion result register (ADCR)		Undefined
D/A converter	D/A converter mode register (DAM)		03H
	D/A conversion value setting registers (DACS0, DACS1)		00H

Notes 1. When HALT mode, STOP mode or IDLE mode is released by $\overline{\text{RESET}}$ input, the value before that mode was set is retained.

2. μPD784031: x0H

3. μPD784031: FxH

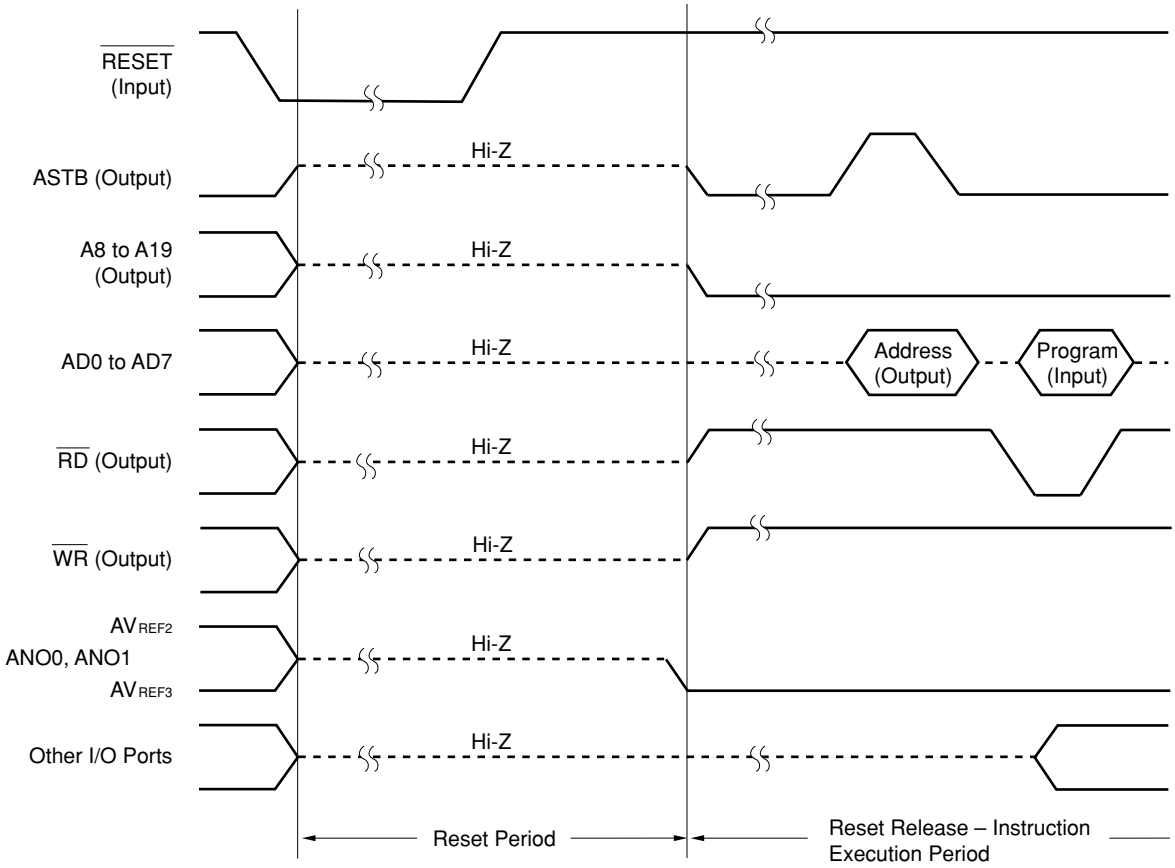
Table 25-2 Hardware States After Reset (2/2)

Hardware			State After Reset
Serial interface	Clocked serial interface mode registers (CSIM, CSIM1, CSIM2)		00H
	Shift registers (SIO, SIO1, SIO2)		Undefined
	Asynchronous serial interface mode registers (ASIM, ASIM2)		00H
	Asynchronous serial interface status registers (ASIS, ASIS2)		00H
	I^2C bus control register (IICC)		00H
	Serial receive buffers (RXB, RXB2)		Undefined
	Serial transmit shift registers (TXS, TXS2)		Undefined
	Baud rate generator control registers (BRGC, BRGC2)		00H
	Prescaler mode register for serial clock (SPRM)		04H
	Slave address register (SVA) **Note**		01H
Clock output function (CLOM)			00H
Memory extension mode register (MM)			20H
Programmable wait control registers		PWC1	AAH
		PWC2	AAAAH
Refresh function	Refresh mode register (RFM)		00H
	Refresh area specification register (RFA)		00H
Hold mode register (HLDM)			00H
Interrupts	Interrupt control registers (PIC0, PIC1, PIC2, PIC3, PIC4, PIC5, CIC00, CIC01, CIC10, CIC11, CIC20, CIC21, CIC30, ADIC, SERIC, SRIC, STIC, SERIC2, SRIC2, STIC2, CSIIC, CSIIC1, CSIIC2, SPCIC **Note**)		43H
	Interrupt mask registers	MK0	FFFFH
		MK1L	FFH
	In-service priority register (ISPR)		00H
	Interrupt mode control register (IMC)		00H
External interrupt mode registers (INTM0, INTM1)			00H
Sampling clock selection register (SCS0)			00H
Standby control register (STBC)			30H
Oscillation stabilization time specification register (OSTS)			00H
Internal memory size switching register (IMS)			FFH

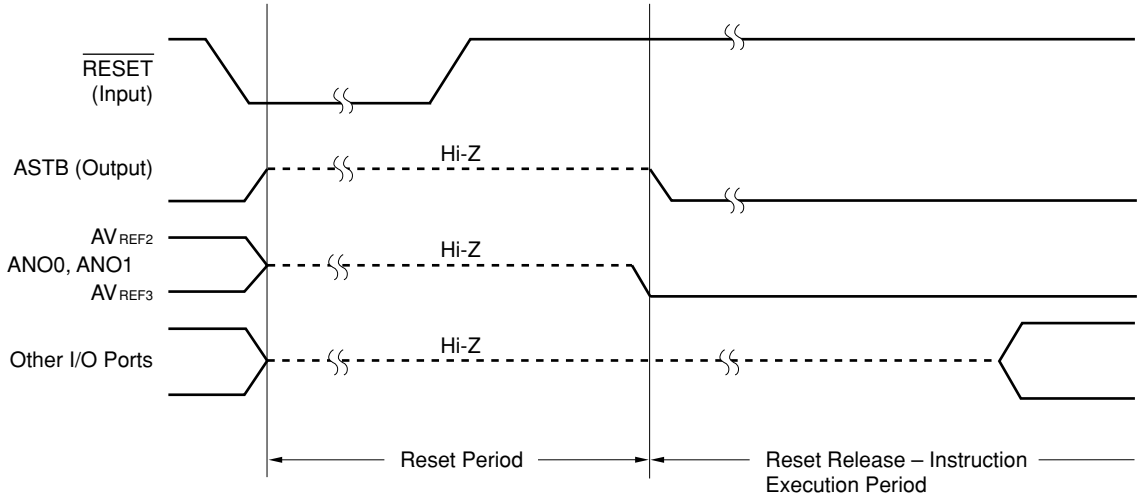
Note μPD784038Y Subseries only

Figure 25-3 Reset Input Timing

(a) μPD784031



(b) μPD784038



25.2 CAUTION

Reset input when powering on must remain at the low level until oscillation stabilizes after the supply voltage has reached the prescribed voltage.

CHAPTER 26 μPD78P4038 PROGRAMMING

The μPD78P4038 incorporates a 128-Kbyte PROM as program memory. When programming the μPD78P4038, the PROM programming mode is set by means of the V_{PP} pin and the $\overline{RESET}$ pin. For the connection of unused pins, see **1.3.2 PROM Programming Mode** in **1.3 PIN CONFIGURATION** (Top View).

26.1 OPERATING MODES

When +5 V or +12.5 V is applied to the V_{PP} pin and a low-level signal is applied to the $\overline{RESET}$ pin, the μPD78P4038 is placed in the PROM programming mode. This is one of the operating modes shown in **Table 26-1** below according to the setting of the $\overline{CE}$, $\overline{OE}$, and $\overline{PGM}$ pins.

The PROM contents can be read by setting the read mode.

Table 26-1 PROM Programming Operating Modes

Operating Mode \ Pins	$\overline{RESET}$	V_{PP}	V_{DD}	$\overline{CE}$	$\overline{OE}$	$\overline{PGM}$	D0 to D7
Page data latch	L	+12.5 V	+6.5 V	H	L	H	Data input
Page write				H	H	L	High-impedance
Byte write				L	H	L	Data input
Program verify				L	L	H	Data output
Program inhibit				×	H	H	High-impedance
				×	L	L	
Read		+5 V	+5 V	L	L	H	Data output
Output disable				L	H	×	High-impedance
Standby				H	×	×	High-impedance

Remark ×: L or H

(1) Read mode

Read mode is set by setting $\overline{CE}$ to L and $\overline{OE}$ to L.

(2) Output disable mode

If $\overline{OE}$ is set to H, data output becomes high impedance and the output disable mode is set.

Therefore, if multiple μPD78P4038s are connected to the data bus, data can be read from any one device by controlling the $\overline{OE}$ pin.

(3) Standby mode

Setting $\overline{CE}$ to H sets the standby mode.

In this mode, data output becomes high-impedance irrespective of the status of $\overline{OE}$.

(4) Page data latch mode

Setting $\overline{CE}$ to H, $\overline{PGM}$ to H, and $\overline{OE}$ to L at the start of the page write mode sets the page data latch mode.

In this mode, 1-page 4-byte data is latched in the internal address/data latch circuit.

(5) Page write mode

After 1-page 4-byte address and data are latched in the page data latch mode, a page write is executed by applying a 0.1 ms program pulse (active-low) to the $\overline{PGM}$ pin while $\overline{CE}$ = H and $\overline{OE}$ = H. After this, program verification can be performed by setting $\overline{CE}$ to L and $\overline{OE}$ to L.

If programming is not performed by one program pulse, repeated write and verify operations are executed X times ($X \leq 10$).

(6) Byte write mode

A byte write is executed by applying a 0.1 ms program pulse (active-low) to the $\overline{PGM}$ pin while $\overline{CE}$ = L and $\overline{OE}$ = H. After this, program verification can be performed by setting $\overline{OE}$ to L.

If programming is not performed by one program pulse, repeated write and verify operations are executed X times ($X \leq 10$).

(7) Program verify mode

Setting $\overline{CE}$ to L, $\overline{PGM}$ to H, and $\overline{OE}$ to L sets the program verify mode.

After writing is performed, this mode should be used to check whether the data has been written correctly.

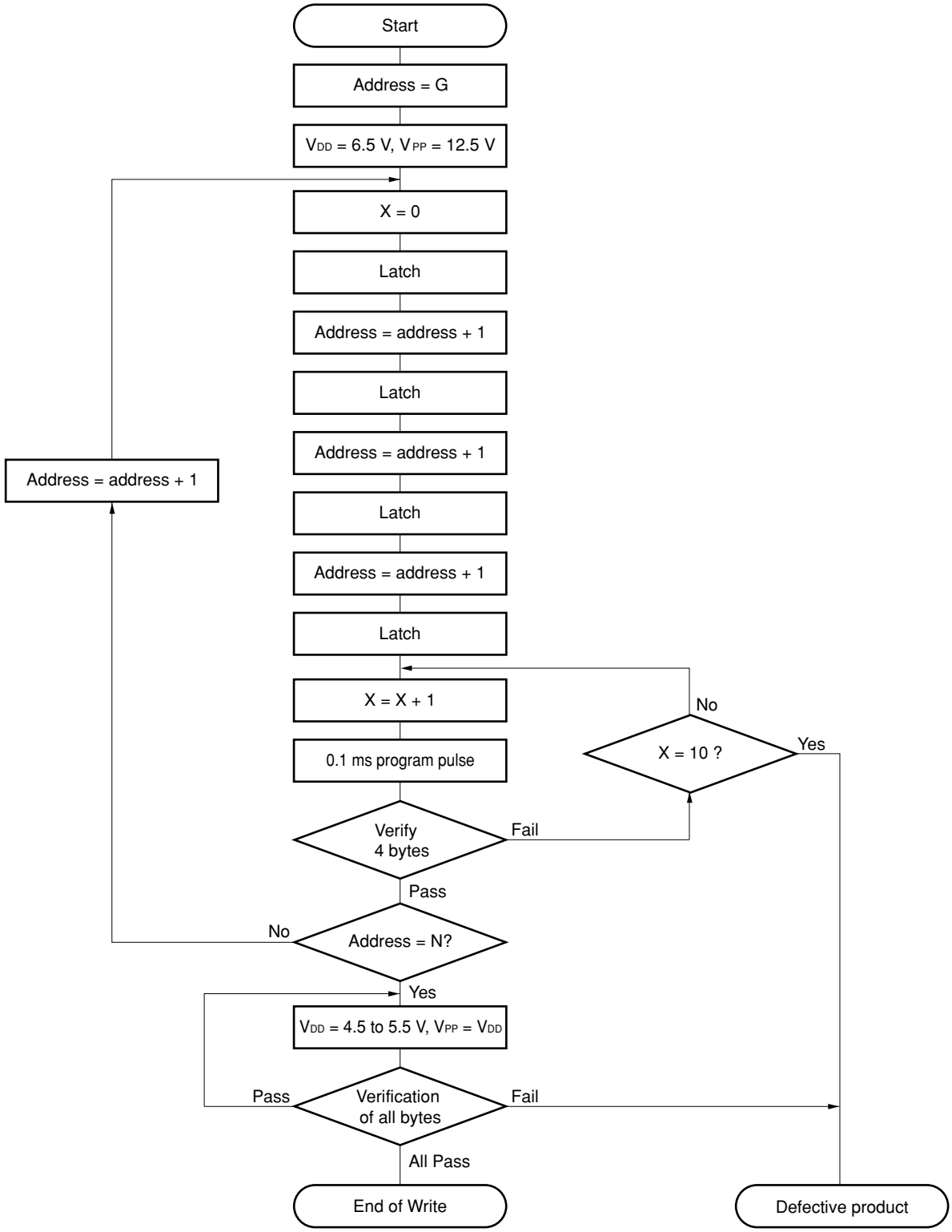
(8) Program inhibit mode

The program inhibit mode is used when the $\overline{OE}$ pins, V_{PP} pins and pins D0 to D7 of multiple μPD78P4038s are connected in parallel, and you wish to write to one of these devices.

The page write mode or byte write mode described above is used to perform a write. At this time, a write is not performed on devices on which the $\overline{PGM}$ pin is driven high.

26.2 PROM WRITE PROCEDURE

Figure 26-1 Page Program Mode Flowchart



Remark G = Start address
N = Last address of program

Figure 26-2 Page Program Mode Timing

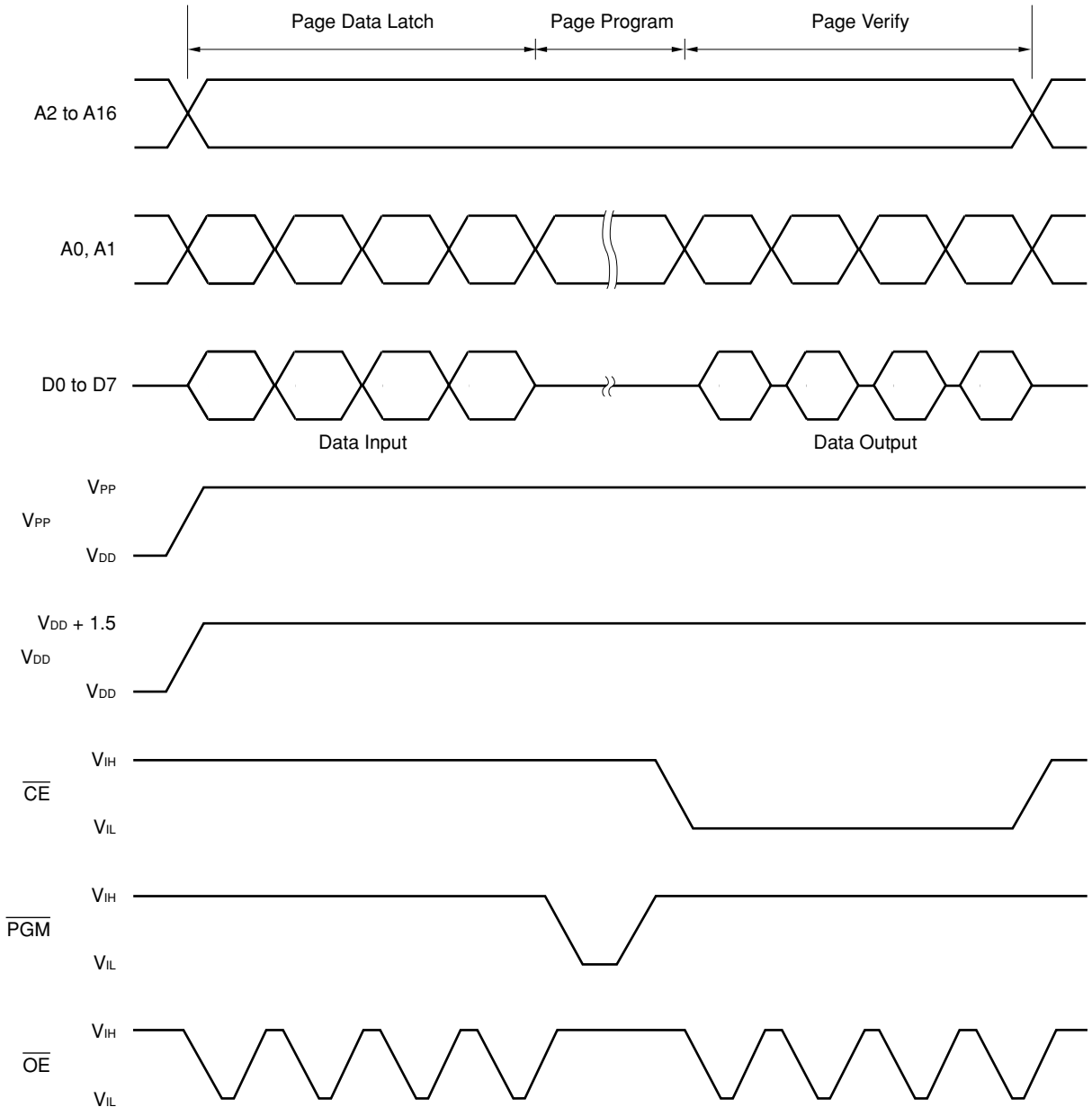
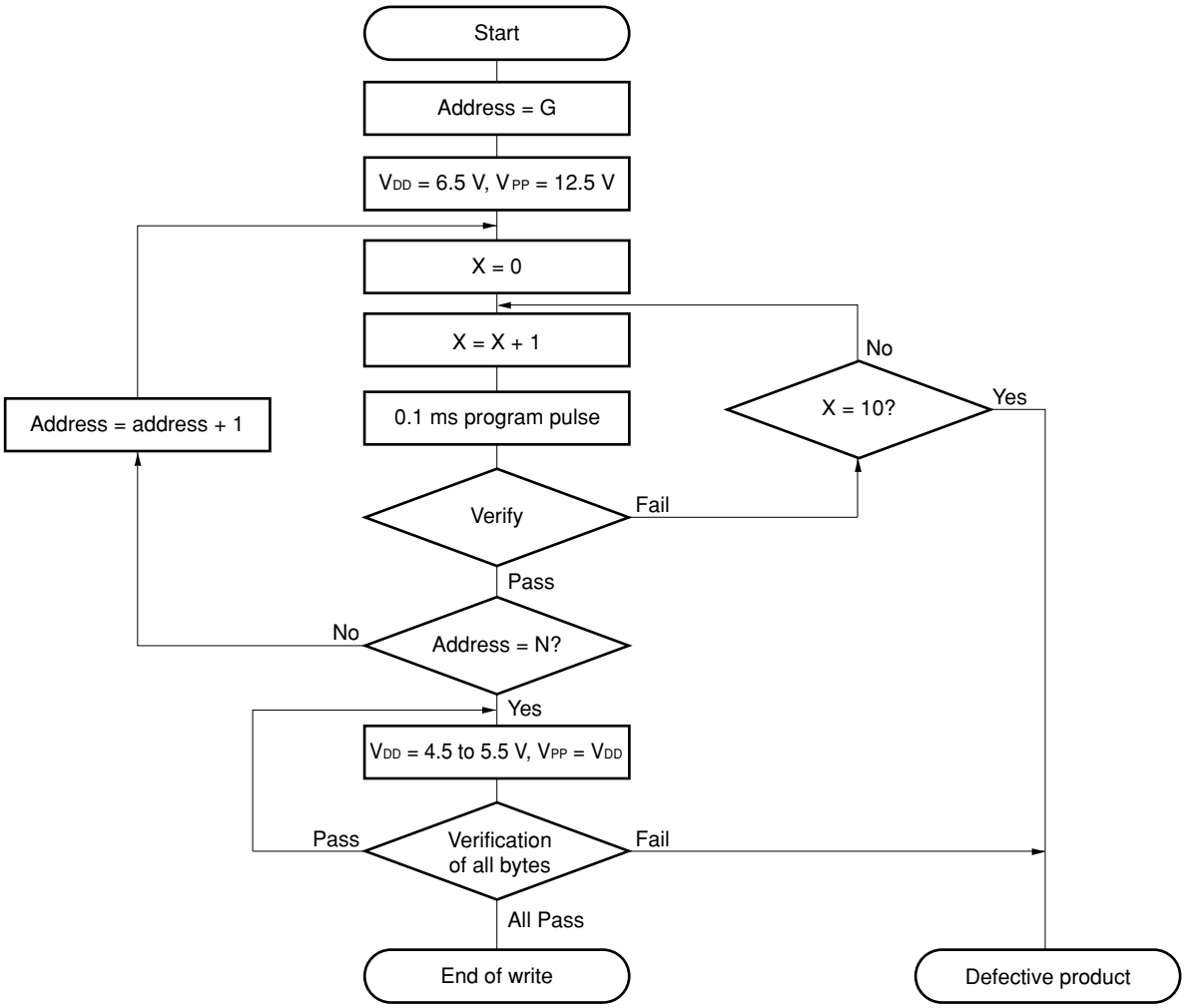
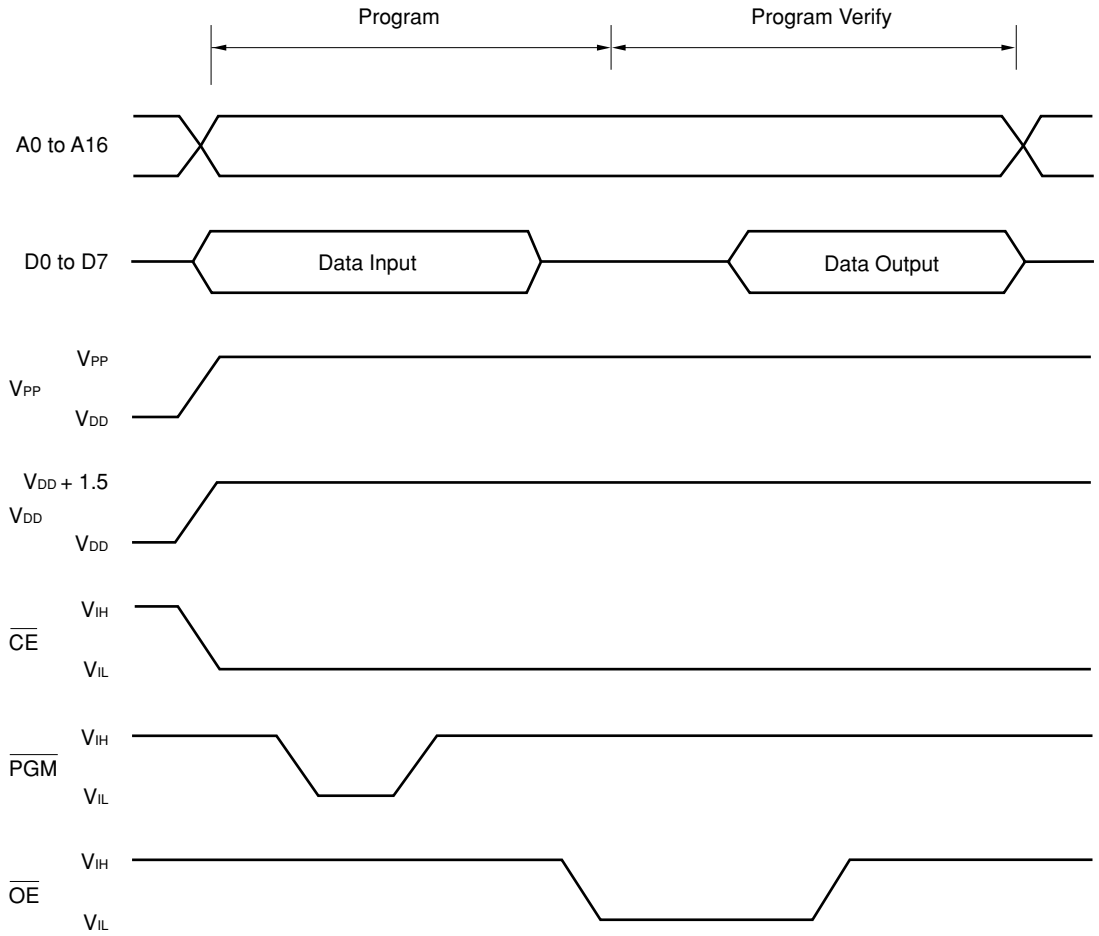


Figure 26-3 Byte Program Mode Flowchart



Remark G = Start address
N = Last address of program

Figure 26-4 Byte Program Mode Timing



Cautions 1. Ensure that V_{DD} is applied before V_{PP}, and cut after V_{PP}.

2. Ensure that V_{PP} does not become +13.5 V or over including overshoot.

3. Removing the device while +12.5 V is being applied to V_{PP} may have an adverse affect on reliability.

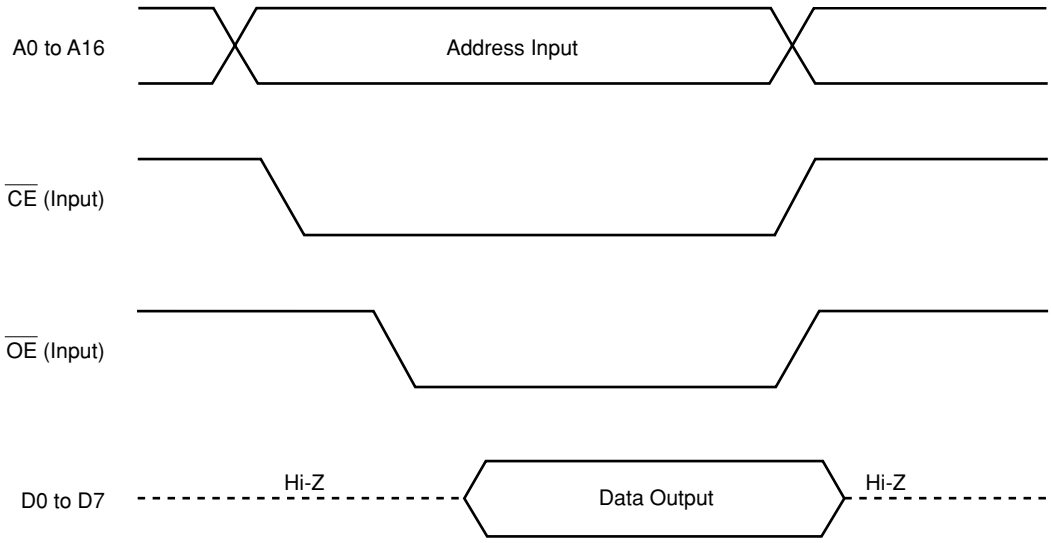
26.3 PROM READING PROCEDURE

PROM contents can be read onto the external data bus (D0 to D7) using the following procedure.

(1) Fix the $\overline{\text{RESET}}$ pin low, and supply +5 V to the V_{PP} pin. Unused pins are handled as shown in **1.3.2 PROM Programming Mode** in **1.3 PIN CONFIGURATION** (Top View).
(2) Supply +5 V to the V_{DD} and V_{PP} pins.
(3) Input address of data to be read to pins A0 to A16.
(4) Read mode.
(5) Output data to pins D0 to D7.

The timing for steps (2) to (5) above is shown in Figure 26-5.

Figure 26-5 PROM Read Timing



26.4 SCREENING OF ONE-TIME PROM PRODUCT

Because of its construction, the one-time PROM product (μPD78P4038GC-8BT, 78P4038YGC-8BT, 78P4038GK-9EU, 78P4038YGK-9EU) cannot be fully tested by NEC Electronics before shipment. After the necessary data has been written, it is recommended that screening be carried out by performing PROM verification after high-temperature storage under the following conditions.

Storage Temperature	Storage Time
125°C	24 hours

26.5 CAUTIONS

(1) Ensure that V_{DD} is applied before V_{PP}, and cut after V_{PP}.

(2) Ensure that V_{PP} does not become +13.5 V or over including overshoot.

(3) Removing the device while +12.5 V is being applied to V_{PP} may have an adverse affect on reliability.

CHAPTER 27 INSTRUCTION OPERATIONS

27.1 LEGEND

(1) Operand identifiers and descriptions (1/2)

Identifier	Description
r, r' Note 1	X(R0), A(R1), C(R2), B(R3), R4, R5, R6, R7, R8, R9, R10, R11, E(R12), D(R13), L(R14), H(R15)
r1 Note 1	X(R0), A(R1), C(R2), B(R3), R4, R5, R6, R7
r2	R8, R9, R10, R11, E(R12), D(R13), L(R14), H(R15)
r3	V, U, T, W
rp, rp' Note 2	AX(RP0), BC(RP1), RP2, RP3, VP(RP4), UP(RP5), DE(RP6), HL(RP7)
rp1 Note 2	AX(RP0), BC(RP1), RP2, RP3
rp2	VP(RP4), UP(RP5), DE(RP6), HL(RP7)
rg, rg'	VVP(RG4), UUP(RG5), TDE(RG6), WHL(RG7)
sfr	Special function register symbol
sfrp	Special function register symbol (register for which 16-bit operation is possible)
post Note 2	AX(RP0), BC(RP1), RP2, RP3, VP(RP4), UP(RP5)/PSW, DE(RP6), HL(RP7) Multiple descriptions are permissible. However, UP is only used with PUSH/POP instructions, and PSW with PUSHU/POPU instructions.
mem	[TDE], [WHL], [TDE+], [WHL+], [TDE–], [WHL–], [VVP], [UUP]: Register indirect addressing [TDE+byte], [WHL+byte], [SP+byte], [UUP+byte], [VVP+byte]: Based addressing imm24 [A], imm24 [B], imm24 [DE], imm24 [HL]: Indexed addressing [TDE+A], [TDE+B], [TDE+C], [WHL+A], [WHL+B], [WHL+C], [VVP+DE], [VVP+HL]: Based indexed addressing
mem1	All mem except [WHL+] and [WHL–]
mem2	[TDE], [WHL]
mem3	[AX], [BC], [RP2], [RP3], [VVP], [UUP], [TDE], [WHL]

Notes 1. Setting the RSS bit to 1 enables R4 to R7 to be used as X, A, C, and B, but this function should only be used when using a 78K/III Series program.

2. Setting the RSS bit to 1 enables RP2 and RP3 to be used as AX and BC, but this function should only be used when using a 78K/III Series program.

(1) Operand identifiers and descriptions (2/2)

Identifier	Description
saddr, saddr'	FD20H to FF1FH immediate data or label
saddr1	FE00H to FEFFH immediate data or label
saddr2	FD20H to FDFFH, FF00H to FF1FH immediate data or label
saddrp	FD20H to FF1EH immediate data or label (16-bit operation)
saddrp1	FE00H to FEFFH immediate data or label (16-bit operation)
saddrp2	FD20H to FDFFH, FF00H to FF1EH immediate data or label (16-bit operation)
saddrg	FD20H to FEFDH immediate data or label (24-bit operation)
saddrg1	FE00H to FEFDH immediate data or label (24-bit operation)
saddrg2	FD20H to FDFFH immediate data or label (24-bit operation)
addr24	0H to FFFFFFH immediate data or label
addr20	0H to FFFFFH immediate data or label
addr16	0H to FFFFH immediate data or label
addr11	800H to FFFH immediate data or label
addr8	0FE00H to 0FEFFH **Note** immediate data or label
addr5	40H to 7EH immediate data or label
imm24	24-bit immediate data or label
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
n	3-bit immediate data
locaddr	0H or 0FH

Note The addresses shown here apply when 0H is specified by the LOCATION instruction.
When 0FH is specified by the LOCATION instruction, F0000H should be added to the address values shown.

(2) Operand column symbols

Symbol	Description
+	Auto-increment
–	Auto-decrement
#	Immediate data
!	16-bit absolute address
!!	24-bit/20-bit absolute address
$	8-bit relative address
$!	16-bit relative address
/	Bit inversion
[ ]	Indirect addressing
[%]	24-bit indirect addressing

(3) Flag column symbols

Symbol	Description
(Blank)	No change
0	Cleared to 0
1	Set to 1
×	Set or cleared depending on result
P	P/V flag operates as parity flag
V	P/V flag operates as overflow flag
R	Previously saved value is restored

(4) Operation column symbols

Symbol	Description
jdisp8	Signed two's complement data (8 bits) indicating relative address distance between start address of next instruction and branch address
jdisp16	Signed two's complement data (16 bits) indicating relative address distance between start address of next instruction and branch address
PC_{HW}	PC bits 16 to 19
PC_{LW}	PC bits 0 to 15

(5) Number of bytes of instruction that includes mem in operands

mem Mode	Register Indirect Addressing		Based Addressing	Indexed Addressing	Based Indexed Addressing
Number of bytes	1	2 **Note**	3	5	2

Note One-byte instruction only when [TDE], [WHL], [TDE+], [TDE-], [WHL+], or [WHL–] is written as mem in an MOV instruction.

(6) Number of bytes of instruction that includes saddr, saddrp, r or rp in operands

For some instructions that include saddr, saddrp, r, or rp in their operands, two "Bytes" entries are given, separated by a slash ("/"). The entry that applies is shown in the table below.

Identifier	Left-Hand "Bytes" Figure	Right-Hand "Bytes" Figure
saddr	saddr2	saddr1
saddrp	saddrp2	saddrp1
r	r1	r2
rp	rp1	rp2

(7) Description of instructions that include mem in operands and string instructions

Operands TDE, WHL, VVP, and UUP (24-bit registers) can also be written as DE, HL, VP, and UP respectively. However, they are still treated as TDE, WHL, VVP, and UUP (24-bit registers) when written as DE, HL, VP, and UP.

27.2 LIST OF OPERATIONS

(1) 8-bit data transfer instruction: MOV

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
MOV	r, #byte	2/3	r ← byte					
	saddr, #byte	3/4	(saddr) ← byte					
	sfr, #byte	3	sfr ← byte					
	!addr16, #byte	5	(saddr16) ← byte					
	!!addr24, #byte	6	(addr24) ← byte					
	r, r'	2/3	r ← r'					
	A, r	1/2	A ← r					
	A, saddr2	2	A ← (saddr2)					
	r, saddr	3	r ← (saddr)					
	saddr2, A	2	(saddr2) ← A					
	saddr, r	3	(saddr) ← r					
	A, sfr	2	A ← sfr					
	r, sfr	3	r ← sfr					
	sfr, A	2	sfr ← A					
	sfr, r	3	sfr ← r					
	saddr, saddr'	4	(saddr) ← (saddr')					
	r, !addr16	4	r ← (addr16)					
	!addr16, r	4	(addr16) ← r					
	r, !!addr24	5	r ← (addr24)					
	!!addr24, r	5	(addr24) ← r					
	A, [saddrp]	2/3	A ← ((saddrp))					
	A, [%saddrg]	3/4	A ← ((saddrg))					
	A, mem	1-5	A ← (mem)					
	[saddrp], A	2/3	((saddrp)) ← A					
	[%saddrg], A	3/4	((saddrg)) ← A					
	mem, A	1-5	(mem) ← A					
	PSWL, #byte	3	PSW_L ← byte	×	×	×	×	×
	PSWH, #byte	3	PSW_H ← byte					
	PSWL, A	2	PSW_L ← A	×	×	×	×	×
	PSWH, A	2	PSW_H ← A					
	A, PSWL	2	A ← PSW_L					
	A, PSWH	2	A ← PSW_H					
	r3, #byte	3	r3 ← byte					
	A, r3	2	A ← r3					
	r3, A	2	r3 ← A					

(2) 16-bit data transfer instruction: MOVW

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
MOVW	rp, #word	3	rp ← word					
	saddrp, #word	4/5	(saddrp) ← word					
	sfrp, #word	4	sfrp ← word					
	!addr16, #word	6	(addr16) ← word					
	!!addr24, #word	7	(addr24) ← word					
	rp, rp'	2	rp ← rp'					
	AX, saddrp2	2	AX ← (saddrp2)					
	rp, saddrp	3	rp ← (saddrp)					
	saddrp2, AX	2	(saddrp2) ← AX					
	saddrp, rp	3	(saddrp) ← rp					
	AX, sfrp	2	AX ← sfrp					
	rp, sfrp	3	rp ← sfrp					
	sfrp, AX	2	sfrp ← AX					
	sfrp, rp	3	sfrp ← rp					
	saddrp, saddrp'	4	(saddrp) ← (saddrp')					
	rp, !addr16	4	rp ← (addr16)					
	!addr16, rp	4	(addr16) ← rp					
	rp, !!addr24	5	rp ← (addr24)					
	!!addr24, rp	5	(addr24) ← rp					
	AX, [saddrp]	3/4	AX ← ((saddrp))					
	AX, [%saddrg]	3/4	AX ← ((saddrg))					
	AX, mem	2-5	AX ← (mem)					
	[saddrp], AX	3/4	((saddrp)) ← AX					
	[%saddrg], AX	3/4	((saddrg)) ← AX					
	mem, AX	2-5	(mem) ← AX					

(3) 24-bit data transfer instruction: MOVG

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
MOVG	rg, #imm24	5	rg ← imm24					
	rg, rg'	2	rg ← rg'					
	rg, !!addr24	5	rg ← (addr24)					
	!!addr24, rg	5	(addr24) ← rg					
	rg, saddrg	3	rg ← (saddrg)					
	saddrg, rg	3	(saddrg) ← rg					
	WHL, [%saddrg]	3/4	WHL ← ((saddrg))					
	[%saddrg], WHL	3/4	((saddrg)) ← WHL					
	WHL, mem1	2-5	WHL ← (mem1)					
	mem1, WHL	2-5	(mem1) ← WHL					

(4) 8-bit data exchange instruction: XCH

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
XCH	r, r'	2/3	r ↔ r'					
	A, r	1/2	A ↔ r					
	A, saddr2	2	A ↔ (saddr2)					
	r, saddr	3	r ↔ (saddr)					
	r, sfr	3	r ↔ sfr					
	saddr, saddr'	4	(saddr) ↔ (saddr')					
	r, !addr16	4	r ↔ (addr16)					
	r, !!addr24	5	r ↔ (addr24)					
	A, [saddrp]	2/3	A ↔ ((saddrp))					
	A, [%saddrg]	3/4	A ↔ ((saddrg))					
	A, mem	2-5	A ↔ (mem)					

(5) 16-bit data exchange instruction: XCHW

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
XCHW	rp, rp'	2	rp ↔ rp'					
	AX, saddrp2	2	AX ↔ (saddrp2)					
	rp, saddrp	3	rp ↔ (saddrp)					
	rp, sfrp	3	rp ↔ sfrp					
	AX, [saddrp]	3/4	AX ↔ ((saddrp))					
	AX, [%saddrg]	3/4	AX ↔ ((saddrg))					
	AX, !addr16	4	AX ↔ (addr16)					
	AX, !!addr24	5	AX ↔ (addr24)					
	saddrp, saddrp'	4	(saddrp) ↔ (saddrp')					
	AX, mem	2-5	AX ↔ (mem)					

(6) 8-bit operation instructions: ADD, ADDC, SUB, SUBC, CMP, AND, OR, XOR

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
ADD	A, #byte	2	A, CY ← A + byte	×	×	×	V	×
	r, #byte	3	r, CY ← r + byte	×	×	×	V	×
	saddr, #byte	3/4	(saddr), CY ← (saddr) + byte	×	×	×	V	×
	sfr, #byte	4	sfr, CY ← sfr + byte	×	×	×	V	×
	r, r'	2/3	r, CY ← r + r'	×	×	×	V	×
	A, saddr2	2	A, CY ← A + (saddr2)	×	×	×	V	×
	r, saddr	3	r, CY ← r + (saddr)	×	×	×	V	×
	saddr, r	3	(saddr), CY ← (saddr) + r	×	×	×	V	×
	r, sfr	3	r, CY ← r + sfr	×	×	×	V	×
	sfr, r	3	sfr, CY ← sfr + r	×	×	×	V	×
	saddr, saddr'	4	(saddr), CY ← (saddr) + (saddr')	×	×	×	V	×
	A, [saddrp]	3/4	A, CY ← A + ((saddrp))	×	×	×	V	×
	A, [%saddrg]	3/4	A, CY ← A + ((saddrg))	×	×	×	V	×
	[saddrp], A	3/4	((saddrp)), CY ← ((saddrp)) + A	×	×	×	V	×
	[%saddrg], A	3/4	((saddrg)), CY ← ((saddrg)) + A	×	×	×	V	×
	A, !addr16	4	A, CY ← A + (addr16)	×	×	×	V	×
	A, !!addr24	5	A, CY ← A + (addr24)	×	×	×	V	×
	!addr16, A	4	(addr16), CY ← (addr16) + A	×	×	×	V	×
	!!addr24, A	5	(addr24), CY ← (addr24) + A	×	×	×	V	×
	A, mem	2-5	A, CY ← A + (mem)	×	×	×	V	×
	mem, A	2-5	(mem), CY ← (mem) + A	×	×	×	V	×

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
ADDC	A, #byte	2	A, CY ← A + byte + CY	×	×	×	V	×
	r, #byte	3	r, CY ← r + byte + CY	×	×	×	V	×
	saddr, #byte	3/4	(saddr), CY ← (saddr) + byte + CY	×	×	×	V	×
	sfr, #byte	4	sfr, CY ← sfr + byte + CY	×	×	×	V	×
	r, r'	2/3	r, CY ← r + r' + CY	×	×	×	V	×
	A, saddr2	2	A, CY ← A + (saddr2) + CY	×	×	×	V	×
	r, saddr	3	r, CY ← r + (saddr) + CY	×	×	×	V	×
	saddr, r	3	(saddr), CY ← (saddr) + r + CY	×	×	×	V	×
	r, sfr	3	r, CY ← r + sfr + CY	×	×	×	V	×
	sfr, r	3	sfr, CY ← sfr + r + CY	×	×	×	V	×
	saddr, saddr'	4	(saddr), CY ← (saddr) + (saddr') + CY	×	×	×	V	×
	A, [saddrp]	3/4	A, CY ← A + ((saddrp)) + CY	×	×	×	V	×
	A, [%saddrg]	3/4	A, CY ← A + ((saddrg)) + CY	×	×	×	V	×
	[saddrp], A	3/4	((saddrp)), CY ← ((saddrp)) + A + CY	×	×	×	V	×
	[%saddrg], A	3/4	((saddrg)), CY ← ((saddrg)) + A + CY	×	×	×	V	×
	A, !addr16	4	A, CY ← A + (addr16) + CY	×	×	×	V	×
	A, !!addr24	5	A, CY ← A + (addr24) + CY	×	×	×	V	×
	!addr16, A	4	(addr16), CY ← (addr16) + A + CY	×	×	×	V	×
	!!addr24, A	5	(addr24), CY ← (addr24) + A + CY	×	×	×	V	×
	A, mem	2-5	A, CY ← A + (mem) + CY	×	×	×	V	×
	mem, A	2-5	(mem), CY ← (mem) + A + CY	×	×	×	V	×

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
SUB	A, #byte	2	A, CY ← A – byte	×	×	×	V	×
	r, #byte	3	r, CY ← r – byte	×	×	×	V	×
	saddr, #byte	3/4	(saddr), CY ← (saddr) – byte	×	×	×	V	×
	sfr, #byte	4	sfr, CY ← sfr – byte	×	×	×	V	×
	r, r'	2/3	r, CY ← r – r'	×	×	×	V	×
	A, saddr2	2	A, CY ← A – (saddr2)	×	×	×	V	×
	r, saddr	3	r, CY ← r – (saddr)	×	×	×	V	×
	saddr, r	3	(saddr), CY ← (saddr) – r	×	×	×	V	×
	r, sfr	3	r, CY ← r – sfr	×	×	×	V	×
	sfr, r	3	sfr, CY ← sfr – r	×	×	×	V	×
	saddr, saddr'	4	(saddr), CY ← (saddr) – (saddr')	×	×	×	V	×
	A, [saddrp]	3/4	A, CY ← A – ((saddrp))	×	×	×	V	×
	A, [%saddrg]	3/4	A, CY ← A – ((saddrg))	×	×	×	V	×
	[saddrp], A	3/4	((saddrp)), CY ← ((saddrp)) – A	×	×	×	V	×
	[%saddrg], A	3/4	((saddrg)), CY ← ((saddrg)) – A	×	×	×	V	×
	A, !addr16	4	A, CY ← A – (addr16)	×	×	×	V	×
	A, !!addr24	5	A, CY ← A – (addr24)	×	×	×	V	×
	!addr16, A	4	(addr16), CY ← (addr16) – A	×	×	×	V	×
	!!addr24, A	5	(addr24), CY ← (addr24) – A	×	×	×	V	×
	A, mem	2-5	A, CY ← A – (mem)	×	×	×	V	×
	mem, A	2-5	(mem), CY ← (mem) – A	×	×	×	V	×

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
SUBC	A, #byte	2	A, CY ← A – byte – CY	×	×	×	V	×
	r, #byte	3	r, CY ← r – byte – CY	×	×	×	V	×
	saddr, #byte	3/4	(saddr), CY ← (saddr) – byte – CY	×	×	×	V	×
	sfr, #byte	4	sfr, CY ← sfr – byte – CY	×	×	×	V	×
	r, r'	2/3	r, CY ← r – r' – CY	×	×	×	V	×
	A, saddr2	2	A, CY ← A – (saddr2) – CY	×	×	×	V	×
	r, saddr	3	r, CY ← r – (saddr) – CY	×	×	×	V	×
	saddr, r	3	(saddr), CY ← (saddr) – r – CY	×	×	×	V	×
	r, sfr	3	r, CY ← r – sfr – CY	×	×	×	V	×
	sfr, r	3	sfr, CY ← sfr – r – CY	×	×	×	V	×
	saddr, saddr'	4	(saddr), CY ← (saddr) – (saddr') – CY	×	×	×	V	×
	A, [saddrp]	3/4	A, CY ← A – ((saddrp)) – CY	×	×	×	V	×
	A, [%saddrg]	3/4	A, CY ← A – ((saddrg)) – CY	×	×	×	V	×
	[saddrp], A	3/4	((saddrp)), CY ← ((saddrp)) – A – CY	×	×	×	V	×
	[%saddrg], A	3/4	((saddrg)), CY ← ((saddrg)) – A – CY	×	×	×	V	×
	A, !addr16	4	A, CY ← A – (addr16) – CY	×	×	×	V	×
	A, !!addr24	5	A, CY ← A – (addr24) – CY	×	×	×	V	×
	!addr16, A	4	(addr16), CY ← (addr16) – A – CY	×	×	×	V	×
	!!addr24, A	5	(addr24), CY ← (addr24) – A – CY	×	×	×	V	×
	A, mem	2-5	A, CY ← A – (mem) – CY	×	×	×	V	×
	mem, A	2-5	(mem), CY ← (mem) – A – CY	×	×	×	V	×

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
CMP	A, #byte	2	A – byte	×	×	×	V	×
	r, #byte	3	r – byte	×	×	×	V	×
	saddr, #byte	3/4	(saddr) – byte	×	×	×	V	×
	sfr, #byte	4	sfr – byte	×	×	×	V	×
	r, r'	2/3	r – r'	×	×	×	V	×
	A, saddr2	2	A – (saddr2)	×	×	×	V	×
	r, saddr	3	r – (saddr)	×	×	×	V	×
	saddr, r	3	(saddr) – r	×	×	×	V	×
	r, sfr	3	r – sfr	×	×	×	V	×
	sfr, r	3	sfr – r	×	×	×	V	×
	saddr, saddr'	4	(saddr) – (saddr')	×	×	×	V	×
	A, [saddrp]	3/4	A – ((saddrp))	×	×	×	V	×
	A, [%saddrg]	3/4	A – ((saddrg))	×	×	×	V	×
	[saddrp], A	3/4	((saddrp)) – A	×	×	×	V	×
	[%saddrg], A	3/4	((saddrg)) – A	×	×	×	V	×
	A, !addr16	4	A – (addr16)	×	×	×	V	×
	A, !!addr24	5	A – (addr24)	×	×	×	V	×
	!addr16, A	4	(addr16) – A	×	×	×	V	×
	!!addr24, A	5	(addr24) – A	×	×	×	V	×
	A, mem	2-5	A – (mem)	×	×	×	V	×
	mem, A	2-5	(mem) – A	×	×	×	V	×

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
AND	A, #byte	2	A ← A ∧ byte	×	×		P	
	r, #byte	3	r ← r ∧ byte	×	×		P	
	saddr, #byte	3/4	(saddr) ← (saddr) ∧ byte	×	×		P	
	sfr, #byte	4	sfr ← sfr ∧ byte	×	×		P	
	r, r'	2/3	r ← r ∧ r'	×	×		P	
	A, saddr2	2	A ← A ∧ (saddr2)	×	×		P	
	r, saddr	3	r ← r ∧ (saddr)	×	×		P	
	saddr, r	3	(saddr) ← (saddr) ∧ r	×	×		P	
	r, sfr	3	r ← r ∧ sfr	×	×		P	
	sfr, r	3	sfr ← sfr ∧ r	×	×		P	
	saddr, saddr'	4	(saddr) ← (saddr) ∧ (saddr')	×	×		P	
	A, [saddrp]	3/4	A ← A ∧ ((saddrp))	×	×		P	
	A, [%saddrg]	3/4	A ← A ∧ ((saddrg))	×	×		P	
	[saddrp], A	3/4	((saddrp)) ← ((saddrp)) ∧ A	×	×		P	
	[%saddrg], A	3/4	((saddrg)) ← ((saddrg)) ∧ A	×	×		P	
	A, !addr16	4	A ← A ∧ (addr16)	×	×		P	
	A, !!addr24	5	A ← A ∧ (addr24)	×	×		P	
	!addr16, A	4	(addr16) ← (addr16) ∧ A	×	×		P	
	!!addr24, A	5	(addr24) ← (addr24) ∧ A	×	×		P	
	A, mem	2-5	A ← A ∧ (mem)	×	×		P	
	mem, A	2-5	(mem) ← (mem) ∧ A	×	×		P	

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
OR	A, #byte	2	A ← A ∨ byte	×	×		P	
	r, #byte	3	r ← r ∨ byte	×	×		P	
	saddr, #byte	3/4	(saddr) ← (saddr) ∨ byte	×	×		P	
	sfr, #byte	4	sfr ← sfr ∨ byte	×	×		P	
	r, r'	2/3	r ← r ∨ r'	×	×		P	
	A, saddr2	2	A ← A ∨ (saddr2)	×	×		P	
	r, saddr	3	r ← r ∨ (saddr)	×	×		P	
	saddr, r	3	(saddr) ← (saddr) ∨ r	×	×		P	
	r, sfr	3	r ← r ∨ sfr	×	×		P	
	sfr, r	3	sfr ← sfr ∨ r	×	×		P	
	saddr, saddr'	4	(saddr) ← (saddr) ∨ (saddr')	×	×		P	
	A, [saddrp]	3/4	A ← A ∨ ((saddrp))	×	×		P	
	A, [%saddrg]	3/4	A ← A ∨ ((saddrg))	×	×		P	
	[saddrp], A	3/4	((saddrp)) ← ((saddrp)) ∨ A	×	×		P	
	[%saddrg], A	3/4	((saddrg)) ← ((saddrg)) ∨ A	×	×		P	
	A, !addr16	4	A ← A ∨ (addr16)	×	×		P	
	A, !!addr24	5	A ← A ∨ (addr24)	×	×		P	
	!addr16, A	4	(addr16) ← (addr16) ∨ A	×	×		P	
	!!addr24, A	5	(addr24) ← (addr24) ∨ A	×	×		P	
	A, mem	2-5	A ← A ∨ (mem)	×	×		P	
	mem, A	2-5	(mem) ← (mem) ∨ A	×	×		P	

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
XOR	A, #byte	2	A ← A ∀ byte	×	×		P	
	r, #byte	3	r ← r ∀ byte	×	×		P	
	saddr, #byte	3/4	(saddr) ← (saddr) ∀ byte	×	×		P	
	sfr, #byte	4	sfr ← sfr ∀ byte	×	×		P	
	r, r'	2/3	r ← r ∀ r'	×	×		P	
	A, saddr2	2	A ← A ∀ (saddr2)	×	×		P	
	r, saddr	3	r ← r ∀ (saddr)	×	×		P	
	saddr, r	3	(saddr) ← (saddr) ∀ r	×	×		P	
	r, sfr	3	r ← r ∀ sfr	×	×		P	
	sfr, r	3	sfr ← sfr ∀ r	×	×		P	
	saddr, saddr'	4	(saddr) ← (saddr) ∀ (saddr')	×	×		P	
	A, [saddrp]	3/4	A ← A ∀ ((saddrp))	×	×		P	
	A, [%saddrg]	3/4	A ← A ∀ ((saddrg))	×	×		P	
	[saddrp], A	3/4	((saddrp)) ← ((saddrp)) ∀ A	×	×		P	
	[%saddrg], A	3/4	((saddrg)) ← ((saddrg)) ∀ A	×	×		P	
	A, !addr16	4	A ← A ∀ (addr16)	×	×		P	
	A, !!addr24	5	A ← A ∀ (addr24)	×	×		P	
	!addr16, A	4	(addr16) ← (addr16) ∀ A	×	×		P	
	!!addr24, A	5	(addr24) ← (addr24) ∀ A	×	×		P	
	A, mem	2-5	A ← A ∀ (mem)	×	×		P	
	mem, A	2-5	(mem) ← (mem) ∀ A	×	×		P	

(7) 16-bit operation instructions: ADDW, SUBW, CMPW

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
ADDW	AX, #word	3	AX, CY ← AX + word	×	×	×	V	×
	rp, #word	4	rp, CY ← rp + word	×	×	×	V	×
	rp, rp'	2	rp, CY ← rp + rp'	×	×	×	V	×
	AX, saddrp2	2	AX, CY ← AX + (saddrp2)	×	×	×	V	×
	rp, saddrp	3	rp, CY ← rp + (saddrp)	×	×	×	V	×
	saddrp, rp	3	(saddrp), CY ← (saddrp) + rp	×	×	×	V	×
	rp, sfrp	3	rp, CY ← rp + sfrp	×	×	×	V	×
	sfrp, rp	3	sfrp, CY ← sfrp + rp	×	×	×	V	×
	saddrp, #word	4/5	(saddrp), CY ← (saddrp) + word	×	×	×	V	×
	sfrp, #word	5	sfrp, CY ← sfrp + word	×	×	×	V	×
	saddrp, saddrp'	4	(saddrp), CY ← (saddrp) + (saddrp')	×	×	×	V	×
SUBW	AX, #word	3	AX, CY ← AX – word	×	×	×	V	×
	rp, #word	4	rp, CY ← rp – word	×	×	×	V	×
	rp, rp'	2	rp, CY ← rp – rp'	×	×	×	V	×
	AX, saddrp2	2	AX, CY ← AX – (saddrp2)	×	×	×	V	×
	rp, saddrp	3	rp, CY ← rp – (saddrp)	×	×	×	V	×
	saddrp, rp	3	(saddrp), CY ← (saddrp) – rp	×	×	×	V	×
	rp, sfrp	3	rp, CY ← rp – sfrp	×	×	×	V	×
	sfrp, rp	3	sfrp, CY ← sfrp – rp	×	×	×	V	×
	saddrp, #word	4/5	(saddrp), CY ← (saddrp) – word	×	×	×	V	×
	sfrp, #word	5	sfrp, CY ← sfrp – word	×	×	×	V	×
	saddrp, saddrp'	4	(saddrp), CY ← (saddrp) – (saddrp')	×	×	×	V	×
CMPW	AX, #word	3	AX – word	×	×	×	V	×
	rp, #word	4	rp – word	×	×	×	V	×
	rp, rp'	2	rp – rp'	×	×	×	V	×
	AX, saddrp2	2	AX – (saddrp2)	×	×	×	V	×
	rp, saddrp	3	rp – (saddrp)	×	×	×	V	×
	saddrp, rp	3	(saddrp) – rp	×	×	×	V	×
	rp, sfrp	3	rp – sfrp	×	×	×	V	×
	sfrp, rp	3	sfrp – rp	×	×	×	V	×
	saddrp, #word	4/5	(saddrp) – word	×	×	×	V	×
	sfrp, #word	5	sfrp – word	×	×	×	V	×
	saddrp, saddrp'	4	(saddrp) – (saddrp')	×	×	×	V	×

(8) 24-bit operation instructions: ADDG, SUBG

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
ADDG	rg, rg'	2	rg, CY ← rg + rg'	×	×	×	V	×
	rg, # imm24	5	rg, CY ← rg + # imm24	×	×	×	V	×
	WHL, saddrg	3	WHL, CY ← WHL + (saddrg)	×	×	×	V	×
SUBG	rg, rg'	2	rg, CY ← rg – rg'	×	×	×	V	×
	rg, # imm24	5	rg, CY ← rg – imm24	×	×	×	V	×
	WHL, saddrg	3	WHL, CY ← WHL – (saddrg)	×	×	×	V	×

(9) Multiplication instructions: MULU, MULUW, MULW, DIVUW, DIVUX

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
MULU	r	2/3	AX ← A × r					
MULUW	rp	2	AX (upper half), rp (lower half) ← AX × rp					
MULW	rp	2	AX (upper half), rp (lower half) ← AX × rp					
DIVUW	r	2/3	AX (quotient), r (remainder) ← AX ÷ r **Note 1**					
DIVUX	rp	2	AXDE (quotient), rp (remainder) ← AXDE ÷ rp **Note 2**					

Notes 1. When r = 0, r ← X, AX ← FFFFH

2. When rp = 0, pr ← DE, AXDE ← FFFFFFFFH

(10) Special operation instructions: MACW, MACSW, SACW

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
MACW	byte	3	AXDE ← (B) × (C) + AXDE, B ← B + 2, C ← C + 2, byte ← byte – 1 End if (byte = 0 or P/V = 1)	×	×	×	V	×

MACSW	byte	3	AXDE ← (B) × (C) + AXDE, B ← B + 2,
			C ← C + 2, byte ← byte – 1
			if byte = 0 then End
			if P/V = 1 then
			  if overflow AXDE ← 7FFFFFFFH, End
			  if underflow AXDE ← 80000000H, End
				×	×	×	V	×

SACW	[TDE +], [WHL +]	4	AX ← |(TDE) – (WHL)| + AX, TDE ← TDE + 2, WHL ← WHL + 2 C ← C – 1 End if (C = 0 or CY = 1)	×	×	×	V	×

(11) Increment/decrement instructions: INC, DEC, INCW, DECW, INCG, DECG

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
INC	r	1/2	r ← r + 1	×	×	×	V	
	saddr	2/3	(saddr) ← (saddr) + 1	×	×	×	V	
DEC	r	1/2	r ← r –1	×	×	×	V	
	saddr	2/3	(saddr) ← (saddr) – 1	×	×	×	V	
INCW	rp	2/1	rp ← rp + 1					
	saddrp	3/4	(saddrp) ← (saddrp) + 1					
DECW	rp	2/1	rp ← rp – 1					
	saddrp	3/4	(saddrp) ← (saddrp) – 1					
INCG	rg	2	rg ← rg + 1					
DECG	rg	2	rg ← rg – 1					

(12) Adjustment instructions: ADJBA, ADJBS, CVTBW

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
ADJBA		2	Decimal Adjust Accumulator after Addition	×	×	×	P	×
ADJBS		2	Decimal Adjust Accumulator after Subtract	×	×	×	P	×
CVTBW		1	X ← A, A ← 00H if A_7 = 0 X ← A, A ← FFH if A_7 = 1					

(13) Shift/rotate instructions: ROR, ROL, RORC, ROLC, SHR, SHL, SHRW, SHLW, ROR4, ROL4

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
ROR	r, n	2/3	(CY, $r7 \leftarrow r0$, $rm-1 \leftarrow rm$) × n times n = 0 to 7				P	×
ROL	r, n	2/3	(CY, $r0 \leftarrow r7$, $rm+1 \leftarrow rm$) × n times n = 0 to 7				P	×
RORC	r, n	2/3	(CY$\leftarrow r0$, $r7 \leftarrow$ CY, $rm-1 \leftarrow rm$) × n times n = 0 to 7				P	×
ROLC	r, n	2/3	(CY$\leftarrow r7$, $r0 \leftarrow$ CY, $rm+1 \leftarrow rm$) × n times n = 0 to 7				P	×
SHR	r, n	2/3	(CY$\leftarrow r0$, $r7 \leftarrow 0$, $rm-1 \leftarrow rm$) × n times n = 0 to 7	×	×	0	P	×
SHL	r, n	2/3	(CY$\leftarrow r7$, $r0 \leftarrow 0$, $rm+1 \leftarrow rm$) × n times n = 0 to 7	×	×	0	P	×
SHRW	rp, n	2	(CY$\leftarrow rp0$, $rp15 \leftarrow 0$, $rpm-1 \leftarrow$ rpm) × n times n = 0 to 7	×	×	0	P	×
SHLW	rp, n	2	(CY$\leftarrow rp15$, $rp0 \leftarrow 0$, $rpm+1 \leftarrow$ rpm) × n times n = 0 to 7	×	×	0	P	×
ROR4	mem3	2	$A_{3-0} \leftarrow (mem3)_{3-0}$, $(mem3)_{7-4} \leftarrow A_{3-0}$, $(mem3)_{3-0} \leftarrow (mem3)_{7-4}$					
ROL4	mem3	2	$A_{3-0} \leftarrow (mem3)_{7-4}$, $(mem3)_{3-0} \leftarrow A_{3-0}$, $(mem3)_{7-4} \leftarrow (mem3)_{3-0}$					

(14) Bit manipulation instructions: MOV1, AND1, OR1, XOR1, NOT1, SET1, CLR1

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
MOV1	CY, saddr. bit	3/4	CY ← (saddr. bit)					×
	CY, sfr. bit	3	CY ← sfr. bit					×
	CY, X. bit	2	CY ← X. bit					×
	CY, A. bit	2	CY ← A. bit					×
	CY, PSWL. bit	2	CY ← PSWL. bit					×
	CY, PSWH. bit	2	CY ← PSWH. bit					×
	CY, !addr16. bit	5	CY ← !addr16.bit					×
	CY, !!addr24. bit	2	CY ← !!addr24. bit					×
	CY, mem2. bit	2	CY ← mem2. bit					×
	saddr. bit, CY	3/4	(saddr. bit) ← CY					
	sfr. bit, CY	3	sfr. bit ← CY					
	X. bit, CY	2	X.bit ← CY					
	A. bit, CY	2	A. bit ← CY					
	PSWL. bit, CY	2	PSW_L. bit ← CY	×	×	×	×	×
	PSWH. bit, CY	2	PSW_H. bit ← CY					
	!addr16. bit, CY	5	!addr16.bit ← CY					
	!!addr24.bit, CY	6	!!addr24.bit ← CY					
	mem2. bit, CY	2	mem2. bit ← CY					

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
AND1	CY, saddr. bit	3/4	CY ← CY ∧ (saddr. bit)					×
	CY, /saddr. bit	3/4	CY ← CY ∧ $\overline{\text{(saddr. bit)}}$					×
	CY, sfr. bit	3	CY ← CY ∧ sfr. bit					×
	CY, /sfr. bit	3	CY ← CY ∧ $\overline{\text{sfr. bit}}$					×
	CY, X. bit	2	CY ← CY ∧ X. bit					×
	CY, /X. bit	2	CY ← CY ∧ $\overline{\text{X. bit}}$					×
	CY, A. bit	2	CY ← CY ∧ A. bit					×
	CY, /A. bit	2	CY ← CY ∧ $\overline{\text{A. bit}}$					×
	CY, PSWL. bit	2	CY ← CY ∧ PSW_{L}. bit					×
	CY, /PSWL. bit	2	CY ← CY ∧ $\overline{\text{PSW}_{\text{L}}\text{. bit}}$					×
	CY, PSWH. bit	2	CY ← CY ∧ PSW_{H}. bit					×
	CY, /PSWH. bit	2	CY ← CY ∧ $\overline{\text{PSW}_{\text{H}}\text{. bit}}$					×
	CY, !addr16. bit	5	CY ← CY ∧ !addr16. bit					×
	CY, /!addr16. bit	5	CY ← CY ∧ $\overline{\text{!addr16. bit}}$					×
	CY, !!addr24. bit	2	CY ← CY ∧ !!addr24. bit					×
	CY, /!!addr24. bit	6	CY ← CY ∧ $\overline{\text{!!addr24. bit}}$					×
	CY, mem2. bit	2	CY ← CY ∧ mem2. bit					×
	CY, /mem2. bit	2	CY ← CY ∧ $\overline{\text{mem2. bit}}$					×
OR1	CY, saddr. bit	3/4	CY ← CY ∨ (saddr. bit)					×
	CY, /saddr. bit	3/4	CY ← CY ∨ $\overline{\text{(saddr. bit)}}$					×
	CY, sfr. bit	3	CY ← CY ∨ sfr. bit					×
	CY, /sfr. bit	3	CY ← CY ∨ $\overline{\text{sfr. bit}}$					×
	CY, X. bit	2	CY ← CY ∨ X. bit					×
	CY, /X. bit	2	CY ← CY ∨ $\overline{\text{X. bit}}$					×
	CY, A. bit	2	CY ← CY ∨ A. bit					×
	CY, /A. bit	2	CY ← CY ∨ $\overline{\text{A. bit}}$					×
	CY, PSWL. bit	2	CY ← CY ∨ PSW_{L}. bit					×
	CY, /PSWL. bit	2	CY ← CY ∨ $\overline{\text{PSW}_{\text{L}}\text{. bit}}$					×
	CY, PSWH. bit	2	CY ← CY ∨ PSW_{H}. bit					×
	CY, /PSWH. bit	2	CY ← CY ∨ $\overline{\text{PSW}_{\text{H}}\text{. bit}}$					×
	CY, !addr16. bit	5	CY ← CY ∨ !addr16. bit					×
	CY, /!addr16. bit	5	CY ← CY ∨ $\overline{\text{!addr16. bit}}$					×
	CY, !!addr24. bit	2	CY ← CY ∨ !!addr24. bit					×
	CY, /!!addr24. bit	6	CY ← CY ∨ $\overline{\text{!!addr24. bit}}$					×
	CY, mem2. bit	2	CY ← CY ∨ mem2. bit					×
	CY, /mem2. bit	2	CY ← CY ∨ $\overline{\text{mem2. bit}}$					×

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
XOR1	CY, saddr. bit	3/4	CY ← CY ∀ (saddr. bit)					×
	CY, sfr. bit	3	CY ← CY ∀ sfr. bit					×
	CY, X. bit	2	CY ← CY ∀ X. bit					×
	CY, A. bit	2	CY ← CY ∀ A. bit					×
	CY, PSWL. bit	2	CY ← CY ∀ PSWL. bit					×
	CY, PSWH. bit	2	CY ← CY ∀ PSWH. bit					×
	CY, !addr16. bit	5	CY ← CY ∀ !addr16. bit					×
	CY, !!addr24. bit	2	CY ← CY ∀ !!addr24. bit					×
	CY, mem2. bit	2	CY ← CY ∀ mem2. bit					×
NOT1	saddr. bit	3/4	(saddr. bit) ← ($\overline{\text{saddr. bit}}$)					
	sfr. bit	3	sfr. bit ← $\overline{\text{sfr. bit}}$					
	X. bit	2	X. bit ← $\overline{\text{X. bit}}$					
	A. bit	2	A. bit ← $\overline{\text{A. bit}}$					
	PSWL. bit	2	PSWL. bit ← $\overline{\text{PSW}_{\text{L}}\text{. bit}}$	×	×	×	×	×
	PSWH. bit	2	PSWH. bit ← $\overline{\text{PSW}_{\text{H}}\text{. bit}}$					
	!addr16. bit	5	!addr16. bit ← $\overline{\text{!addr16. bit}}$					
	!!addr24. bit	2	!!addr24. bit ← $\overline{\text{!!addr24. bit}}$					
	mem2. bit	2	mem2. bit ← $\overline{\text{mem2. bit}}$					
	CY	1	CY ← $\overline{\text{CY}}$					×
SET1	saddr. bit	2/3	(saddr. bit) ← 1					
	sfr. bit	3	sfr. bit ← 1					
	X. bit	2	X. bit ← 1					
	A. bit	2	A. bit ← 1					
	PSWL. bit	2	PSWL. bit ← 1	×	×	×	×	×
	PSWH. bit	2	PSWH. bit ← 1					
	!addr16. bit	5	!addr16. bit ← 1					
	!!addr24. bit	2	!!addr24. bit ← 1					
	mem2. bit	2	mem2. bit ← 1					
	CY	1	CY ← 1					1
CLR1	saddr. bit	2/3	(saddr. bit) ← 0					
	sfr. bit	3	sfr. bit ← 0					
	X. bit	2	X. bit ← 0					
	A. bit	2	A. bit ← 0					
	PSWL. bit	2	PSWL. bit ← 0	×	×	×	×	×
	PSWH. bit	2	PSWH. bit ← 0					
	!addr16. bit	5	!addr16. bit ← 0					
	!!addr24. bit	2	!!addr24. bit ← 0					
	mem2. bit	2	mem2. bit ← 0					
	CY	1	CY ← 0					0

(15) Stack manipulation instructions: PUSH, PUSHU, POP, POPU, MOVG, ADDWG, SUBWG, INCG, DECG

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
PUSH	PSW	1	(SP – 2) ← PSW, SP ← SP – 2					
	sfrp	3	(SP – 2) ← sfrp, SP ← SP – 2					
	sfr	3	(SP – 1) ← sfr, SP ← SP – 1					
	post	2	{(SP – 2) ← post, SP ← SP – 2} × m times Note					
	rg	2	(SP – 3) ← rg, SP ← SP – 3					
PUSHU	post	2	{(UUP – 2) ← post, UUP ← UUP – 2} × m times Note					
POP	PSW	1	PSW ← (SP), SP ← SP + 2	R	R	R	R	R
	sfrp	3	sfrp ← (SP), SP ← SP + 2					
	sfr	3	sfr ← (SP), SP ← SP + 1					
	post	2	{post ← (SP), SP ← SP + 2} × m times Note					
	rg	2	rg ← (SP), SP ← SP + 3					
POPU	post	2	{post ← (UUP), UUP ← UUP + 2} × m times Note					
MOVG	SP, # imm24	5	SP ← imm24					
	SP, WHL	2	SP ← WHL					
	WHL, SP	2	WHL ← SP					
ADDWG	SP, #word	4	SP ← SP + word					
SUBWG	SP, #word	4	SP ← SP – word					
INCG	SP	2	SP ← SP + 1					
DECG	SP	2	SP ← SP – 1					

Note m = number of registers specified by “post”

(16) Call/return instructions: CALL, CALLF, CALLT, BRK, BRKCS, RET, RETI, RETB, RETCS, RETCSB

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
CALL	!addr16	3	(SP – 3) ← (PC + 3), SP ← SP – 3, PC_{HW} ← 0, PC_{LW} ← addr16					
	!!addr20	4	(SP – 3) ← (PC + 4), SP ← SP – 3, PC ← addr20					
	rp	2	(SP – 3) ← (PC + 2), SP ← SP – 3, PC_{HW} ← 0, PC_{LW} ← rp					
	rg	2	(SP – 3) ← (PC + 2), SP ← SP – 3, PC ← rg					
	[rp]	2	(SP – 3) ← (PC + 2), SP ← SP – 3, PC_{HW} ← 0, PC_{LW} ← (rp)					
	[rg]	2	(SP – 3) ← (PC + 2), SP ← SP – 3, PC ← (rg)					
	$!addr20	3	(SP – 3) ← (PC + 3), SP ← SP – 3, PC ← PC + 3 + jdisp16					
CALLF	!addr11	2	(SP – 3) ← (PC + 2), SP ← SP – 3, PC_{19-12} ← 0, PC11 ← 1, PC_{10-0} ← addr11					
CALLT	[addr5]	1	(SP – 3) ← (PC + 1), SP ← SP – 3, PC_{HW} ← 0, PC_{LW} ← (addr5)					
BRK		1	(SP – 2) ← PSW, $(SP - 1)_{0-3}$ ← $(PC + 1)_{HW}$, $(SP - 4)$ ← $(PC + 1)_{LW}$, SP ← SP – 4 PC_{HW} ← 0, PC_{LW} ← (003EH)					
BRKCS	RBn	2	PC_{LW} ← RP2, RP3 ← PSW, RBS2 – 0 ← n, RSS ← 0, IE ← 0, $RP3_{8-11}$ ← PC_{HW}, PC_{HW} ← 0					
RET		1	PC ← (SP), SP ← SP + 3					
RET1		1	PC_{LW} ← (SP), PC_{HW} ← $(SP + 3)_{0-3}$, PSW ← (SP + 2), SP ← SP + 4 Clears to 0 flag with highest priority of flags of ISPR that are set (to 1)	R	R	R	R	R
RETB		1	PC_{LW} ← (SP), PC_{HW} ← $(SP + 3)_{0-3}$, PSW ← (SP + 2), SP ← SP + 4	R	R	R	R	R
RETCS	!addr16	3	PSW ← RP3, PC_{LW} ← RP2, RP2 ← addr16, PC_{HW} ← $RP3_{8-11}$ Clears to 0 flag with highest priority of flags of ISPR that are set (to 1)	R	R	R	R	R
RETCSB	!addr16	4	PSW ← RP3, PC_{LW} ← RP2, RP2 ← addr16, PC_{HW} ← $RP3_{8-11}$	R	R	R	R	R

(17) Unconditional branch instruction: BR

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
BR	!addr16	3	$PC_{HW} \leftarrow 0$, $PC_{LW} \leftarrow$ addr16					
	!!addr20	4	PC ← addr20					
	rp	2	$PC_{HW} \leftarrow 0$, $PC_{LW} \leftarrow$ rp					
	rg	2	PC ← rg					
	[rp]	2	$PC_{HW} \leftarrow 0$, $PC_{LW} \leftarrow$ (rp)					
	[rg]	2	PC ← (rg)					
	$addr20	2	PC ← PC + 2 + jdisp8					
	$!addr20	3	PC ← PC + 3 + jdisp16					

(18) Conditional branch instructions: BNZ, BNE, BZ, BE, BNC, BNL, BC, BL, BNV, BPO, BV, BPE, BP, BN, BLT, BGE, BLE, BGT, BNH, BH, BF, BT, BTCLR, BFSET, DBNZ

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
BNZ	$addr20	2	PC ← PC + 2 + jdisp8 if Z = 0					
BNE								
BZ	$addr20	2	PC ← PC + 2 + jdisp8 if Z = 1					
BE								
BNC	$addr20	2	PC ← PC + 2 + jdisp8 if CY = 0					
BNL								
BC	$addr20	2	PC ← PC + 2 + jdisp8 if CY = 1					
BL								
BNV	$addr20	2	PC ← PC + 2 + jdisp8 if P/V = 0					
BPO								
BV	$addr20	2	PC ← PC + 2 + jdisp8 if P/V = 1					
BPE								
BP	$addr20	2	PC ← PC + 2 + jdisp8 if S = 0					
BN	$addr20	2	PC ← PC + 2 + jdisp8 if S = 1					
BLT	$addr20	3	PC ← PC + 3 + jdisp8 if P/V ⊻ S = 1					
BGE	$addr20	3	PC ← PC + 3 + jdisp8 if P/V ⊻ S = 0					
BLE	$addr20	3	PC ← PC + 3 + jdisp8 if (P/V ⊻ S) ∨ Z = 1					
BGT	$addr20	3	PC ← PC + 3 + jdisp8 if (P/V ⊻ S) ∨ Z = 0					
BNH	$addr20	3	PC ← PC + 3 + jdisp8 if Z ∨ CY = 1					
BH	$addr20	3	PC ← PC + 3 + jdisp8 if Z ∨ CY = 0					
BF	saddr. bit, $addr20	4/5	PC ← PC + 4 [Note] + jdisp8 if (saddr. bit) = 0					
	sfr. bit, $addr20	4	PC ← PC + 4 + jdisp8 if sfr. bit = 0					
	X. bit, $addr20	3	PC ← PC + 3 + jdisp8 if X. bit = 0					
	A. bit, $addr20	3	PC ← PC + 3 + jdisp8 if A. bit = 0					
	PSWL. bit, $addr20	3	PC ← PC + 3 + jdisp8 if PSWL. bit = 0					
	PSWH. bit, $addr20	3	PC ← PC + 3 + jdisp8 if PSWH. bit = 0					
	!addr16. bit, $addr20	6	PC ← PC + 3 + jdisp8 if !addr16. bit = 0					
	!!addr24. bit, $addr20	3	PC ← PC + 3 + jdisp8 if !!addr24. bit = 0					
	mem2. bit, $addr20	3	PC ← PC + 3 + jdisp8 if mem2. bit = 0					

Note When the number of bytes is 4. When 5, the operation is: PC ← PC + 5 + jdisp8.

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
BT	saddr. bit, $addr20	3/4	PC ← PC + 3 **Note 1** + jdisp8 if (saddr. bit) = 1					
	sfr. bit, $addr20	4	PC ← PC + 4 + jdisp8 if sfr. bit = 1					
	X. bit, $addr20	3	PC ← PC + 3 + jdisp8 if X. bit = 1					
	A. bit, $addr20	3	PC ← PC + 3 + jdisp8 if A. bit = 1					
	PSWL. bit, $addr20	3	PC ← PC + 3 + jdisp8 if PSW_L. bit = 1					
	PSWH. bit, $addr20	3	PC ← PC + 3 + jdisp8 if PSW_H. bit = 1					
	!addr16. bit, $addr20	6	PC ← PC + 3 + jdisp8 if !addr16. bit = 1					
	!!addr24. bit, $addr20	3	PC ← PC + 3 + jdisp8 if !!addr24. bit = 1					
	mem2. bit, $addr20	3	PC ← PC + 3 + jdisp8 if mem2. bit = 1					
BTCLR	saddr. bit, $addr20	4/5	{PC ← PC + 4 **Note 2** + jdisp8, (saddr. bit) ← 0} if (saddr. bit) = 1					
	sfr. bit, $addr20	4	{PC ← PC + 4 + jdisp8, sfr. bit ← 0} if sfr. bit = 1					
	X. bit, $addr20	3	{PC ← PC + 3 + jdisp8, X. bit ← 0} if X. bit = 1					
	A. bit, $addr20	3	{PC ← PC + 3 + jdisp8, A. bit ← 0} if A. bit = 1					
	PSWL. bit, $addr20	3	{PC ← PC + 3 + jdisp8, PSW_L. bit ← 0} if PSW_L. bit = 1	×	×	×	×	×
	PSWH. bit, $addr20	3	{PC ← PC + 3 + jdisp8, PSW_H. bit ← 0} if PSW_H. bit = 1					
	!addr16. bit, $addr20	6	{PC ← PC + 3 + jdisp8, !addr16. bit ← 0} if !addr16. bit = 1					
	!!addr24. bit, $addr20	3	{PC ← PC + 3 + jdisp8, !!addr24. bit ← 0} if !!addr24. bit = 1					
	mem2. bit, $addr20	3	{PC ← PC + 3 + jdisp8, mem2. bit ← 0} if mem2. bit = 1					

Notes 1. When the number of bytes is 3. When 4, the operation is: PC ← PC + 4 + jdisp8.

2. When the number of bytes is 4. When 5, the operation is: PC ← PC + 5 + jdisp8.

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
BFSET	saddr. bit, $addr20	4/5	{PC ← PC + 4 Note 1 + jdisp8, (saddr. bit) ← 1} if (saddr. bit) = 0					
	sfr. bit, $addr20	4	{PC ← PC + 4 + jdisp8, sfr. bit ← 1} if sfr. bit = 0					
	X. bit, $addr20	3	{PC ← PC + 3 + jdisp8, X. bit ← 1} if X. bit = 0					
	A. bit, $addr20	3	{PC ← PC + 3 + jdisp8, A. bit ← 1} if A. bit = 0					
	PSWL. bit, $addr20	3	{PC ← PC + 3 + jdisp8, PSW_L. bit ← 1} if PSW_L. bit = 0	×	×	×	×	×
	PSWH. bit, $addr20	3	{PC ← PC + 3 + jdisp8, PSW_H. bit ← 1} if PSW_H. bit = 0					
	!addr16. bit, $addr20	6	{PC ← PC + 3 + jdisp8, !addr16. bit ← 1} if !addr16. bit = 0					
	!!addr24. bit, $addr20	3	{PC ← PC + 3 + jdisp8, !!addr24. bit ← 1} if !!addr24. bit = 0					
	mem2. bit, $addr20	3	{PC ← PC + 3 + jdisp8, mem2. bit ← 1} if mem2. bit = 0					
DBNZ	B, $addr20	2	B ← B – 1, PC ← PC + 2 + jdisp8 if B ≠ 0					
	C, $addr20	2	C ← C – 1, PC ← PC + 2 + jdisp8 if C ≠ 0					
	$addr, $addr20	3/4	(saddr) ← (saddr) – 1, PC ← PC + 3 Note 2 = jdisp8 if (saddr) ≠ 0					

Notes 1. When the number of bytes is 4. When 5, the operation is: PC ← PC + 5 + jdisp8.

2. When the number of bytes is 3. When 4, the operation is: PC ← PC + 4 + jdisp8.

(19) CPU control instructions: MOV, LOCATION, SEL, SWRS, NOP, EI, DI

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
MOV	STBC, #byte	4	STBC ← byte					
	WDM, #byte	4	WDM ← byte					
LOCATION	locaddr	4	SFR, internal data area location address upper word specification					
SEL	RBn	2	RSS ← 0, RBS2 – 0 ← n					
	RBn, ALT	2	RSS ← 1, RBS2 – 0 ← n					
SWRS		2	RSS ← $\overline{\text{RSS}}$					
NOP		1	No Operaton					
EI		1	IE ← 1 (Enable interrupt)					
DI		1	IE ← 0 (Disable interrupt)					

(20) Special instructions: CHKL, CHKLA

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
CHKL	sfr	3	(Pin level) ∀ (output latch)	×	×		P	
CHKLA	sfr	3	A ← (pin level) ∀ (output latch)	×	×		P	

(21) String instructions: MOVTBLW, MOVM, XCHM, MOVBK, XCHBK, CMPME, CMPMNE, CMPMC, CMPMNC, CMPBKE, CMPBKNE, CMPBKC, CMPBKNC

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
MOVTBLW	!addr8, byte	4	(addr8 + 2) ← (addr8), byte ← byte – 1, addr8 ← addr8 – 2 End if byte = 0					
MOVW	[TDE +], A	2	(TDE) ← A, TDE ← TDE + 1, C ← C – 1 End if C = 0					
	[TDE –], A	2	(TDE) ← A, TDE ← TDE – 1, C ← C – 1 End if C = 0					
XCHM	[TDE +], A	2	(TDE) ↔ A, TDE ← TDE + 1, C ← C – 1 End if C = 0					
	[TDE –], A	2	(TDE) ↔ A, TDE ← TDE – 1, C ← C – 1 End if C = 0					
MOVBK	[TDE +], [WHL +]	2	(TDE) ← (WHL), TDE ← TDE + 1, WHL ← WHL + 1, C ← C – 1 End if C = 0					
	[TDE –], [WHL –]	2	(TDE) ← (WHL), TDE ← TDE – 1, WHL ← WHL – 1, C ← C – 1 End if C = 0					
XCHBK	[TDE +], [WHL +]	2	(TDE) ↔ (WHL), TDE ← TDE +1, WHL ← WHL + 1, C ← C – 1 End if C = 0					
	[TDE –], [WHL –]	2	(TDE) ↔ (WHL), TDE ← TDE – 1, WHL ← WHL – 1, C ← C – 1 End if C = 0					
CMPME	[TDE +], A	2	(TDE) – A, TDE ← TDE + 1, C ← C – 1 End if C = 0 or Z = 0	×	×	×	V	×
	[TDE –], A	2	(TDE) – A, TDE ← TDE – 1, C ← C – 1 End if C = 0 or Z = 0	×	×	×	V	×
CMPMNE	[TDE +], A	2	(TDE) – A, TDE ← TDE + 1, C ← C – 1 End if C = 0 or Z = 1	×	×	×	V	×
	[TDE –], A	2	(TDE) – A, TDE ← TDE – 1, C ← C – 1 End if C = 0 or Z = 1	×	×	×	V	×
CMPMC	[TDE +], A	2	(TDE) – A, TDE ← TDE + 1, C ← C – 1 End if C = 0 or CY = 0	×	×	×	V	×
	[TDE –], A	2	(TDE) – A, TDE ← TDE – 1, C ← C – 1 End if C = 0 or CY = 0	×	×	×	V	×
CMPMNC	[TDE +], A	2	(TDE) – A, TDE ← TDE + 1, C ← C – 1 End if C = 0 or CY = 1	×	×	×	V	×
	[TDE –], A	2	(TDE) – A, TDE ← TDE – 1, C ← C – 1 End if C = 0 or CY = 1	×	×	×	V	×
CMPBKE	[TDE +], [WHL +]	2	(TDE) ← (WHL), TDE ← TDE + 1, WHL ← WHL + 1, C ← C – 1 End if C = 0 or Z = 0	×	×	×	V	×
	[TDE –], [WHL –]	2	(TDE) ← (WHL), TDE ← TDE – 1, WHL ← WHL – 1, C ← C – 1 End if C = 0 or Z = 0	×	×	×	V	×

Mnemonic	Operands	Bytes	Operation	Flags				
				S	Z	AC	P/V	CY
CMPBKNE	[TDE +], [WHL +]	2	(TDE) – (WHL), TDE ← TDE + 1, WHL ← WHL + 1, C ← C – 1 End if C = 0 or Z = 1	×	×	×	V	×
	[TDE –], [WHL –]	2	(TDE) – (WHL), TDE ← TDE – 1, WHL ← WHL – 1, C ← C – 1 End if C = 0 or Z = 1	×	×	×	V	×
CMPBKC	[TDE +], [WHL +]	2	(TDE) – (WHL), TDE ← TDE + 1, WHL ← WHL + 1, C ← C – 1 End if C = 0 or CY = 0	×	×	×	V	×
	[TDE –], [WHL –]	2	(TDE) – (WHL), TDE ← TDE – 1, WHL ← WHL – 1, C ← C – 1 End if C = 0 or CY = 0	×	×	×	V	×
CMPBKNC	[TDE +], [WHL +]	2	(TDE) – (WHL), TDE ← TDE + 1, WHL ← WHL + 1, C ← C – 1 End if C = 0 or CY = 1	×	×	×	V	×
	[TDE –], [WHL –]	2	(TDE) – (WHL), TDE ← TDE – 1, WHL ← WHL – 1, C ← C – 1 End if C = 0 or CY = 1	×	×	×	V	×

27.3 INSTRUCTIONS LISTED BY TYPE OF ADDRESSING

(1) 8-bit instructions (combinations expressed by writing A for r are shown in parentheses)

MOV, XCH, ADD, ADDC, SUB, SUBC, AND OR XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, SHR, SHL, ROR4, ROL4, DBNZ, PUSH, POP, MOVM, XCHM, CMPME, CMPMNE, CMPMNC, CMPMC, MOVBK, XCHBK, CMPBKE, CMPBKNE, CMPBKNC, CMPBKC, CHKL, CHKLA

Table 27-1 List of Instructions by 8-Bit Addressing (1/2)

2nd Operand / 1st Operand	# byte	A	r r'	saddr saddr'	sfr
A	(MOV) ADD Note 1	(MOV) (XCH) (ADD) Note 1	MOV XCH (ADD) Note 1	(MOV) Note 6 (XCH) Note 6 (ADD) Notes 1, 6	MOV (XCH) (ADD) Note 1
r	MOV ADD Note 1	(MOV) (XCH) (ADD) Note 1	MOV XCH ADD Note 1	MOV XCH ADD Note 1	MOV XCH ADD Note 1
saddr	MOV ADD Note 1	(MOV) Note 6 (ADD) Note 1	MOV ADD Note 1	MOV XCH ADD Note 1	
sfr	MOV ADD Note 1	MOV (ADD) Note 1	MOV ADD Note 1		
!addr16 !!addr24	MOV	(MOV) ADD Note 1	MOV		
mem [saddrp] [%saddrg]		MOV ADD Note 1			
mem3					
r3 PSWL PSWH	MOV	MOV			
B, C					
STBC, WDM	MOV				
[TDE +] [TDE –]		(MOV) (ADD) Note 1 MOVM Note 4			

(See the following page for the explanation of **Note**.)

Table 27-1 List of Instructions by 8-Bit Addressing (2/2)

2nd Operand / 1st Operand	!addr16 !!addr24	mem [saddrp] [%saddrg]	r3 PSWL PSWH	[WHL +] [WHL –]	n	None Note 2
A	(MOV) (XCH) ADD Note 1	MOV XCH ADD Note 1	MOV	(MOV) (XCH) (ADD) Note 1		
r	MOV XCH				ROR Note 3	MULU DIVUW INC DEC
saddr						INC DEC DBNZ
sfr						PUSH POP CHKL CHKLA
!addr16 !!addr24						
mem [saddrp] [%saddrg]						
mem3						ROR4 ROL4
r3 PSWL PSWH						
B, C						DBNZ
STBC, WDM						
[TDE +] [TDE –]				MOVBK Note 5		

Notes 1. ADDC, SUB, SUBC, AND, OR, XOR and CMP are the same as ADD.

2. There is no 2nd operand, or the 2nd operand is not an operand address.

3. ROL, RORC, ROLC, SHR and SHL are the same as ROR.

4. XCHM, CMPME, CMPMNE, CMPMNC and CMPMC are the same as MOVM.

5. XCHBK, CMPBKE, CMPBKNE, CMPBKNC and CMPBKC are the same as MOVBK.

6. If saddr is saddr2 in this combination, there is a short code length instruction.

(2) 16-bit instructions (combinations expressed by writing AX for rp are shown in parentheses)

MOVM, XCHW, ADDW, SUBW, CMPW, MULUW, MULW, DIVUX, INCW, DECW, SHRW, SHLW, PUSH, POP, ADDWG, SUBWG, PUSHU, POPU, MOVTBLW, MACW, MACSW, SACW

Table 27-2 List of Instructions by 16-Bit Addressing (1/2)

2nd Operand / 1st Operand	# word	A	r r'	saddr saddr'	sfrp
AX	(MOVW) ADDW Note 1	(MOVW) (XCHW) (ADD) Note 1	(MOVW) (XCHW) (ADDW) Note 1	(MOVW) Note 3 (XCHW) Note 3 (ADDW) Notes 1,3	MOVW (XCHW) (ADDW) Note 1
rp	MOVW ADDW Note 1	(MOVW) (XCHW) (ADDW) Note 1	MOVW XCHW ADDW Note 1	MOVW XCHW ADDW Note 1	MOVW XCHW ADDW Note 1
saddrp	MOVW ADDW Note 1	(MOVW) Note 3 (ADDW) Note 1	MOVW ADDW Note 1	MOVW XCHW ADDW Note 1	
sfrp	MOVW ADDW Note 1	MOVW (ADDW) Note 1	MOVW ADDW Note 1		
!addr16 !!addr24	MOVW	(MOVW)	MOVW		
mem [saddrp] [%saddrg]		MOVW			
PSW					
SP	ADDWG SUBWG				
post					
[TDE +]		(MOVW)			
byte					

(See the following page for the explanation of **Note**.)

Table 27-2 List of Instructions by 16-Bit Addressing (2/2)

1st Operand \ 2nd Operand	!!addr16 !!addr24	mem [saddrp] [%saddrg]	[WHL +]	byte	n	None Note 2
AX	(MOVW) XCHW	MOVW XCHW	(MOVW) (XCHW)			
rp	MOVW				SHRW SHLW	MULW Note 4 INCW DECW
saddrp						INCW DECW
sfrp						PUSH POP
!addr16 !!addr24					MOVTBLW	
mem [saddrp] [%saddrg]						
PSW						PUSH POP
SP						
post						PUSH POP PUSHU POPU
[TDE +]			SACW			
byte						MACW MACSW

Notes 1. SUBW and CMPW are the same as ADDW.

2. There is no 2nd operand, or the 2nd operand is not an operand address.

3. If saddrp is saddrp2 in this combination, there is a short code length instruction.

4. MULUW and DIVUX are the same as MULW.

(3) 24-bit instructions (combinations expressed by writing WHL for rg are shown in parentheses)

MOVG, ADDG, SUBG, INCG, DECG, PUSH, POP

Table 27-3 List of Instructions by 24-Bit Addressing

1st Operand \ 2nd Operand	# imm24	WHL	rg rg'	saddrg	!!addr24	mem1	[%saddrg]	SP	None Note
WHL	(MOVG) (ADDG) (SUBG)	(MOVG) (ADDG) (SUBG)	(MOVG) (ADDG) (SUBG)	(MOVG) ADDG SUBG	(MOVG)	MOVG	MOVG	MOVG	
rg	MOVG ADDG SUBG	(MOVG) (ADDG) (SUBG)	MOVG ADDG SUBG	MOVG	MOVG				INCG DECG PUSH POP
saddrg		(MOVG)	MOVG						
!!addr24		(MOVG)	MOVG						
mem1		MOVG							
[%saddrg)		MOVG							
SP	MOVG	MOVG							INCG DECG

Note There is no 2nd operand, or the 2nd operand is not an operand address.

(4) Bit manipulation instructions

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR, BFSET

Table 27-4 List of Instructions by Bit Manipulation Instruction Addressing

1st Operand \ 2nd Operand	CY	saddr. bit sfr. bit A.bit X. bit PSWL. bit PSWH. bit mem2. bit !addr16. bit !!addr24. bit	/saddr.bit /sfr. bit /A. bit /X. bit /PSWL. bit /PSWH. bit /mem2. bit /!addr16. bit /!!addr24. bit	None Note
CY		MOV1 AND1 OR1 XOR1	AND1 SET1	NOT1 SET1 CLR1
saddr. bit sfr. bit A. bit X. bit PSWL. bit PSWH. bit mem2. bit !addr16. bit !!addr24. bit	MOV1			NOT1 SET1 CLR1 BF BT BTCLR BFSET

Note There is no 2nd operand, or the 2nd operand is not an operand address.

(5) Call/return instructions / branch instructions

CALL, CALLF, CALLT, BRK, RET, RETI, RETB, RETCS, RETCSB, BRKCS, BR, BNZ, BNE, BZ, BE, BNC, BNL, BC, BL, BNV, BPO, BV, BPE, BP, BN, BLT, BGE, BLE, BGT, BNH, BH, BF, BT, BTCLR, BFSET, DBNZ

Table 27-5 List of Instructions by Call/Return Instruction / Branch Instruction Addressing

Instruction Address Operand	$addr20	$!addr20	!addr16	!!addr20	rp	rg	[rp]	[rg]	!addr11	[addr5]	RBn	None
Basic instructions	BC [Note] BR	CALL BR	CALL BR RETCSB	CALL BR RETCS	CALL BR	CALL BR	CALL BR	CALL BR	CALLF	CALLT	BRKCS	BRK RET RETI RETB
Compound instructions	BF BT BTCLR BFSET DBNZ											

Note BNZ, BNE, BZ, BE, BNC, BNL, BL, BNV, BPO, BV, BPE, BP, BN, BLT, BGE, BLE, BGT, BNH, and BH are the same as BC.

(6) Other instructions

ADJBA, ADJBS, CVTBW, LOCATION, SEL, NOT, EI, DI, SWRS

★

CHAPTER 28 ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS (T_A = 25°C)

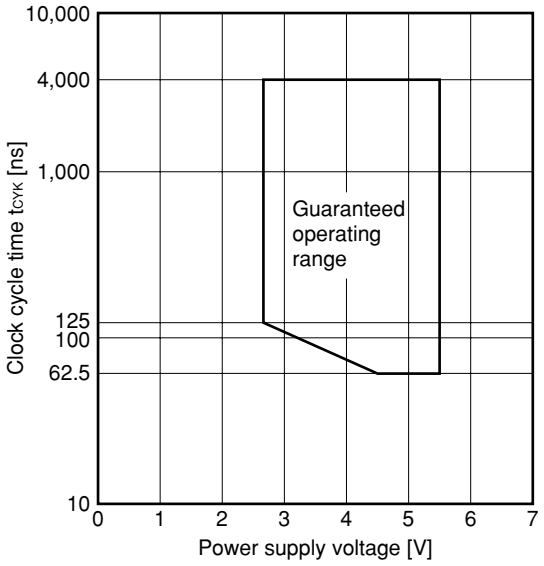
Parameter	Symbol	Conditions	Rating	Unit
Supply voltage	V_{DD}		–0.5 to +7.0	V
	AV_{DD}		AV_{SS} to V_{DD} + 0.5	V
	AV_{SS}		–0.5 to +0.5	V
Input voltage	V_{I1}		–0.5 to V_{DD} + 0.5	V
	V_{I2}	μPD78P4038, 78P4038Y only. TEST/V_{PP} pin and P21/INTP0/A9 pin in PROM programming mode	–0.5 to +13.5	V
Output voltage	V_O		–0.5 to V_{DD} + 0.5	V
Output low current	I_{OL}	At one pin	15	mA
		Total of all output pins	100	mA
Output high current	I_{OH}	At one pin	–10	mA
		Total of all output pins	–100	mA
A/D converter reference input voltage	AV_{REF1}		–0.5 to V_{DD} + 0.3	V
D/A converter reference input voltage	AV_{REF2}		–0.5 to V_{DD} + 0.3	V
	AV_{REF3}		–0.5 to V_{DD} + 0.3	V
Operating ambient temperature	T_A		–40 to +85	°C
Storage temperature	T_{stg}		–65 to +150	°C

Caution Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

OPERATING CONDITIONS

- Operating ambient temperature (T_A) : –40 to +85°C
- Rise time and fall time (t_r, t_f) (at pins which are not specified) : 0 to 200 μs
- Power supply voltage and clock cycle time : See **Figure 28-1**.

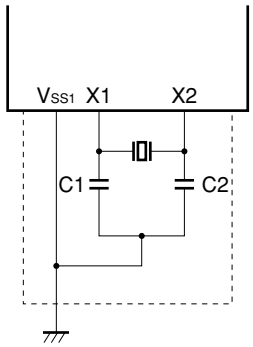
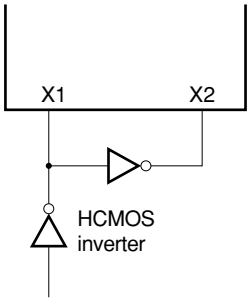
Figure 28-1 Power Supply Voltage and Clock Cycle Time



CAPACITANCE (T_A = 25°C, V_{DD} = V_{SS} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Input capacitance	C_I	f = 1 MHz 0 V on pins other than measured pins			10	pF
Output capacitance	C_O				10	pF
I/O capacitance	C_{IO}				10	pF

OSCILLATOR CHARACTERISTICS (T_A = –40 to +85°C, V_{DD} = +4.5 to 5.5 V, V_{SS} = 0 V)

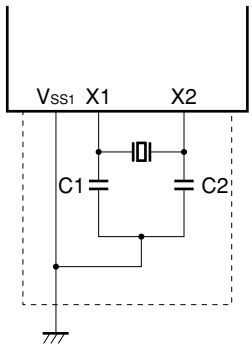
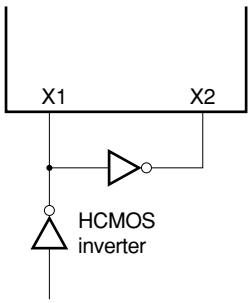
Resonator	Recommended Circuit	Parameter	MIN.	MAX.	Unit
Ceramic or crystal resonator		Oscillator frequency (f_{XX})	4	32	MHz
External clock		X1 input frequency (f_X)	4	32	MHz
		X1 input rise and fall times (t_{XR}, t_{XF})	0	10	ns
		X1 input high-level and low-level widths (t_{WXH}, t_{WXL})	10	125	ns

Caution When using the main clock oscillator, wire as follows in the area enclosed by the broken lines in the above figures to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS1}.**
- **Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

Remark For the resonator selection and oscillator constant, customers are required to either evaluate the oscillation themselves or apply to the resonator manufacturer for evaluation.

OSCILLATOR CHARACTERISTICS (T_A = –40 to +85°C, V_{DD} = +2.7 to 5.5 V, V_{SS} = 0 V)

Resonator	Recommended Circuit	Parameter	MIN.	MAX.	Unit
Ceramic resonator or crystal		Oscillator frequency (f_{XX})	4	16	MHz
External clock		X1 input frequency (f_X)	4	16	MHz
		X1 input rise and fall times (t_{XR}, t_{XF})	0	10	ns
		X1 input high-level and low-level widths (t_{WXH}, t_{WXL})	10	125	ns

Caution When using the main clock oscillator, wire as follows in the area enclosed by the broken lines in the above figures to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS1}.**
- **Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

Remark For the resonator selection and oscillator constant, customers are required to either evaluate the oscillation themselves or apply to the resonator manufacturer for evaluation.

DC CHARACTERISTICS (T_A = –40 to +85°C, V_{DD} = AV_{DD} = +2.7 to 5.5 V, V_{SS} = AV_{SS} = 0 V) (1/2)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Input low voltage	V_{IL1}	For pins other than those described in **Notes 1**, **2**, **3**, **4**, and **6**	–0.3		0.3V_{DD}	V
	V_{IL2}	For pins described in **Notes 1**, **2**, **3**, **4**, and **6**	–0.3		0.2V_{DD}	V
	V_{IL3}	V_{DD} = +5.0 V ±10% For pins described in **Notes 2**, **3**, and **4**	–0.3		+0.8	V
Input high voltage	V_{IH1}	For pins other than those described in **Notes 1** and **6**	0.7V_{DD}		V_{DD} + 0.3	V
	V_{IH2}	For pins described in **Notes 1** and **6**	0.8V_{DD}		V_{DD} + 0.3	V
	V_{IH3}	V_{DD} = +5.0 V ±10% For pins described in **Notes 2**, **3**, and **4**	2.2		V_{DD} + 0.3	V
Output low voltage	V_{OL1}	I_{OL} = 2 mA For pins other than those described in **Note 6**			0.4	V
	V_{OL2}	I_{OL} = 3 mA For pins described in **Note 6**			0.4	V
		I_{OL} = 6 mA For pins described in **Note 6**			0.6	V
	V_{OL3}	V_{DD} = +5.0 V ±10% I_{OL} = 8 mA For pins described in **Notes 2** and **5**			1.0	V
Output high voltage	V_{OH1}	I_{OH} = –2 mA	V_{DD} – 1.0			V
	V_{OH2}	V_{DD} = +5.0 V ±10% I_{OH} = –5 mA For pins described in **Note 4**	V_{DD} – 1.4			V
X1 input low current	I_{IL}	EXTC = 0 0 V ≤ V_I ≤ V_{IL2}			–30	μA
X1 input high current	I_{IH}	EXTC = 0 V_{IH2} ≤ V_I ≤ V_{DD}			+30	μA

Notes 1. μPD784038 Subseries: X1, X2, $\overline{\text{RESET}}$, P12/ASCK2/SCK2, P20/NMI, P21/INTP0, P22/INTP1, P23/INTP2/CI, P24/INTP3, P25/INTP4/ASCK/$\overline{\text{SCK1}}$, P26/INTP5, P27/SI0, P32/$\overline{\text{SCK0}}$/SCL, P33/SO0/SDA,TEST
μPD784038Y Subseries: X1, X2, $\overline{\text{RESET}}$, P12/ASCK2/$\overline{\text{SCK2}}$, P20/NMI, P21/INTP0, P22/INTP1, P23/INTP2/CI, P24/INTP3, P25/INTP4/ASCK/$\overline{\text{SCK1}}$, P26/INTP5, P27/SI0,TEST

2. μPD784031, 784031Y, 784031(A): AD0 to AD7, A8 to A15
Other: P40/AD0 to P47/AD7, P50/A8 to P57/A15

3. μPD784031, 784031Y, 784031(A): P60/A16 to P63/A19, $\overline{\text{RD}}$, $\overline{\text{WR}}$, P66/$\overline{\text{WAIT}}$/HLDRQ, P67/$\overline{\text{REFRQ}}$/HLDAK
Other: P60/A16 to P63/A19, P64/$\overline{\text{RD}}$, P65/$\overline{\text{WR}}$, P66/$\overline{\text{WAIT}}$/HLDRQ, P67/$\overline{\text{REFRQ}}$/HLDAK

4. P00 to P07

5. P10 to P17

6. P32/$\overline{\text{SCK0}}$/SCL, P33/SO0/SDA (μPD784038Y Subseries only)

DC CHARACTERISTICS (T_A = –40 to +85°C, V_{DD} = AV_{DD} = +2.7 to 5.5 V, V_{SS} = AV_{SS} = 0 V) (2/2)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Input leakage current	I_{LI}	0 V ≤ V_I ≤ V_{DD} For pins other than X1 when EXTC = 0				±10	μA
Output leakage current	I_{LO}	0 V ≤ V_O ≤ V_{DD}				±10	μA
V_{DD} supply current	I_{DD1}	Operation mode	f_{XX} = 32 MHz V_{DD} = +5.0 V ±10%		25	45	mA
			f_{XX} = 16 MHz V_{DD} = +2.7 to 3.3 V		12	25	mA
	I_{DD2}	HALT mode	f_{XX} = 32 MHz V_{DD} = +5.0 V ±10%		13	26	mA
			f_{XX} = 16 MHz V_{DD} = +2.7 to 3.3 V		8	12	mA
	I_{DD3}	IDLE mode (EXTC = 0)	f_{XX} = 32 MHz V_{DD} = +5.0 V ±10%			12	mA
			f_{XX} = 16 MHz V_{DD} = +2.7 to 3.3 V			8	mA
Pull-up resistor	R_L	V_I = 0 V		15		80	kΩ

AC CHARACTERISTICS (T_A = –40 to +85°C, V_{DD} = AV_{DD} = +2.7 to 5.5 V, V_{SS} = AV_{SS} = 0 V)

(1) Read/write operation (1/2)

Parameter	Symbol	Conditions		MIN.	MAX.	Unit
Address setup time	t_{SAST}	V_{DD} = +5.0 V ±10%		(0.5 + a) T – 15		ns
		V_{DD} = +2.7 to 5.5 V		(0.5 + a) T – 31		ns
ASTB high-level width	t_{WSTH}	V_{DD} = +5.0 V ±10%		(0.5 + a) T – 17		ns
		V_{DD} = +2.7 to 5.5 V		(0.5 + a) T – 40		ns
Address hold time (from ASTB↓)	t_{HSTLA}	V_{DD} = +5.0 V ±10%		0.5T – 24		ns
		V_{DD} = +2.7 to 5.5 V		0.5T – 34		ns
Address hold time (from $\overline{RD}$↑)	t_{HRA}			0.5T – 14		ns
Delay from address to $\overline{RD}$↓	t_{DAR}	V_{DD} = +5.0 V ±10%		(1 + a) T – 9		ns
		V_{DD} = +2.7 to 5.5 V		(1 + a) T – 15		ns
Address float time (from $\overline{RD}$↓)	t_{FRA}				0	ns
Delay from address to data input	t_{DAID}	V_{DD} = +5.0 V ±10%			(2.5 + a + n) T – 37	ns
		V_{DD} = +2.7 to 5.5 V			(2.5 + a + n) T – 52	ns
Delay from ASTB↓ to data input	t_{DSTID}	V_{DD} = +5.0 V ±10%			(2 + n) T – 40	ns
		V_{DD} = +2.7 to 5.5 V			(2 + n) T – 60	ns
Delay from $\overline{RD}$↓ to data input	t_{DRID}	V_{DD} = +5.0 V ±10%			(1.5 + n) T – 50	ns
		V_{DD} = +2.7 to 5.5 V			(1.5 + n) T – 70	ns
Delay from ASTB↓ to $\overline{RD}$↓	t_{DSTR}			0.5T – 9		ns
Data hold time (from $\overline{RD}$↑)	t_{HRID}			0		ns
Delay from $\overline{RD}$↑ to address active	t_{DRA}	After program is read	V_{DD} = +5.0 V ±10%	0.5T – 8		ns
			V_{DD} = +2.7 to 5.5 V	0.5T – 12		ns
		After data is read	V_{DD} = +5.0 V ±10%	1.5T – 8		ns
			V_{DD} = +2.7 to 5.5 V	1.5T – 12		ns
Delay from $\overline{RD}$↑ to ASTB↑	t_{DRST}			0.5T – 17		ns
$\overline{RD}$ low-level width	t_{WRL}	V_{DD} = +5.0 V ±10%		(1.5 + n) T – 30		ns
		V_{DD} = +2.7 to 5.5 V		(1.5 + n) T – 40		ns
Address hold time (from $\overline{WR}$↑)	t_{HWA}			0.5T – 14		ns
Delay from address to $\overline{WR}$↓	t_{DAW}	V_{DD} = +5.0 V ±10%		(1 + a) T – 5		ns
		V_{DD} = +2.7 to 5.5 V		(1 + a) T – 15		ns
Delay from ASTB↓ to data output	t_{DSTOD}	V_{DD} = +5.0 V ±10%			0.5T + 19	ns
		V_{DD} = +2.7 to 5.5 V			0.5T + 35	ns
Delay from $\overline{WR}$↓ to data output	t_{DWOD}				0.5T – 11	ns
Delay from ASTB↓ to $\overline{WR}$↓	t_{DSTW}			0.5T – 9		ns

Remarks T: t_{CYK} (system clock cycle time)

a: 1 (during address wait); otherwise 0

n: Number of wait states (n ≥ 0)

(1) Read/write operation (2/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Data setup time (to $\overline{WR}\uparrow$)	t_{SODW}	V_{DD} = +5.0 V ±10%	(1.5 + n) T – 30		ns
		V_{DD} = +2.7 to 5.5 V	(1.5 + n) T – 40		ns
Data hold time (from $\overline{WR}\uparrow$)**Note**	t_{HWOD}	V_{DD} = +5.0 V ±10%	0.5T – 5		ns
		V_{DD} = +2.7 to 5.5 V	0.5T – 25		ns
Delay from $\overline{WR}\uparrow$ to ASTB↑	t_{DWST}		0.5T – 12		ns
$\overline{WR}$ low-level width	t_{WWL}	V_{DD} = +5.0 V ±10%	(1.5 + n) T – 30		ns
		V_{DD} = +2.7 to 5.5 V	(1.5 + n) T – 40		ns

Note The hold time includes the time during which V_{OH1} and V_{OL1} are held under the load conditions of C_L = 50 pF and R_L = 4.7 kΩ.

Remarks T: t_{CYK} (system clock cycle time)
n: Number of wait states (n ≥ 0)

(2) Bus hold timing

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Delay from HLDRQ↑ to float	t_{FHQC}			(6 + a + n) T + 50	ns
Delay from HLDRQ↑ to HLDAK↑	$t_{DHQHHAH}$	V_{DD} = +5.0 V ±10%		(7 + a + n) T + 30	ns
		V_{DD} = +2.7 to 5.5 V		(7 + a + n) T + 40	ns
Delay from float to HLDAK↑	t_{DCFHA}			1T + 30	ns
Delay from HLDRQ↓ to HLDAK↓	$t_{DHQLHAL}$	V_{DD} = +5.0 V ±10%		2T + 40	ns
		V_{DD} = +2.7 to 5.5 V		2T + 60	ns
Delay from HLDAK↓ to active	t_{DHAC}	V_{DD} = +5.0 V ±10%	1T – 20		ns
		V_{DD} = +2.7 to 5.5 V	1T – 30		ns

Remarks T: t_{CYK} (system clock cycle time)
a: 1 (during address wait); otherwise 0
n: Number of wait states (n ≥ 0)

(3) External wait timing

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Delay from address to $\overline{WAIT}\downarrow$ input	t_{DAWT}	V_{DD} = +5.0 V ±10%		(2 + a) T – 40	ns
		V_{DD} = +2.7 to 5.5 V		(2 + a) T – 60	ns
Delay from ASTB↓ to $\overline{WAIT}\downarrow$ input	t_{DSTWT}	V_{DD} = +5.0 V ±10%		1.5T – 40	ns
		V_{DD} = +2.7 to 5.5 V		1.5T – 60	ns
Hold time from ASTB↓ to $\overline{WAIT}$	t_{HSTWTH}	V_{DD} = +5.0 V ±10%	(0.5 + n) T + 5		ns
		V_{DD} = +2.7 to 5.5 V	(0.5 + n) T +10		ns
Delay from ASTB↓ to $\overline{WAIT}\uparrow$	t_{DSTWTH}	V_{DD} = +5.0 V ±10%		(1.5 + n) T – 40	ns
		V_{DD} = +2.7 to 5.5 V		(1.5 + n) T – 60	ns
Delay from $\overline{RD}\downarrow$ to $\overline{WAIT}\downarrow$ input	t_{DRWTL}	V_{DD} = +5.0 V ±10%		T – 50	ns
		V_{DD} = +2.7 to 5.5 V		T – 70	ns
Hold time from $\overline{RD}\downarrow$ to $\overline{WAIT}$	t_{HRWT}	V_{DD} = +5.0 V ±10%	nT + 5		ns
		V_{DD} = +2.7 to 5.5 V	nT + 10		ns
Delay from $\overline{RD}\downarrow$ to $\overline{WAIT}\uparrow$	t_{DRWTH}	V_{DD} = +5.0 V ±10%		(1 + n) T – 40	ns
		V_{DD} = +2.7 to 5.5 V		(1 + n) T – 60	ns
Delay from $\overline{WAIT}\uparrow$ to data input	t_{DWTID}	V_{DD} = +5.0 V ±10%		0.5T – 5	ns
		V_{DD} = +2.7 to 5.5 V		0.5T – 10	ns
Delay from $\overline{WAIT}\uparrow$ to $\overline{WR}\uparrow$	t_{DWTW}		0.5T		ns
Delay from $\overline{WAIT}\uparrow$ to $\overline{RD}\uparrow$	t_{DWTR}		0.5T		ns
Delay from $\overline{WR}\downarrow$ to $\overline{WAIT}\downarrow$ input	t_{DWWTL}	V_{DD} = +5.0 V ±10%		T – 50	ns
		V_{DD} = +2.7 to 5.5 V		T – 75	ns
Hold time from $\overline{WR}\downarrow$ to $\overline{WAIT}$	t_{HWWT}	V_{DD} = +5.0 V ±10%	nT + 5		ns
		V_{DD} = +2.7 to 5.5 V	nT + 10		ns
Delay from $\overline{WR}\downarrow$ to $\overline{WAIT}\uparrow$	t_{DWWTH}	V_{DD} = +5.0 V ±10%		(1 + n) T – 40	ns
		V_{DD} = +2.7 to 5.5 V		(1 + n) T – 70	ns

Remarks T: t_{CYK} (system clock cycle time)

a: 1 (during address wait); otherwise 0

n: Number of wait states ($n \geq 0$)

(4) Refresh timing

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Random read/write cycle time	t_{RC}		3T		ns
$\overline{REFRQ}$ low-level pulse width	t_{WRFQL}	V_{DD} = +5.0 V ±10%	1.5T – 25		ns
		V_{DD} = +2.7 to 5.5 V	1.5T – 30		ns
Delay from ASTB↓ to $\overline{REFRQ}$	t_{DSTRFQ}		0.5T – 9		ns
Delay from $\overline{RD}\uparrow$ to $\overline{REFRQ}$	t_{DRRFQ}		1.5T – 9		ns
Delay from $\overline{WR}\uparrow$ to $\overline{REFRQ}$	t_{DWRFQ}		1.5T – 9		ns
Delay from $\overline{REFRQ}\uparrow$ to ASTB	t_{DRFQST}		0.5T – 15		ns
$\overline{REFRQ}$ high-level pulse width	t_{WRFQH}	V_{DD} = +5.0 V ±10%	1.5T – 25		ns
		V_{DD} = +2.7 to 5.5 V	1.5T – 30		ns

Remark T: t_{CYK} (system clock cycle time)

SERIAL OPERATION (T_A = –40 to +85°C, V_{DD} = +2.7 to 5.5 V, AV_{SS} = V_{SS} = 0 V)

(1) CSI

Parameter	Symbol	Conditions		MIN.	MAX.	Unit
Serial clock cycle time ($\overline{SCK0}$)	t_{CYSK0}	Input	External clock When $\overline{SCK0}$ and SO0 are CMOS I/O	10/f_{XX} + 380		ns
		Output		T		μs
Serial clock low-level width ($\overline{SCK0}$)	t_{WSKL0}	Input	External clock When $\overline{SCK0}$ and SO0 are CMOS I/O	5/f_{XX} + 150		ns
		Output		0.5T – 40		μs
Serial clock high-level width ($\overline{SCK0}$)	t_{WSKH0}	Input	External clock When $\overline{SCK0}$ and SO0 are CMOS I/O	5/f_{XX} + 150		ns
		Output		0.5T – 40		μs
SI0 setup time (to $\overline{SCK0}$↑)	t_{SSSK0}			40		ns
SI0 hold time (to $\overline{SCK0}$↑)	t_{HSSK0}			5/f_{XX} + 40		ns
SO0 output delay time (to $\overline{SCK0}$↓)	t_{DSBSK1}	CMOS push-pull output (3-wire serial I/O mode)		0	5/f_{XX} + 150	ns
	t_{DSBSK2}	Open-drain output (2-wire serial I/O mode), R_L = 1 kΩ		0	5/f_{XX} + 400	ns

Remarks 1. The values in this table are those when C_L is 100 pF.

2. T : Serial clock cycle set by software. The minimum value is 16/f_{XX}.

3. f_{XX} : Oscillator frequency

(2) I²C (μPD784038Y Subseries Only)

Parameter	Symbol	I²C Bus in Standard Mode f_{XX} = 4 to 32 MHz		I²C Bus in Standard Mode f_{XX} = 8 to 32 MHz		Unit
		MIN.	MAX.	MIN.	MAX.	
SCL clock frequency	f_{SCL}	0	100	0	400	kHz
Time to hold low SCL clock	t_{LOW}	4.7		1.3		μs
Time to hold high SCL clock	t_{HIGH}	4.0		0.6		μs
Data hold time	t_{HD}; DAT	300		300	900	ns
Data setup time	t_{SU}; DAT	250		100		ns
Rise time of SDA or SCL signal	t_R		1,000	20 + 0.1Cb	300	ns
Fall time of SDA or SCL signal	t_F		300	20 + 0.1Cb	300	ns
Load capacitance of each bus line	Cb		400		400	pF

(3) IOE1, IOE2

Parameter	Symbol	Conditions		MIN.	MAX.	Unit
Serial clock cycle time ($\overline{SCK1}$, $\overline{SCK2}$)	t_{CYSK1}	Input	V_{DD} = +5.0 V ±10%	250		ns
			V_{DD} = +2.7 to 5.5 V	500		ns
		Output	Internal, divided by 16	T		ns
Serial clock low-level width ($\overline{SCK1}$, $\overline{SCK2}$)	t_{WSKL1}	Input	V_{DD} = +5.0 V ±10%	85		ns
			V_{DD} = +2.7 to 5.5 V	210		ns
		Output	Internal, divided by 16	0.5T – 40		ns
Serial clock high-level width ($\overline{SCK1}$, $\overline{SCK2}$)	t_{WSKH1}	Input	V_{DD} = +5.0 V ±10%	85		ns
			V_{DD} = +2.7 to 5.5 V	210		ns
		Output	Internal, divided by 16	0.5T – 40		ns
Setup time for SI1 and SI2 (to $\overline{SCK1}$, $\overline{SCK2}$↑)	t_{SSSK1}			40		ns
Hold time for SI1 and SI2 (to $\overline{SCK1}$, $\overline{SCK2}$↑)	t_{HSSK1}			40		ns
Output delay time for SO1 and SO2 (to $\overline{SCK1}$, $\overline{SCK2}$↓)	t_{DSOSK}			0	50	ns
Output hold time for SO1 and SO2 (to $\overline{SCK1}$, $\overline{SCK2}$↑)	t_{HSOSK}	When data is transferred		$0.5t_{CYSK1}$ – 40		ns

Remarks 1. The values in this table are those when C_L is 100 pF.

2. T: Serial clock cycle set by software. The minimum value is 16/f_{XX}.

(4) UART, UART2

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
ASCK clock input cycle time	t_{CYASK}	V_{DD} = +5.0 V ±10%	125		ns
		V_{DD} = +2.7 to 5.5 V	250		ns
ASCK clock low-level width	t_{WASKL}	V_{DD} = +5.0 V ±10%	52.5		ns
		V_{DD} = +2.7 to 5.5 V	85		ns
ASCK clock high-level width	t_{WASKH}	V_{DD} = +5.0 V ±10%	52.5		ns
		V_{DD} = +2.7 to 5.5 V	85		ns

CLOCK OUTPUT[Note] OPERATION

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
CLKOUT cycle time	t_{CYCL}		nT		ns
CLKOUT low-level width	t_{CLL}	V_{DD} = +5.0 V ±10%	$0.5t_{CYCL}$ – 10		ns
		V_{DD} = +2.7 to 5.5 V	$0.5t_{CYCL}$ – 20		ns
CLKOUT high-level width	t_{CLH}	V_{DD} = +5.0 V ±10%	$0.5t_{CYCL}$ – 10		ns
		V_{DD} = +2.7 to 5.5 V	$0.5t_{CYCL}$ – 20		ns
CLKOUT rise time	t_{CLR}	V_{DD} = +5.0 V ±10%		10	ns
		V_{DD} = +2.7 to 5.5 V		20	ns
CLKOUT fall time	t_{CLF}	V_{DD} = +5.0 V ±10%		10	ns
		V_{DD} = +2.7 to 5.5 V		20	ns

Note Not provided in the μPD784031, 784031Y, and 784031(A).

Remark n: Divided frequency ratio set by software in the CPU (n = 1, 2, 4, 8, 16)
T: t_{CYK} (system clock cycle time)

OTHER OPERATIONS

(a) μPD784035, 784036, 784037, 784038, 78P4038, 78P4038Y, 784031(A), 784035(A), 784036(A)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
NMI low-level width	t_{WNIL}		10		μs
NMI high-level width	t_{WNIH}		10		μs
INTP0 low-level width	t_{WIT0L}		$4t_{CYSMP}$		ns
INTP0 high-level width	t_{WIT0H}		$4t_{CYSMP}$		ns
Low-level width for INTP1-INTP3 and CI	t_{WIT1L}		$4t_{CYCPU}$		ns
High-level width for INTP1-INTP3 and CI	t_{WIT1H}		$4t_{CYCPU}$		ns
Low-level width for INTP4 and INTP5	t_{WIT2L}		10		μs
High-level width for INTP4 and INTP5	t_{WIT2H}		10		μs
$\overline{\text{RESET}}$ low-level width	t_{WRSL}		10		μs
$\overline{\text{RESET}}$ high-level width	t_{WRSH}		10		μs

Remark t_{CYSMP}: Sampling clock set by software
t_{CYCPU}: CPU operation clock set by software in the CPU

(b) μPD784031, 784031Y, 784035Y, 784036Y, 784037Y, 784038Y

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
NMI low-level width	t_{WNIL}		10		μs
NMI high-level width	t_{WNIH}		10		μs
INTP0 low-level width	t_{WIT0L}		$3t_{CYSMP}$		ns
INTP0 high-level width	t_{WIT0H}		$3t_{CYSMP}$		ns
Low-level width for INTP1-INTP3 and CI	t_{WIT1L}		$3t_{CYCPU}$		ns
High-level width for INTP1-INTP3 and CI	t_{WIT1H}		$3t_{CYCPU}$		ns
Low-level width for INTP4 and INTP5	t_{WIT2L}		10		μs
High-level width for INTP4 and INTP5	t_{WIT2H}		10		μs
$\overline{\text{RESET}}$ low-level width	t_{WRSL}		10		μs
$\overline{\text{RESET}}$ high-level width	t_{WRSH}		10		μs

Remark t_{CYSMP}: Sampling clock set by software
t_{CYCPU}: CPU operation clock set by software in the CPU

A/D CONVERTER CHARACTERISTICS

(T_A = –40 to +85°C, V_{DD} = AV_{DD} = AV_{REF1} = +2.7 to 5.5 V, V_{SS} = AV_{SS} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution			8	8	8	bit
Total error**Note** (μPD784031, 784031Y, 784031(A), mask ROM version)					1.0	%FSR
Total error**Note** (μPD78P4038, 78P4038Y)		V_{DD} = AV_{DD} = +5.0 V ±10%			1.0	%FSR
		V_{DD} = AV_{DD} = +2.7 to 4.5 V T_A = -10 to +85°C			1.0	%FSR
Linearity calibration**Note**					0.8	%FSR
Quantization error					±1/2	LSB
Conversion time	t_{CONV}	FR = 1	120			t_{CYK}
		FR = 0	180			t_{CYK}
Sampling time	t_{SAMP}	FR = 1	24			t_{CYK}
		FR = 0	36			t_{CYK}
Analog input voltage	V_{IAN}		–0.3		AV_{REF1} + 0.3	V
Analog input impedance	R_{AN}			1,000		MΩ
AV_{REF1} current	AI_{REF1}			0.5	1.5	mA
AV_{DD} supply current	AI_{DD1}	f_{XX} = 32 MHz, CS = 1		2.0	5.0	mA
	AI_{DD2}	STOP mode, CS = 0		1.0	20	μA

Note Quantization error (±1/2 LSB) is not included. This parameter is indicated as the ratio (%FSR) to the full-scale value.

Remark f_{XX}: Oscillation frequency

D/A CONVERTER CHARACTERISTICS (T_A = –40 to +85°C, V_{DD} = AV_{DD} = +2.7 to 5.5 V, V_{SS} = AV_{SS} = 0 V)

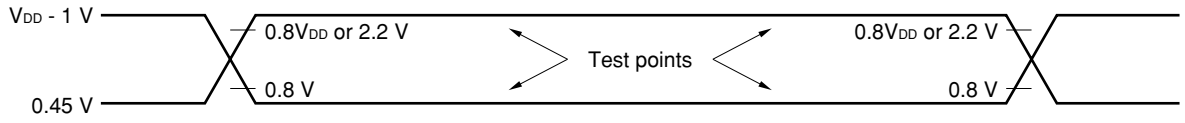
Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Resolution				8			bit
Total error		Load conditions: 4 MΩ, 30 pF	V_{DD} = AV_{DD} = AV_{REF2} = +2.7 to 5.5 V AV_{REF3} = 0 V			0.6	%
			V_{DD} = AV_{DD} = +2.7 to 5.5 V AV_{REF2} = 0.75V_{DD} AV_{REF3} = 0.25V_{DD}			0.8	%
		Load conditions: 2 MΩ, 30 pF	V_{DD} = AV_{DD} = AV_{REF2} = +2.7 to 5.5 V AV_{REF3} = 0 V			0.8	%
			V_{DD} = AV_{DD} = +2.7 to 5.5 V AV_{REF2} = 0.75V_{DD} AV_{REF3} = 0.25V_{DD}			1.0	%
Settling time		Load conditions: 2 MΩ, 30 pF				10	μs
Output resistance	R_O	DACS0, 1 = 55 H			10		kΩ
Analog reference voltage	AV_{REF2}			0.75V_{DD}		V_{DD}	V
	AV_{REF3}			0		0.25V_{DD}	V
Resistance of AV_{REF2} and AV_{REF3}	R_{AIREF}	DACS0, 1 = 55 H		4	8		kΩ
Reference power supply input current	AI_{REF2}			0		5	mA
	AI_{REF3}			–5		0	mA

DATA RETENTION CHARACTERISTICS (T_A = –40 to +85°C)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data retention voltage	V_{DDDR}	STOP mode	2.5		5.5	V
Data retention current[Note] (μPD784031, 784031Y, 784031(A), mask ROM version)	I_{DDDR}	V_{DDDR} = +2.7 to 5.5 V		10	50	μA
		V_{DDDR} = +2.5 V		2	10	μA
Data retention current (μPD78P4038, 78P4038Y)	I_{DDDR}	V_{DDDR} = +2.7 to 5.5 V		30	50	μA
		V_{DDDR} = +2.5 V		10	40	μA
V_{DD} rise time	t_{RVD}		200			μs
V_{DD} fall time	t_{FVD}		200			μs
V_{DD} hold time (to STOP mode setting)	t_{HVD}		0			ms
STOP clear signal input time	t_{DREL}		0			ms
Oscillation settling time	t_{WAIT}	Crystal	30			ms
		Ceramic resonator	5			ms
Input low voltage	V_{IL}	Specific pins[Note]	0		0.1V_{DDDR}	V
Input high voltage	V_{IH}		0.9V_{DDDR}		V_{DDDR}	V

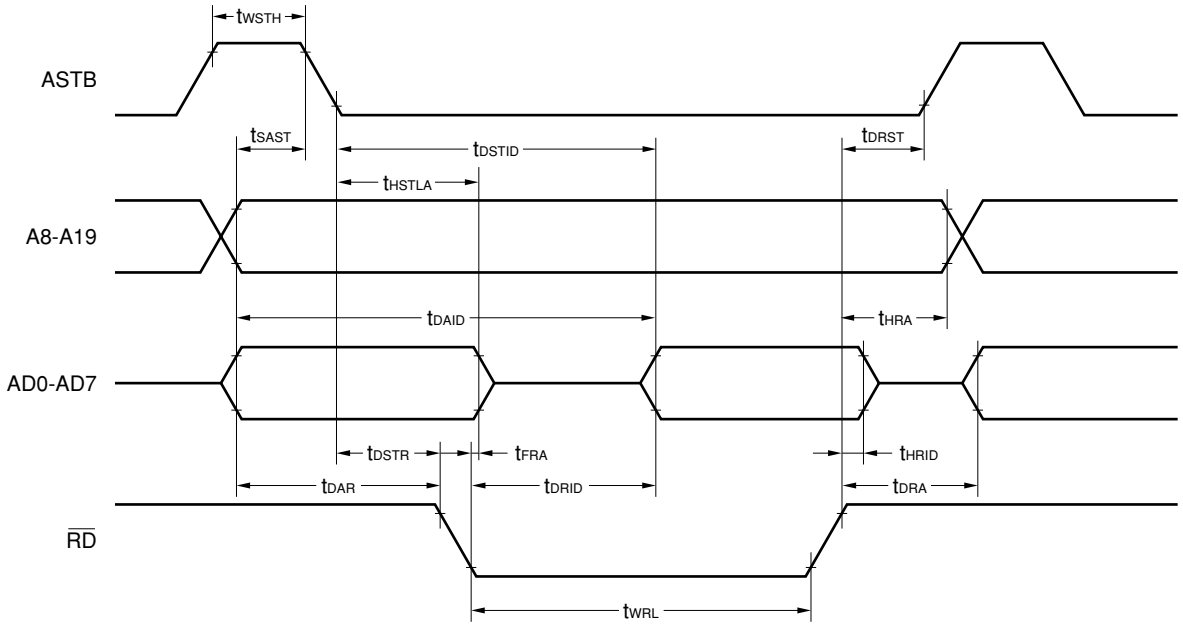
Note $\overline{\text{RESET}}$, P20/NMI, P21/INTP0, P22/INTP1, P23/INTP2/CI, P24/INTP3, P25/INTP4/ASCK/$\overline{\text{SCK1}}$, P26/INTP5, P27/SI0, P32/$\overline{\text{SCK0}}$/SCL, and P33/SO0/SDA pins

AC TIMING TEST POINTS

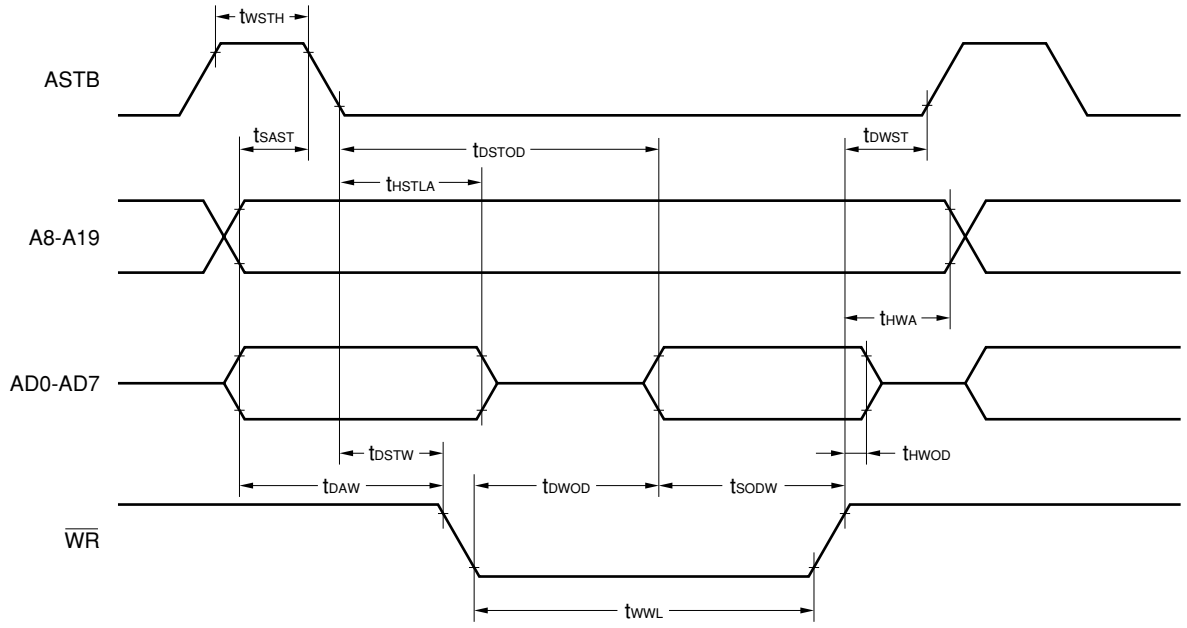


TIMING WAVEFORM

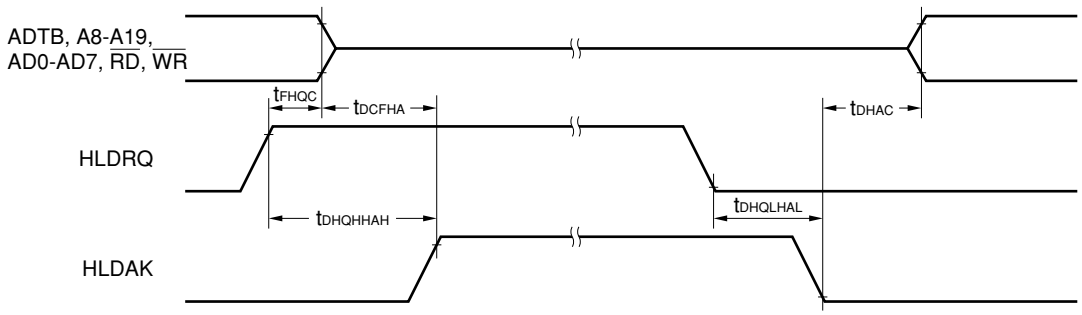
(1) Read operation



(2) Write operation

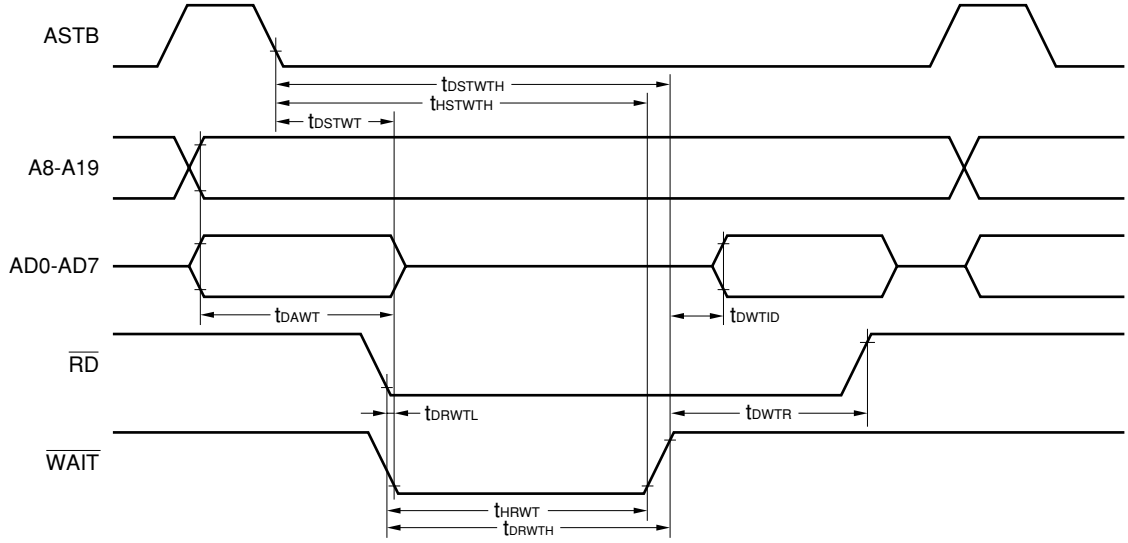


HOLD TIMING

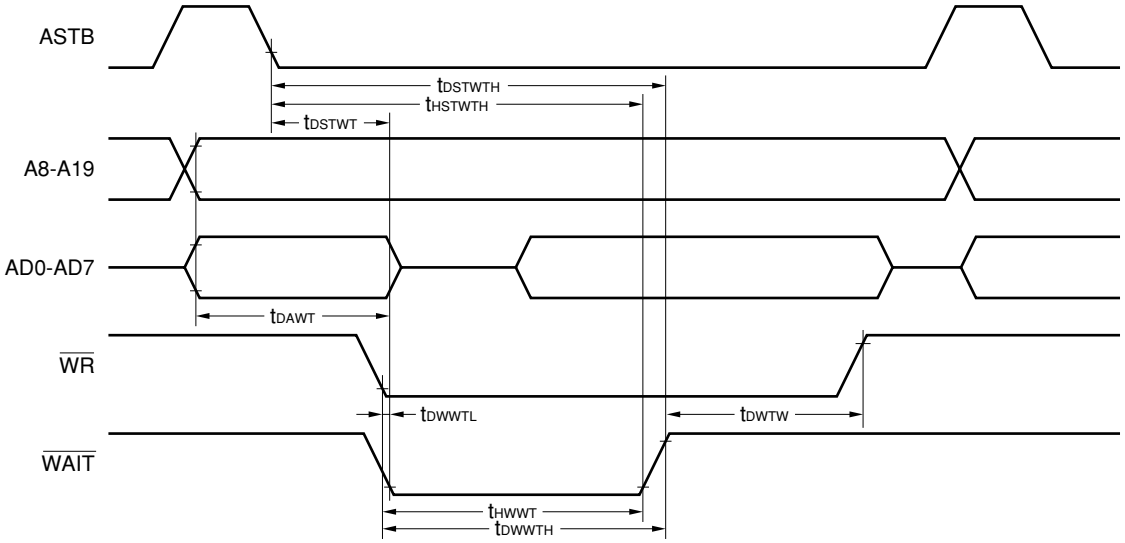


EXTERNAL WAIT SIGNAL INPUT TIMING

(1) Read operation

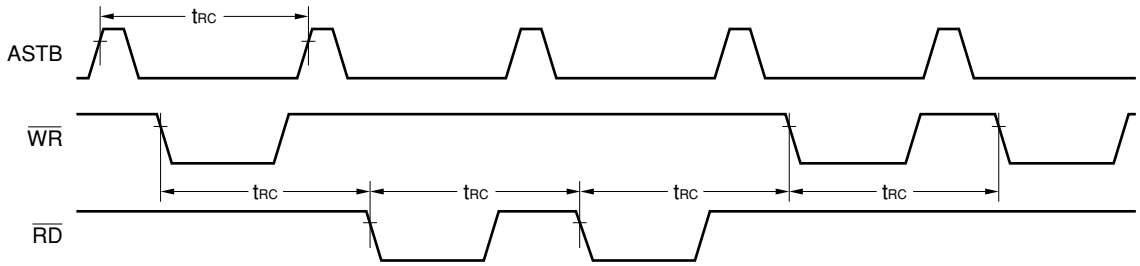


(2) Write operation

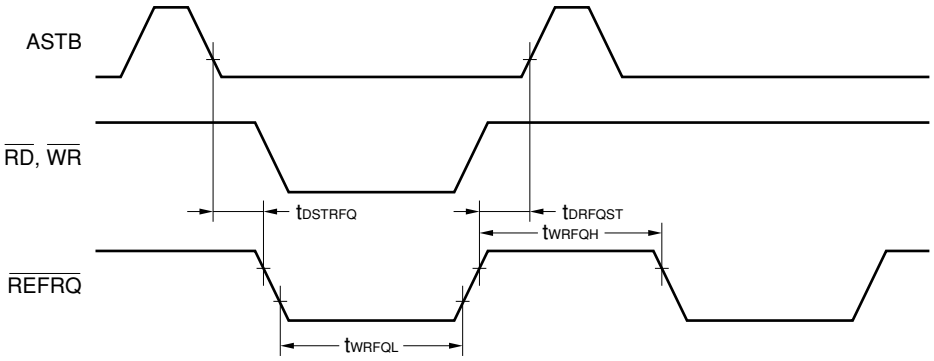


REFRESH TIMING WAVEFORM

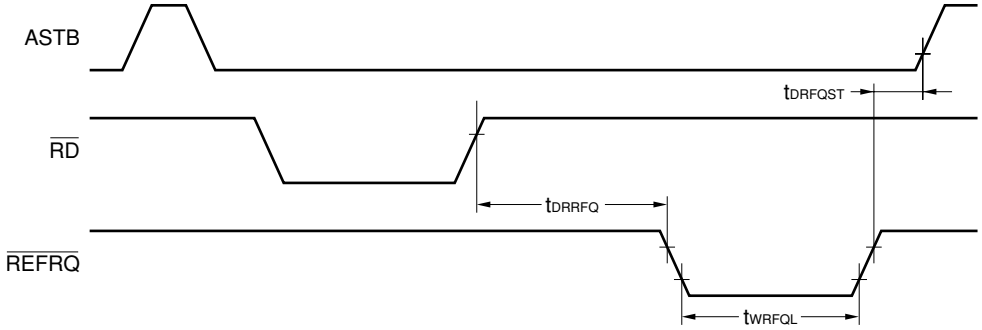
(1) Random read/write cycle



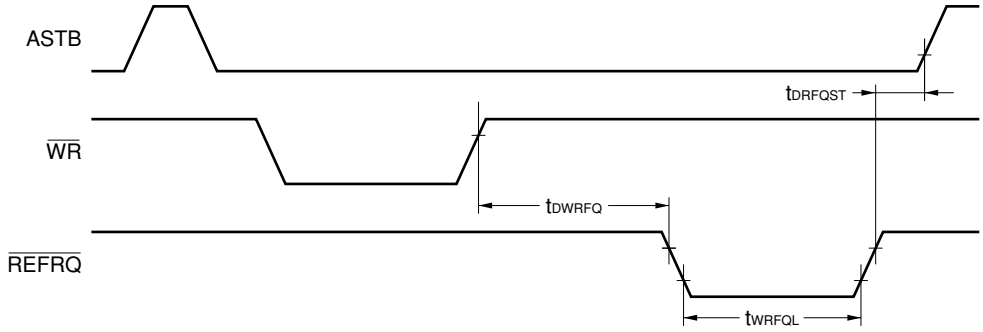
(2) When refresh memory is accessed for a read and write at the same time



(3) Refresh after a read

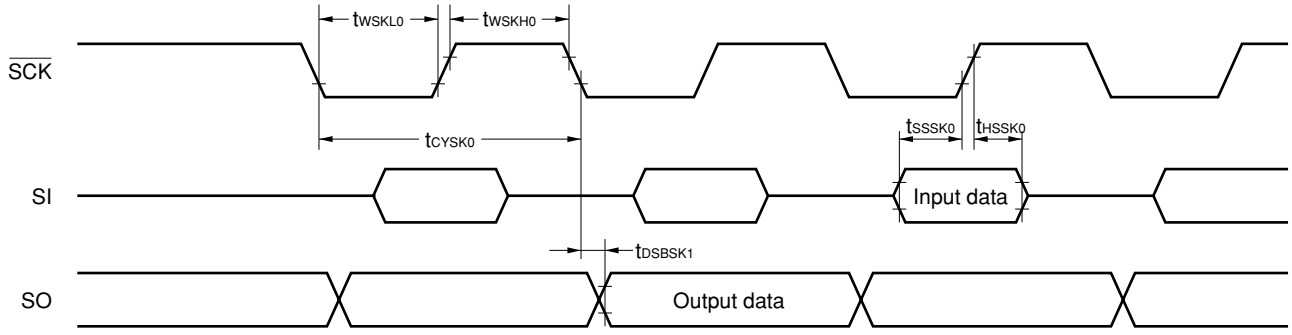


(4) Refresh after a write

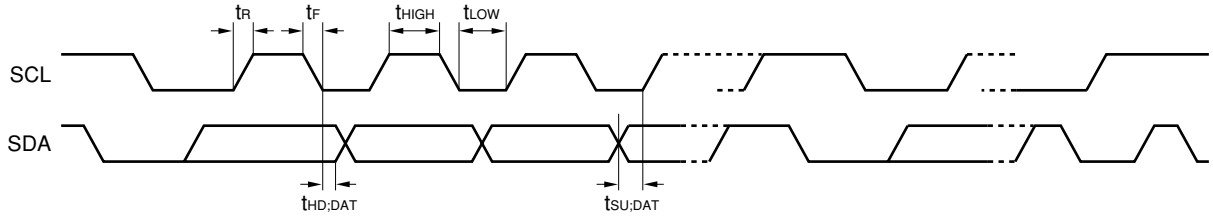


SERIAL OPERATION

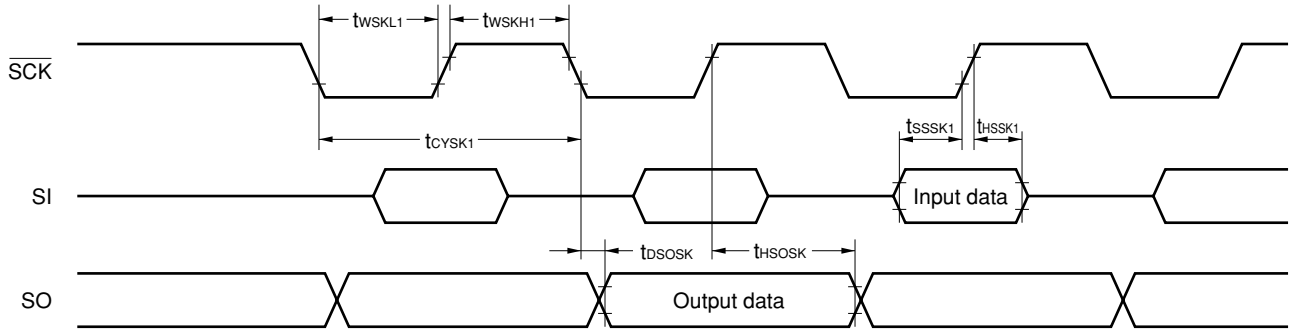
(1) CSI



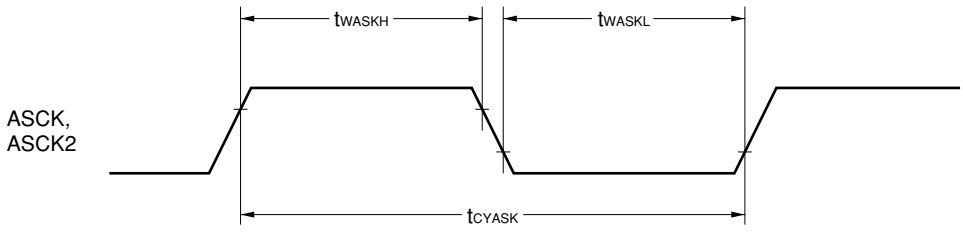
(2) I²C (μPD784038Y Subseries Only)



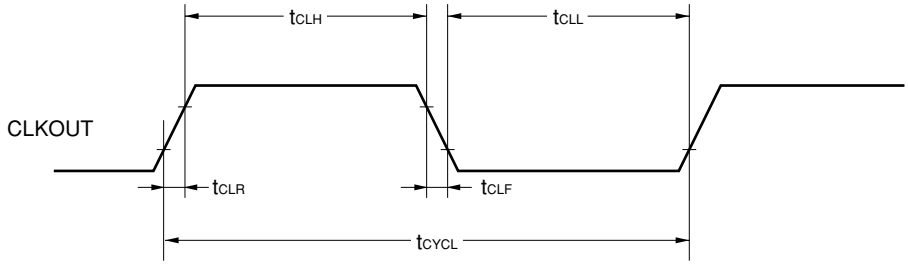
(3) IOE1, IOE2



(4) UART, UART2

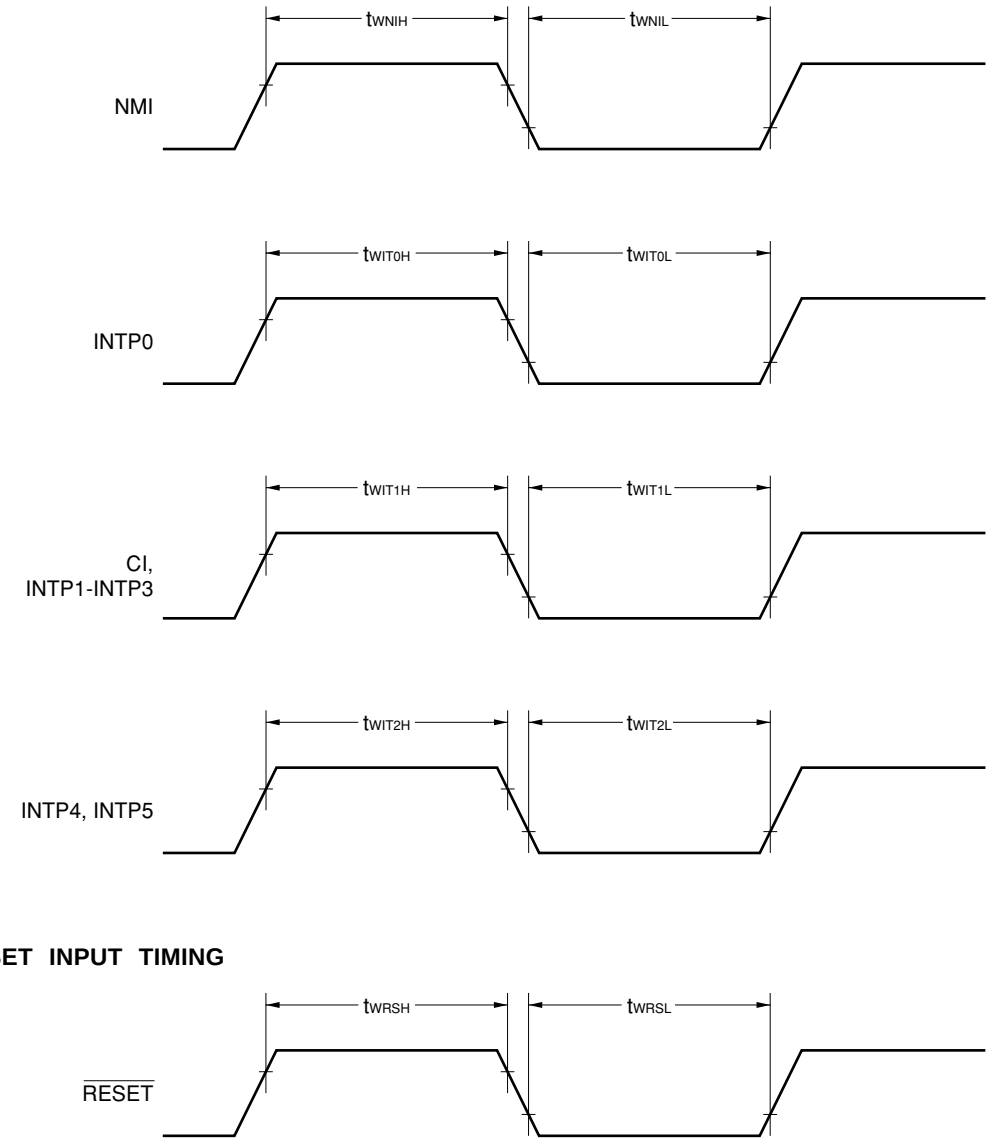


CLOCK OUTPUT[Note] TIMING



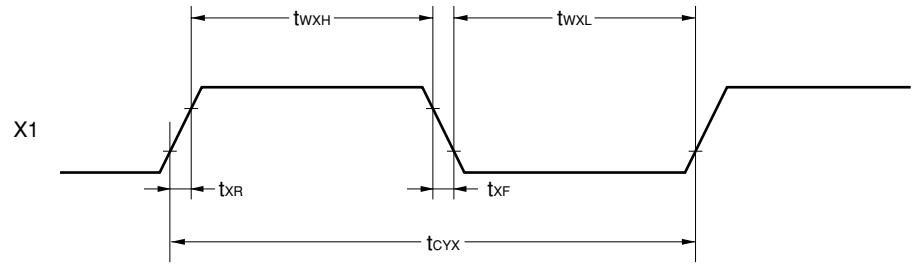
Note Not provided in the μPD784031, 784031Y, and 784031(A).

INTERRUPT INPUT TIMING

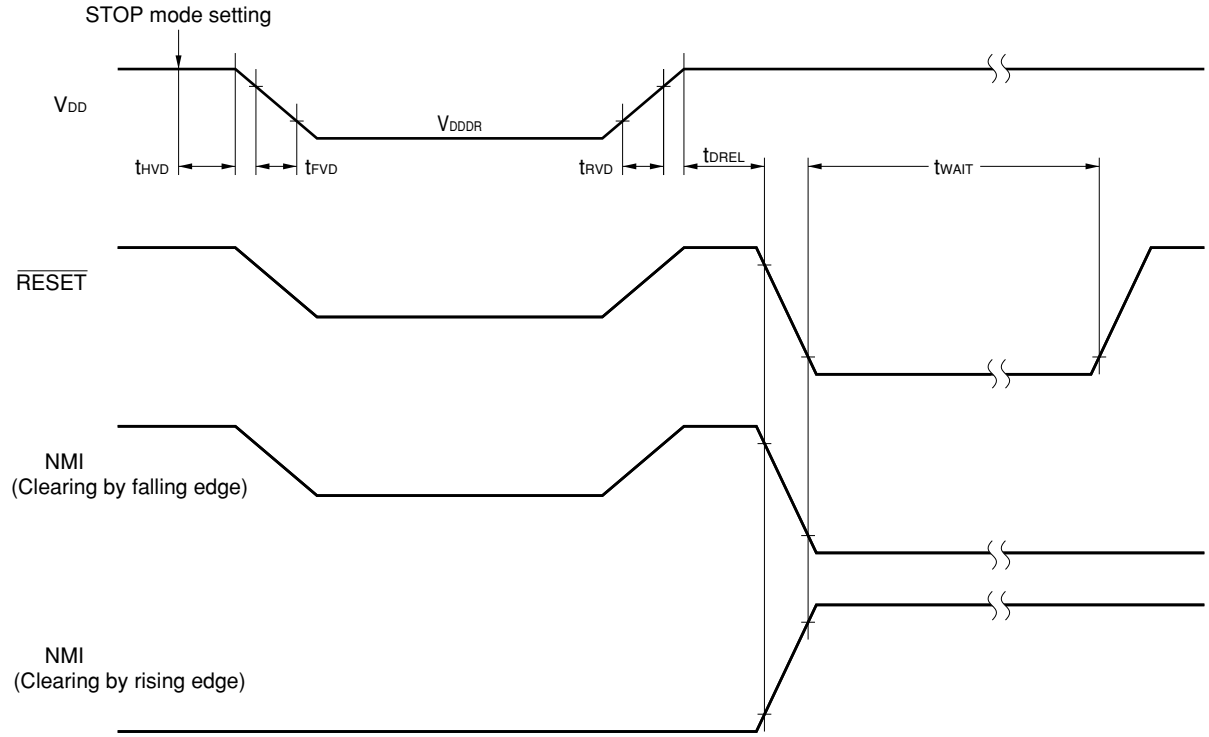


RESET INPUT TIMING

EXTERNAL CLOCK TIMING



DATA RETENTION CHARACTERISTICS



DC PROGRAMMING CHARACTERISTICS (T_A = 25 ± 5°C, V_{SS} = 0 V) : μPD78P4038 and 78P4038Y only

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
High-level input voltage	V_{IH}		2.2		V_{DDP} + 0.3	V
Low-level input voltage	V_{IL}		–0.3		0.8	V
Input leakage current	I_{LIP}	$0 \le V_I \le V_{DDP}$ **Note**	2.4		±10	μA
High-level output voltage	V_{OH}	I_{OH} = –400 μA				V
Low-level output voltage	V_{OL}	I_{OL} = 2.1 mA			0.45	V
Output leakage current	I_{LO}	$0 \le V_O \le V_{DDP}$, $\overline{OE} = V_{IH}$			±10	μA
V_{DDP} supply voltage	V_{DD}	Program memory write mode	6.25	6.5	6.75	V
		Program memory read mode	4.5	5.0	5.5	V
V_{PP} supply voltage	V_{PP}	Program memory write mode	12.2	12.5	12.8	V
		Program memory read mode	$V_{PP} = V_{DDP}$			V
V_{DDP} supply current	I_{DD}	Program memory write mode		10	40	mA
		Program memory read mode		10	40	mA
V_{PP} supply current	I_{PP}	Program memory write mode		5	50	mA
		Program memory read mode		1.0	100	μA

Note The V_{DDP} represents the V_{DD} pin as viewed in the programming mode.

AC PROGRAMMING CHARACTERISTICS (T_A = 25 ± 5°C, V_{SS} = 0 V) : μPD78P4038 and 78P4038Y only

PROM Write Mode (Page Program Mode)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Address setup time	t_{AS}		2			μs
$\overline{CE}$ set time	t_{CES}		2			μs
Input data setup time	t_{DS}		2			μs
Address hold time	t_{AH}		2			μs
	t_{AHL}		2			μs
	t_{AHV}		0			μs
Input data hold time	t_{DH}		2			μs
Output data hold time	t_{DF}		0		130	ns
V_{PP} setup time	t_{VPS}		2			μs
V_{DDP} setup time	t_{VDS}		2			μs
Initial program pulse width	t_{PW}		0.095	0.1	0.105	ms
$\overline{OE}$ set time	t_{OES}		2			μs
Valid data delay time from $\overline{OE}$	t_{OE}			1	2	ns
$\overline{OE}$ pulse width in the data latch	t_{LW}		1			μs
$\overline{PGM}$ setup time	t_{PGMS}		2			μs
$\overline{CE}$ hold time	t_{CEH}		2			μs
$\overline{OE}$ hold time	t_{OEH}		2			μs

PROM Write Mode (Byte Program Mode)

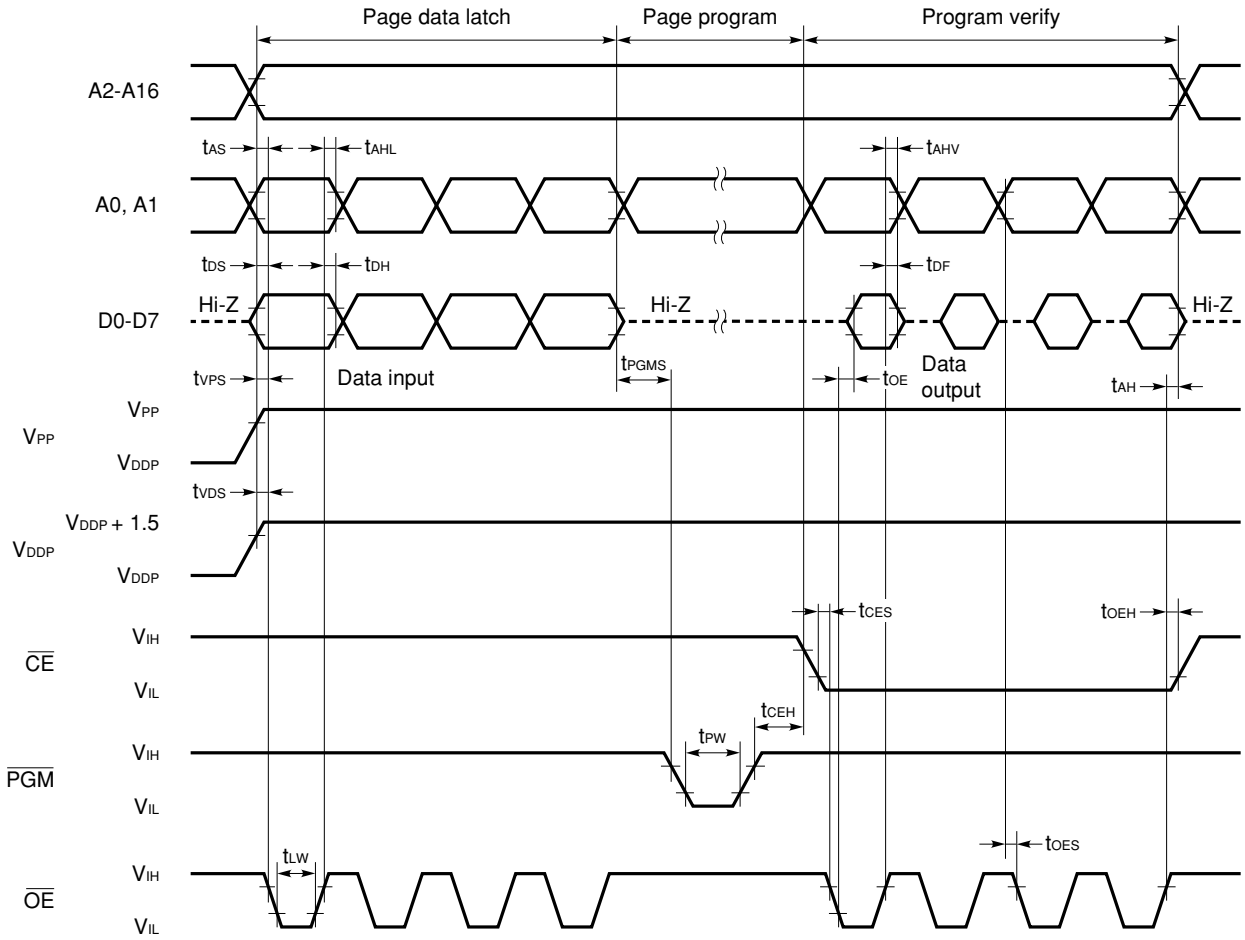
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Address setup time	t_{AS}		2			μs
$\overline{CE}$ set time	t_{CES}		2			μs
Input data setup time	t_{DS}		2			μs
Address hold time	t_{AH}		2			μs
Input data hold time	t_{DH}		2			μs
Output data hold time	t_{DF}		0		130	ns
V_{PP} setup time	t_{VPS}		2			μs
V_{DDP} setup time	t_{VDS}		2			μs
Initial program pulse width	t_{PW}		0.095	0.1	0.105	ms
$\overline{OE}$ set time	t_{OES}		2			μs
Valid data delay time from $\overline{OE}$	t_{OE}			1	2	ns

PROM Read Mode

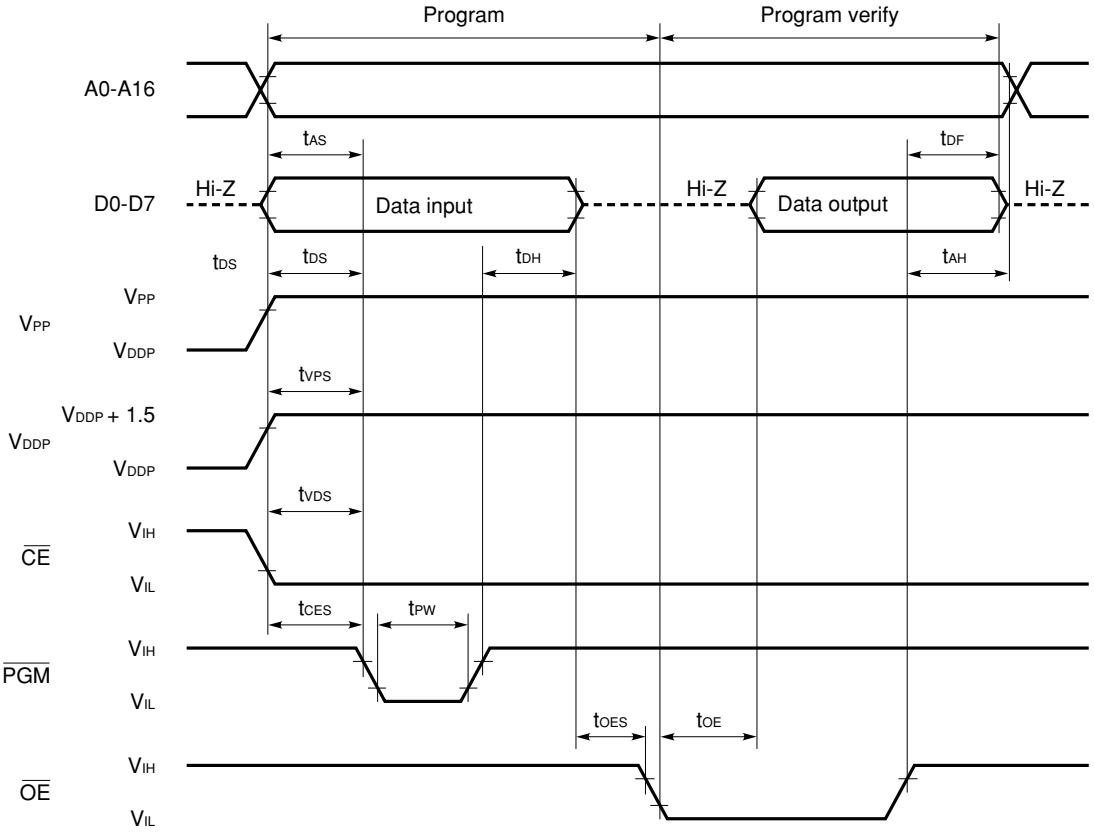
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data output time from address	t_{ACC}	$\overline{CE} = \overline{OE} = V_{IL}$			200	ns
Delay from $\overline{CE} \downarrow$ to data output	t_{CE}	$\overline{OE} = V_{IL}$		1	2	μs
Delay from $\overline{OE} \downarrow$ to data output	t_{OE}	$\overline{CE} = V_{IL}$		1	2	μs
Data hold time to $\overline{OE}\uparrow$ or $\overline{CE}\uparrow$ **Note**	t_{DF}	$\overline{CE} = V_{IL}$ or $\overline{OE} = V_{IL}$	0		60	ns
Data hold time to address	t_{OH}	$\overline{CE} = \overline{OE} = V_{IL}$	0			ns

Note t_{DF} is the time measured from when either $\overline{OE}$ or $\overline{CE}$ reaches V_{IH}, whichever is faster.

PROM Write Mode Timing (Page Program Mode) : μPD78P4038 and 78P4038Y only



PROM Write Mode Timing (Byte Program Mode) : μPD78P4038 and 78P4038Y only

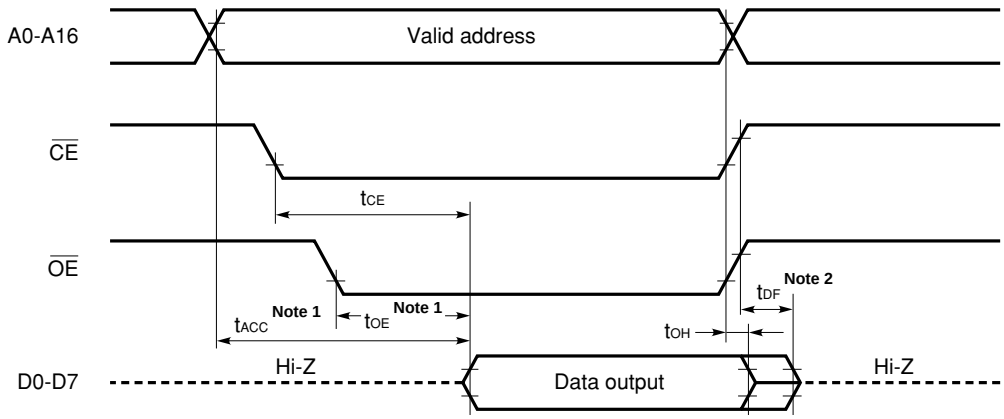


Cautions 1. V_{DDP} must be applied before V_{PP}, and must be cut after V_{PP}.

2. V_{PP} including overshoot must not exceed +13.5 V.

3. Plugging in or out the board with the V_{PP} pin supplied with 12.5 V may adversely affect its reliability.

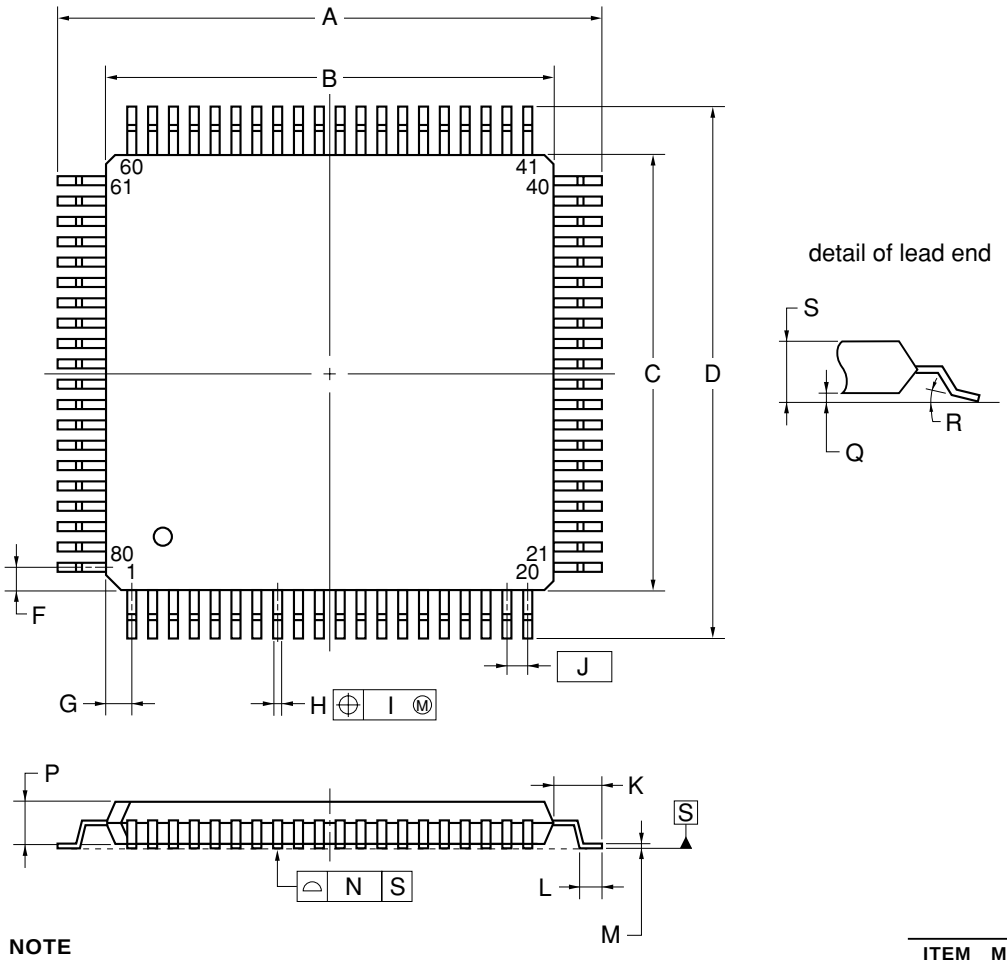
PROM Read Mode Timing : μPD78P4038 and 78P4038Y only



Notes 1. For reading within t_{ACC}, the delay of the $\overline{OE}$ input from falling edge of $\overline{CE}$ must be within t_{ACC}-t_{OE}.

2. t_{DF} is the time measured from when either $\overline{OE}$ or $\overline{CE}$ reaches V_{IH}, whichever is faster.

80-PIN PLASTIC QFP (14x14)



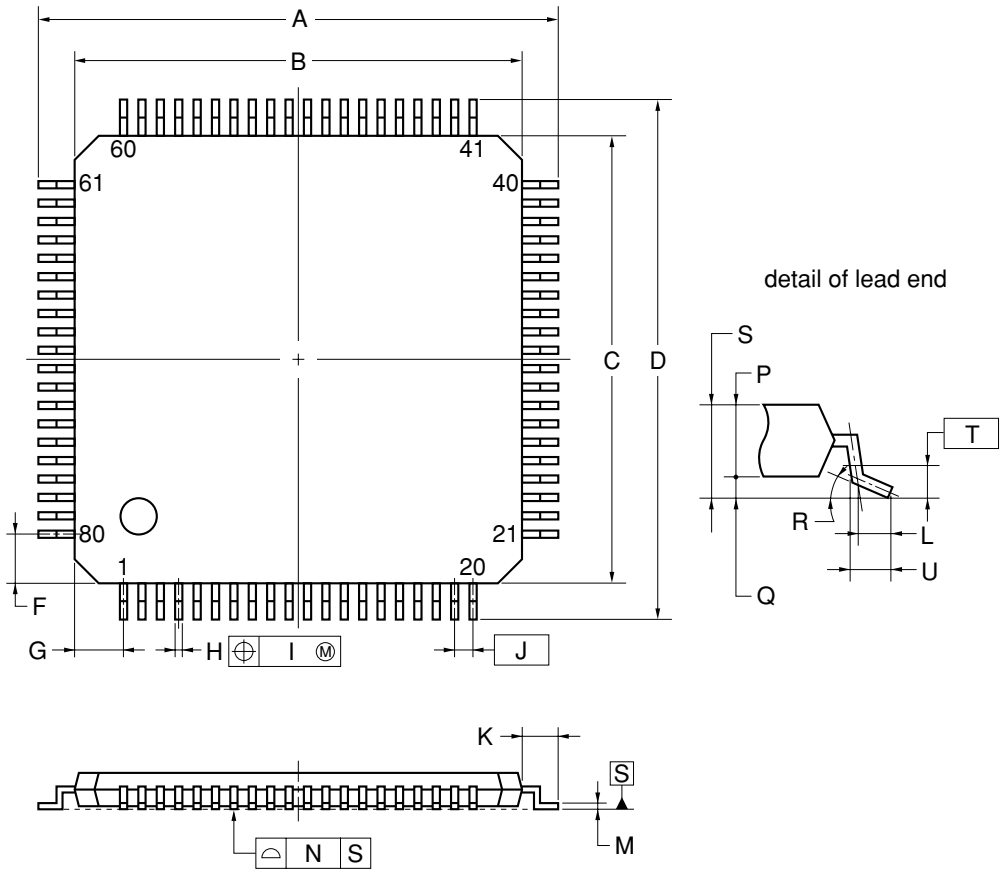
NOTE

Each lead centerline is located within 0.13 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	17.20±0.20
B	14.00±0.20
C	14.00±0.20
D	17.20±0.20
F	0.825
G	0.825
H	0.32±0.06
I	0.13
J	0.65 (T.P.)
K	1.60±0.20
L	0.80±0.20
M	$0.17^{+0.03}_{-0.07}$
N	0.10
P	1.40±0.10
Q	0.125±0.075
R	$3^{\circ}{}^{+7^{\circ}}_{-3^{\circ}}$
S	1.70 MAX.

P80GC-65-8BT-1

80-PIN PLASTIC TQFP (FINE PITCH) (12x12)



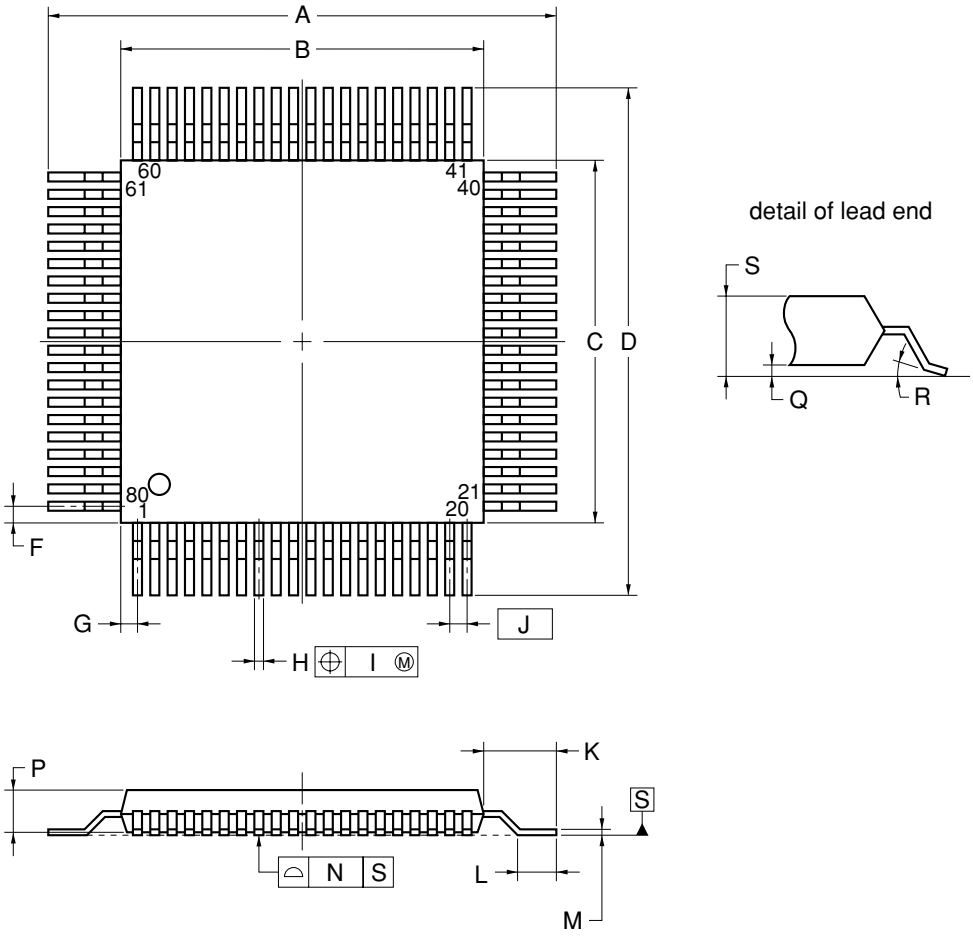
NOTE

Each lead centerline is located within 0.08 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	14.0±0.2
B	12.0±0.2
C	12.0±0.2
D	14.0±0.2
F	1.25
G	1.25
H	0.22±0.05
I	0.08
J	0.5 (T.P.)
K	1.0±0.2
L	0.5
M	0.145±0.05
N	0.08
P	1.0
Q	0.1±0.05
R	$3^{\circ}{}^{+4^{\circ}}_{-3^{\circ}}$
S	1.1±0.1
T	0.25
U	0.6±0.15

P80GK-50-9EU-1

80-PIN PLASTIC QFP (14x14)



NOTE

Each lead centerline is located within 0.13 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	17.2±0.4
B	14.0±0.2
C	14.0±0.2
D	17.2±0.4
F	0.825
G	0.825
H	0.30±0.10
I	0.13
J	0.65 (T.P.)
K	1.6±0.2
L	0.8±0.2
M	$0.15^{+0.10}_{-0.05}$
N	0.10
P	2.7±0.1
Q	0.1±0.1
R	5°±5°
S	3.0 MAX.

S80GC-65-3B9-6

★ CHAPTER 30 RECOMMENDED SOLDERING CONDITIONS

This product should be soldered and mounted under the following recommended conditions.

For details of the recommended soldering conditions, refer to the document **Semiconductor Device Mounting Technology Manual (C10535E)**.

For soldering methods and conditions other than those recommended below, contact an NEC Electronics sales representative.

Remark The soldering conditions for the μPD784031YGC-8BT, 784031GC(A)-3B9, 784035YGK-×××-9EU, 784035GC(A)-×××-3B9, 784036YGK-×××-9EU, 784036GC(A)-×××-3B9, and 78P4038YGK-9EU are undetermined. Contact an NEC Electronics sales representative for details.

Table 30-1 Surface Mounting Type Soldering Conditions (1/2)

(1) μPD784031GC-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD784035GC-×××-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD784035YGC-×××-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD784036GC-×××-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD784036YGC-×××-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD784037GC-×××-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD784037YGC-×××-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD784038GC-×××-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD784038YGC-×××-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD78P4038GC-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)
μPD78P4038YGC-8BT: 80-pin plastic QFP (14 x 14, 1.4 mm thickness)

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C, Time: 30 seconds max. (at 210°C or higher), Count: Two times or less	IR35-00-2
VPS	Package peak temperature: 215°C, Time: 40 seconds max. (at 200°C or higher), Count: Two times or less	VP15-00-2
Wave soldering	Solder bath temperature: 260°C max., Time: 10 seconds max., Count: Once, Preheating temperature: 120°C Max. (package surface temperature)	WS60-00-1
Partial heating	Pin temperature: 300°C max., Time: 3 seconds max. (per pin row)	–

Caution Do not use different soldering methods together (except for partial heating).

Table 30-1 Surface Mounting Type Soldering Conditions (2/2)

(2) μPD784031GK-9EU: 80-pin plastic TQFP (fine pitch) (12 x 12)
μPD784031YGK-9EU: 80-pin plastic TQFP (fine pitch) (12 x 12)
μPD784035GK-xxx-9EU: 80-pin plastic TQFP (fine pitch) (12 x 12)
μPD784036GK-xxx-9EU: 80-pin plastic TQFP (fine pitch) (12 x 12)
μPD784037GK-xxx-9EU: 80-pin plastic TQFP (fine pitch) (12 x 12)
μPD784037YGK-xxx-9EU: 80-pin plastic TQFP (fine pitch) (12 x 12)
μPD784038GK-xxx-9EU: 80-pin plastic TQFP (fine pitch) (12 x 12)
μPD784038YGK-xxx-9EU: 80-pin plastic TQFP (fine pitch) (12 x 12)
μPD78P4038GK-9EU: 80-pin plastic TQFP (fine pitch) (12 x 12)

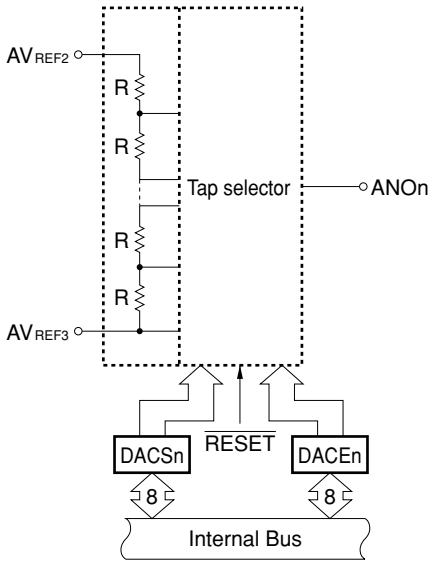
Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C, Time: 30 seconds max. (at 210°C or higher), Count: Two times or less, Exposure limit: 7 days[Note] (after that, prebake at 125°C for 10 hours)	IR35-107-2
VPS	Package peak temperature: 215°C, Time: 40 seconds max. (at 200°C or higher), Count: Two times or less, Exposure limit: 7 days[Note] (after that, prebake at 125°C for 10 hours)	VP15-107-2
Partial heating	Pin temperature: 300°C max., Time: 3 seconds max. (per pin row)	—

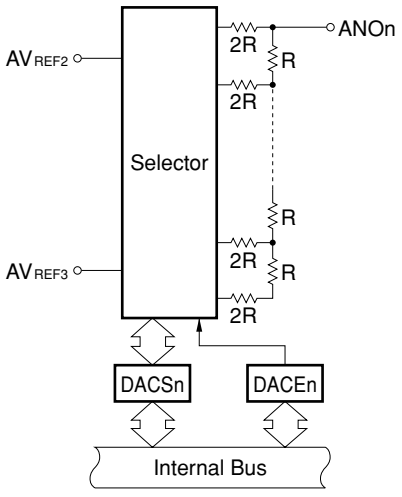
Note After opening the dry pack, store it at 25°C or less and 65%RH or less for the allowable storage period.

Caution Do not use different soldering methods together (except for partial heating).

APPENDIX A DIFFERENCES WITH μPD784026 SUBSERIES

Table A-1 Differences with μPD784026 Subseries (1/3)

<table>
<tr><th>Item</th><th>μPD784026 Subseries</th><th>μPD784038 Subseries</th></tr>
<tr><td>Operating frequency</td><td>4 MHz ≤ fXX ≤ 25 MHz</td><td>4 MHz ≤ fXX ≤ 32 MHz (target value)</td></tr>
<tr><td>Minimum instruction execution time</td><td>160 ns (at 25 MHz)</td><td>125 ns (at 32 MHz)</td></tr>
<tr><td>Internal ROM/RAM capacity</td><td>μPD784020 : None/512 bytes
μPD784021 : None/2,048 bytes
μPD784025 : 48 Kbytes/2,048 bytes
μPD784026 : 64 Kbytes/2,048 bytes
μPD78P4026 : 64 Kbytes/2,048 bytes (PROM)</td><td>μPD784031 : None/2,048 bytes
μPD784035 : 48 Kbytes/2,048 bytes
μPD784036 : 64 Kbytes/2,048 bytes
μPD784037 : 96 Kbytes/3,584 bytes
μPD784038 : 128 Kbytes/4,352 bytes
μPD78P4038 : 128 Kbytes/4,352 bytes (PROM)</td></tr>
<tr><td>PROM size selection</td><td>Two types of memory size can be selected according to mask ROM model.
IMS: 7 6 5 4 3 2 1 0 — IMS7 IMS6 IMS5 IMS4 IMS3 IMS2 IMS1 IMS0
<table><tr><th>IMS7 to 0</th><th>Memory Size</th></tr><tr><td>FFH</td><td>Same as μPD784026</td></tr><tr><td>EFH</td><td>Same as μPD784025</td></tr></table></td><td>Four types of memory size can be selected according to mask ROM model.
IMS: 7 6 5 4 3 2 1 0 — IMS7 IMS6 IMS5 IMS4 IMS3 IMS2 IMS1 IMS0
<table><tr><th>IMS7 to 0</th><th>Memory Size</th></tr><tr><td>FFH</td><td>Same as μPD784038</td></tr><tr><td>EEH</td><td>Same as μPD784037</td></tr><tr><td>DCH</td><td>Same as μPD784036</td></tr><tr><td>CCH</td><td>Same as μPD784035</td></tr></table></td></tr>
<tr><td>D/A conversion mode</td><td>Resistor string mode
 

Remark n = 0, 1</td><td>R-2R resistor ladder mode
 

Remark n = 0, 1</td></tr>
</table>

Table A-1 Differences with μPD784026 Subseries (2/3)

<table>
<tr><th>Item</th><th>μPD784026 Subseries</th><th>μPD784038 Subseries</th></tr>
<tr><td>Serial interface</td><td>• UART/IOE (3-wire serial I/O) × 2 channels
• CSI (3-wire serial I/O, SBI) × 1 channel</td><td>• UART/IOE (3-wire serial I/O) × 2 channels
• CSI (3-wire serial I/O, 2-wire serial I/O) × 1 channel</td></tr>
<tr><td></td><td><Clocked serial interface mode register>

CSM: ⑦ CTXE | ⑥ CRXE | ⑤ WUP | 4 0 | 3 MOD1 | 2 MOD0 | 1 CLS1 | 0 CLS0

CTXE: 0 Disabled; 1 Enabled

CRXE: 0 Disabled; 1 Enabled

WUP: 0 Generates interrupt request signal each time serial transfer is executed in each mode.; 1 Generates interrupt request signal only when address is received in SBI mode.

MOD1, MOD0 / Operation Mode Select bit (Operation mode, Transfer direction, Pins used):
0, 0: 3-wire, MSB first, SO0, SI0, SCK0
0, 1: 3-wire, LSB first, SCK0
1, 0: SBI, MSB first, SB0, SCK
1, 1: Setting prohibited

CLS1, CLS0 / Specifies Serial Clock / SCK pin:
0, 0: External, Slave, Input
0, 1: Internal, Master, TM3/2, Output
1, 0: Internal, Master, fCLK/32
1, 1: Internal, Master, fCLK/8

fCLK: Internal system clock frequency</td><td><Clocked serial interface mode register>

CSM: ⑦ CTXE | ⑥ CRXE | 5 WUP | 4 0 | 3 MOD1 | 2 MOD0 | 1 CLS1 | 0 CLS0

CTXE: 0 Disabled; 1 Enabled

CRXE: 0 Disabled; 1 Enabled

MOD1, MOD0 / Operation Mode Select Bit (Operation mode, Transfer direction, Pins used):
0, 0: 3-wire, MSB first, SO0, SI0, SCK0
0, 1: 3-wire, LSB first, SCK0
1, 0: 2-wire, MSB first, SDA, SCL
1, 1: Setting prohibited

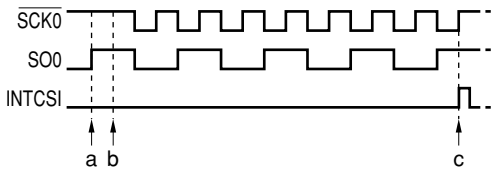
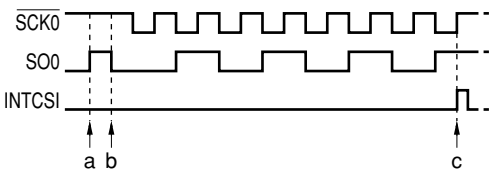
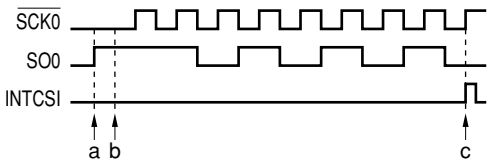
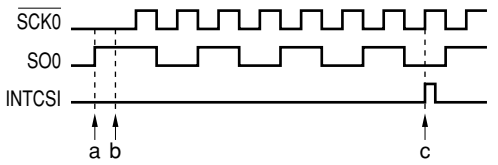
CLS1, CLS0 / Specifies Serial Clock / SCK0, SCL pin:
0, 0: External, Slave, Input
0, 1: Internal, Master, TM3/2, Output
1, 0: Internal, Master, SPRS
1, 1: Internal, Master, fXX/16

fXX: Oscillation frequency</td></tr>
</table>

Remark If the fastest internal system clock is used ($f_{CLK} = f_{XX}/2$) **Note** when the μPD784026 Subseries is replaced with the μPD784038 Subseries, the same serial clock is selected without changing the CLS1 and CLS0 bits (the same clock is selected when CLS1, CLS0 = 1, 0 because SPRS selects $f_{XX}/64$ after reset).
With the μPD784038 Subseries, the serial clock is not changed even when the system clock has been changed because the serial clock is generated by dividing f_{XX} when CLS1, CLS0 = 1, 0 or 1, 1.

Note When CK1, CK0 of STBC = 0, 0
CK1, CK0 = 1, 1 ($f_{CLK} = f_{XX}/16$) after reset.

Table A-1 Differences from μPD784026 Subseries (3/3)

Item	μPD784026 Subseries	μPD784038 Subseries
Serial interface	**<Register and bit name change>** SBIC mode register (SBIC) SBIC: 7 6 5 4 3 2 1 0 / BSYE ACKD ACKE ACKT CMDD RELD CMDT RELT	I²C bus control register (IICC) IICC: 7 6 5 4 3 2 ① ⓪ / 0 0 0 0 0 0 STT SPT **Remark** The STT and SPT bit differ from the CMDT and RELT bit only in name and the same in terms of operation that is performed through bit manipulation.
	<Additional register> —	Prescaler mode register for serial clock (SPRM)
★	**<Transmit data (SO0) write>**  a: Setting the transmission enable bit (CTXE ← 1) b: Data (55H) written to shift register (SIO ← 55H) c: Generation of transfer completion interrupt request	 a: Setting the transmission enable bit (CTXE ← 1) b: Data (55H) written to shift register (SIO ← 55H) c: Generation of transfer completion interrupt request
★	**<Serial clock ($\overline{SCK0}$) count operation>**  a: Setting the transmission enable bit (CTXE ← 1) b: Data (AAH) written to shift register (SIO ← AAH) c: Generation of transfer completion interrupt request	 a: Setting the transmission enable bit (CTXE ← 1) b: Data (AAH) written to shift register (SIO ← AAH) c: Generation of transfer completion interrupt request
★	Package	• 80-pin plastic QFP (14 × 14, 2.7 mm thickness) • 80-pin plastic TQFP (fine pitch, 12 × 12) : μPD784021 only	• 80-pin plastic QFP (14 × 14, 2.7 mm thickness)[Note] • 80-pin plastic QFP (14 × 14, 1.4 mm thickness) • 80-pin plastic TQFP (fine pitch, 12 × 12)

Note μPD784031(A), 784035(A), 784036(A) only

★

APPENDIX B DEVELOPMENT TOOLS

The following development tools are available for system development using the μPD784038 Subseries.
Figure B-1 shows the development tools.

- For PC98-NX series
  Unless otherwise specified, products supported by IBM PC/AT™ and compatible machines can be used for the PC98-NX series. When using the PC98-NX series, refer to the explanation of IBM PC/AT and compatible machines.
- For Windows
  Unless otherwise specified, “Windows” indicates the following OSs.
  - Windows 3.1
  - Windows 95, 98, 2000
  - Windows NT™ Ver. 4.0

Figure B-1 Development Tool Configuration (1/2)

(1) When using in-circuit emulator IE-78K4-NS

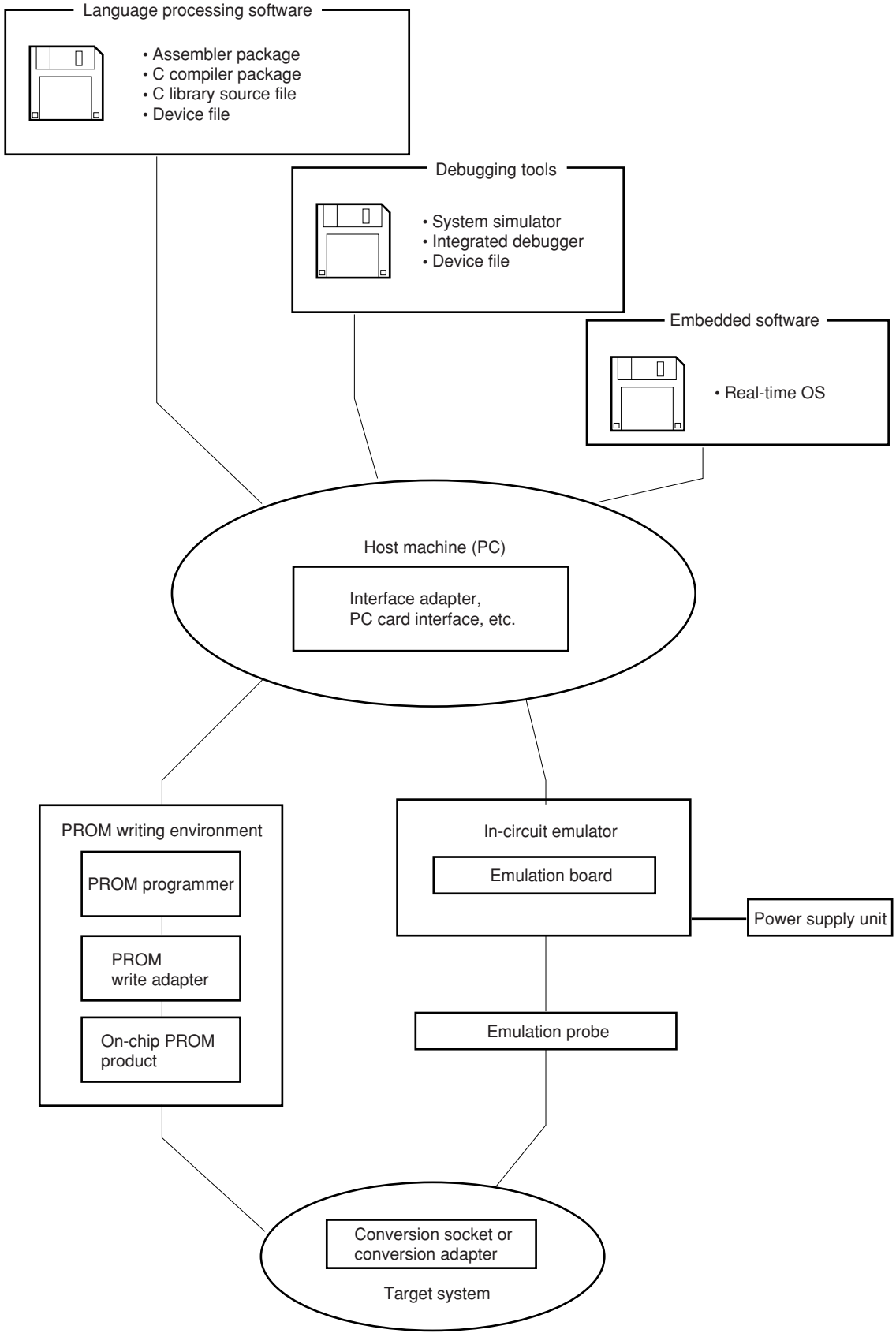
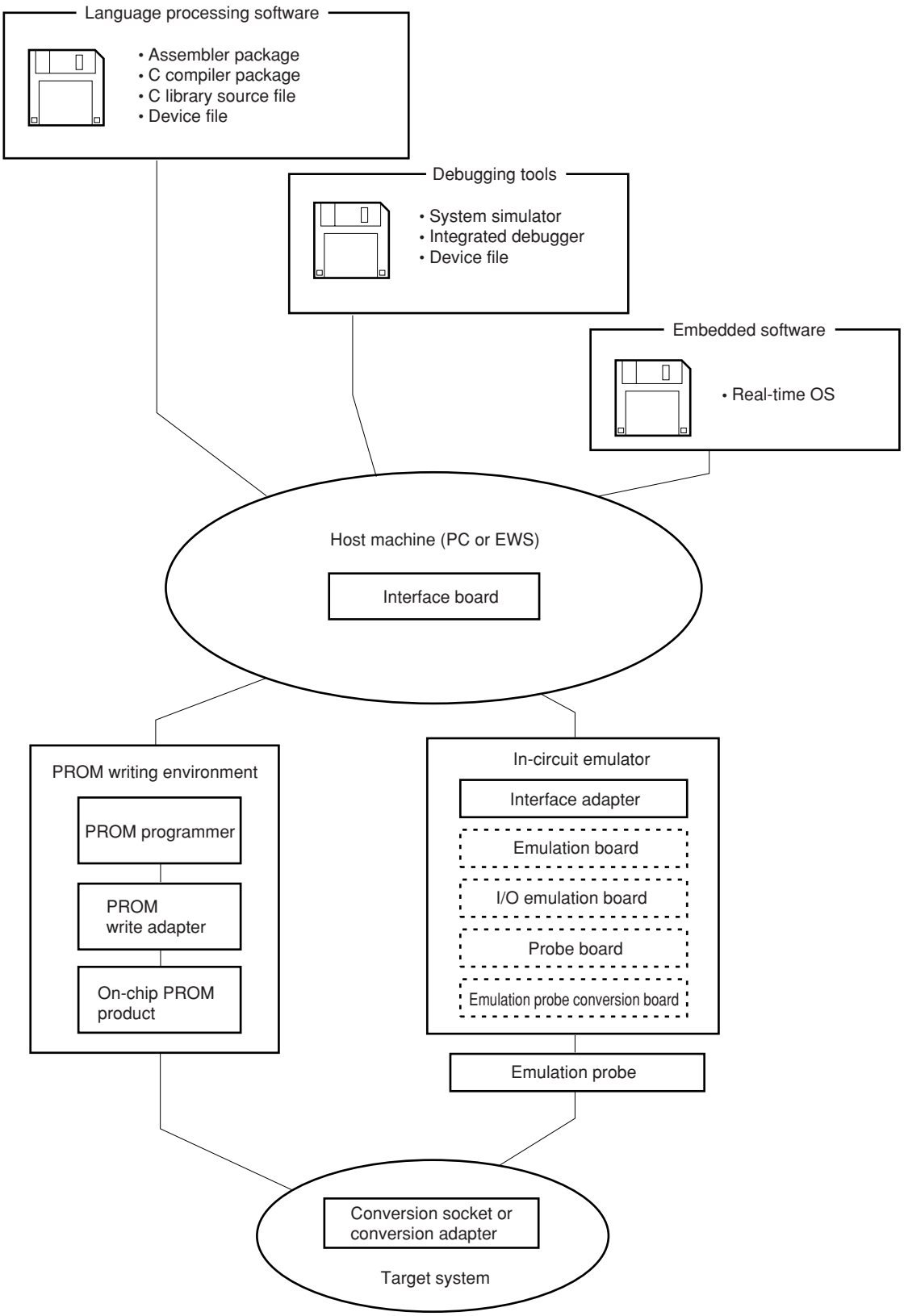


Figure B-1 Development Tool Configuration (2/2)

(2) When using in-circuit emulator IE-784000-R



Remark Parts enclosed by broken lines vary depending on the product. Refer to **B.3.1 Hardware**.

B.1 LANGUAGE PROCESSING SOFTWARE

SP78K4 78K/IV Series software package	Development tools (software) common to the 78K/IV Series are combined in this package.
	Part number: μS××××SP78K4
RA78K4 Assembler package	Program that converts a program written in mnemonic to an executable microcontroller object code. In addition, this assembler package has functions to create symbol tables and optimize branch instructions, etc. automatically. Use this in combination with the device file (DF784038) sold separately. **<Caution on using in PC environment>** Although the assembler package is a DOS-based application, it can be used in the Windows environment by using the Project Manager (included in the assembler package) on Windows.
	Part number: μS××××RA78K4
CC78K4 C compiler package	Program that converts a program written in C language to an executable microcontroller object code. Use this in combination with the assembler package and device file sold separately. **<Caution on using in PC environment>** Although the C compiler package is a DOS-based application, it can be used in the Windows environment by using the Project Manager (included in the assembler package) on Windows.
	Part number: μS××××CC78K4
DF784038[Note] Device file	File containing device-specific information. Use this in combination with the tools sold separately (RA78K4, CC78K4, SM78K4, ID78K4-NS, ID78K4). The supported OS and host machine differ depending on the tool combinations.
	Part number: μS××××DF784038
CC78K4-L C library source file	Function source file configuring the object library included in the C compiler package. This is required when changing the object library included in the C compiler package to accord with the user's specifications. Because this is a source file, the operating environment does not depend on the OS.
	Part number: μS××××CC78K4-L

Note The DF784038 can be used commonly for all the RA78K4, CC78K4, SM78K4, ID78K4-NS, and ID78K4.

Remark The ××××part number differs depending on the host machine and operating system used.

μS××××SP78K4

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Japanese Windows	CD-ROM
BB17		English Windows	

μS××××RA78K4

μS××××CC78K4

××××	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT compatibles	Japanese Windows	3.5-inch 2HD FD
BB13		English Windows	
AB17		Japanese Windows	CD-ROM
BB17		English Windows	
3P17	HP9000 series 700™	HP-UX™ (Rel. 10.10)	
3K17	SPARCstation™	SunOS™ (Rel. 4.1.4), Solaris™ (Rel. 2.5.1)	

μS××××DF784038

μS××××CC78K4-L

××××	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT compatibles	Japanese Windows	3.5-inch 2HD FD
BB13		English Windows	
3P16	HP9000 series 700	HP-UX (Rel. 10.10)	DAT
3K13	SPARCstation	SunOS (Rel. 4.1.4)	3.5-inch 2HD FD
3K15		Solaris (Rel. 2.5.1)	1/4-inch CGMT

B.2 PROM WRITING TOOLS

(1) Hardware

PG-1500	This PROM programmer can program single-chip microcontrollers containing PROM in a stand-alone mode or under the control of the host machine, when a board supplied as an accessory and an optionally available PROM programmer adapter are connected. It can also program representative PROMs, from 256-Kbit to 4-Mbit models.
PA-78P4026GC PA-78P4038GK	This is a PROM programmer adapter for the μPD78P4038 and is connected to the PG-1500. PA-78P4026GC: for 80-pin plastic QFP (GC-3B9, GC-8BT type) PA-78P4038GK: 80-pin plastic TQFP (fine pitch) (GK-9EU type)

(2) Software

PG-1500 controller	The PG-1500 controller connects the PG-1500 and the host machine with a serial and parallel interfaces to control the PG-1500 on the host machine.			
	Host Machine	OS	Supply Media	Part Number
	PC-9800 series	MS-DOS (Ver.3.30 to Ver.6.2 Note1)	3.5" 2HD	μS5A13PG1500
			5" 2HD	μS5A10PG1500
	IBM PC/AT and compatible machines	**Note 2**	3.5" 2HD	μS7B13PG1500
			5" 2HD	μS7B10PG1500

Notes 1. Ver. 5.0 or above of MS-DOS has a task swap function, but this function cannot be used with the above software.

2. The following OSs for the IBM PC are supported. (Ver. 5.0 or above of MS-DOS has a task swap function, but this function cannot be used with the above software.)

OS	Version
PC DOS	Ver.5.02 to Ver.6.3 J6.1/V to J6.3/V (Only the English version is supported.)
MS-DOS	Ver.5.0 to Ver.6.22 5.0/V to 6.2/V (Only the English version is supported.)
IBM DOS™	J5.02/V (Only English version is supported.)

B.3 DEBUGGING TOOLS

B.3.1 Hardware (1/2)

(1) When using in-circuit emulator IE-78K4-NS

IE-78K4-NS In-circuit emulator	In-circuit emulator used to debug hardware and software when developing application systems using the 78K/IV Series. Supports the integrated debugger (ID78K4-NS). Use in combination with an interface adapter to connect to the power supply unit, emulation probe, and host machine.
IE-70000-MC-PS-B Power supply unit	Adapter to supply power from a socket of AC 100 V to 240 V
IE-70000-98-IF-C Interface adapter	Interface adapter required when a PC-9800 series PC (except notebook type) is used as the host machine for the IE-78K4-NS (C bus supported)
IE-70000-CD-IF-A PC card Interface	PC card and interface cable required when a notebook PC is used as the host machine for the IE-78K4-NS (PCMCIA socket supported)
IE-70000-PC-IF-C Interface adapter	Interface adapter required when using an IBM PC/AT compatible as the host machine for the IE-78K4-NS (ISA bus supported)
IE-70000-PCI-IF-A Interface adapter	Interface adapter required when using a PC that incorporates PCI bus as the host machine for the IE-78K4-NS
IE-784038-NS-EM1 Emulation board	Board to emulate the peripheral hardware specific to device. Used in combination with an in-circuit emulator.
NP-80GK Emulation probe	Probe used to connect the in-circuit emulator and the target system. This is for an 80 pin plastic TQFP (fine pitch) (GK-9EU type).
TGK-080SDW Conversion adapter (refer to **Figure B-4**)	Conversion adapter to connect the NP-80GK and a target system board on which an 80-pin plastic TQFP (fine pitch) (GK-9EU type) can be mounted
NP-80GC-TQ NP-H80GC-TQ Emulation probe	Probe used to connect the in-circuit emulator and the target system. This is for an 80-pin plastic QFP (GC-3B9 or GC-8BT type).
TGC-080SBP Conversion socket (refer to **Figure B-3**)	Conversion socket to connect the NP-80GC-TQ or NP-H80GC-TQ and a target system board on which an 80-pin plastic QFP (GC-3B9 or GC-8BT type) can be mounted

Remarks 1. NP-80GK, NP-80GC-TQ, and NP-H80GC-TQ are products made by Naito Densei Machida Mfg.Co., Ltd.
For further information, contact Naito Densei Machida Mfg. Co., Ltd. (TEL: +81-45-475-4191)

2. TGK-080SDW and TGC-080SBP products made by Tokyo Eletech Corporation.
For further information, contact Daimaru Kogyo, Ltd.
Tokyo Electronics Department (TEL: +81-3-3820-7112)
Osaka Electronics Department (TEL: +81-6-6244-6672)

3. The TGK-080SDW and TGC-080SBP are sold individually.

B.3.1 Hardware (2/2)

(2) When using in-circuit emulator IE-784000-R

IE-784000-R In-circuit emulator	The IE-784000-R is an in-circuit emulator common to the 78K/IV Series, and is used in combination with IE-784000-R-EM and IE-784038-R-EM1, which are sold separately. This in-circuit emulator debugs the connected host machine. An integrated debugger (ID78K4) and device file (sold separately) are required to enable debugging in C language and structured assembly language at the source program level. More efficient debugging and program verification is possible with functions such as C0 coverage. Connect to a host machine via Ethernet™ or a dedicated bus. An interface adapter (sold separately) is required for connection.
IE-70000-98-IF-C Interface adapter	Interface adapter required when a PC-9800 series (except notebook type PC) is used as the host machine for the IE-784000-R (C bus supported)
IE-70000-PC-IF-C Interface adapter	Interface adapter required when using an IBM PC/AT compatible as the host machine (ISA bus supported)
IE-78000-R-SV3 Interface adapter	Interface adapter and cable required when an EWS is used as the host machine for the IE-784000-R. Connect to a board inside the IE-784000-R. Note that 10Base-5 is supported as the Ethernet. A commercial conversion adapter is required for other systems.
IE-784000-R-EM	Emulation board common to 78K/IV Series
IE-784038-R-EM1 Emulation board	Board to emulate peripheral hardware specific to device
IE-78K4-R-EX2 Emulation probe conversion board	Conversion board for 80-pin packages required when using the IE-784038-R-EM1 on IE-784000-R
EP-78054GK-R Emulation probe	Probe to connect the in-circuit emulator and the target system. For 80-pin plastic TQFP (fine pitch)(GK-9EU type).
TGK-080SDW adapter (refer to **Figure B-7**)	Conversion adapter to connect the EP-78054GK-R and a target system board on Conversion which an 80-pin plastic TQFP (fine pitch)(GK-9EU type) can be mounted
EP-78230GC-R Emulation probe	Probe to connect the in-circuit emulator and the target system. For 80-pin plastic QFP (GC-3B9 or GC-8BT type).
EV-9200GC-80 Conversion socket (refer to **Figures B-5** and **B-6**)	Conversion socket to connect the EP-78230GC-R and a target system board on which an 80-pin plastic QFP (GC-3B9 or GC-8BT type) can be mounted

Remarks 1. TGK-080SDW is a product made by Tokyo Eletech Corporation.
For further information, contact Daimaru Kogyo, Ltd.
Tokyo Electronics Department (TEL: +81-3-3820-7112)
Osaka Electronics Department (TEL: +81-6-6244-6672)

2. The EV-9200GC-80 is sold in sets of 5.

3. The TGK-080SDW is sold individually.

B.3.2 Software

SM78K4 System simulator	This enables debugging at the C source level or assembler level while simulating operation of the target system on the host machine. The SM78K4 operates on Windows. By using the SM78K4, logic verification and performance verification can be performed separately to hardware development without using an in-circuit emulator, thus improving development efficiency and software quality. Use the SM78K4 in combination with the device file (DF784038) sold separately.
	Part number: μS××××SM78K4

Remark The ×××× part number differs depending on the host machine and operating system used.

μS××××SM78K4

××××	Host Machine	OS	Supply Medium
AB13	IBM PC/AT compatible	Japanese Windows	3.5-inch 2HC FD
BB13		English Windows	
AB17		Japanese Windows	CD-ROM
BB17		English Windows	

ID78K4-NS Integrated debugger (supporting in-circuit emulator IE-78K4-NS)	Windows and OSF/Motif™ are employed as the GUI for PC and EWS respectively providing users with their unique look and operability. In addition, the enhanced C language supported debug function enables the result of a trace to be displayed at the C language level using the window integration function in which the source program, disassemble display, and memory display are linked to the result of trace. Moreover, the efficiency of debugging programs that use a real-time OS can be raised by installing function expansion modules such as task debuggers and system performance analyzers. Control program to debug the 78K/IV Series. Use these integrated debuggers in combination with the device file (DF784038) sold separately.
ID78K4 Integrated debugger (supporting in-circuit emulator IE-784000-R)	
	Part number: μS××××ID78K4-NS, μS××××ID78K4

Remark The ×××× part number differs depending on the host machine and operating system used.

μS××××ID78K4-NS

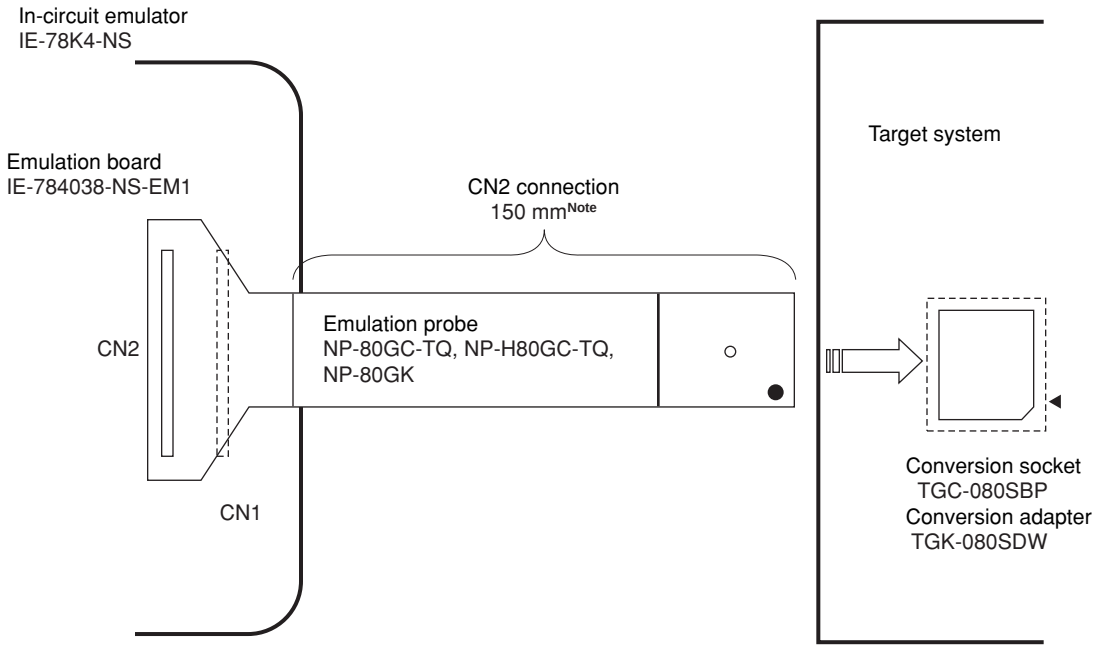
μS××××ID78K4

××××	Host Machine	OS	Supply Medium
AB13	IBM PC/AT compatible	Japanese Windows	3.5-inch 2HC FD
BB13		English Windows	
AB17		Japanese Windows	CD-ROM
BB17		English Windows	

B.4 CAUTIONS ON DESIGNING TARGET SYSTEM

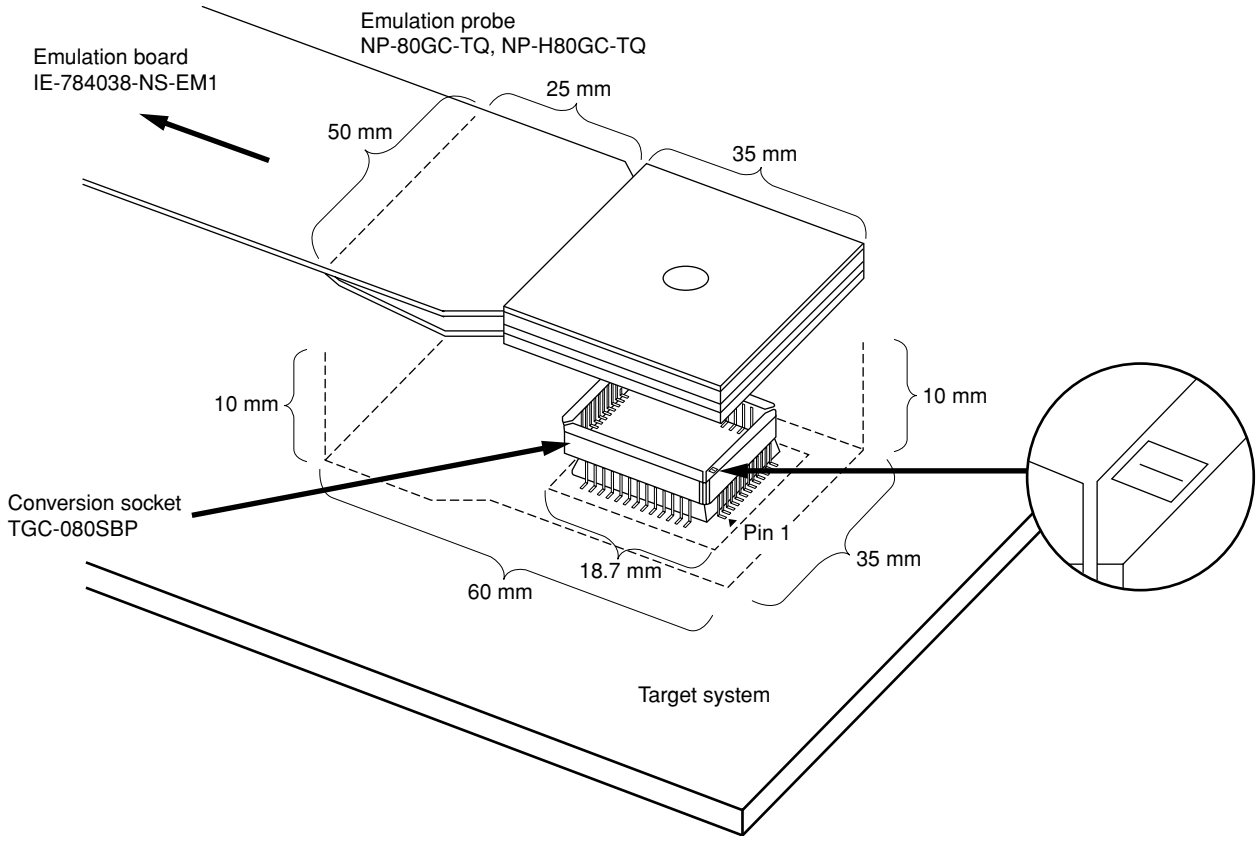
The connection condition diagrams for the emulation probe, conversion socket, and conversion adapter are shown below. Design the system considering the shape of components, etc. to be mounted on the target system in accordance with this configuration.

Figure B-2 Distance Between In-Circuit Emulator and Conversion Socket



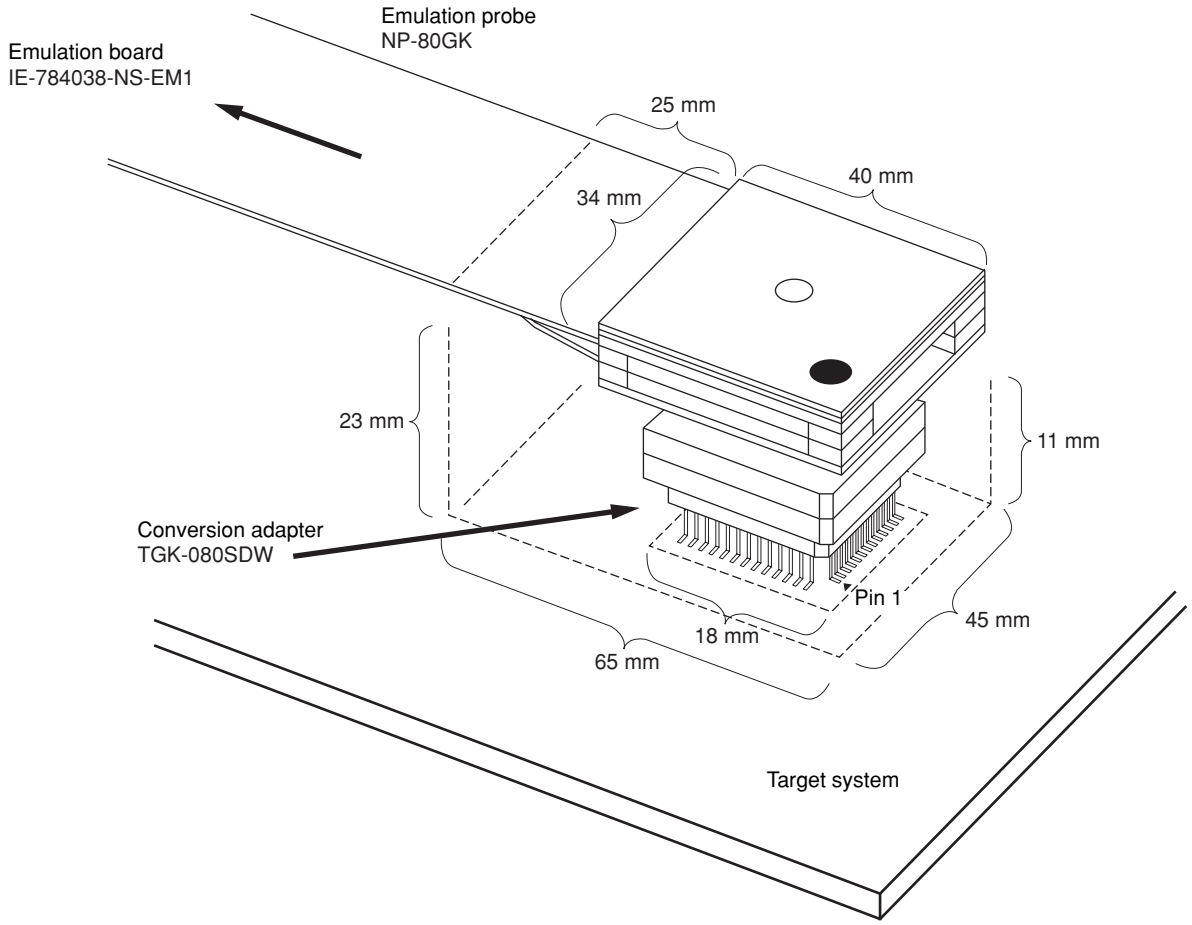
Note 350 mm in case of the NP-H80GC-TQ.

Figure B-3 Target System Connection Conditions (1)



Remark NP-80GC-TQ and NP-H80GC-TQ are products made by Naito Densei Machida Mfg. Co., Ltd.

Figure B-4 Target System Connection Conditions (2)

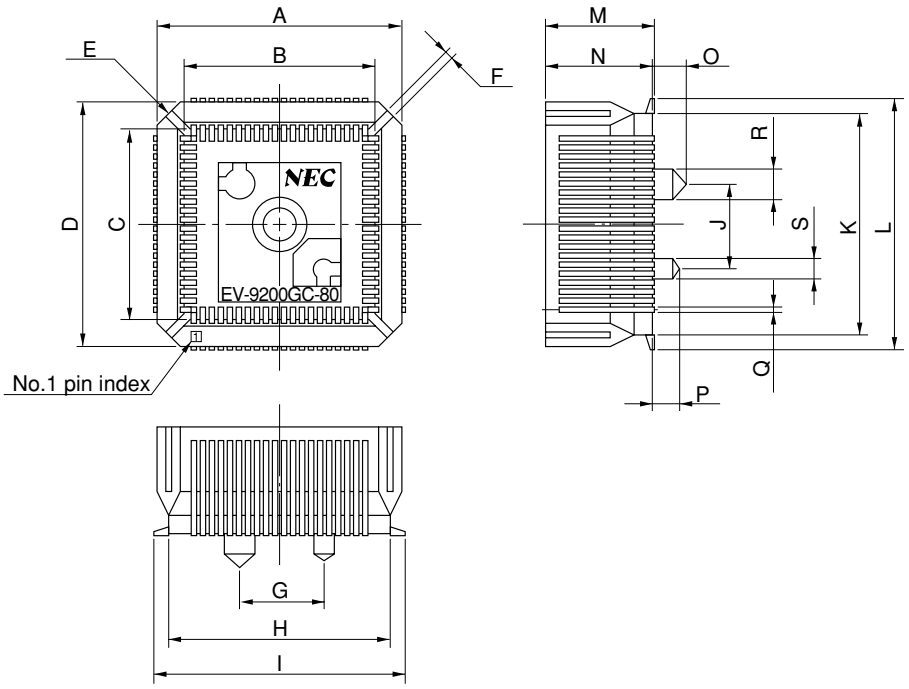


Remark NP-80GK is a product made by Naito Densei Machida Mfg. Co., Ltd.
TGK-080SDW is a product made by TOKYO ELETECH CORPORATION.

B.5 CONVERSION SOCKET (EV-9200GC-80) AND CONVERSION ADAPTER (TGK-080SDW)

(1) The package drawing of the conversion socket (EV-9200GC-80) and recommended board installation pattern

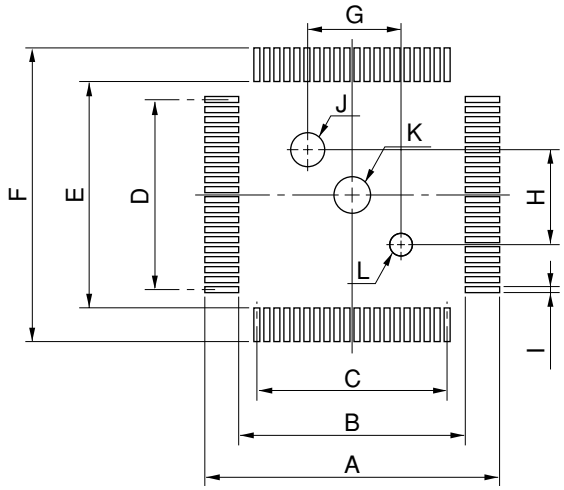
Figure B-5 Package Drawing of EV-9200GC-80 (Reference) (Unit: mm)



EV-9200GC-80-G1E

ITEM	MILLIMETERS	INCHES
A	18.0	0.709
B	14.4	0.567
C	14.4	0.567
D	18.0	0.709
E	4-C 2.0	4-C 0.079
F	0.8	0.031
G	6.0	0.236
H	16.0	0.63
I	18.7	0.736
J	6.0	0.236
K	16.0	0.63
L	18.7	0.736
M	8.2	0.323
N	8.0	0.315
O	2.5	0.098
P	2.0	0.079
Q	0.35	0.014
R	ϕ2.3	ϕ0.091
S	ϕ1.5	ϕ0.059

Figure B-6 Recommended Board Installation Pattern of EV-9200GC-80 (Reference) (Unit: mm)



EV-9200GC-80-P1E

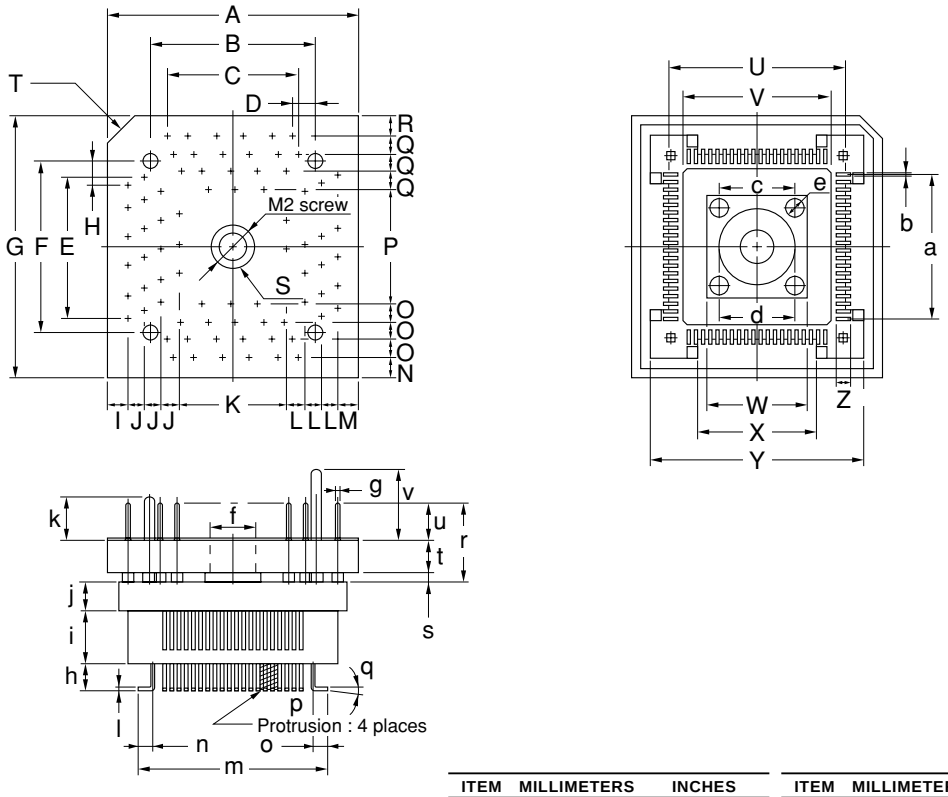
ITEM	MILLIMETERS	INCHES
A	19.7	0.776
B	15.0	0.591
C	$0.65\pm0.02\times19=12.35\pm0.05$	$0.026^{+0.001}_{-0.002}\times0.748=0.486^{+0.003}_{-0.002}$
D	$0.65\pm0.02\times19=12.35\pm0.05$	$0.026^{+0.001}_{-0.002}\times0.748=0.486^{+0.003}_{-0.002}$
E	15.0	0.591
F	19.7	0.776
G	6.0 ± 0.05	$0.236^{+0.003}_{-0.002}$
H	6.0 ± 0.05	$0.236^{+0.003}_{-0.002}$
I	0.35 ± 0.02	$0.014^{+0.001}_{-0.001}$
J	$\phi2.36\pm0.03$	$\phi0.093^{+0.001}_{-0.002}$
K	$\phi2.3$	$\phi0.091$
L	$\phi1.57\pm0.03$	$\phi0.062^{+0.001}_{-0.002}$

Caution Dimensions of mount pad for EV-9200 and that for target device (QFP) may be different in some parts. For the recommended mount pad dimensions for QFP, refer to "SEMICONDUCTOR DEVICE MOUNTING TECHNOLOGY MANUAL" (C10535E).

(2) Package drawing of the conversion adapter (TGK-080SDW)

Combined with the emulation probe and mounted on the board.

Figure B-7 TGK-080SDW Package Drawing (Reference) (Unit: mm)



ITEM	MILLIMETERS	INCHES
A	18.0	0.709
B	11.77	0.463
C	0.5x19=9.5	0.020x0.748=0.374
D	0.5	0.020
E	0.5x19=9.5	0.020x0.748=0.374
F	11.77	0.463
G	18.0	0.709
H	0.5	0.020
I	1.58	0.062
J	1.2	0.047
K	7.64	0.301
L	1.2	0.047
M	1.58	0.062
N	1.58	0.062
O	1.2	0.047
P	7.64	0.301
Q	1.2	0.047
R	1.58	0.062
S	ϕ3.55	ϕ0.140
T	C 2.0	C 0.079
U	12.31	0.485
V	10.17	0.400
W	6.8	0.268
X	8.24	0.324
Y	14.8	0.583
Z	1.4–0.2	0.055–0.008

ITEM	MILLIMETERS	INCHES
a	0.5x19=9.5–0.10	0.020x0.748=0.374–0.004
b	0.25	0.010
c	ϕ5.3	ϕ0.209
d	ϕ5.3	ϕ0.209
e	ϕ1.3	ϕ0.051
f	ϕ3.55	ϕ0.140
g	ϕ0.3	ϕ0.012
h	1.85–0.2	0.073–0.008
i	3.5	0.138
j	2.0	0.079
k	3.0	0.118
l	0.25	0.010
m	14.0	0.551
n	1.4–0.2	0.055–0.008
o	1.4–0.2	0.055–0.008
p	h=1.8 ϕ1.3	h=0.071 ϕ0.051
q	0~5°	0.000~0.197°
r	5.9	0.232
s	0.8	0.031
t	2.4	0.094
u	2.7	0.106
v	3.9	0.154

TGK-080SDW-G1E

Note Made by TOKYO ELETECH Corp.

★

APPENDIX C EMBEDDED SOFTWARE

The following embedded software is available for more efficient program development or maintenance of the μPD784038, 784038Y Subseries.

REAL-TIME OPERATING SYSTEM

RX78K4 real-time OS	This is a real-time OS complying with the μITRON specification. The RX78K4 nucleus and tools to create multiple information tables (configurator) have been added. Use the RX78K4 in combination with the assembler package (RA78K4) and device file (DF784038) (sold separately). **<Caution on using in PC environment>** This real-time OS is a DOS-based application. With Windows, use the RX78K/IV at the DOS prompt.
	Part number: μS××××RX78K4-ΔΔΔΔ

Caution When purchasing the RX78K4, fill out the purchase application and sign the license agreement.

Remark The ×××× and ΔΔΔΔ part numbers vary depending on the host machine and operating system used.

μS××××RX78K4-ΔΔΔΔ

ΔΔΔΔ	Product Overview	Maximum Number Used During Production
001	Evaluation object	Do not use in mass-produced products.
100K	Production object	100,000
001M		1,000,000
010M		10,000,000
S01	Source program	Source program for the production object

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 series	Japanese Windows[Note]	3.5-inch 2HD FD
AB13	IBM PC/AT and compatibles	Japanese Windows[Note]	3.5-inch 2HC FD
BB13		English Windows[Note]	
3P16	HP9000 series 700	HP-UX (Rel.10.10)	DAT (DDS)
3K13	SPARCstation	SunOS (Rel.4.1.4)	3.5-inch 2HC FD
3K15		Solaris (Rel.2.5.1)	1/4-inch CGMT

Note Also operates in DOS environment.

APPENDIX D REGISTER INDEX

D.1 REGISTER INDEX (REGISTER NAME)

D.2 REGISTER INDEX (REGISTER SYMBOL)

APPENDIX E REVISION HISTORY

The history of revisions hitherto made is shown as follows.

(1/3)

Edition	Revisions	Chapter
Second	• Addition of description on μPD784031 and 784031Y • Addition of 80-pin plastic QFP (14 × 14 mm, 1.4 mm thick)	General
	Addition of description on μPD784908 Subseries and 78F4943 Subseries to **78K/IV Series Product Development Diagram**	**CHAPTER 1 GENERAL**
	Division of description on V_{DD} and V_{SS} pins into following two: • V_{DD} → V_{DD0}: Positive power supply pin of ports V_{DD1}: Positive power supply pin of function blocks other than ports • V_{SS} → V_{SS0}: GND pin of ports V_{SS1}: GND pin of function blocks other than ports	**CHAPTER 2 PIN FUNCTIONS**
	Addition of note on internal memory size switching register (IMS)	**CHAPTER 3 CPU ARCHITECTURE**
	8.5 EXTERNAL EVENT COUNTER FUNCTION in **CHAPTER 8 TIMER/COUNTER 0** Correction of TM0 timing of Figure 8-10 Timer/Counter 0 External Event Count Timing	**CHAPTER 8 TIMER/COUNTER 0**
	9.5 EXTERNAL EVENT COUNTER FUNCTION in **CHAPTER 9 TIMER/COUNTER 1** Correction of TM1 timing of Figure 9-10 Timer/Counter 1 External Event Count Timing	**CHAPTER 9 TIMER/COUNTER 1**
	10.5 EXTERNAL EVENT COUNTER FUNCTION in **CHAPTER 10 TIMER/COUNTER 2** Correction of TM2 timing of Figure 10-11 Timer/Counter 2 External Event Count Timing	**CHAPTER 10 TIMER/COUNTER 2**
	Low-speed conversion (f_{CLK} = 16 MHz) 240/f_{CLK} (15μs) → 180/f_{CLK} (11.25 μs)	**CHAPTER 14 A/D CONVERTER**
	• Addition of notes on switching of MSB/LSB first • Change of **Table 17-4 Example of BRGC Settings When Baud Rate Generator is Used**	**CHAPTER 17 ASYNCHRONOUS SERIAL INTERFACE/3-WIRE SERIAL I/O**
	Unification of CLO pin to CLKOUT pin	**CHAPTER 20 CLOCK OUTPUT FUNCTION**
	• Addition of notes of external wait function • Change of **Figure 23-10 Programmable Wait Control Register (PWC1, PWC2) Format**	**CHAPTER 23 LOCAL BUS INTERFACE FUNCTION**
	• Addition of notes on releasing standby mode • Addition of **Figure 24-5 Operation after HALT Mode Release** • Addition of **Figure 24-6 Operation after STOP Mode Release** • Addition of **Figure 24-9 Operation after IDLE Mode Release**	**CHAPTER 24 STANDBY FUNCTION**
	Addition of **APPENDIX E GENERAL INDEX**	**APPENDIX E GENERAL INDEX**

(2/3)

Edition	Revisions	Chapter
Third	• Change in **78K/IV Series Product Development Diagram**	**CHAPTER 1 GENERAL**
	• Addition of **Table 3-6 Limits of Reading Timer Register**	**CHAPTER 3 CPU ARCHITECTURE**
	• Addition of **Table 8-5 Limits of Reading Timer Register**	**CHAPTER 8 TIMER/COUNTER 0**
	• Addition of **Table 9-4 Limits of Reading Timer Register** • Addition of **Figures 9-5 and 9-20 Example of Occurrence of Unnecessary Interrupt Requests from Compare Register**, and **Caution**	**CHAPTER 9 TIMER/COUNTER**
	• Addition of **Table 10-5 Limits of Reading Timer Register** • Addition of **Figures 10-5** and **10-22 Example of Occurrence of Unnecessary Interrupt Requests from Compare Register**, and **Caution**	**CHAPTER 10 TIMER/COUNTER 2**
	• Addition of **Table 11-2 Limits of Reading Timer Register**	**CHAPTER 11 TIMER 3**
	• Change from "If the STOP mode or IDLE mode is entered as the result of an inadvertent program loop" to "If the STOP mode, HALT mode, or IDLE mode is entered as the result of an inadvertent program loop" in (2) <5> in **12.4.1 General Cautions on Use of Watchdog Timer**	**CHAPTER 12 WATCHDOG TIMER**
	• Change of **Figure 14-11 Hardware Start Select Mode A/D Conversion Operation**	**CHAPTER 14 A/D CONVERTER**
	• Addition of notes on disabling reception completion interrupt in case of reception error and how to calculate wait time	**CHAPTER 17 ASYNCHRONOUS SERIAL INTERFACE/3-WIRE SERIAL I/O**
	• Change and addition of “The watchdog timer must not be used to release the standby mode (STOP or IDLE mode” to “The watchdog timer must not be used to release the standby mode (STOP, HALT, or IDLE mode” • Deletion of watchdog timer of "non-maskable interrupt requests (NMI pin input and watchdog timer)"	**CHAPTER 24 STANDBY FUNCTION**
	• Addition of **Figure B-4 Drawing of TGK-080SDW**	**APPENDIX B DEVELOPMENT TOOL**
Fourth	Addition of the following special grade products to the target products • μPD784031GC(A)-3B9, 784035GC(A)-×××-3B9, 784036GC(A)-×××-3B9 Deletion of the following packages • μPD784031GC-3B9, 784031GK-BE9, 784035GC-×××-3B9, 784035GK-×××-BE9, 784036GC-×××-3B9, 784036GK-×××-BE9, 784037GC-×××-3B9, 784037GK-×××-BE9, 784038GC-×××-3B9, 784038GK-×××-BE9, 78P4038GC-3B9, 78P4038GC-×××-3B9, 78P4038GC-×××-8BT, 78P4038GK-BE9, 78P4038GK-×××-BE9, 78P4038KK-T • μPD784031YGC-3B9, 784031YGK-BE9, 784035YGC-×××-3B9, 784035YGK-×××-BE9, 784036YGC-×××-3B9, 784036YGK-×××-BE9, 784037YGC-×××-3B9, 784037YGK-×××-BE9, 784038YGC-×××-3B9, 784038YGK-×××-BE9, 78P4038YGC-3B9, 78P4038YGC-×××-3B9, 78P4038YGC-×××-8BT, 78P4038YGK-BE9, 78P4038YGK-×××-BE9, 78P4038YKK-T Addition of the following packages • μPD784031GK-9EU, 784035GK-×××-9EU, 784036GK-×××-9EU, 784037GK-×××-9EU, 784038GK-×××-9EU, 78P4038GK-9EU\ • μPD784031YGK-9EU, 784035YGK-×××-9EU, 784036YGK-×××-9EU, 784037YGK-×××-9EU, 784038YGK-xxx-9EU, 78P4038YGK-9EU	Throughout

(3/3)

Edition	Revisions	Chapter
Fourth	• Update of **78K/IV Series Product Development Diagram** • Addition and deletion of products in **1.2 ORDERING INFORMATION AND QUALITY GRADES** • Addition of **1.7 DIFFERENCES BETWEEN STANDARD-GRADE PRODUCTS AND SPECIAL-GRADE PRODUCTS**	**CHAPTER 1 GENERAL**
	Addition of caution on compare register CR00 match interrupt to **8.9 CAUTIONS**	**CHAPTER 8 TIMER/COUNTER 0**
	Addition of caution on compare register CR10 match interrupt to **9.8 CAUTIONS**	**CHAPTER 9 TIMER/COUNTER 1**
	Addition of caution on compare register CR20 match interrupt to 1**0.10 CAUTIONS**	**CHAPTER 10 TIMER/COUNTER 2**
	Modification of description in **Figure 14-3 A/D Converter Mode Register (ADM) Format**	**CHAPTER 14 A/D CONVERTER**
	Addition of caution on successive reception in 3-wire serial I/O mode to **17.5 CAUTIONS**	**CHAPTER 17 ASYNCHRONOUS SERIAL INTERFACE/3-WIRE SERIAL I/O**
	• Modification of **Figure 18-6 3-Wire Serial I/O Mode Timing** • **18.6 CAUTIONS** • Addition of caution on transmit data write in 3-wire serial I/O mode • Addition of caution on serial clock count operation in 3-wire serial I/O mode • Addition of caution on serial clock output in 3-wire serial I/O mode • Addition of caution on successive reception in 3-wire serial I/O mode	**CHAPTER 18 3-WIRE/2-WIRE SERIAL I/O MODE**
	Addition of description to **21.2 EDGE DETECTION FOR PINS P20, P25 AND P26**	**CHAPTER 21 EDGE DETECTION FUNCTION**
	Addition of chapter	**CHAPTER 28 ELECTRICAL SPECIFICATIONS**
	Addition of chapter	**CHAPTER 29 PACKAGE DRAWINGS**
	Addition of chapter	**CHAPTER 30 RECOMMENDED SOLDERING CONDITIONS**
	Addition of description in **Table A-1 Differences with μPD784026 Subseries**	**APPENDIX A DIFFERENCES WITH μPD784026 SUBSERIES**
	Modification of description	**APPENDIX B DEVELOPMENT TOOLS**
	Modification of description	**APPENDIX C EMBEDDED SOFTWARE**