To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

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

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

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

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

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

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

RENESAS

User's Manual

μPD178078, 178098A Subseries

8-Bit Single-Chip Microcontrollers

μPD178076
μPD178078
μPD178096A
μPD178098A
μPD178F098

Document No. U12790EJ2V0UD00 (2nd edition)
Date Published October 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.

IEBus is a trademark of NEC Electronics Corporation.
Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
PC/AT is a trademark of IBM Corporation.
HP9000 Series 700 and HP-UX are trademarks of Hewlett-Packard Company.
SPARCstation is a trademark of SPARC International, Inc.
Solaris and SunOS are trademarks of Sun Microsystems, Inc.
TRON stands for The Realtime Operating system Nucleus.
ITRON is an abbreviation of Industrial TRON.

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 application circuits and their parameters are for reference only and are not intended for use in actual design-ins.

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.

- **The information in this document is current as of March, 2003. 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.

[GLOBAL SUPPORT]
http://www.necel.com/en/support/support.html

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

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

- **Sucursal en España**
  Madrid, Spain
  Tel: 091-504 27 87
- **Succursale Française**
  Vélizy-Villacoublay, France
  Tel: 01-30-67 58 00
- **Filiale Italiana**
  Milano, Italy
  Tel: 02-66 75 41
- **Branch The Netherlands**
  Eindhoven, The Netherlands
  Tel: 040-244 58 45
- **Tyskland Filial**
  Taeby, Sweden
  Tel: 08-63 80 820
- **United Kingdom Branch**
  Milton Keynes, UK
  Tel: 01908-691-133

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-558-3737

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

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

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

J03.4

Major Revision in This Edition (1/2)

Page	Description
Throughout	Addition of μPD178096A and 178098A
	Modification of μPD178F098 from under development to developed
p.35	Modification of **1.5 Development of 8-Bit DTS Series**
p.48	Modification of pin handling in **2.2.26 V_{PP} (μPD178F098 only)**
pp.50 to 53	Modification of **Table 2-1 Pin I/O Circuit Types** and **Figure 2-1 Pin I/O Circuits**
p.59	Addition of description of programming area in **3.1.2 Internal data memory space**
pp.66 and 67	Modification of **Figure 3-10 Data to Be Saved to Stack Memory** and **Figure 3-11 Data to Be Restored from Stack Memory**
p.82	Modification of **[Example]** in **3.4.4 Short direct addressing**
pp.85 to 87	Addition of **[Illustration]** to **3.4.7 Based addressing**, **3.4.8 Based indexed addressing**, and **3.4.9 Stack addressing**
p.109	Addition of description of output latches after reset to **4.4 Port Function Operations**
pp.122 and 123	**6.2 Configuration of 16-Bit Timer/Event Counter 0** • Addition of **Cautions** to **(2) 16-bit capture/compare register 00 (CR00)** • Addition of **Table 6-3 CR01 Capture Trigger and Valid Edge of TI00 Pin (CRC02 = 1)** • Addition of **Caution** to **(3) 16-bit capture/compare register 01 (CR01)**
p.128	Addition of **Caution** to **Figure 6-5 Format of Prescaler Mode Register 0 (PRM0)**
p.144	**6.4.5 One-shot pulse output operation** • Modification of **Figure 6-26 Timing of One-Shot Pulse Output Operation with Software Trigger** • Addition of **Note** to **(2) One-shot pulse output with external trigger**
pp.149 to 155	Addition of **6.5 Program List**
p.157	Addition of **(6) (c) One-shot pulse output function** to **6.6 Notes on 16-Bit Timer/Event Counter 0**
pp.163 and 164	**7.2 Configuration of 8-Bit Timer/Event Counters 50 and 51** • Addition of **Note** to **(1) 8-bit timer counters 50 and 51 (TM50 and TM51)** • Addition of description of PWM mode to **(2) 8-bit compare registers 50 and 51 (CR50 and CR51)**
pp.181 to 183	Addition of **7.5 Program List**
p.215	Addition of **(4) Noise countermeasures** and **(6) Input impedance of ANI0 to ANI7 pins** to **11.5 A/D Converter Cautions**
p.348	Addition of **Figure 16-2 Block Diagram of Baud Rate Generator**
p.361	Addition of **Caution** to **Figure 16-6 Permissible Error of Baud Rate Allowing for Sampling Error (k = 0)**
pp.369 and 382	Addition of **Caution** about inversion of IEBus protocol and signal inside the microcontroller to **17.1.6 Transfer format of IEBus** and **17.1.8 Bit format**
pp.378 and 379	Modification of **Note** and **Caution** in **17.1.6 (9) Acknowledge bit**
p.382	Addition of description of lock setting conditions and lock release conditions to **17.1.7 (4) Locking and unlocking**
p.382	Addition of description of timing error detection for each period to **17.1.8 Bit format**
p.383	Addition of **Notes** about automatic master reprocessing to **Table 17-7 Comparison of Existing and Simple IEBus Interface Functions**
pp.387 to 389, 391, 392, and 394 to 399	**17.4.2 Description of internal registers** • Explanation of each register thoroughly modified and **Note** added • Addition of figures of interrupt timing to **Figures 17-16** to **17-19** • Addition of **Figure 17-23 Example of Broadcast Communication Flag Operation** • Addition of **Table 17-9 Reset Conditions of Flags in ISR Register**

Major Revision in This Edition (2/2)

Page	Description
pp.406, 408, and 409	**17.5 Interrupt Operations of IEBus Controller** • Thorough modification of contents • Addition of **17.5.3 Communication error source processing list**
p.413	Addition of description of wait of slave unit to **17.6.2 Master reception**
p.459	Correction of description of drive type of EO1 pin in **19.4.1 Operation of each block of PLL frequency synthesizer**
p.488	Correction of **Note** in **Table 22-1 Hardware Status After Reset**
p.496	Thorough modification of descriptions of flash memory programming as **23.3 Flash Memory Features**
pp.520 to 538	Addition of **CHAPTER 25 ELECTRICAL SPECIFICATIONS**
p.539	Addition of **CHAPTER 26 PACKAGE DRAWING**
p.540	Addition of **CHAPTER 27 RECOMMENDED SOLDERING CONDITIONS**
pp.541 to 551	Thorough modification of descriptions in **APPENDIX A DEVELOPMENT TOOLS** Deletion of **EMBEDDED SOFTWARE**
pp.552 to 557	Addition of **APPENDIX B REGISTER INDEX**
pp.558 and 559	Addition of **APPENDIX C REVISION HISTORY**

The mark ★ shows major revised points.

PREFACE

Readers

This manual has been prepared for user engineers who want to understand the functions of the μPD178078 and 178098A Subseries in order to design and develop its application systems and programs.

Purpose

This manual is intended to give users an understanding of the functions described in the Organization below.

Organization

The μPD178078 and 178098A Subseries manual is separated into two parts: this manual and the instruction edition (common to the 78K/0 Series).

μPD178078, 178098A Subseries User's Manual	78K/0 Series Instruction User's Manual
• Pin functions • Internal block functions • Interrupts • Other on-chip peripheral functions • Electrical specifications	• CPU functions • Instruction set • Explanation of each instruction

How to Read This Manual

It is assumed that readers of this manual have general knowledge in the fields of electricity, logic circuits, and microcontrollers.

- ○ To understand the functions in general:
  → Read this manual in the order of the contents.
- ○ To know the μPD178078, 178098A Subseries instruction functions in detail:
  → Refer to the **78K/0 Series Instruction User's Manual (U12326E)**
- ○ How to interpret the register format:
  → The name is defined as a reserved word in the DF178098 and RA78K0, and is defined in the header file named sfrbit.h in the CC78K0 of a bit whose number is in angle brackets (<>).
- ○ To know the electrical specifications of the μPD178078, 178098A Subseries:
  → Refer to **CHAPTER 25 ELECTRICAL SPECIFICATIONS**.

Conventions

Data significance:	Higher digits on the left and lower digits on the right
Active low representation:	$\overline{xxx}$ (overscore over pin or signal name)
Note:	Footnote for item marked with **Note** in the text
Caution:	Information requiring particular attention
Remark:	Supplementary information
Numeric representation:	Binary ... xxxx or xxxxB
	Decimal ... xxxx
	Hexadecimal ... xxxxH

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

- **Documents Related to μPD178078, 178098A Subseries**

Document Name	Document No.
μPD178078, 178098A Subseries User's Manual	This manual
78K/0 Series Instruction User's Manual	U12326E
78K/0 Series Application Note Basics (I)	U12704E

- **Documents Related to Software Development Tools (User's Manuals)**

Document Name		Document No.
RA78K0 Assembler Package	Operation	U14445E
	Language	U14446E
	Structured Assembly Language	U11789E
CC78K0 C Compiler	Operation	U14297E
	Language	U14298E
SM78K Series System Simulator Ver.2.30 or Later	Operation (Windows™ Based)	U15373E
	External Part User Open Interface Specifications	U15802E
ID78K Series Integrated Debugger Ver.2.30 or Later	Operation (Windows Based)	U15185E
RX78K0 Real-Time OS	Fundamentals	U11537E
	Installation	U11536E
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 when designing.

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

Document Name	Document No.
IE-78K0-NS In-Circuit Emulator	U13731E
IE-78K0-NS-A In-Circuit Emulator	U14889E
IE-78K0-NS-PA Performance Board	U16109E
IE-178098-NS-EM1 Emulation Board	U14013E
IE-78001-R-A In-Circuit Emulator	U14142E
IE-78K0-R-EX1 In-Circuit Emulator	To be prepared

Documents Related to Flash Memory Writing

Document Name	Document No.
PG-FP3 Flash Memory Programmer User's Manual	U13502E
PG-FP4 Flash Memory Programmer User's Manual	U15260E

Other Related Documents

Document Name	Document No.
SEMICONDUCTOR SELECTION GUIDE - Products and Packages -	X13769X
Semiconductor Device Mounting Technology Manual	**Note**
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

Note See the "Semiconductor Device Mount Manual" website (http://www.necel.com/pkg/en/mount/index.html).

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 (1/8)

LIST OF FIGURES (2/8)

LIST OF FIGURES (3/8)

LIST OF FIGURES (4/8)

LIST OF FIGURES (5/8)

LIST OF FIGURES (6/8)

LIST OF FIGURES (7/8)

LIST OF FIGURES (8/8)

LIST OF TABLES (1/3)

LIST OF TABLES (2/3)

LIST OF TABLES (3/3)

CHAPTER 1 OUTLINE

1.1 Features

- On-chip high-capacity ROM and RAM

Type / Part Number	Program Memory (ROM)	Data Memory		
		Internal High-Speed RAM	Buffer RAM	Internal Expansion RAM
μPD178076	48 KB	1024 bytes	32 bytes	1024 bytes
μPD178078	60 KB			2048 bytes
μPD178096A	48 KB			1024 bytes
μPD178098A	60 KB			2048 bytes
μPD178F098	60 KB (Flash memory)			

- Instruction set suitable for system control
  - Bit processing across entire address space
  - Multiplication/division instructions
- General-purpose I/O ports: 80 pins
- IEBus™ controller (μPD178096A, 178098A, and 178F098 only)
- Hardware for PLL frequency synthesizer
  - Dual modulus prescaler (160 MHz MAX.)
  - Programmable divider
  - Phase comparator
  - Charge pump
- Frequency counter
- 8-bit resolution A/D converter: 8 channels
- Serial interface: 4-channel clocked
  - 3-wire serial I/O/SBI/2-wire serial I/O/I^2C bus[Note] mode selectable: 1 channel
  - 3-wire serial I/O mode: 1 channel
  - 3-wire serial I/O mode (with automatic transfer/receive function): 1 channel
  - UART mode: 1 channel
    (μPD178076, 178078, and 178F098 only)

Note When using the I^2C bus mode (including when this mode is implemented by software without using the internal hardware), consult NEC Electronics when placing a mask order.

- Timer: 5 channels
  - Basic timer (timer carry FF): 1 channel
  - 16-bit timer/event counter: 1 channel
  - 8-bit timer/event counter: 2 channels
  - Watchdog timer: 1 channel
- Vectored interrupts
  - μPD178076, 178078: 22 sources
  - μPD178096A, 178098A: 21 sources
  - μPD178F098: 24 sources

Item / Part Number	Non-Maskable Interrupt	Maskable Interrupt		Software Interrupt
		External	Internal	
μPD178076	1 source[Note]	8 sources	13 sources[Note]	1 source
μPD178078				
μPD178096A			12 sources[Note]	
μPD178098A				
μPD178F098			15 sources[Note]	

Note Either a non-maskable interrupt or maskable interrupt (internal) can be selected as the interrupt source of the watchdog timer (INTWDT).

- Test input: 1 pin
- Instruction cycle: 0.317/0.635/1.27/2.54/5.08 μs (with 6.3 MHz crystal resonator)
  0.444/0.889/1.778/3.556/7.111 μs (with 4.5 MHz crystal resonator)
- Supply voltage: V_{DD} = 4.5 to 5.5 V (with PLL, CPU operating)
  V_{DD} = 3.5 to 5.5 V (with CPU operating)
- Power-on clear circuit

1.2 Applications

Car stereo

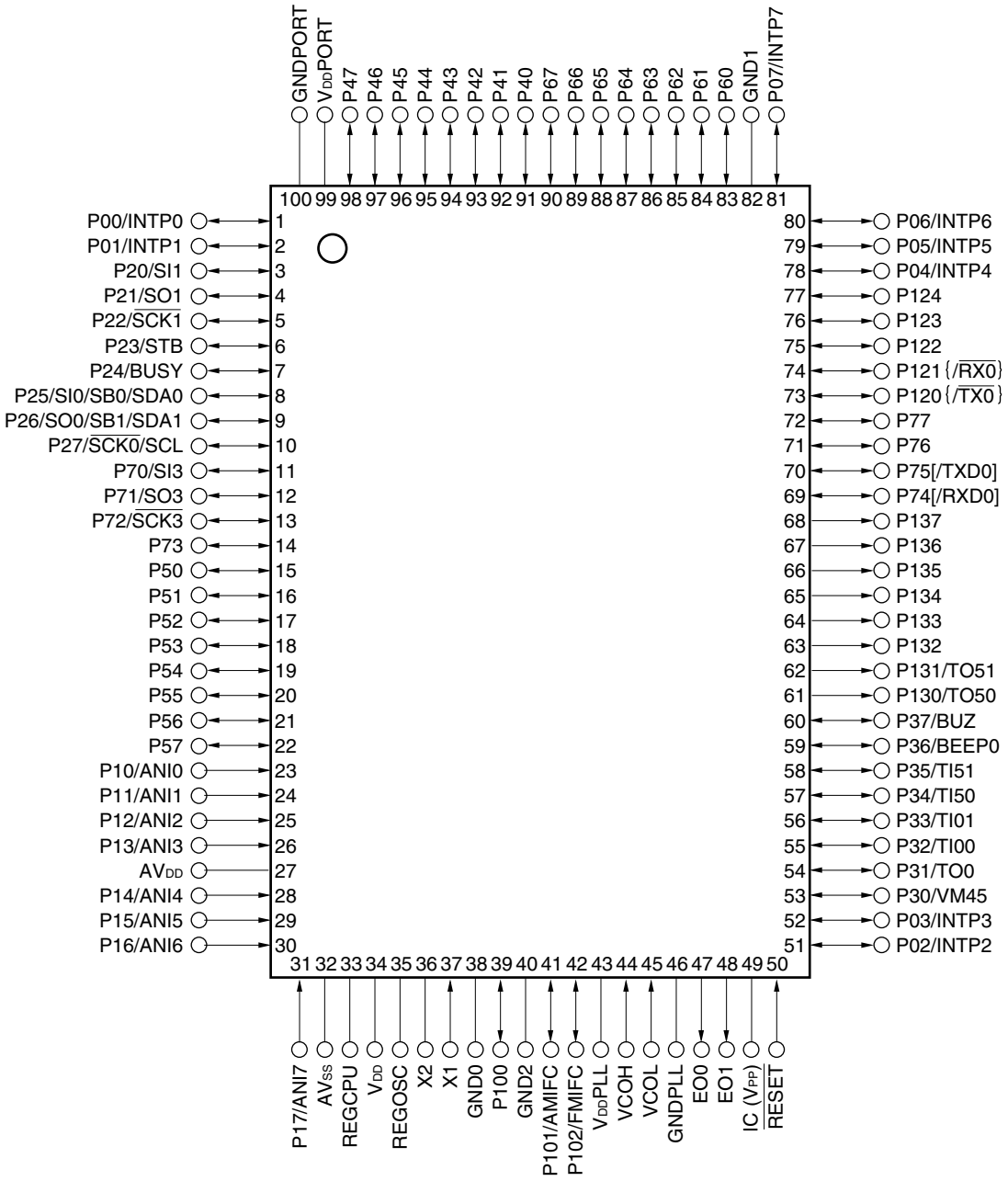
1.3 Ordering Information

Part Number	Package	Internal ROM
μPD178076GF-×××-3BA	100-pin plastic QFP (14 × 20 mm, 0.65-mm pitch)	Mask ROM
μPD178078GF-×××-3BA	100-pin plastic QFP (14 × 20 mm, 0.65-mm pitch)	Mask ROM
μPD178096AGF-×××-3BA	100-pin plastic QFP (14 × 20 mm, 0.65-mm pitch)	Mask ROM
μPD178098AGF-×××-3BA	100-pin plastic QFP (14 × 20 mm, 0.65-mm pitch)	Mask ROM
μPD178F098GF-3BA	100-pin plastic QFP (14 × 20 mm, 0.65-mm pitch)	Flash memory

Remark ××× indicates ROM code suffix. When using I^2C bus mode, E×× is the ROM code suffix.

1.4 Pin Configuration (Top View)

- **100-pin plastic QFP (14 × 20 mm, 0.65 mm pitch)**
  μPD178076GF-×××-3BA, 178078GF-×××-3BA
  μPD178096AGF-×××-3BA, 178098AGF-×××-3BA
  μPD178F098GF-3BA



Cautions 1. Directly connect the IC (Internally Connected) pin to GND0, GND1, or GND2.

2. Keep the voltage at the AV_{DD}, V_{DD}PORT, and V_{DD}PLL pins the same as that at the V_{DD} pin.

3. Keep the voltage at the AV_{SS}, GNDPORT, and GNDPLL pins the same as that at GND0, GND1, or GND2.

4. Connect each of the REGOSC and REGCPU pins to GND via a 0.1 μF capacitor.

Remark ( ) : μPD178F098 only

[ ] : μPD178076 and 178078 only

{ } : μPD178096A and 178098A only

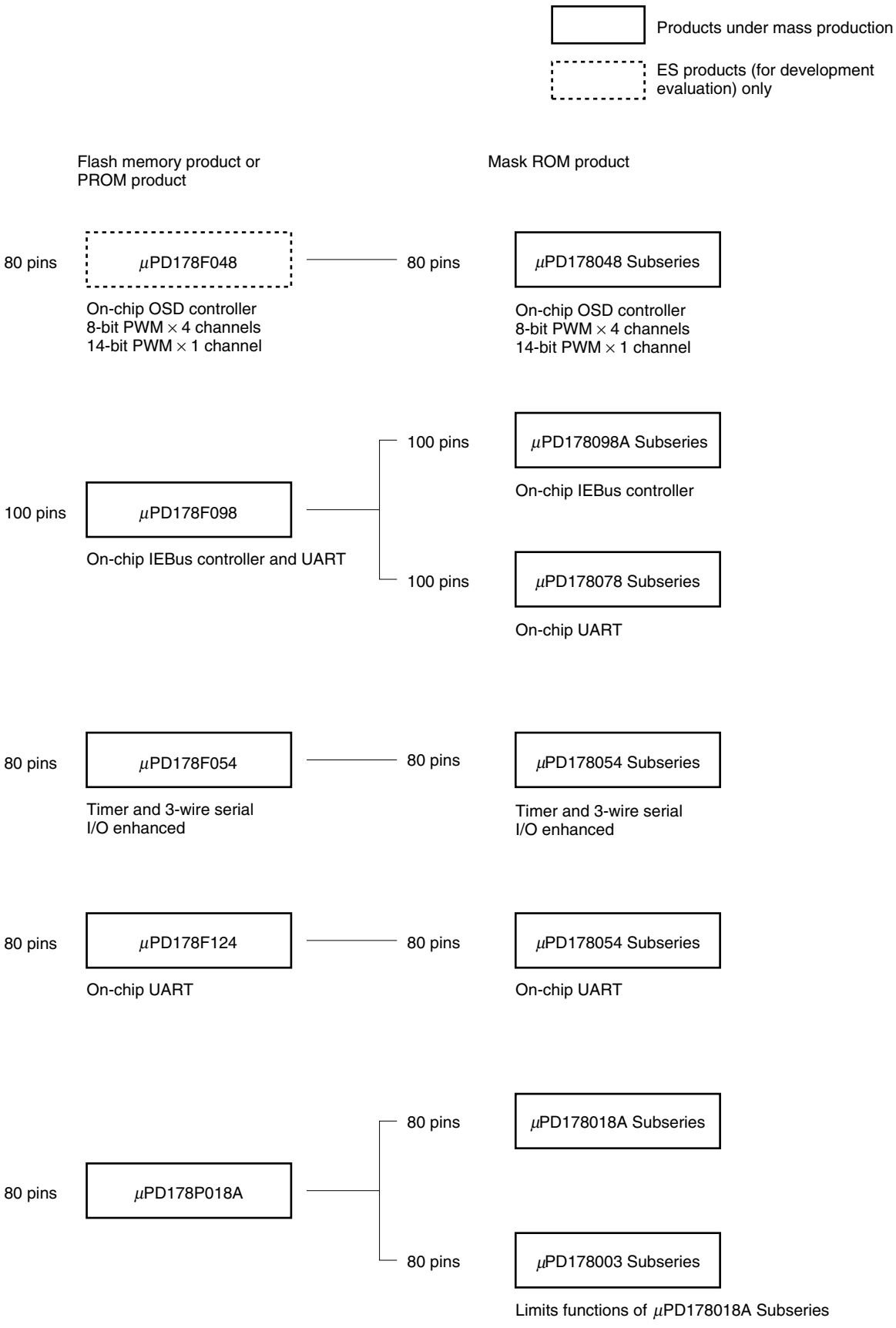
Pin Name	
AMIFC:	AM intermediate frequency counter input
ANI0 to ANI7:	A/D converter input
AV_{DD}:	A/D converter power supply
AV_{SS}:	A/D converter ground
BUSY:	Busy input
BEEP0, BUZ:	Buzzer output
EO0, EO1:	Error out output
FMIFC:	FM intermediate frequency counter input
GNDPLL:	PLL ground
GNDPORT:	Port ground
GND0 to GND2:	Ground
IC:	Internally connected
INTP0 to INTP7:	Interrupt input
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
P100 to P102:	Port 10
P120 to P124:	Port 12
P130 to P137:	Port 13
REGCPU:	Regulator for CPU power supply
REGOSC:	Regulator for oscillator
$\overline{\text{RESET}}$:	Reset input
RXD0[Note 1]:	UART0 serial data input
$\overline{\text{RX0}}$[Note 2]:	IEBus serial data input
SB0, SB1:	Serial data bus input/output
$\overline{\text{SCK0}}$, $\overline{\text{SCK1}}$, $\overline{\text{SCK3}}$:	Serial clock input/output
SCL:	Serial clock input/output
SDA0, SDA1:	Serial data input/output
SI0, SI1, SI3:	Serial data input
SO0, SO1, SO3:	Serial data output
STB:	Strobe output
TI00, TI01:	16-bit timer capture trigger input
TI50, TI51:	8-bit timer clock input
TO0:	16-bit timer output
TO50, TO51:	8-bit timer output
TXD0[Note 1]:	UART0 serial data output
$\overline{\text{TX0}}$[Note 2]:	IEBus serial data output
VCOL, VCOH:	Local oscillation input
V_{DD}PORT:	Port power supply
V_{DD}PLL:	PLL power supply
V_{DD}:	Power supply
VM45:	V_{DD} = 4.5 V monitor output
V_{PP}[Note 3]:	Programming power supply
X1, X2:	Crystal resonator

Notes 1. μPD178076, 178078, and 178F098 only

2. μPD178096A, 178098A, and 178F098 only

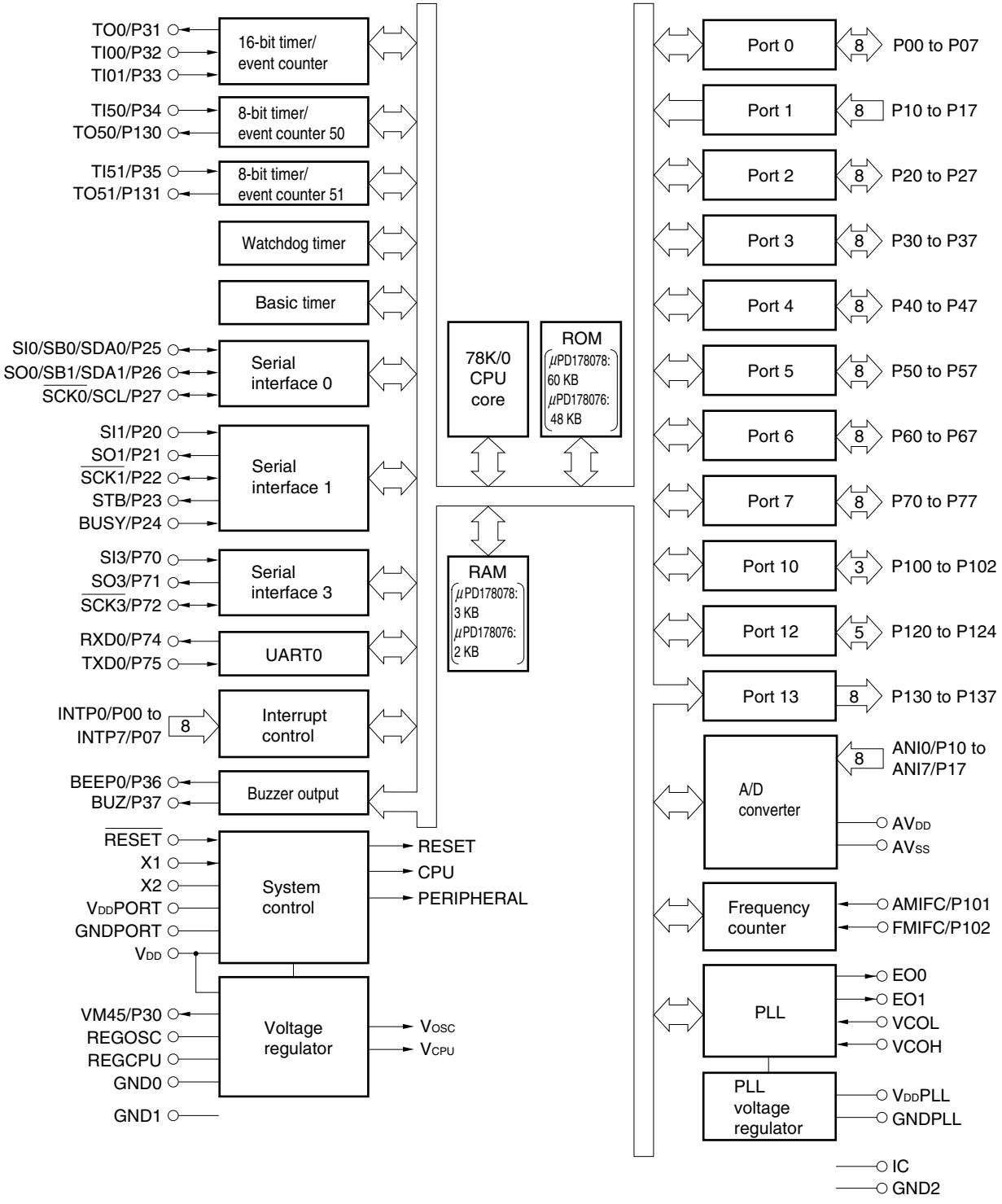
3. μPD178F098 only

★ 1.5 Development of 8-Bit DTS Series

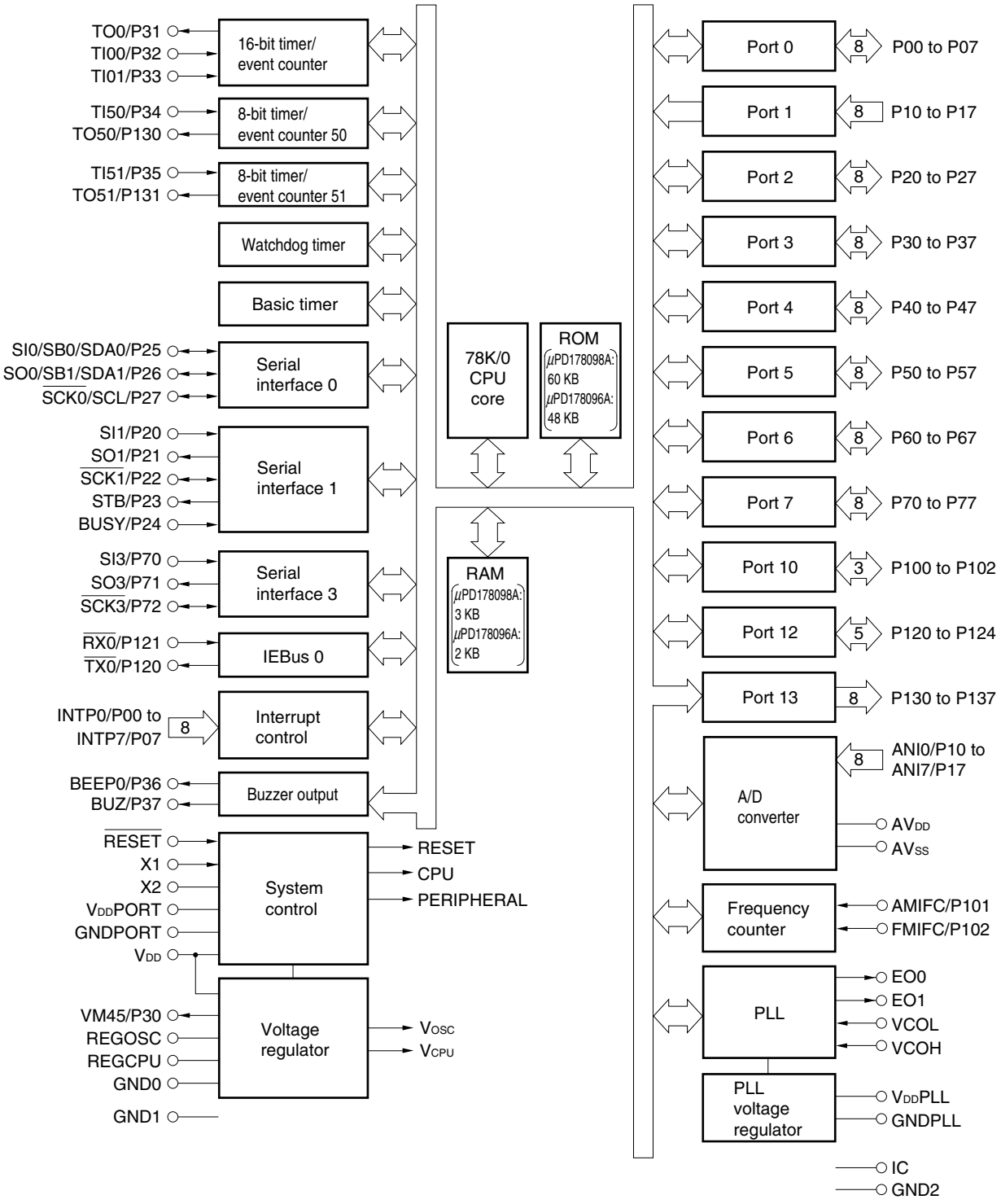


1.6 Block Diagram

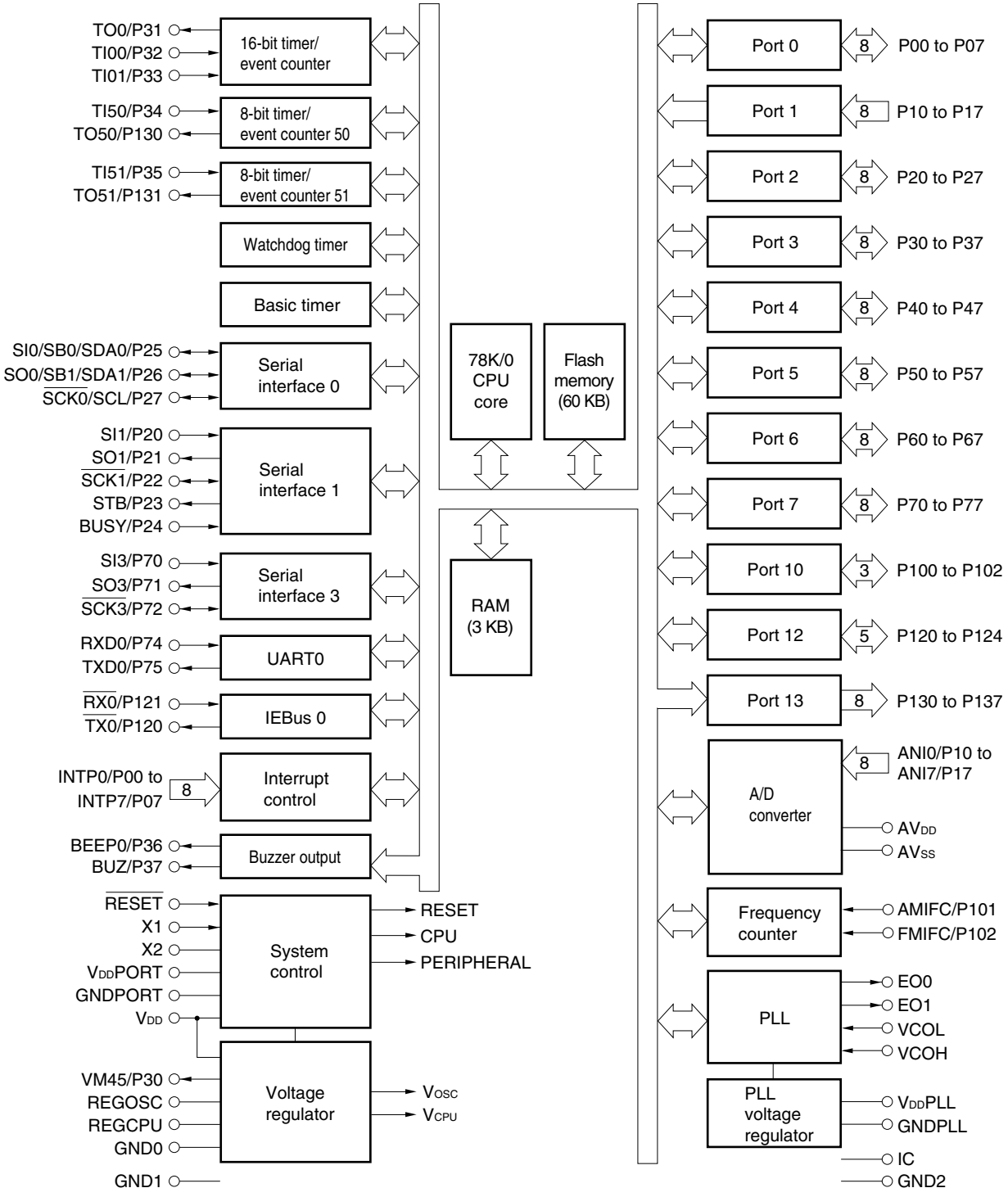
(1) μPD178076, 178078



(2) μPD178096A, 178098A



(3) μPD178F098



1.7 Functional Outline

(1/2)

Item		μPD178076	μPD178078	μPD178096A	μPD178098A	μPD178F098
Internal memory	ROM (Mask ROM)	48 KB (Mask ROM)	60 KB (Mask ROM)	48 KB (Mask ROM)	60 KB (Flash memory)	60 KB
	High-speed RAM	1024 bytes				
	Buffer RAM	32 bytes				
	Expansion RAM	1024 bytes	2048 bytes	1024 bytes	2048 bytes	
General-purpose registers		8 bits × 32 registers (8 bits × 8 registers × 4 banks)				
Minimum instruction execution time		• 0.317 μs/0.635 μs/1.27 μs/2.54 μs/5.08 μs (with crystal resonator of fx = 6.3 MHz) • 0.444 μs/0.889 μs/1.778 μs/3.556 μs/7.111 μs (with crystal resonator of fx = 4.5 MHz)**Note 1**				
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.				
I/O ports		Total: 80 pins • CMOS input: 8 pins • CMOS I/O: 64 pins • N-ch open-drain output: 8 pins				
A/D converter		8-bit resolution × 8 channels				
Serial interface		• 3-wire/SBI/2-wire/I²C bus**Note 2** mode selectable: 1 channel • 3-wire mode: 1 channel • 3-wire mode (with automatic transmit/receive function of up to 32 bytes): 1 channel • UART mode: 1 channel		• 3-wire/SBI/2-wire/I²C bus**Note 2** mode selectable: 1 channel • 3-wire mode: 1 channel • 3-wire mode (with automatic transmit/receive function of up to 32 bytes): 1 channel		• 3-wire/SBI/2-wire/I²C bus**Note 2** mode selectable: 1 channel • 3-wire mode: 1 channel • 3-wire mode (with automatic transmit/receive function of up to 32 bytes): 1 channel • UART mode: 1 channel
IEBus controller		Not provided		Provided		
Timer		• Basic timer (timer carry FF (10 Hz)): 1 channel • 16-bit timer/event counter: 1 channel • 8-bit timer/event counter: 2 channels • Watchdog timer: 1 channel				
Buzzer output		BEEP0 pin: 1 kHz, 1.5 kHz, 3 kHz, 4 kHz				
		BUZ pin: 0.77 kHz, 1.54 kHz, 3.08 kHz, 6.15 kHz (with crystal resonator of fx = 6.3 MHz)				

Notes 1. When using the IEBus controller of the μPD178096A, 178098A or 178F098, the 4.5 MHz crystal resonator cannot be used. Use the 6.3 MHz crystal resonator.

2. When the I²C bus mode is used (including when the mode is implemented in software without using the peripheral hardware), consult NEC Electronics when ordering a mask.

(2/2)

Item		μPD178076	μPD178078	μPD178096A	μPD178098A	μPD178F098
Vectored interrupt sources	Maskable	Internal: 13 External: 8		Internal: 12 External: 8		Internal: 15 External: 8
	Non-maskable	Internal: 1				
	Software	1				
PLL frequency synthesizer	Division mode	2 types • Direct division mode (VCOL pin) • Pulse swallow mode (VCOL and VCOH pins)				
	Reference frequency	Seven types selectable by software (1, 3, 9, 10, 12.5, 25, 50 kHz)				
	Charge pump	Error out output: 2 pins				
	Phase comparator	Unlock detectable in software				
Frequency counter		Frequency measurement • AMIFC pin: For 450 kHz counting • FMIFC pin: For 450 kHz/10.7 MHz counting				
Standby function		• HALT mode • STOP mode				
Reset		• Reset by $\overline{\text{RESET}}$ pin • Internal reset by watchdog timer • Reset by power-on clear circuit • Detection of less than 4.5 V[Note] (reset does not occur) • Detection of less than 3.5 V[Note] (during CPU operation) • Detection of less than 2.3 V[Note] (in STOP mode)				
Supply voltage		• V_{DD} = 4.5 to 5.5 V (during CPU, PLL operation) • V_{DD} = 3.5 to 5.5 V (during CPU operation)				
Package		• 100-pin plastic QFP (14 × 20 mm, 0.65 mm pitch)				

Note For details, refer to **CHAPTER 22 RESET FUNCTION**.

2.1 Pin Function List

(1) Port pins (1/2)

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P07	I/O	Port 0. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	INTP0 to INTP7
P10 to P17	Input	Port 1. 8-bit input port.	Input	ANI0 to ANI7
P20	I/O	Port 2. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	SI1
P21				SO1
P22				$\overline{\text{SCK1}}$
P23				STB
P24				BUSY
P25				SI0/SB0/SDA0
P26				SO0/SB1/SDA1
P27				$\overline{\text{SCK0}}$/SCL
P30	I/O	Port 3. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	VM45
P31				TO0
P32				TI00
P33				TI01
P34				TI50
P35				TI51
P36				BEEP0
P37				BUZ
P40 to 47	I/O	Port 4. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	–
P50 to P57	I/O	Port 5. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	–
P60 to P67	I/O	Port 6. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	–
P70	I/O	Port 7. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	SI3
P71				SO3
P72				$\overline{\text{SCK3}}$
P73				–
P74				RXD0**Note**
P75				TXD0**Note**
P76, P77				–

Note μPD178076, 178078, and 178F098 only.

(1) Port pins (2/2)

Pin Name	I/O	Function	After Reset	Alternate Function
P100	I/O	Port 10.	Input	–
P101		3-bit I/O port.		AMIFC
P102		Can be set to input or output mode in 1-bit units.		FMIFC
P120	I/O	Port 12.	Input	$\overline{TX0}$**Note**
P121		5-bit I/O port.		$\overline{RX0}$**Note**
P122 to P124		Can be set to input or output mode in 1-bit units.		–
P130	Output	Port 13.	Low level output	TO50
P131		8-bit output port.		TO51
P132 to P137		N-ch open-drain output port (12 V tolerant)		

Note μPD178096A, 178098A, and 178F098 only.

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

Pin Name	I/O	Function		After Reset	Alternate Function
INTP0 to INTP7	Input	External maskable interrupt input whose valid edge (rising edge, falling edge, or both rising and falling edges) can be specified.		Input	P00 to P07
SI0	Input	Serial data input to serial interface.		Input	P25/SB0/SDA0
SI1					P20
SI3					P70
SO0	Output	Serial data output from serial interface.		Input	P26/SB1/SDA1
SO1					P21
SO3					P71
SB0	I/O	Serial data input/output to/from serial interface.	N-ch open-drain I/O	Input	P25/SI0/SDA0
SB1					P26/SO0/SDA1
SDA0					P25/SI0/SB0
SDA1					P26/SO0/SB1
$\overline{SCK0}$	I/O	Serial clock input/output to/from serial interface.		Input	P27/SCL
$\overline{SCK1}$					P22
$\overline{SCK3}$					P72
SCL			N-ch open-drain I/O		P27/$\overline{SCK0}$
STB	Output	Strobe output for serial interface automatic transmission/reception.		Input	P23
BUSY	Input	Busy input for serial interface automatic transmission/reception.		Input	P24
VW45	Output	V_{DD} = 4.5 V monitor output		Input	P30
TI00	Input	External count clock input to 16-bit timer (TM0).		Input	P32
TI01					P33
TI50	Input	External count clock input to 8-bit timer (TM50).		Input	P34
TI51		External count clock input to 8-bit timer (TM51).			P35

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

Pin Name	I/O	Function	After Reset	Alternate Function
TO0	Output	16-bit timer (TM0) output.	Input	P31
TO50		8-bit timer (TM50) output.	Low level output	P130
TO51		8-bit timer (TM51) output.		P131
BEEP0	Output	Buzzer output.	Input	P36
BUZ				P37
ANI0 to ANI7	Input	Analog input to A/D converter.	Input	P10 to P17
EO0, EO1	Output	Error out output from charge pump of PLL frequency synthesizer.	–	–
VCOL	Input	Inputs local oscillation frequency of PLL (in HF and MF modes).	–	–
VCOH	Input	Inputs local oscillation frequency of PLL (in VHF mode).	–	–
AMIFC	Input	Input to AM intermediate frequency counter.	Input	P101
FMIFC	Input	Input to FM intermediate frequency or AM intermediate frequency counter.	Input	P102
RXD0**Note 1**	Input	Serial data input to asynchronous serial interface (UART0).	Input	P74
TXD0**Note 1**	Output	Serial data output from asynchronous serial interface (UART0).	Input	P75
$\overline{\text{TX0}}$**Note 2**	Output	IEBus controller data output.	Input	P120
$\overline{\text{RX0}}$**Note 2**	Input	IEBus controller data input.	Input	P121
$\overline{\text{RESET}}$	Input	System reset input.	–	–
X1	Input	Connection of crystal resonator for system clock oscillation.	–	–
X2	–		–	–
REGOSC	–	Regulator for oscillator. Connect this pin to GND via a 0.1 μF capacitor.	–	–
REGCPU	–	Regulator for CPU power supply. Connect this pin to GND via a 0.1 μF capacitor.	–	–
V_{DD}	–	Positive power supply.	–	–
GND0 to GND2	–	Ground.	–	–
V_{DD}PORT	–	Port power supply. Make the same potential as V_{DD}.	–	–
GNDPORT	–	Port ground. Make the same potential as GND0 to GND2.	–	–
AV_{DD}	–	A/D converter positive power supply. Make the same potential as V_{DD}.	–	–
AV_{SS}	–	A/D converter ground. Make the same potential as GND0 to GND2.	–	–
V_{DD}PLL**Note 4**	–	PLL positive power supply. Make the same potential as V_{DD}.	–	–
GNDPLL**Note 4**	–	PLL ground. Make the same potential as GND0 to GND2.	–	–
IC	–	Internally connected. Directly connect this pin to GND0, GND1, or GND2.	–	–
V_{PP}**Note 3**	–	Pin to apply high voltage at program write/verify.	–	–

Notes 1. μPD178076, 178078, and 178F098 only.

2. μPD178096A, 178098A, and 178F098 only.

3. μPD178F098 only.

4. Connect a capacitor of about 1000 pF between the V_{DD}PLL and GNDPLL pins.

2.2 Description of Pin Functions

2.2.1 P00 to P07 (Port 0)

These are 8-bit I/O port pins. Besides serving as I/O port pins, they also function as external interrupt inputs. The following operating modes can be specified in 1-bit units.

(1) Port mode

These pins function as an 8-bit I/O port that can be set to input or output mode in 1-bit units using port mode register 0.

(2) Control mode

These pins function as external interrupt input pins (INTP0 to INTP7).

INTP0 to INTP7 are external interrupt input pins for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified.

2.2.2 P10 to P17 (Port 1)

These are 8-bit input port pins. Besides serving as input port pins, they also function as A/D converter analog inputs.

The following operating modes can be specified in 1-bit units.

(1) Port mode

These pins function as an 8-bit input port.

(2) Control mode

These pins function as A/D converter analog input pins (ANI0 to ANI7).

2.2.3 P20 to P27 (Port 2)

These are 8-bit I/O port pins. Besides serving as I/O port pins, they also have serial interface data input/output, clock I/O, automatic transmit/receive busy input, and strobe output functions.

The following operating modes can be specified in 1-bit units.

(1) Port mode

These pins function as an 8-bit I/O port that can be set to input or output mode in 1-bit units using port mode register 2.

(2) Control mode

These pins function as serial interface data I/O, clock I/O, automatic transmit/receive busy input, and strobe output pins.

(a) SI0, SI1, SO0, SO1, SDA0[Note], SDA1[Note]

These are serial interface serial data I/O pins. SDA0 and SDA1 are N-ch open drain.

(b) $\overline{\text{SCK0}}$ and $\overline{\text{SCK1}}$, SCL[Note]

These are serial interface serial clock I/O pins.

SCL is N-ch open drain.

Note For I^2C bus mode

(c) SB0 and SB1

These are NEC standard serial bus interface I/O pins.

SB0 and SB1 are N-ch open drain.

(d) BUSY

This is the serial interface automatic transmit/receive busy input pin.

(e) STB

This is the serial interface automatic transmit/receive strobe output pin.

Caution When this port is used for the serial interface, the I/O and output latches must be set according to the required function. For the setting, refer to Table 4-3 Port Mode Register and Output Latch Settings When Using Alternate Functions and Figure 13-5 Format of Serial Operating Mode Register 0 (CSIM0).

2.2.4 P30 to P37 (Port 3)

These are 8-bit I/O port pins. Beside serving as I/O port pins, they also have V_{DD} = 4.5 V monitor output, timer output, timer input and buzzer output functions.

The following operating modes can be specified in 1-bit units.

(1) Port mode

These pins function as an 8-bit I/O port that can be set to input or output mode in 1-bit units using port mode register 3.

(2) Control mode

These pins function as V_{DD} = 4.5 V monitor output, timer output, timer input, and buzzer (BEEP0, BUZ) output pins.

(a) VM45

This is the monitor output pin of V_{DD} = 4.5 V.

(b) TO0

This is the output pin of the 16-bit timer/event counter.

(c) TI00

This is the external count clock input pin of the 16-bit timer/event counter and a capture trigger signal input pin.

(d) TI01

This is a capture trigger signal input pin of the 16-bit timer/event counter.

(e) TI50, TI51

These are pins for external clock input to the 8-bit timer/event counter.

(f) BEEP0, BUZ

These are buzzer output pins.

2.2.5 P40 to P47 (Port 4)

These are 8-bit I/O port pins.

They can be set to input or output mode in 8-bit units mode using port mode register 4.

2.2.6 P50 to P57 (Port 5)

These are 8-bit I/O port pins.

They can be set to input or output mode in 1-bit units using port mode register 5.

2.2.7 P60 to P67 (Port 6)

These are 8-bit I/O port pins.

They can be set to input or output mode in 1-bit units using port mode register 6.

2.2.8 P70 to P77 (Port 7)

These are 8-bit I/O port pins. Besides serving as I/O port pins, they also have serial interface data I/O, clock I/O, and asynchronous serial interface data I/O functions.

The following operating modes can be specified in 1-bit units.

(1) Port mode

These pins function as an 8-bit I/O port that can be set to input or output mode in 1-bit units using port mode register 7.

(2) Control mode

These pins function as serial interface data I/O, clock I/O, and asynchronous serial interface data I/O pins.

(a) SI3 and SO3

These are serial data I/O pins of the serial interface.

(b) $\overline{\text{SCK3}}$

This is a serial clock I/O pin of the serial interface.

(c) RXD0, TXD0 (μPD178076, 178078, and 178F098 only)

These are serial data I/O pins of the asynchronous serial interface.

2.2.9 P100 to P102 (Port 10)

These are 3-bit I/O port pins. Besides serving as I/O port pins, they can also be used as the input pins of an AM intermediate frequency counter and an FM intermediate frequency counter.

The following operating modes can be specified in 1-bit units.

(1) Port mode

These pins function as a 3-bit I/O port that can be set to input or output mode in 1-bit units using port mode register 10.

(2) Control mode

These pins can be used as the input pins of the AM intermediate frequency counter and FM intermediate frequency counter.

(a) AMIFC

This is an input pin of the AM intermediate frequency counter.

(b) FMIFC

This is an input pin of the FM intermediate frequency counter or AM intermediate frequency counter.

2.2.10 P120 to P124 (Port 12)

These pins are 5-bit I/O port pins that can also be used to input IEBus data.

They can be set to input or output mode in 1-bit units using port mode register 12.

The following operating modes can be specified in 1-bit units.

(1) Port mode

These pins function as a 5-bit I/O port that can be set to input or output mode in 1-bit units using port mode register 12.

(2) Control mode (μPD178096A, 178098A, and 178F098 only)

These pins can be used as the IEBus controller data I/O ($\overline{RX0}$ and $\overline{TX0}$).

2.2.11 P130 to P137 (Port 13)

These are 8-bit output port pins. They are N-ch open-drain pins with a 12 V withstanding voltage. Besides serving as output port pins, they are used for timer output.

The following operating modes can be specified in 1-bit units.

(1) Port mode

These pins function as an 8-bit output port.

(2) Control mode

These pins function as outputs for the 8-bit timer/event counter (TO50, TO51).

2.2.12 EO0, EO1

These are the output pins of the charge pump of the PLL frequency synthesizer.

They output the result of phase comparison between the frequency divided by the programmable divider of the local oscillation input (VCOL and VCOH pins) and the reference frequency.

2.2.13 VCOL, VCOH

These pins input the local oscillation frequency (VCO) of the PLL.

Because signals are input to these pins via an AC amplifier, cut the DC component of the input signals by using a capacitor.

- VCOL
  - HF, MF input
  - This pin becomes active when the HF or MF mode is selected by software. Otherwise, the pin is in the status set by bit 2 (VCOLDMD) of the PLL mode select register (PLLMD). If VCOLDMD is reset to 0 (to connect a pull-down resistor), however, the VCOL pin does not become active even if the HF or MF mode is selected. In this case, set VCOLDMD to 1 (high-impedance state).
- VCOH
  - VHF input
  - This pin becomes active when the FM mode is selected by software. Otherwise, the pin is in the status set by bit 3 (VCOHDMD) of the PLL mode select register (PLLMD). If VCOHDMD is reset to 0 (to connect a pull-down resistor), however, the VCOH pin does not become active even if the FM mode is selected. In this case, set VCOHDMD to 1 (high-impedance state).

2.2.14 $\overline{RESET}$

This is a low-level active system reset input pin.

2.2.15 X1, X2

These are crystal resonator connection pins for system clock oscillation.

2.2.16 REGOSC

This is the oscillator regulator pin. Connect to GND via a 0.1 μF capacitor.

2.2.17 REGCPU

This is the CPU power supply regulator pin. Connect to GND via a 0.1 μF capacitor.

2.2.18 V_{DD}

This is the positive power supply pin.

2.2.19 GND0 to GND2

These are ground potential pins.

2.2.20 V_{DD}PORT

This is the positive power supply pin for ports.

2.2.21 GNDPORT

This is the ground potential pin for ports.

2.2.22 V_{DD}PLL

This is the positive power supply pin for the PLL.

2.2.23 GNDPLL

This is the ground potential pin for the PLL.

2.2.24 AV_{DD}

This is the analog power supply pin of the A/D converter.

Always keep the voltage on this pin at the same level as that on the V_{DD} pin even when the A/D converter is not used.

2.2.25 AV_{SS}

This is the ground pin of the A/D converter.

Always keep the voltage on this pin at the same level as that on the GND0, GND1, or GND2 pin even when the A/D converter is not used.

2.2.26 V_{PP} (μPD178F098 only)

This pin applies a high voltage when the flash memory programming mode is set or when a program is written or verified.

★ Connect this pin in either of the following ways.

- Connect independently to a 10 kΩ pull-down resistor.
- By using a jumper on the board, connect directly to the dedicated flash programmer in the programming mode or to GND in the normal operation mode.

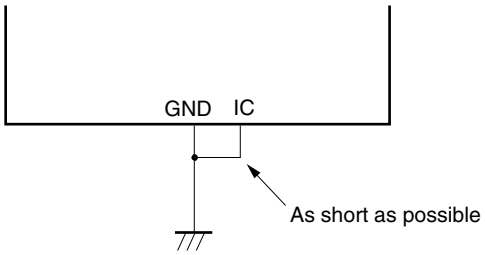
When the wiring between the V_{PP} pin and GND pin is long or external noise is input to the V_{PP} pin, the user's program may not run normally.

2.2.27 IC (Mask ROM versions only)

The IC (Internally Connected) pin is provided to set the test mode to check the μPD178078 and 178098A Subseries at delivery. Connect it directly to the GND pin with the shortest possible wire in the normal operating mode.

When a potential difference is produced between the IC pin and GND pin because the wiring between the two pins is too long or external noise is input to the IC pin, the user's program may not run normally.

○ **Connect IC pin to GND pin directly.**



2.3 Pin I/O Circuits and Recommended Connection of Unused Pins

Table 2-1 shows the pin I/O circuit types and the recommended connection of the pins when they are not used. For the configuration of the I/O circuit of each pin, refer to Figure 2-1.

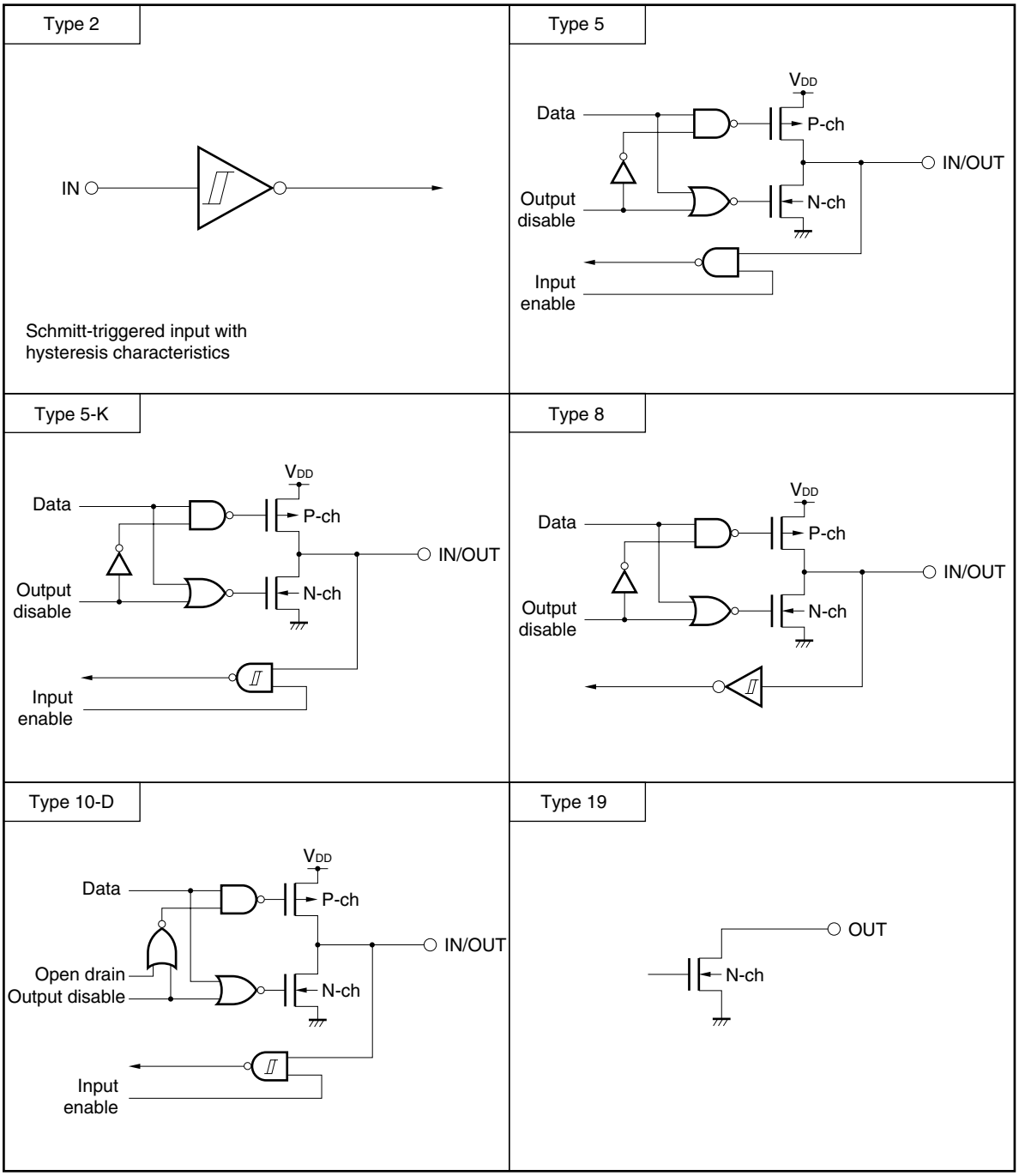
★ **Table 2-1. Pin I/O Circuit Types (1/2)**

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pin
P00/INTP0 to P07/INTP7	8	I/O	Input: Independently connect to V_{DD}, V_{DD}PORT, GND0 to GND2, or GNDPORT via a resistor. Output: Leave open.
P10/ANI0 to P17/ANI7	25	Input	Directly connect to V_{DD}, V_{DD}PORT, GND0 to GND2, or GNDPORT.
P20/SI1	5-K	I/O	Input: Independently connect to V_{DD}, V_{DD}PORT, GND0 to GND2, or GNDPORT via a resistor. Output: Leave open.
P21/SO1	5		
P22/$\overline{\text{SCK1}}$	5-K		
P23/STB	5		
P24/BUSY	5-K		
P25/SI0/SB0/SDA0	10-D		
P26/SO0/SB1/SDA1			
P27/$\overline{\text{SCK0}}$/SCL			
P30/VM45	5		
P31/TO0			
P32/TI00	5-K		
P33/TI01			
P34/TI50			
P35/TI51			
P36/BEEP0	5		
P37/BUZ			
P40 to P47			
P50 to P57			
P60 to P67			
P70/SI3	5-K		
P71/SO3	5		
P72/$\overline{\text{SCK3}}$	5-K		
P73	5		
P74/RXD0	5-K		
P75/TXD0	5		
P76, P77			
P100			
P101/AMIFC			
P102/FMIFC			
P120/$\overline{\text{TX0}}$			
P121/$\overline{\text{RX0}}$	5-K		
P122 to P124	5		

★ **Table 2-1. Pin I/O Circuit Types (2/2)**

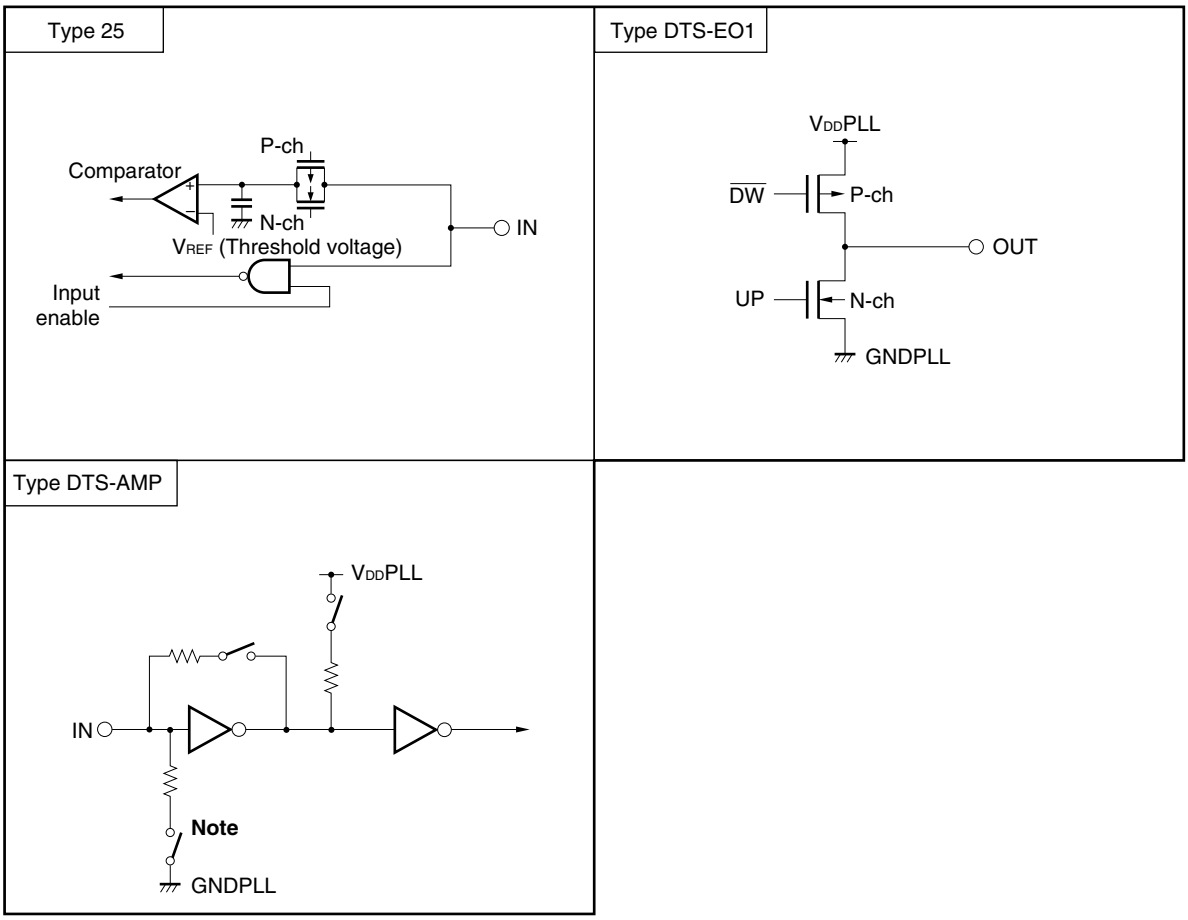
Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pin
P130/TO50	19	Output	Leave open.
P131/TO51			
P132 to P137			
EO0	DTS-EO1		
EO1			
VCOL, VCOH	DTS-AMP2	Input	Disable PLL by software and select pull-down.
REGOSC, REGCPU	–	–	Connect to GND0, GND1, or GND2 via 0.1 μF capacitor.
$\overline{\text{RESET}}$	2	Input	–
AV_{DD}	–	–	Connect to V_{DD} or V_{DD}PORT.
AV_{SS}			Directly connect to GND0 to GND2, or GNDPORT.
IC (mask ROM version)			
V_{PP} (flash memory version)			Independently connect a 10 kΩ pull-down resistor or directly connect to GND0, GND1, GND2, or GNDPORT.

★ **Figure 2-1. Pin I/O Circuits (1/2)**



Remark V_{DD} and GND are the positive power supply and ground pins for all port pins. Read V_{DD} and GND as V_{DD}PORT and GNDPORT.

★ **Figure 2-1. Pin I/O Circuits (2/2)**



Note This switch is selectable by software only for the VCOL and VCOH pins.

Remark V_{DD} and GND are the positive power supply and ground pins for all port pins. Read V_{DD} and GND as V_{DD}PORT and GNDPORT.

CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Spaces

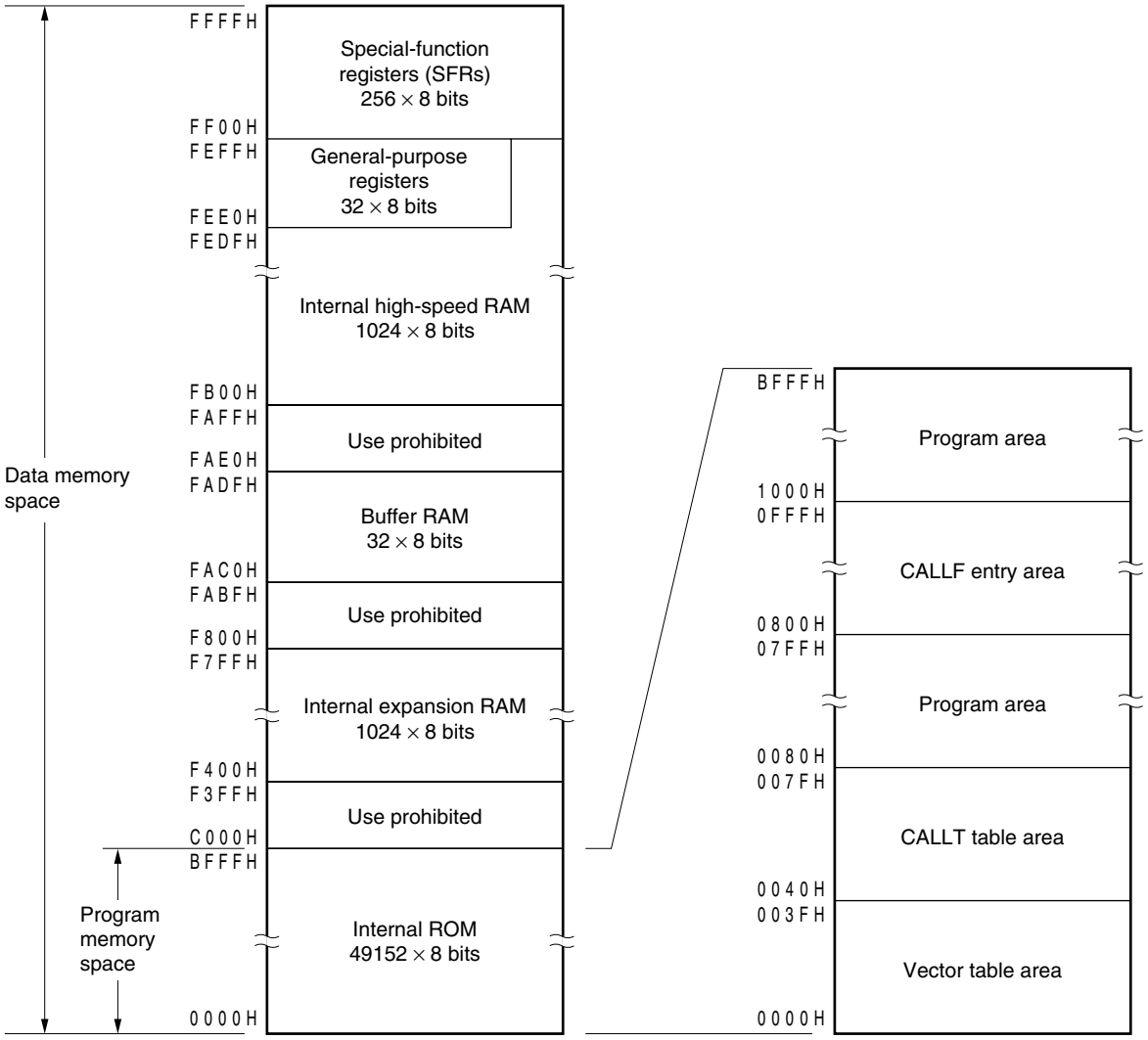
The initial values of the memory size select register (IMS) and internal expansion RAM size select register (IXS) are CFH and 0CH, respectively. The following values must be set to the registers of each product.

Part Number	IMS	IXS
μPD178076, 178096A	CCH	0AH
μPD178078, 178098A	CFH	08H
μPD178F098	Value equivalent to mask ROM version	Value equivalent to mask ROM version

(1) μPD178076, 178096A

Set the values of the memory size select register (IMS) and internal expansion RAM size select register (IXS) to CCH and 0AH, respectively (the initial values are CFH and 0CH).

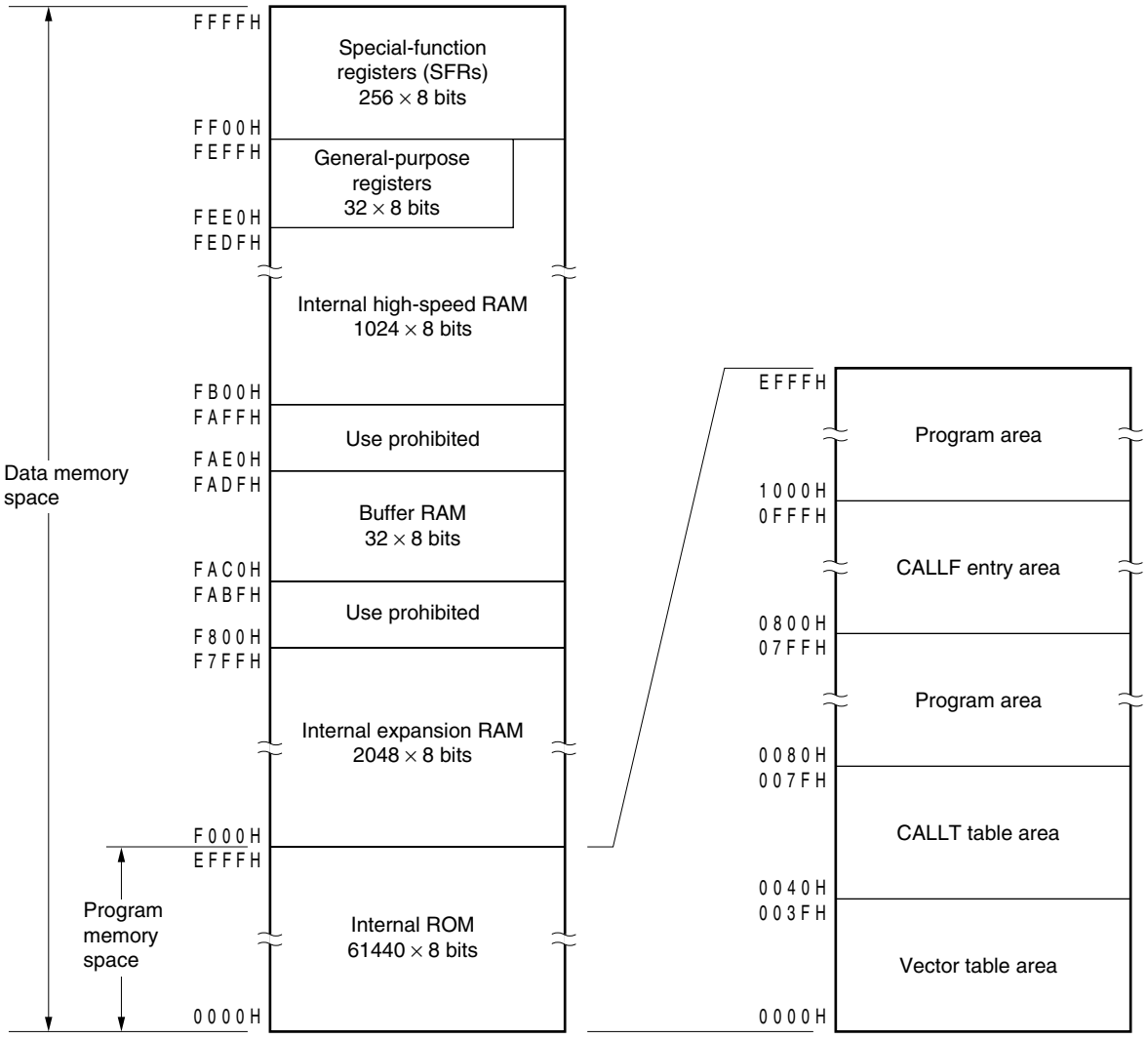
Figure 3-1. Memory Map (μPD178076, 178096A)



(2) μPD178078, 178098A

Set the values of the memory size select register (IMS) and internal expansion RAM size select register (IXS) to CFH and 08H, respectively (the initial values are CFH and 0CH).

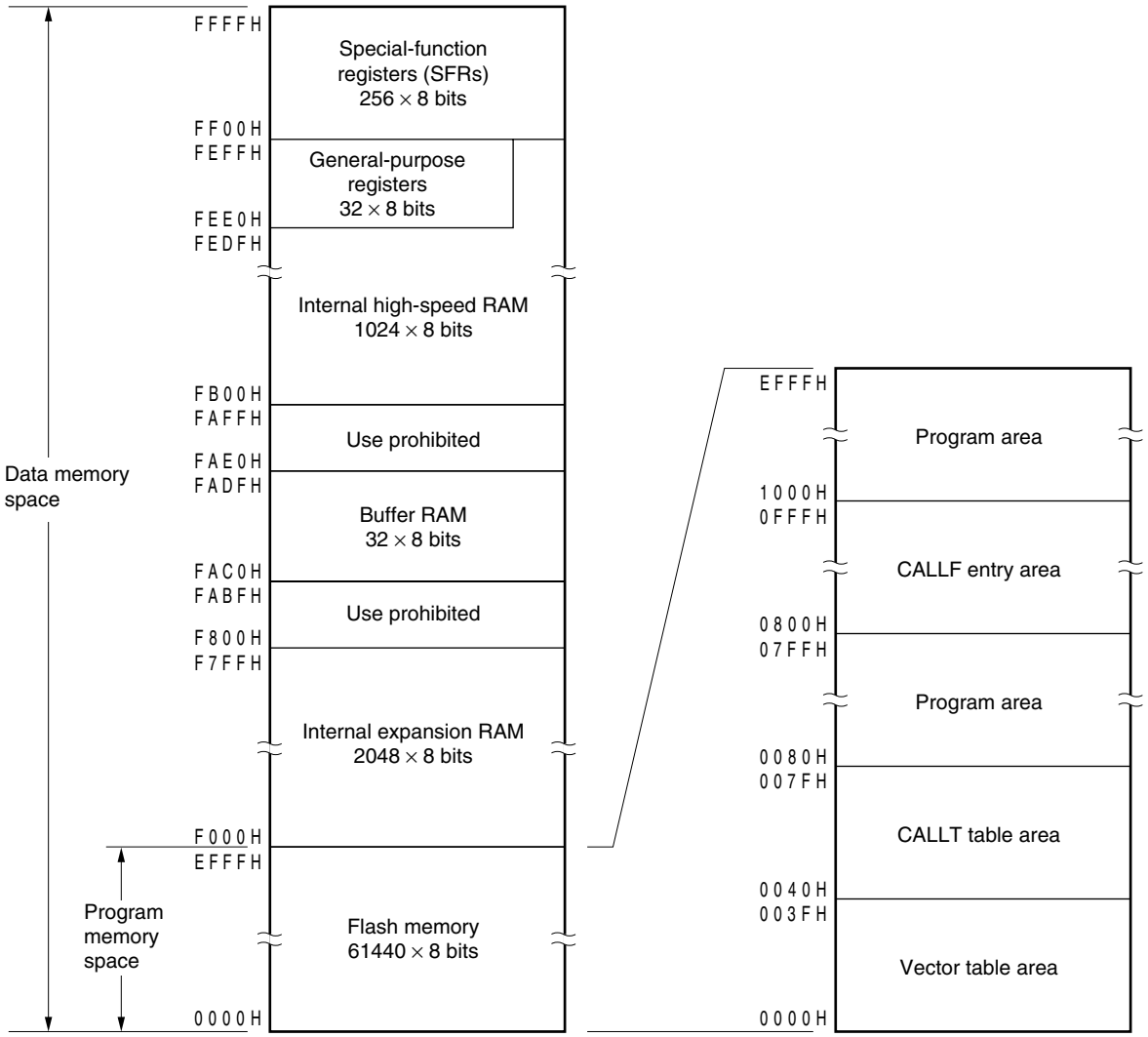
Figure 3-2. Memory Map (μPD178078, 178098A)



(3) μPD178F098

Set the value of the memory size select register (IMS) and internal expansion RAM size select register (IXS) to the value corresponding to that of the mask ROM versions (the initial values are CFH and 0CH).

Figure 3-3. Memory Map (μPD178F098)



3.1.1 Internal program memory space

The internal program memory space stores programs and table data, and is usually addressed by the program counter (PC).

The μPD178078, 178098A Subseries products incorporate the following internal ROM (or flash memory).

Table 3-1. Internal Memory Capacity

Product	Structure	Capacity
μPD178076, 178078	Mask ROM	49152 × 8 bits (0000H to BFFFH)
μPD178096A, 178098A		61440 × 8 bits (0000H to EFFFH)
μPD178F098	Flash memory	

(1) Vector table area

The 64-byte area 0000H to 003FH is reserved as a vector table area. The reset input and program start addresses for branch upon generation of an interrupt request are stored in the vector table area. Of the 16-bit address, the lower 8 bits are stored at even addresses and the higher 8 bits are stored at odd addresses.

Table 3-2. Vector Table

Vector Table Address	Interrupt Request	Vector Table Address	Interrupt Request
0000H	Reset input	001CH	INTTM50
0004H	INTWDT	001EH	INTTM51
0006H	INTP0	0020H	INTSER0Note 1
0008H	INTP1	0022H	INTSR0Note 1
000AH	INTP2	0024H	INTST0Note 1
000CH	INTP3	0026H	INTBTM0
000EH	INTP4	0028H	INTTM00
0010H	INTP5	002AH	INTTM01
0012H	INTP6	002CH	INTIE1Note 2
0014H	INTP7	002EH	INTIE2Note 2
0016H	INTCSI0	0030H	INTAD
0018H	INTCSI1	003EH	BRK
001AH	INTCSI3		

Notes 1. μPD178076, 178078, and 178F098 only.

2. μPD178096A, 178098A, and 178F098 only.

(2) CALLT instruction table area

The 64-byte area 0040H to 007FH can store the subroutine entry address of a 1-byte call instruction (CALLT).

(3) CALLF instruction entry area

The area 0800H to 0FFFH can perform a direct subroutine call with a 2-byte call instruction (CALLF).

3.1.2 Internal data memory space

The μPD178078 and 178098A Subseries products incorporate the following RAMs.

(1) Internal high-speed RAM

A high-speed memory of 1024 bytes is incorporated.

In this area, four banks of general-purpose registers, each bank consisting of eight 8-bit registers, are allocated in the 32-byte area FEE0H to FEFFH.

★ This area cannot be used as a program area to which instructions are written and executed.

The internal high-speed RAM can also be used as a stack memory area.

(2) Buffer RAM

Buffer RAM is allocated to the 32-byte area from FAC0H to FADFH. Buffer RAM is used to store transmit/receive data for serial interface channel 1 (3-wire serial I/O mode with automatic transmit/receive function). When not used in this mode, buffer RAM can also be used as normal RAM.

(3) Internal expansion RAM

Internal expansion RAM is allocated to the 1024-byte area from F400H to F7FFH in the μPD178076 and 178096A. For the μPD178078, 178098A, and 178F098, it is allocated to the 2048-byte area from F000H to F7FFH.

★ This area can be used as a normal data area like the internal high-speed RAM, and also as a program area to which instructions are written and executed.

The internal expansion RAM cannot be used as a stack memory.

3.1.3 Special-function register (SFR) area

On-chip peripheral hardware special-function registers (SFRs) are allocated in the area FF00H to FFFFH. (Refer to **Table 3-4 Special-Function Registers**.)

Caution Do not access addresses where an SFR is not assigned.

3.1.4 Data memory addressing

The method to specify the address of the instruction to be executed next, or the address of a register or memory to be manipulated when an instruction is executed is called addressing.

To address the memory that is manipulated when an instruction is executed, the μPD178078, 178098A Subseries products are provided with many addressing modes to enable high operability. Especially at addresses corresponding to data memory area, particular addressing modes can be used in accordance with the functions of the special-function registers (SFRs) and general-purpose registers. Figures 3-4 to 3-6 show the data memory addressing modes. For details of each addressing, see **3.4 Operand Address Addressing.**

Figure 3-4. Correspondence Between Data Memory and Addressing (μPD178076, 178096A)

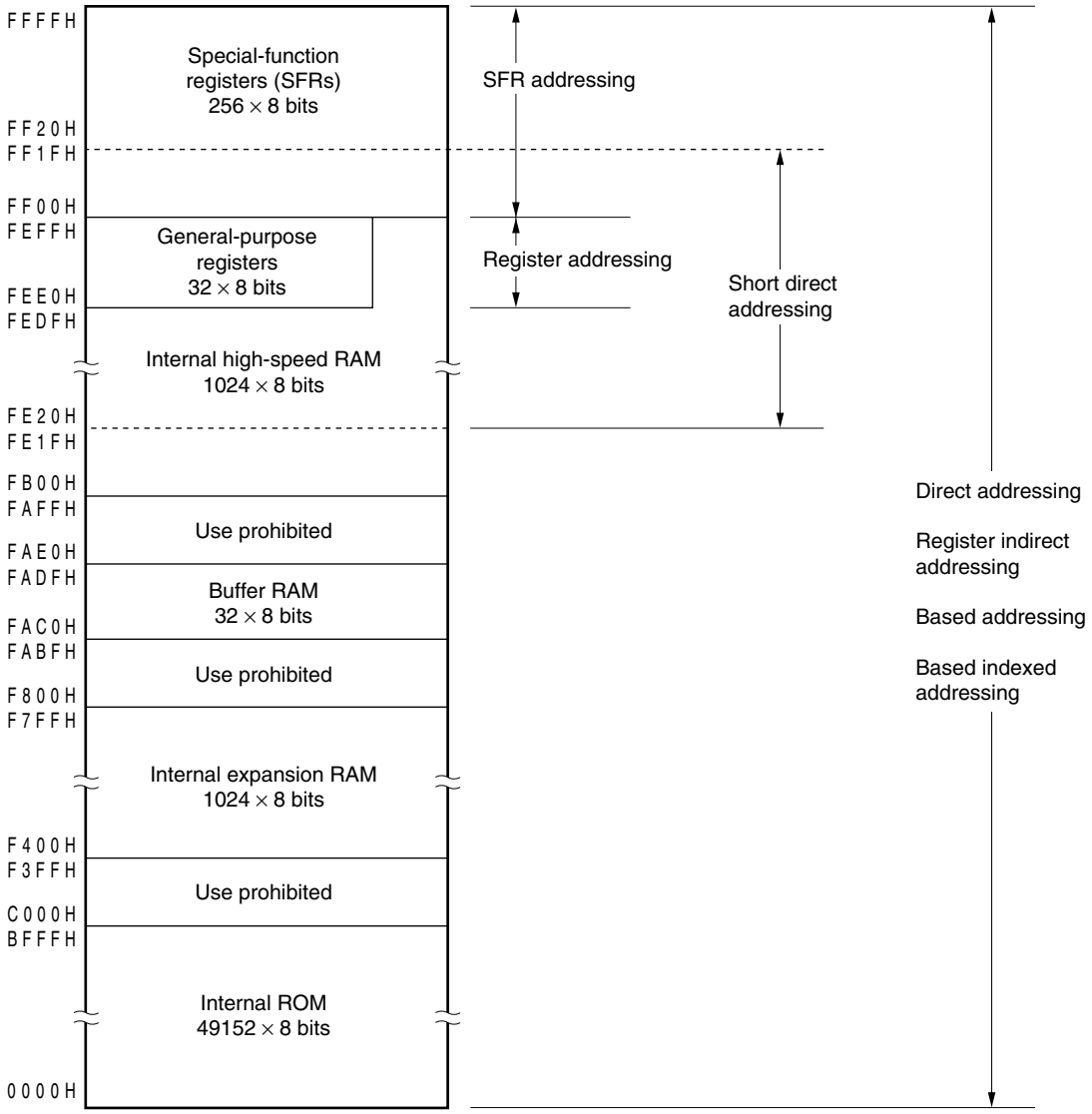


Figure 3-5. Correspondence Between Data Memory and Addressing (μPD178078, 178098A)

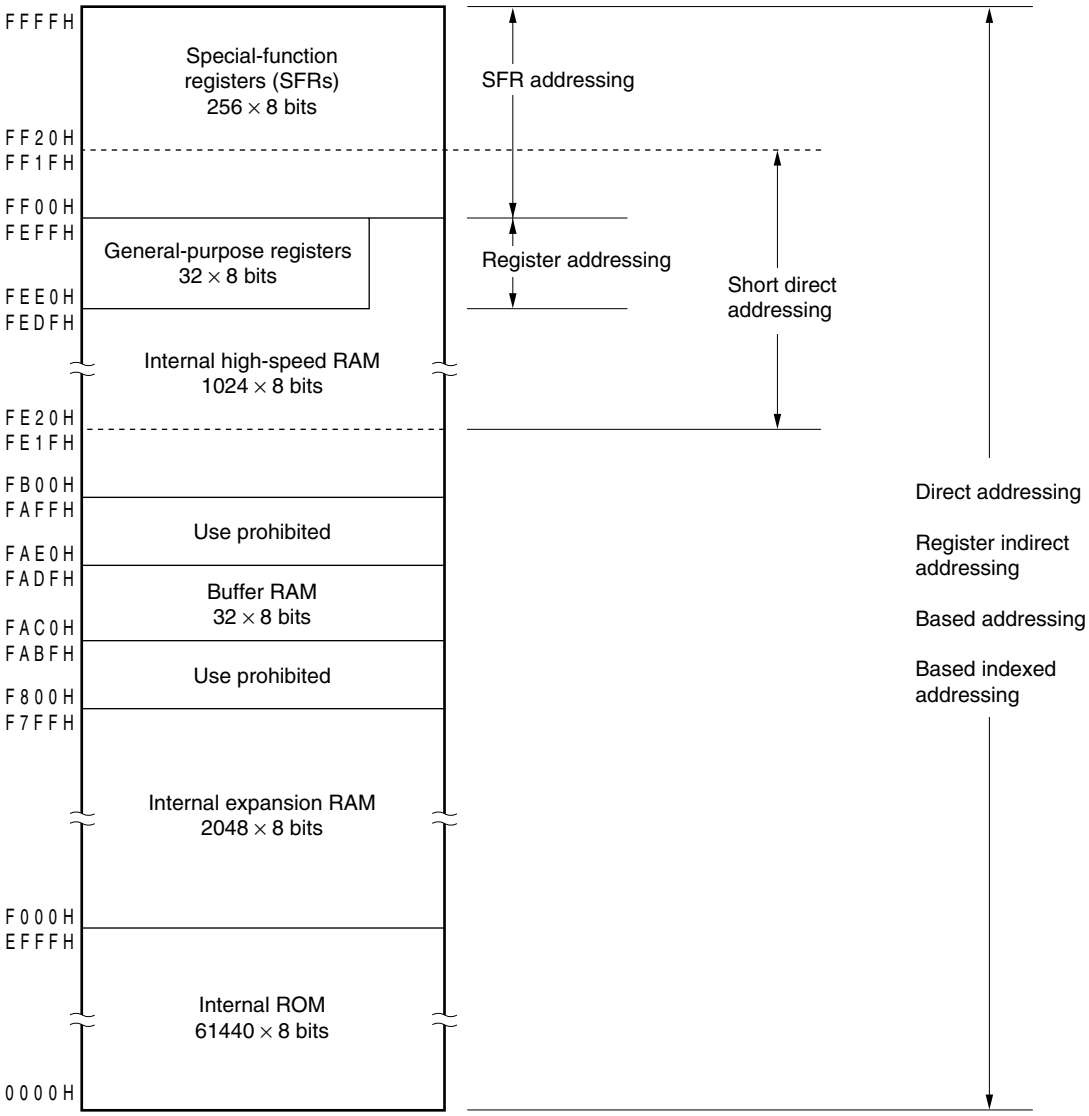
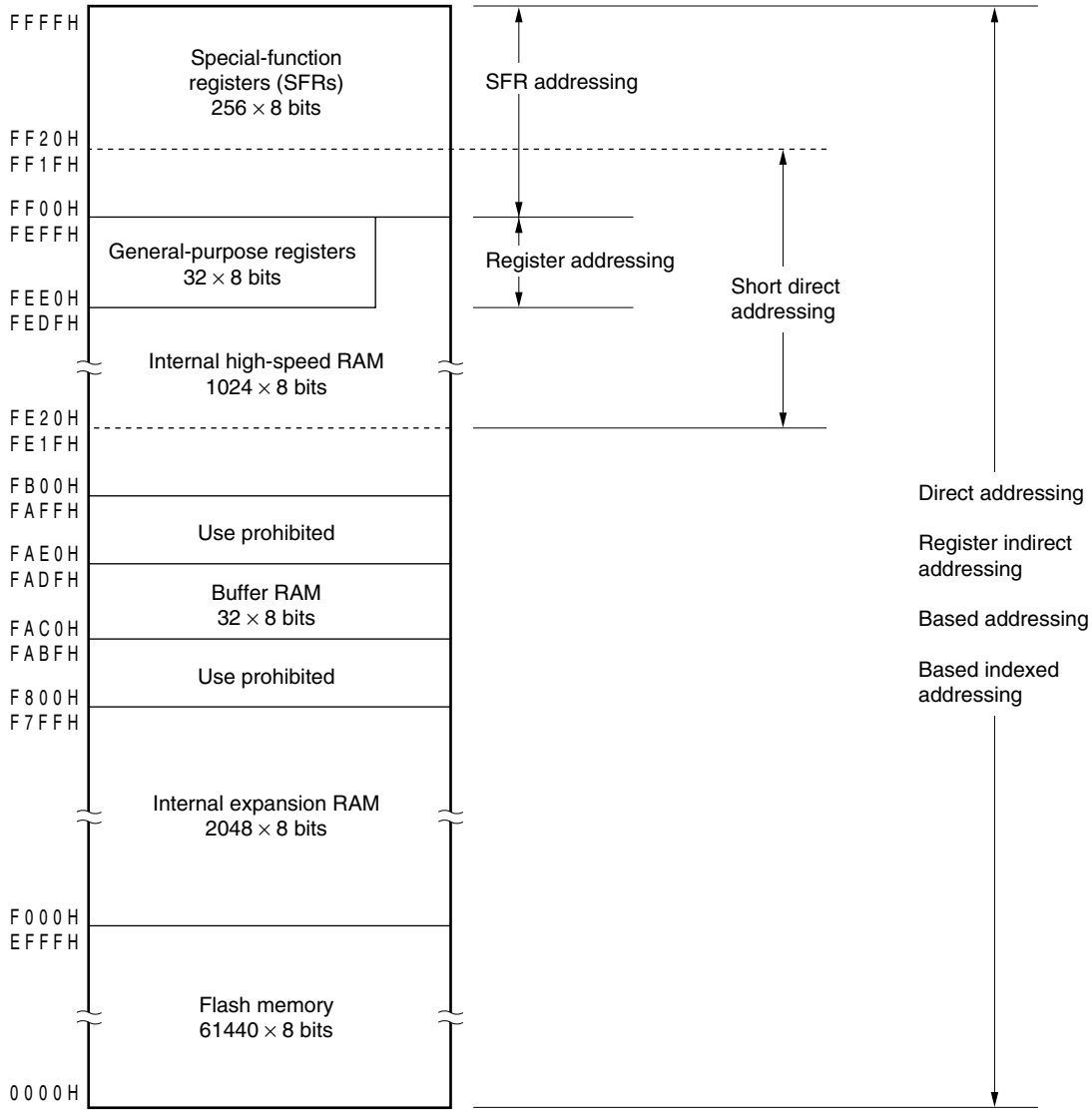


Figure 3-6. Correspondence Between Data Memory and Addressing (μPD178F098)



3.2 Processor Registers

The μPD178078 and 178098A Subseries products incorporate the following processor registers.

3.2.1 Control registers

The control registers control the program sequence, statuses and stack memory. The control registers consist of a program counter (PC), a program status word (PSW), and a stack pointer (SP).

(1) Program counter (PC)

The program counter is a 16-bit register that holds the address information of the next program to be executed. In normal operation, the PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data and register contents are set.

Reset input sets the reset vector table values at addresses 0000H and 0001H to the program counter.

Figure 3-7. Configuration of Program Counter

(2) Program status word (PSW)

	15															0
PC	PC15	PC14	PC13	PC12	PC11	PC10	PC9	PC8	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

The program status word is an 8-bit register consisting of various flags to be set/reset by instruction execution. The program status word contents are automatically stacked upon interrupt request generation or PUSH PSW instruction execution and are automatically restored upon execution of the RETB, RETI and POP PSW instructions.

Reset input sets the PSW to 02H.

Figure 3-8. Configuration of Program Status Word

	7							0
PSW	IE	Z	RBS1	AC	RBS0	0	ISP	CY

(a) Interrupt enable flag (IE)

This flag controls the interrupt request acknowledgment operations of the CPU.

When IE = 0, all the interrupts are disabled (DI) except the non-maskable interrupt.

When IE = 1, the interrupts are enabled (EI). At this time, the acknowledgment of interrupts is controlled by the in-service priority flag (ISP), the interrupt mask flag corresponding to each interrupt, and the interrupt priority specification flag.

The IE flag is reset to (0) upon DI instruction execution or interrupt acknowledgment and is set to (1) upon EI instruction execution.

(b) Zero flag (Z)

When the operation result is zero, this flag is set (1). It is reset (0) in all other cases.

(c) Register bank select flags (RBS0 and RBS1)

These are 2-bit flags to select one of the four register banks.

The 2-bit information that indicates the register bank selected by SEL RBn instruction execution is stored in these flags.

(d) Auxiliary carry flag (AC)

If the operation result has a carry from bit 3 or a borrow at bit 3, this flag is set (1). It is reset (0) in all other cases.

(e) In-service priority flag (ISP)

This flag manages the priority of acknowledgeable maskable vectored interrupts.

When ISP = 0, acknowledging the vectored interrupt requests to which a low priority is assigned by the priority specification flag registers (PR0L, PR0H, PR1L) (refer to **18.3 (3) Priority specification flag registers (PR0L, PR0H, PR1L)**) is disabled. Whether an interrupt request is actually acknowledged depends on the status of the interrupt enable flag (IE).

(f) Carry flag (CY)

This flag stores an overflow and underflow upon add/subtract instruction execution. It stores the shift-out value upon rotate instruction execution and functions as a bit accumulator during bit manipulation instruction execution.

(3) Stack pointer (SP)

This is a 16-bit register that holds the start address of the memory stack area. Only the internal high-speed RAM area (FB00H to FEFFH) can be set as the stack area.

Figure 3-9. Configuration of Stack Pointer

	15															0
SP	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0

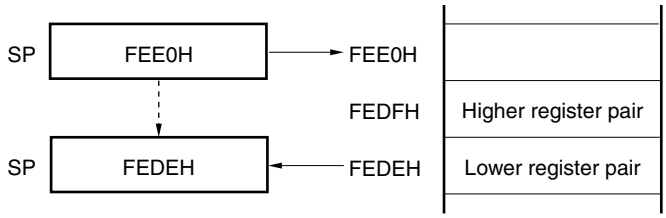
The SP is decremented ahead of write (save) to the stack memory and is incremented after read (restore) from the stack memory.

Each stack operation saves/restores data as shown in Figures 3-10 and 3-11.

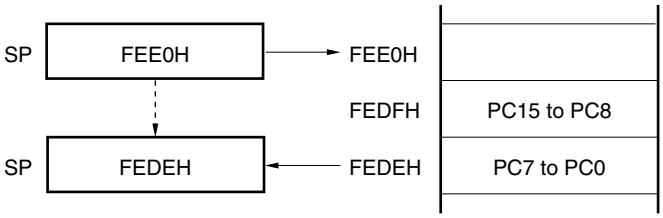
Caution Since reset input makes the SP contents undefined, be sure to initialize the SP before instruction execution.

★ **Figure 3-10. Data to Be Saved to Stack Memory**

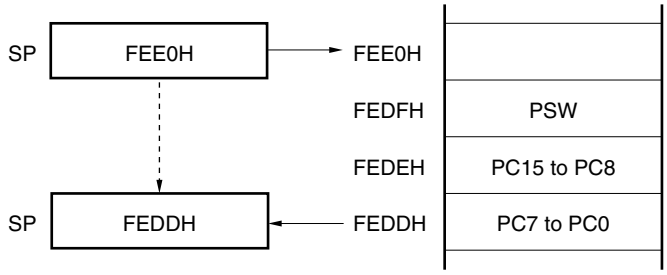
(a) PUSH rp instruction (when SP is FEE0H)



(b) CALL, CALLF, CALLT instructions (when SP is FEE0H)

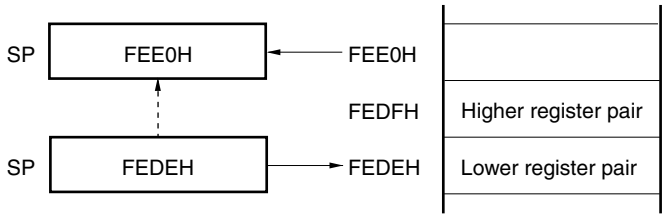


(c) Interrupt, BRK instructions (when SP is FEE0H)

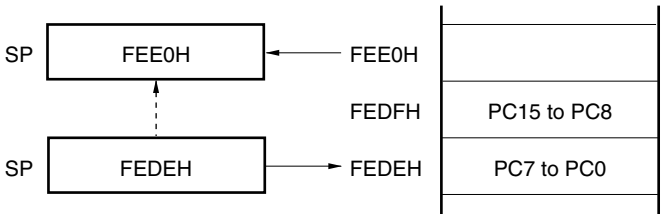


★ **Figure 3-11. Data to Be Restored from Stack Memory**

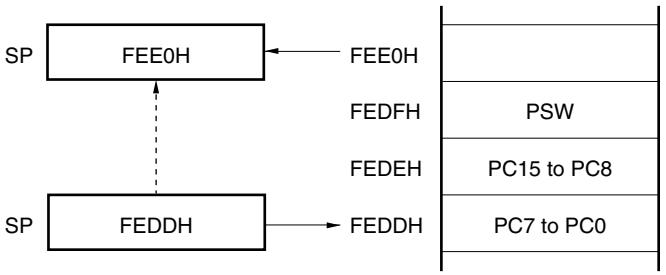
(a) POP rp instruction (when SP is FEDEH)



(b) RET instruction (when SP is FEDEH)



(c) RETI, RETB instructions (when SP is FEDDH)



3.2.2 General-purpose registers

General-purpose registers are mapped at particular addresses (FEE0H to FEFFH) of the data memory. They consist of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L and H).

Each register can be used as an 8-bit register, and two 8-bit registers can also be used in pairs as a 16-bit register (AX, BC, DE and HL).

They can be written using function names (X, A, C, B, E, D, L, H, AX, BC, DE and HL) and absolute names (R0 to R7 and RP0 to RP3).

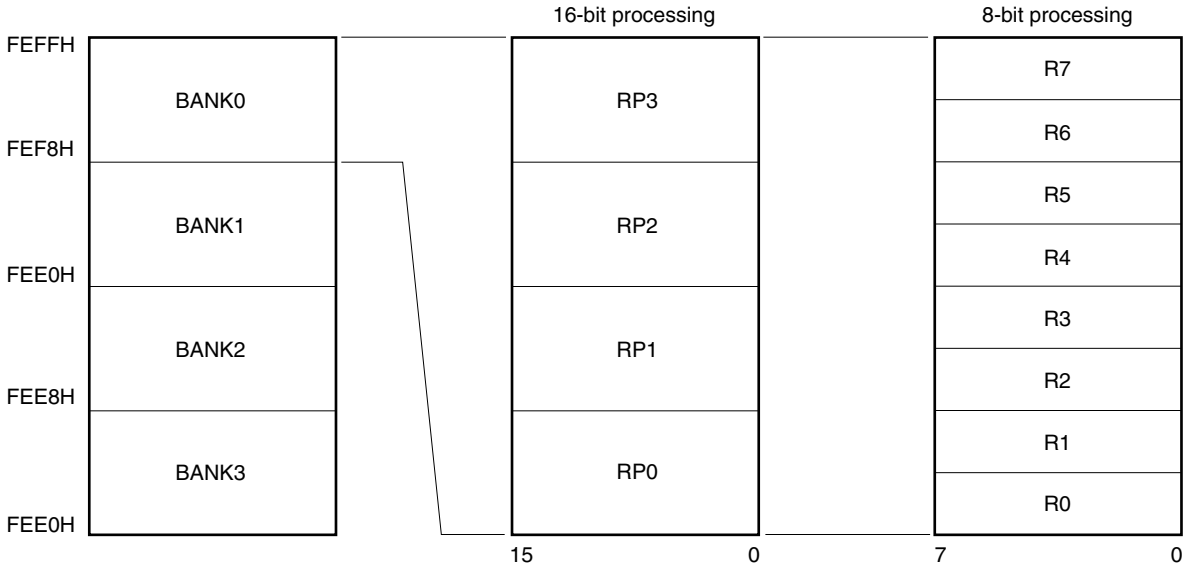
Register banks to be used for instruction execution are set by the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interrupt for each bank.

Table 3-3. Absolute Addresses of General-Purpose Registers

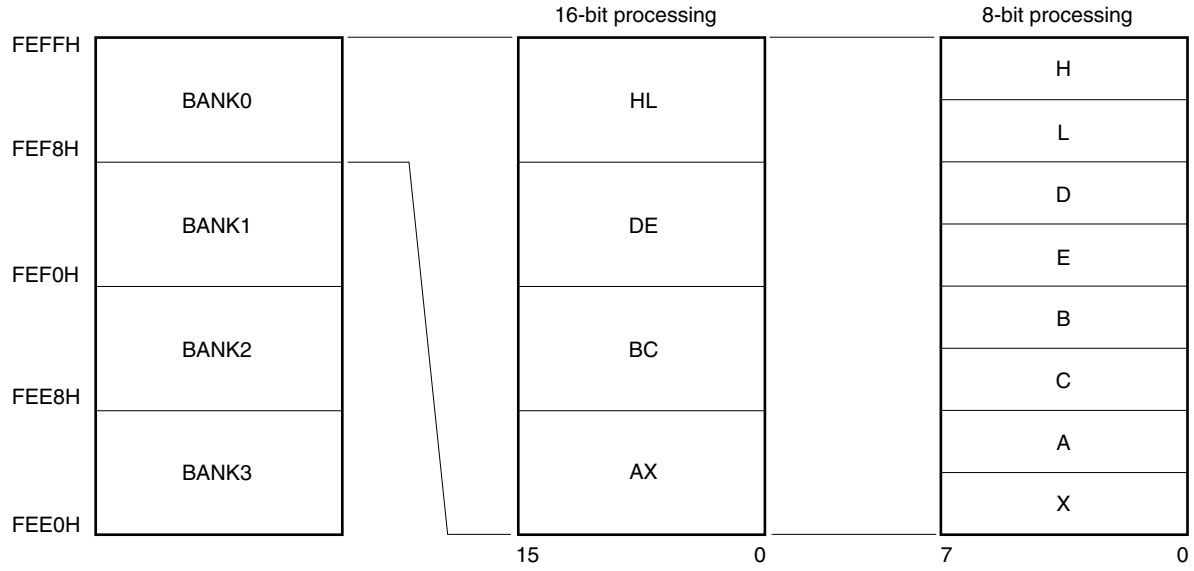
Bank	Register		Absolute Address	Bank	Register		Absolute Address
	Function Name	Absolute Name			Function Name	Absolute Name	
BANK0	H	R7	FEFFH	BANK2	H	R7	FEEFH
	L	R6	FEFEH		L	R6	FEEEH
	D	R5	FEFDH		D	R5	FEEDH
	E	R4	FEFCH		E	R4	FEECH
	B	R3	FEFBH		B	R3	FEEBH
	C	R2	FEFAH		C	R2	FEEAH
	A	R1	FEF9H		A	R1	FEE9H
	X	R0	FEF8H		X	R0	FEE8H
BANK1	H	R7	FEF7H	BANK3	H	R7	FEE7H
	L	R6	FEF6H		L	R6	FEE6H
	D	R5	FEF5H		D	R5	FEE5H
	E	R4	FEF4H		E	R4	FEE4H
	B	R3	FEF3H		B	R3	FEE3H
	C	R2	FEF2H		C	R2	FEE2H
	A	R1	FEF1H		A	R1	FEE1H
	X	R0	FEF0H		X	R0	FEE0H

Figure 3-12. General-Purpose Register Configuration

(a) Absolute name



(b) Function name



3.2.3 Special-function registers (SFR)

Unlike a general-purpose register, each special function register has a special function.

Special-function registers are allocated in the area FF00H to FFFFH.

Special-function registers can be manipulated like general-purpose registers, using operation, transfer, and bit manipulation instructions. The manipulatable bit units, 1, 8 and 16, depend on the special-function register type.

Each bit manipulation unit can be specified as follows.

- **1-bit manipulation**
  Use the symbol reserved in the assembler for the 1-bit manipulation instruction operand (sfr.bit).
  This manipulation can also be specified with an address.
- **8-bit manipulation**
  Use the symbol reserved in the assembler for the 8-bit manipulation instruction operand (sfr).
  This manipulation can also be specified with an address.
- **16-bit manipulation**
  Use the symbol reserved in the assembler for the 16-bit manipulation instruction operand (sfrp).
  When addressing an address, use an even address.

Table 3-4 gives a list of special-function registers. The meanings of items in the table are as follows.

- **Symbol**
  This is a symbol to indicate the address of the special-function register.
  These symbols are reserved in the DF178098 and RA78K0, and defined by the header file sfrbit.h in the CC78K0.
  They can be written as instruction operands when the RA78K0, ID78K0, ID78K0-NS, ID78K0-NS-A, or SM78K0 is used.
- **R/W**
  Indicates whether the corresponding special-function register can be read or written.
  R/W: Read/write enabled
  R: Read only
  R&Reset: Read only (reset to 0 when read)
  W: Write only
- **Manipulatable bits**
  ○ indicates the manipulatable bit units 1, 8, and 16. – indicates the bit units that cannot be manipulated.
- **After reset**
  Indicates the status of each register upon reset. The values of special-function registers whose addresses are not shown in the table are undefined after reset.

Table 3-4. Special-Function Registers (1/4)

Address	Special-Function Register (SFR) Name	Symbol	R/W	Manipulatable Bits			After Reset
				1	8	16	
FF00H	Port 0	P0	R/W	○	○	—	00H
FF01H	Port 1	P1	R	○	○	—	Undefined
FF02H	Port 2	P2	R/W	○	○	—	00H
FF03H	Port 3	P3		○	○	—	
FF04H	Port 4	P4		○	○	—	
FF05H	Port 5	P5		○	○	—	
FF06H	Port 6	P6		○	○	—	
FF07H	Port 7	P7		○	○	—	
FF0AH	Port 10	P10		○	○	—	
FF0CH	Port 12	P12		○	○	—	
FF0DH	Port 13	P13		○	○	—	
FF10H	A/D conversion result register 3[Note 1]	ADCR3	—	—	—	—	—
FF11H			R	—	○	—	Undefined
FF12H	A/D converter mode register 3	ADM3	R/W	○	○	—	00H
FF13H	Analog input channel specification register 3	ADS3		—	○	—	
FF15H	Power-fail comparison threshold value register 3	PFT3		—	○	—	
FF16H	Power-fail comparison mode register 3	PFM3		○	○	—	
FF19H	VM45 control register	VM45C		○	○	—	
FF1AH	Serial I/O shift register 0	SIO0		—	○	—	Undefined
FF1BH	POC status register	POCS	R&Reset	—	○	—	Held[Note 2]
FF20H	Port mode register 0	PM0	R/W	○	○	—	FFH
FF22H	Port mode register 2	PM2		○	○	—	
FF23H	Port mode register 3	PM3		○	○	—	
FF24H	Port mode register 4	PM4		○	○	—	
FF25H	Port mode register 5	PM5		○	○	—	
FF26H	Port mode register 6	PM6		○	○	—	
FF27H	Port mode register 7	PM7		○	○	—	
FF2AH	Port mode register 10	PM10		○	○	—	
FF2CH	Port mode register 12	PM12		○	○	—	
FF40H	Clock output select register	CKS		○	○	—	00H
FF41H	BEEP frequency select register 0	BEEPCL0		○	○	—	
FF42H	Watchdog timer clock select register	WDCS		○	○	—	
FF43H	Serial interface clock select register 0	SCL0		○	○	—	08H
FF48H	External interrupt rising edge enable register	EGP		○	○	—	00H
FF49H	External interrupt falling edge enable register	EGN		○	○	—	

Notes 1. This register can be accessed only in 8-bit units. When ADCR3 is read, the value of FF11H is read.

2. The value of this register is 03H only after reset by power-on clear. This register is not reset by the $\overline{\text{RESET}}$ pin or watchdog timer.

Caution Do not access addresses to which no SFR is assigned.

Table 3-4. Special-Function Registers (2/4)

Address	Special-Function Register (SFR) Name	Symbol	R/W	Manipulatable Bits			After Reset
				1	8	16	
FF5AH	Asynchronous serial interface mode register 0[Note]	ASIM0	R/W	○	○	—	00H
FF5BH	Asynchronous serial interface status register 0[Note]	ASIS0	R	—	○	—	
FF5CH	Baud rate generator control register 0[Note]	BRGC0	R/W	—	○	—	
FF5DH	Transmit shift register 0[Note]	TXS0	W	—	○	—	FFH
	Receive buffer register 0[Note]	RXB0	R	—	○	—	FFH
FF60H	Serial operating mode register 0	CSIM0	R/W	○	○	—	00H
FF61H	Serial bus interface control register 0	SBIC0		○	○	—	
FF62H	Slave address register 0	SVA0		—	○	—	Undefined
FF63H	Interrupt timing specification register 0	SINT0		○	○	—	00H
FF67H	Serial I/O shift register 1	SIO1		—	○	—	Undefined
FF68H	Serial operating mode register 1	CSIM1		○	○	—	00H
FF69H	Automatic data transmit/receive address pointer	ADTP		—	○	—	
FF6AH	Automatic data transmit/receive control register	ADTC		○	○	—	
FF6BH	Automatic data transmit/receive interval specification register	ADTI		○	○	—	
FF6EH	Serial I/O shift register 3	SIO3		—	○	—	Undefined
FF6FH	Serial operating mode register 3	CSIM3		○	○	—	00H
FF70H	16-bit timer counter 0	TM0	R	—	—	○	0000H
FF71H				—	—		
FF72H	16-bit capture/compare register 00	CR00	R/W	—	—	○	Undefined
FF73H				—	—		
FF78H	16-bit timer mode control register 0	TMC0		○	○	—	00H
FF7AH	Prescaler mode register 0	PRM0		—	○	—	
FF7CH	Capture/compare control register 0	CRC0		○	○	—	
FF7EH	16-bit timer output control register 0	TOC0		○	○	—	
FF80H	8-bit compare register 50	CR50		—	○	—	Undefined
FF81H	8-bit compare register 51	CR51		—	○	—	
FF82H	8-bit timer counter 50	TM5 TM50	R	—	○	○	00H
FF83H	8-bit timer counter 51	TM51		—	○		
FF84H	Timer clock select register 50	TCL50	R/W	—	○	—	
FF85H	8-bit timer mode control register 50	TMC50		○	○	—	
FF87H	Timer clock select register 51	TCL51		—	○	—	
FF88H	8-bit timer mode control register 51	TMC51		○	○	—	
FFA0H	PLL mode select register	PLLMD		○	○	—	
FFA1H	PLL reference mode register	PLLRF		○	○	—	0FH

Note μPD178076, 178078, and 178F098 only

Caution Do not access addresses to which no SFR is assigned.

Table 3-4. Special-Function Registers (3/4)

Address	Special-Function Register (SFR) Name		Symbol		R/W	Manipulatable Bits			After Reset
						1	8	16	
FFA2H	PLL unlock F/F judge register		PLLUL		R&Reset	○	○	—	Undefined
FFA3H	PLL data transfer register		PLLNS		W	○	○	—	00H
FFA6H	PLL data registers	PLL data register L	PLLR	PLLRL	R/W	○	○	○	Undefined
FFA7H		PLL data register H		PLLRH		○	○		
FFA8H	PLL data register 0		PLLR0			○	○	—	
FFA9H	IF counter mode select register		IFCMD			○	○	—	00H
FFAAH	DTS system clock select register		DTSCK			○	○	—	
FFABH	IF counter gate judge register		IFCJG		R	○	○	—	
FFACH	IF counter control register		IFCCR		W	○	○	—	
FFAEH	IF counter register		IFCR	IFCRL	R	—	—	○	
FFAFH				IFCRH		—	—		
FFB0H	IEBus control register 0[Note 1]		BCR0		R/W	○	○	—	
FFB2H	IEBus unit address register[Note 1]		UAR	UARL		—	○	○	
FFB3H				UARH		—	○		
FFB4H	IEBus slave address register[Note 1]		SAR	SARL		—	○	○	
FFB5H				SARH		—	○		
FFB6H	IEBus partner unit address register[Note 1]		PAR	PARL		—	○	○	
FFB7H				PARH		—	○		
FFB8H	IEBus control data register[Note 1]		CDR			—	○	—	01H
FFB9H	IEBus telegraph length register[Note 1]		DLR			—	○	—	
FFBAH	IEBus data register[Note 1]		DR			—	○	—	00H
FFBBH	IEBus unit status register[Note 1]		USR		R	○	○	—	
FFBCH	IEBus interrupt status register[Note 1]		ISR		R/W	○	○	—	
FFBDH	IEBus slave status register[Note 1]		SSR		R	○	○	—	41H
FFBEH	IEBus communication successful counter[Note 1]		SCR			—	○	—	01H
FFBFH	IEBus transmission counter[Note 1]		CCR			—	○	—	20H
FFD0H | FFDFH	External access area[Note 2]				R/W	○	○	—	Undefined
FFE0H	Interrupt request flag register 0L		IF0	IF0L		○	○	○	00H
FFE1H	Interrupt request flag register 0H			IF0H		○	○		
FFE2H	Interrupt request flag register 1L		IF1L			○	○	—	
FFE4H	Interrupt mask flag register 0L		MK0	MK0L		○	○	○	FFH
FFE5H	Interrupt mask flag register 0H			MK0H		○	○		
FFE6H	Interrupt mask flag register 1L		MK1L			○	○	—	

Notes 1. μPD178096A, 178098A, and 178F098 only

2. The external access area cannot be accessed by means of SFR addressing. Use direct addressing to access this area.

Caution Do not access addresses to which no SFR is assigned.

Table 3-4. Special-Function Registers (4/4)

Address	Special-Function Register (SFR) Name	Symbol		R/W	Manipulatable Bits			After Reset
					1	8	16	
FFE8H	Priority specification flag register 0L	PR0	PR0L	R/W	○	○	○	FFH
FFE9H	Priority specification flag register 0H		PR0H		○	○		
FFEAH	Priority specification flag register 1L	PR1L			○	○	—	
FFF0H	Memory size select register**Note**	IMS			—	○	—	CFH**Note**
FFF4H	Internal expansion RAM size select register**Note**	IXS			—	○	—	0CH**Note**
FFF9H	Watchdog timer mode register	WDTM			○	○	—	00H
FFFAH	Oscillation stabilization time select register	OSTS			—	○	—	04H
FFFBH	Processor clock control register	PCC			○	○	—	

Note The initial values of the memory size select register (IMS) and internal expansion RAM size select register (IXS) are CFH and 0CH, respectively. Set the values of these registers in each product as follows.

Part Number	IMS	IXS
μPD178076, 178096A	CCH	0AH
μPD178078, 178098A	CFH	08H
μPD178F098	Value corresponding to mask ROM version	Value corresponding to mask ROM version

Caution Do not access addresses to which no SFR is assigned.

3.3 Instruction Address Addressing

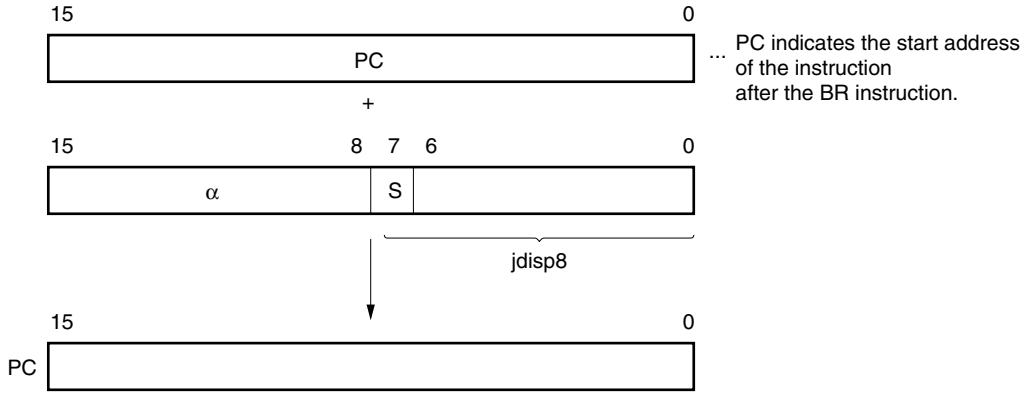
An instruction address is determined by the program counter (PC) contents, which are normally incremented (+1 for each byte) automatically according to the number of bytes of an instruction to be fetched each time another instruction is executed. When a branch instruction is executed, the branch destination information is set to the PC and branched by the following addressing. (For details of instructions, refer to the **78K/0 Instruction User's Manual (U12326E)**.)

3.3.1 Relative Addressing

[Function]

The value obtained by adding the 8-bit immediate data (displacement value: jdisp8) of an instruction code to the start address of the following instruction is transferred to the program counter (PC) and branched. The displacement value is treated as signed two's complement data (–128 to +127) and bit 7 becomes a sign bit. That is, using relative addressing, the program branches in the range –128 to +127 relative to the first address of the next instruction. This function is carried out when the BR $addr16 instruction or a conditional branch instruction is executed.

[Illustration]



When S = 0, all bits of α are 0.
When S = 1, all bits of α are 1.

3.3.2 Immediate addressing

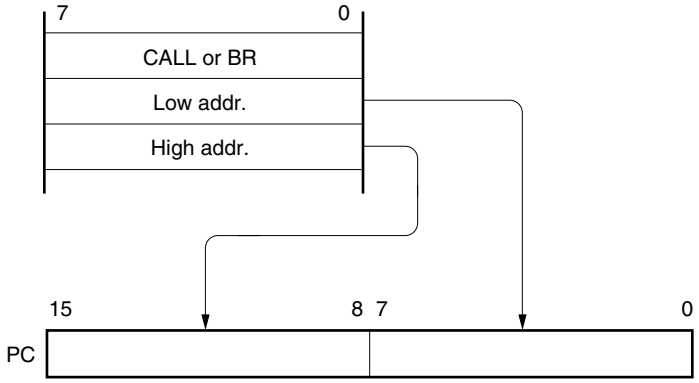
[Function]

The immediate data in the instruction word is transferred to the program counter (PC) and branched.

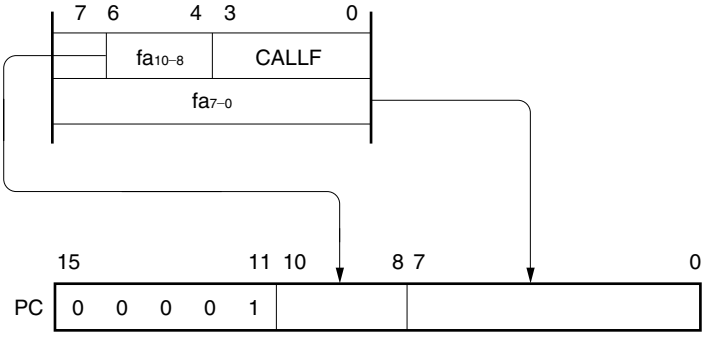
This function is carried out when the CALL !addr16, BR !addr16, or CALLF !addr11 instruction is executed. The CALL !addr16 and BR !add16 instructions can be used to branch to any location in the memory. The CALLF !addr11 instruction is used to branch to the area between 0800H and 0FFFH.

[Illustration]

In the case of the CALL !addr16 and BR !addr16 instructions



In the case of the CALLF !addr11 instruction



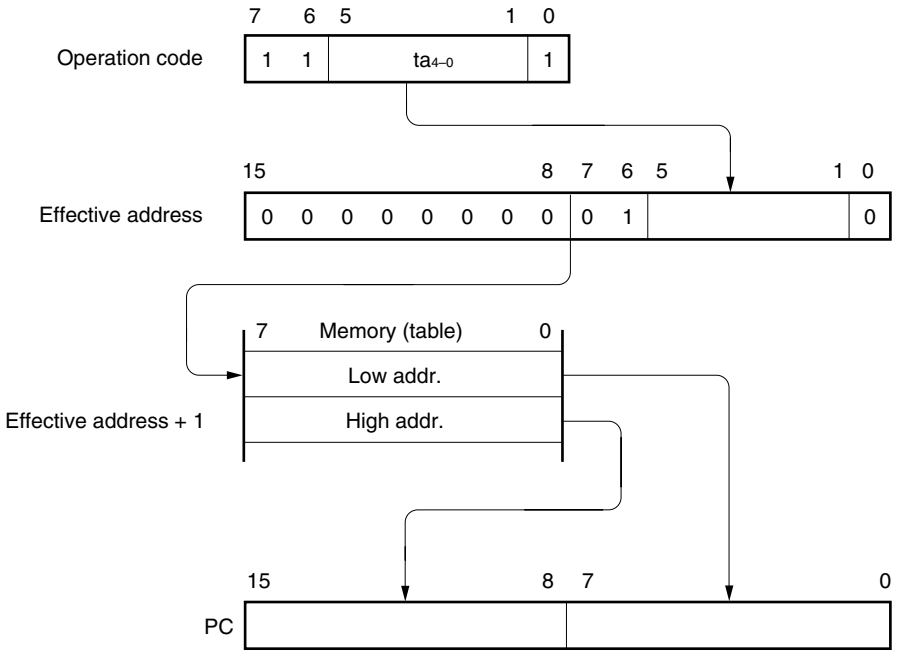
3.3.3 Table indirect addressing

[Function]

The table contents (branch destination address) of the particular location to be addressed by bits 1 to 5 of the immediate data of an operation code are transferred to the program counter (PC) and branched.

This addressing is used when the CALLT [addr5] instruction is executed. This instruction references an address stored in the memory table between 40H and 7FH, and can be used to branch to any location in the memory.

[Illustration]



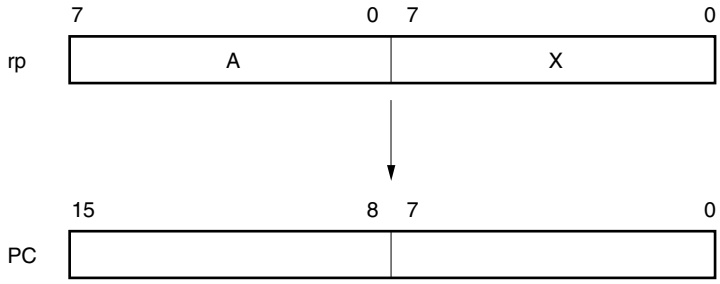
3.3.4 Register addressing

[Function]

The register pair (AX) contents to be specified with an instruction word are transferred to the program counter (PC) and branched.

This function is carried out when the BR AX instruction is executed.

[Illustration]



3.4 Operand Address Addressing

The following methods are available to specify the register and memory (addressing) that undergo manipulation during instruction execution.

3.4.1 Implied addressing

[Function]

The register that functions as an accumulator (A and AX) in the general-purpose register area is automatically addressed (implied). Of the μPD178078 and 178098A Subseries instruction words, the following instructions employ implied addressing.

Instruction	Register Specified by Implied Addressing
MULU	A register for multiplicand and AX register for product storage
DIVUW	AX register for dividend and quotient storage
ADJBA/ADJBS	A register for storage of numeric values which become decimal correction targets
ROR4/ROL4	A register for storage of digit data that undergoes digit rotation

[Operand format]

Because implied addressing can be automatically employed with an instruction, no particular operand format is necessary.

[Example]

In the case of MULU X

With an 8-bit × 8-bit multiply instruction, the product of the A register and X register is stored in AX. In this example, the A and AX registers are specified by implied addressing.

3.4.2 Register addressing

[Function]

This addressing mode is used to access a general-purpose register as an operand. The register to be accessed is specified by the register bank select flags (RBS0 and RBS1) and the register specification codes (Rn and RPn) in the operation code.

Register addressing is carried out when an instruction with the following operand format is executed. When an 8-bit register is specified, one of the eight registers is specified with 3 bits in the operation code.

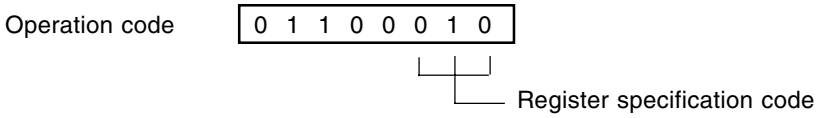
[Operand format]

Symbol	Description
r	X, A, C, B, E, D, L, H
rp	AX, BC, DE, HL

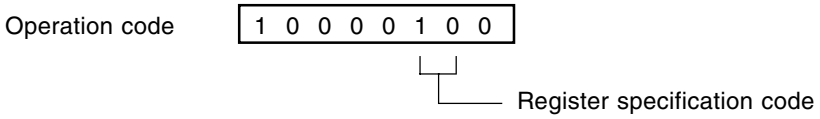
'r' and 'rp' can be written using function names (X, A, C, B, E, D, L, H, AX, BC, DE and HL) as well as absolute names (R0 to R7 and RP0 to RP3).

[Example]

MOV A, C; when selecting C register as r



INCW DE; when selecting DE register pair as rp



3.4.3 Direct addressing

[Function]

The memory with immediate data in an instruction word is directly addressed.

[Operand format]

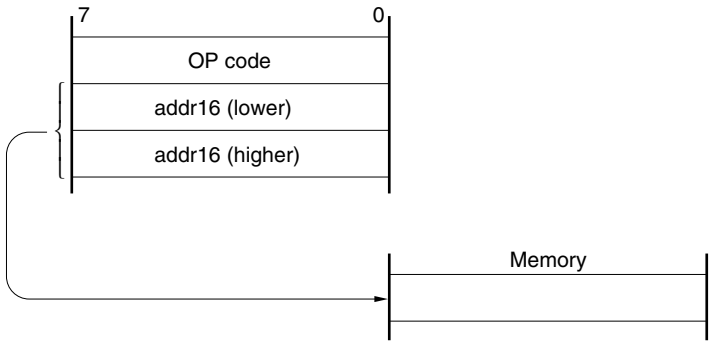
Symbol	Description
addr16	Label or 16-bit immediate data

[Example]

MOV A, !0FE00H; when setting !addr16 to FE00H

Operation code	1 0 0 0 1 1 1 0	Opcode
	0 0 0 0 0 0 0 0	00H
	1 1 1 1 1 1 1 0	FEH

[Illustration]



3.4.4 Short direct addressing

[Function]

The memory to be manipulated in the fixed space is directly addressed with 8-bit data in an instruction word. The fixed space to which this addressing is applied is the 256-byte space FE20H to FF1FH. An internal RAM and special-function registers (SFRs) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.

The SFR area (FF00H to FF1FH) where short direct addressing is applied is part of the total the SFR area. In this area, ports which are frequently accessed in a program and a compare register and a capture register of the timer/event counter are mapped and these SFRs can be manipulated with a small number of bytes and clocks.

When 8-bit immediate data is at 20H to FFH, bit 8 of an effective address is set to 0. When it is at 00H to 1FH, bit 8 is set to 1. Refer to [**Illustration**] below.

[Operand format]

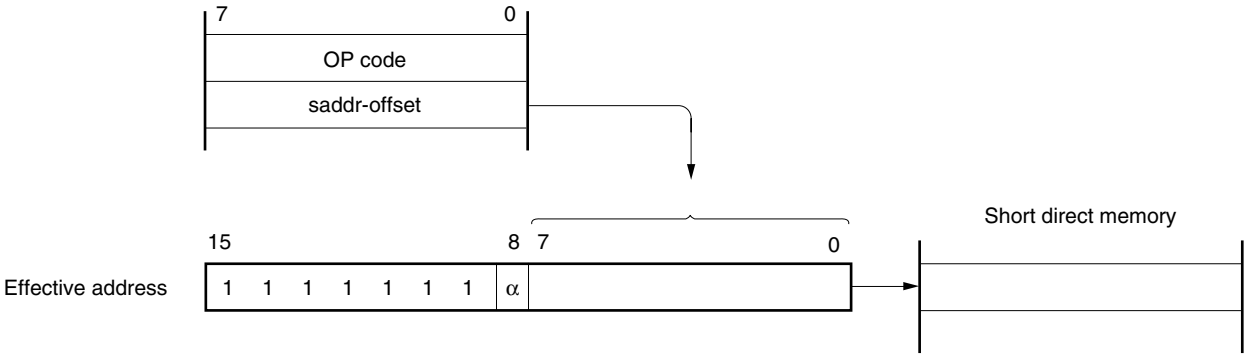
Symbol	Description
saddr	Label or immediate data indicating FE20H to FF1FH
saddrp	Label or immediate data indicating FE20H to FF1FH (even addresses only)

★ **[Example]**

MOV FE30H, A; when transferring the value of register A to saddr (FE30H)

Operation code	1 1 1 1 0 0 1 0	Opcode
	0 0 1 1 0 0 0 0	30H (saddr-offset)

[Illustration]



When 8-bit immediate data is 20H to FFH, $\alpha = 0$

When 8-bit immediate data is 00H to 1FH, $\alpha = 1$

3.4.5 Special-function register (SFR) addressing

[Function]

A memory-mapped special-function register (SFR) is addressed with 8-bit immediate data in an instruction word. This addressing is applied to the 240-byte spaces FF00H to FFCFH and FFE0H to FFFFH. However, the SFRs mapped at FF00H to FF1FH can be accessed with short direct addressing.

[Operand format]

Symbol	Description
sfr	Special-function register name
sfrp	16-bit manipulatable special-function register name (even addresses only)

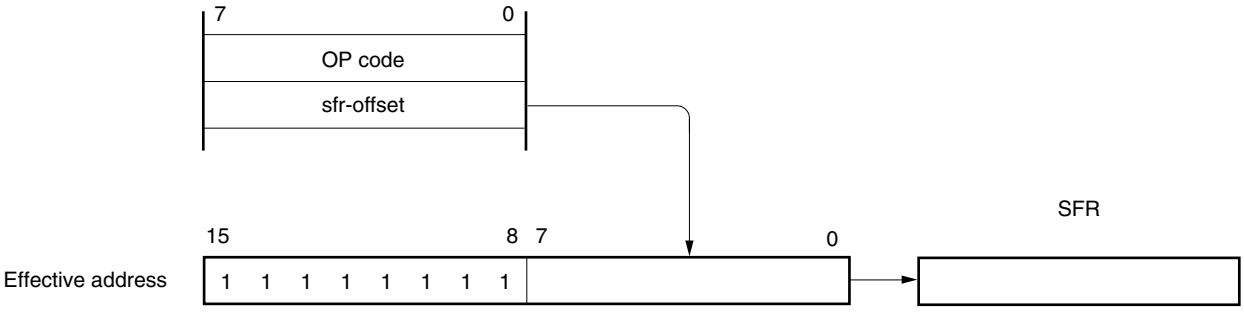
[Example]

MOV PM0, A; when selecting PM0 (FF20H) as sfr

Operation code 1 1 1 1 0 1 1 0

0 0 1 0 0 0 0 0

[Illustration]



3.4.6 Register indirect addressing

[Function]

This addressing is used to address the memory to be manipulated by using the contents of the register pair specified by the register pair code in an instruction word as the operand address. The register pair specified is in the register bank specified by the register bank select flags (RBS0 and RBS1). This addressing can be used for the entire memory space.

[Operand format]

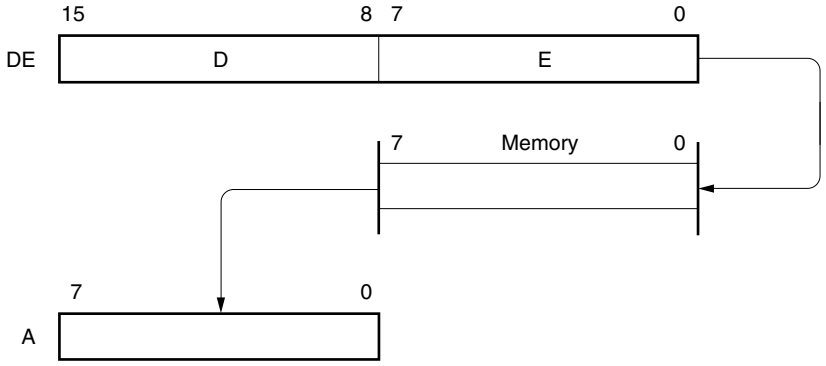
Symbol	Description
—	[DE], [HL]

[Example]

MOV A, [DE]; when selecting [DE] as register pair

Operation code	1	0	0	0	0	1	0	1

[Illustration]



3.4.7 Based addressing

[Function]

This addressing mode is used to address a memory location specified by the result of adding the 8-bit immediate data to the contents of the HL register pair which is used as a base register. The HL register pair accessed is the register in the register bank specified by the register bank select flags (RBS0 and RBS1). Addition is performed by expanding the offset data as a positive number to 16 bits. A carry from the 16th bit is ignored. This addressing can be used for the entire memory space.

[Operand format]

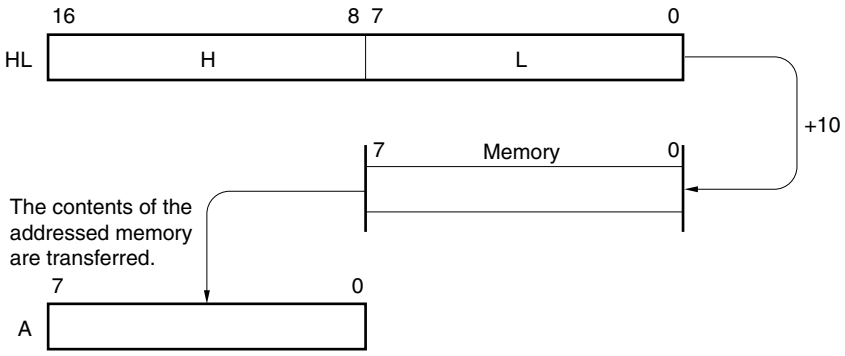
Symbol	Description
—	[HL + byte]

[Example]

MOV A, [HL + 10H]; when setting byte to 10H

Operation code	1 0 1 0 1 1 1 0
	0 0 0 1 0 0 0 0

★ **[Illustration]**



3.4.8 Based indexed addressing

[Function]

This addressing mode is used to address a memory location specified by the result of adding the contents of the B or C register specified in the instruction word to the contents of the HL register pair which is used as a base register. The HL, B, and C registers accessed are the registers in the register bank specified by the register bank select flags (RBS0 and RBS1).

Addition is performed by expanding the offset data as a positive number to 16 bits. A carry from the 16th bit is ignored. This addressing can be used for the entire memory space.

[Operand format]

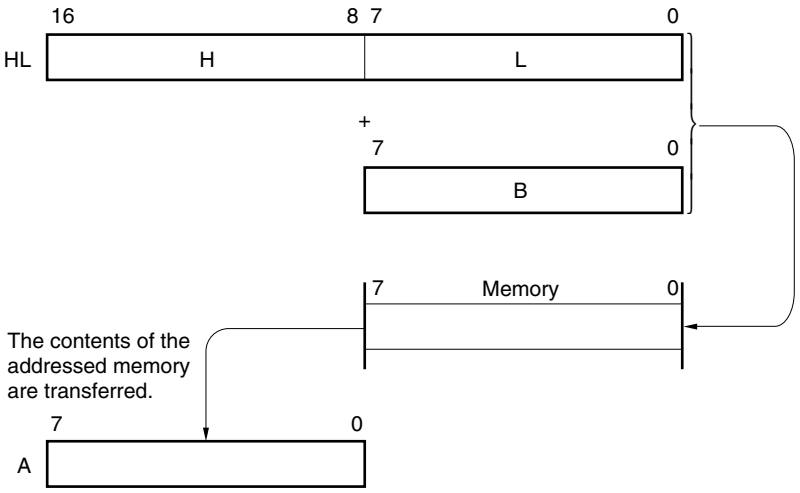
Symbol	Description
—	[HL + B], [HL + C]

[Example]

In the case of MOV A, [HL + B]

Operation code	1	0	1	0	1	0	1	1

★ **[Illustration]**



3.4.9 Stack addressing

[Function]

The stack area is indirectly addressed with the stack pointer (SP) contents.

This addressing method is automatically employed when the PUSH, POP, subroutine call, and RETURN instructions are executed or the register is saved/restored upon generation of an interrupt request.

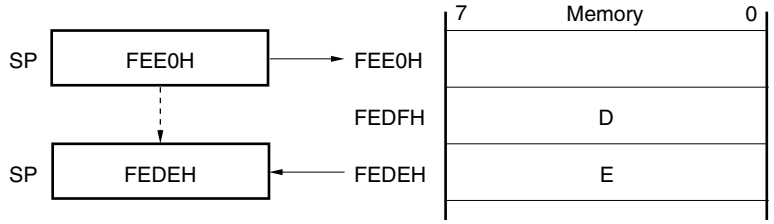
Stack addressing can be used to address the internal high-speed RAM area only.

[Example]

In the case of PUSH DE

Operation code	1	0	1	1	0	1	0	1

★ **[Illustration]**



CHAPTER 4 PORT FUNCTIONS

4.1 Port Functions

The μPD178078 and 178098A Subseries products incorporate eight input ports, eight output ports and 64 I/O ports. Figure 4-1 shows the port configuration. Every port is capable of 1-bit and 8-bit manipulations and can carry out considerably varied control operations. Besides port functions, the ports can also serve as on-chip hardware I/O pins.

Figure 4-1. Port Types

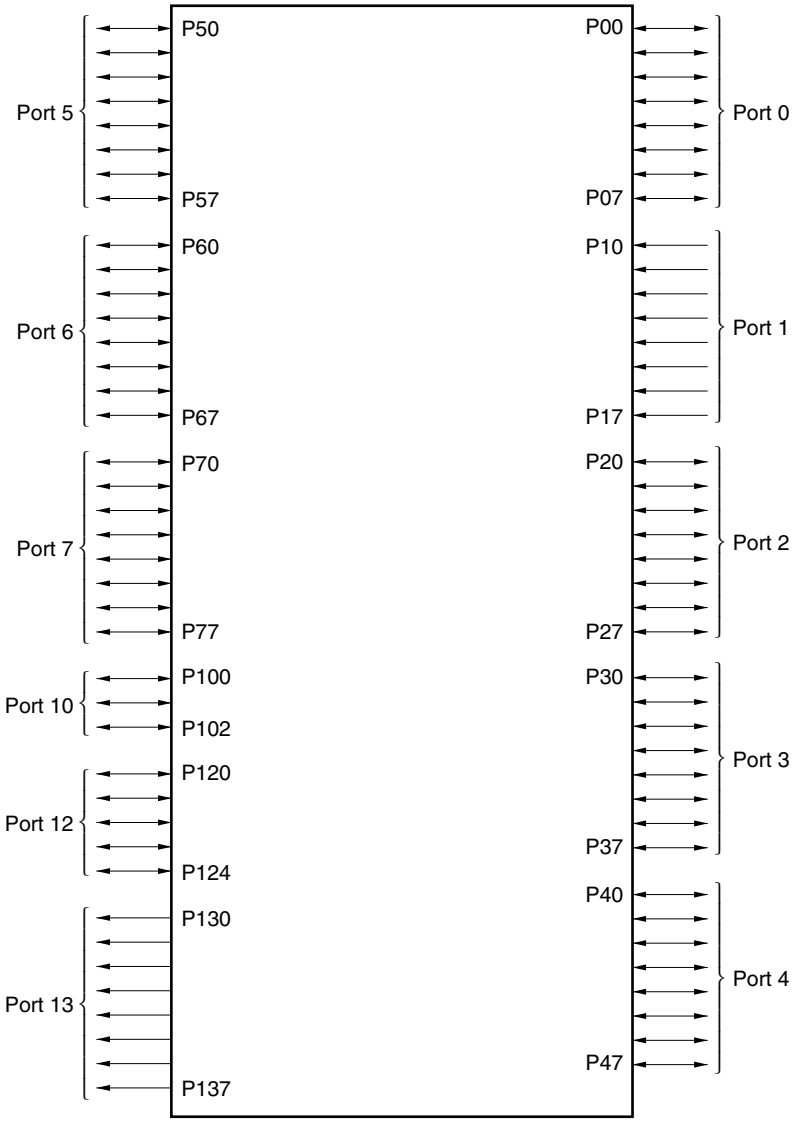


Table 4-1. Port Functions (1/2)

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P07	I/O	Port 0. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	INTP0 to INTP7
P10 to P17	Input	Port 1. 8-bit input port.	Input	ANI0 to ANI7
P20	I/O	Port 2. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	SI1
P21				SO1
P22				$\overline{\text{SCK1}}$
P23				STB
P24				BUSY
P25				SI0/SB0/SDA0
P26				SO0/SB1/SDA1
P27				$\overline{\text{SCK0}}$/SCL
P30	I/O	Port 3. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	VM45
P31				TO0
P32				TI00
P33				TI01
P34				TI50
P35				TI51
P36				BEEP0
P37				BUZ
P40 to 47	I/O	Port 4. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	–
P50 to P57	I/O	Port 5. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	–
P60 to P67	I/O	Port 6. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	–
P70	I/O	Port 7. 8-bit I/O port. Can be set to input or output mode in 1-bit units.	Input	SI3
P71				SO3
P72				$\overline{\text{SCK3}}$
P73				–
P74				RXD0**Note 1**
P75				TXD0**Note 1**
P76, P77				–

Table 4-1. Port Functions (2/2)

Pin Name	I/O	Function	After Reset	Alternate Function
P100	I/O	Port 10.	Input	–
P101		3-bit I/O port.		AMIFC
P102		Can be set to input or output mode in 1-bit units.		FMIFC
P120	I/O	Port 12.	Input	$\overline{TX0}$ Note 2
P121		5-bit I/O port.		$\overline{RX0}$ Note 2
P122 to P124		Can be set to input or output mode in 1-bit units.		–
P130	Output	Port 13.	–	TO50
P131		8-bit output port.		TO51
P132 to P137		N-ch open-drain output port (12 V tolerant)		–

Notes 1. μPD178076, 178078, and 178F098 only.

2. μPD178096A, 178098A, and 178F098 only.

4.2 Port Configuration

A port consists of the following hardware.

Table 4-2. Configuration of Port

Item	Configuration
Control register	Port mode register (PMm: m = 0, 2 to 7, 10, 12)
Port	Total: 80 pins (8 input, 8 output, 64 I/O)

4.2.1 Port 0

Port 0 is an 8-bit I/O port with an output latch. The P00 to P07 pins can be set to input mode/output mode in 1-bit units using port mode register 0 (PM0).

Alternate functions include external interrupt request input.

Reset input sets port 0 to the input mode.

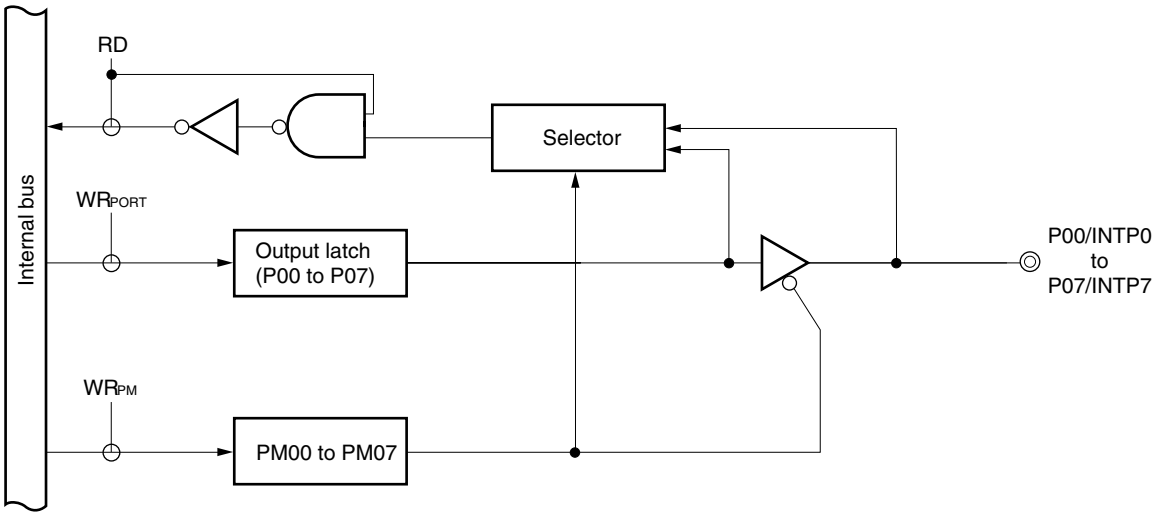
Figure 4-2 shows the block diagram of port 0.

Cautions 1. Because port 0 also serves as an external interrupt request input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. When using the output mode, therefore, preset the interrupt mask flag to 1.

★ **2. When the external interrupt request function is switched to the port function, edge detection may be performed. Therefore, set bit n (EGPn) of the external interrupt rising edge enable register (EGP) and bit n (EGNn) of the external interrupt falling edge enable register (EGN) to 0 before selecting the port mode.**

Remark n = 0 to 7

Figure 4-2. Block Diagram of P00 to P07



PM: Port mode register

RD: Port 0 read signal

WR: Port 0 write signal

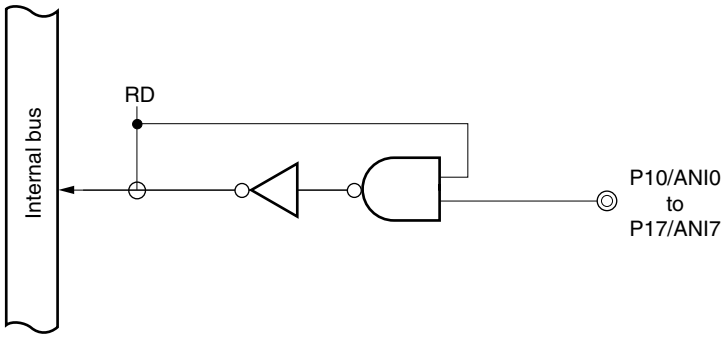
4.2.2 Port 1

Port 1 is an 8-bit input port.

Alternate functions include A/D converter analog input.

Figure 4-3 shows the block diagram of port 1.

Figure 4-3. Block Diagram of P10 to P17



RD: Port 1 read signal

4.2.3 Port 2

Port 2 is an 8-bit I/O port with an output latch. The P20 to P27 pins can be set to input mode/output mode in 1-bit units using port mode register 2 (PM2).

Alternate functions include serial interface data I/O, clock I/O, automatic transmit/receive busy input, and strobe output.

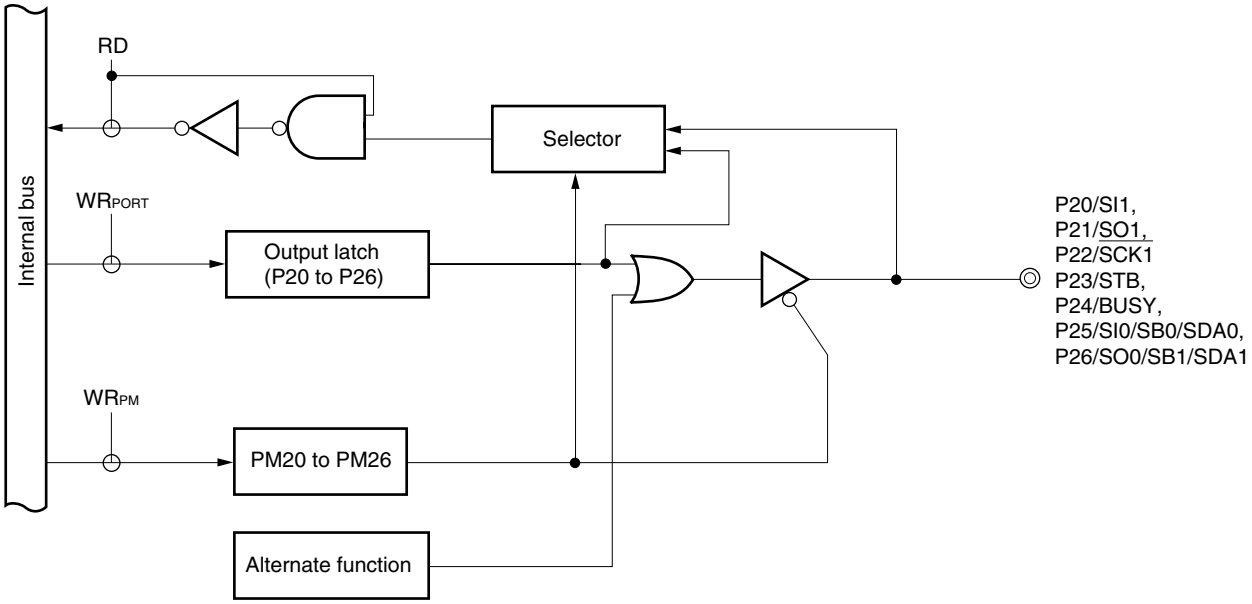
Reset input sets port 2 to the input mode.

Figures 4-4 and 4-5 show the block diagrams of port 2.

Cautions 1. When using port 2 for the serial interface, set the input/output mode and output latch according to the function used. For the setting method, refer to Table 4-3 Port Mode Register and Output Latch Settings When Using Alternate Functions and Figure 13-5 Format of Serial Operating Mode Register 0 (CSIM0).

2. When reading the pin state in SBI mode, set PM2n to 1 (n = 5, 6) (refer to the description of (10) Method used to judge busy state of a slave (e) in 13.4.3 SBI mode operation).

Figure 4-4. Block Diagram of P20 to P26

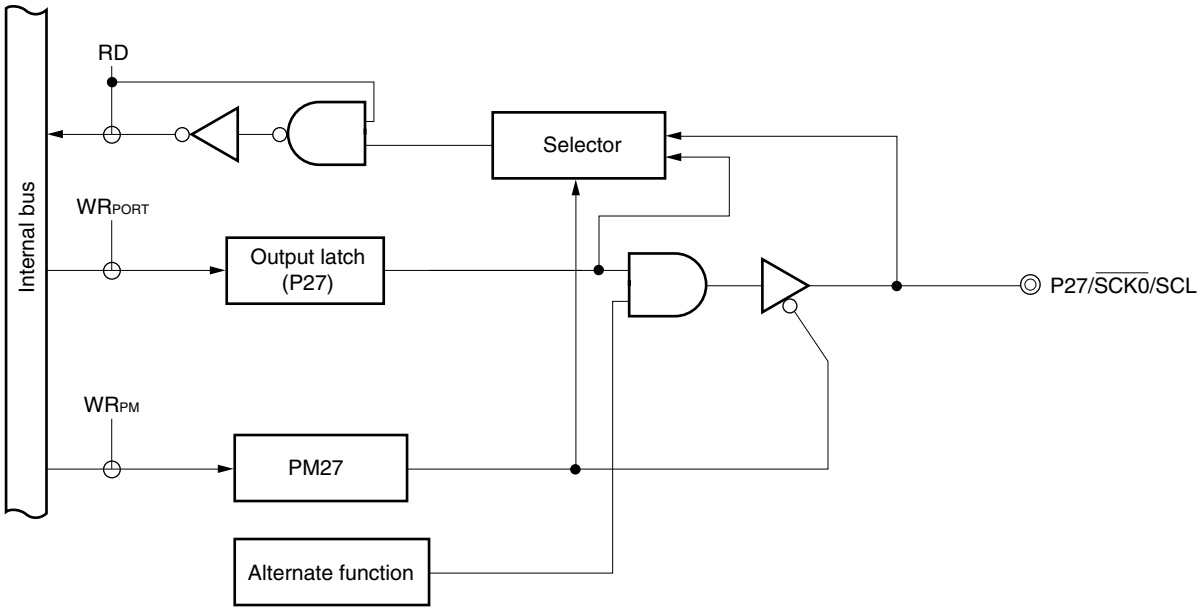


PM: Port mode register

RD: Port 2 read signal

WR: Port 2 write signal

Figure 4-5. Block Diagram of P27



PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

4.2.4 Port 3

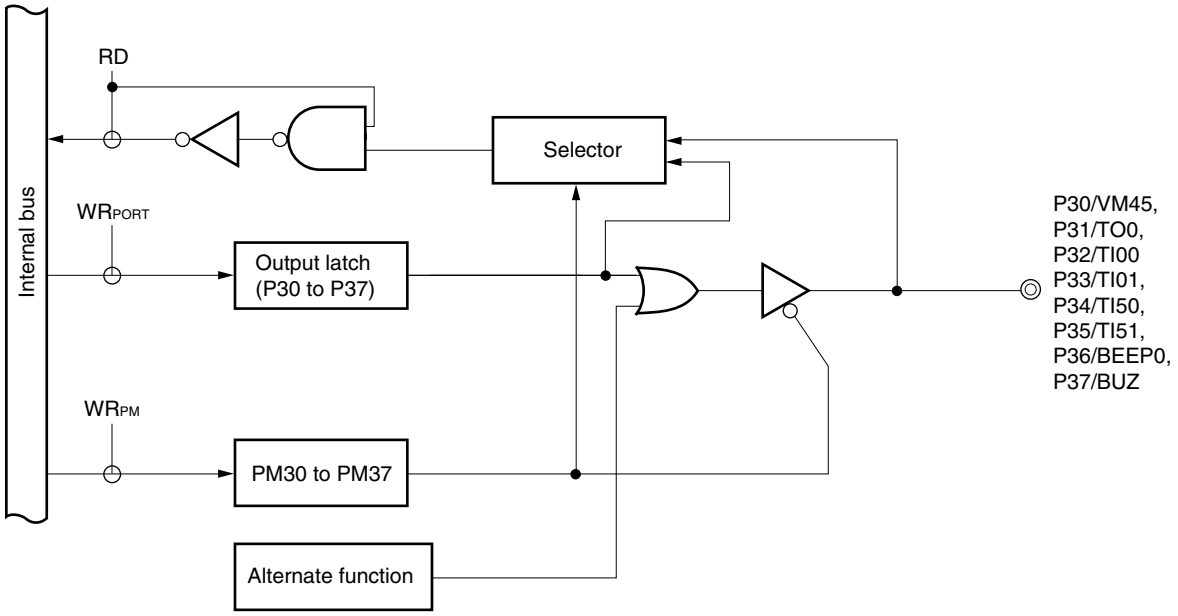
Port 3 is an 8-bit I/O port with an output latch. The P30 to P37 pins can be set to input mode/output mode in 1-bit units using port mode register 3 (PM3).

Alternate functions include V_{DD} = 4.5 V monitor output, timer output, timer input, and buzzer output.

Reset input sets port 3 to the input mode.

Figure 4-6 shows the block diagram of port 3.

Figure 4-6. Block Diagram of P30 to P37



PM: Port mode register

RD: Port 3 read signal

WR: Port 3 write signal

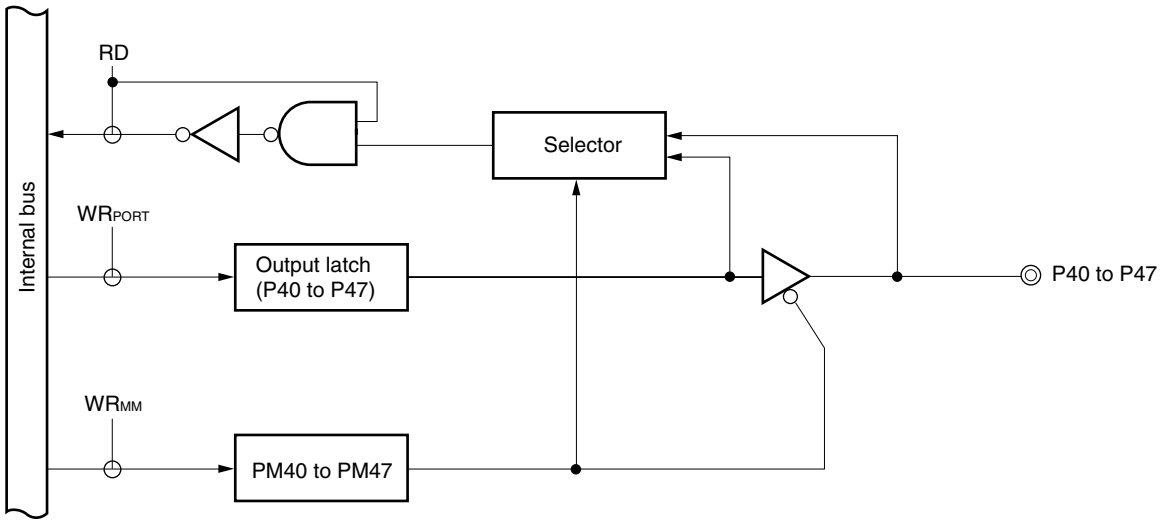
4.2.5 Port 4

Port 4 is an 8-bit I/O port with an output latch. The P40 to P47 pins can be set to input mode/output mode in 8-bit units using port mode register 4 (PM4).

Reset input sets port 4 to the input mode.

Figure 4-7 shows the block diagram of port 4.

Figure 4-7. Block Diagram of P40 to P47



PM: Port mode register
RD: Port 4 read signal
WR: Port 4 write signal

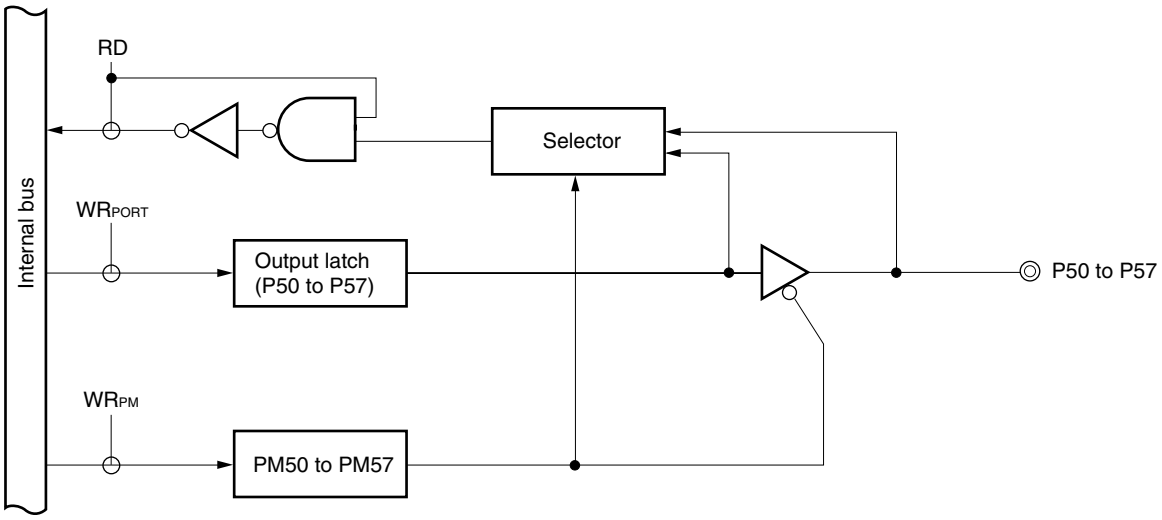
4.2.6 Port 5

Port 5 is an 8-bit I/O port with an output latch. The P50 to P57 pins can be set to input mode/output mode in 1-bit units using port mode register 5 (PM5).

Reset input sets port 5 to the input mode.

Figure 4-8 shows the block diagram of port 5.

Figure 4-8. Block Diagram of P50 to P57



PM: Port mode register
RD: Port 5 read signal
WR: Port 5 write signal

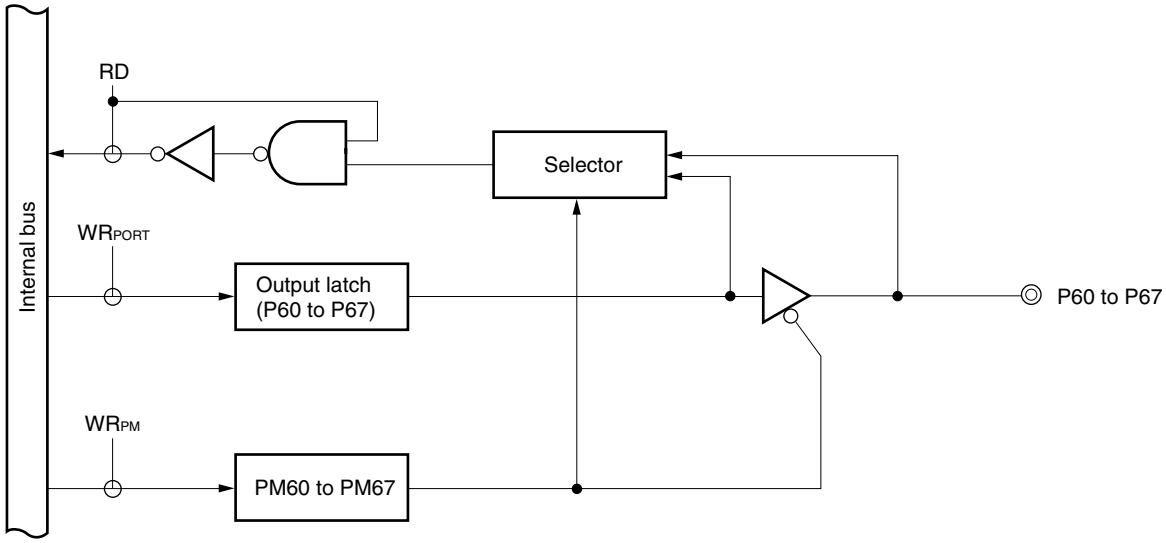
4.2.7 Port 6

Port 6 is an 8-bit I/O port with an output latch. The P60 to P67 pins can be set to input mode/output mode in 1-bit units using port mode register 6 (PM6).

Reset input sets port 6 to the input mode.

Figure 4-9 shows the block diagram of port 6.

Figure 4-9. Block Diagram of P60 to P67



PM: Port mode register
RD: Port 6 read signal
WR: Port 6 write signal

4.2.8 Port 7

Port 7 is an 8-bit I/O port with an output latch. The P70 to P77 pins can be set to input mode/output mode in 1-bit units using port mode register 7 (PM7).

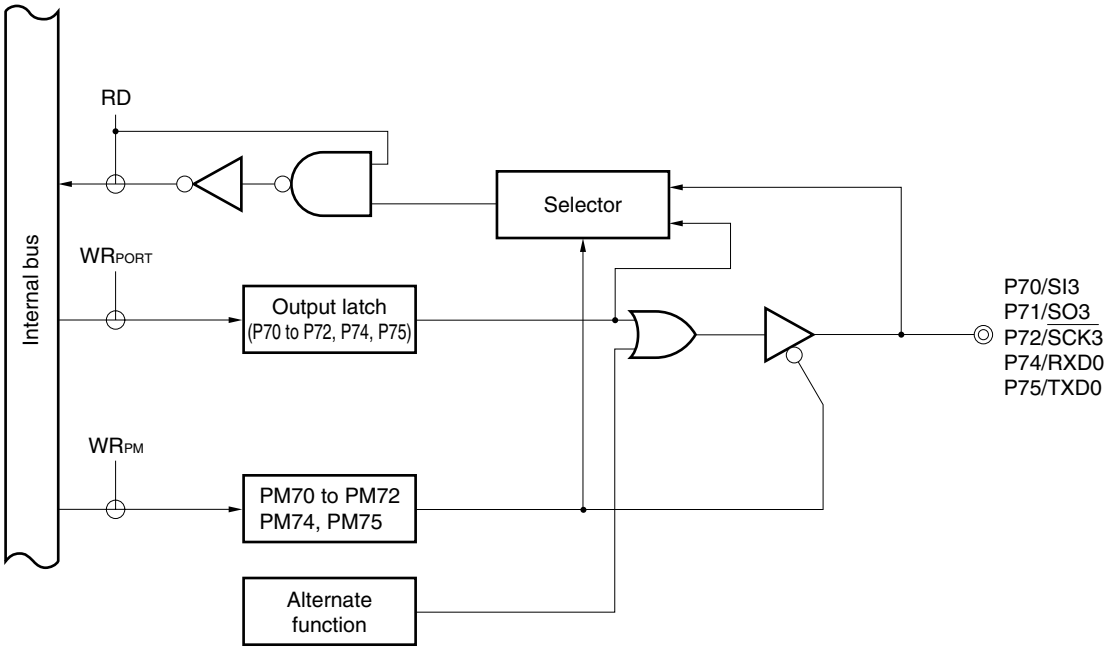
Alternate functions include serial interface data I/O, clock I/O, and asynchronous interface data I/O**Note**.

Reset input sets port 7 to the input mode.

Figures 4-10 and 4-11 show the block diagrams of port 7.

Note μPD178076, 178078, and 178F098 only

Figure 4-10. Block Diagram of P70 to P72, P74, and P75

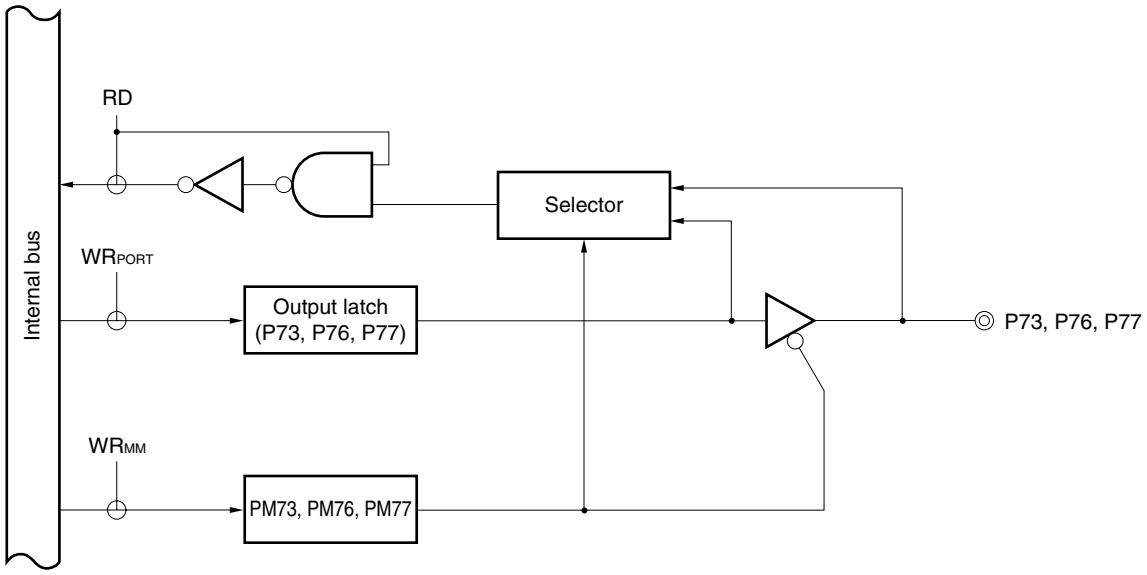


PM: Port mode register

RD: Port 7 read signal

WR: Port 7 write signal

Figure 4-11. Block Diagram of P73, P76, and P77



PM: Port mode register
RD: Port 7 read signal
WR: Port 7 write signal

4.2.9 Port 10

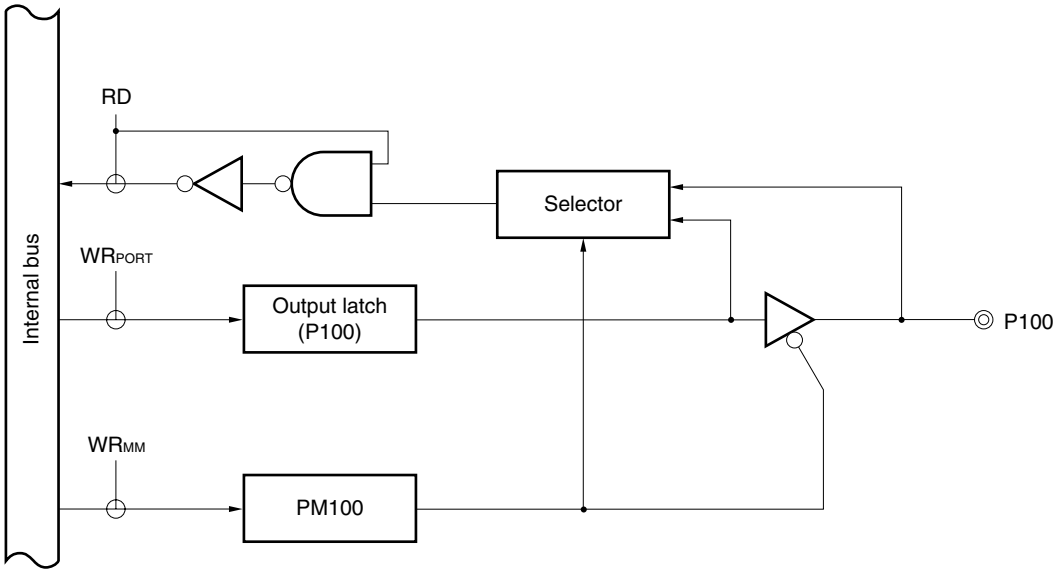
Port 10 is a 3-bit I/O port with an output latch. The P100 to P102 pins can be set to input mode/output mode in 1-bit units using port mode register 10 (PM10).

Alternate functions include AM intermediate frequency counter and FM intermediate frequency counter input.

Reset input sets port 10 to the input mode.

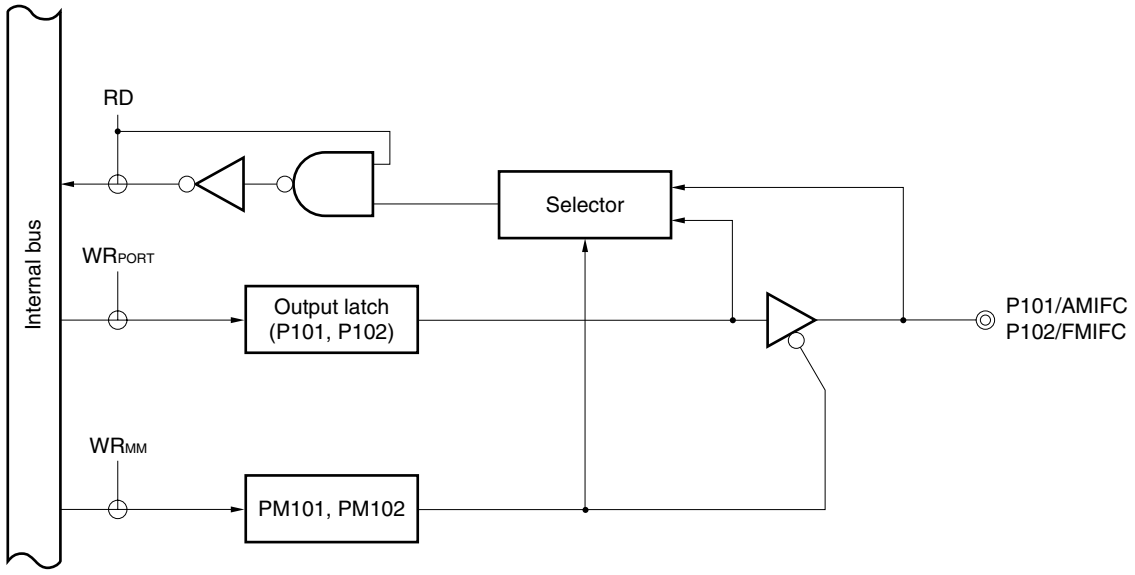
Figures 4-12 and 4-13 show the block diagrams of port 10.

Figure 4-12. Block Diagram of P100



PM: Port mode register
RD: Port 10 read signal
WR: Port 10 write signal

Figure 4-13. Block Diagram of P101 and P102



PM: Port mode register
RD: Port 10 read signal
WR: Port 10 write signal

4.2.10 Port 12

Port 12 is a 5-bit I/O port with an output latch. The P120 to P124 pins can be set to input mode/output mode in 1-bit units using port mode register 12 (PM12).

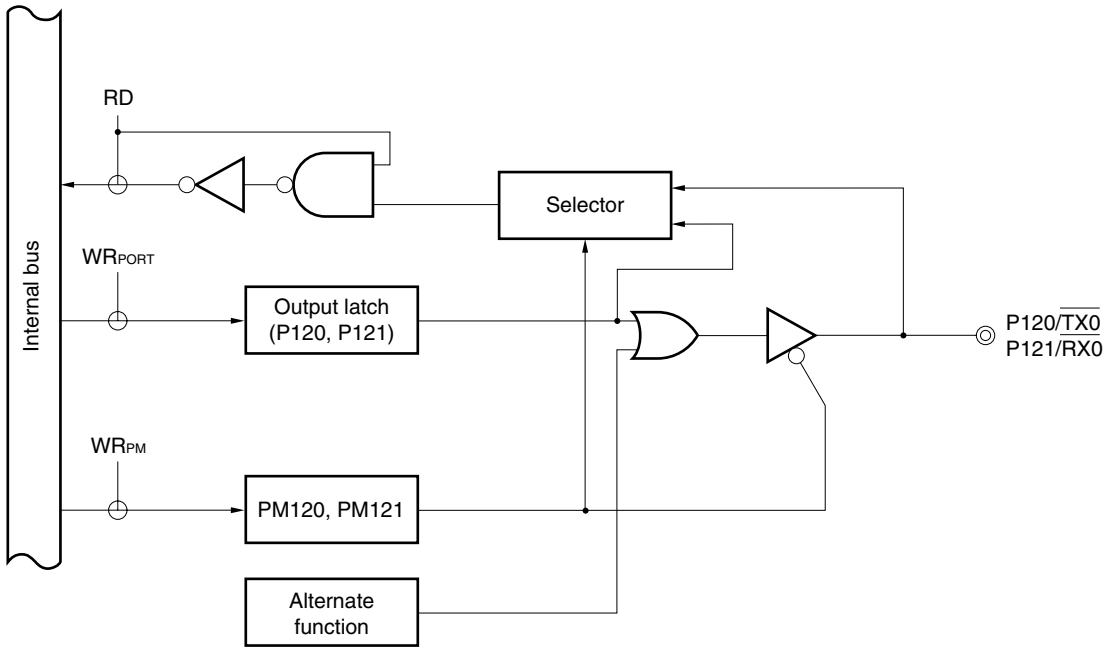
Alternate functions include data I/O[Note] of the IEBus controller.

Reset input sets port 12 to the input mode.

Figures 4-14 and 4-15 show the block diagrams of port 12.

Note μPD178096A, 178098A, and 178F098 only

Figure 4-14. Block Diagram of P120 and P121

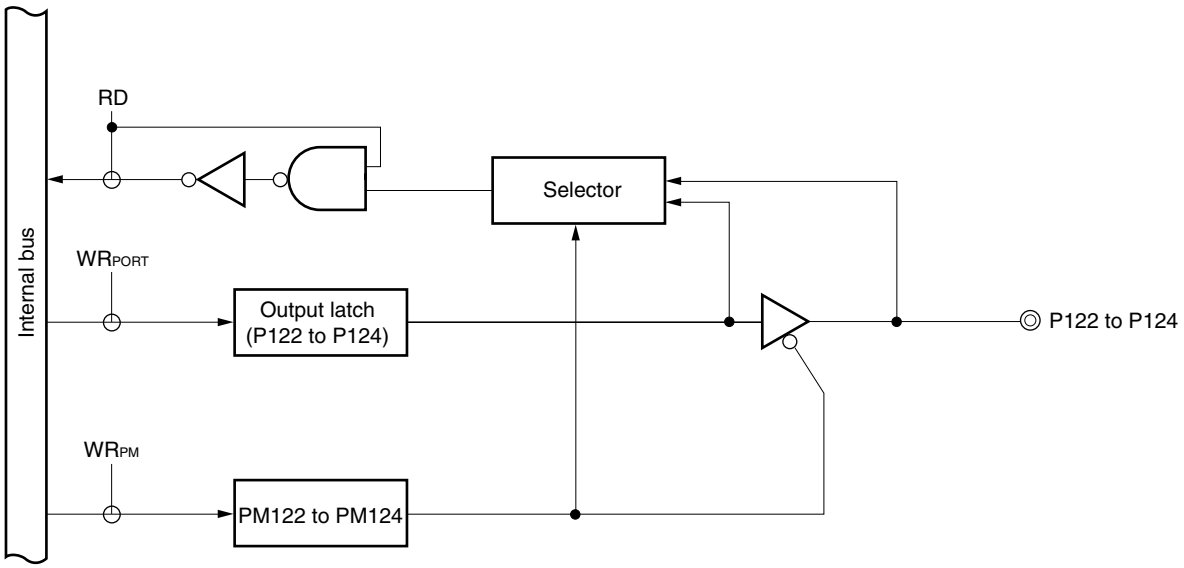


PM: Port mode register

RD: Port 12 read signal

WR: Port 12 write signal

Figure 4-15. Block Diagram of P122 to P124



PM: Port mode register
RD: Port 12 read signal
WR: Port 12 write signal

4.2.11 Port 13

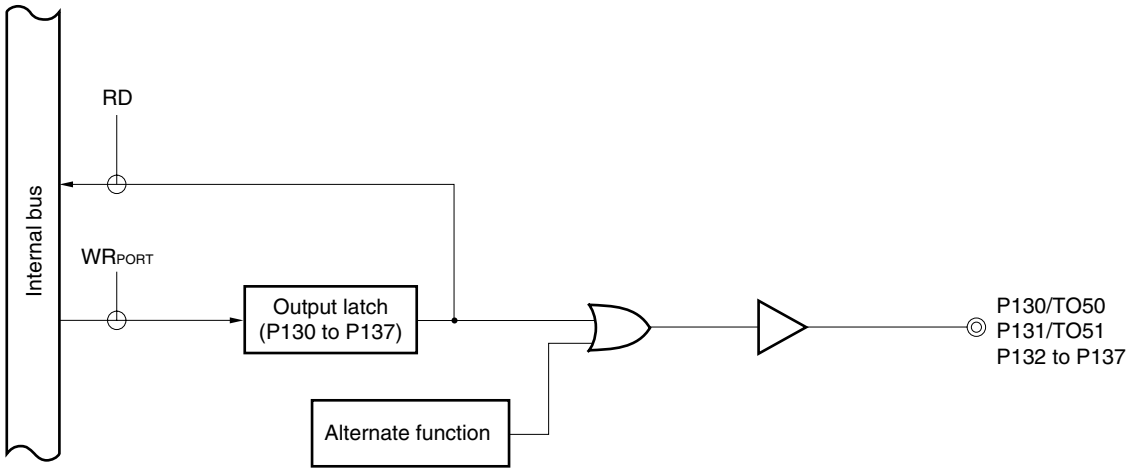
Port 13 is an 8-bit N-ch open-drain output port with an output latch.

Alternate functions include timer output.

Reset input sets port 13 to the general-purpose output port mode.

Figure 4-16 shows the block diagram of port 13.

Figure 4-16. Block Diagram of P130 to P137



RD: Port 13 read signal

WR: Port 13 write signal

4.3 Registers Controlling Port Function

Ports are controlled by the corresponding port mode registers (PM0, PM2 to PM7, PM10, and PM12). These registers set the corresponding ports to the input or output mode in 1-bit units.

These port mode registers can be set by using a 1-bit or 8-bit memory manipulation instruction.

Reset input sets the values of these registers to FFH.

When using the alternate-function pins of each port, set the corresponding port mode register and output latch as shown in Table 4-3.

Caution Because port 0 also serves as an external interrupt input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. When using the output mode, therefore, preset the interrupt mask flag to 1.

Remark The P10 to P17 pins are input-only pins and the P130 to P137 pins are output-only pins.

Table 4-3. Port Mode Register and Output Latch Settings When Using Alternate Functions

Pin Name	Alternate Function		PM××	P××
	Name	I/O		
P00 to P07	INTP0 to INTP7	Input	1	×
P20	SI1	Input	1	×
P21	SO1	Output	0	0
P22	$\overline{SCK1}$	Input	1	×
		Output	0	0
P23	STB	Output	0	0
P24	BUSY	Input	1	×
P30	VM45	Output	0	0
P31	TO0	Output	0	0
P32	TI00	Input	1	×
P33	TI01	Input	1	×
P34	TI50	Input	1	×
P35	TI51	Input	1	×
P36	BEEP0	Output	0	0
P37	BUZ	Output	0	0
P70	SI3	Input	1	×
P71	SO3	Output	0	0
P72	$\overline{SCK3}$	Input	1	×
		Output	0	0
P74	RXD0	Input	1	×
P75	TXD0	Output	0	0
P101	AMIFC	Input	1	×
P102	FMIFC	Input	1	×
P120	$\overline{TX0}$	Output	0	0
P121	$\overline{RX0}$	Input	1	×
P130	TO50	Output	—	0
P131	TO51	Output	—	0

Cautions 1. When using the above alternate-function pins as output pins, be sure to set the output latch (P××) to 0.

2. When P25 to P27 are used for the serial interface, the input/output mode or output latch must be set according to the function used. For the setting method, refer to Figure 13-5 Format of Serial Operating Mode Register 0 (CSIM0).

Remark ×: Don't care
PM××: Port mode register
P××: Output latch of port

Figure 4-17. Format of Port Mode Registers

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	1	FF20H	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	FF23H	FFH	R/W
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	FF24H	FFH	R/W
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	FF25H	FFH	R/W
PM6	PM67	PM66	PM65	PM64	PM63	PM62	PM61	PM60	FF26H	FFH	R/W
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70	FF27H	FFH	R/W
PM10	1	1	1	1	1	PM102	PM101	PM100	FF2AH	FFH	R/W
PM12	1	1	1	PM124	PM123	PM122	PM121	PM120	FF2CH	FFH	R/W

PMmn	Pmn pin input/output mode selection (m = 0, 2 to 7, 12 : n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

4.4 Port Function Operations

Port operations differ depending on whether the input or output mode is set, as shown below.

4.4.1 Writing to I/O port

(1) Output mode

A value is written to the output latch by a transfer instruction, and the output latch contents are output from the pin.

Once data is written to the output latch, it is held until data is written to the output latch again.

★ The data in the output latch is cleared after reset.

(2) Input mode

A value is written to the output latch by a transfer instruction, but since the output buffer is off, the pin status does not change.

Once data is written to the output latch, it is held until data is written to the output latch again.

4.4.2 Reading from I/O port

(1) Output mode

The output latch contents are read by a transfer instruction. The output latch contents do not change.

(2) Input mode

The pin status is read by a transfer instruction. The output latch contents do not change.

4.4.3 Operations on I/O port

(1) Output mode

An operation is performed on the output latch contents, and the result is written to the output latch. The output latch contents are output from the pins.

Once data is written to the output latch, it is held until data is written to the output latch again.

★ The data in the output latch is cleared after reset.

(2) Input mode

The output latch contents are undefined, but since the output buffer is off, the pin status does not change.

Caution In the case of 1-bit memory manipulation instruction, although a single bit is manipulated, the port is accessed as an 8-bit unit. Therefore, for a port with a mixture of input and output pins, the output latch contents for pins specified as input are undefined, even for bits other than the manipulated bit.

CHAPTER 5 CLOCK GENERATOR

5.1 Clock Generator Functions

The clock generator generates the clock to be supplied to the CPU and peripheral hardware. The system clock oscillator oscillates at frequencies of 6.3 MHz[Note]. Oscillation can be stopped by executing the STOP instruction or setting the processor clock control register (PCC).

Note In addition to the 6.3 MHz crystal resonator, a 4.5 MHz crystal resonator can also be connected to the μPD178078 and 178098A Subseries. When using the system clock at a frequency of 4.5 MHz, set bit 0 (DTSCK0) of the DTS system clock select register (DTSCK) to 1. Set the DTSCK0 flag after power application and reset by the $\overline{\text{RESET}}$ pin, and before using the basic timer, buzzer output controller (BEEP0), PLL frequency synthesizer, and frequency counter.
When using the IEBus controller of the μPD178096A, 178098A, and 178F098, however, be sure to use the 6.3 MHz crystal resonator. At this time, it is not necessary to set the DTSCK0 flag.
The timing of the basic timer, buzzer output controller (BEEP0), PLL frequency synthesizer, and frequency counter described in **8.3 Operation of Basic Timer**, **10.3.1 (1) BEEP frequency select register 0 (BEEPCL0)**, **19.3 (2) PLL reference mode register (PLLRF)**, and **20.3 (1) IF counter mode select register (IFCMD)** is not changed.

Figure 5-1. Format of DTS System Clock Select Register (DTSCK)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DTSCK	0	0	0	0	0	0	0	DTSCK0	FFAAH	00H	R/W

DTSCK0	System clock selection
0	6.3 MHz
1	4.5 MHz

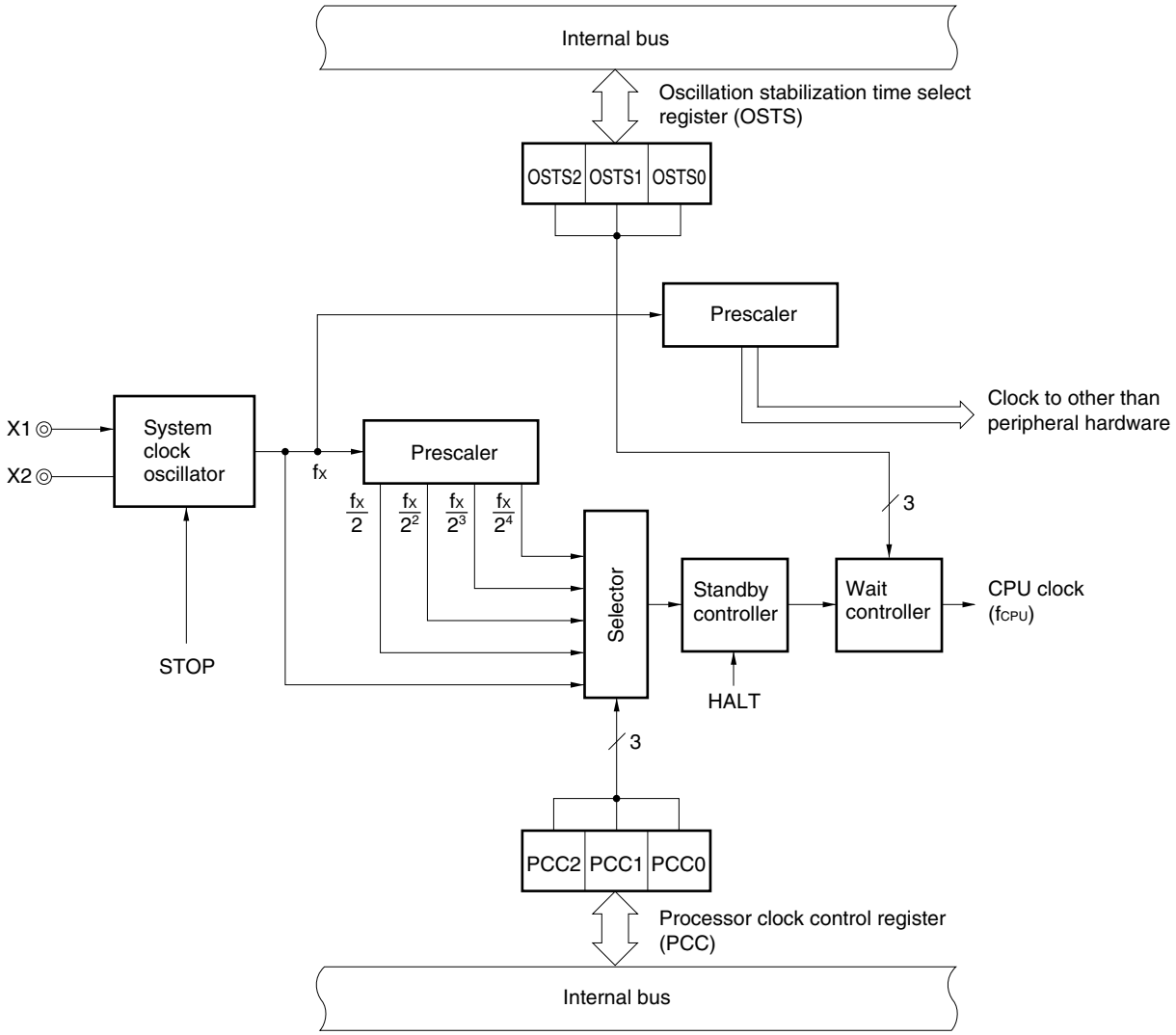
5.2 Clock Generator Configuration

The clock generator consists of the following hardware.

Table 5-1. Configuration of Clock Generator

Item	Configuration
Control registers	Processor clock control register (PCC) Oscillation stabilization time select register (OSTS)
Oscillator	System clock oscillator

Figure 5-2. Block Diagram of Clock Generator



5.3 Clock Generator Control Registers

The clock generator is controlled by the following two registers.

- Processor clock control register (PCC)
- Oscillation stabilization time select register (OSTS)

(1) Processor clock control register (PCC)

The clock generator is controlled by the processor clock control register (PCC).

PCC sets the CPU clock.

PCC is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets PCC to 04H.

Figure 5-3. Format of Processor Clock Control Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PCC	0	0	0	0	0	PCC2	PCC1	PCC0	FFFBH	04H	R/W[Note]

R/W

PCC2	PCC1	PCC0	CPU clock (f_{CPU}) selection	Minimum instruction execution time: $2/f_{CPU}$
				f_X = 6.3 MHz operation
0	0	0	f_X	0.32 μs
0	0	1	$f_X/2$	0.64 μs
0	1	0	$f_X/2^2$	1.27 μs
0	1	1	$f_X/2^3$	2.54 μs
1	0	0	$f_X/2^4$	5.08 μs
Other than above			Setting prohibited	

Note Bits 3 to 7 are read-only.

Remark f_X: System clock oscillation frequency

(2) Oscillation stabilization time select register (OSTS)

This register is used to select the time required for oscillation to stabilize after the $\overline{\text{RESET}}$ signal has been input or the STOP mode has been released.

This register is set by an 8-bit memory manipulation instruction.

Reset input set OSTS to 04H. It therefore takes $2^{17}/f_X$ to release the STOP mode by $\overline{\text{RESET}}$ input.

Figure 5-4. Format of Oscillation Stabilization Time Select Register (OSTS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection
0	0	0	$2^{12}/f_X$ (650 μs)
0	0	1	$2^{14}/f_X$ (2.60 ms)
0	1	0	$2^{15}/f_X$ (5.20 ms)
0	1	1	$2^{16}/f_X$ (10.4 ms)
1	0	0	$2^{17}/f_X$ (20.8 ms)
Other than above			Setting prohibited

Remarks 1. f_X: System clock oscillation frequency

2. ( ): f_X = 6.3 MHz

5.4 System Clock Oscillator

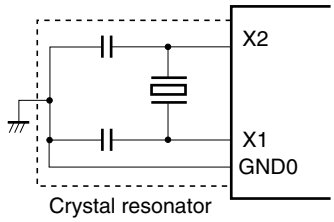
5.4.1 System clock oscillator

The system clock oscillator oscillates with a crystal resonator (6.3 MHz or 4.5 MHz) connected to the X1 and X2 pins.

Figure 5-5 shows an external circuit of the system clock oscillator.

Figure 5-5. External Circuit of System Clock Oscillator

Crystal oscillation



Caution When using the system clock oscillator, wire as follows in the area enclosed by the broken lines in Figure 5-5 to avoid an adverse effect from wiring capacitance.

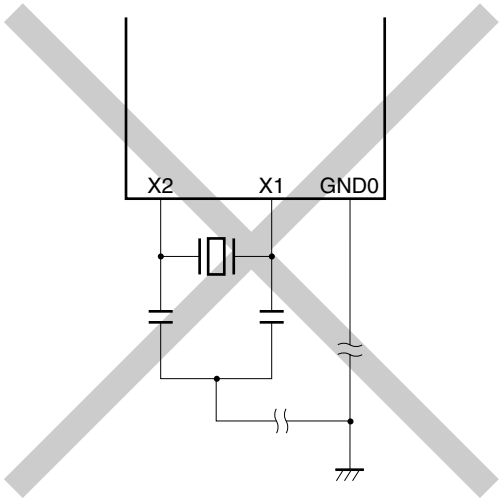
- **Keep the wiring length as short as possible.**
- **Do not cross the wiring to with other signal lines. Do not route wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as GND. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

5.4.2 Incorrect resonator connection

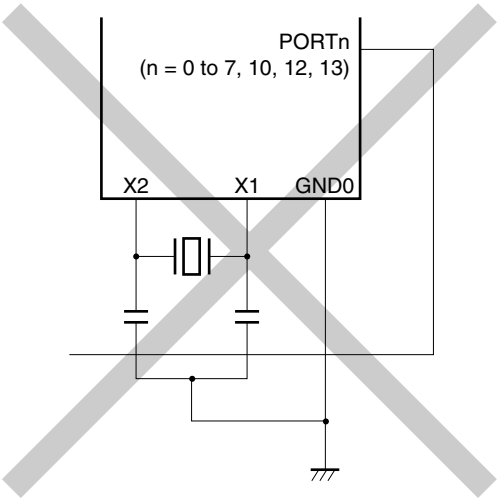
Figure 5-6 shows examples of Incorrect resonator connection.

Figure 5-6. Examples of Incorrect Connection Resonator (1/2)

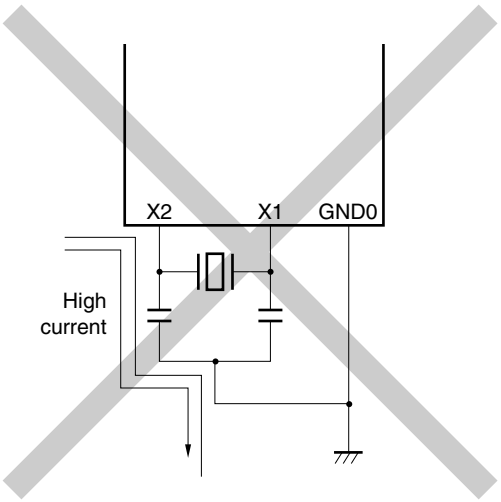
(a) Too long wiring



(b) Crossed signal line



(c) Wiring near high fluctuating current



(d) Current flowing through ground line of oscillator (potential at points A, B, and C fluctuates)

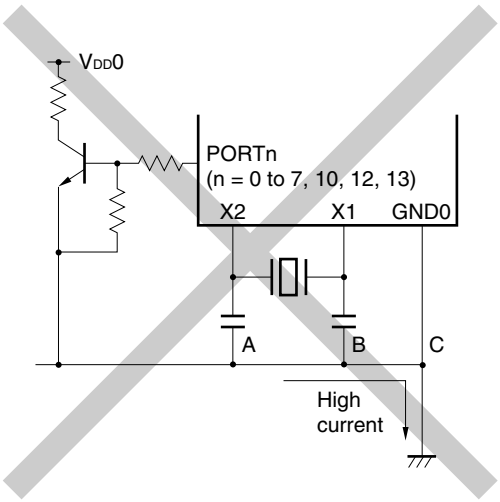
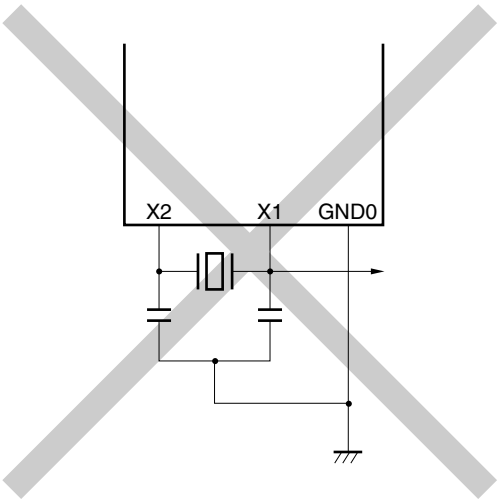


Figure 5-6. Examples of Incorrect Connection Resonator (2/2)

(e) Signals are fetched



5.4.3 Divider

The divider divides the system clock oscillator output (f_X) and generates various clocks.

5.5 Clock Generator Operations

The clock generator generates the following clocks and controls the CPU operating mode including the standby mode.

- System clock f_X
- CPU clock f_{CPU}
- Clock to peripheral hardware

The following clock generator functions and operations are determined by the processor clock control register (PCC).

(a) Upon generation of the $\overline{\text{RESET}}$ signal, the lowest speed mode of the system clock (5.08 μs when operated at 6.3 MHz) is selected (PCC = 04H). System clock oscillation stops while a low level is being applied to the $\overline{\text{RESET}}$ pin.

(b) One of the five minimum instruction execution times (0.317, 0.635, 1.27, 2.54, 5.08 μs at 6.3 MHz) can be selected by setting PCC.

(c) Two standby modes, STOP and HALT, are available.

(d) The system clock is divided and supplied to the peripheral hardware. The peripheral hardware also stops if the system clock is stopped.

5.6 Changing System Clock and CPU Clock Settings

5.6.1 Time required for switchover between system clock and CPU clock

The system clock and CPU clock can be switched over by setting bits 0 to 2 (PCC0 to PCC2) of the processor clock control register (PCC).

The actual switchover operation is not performed directly after writing to the PCC; the operation continues on the pre-switchover clock for several instructions (refer to Table 5-2).

Table 5-2. Maximum Time Required for CPU Clock Switchover

Set Values Before Switchover			Set Values After Switchover														
PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0
			0	0	0	0	0	1	0	1	0	0	1	1	1	0	0
0	0	0				16 instructions			16 instructions			16 instructions			16 instructions		
0	0	1	8 instructions						8 instructions			8 instructions			8 instructions		
0	1	0	4 instructions			4 instructions						4 instructions			4 instructions		
0	1	1	2 instructions			2 instructions			2 instructions						2 instructions		
1	0	0	1 instruction			1 instruction			1 instruction			1 instruction					

Remark One instruction is the minimum instruction execution time with the pre-switchover CPU clock.

CHAPTER 6 16-BIT TIMER/EVENT COUNTER 0

6.1 Functions of 16-Bit Timer/Event Counter 0

16-bit timer/event counter 0 has the following functions.

(1) Interval timer

16-bit timer/event counter 0 generates interrupt requests at the preset time interval.

- Number of counts: 2 to 65536

(2) External event counter

16-bit timer/event counter 0 can measure the number of pulses with a high-/low-level width in a signal input externally.

- Valid level pulse width: 16/f_X or more

(3) Pulse width measurement

16-bit timer/event counter 0 can measure the pulse width of an externally input signal.

- Valid level pulse width: 2/f_X or more

(4) Square-wave output

16-bit timer/event counter 0 can output a square wave with any selected frequency.

- Cycle: $(2 \times 2$ to $65536 \times 2) \times$ count clock cycle

(5) One-shot pulse output

16-bit timer/event counter 0 can output a one-shot pulse for which any output pulse width can be set.

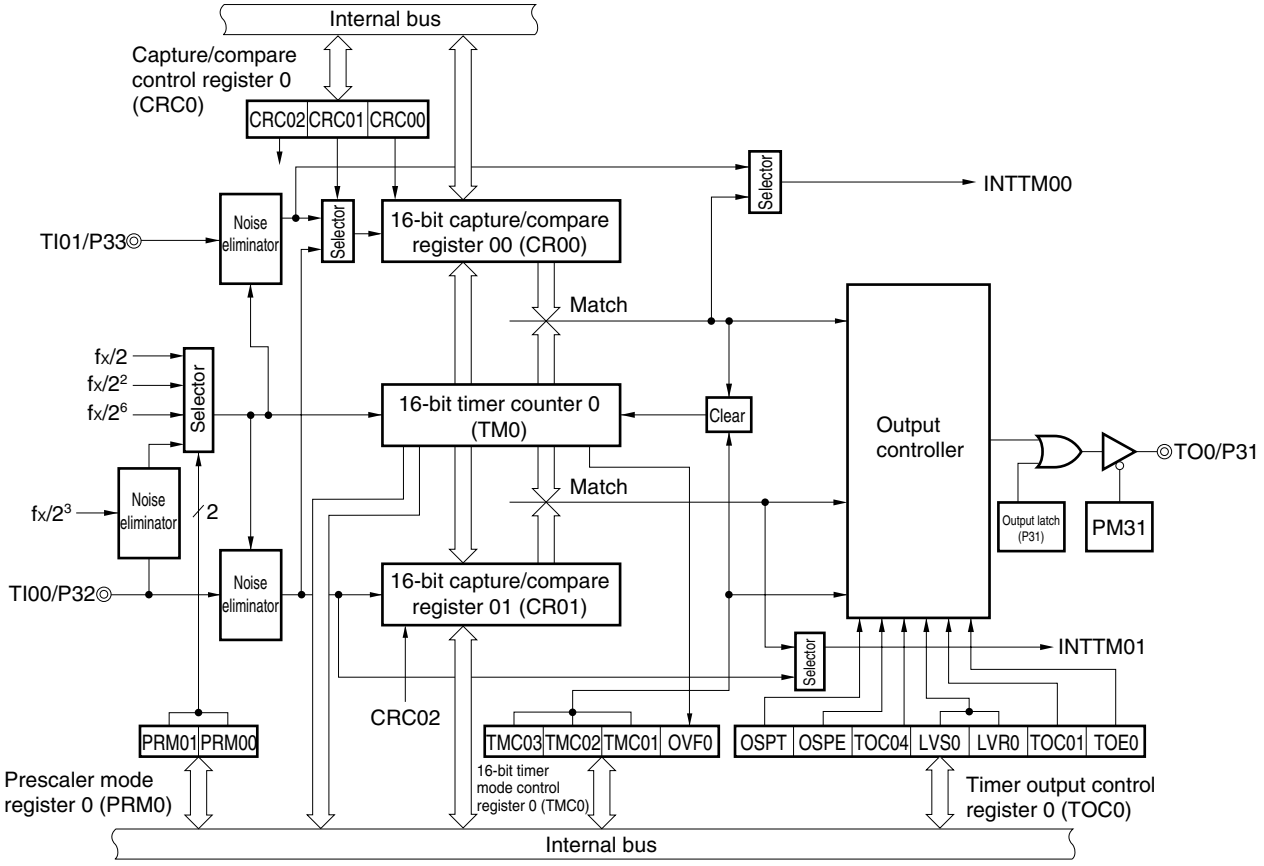
(6) PPG output

16-bit timer/event counter 0 can output a square wave that has an arbitrary cycle and pulse width.

- $2 <$ Pulse width $<$ Cycle $\leq$ (FFFF + 1) H

Figure 6-1 shows the block diagram.

Figure 6-1. Block Diagram of 16-Bit Timer/Event Counter 0



6.2 Configuration of 16-Bit Timer/Event Counter 0

16-bit timer/event counter 0 consists of the following hardware.

Table 6-1. Configuration of 16-Bit Timer/Event Counter 0

Item	Configuration
Timer counter	16-bit timer counter 0 (TM0)
Register	16-bit capture/compare registers 00 and 01 (CR00 and CR01)
Timer input	TI00, TI01
Timer output	TO0
Control registers	• 16-bit timer mode control register 0 (TMC0) • Capture/compare control register 0 (CRC0) • 16-bit timer output control register 0 (TOC0) • Prescaler mode register 0 (PRM0) • Port mode register 3 (PM3) • Port 3 (P3)

(1) 16-bit timer counter 0 (TM0)

TM0 is a 16-bit read-only register that counts count pulses.

The count value is incremented at the rising edge of the count clock. If the count value is read while the register is operating, input of the count clock is temporarily stopped, and the count value at that point is read.

The count value is reset to 0000H in the following cases.

<1> When the $\overline{\text{RESET}}$ signal is input

<2> When TMC03 and TMC02 are cleared.

<3> When the valid TI00 edge is input in the clear & start mode

<4> When TM0 and CR00 match in the clear & start mode

<5> When OSPT is set or the valid TI00 edge is input in the one-shot pulse output mode

(2) 16-bit capture/compare register 00 (CR00)

CR00 is a 16-bit register with both the functions of a capture register and a compare register. Whether this register is used as a capture register or a compare register is specified by bit 0 (CRC00) of capture/compare control register 0.

- **When CR00 is used as compare register**

  The value set to CR00 is always compared with the count value of 16-bit timer counter 0 (TM0). When the two values match, an interrupt request (INTTM00) is generated. When TM0 is used as an interval timer, CR00 is also used to hold the interval time.

- **When CR00 is used as capture register**

  The valid edge of the TI00/P32 pin or the TI01/P33 pin can be selected as the capture trigger. The valid edge of TI00 and TI01 is specified by prescaler mode register 0 (PRM0) (Refer to Table 6-2).

Table 6-2. CR00 Capture Trigger and Valid Edges of TI00 and TI01 Pins

(1) TI00 pin valid edge selected as capture trigger (CRC01 = 1, CRC00 = 1)

CR00 Capture Trigger	TI00 Pin Valid Edge		
		ES01	ES00
Falling edge	Rising edge	0	1
Rising edge	Falling edge	0	0
No capture operation	Both rising and falling edges	1	1

(2) TI01 pin valid edge selected as capture trigger (CRC01 = 0, CRC00 = 1)

CR00 Capture Trigger	TI01 Pin Valid Edge		
		ES11	ES10
Falling edge	Falling edge	0	0
Rising edge	Rising edge	0	1
Both rising and falling edges	Both rising and falling edges	1	1

Remarks 1. Setting ES01, ES00 = 1, 0 and ES11, ES10 = 1, 0 is prohibited.

2. ES01, ES00: Bits 5 and 4 of prescaler mode register 0 (PRM0)
ES11, ES10: Bits 7 and 6 of prescaler mode register 0 (PRM0)
CRC01, CRC00: Bits 1 and 0 of capture/compare control register 0 (CRC0)

CR00 is set by a 16-bit memory manipulation instruction.
CR00 is undefined after reset.

Cautions 1. Set CR00 to a value other than 0000H in the clear & start mode entered on a match between TM0 and CR00. However, in the free-running mode and in the clear mode using the valid edge of TI00, if CR00 is set to 0000H, an interrupt request (INTTM00) is generated when CR00n changes from 0000H to 0001H following overflow (FFFFH).

★ **2. If the new value of CR00 is less than the value of 16-bit timer counter 0 (TM0), TM0 continues counting, overflows, and then starts counting from 0 again. If the new value of CR00 is less than the old value, therefore, the timer must be reset and restarted after the value of CR00 is changed.**

★ **3. CR00 does not generate an interrupt request if it is captured at the valid edge of the TI00 pin.**

★ **4. Do not select the TI00 valid edge as the count clock when using TI00 as a capture trigger.**

(3) 16-bit capture/compare register 01 (CR01)

This is a 16-bit register with both the functions of a capture register and a compare register. Whether CR01 is used as a capture register or a compare register is specified by using bit 2 (CRC02) of capture/compare control register 0.

- **When CR01 is used as compare register**

  The value set to CR01 is always compared with the count value of 16-bit timer counter 0 (TM0). When the two values match, an interrupt request (INTTM01) is generated.

- **When CR01 is used as capture register**

  The valid edge of the TI00/P32 pin can be selected as the capture trigger. The valid edge of TI00/P32 is specified by using prescaler mode register 0 (PRM0) (refer to Table 6-3).

★ **Table 6-3. CR01 Capture Trigger and Valid Edge of TI00 Pin (CRC02 = 1)**

CR01 Capture Trigger	TI00 Pin Valid Edge		
		ES01	ES00
Falling edge	Falling edge	0	0
Rising edge	Rising edge	0	1
Both rising and falling edges	Both rising and falling edges	1	1

Remarks 1. Setting ES01, ES00 = 1, 0 is prohibited.

2. ES01, ES00: Bits 5 and 4 of prescaler mode register 0 (PRM0)
CRC02: Bit 2 of capture/compare control register 0 (CRC0)

CR01 is set by using a 16-bit memory manipulation instruction.
CR01 is undefined after reset.

Cautions 1. Set CR01 to a value other than 0000H in the clear & start mode entered on a match between TM0 and CR00. However, in the free-running mode and in the clear mode using the valid edge of TI00, if CR01 is set to 0000H, an interrupt request (INTTM01) is generated when CR01 changes from 0000H to 0001H following overflow (FFFFH).

★ **2. When using CR01 as a capture register, do not select the TI00 valid edge as the count clock.**

6.3 Registers Controlling 16-Bit Timer/Event Counter 0

The following six registers control 16-bit timer/event counter 0.

- 16-bit timer mode control register 0 (TMC0)
- Capture/compare control register 0 (CRC0)
- 16-bit timer output control register 0 (TOC0)
- Prescaler mode register 0 (PRM0)
- Port mode register 3 (PM3)
- Port 3 (P3)

(1) 16-bit timer mode control register 0 (TMC0)

This register specifies the operating mode of the 16-bit timer, clear mode of 16-bit timer counter 0 (TM0), and output timing, and detects an overflow.

TMC0 is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is cleared to 00H after reset.

Caution 16-bit timer counter 0 (TM0) starts operating when TMC02 and TMC03 are set to values other than 0, 0 (operation stop mode). To stop operation, reset TMC02 and TMC03 to 0, 0.

Figure 6-2. Format of 16-Bit Timer Mode Control Register 0 (TMC0)

Symbol	7	6	5	4	3	2	1	<0>	Address	After reset	R/W
TMC0	0	0	0	0	TMC03	TMC02	TMC01	OVF0	FF78H	00H	R/W

TMC03	TMC02	TMC01	Selection of operating mode and clear mode	Selection of TO0 output timing	Generation of interrupt request
0	0	0	Operation stops (TM0 is cleared to 0).	Not affected	Not generated
0	0	1			
0	1	0	Free-running mode	Match between TM0 and CR00 or CR01	Generated on match between TM0 and CR00 or CR01
0	1	1		Match between TM0 and CR00 or CR01, or valid edge of TI00	
1	0	0	Clear & start at valid edge of TI00	Match between TM0 and CR00 or CR01	
1	0	1		Match between TM0 and CR00 or CR01, or valid edge of TI00	
1	1	0	Clear & start on match between TM0 and CR00	Match between TM0 and CR00 or CR01	
1	1	1		Match between TM0 and CR00 or CR01, or valid edge of TI00	

OVF0	Detection of overflow of 16-bit timer counter 0
0	Overflow
1	No overflow

Cautions 1. Write the bits other than the OVF0 flag after stopping the timer operation.

2. The valid edge of the TI00/P32 pin is selected by prescaler mode register 0 (PRM0).

3. If a mode in which TM0 is cleared and started on a match between TM0 and CR00 is selected, and if the value of TM0 changes from FFFFH to 0000H with CR00 set to FFFFH, the OVF0 flag is set to 1.

Remark TO0: Output pin of 16-bit timer/event counter 0
TI00: Input pin of 16-bit timer/event counter 0
TM0: 16-bit timer counter 0
CR00: 16-bit capture/compare register 00
CR01: 16-bit capture/compare register 01

(2) Capture/compare control register 0 (CRC0)

This register controls the operations of the 16-bit capture/compare registers (CR00 and CR01).

CRC0 is set by using a 1-bit or 8-bit memory manipulation instruction.

This register is cleared to 00H after reset.

Figure 6-3. Format of Capture/Compare Control Register 0 (CRC0)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
CRC0	0	0	0	0	0	CRC02	CRC01	CRC00	FF7CH	00H	R/W

CRC02	Selection of operating mode of CR01
0	Compare register
1	Capture register

CRC01	Selection of capture trigger of CR00
0	Captured at valid edge of TI01
1	Captured at reverse edge to valid edge of TI00**Note**

CRC00	Selection of operating mode of CR00
0	Compare register
1	Capture register

Note If both the rising and falling edges are selected as the valid edges of TI00, CR00 does not perform a capture operation.

Cautions 1. Be sure to set CRC0 after stopping the timer operation.

2. Do not specify CR00 as a capture register when a mode in which TM0 is cleared and started on a match between TM0 and CR00 is selected by 16-bit timer mode control register 0 (TMC0).

(3) 16-bit timer output control register 0 (TOC0)

This register controls the operation of the 16-bit timer/event counter output controller. It sets/resets the R-S flip-flop (LV0); enables/disables output inversion, timer output of 16-bit timer/event counter 0, and the one-shot pulse output operation; and sets the output trigger of the one-shot pulse.

TOC0 is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is cleared to 00H after reset.

Figure 6-4 shows the format of TOC0.

Figure 6-4. Format of 16-Bit Timer Output Control Register 0 (TOC0)

Symbol	7	<6>	<5>	4	<3>	<2>	1	<0>	Address	After reset	R/W
TOC0	0	OSPT	OSPE	TOC04	LVS0	LVR0	TOC01	TOE0	FF7EH	00H	R/W

OSPT	Control of output trigger of one-shot pulse by software
0	No one-shot pulse trigger
1	One-shot pulse trigger

OSPE	Control of one-shot pulse output operation
0	Successive pulse output
1	One-shot pulse output**Note**

TOC04	Control of timer output F/F on match between CR01 and TM0
0	Inversion operation disabled
1	Inversion operation enabled

LVS0	LVR0	Setting of timer output F/F status of 16-bit timer/event counter 0
0	0	Not affected
0	1	Timer output F/F reset (0)
1	0	Timer output F/F set (1)
1	1	Setting prohibited

TOC01	Control of timer output F/F on match between CR00 and TM0
0	Inversion disabled
1	Inversion enabled

TOE0	Control of output of 16-bit timer/event counter 0
0	Output disabled (output is fixed to 0 level)
1	Output enabled

Note One-shot pulse output operates normally only in the free-running mode and the mode in which TM0 is cleared and started at the valid edge of TI00.

Cautions 1. Be sure to set TOC0 after stopping the timer.

2. LVS0 and LVR0 are 0 when they are read.

3. OSPT is always 0 when read because it is automatically cleared after data has been set.

(4) Prescaler mode register 0 (PRM0)

This register selects the count clock of 16-bit timer counter 0 (TM0) and the valid edges of the TI00 and TI01 input pins. PRM0 is set by an 8-bit memory manipulation instruction.

The value of this register is cleared to 00H after reset.

Figure 6-5. Format of Prescaler Mode Register 0 (PRM0)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PRM0	ES11	ES10	ES01	ES00	0	0	PRM01	RPM00	FF7AH	00H	R/W

ES11	ES10	Selection of valid edge of TI01
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES01	ES00	Selection of valid edge of TI00
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

PRM01	PRM00	Selection of count clock
0	0	$f_X/2$ (3.15 MHz)
0	1	$f_X/2^2$ (1.58 MHz)
1	0	$f_X/2^6$ (98.4 kHz)
1	1	Valid edge of TI00

Cautions 1. Be sure to set data to PRM0 after stopping the timer operation.

2. When the valid edge of TI00 is set as the count clock, do not set the mode in which TM0 is cleared and started at the valid edge of TI00, and do not set T100 as the capture trigger.

3. The capture trigger must be a pulse wider than two pulses of the selected count clock to ensure capturing. Similarly, the external clock must have a pulse wider than two internal clocks ($f_X/2^3$).

4. If the TI00 pin or TI01 pin goes high immediately after system reset, the rising edge is detected immediately after TM0 has been enabled to operate. Keep this in mind when the pin is pulled up. However, when TM0 is enabled to operate again after it has been stopped, the rising edge is not detected.

★

Remarks 1. f_X: System clock oscillation frequency

2. TI00, TI01: Input pins of 16-bit timer/event counter 0

3. ( ): f_X = 6.3 MHz

(5) Port mode register 3 (PM3)

This register sets port 3 I/O in 1-bit units.

When using the P31/TO0 pin for timer output, set PM31 and the output latch of P31 to 0.

When using the P32/TI00 pin as a timer input (TI00), set PM32 to 1.

When using the P33/TI01 pin as a timer input (TI01), set PM33 to 1.

PM3 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM3 to FFH.

Figure 6-6. Format of Port Mode Register 3 (PM3)

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

PM3n	Selection of P3n pin I/O mode (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

6.4 Operation of 16-Bit Timer/Event Counter 0

6.4.1 Operation as interval timer

16-bit timer/event counter 0 operates as an interval timer when 16-bit timer mode control register 0 (TMC0) and capture/compare control register 0 (CRC0) are set as shown in Figure 6-7. An interrupt request is repeatedly generated at intervals specified by the count value preset to 16-bit capture/compare register 00 (CR00).

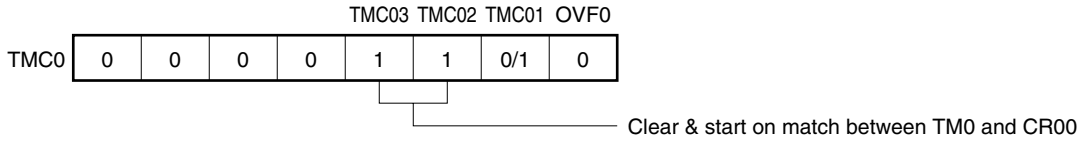
When the count value of 16-bit timer counter 0 (TM0) matches the set value of CR00, the value of TM0 is cleared to 0 and an interrupt request signal (INTTM00) is generated.

The count clock of 16-bit timer/event counter 0 is selected by bits 0 and 1 (PRM00 and PRM01) of prescaler mode register 0 (PRM0).

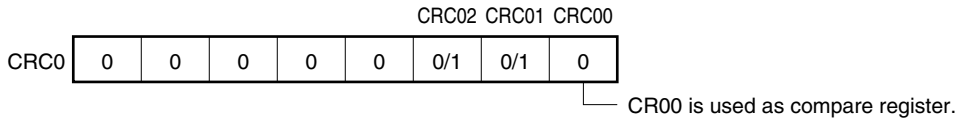
For details of the operation when the value of the compare register is changed during timer count operation, refer to **(3)** in **6.6 Notes on 16-Bit Timer/Event Counter 0**.

Figure 6-7. Setting of Control Registers for Interval Timer Operation

(a) 16-bit timer mode control register 0 (TMC0)

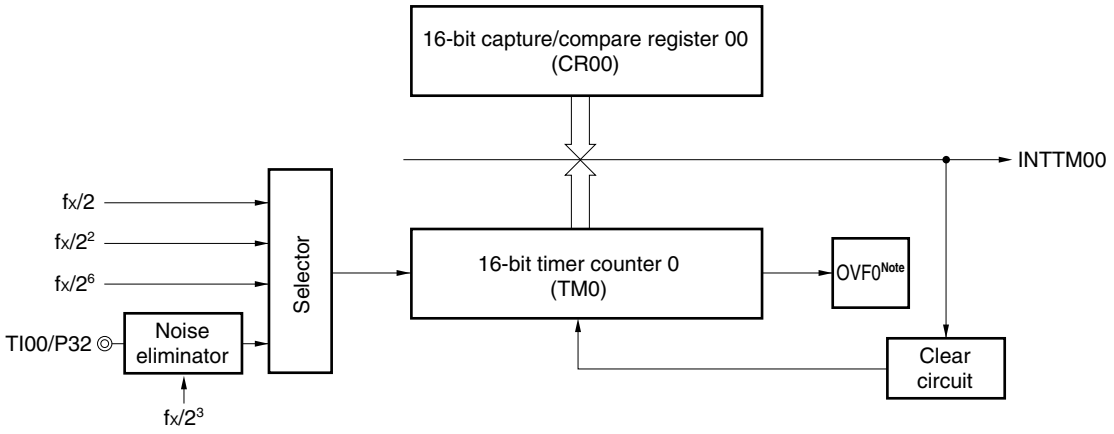


(b) Capture/compare control register 0 (CRC0)



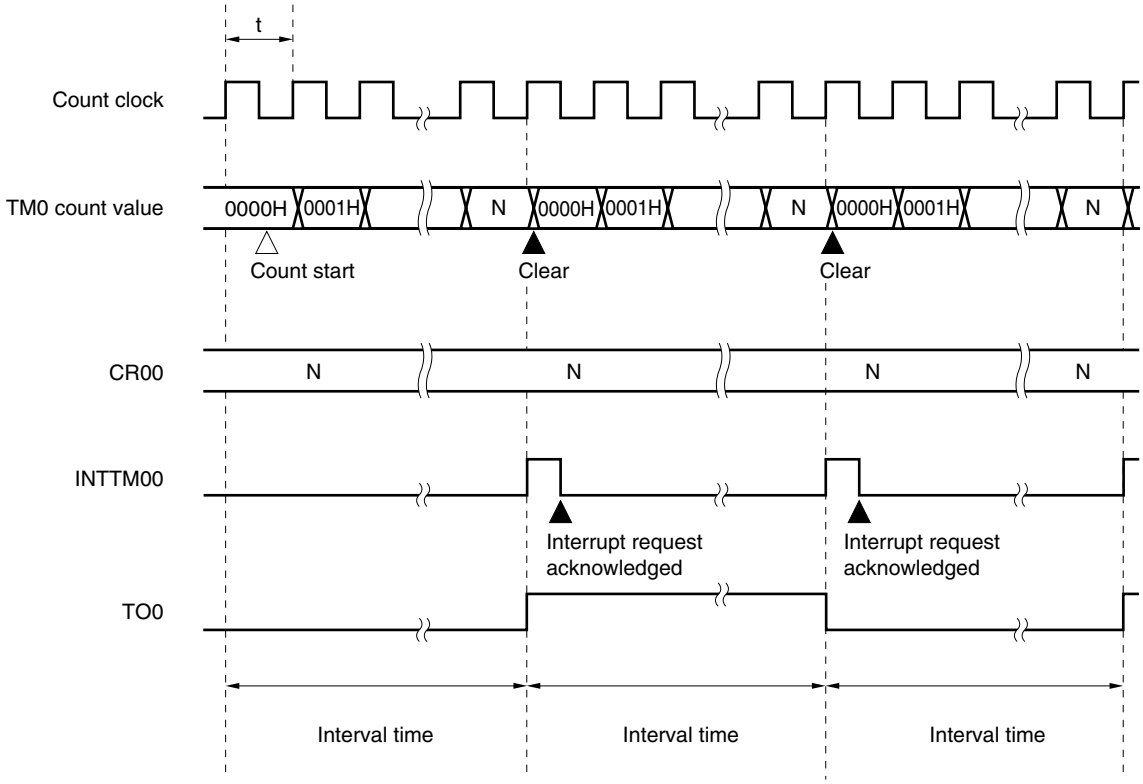
Remark 0/1: When these bits are set to 1 or reset to 0, other functions can be used at the same time as the interval timer function. For details, refer to Figures 6-2 and 6-3.

★ **Figure 6-8. Configuration of Interval Timer**



Note OVF0 is 1 only when CR00 = FFFFH in the interval timer.

★ **Figure 6-9. Timing of Interval Timer Operation**



Remark Interval time = (N + 1) × t
N = 0001H to FFFFH

6.4.2 Operation as external event counter

The external event counter counts the number of external clock pulses input to the TI00/P32 pin by using 16-bit timer counter 0 (TM0).

Each time the valid edge specified by prescaler mode register 0 (PRM0) has been input to TI00/P32, the value of TM0 is incremented.

When the count value of TM0 matches the value of capture/compare register 00 (CR00), TM0 is cleared to 0 and an interrupt request signal (INTTM00) is generated.

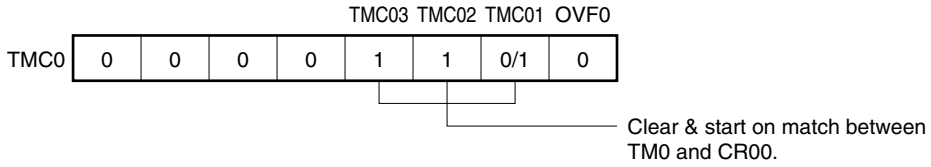
Set CR00 to a value other than 0000H (one pulse cannot be counted).

The rising, falling, or both rising and falling edges can be selected as the valid edge of TI00/P32 by using bits 4 and 5 (ES00 and ES01) of prescaler mode register 0 (PRM0).

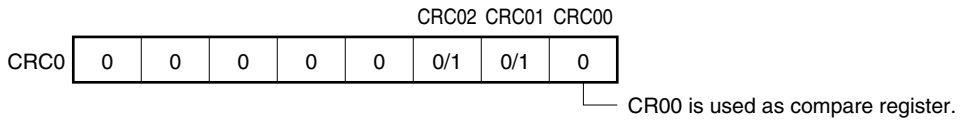
Because the operation is performed only after the valid level of the TI00 pin is detected twice by sampling using the internal clock ($f_X/2^3$), noise with a short pulse width can be eliminated.

Figure 6-10. Setting of Control Registers in External Event Counter Mode

(a) 16-bit timer mode control register 0 (TMC0)

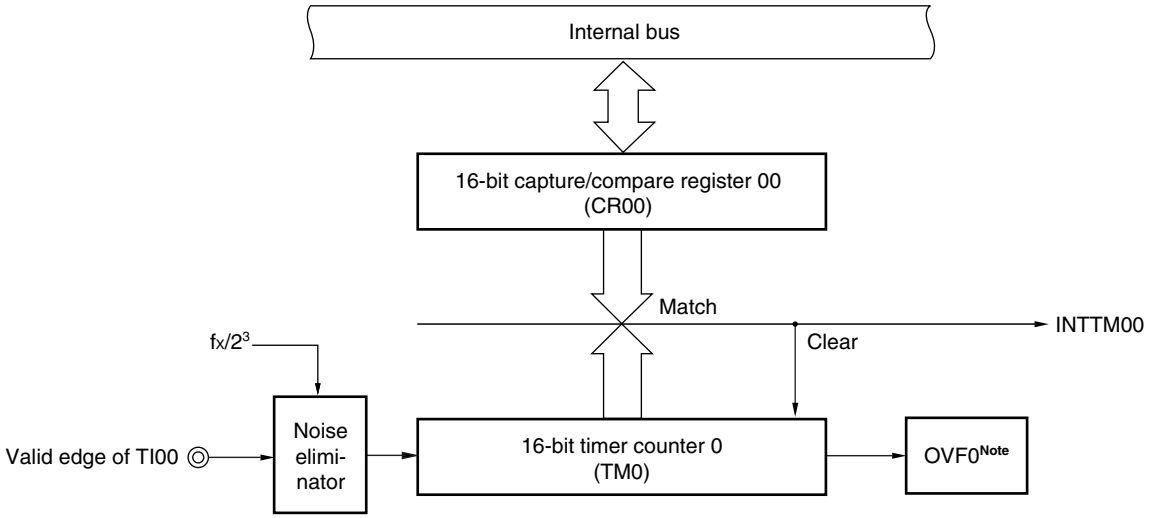


(b) Capture/compare control register 0 (CRC0)



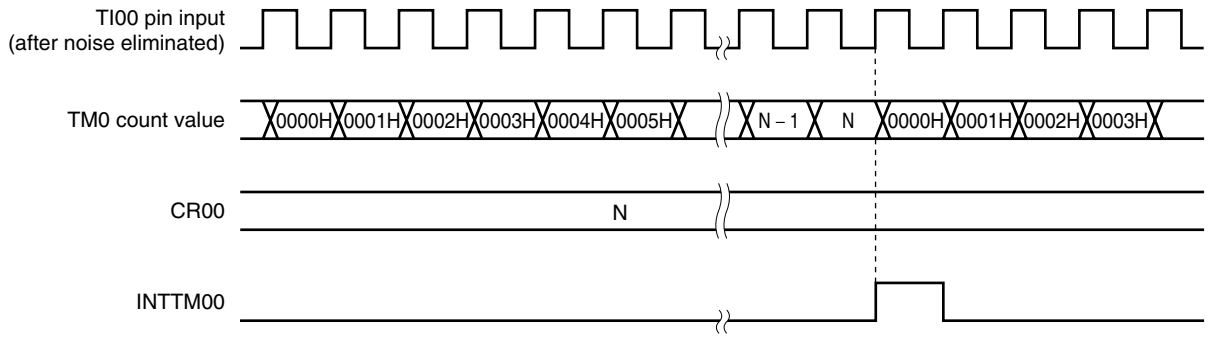
Remark 0/1: When these bits are reset to 0 or set to 1, other functions can be used at the same time as the external event counter function. For details, refer to Figures 6-2 and 6-3.

★ **Figure 6-11. Configuration of External Event Counter**



Note OVF0 is 1 only when CR00 = FFFFH in the external event counter.

Figure 6-12. Timing of External Event Counter (with Rising Edge Specified)



Caution Read TM0 to read the count value of the external event counter.

6.4.3 Pulse width measurement

16-bit timer counter 0 can be used to measure the pulse width of the signal input to the TI00/P32 and TI01/P33 pins.

The pulse width can be measured by operating TM0 in the free-running mode, or by restarting the timer in synchronization with the edge of the signal input to the TI00/P32 pin.

(1) Pulse width measurement with free-running counter and one capture register

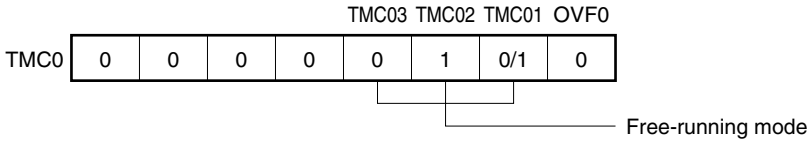
When 16-bit timer counter 0 (TM0) operates in the free-running mode (refer to the register setting in Figure 6-13) and if the edge specified by prescaler mode register 0 (PRM0) is input to the TI00/P32 pin, the value of TM0 is loaded to 16-bit capture/compare register 01 (CR01) and an external interrupt request signal (INTTM01) is set.

The rising, falling, or both the rising and falling edges can be selected as the edge by using bits 6 and 7 (ES10 and ES11) of PRM0.

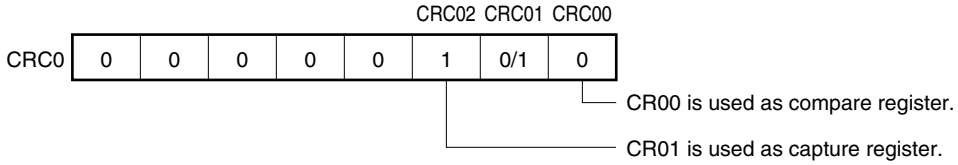
Because the value of TM0 is captured only after the valid level of the TI00 pin is detected twice by sampling using the count clock cycle selected by PRM0, noise with a short pulse width can be eliminated.

Figure 6-13. Setting of Control Registers for Pulse Width Measurement with Free-Running Counter and One Capture Register

(a) 16-bit timer mode control register 0 (TMC0)

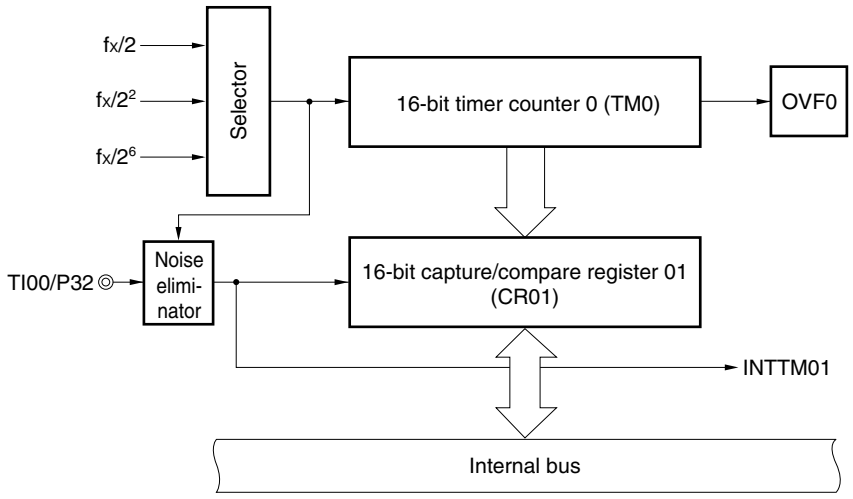


(b) Capture/compare control register 0 (CRC0)

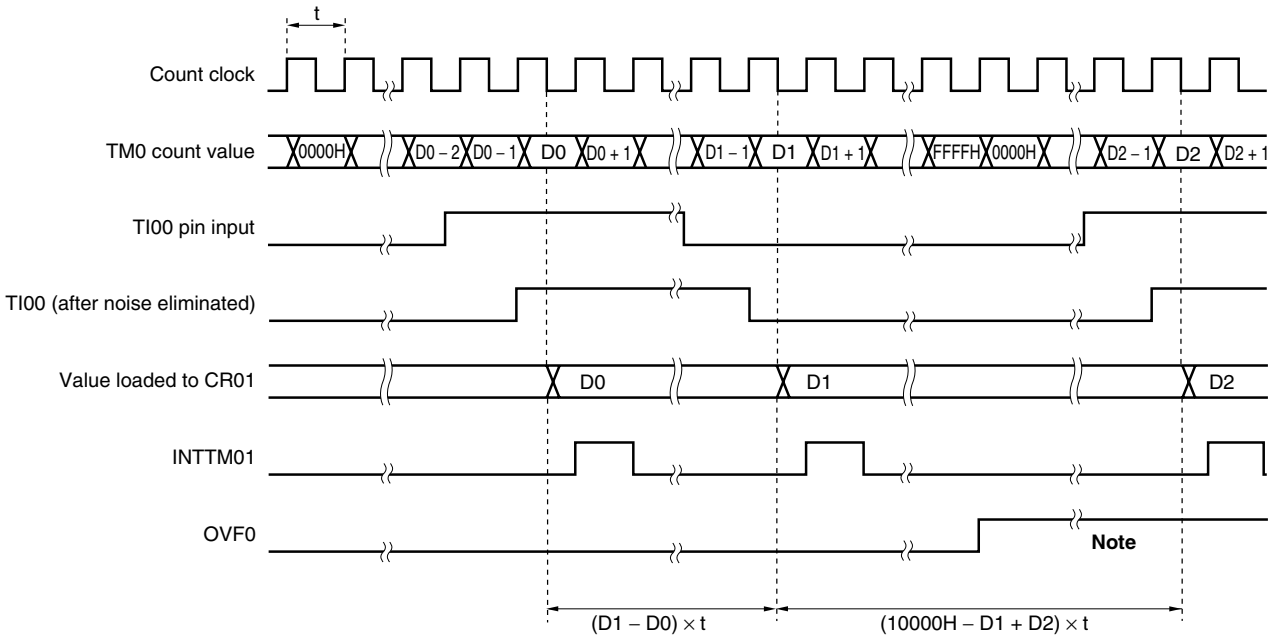


Remark 0/1: When these bits are set to 1 or reset to 0, other functions can be used at the same time as the pulse width measurement function. For details, refer to Figures 6-2 and 6-3.

★ **Figure 6-14. Configuration of Pulse Width Measurement Circuit with Free-Running Counter**



★ **Figure 6-15. Timing of Pulse Width Measurement with Free-Running Counter and One Capture Register (with Both Rising and Falling Edges Specified)**



Note Clear OVF0 by software.

(2) Measurement of two pulse widths with free-running counter

When 16-bit timer counter 0 (TM0) operates in the free-running mode (refer to Figure 6-16), the pulse widths of the two signals input to the TI00/P32 and TI01/P33 pins can be simultaneously measured.

When the edge specified by bits 4 and 5 (ES00 and ES01) of prescaler mode register 0 (PRM0) is input to the TI00/P32 pin, the value of TM0 is captured to 16-bit capture/compare register 01 (CR01), and an external interrupt request signal (INTTM01) is set.

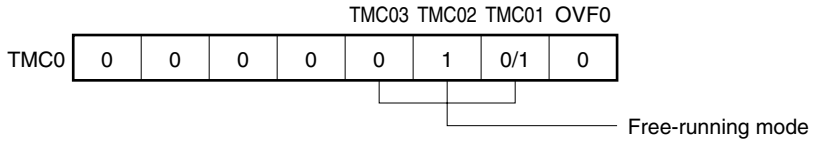
When the edge specified by bits 6 and 7 (ES10 and ES11) of PRM0 is input to the TI01/P33 pin, the value of TM0 is captured to 16-bit capture/compare register 00 (CR00), and an external interrupt request signal (INTTM00) is set.

The edges of the TI00/P32 and TI01/P33 pins are specified by bits 4 and 5 (ES00 and ES01), and bits 6 and 7 (ES10 and ES11) of PRM0, respectively. The rising, falling, or both the rising and falling edges can be specified.

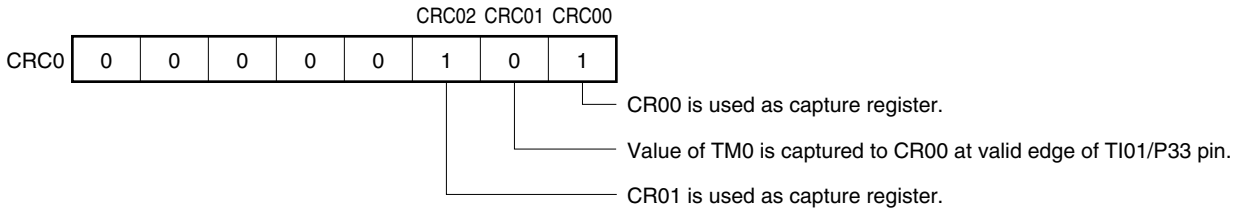
Because the value of TM0 is captured only after the valid level of the TI00 pin is detected twice by sampling using the count clock cycle selected by prescaler mode register 0 (PRM0), noise with a short pulse width can be eliminated.

Figure 6-16. Setting of Control Registers for Measurement of Two Pulse Widths with Free-Running Counter

(a) 16-bit timer mode control register 0 (TMC0)



(b) Capture/compare control register 0 (CRC0)

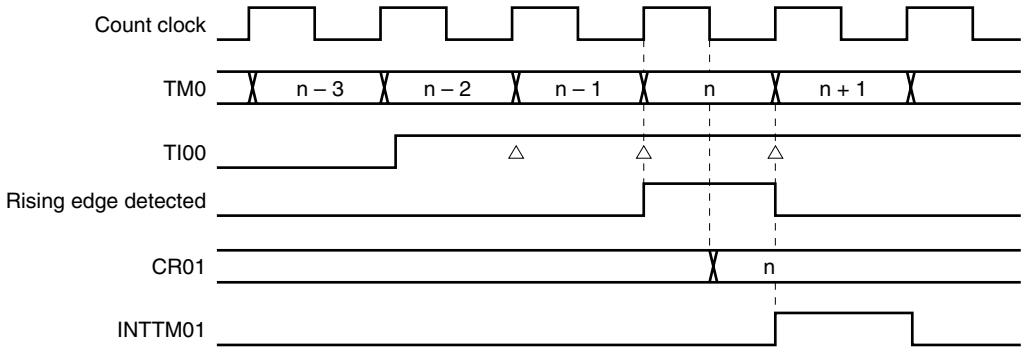


Remark 0/1: When these bits are set to 1 or reset to 0, other functions can be used at the same time as the pulse width measurement function. For details, refer to Figures 6-2 and 6-3.

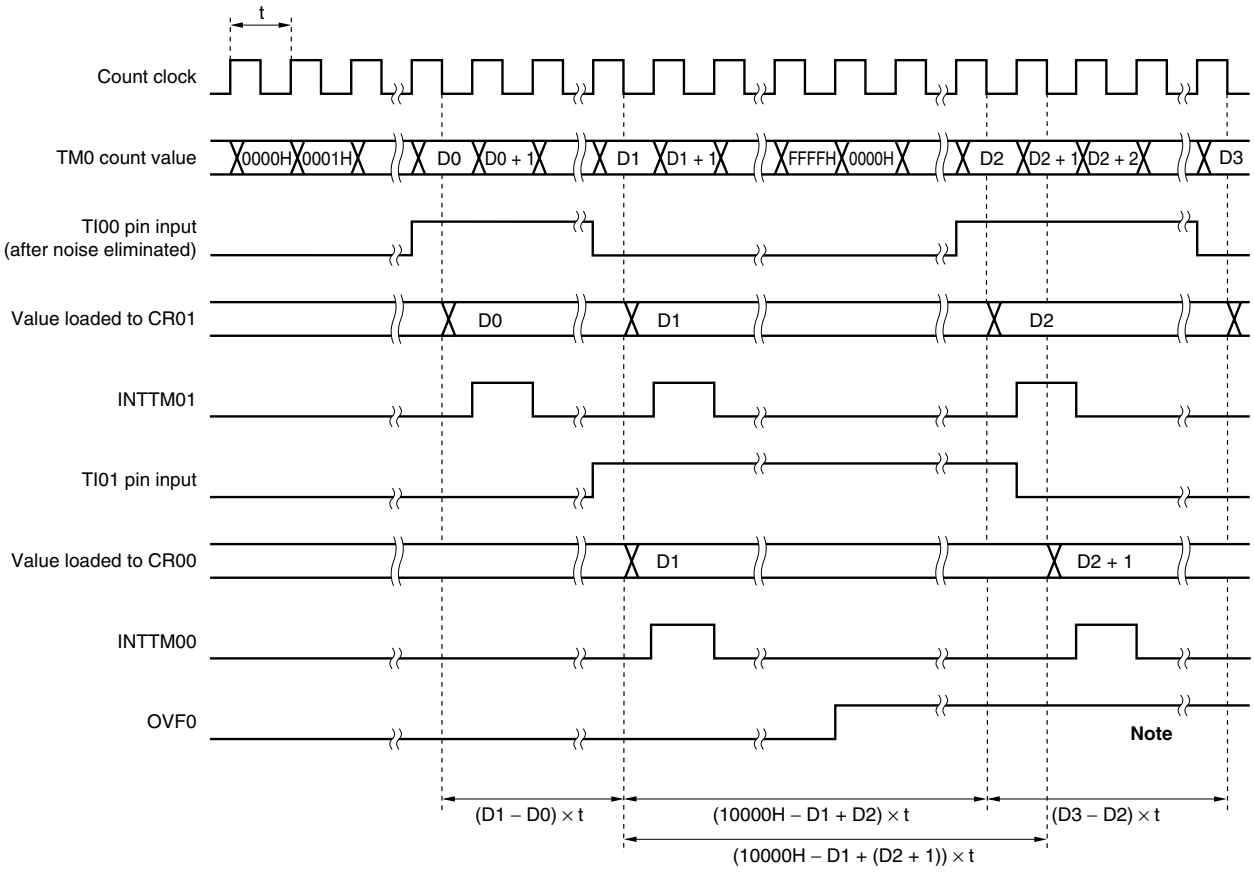
- **About capture operation (free-running mode)**

  The following charts show the operation of the capture register when the capture trigger is input.

Figure 6-17. Capture Operation of CR01 When Rising Edge Is Specified



★ **Figure 6-18. Timing of Pulse Width Measurement by Free-Running Counter (with Both Rising and Falling Edges Specified)**



Note Clear OVF0 by software.

(3) Pulse width measurement with free-running counter and two capture registers

When 16-bit timer counter 0 (TM0) operates in the free-running mode (refer to Figure 6-19), the pulse widths of the signals input to the TI00/P32 pin can be measured.

When the edge specified by bits 4 and 5 (ES00 and ES01) of prescaler mode register 0 (PRM0) is input to the TI00/P32 pin, the value of TM0 is captured to 16-bit capture/compare register 01 (CR01), and an external interrupt request signal (INTTM01) is set.

The value of TM0 is also captured to 16-bit capture/compare register 00 (CR00) when the reverse edge to the edge that triggers capturing to the CR01 is input to the pin.

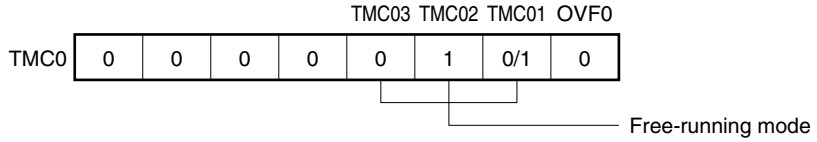
The edge of the TI00/P32 pin is specified by bits 4 and 5 (ES00 and ES01) of prescaler mode register 0 (PRM0). The rising or falling edge can be specified.

Because the value of TM0 is captured only after the valid level of the TI00 pin is detected twice by sampling using the count clock cycle selected by prescaler mode register 0 (PRM0), noise with a short pulse width can be eliminated.

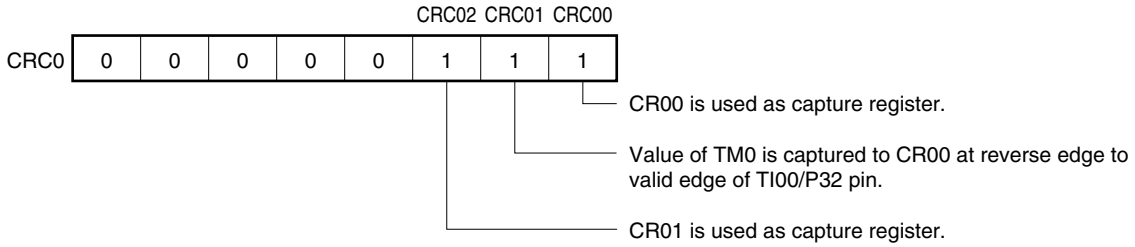
Caution When both the rising and falling edges are specified as the valid edges of the TI00/P32 pin, 16-bit capture/compare register 00 (CR00) cannot perform a capture operation.

Figure 6-19. Setting of Control Registers for Measurement with Free-Running Counter and Two Capture Registers

(a) 16-bit timer mode control register 0 (TMC0)

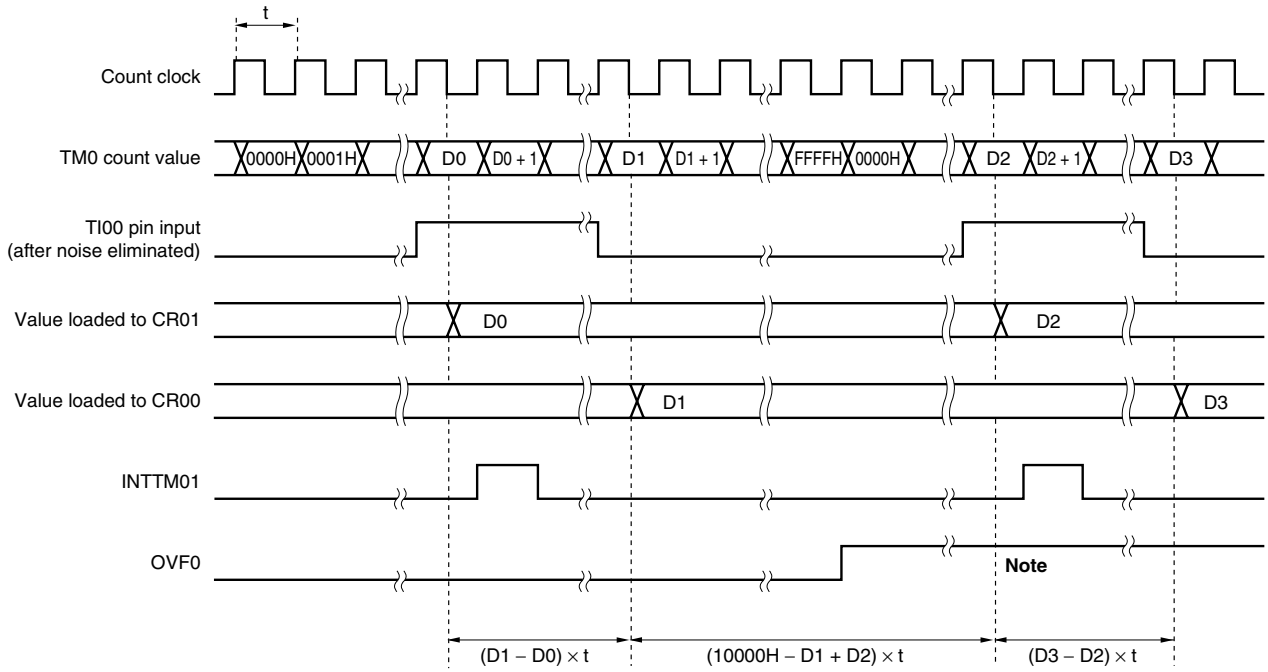


(b) Capture/compare control register 0 (CRC0)



Remark 0/1: When these bits are set to 1 or reset to 0, other functions can be used at the same time as the pulse width measurement function. For details, refer to the descriptions of each control register.

★ **Figure 6-20. Timing of Pulse Width Measurement with Free-Running Counter and Two Capture Registers (with Rising Edge Specified)**



Note Clear OVF0 by software.

(4) Pulse width measurement by restarting

When the valid edge of the TI00/P32 pin is detected, the pulse width of the signal input to the TI00/P32 pin can be measured by clearing 16-bit timer counter 0 (TM0) and restarting counting after the count value of TM0 is captured to 16-bit capture/compare register 01 (CR01) (refer to Figure 6-22).

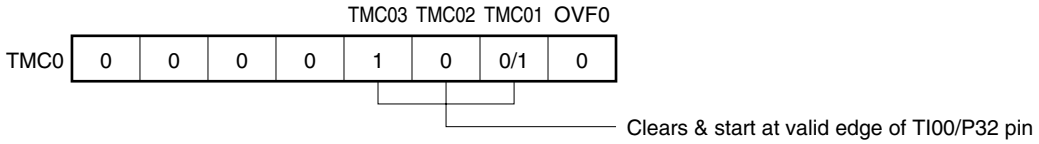
The rising or falling edge can be specified as the valid edge by using bit 4 and 5 (ES00 and ES01) of prescaler mode register 0 (PRM0).

Because the value of TM0 is captured only after the valid level of the TI00 pin is detected twice by sampling using the count clock cycle selected by prescaler mode register 0 (PRM0), noise with a short pulse width can be eliminated.

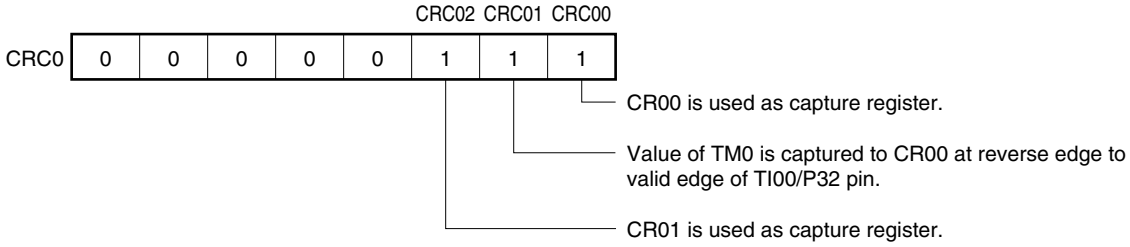
Caution When both the rising and falling edges are specified as the valid edges of the TI00/P32 pin, 16-bit capture/compare register 00 (CR00) cannot perform a capture operation.

Figure 6-21. Setting of Control Registers for Pulse Measurement by Restarting

(a) 16-bit timer mode control register 0 (TMC0)

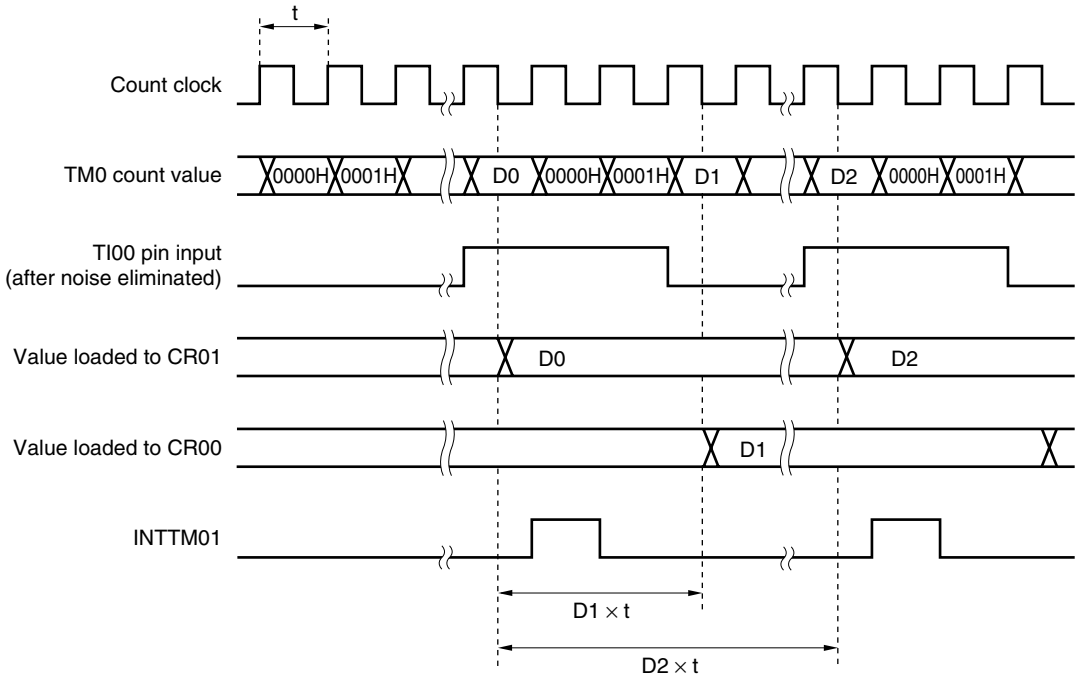


(b) Capture/compare control register 0 (CRC0)



Remark 0/1: When these bits are set to 1 or reset to 0, other functions can be used at the same time as the pulse width measurement function. For details, refer to Figures 6-2 and 6-3.

Figure 6-22. Timing of Pulse Width Measurement by Restarting (with Rising Edge Specified)



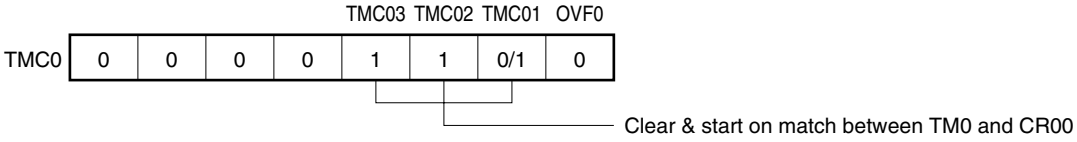
6.4.4 Square wave output operation

The 16-bit timer/event counter can be used to output a square wave of any frequency at intervals specified by the count value preset to 16-bit capture/compare register 00 (CR00).

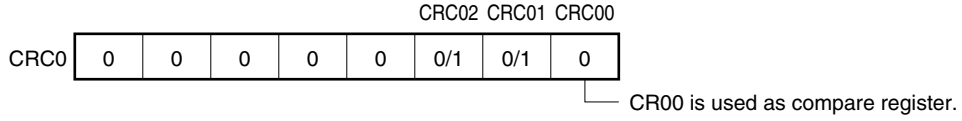
When bits 0 (TOE0) and 1 (TOC01) of 16-bit timer output control register 0 (TOC0) are set to 1, the output status of the TO0 pin is inverted at the intervals specified by the count value preset to CR00. In this way, a square wave of any frequency can be output.

Figure 6-23. Setting of Control Register in Square Wave Output Mode

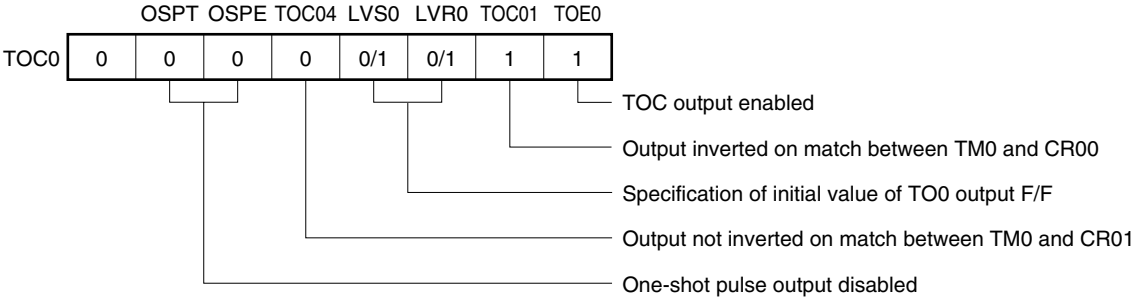
(a) 16-bit timer mode control register 0 (TMC0)



(b) Capture/compare control register 0 (CRC0)

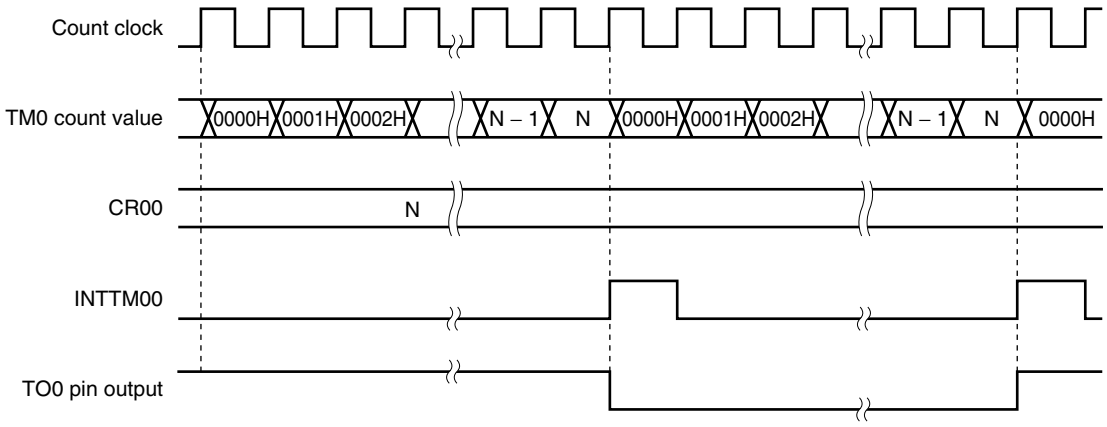


(c) 16-bit timer output control register 0 (TOC0)



Remark 0/1: When these bits are reset to 0 or set to 1, other functions can be used at the same time as the square wave output function. For details, refer to Figures 6-2, 6-3, and 6-4.

Figure 6-24. Timing of Square Wave Output Operation



6.4.5 One-shot pulse output operation

The 16-bit timer/event counter can be used to output a one-shot pulse in synchronization with a software trigger or external trigger (TI00/P32 pin input).

(1) One-shot pulse output with software trigger

A one-shot pulse can be output to the TO0/P31 pin by setting 16-bit timer mode control register 0 (TMC0), capture/compare control register 0 (CRC0), and 16-bit timer output control register 0 (TOC0) as shown in Figure 6-25, and setting bit 6 (OSPT) of TOC0 to 1 by software.

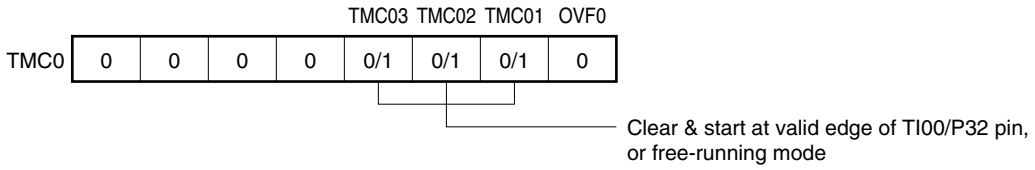
When OSPT is set to 1, 16-bit timer counter 0 (TM0) is cleared and started, and the output to TI00/P32 becomes active at the count value preset to 16-bit capture/compare register 01 (CR01). After that, the output becomes inactive at the count value preset to 16-bit capture/compare register 00 (CR00).

After outputting the one-shot pulse, TM0 continues operating. To stop TM0, set TM0 to 00H.

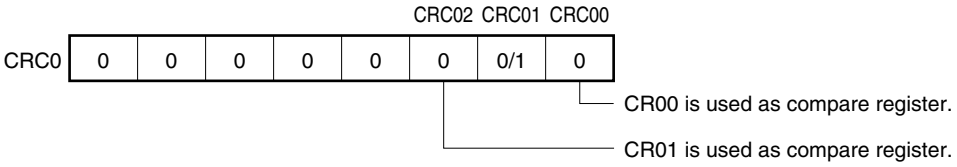
Caution Do not set OSPT to 1 while the one-shot pulse is being output. To output the one-shot pulse again, wait until the INTTM00 interrupt , which occurs on a match between TM0 and CR00, has occurred.

Figure 6-25. Setting of Control Registers for One-Shot Pulse Output Operation with Software Trigger

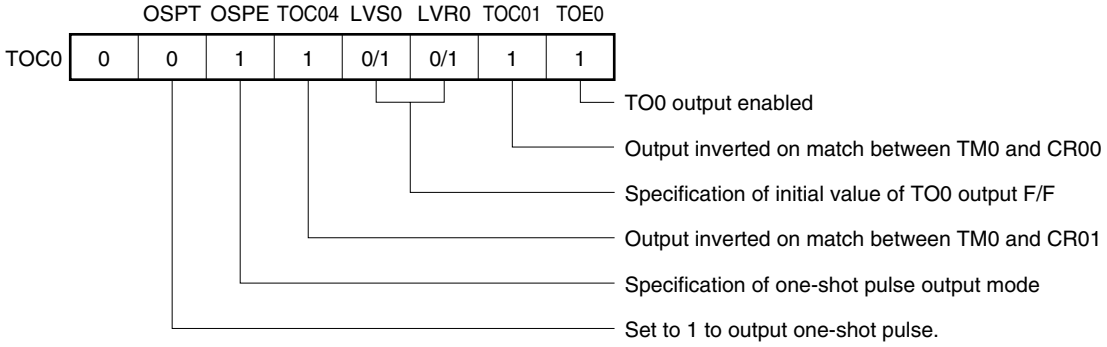
(a) 16-bit timer mode control register 0 (TMC0)



(b) Capture/compare control register 0 (CRC0)



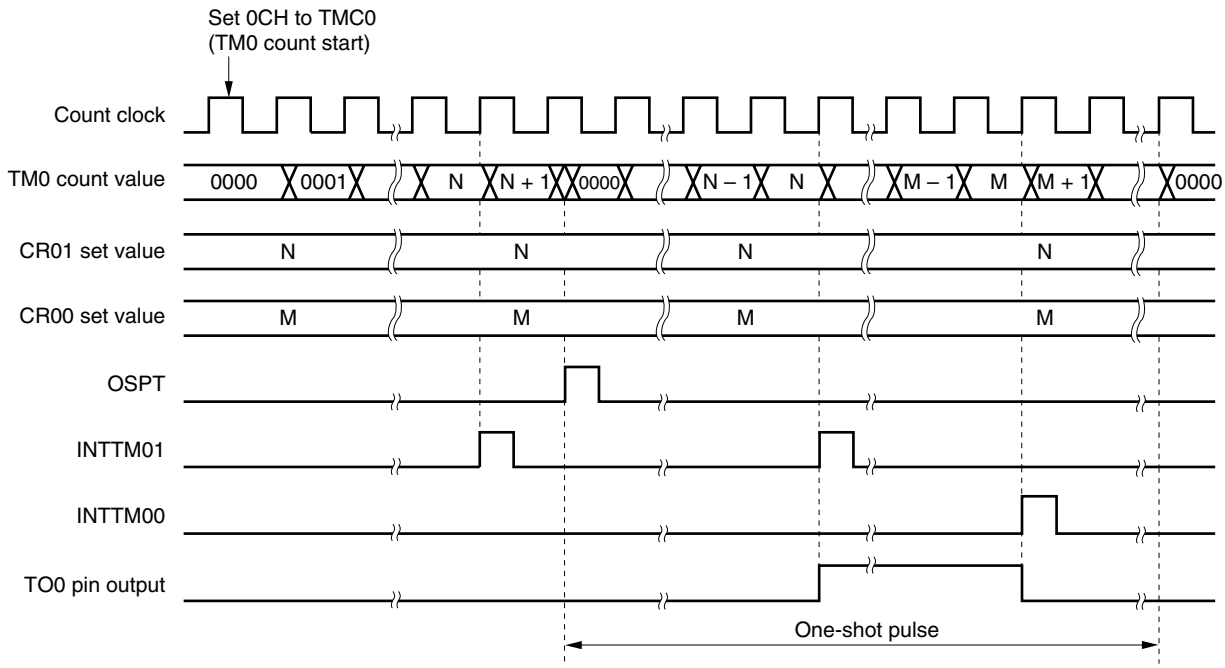
(c) 16-bit timer output control register 0 (TOC0)



Caution Set CR00 and CR01 to a value in the following range.
0000H < CR01 < CR00 ≤ FFFFH

Remark 0/1: When these bits are reset to 0 or set to 1, other functions can be used at the same time as the one-shot pulse output function. For details, refer to Figures 6-2, 6-3, and 6-4.

★ **Figure 6-26. Timing of One-Shot Pulse Output Operation with Software Trigger**



Caution 16-bit timer counter 0 (TM0) starts operating as soon as TMC02 and TMC03 are set to values other than 0, 0 (operation stop mode).

Remark N < M

(2) One-shot pulse output with external trigger

A one-shot pulse can be output to the TO0/P31 pin by using the valid edge of the TI00/P32 pin as an external trigger when 16-bit timer mode control register 0 (TMC0), capture/compare control register 0 (CRC0), and 16-bit timer output control register 0 (TOC0) are set as shown in Figure 6-27.

The rising, falling, or both the rising and falling edges can be selected as the valid edge of the TI00/P32 pin by bit 4 and 5 (ES00 and ES01) of prescaler mode register 0 (PRM0).

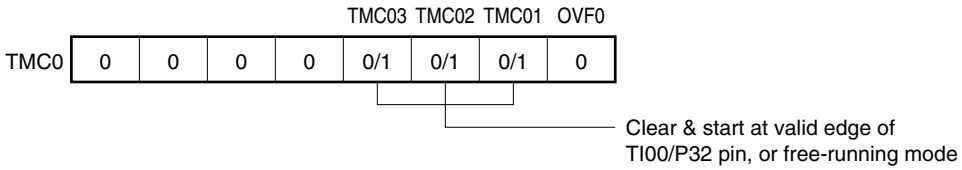
At the valid edge of the TI00/P32 pin, 16-bit timer counter 0 (TM0) is cleared and started, and the output to TO0/P31 becomes active at the count value written in advance to 16-bit capture/compare register 01 (CR01). After that, the output becomes inactive at the count value written in advance in 16-bit capture/compare register 00 (CR00)**Note**.

★ **Note** The case where N < M is described here. When N > M, the output becomes active with the CR00 register and inactive with the CR01 register.

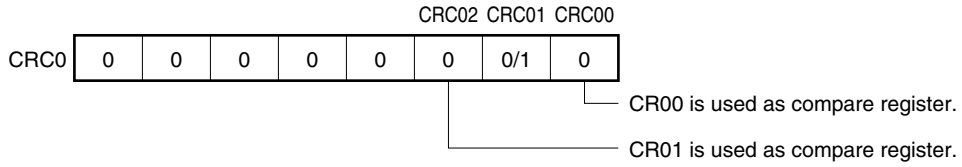
Caution The external trigger is ignored even if generated while the one-shot pulse is being output.

Figure 6-27. Setting of Control Registers for One-Shot Pulse Output Operation with External Trigger

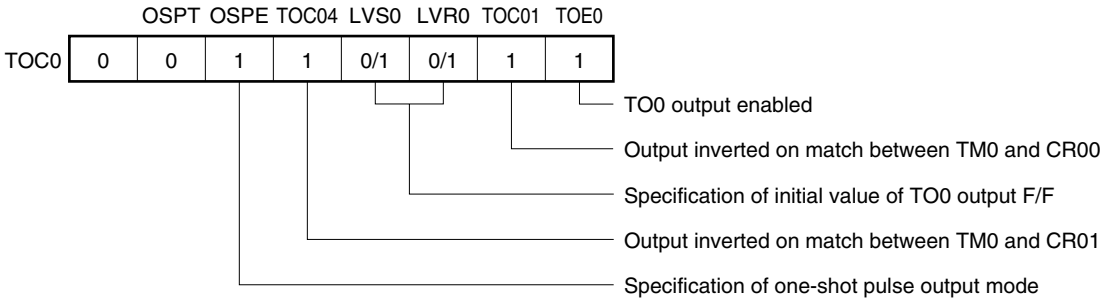
(a) 16-bit timer mode control register 0 (TMC0)



(b) Capture/compare control register 0 (CRC0)



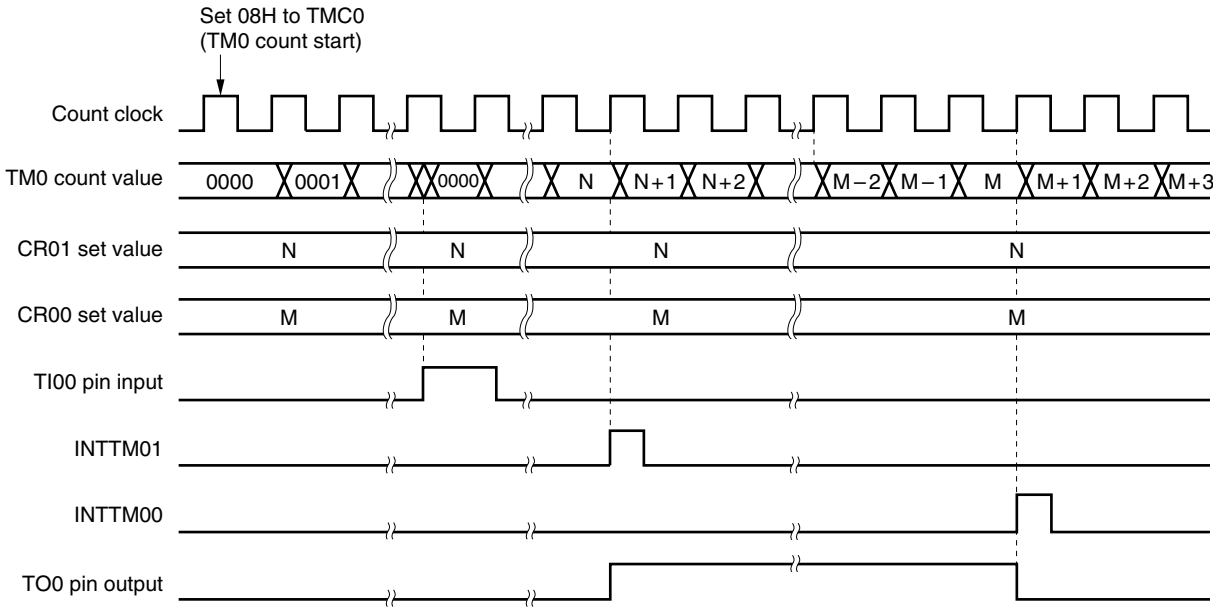
(c) 16-bit timer output control register 0 (TOC0)



Caution Set CR00 and CR01 to a value in the following range.
0000H < CR01 < CR00 ≤ FFFFH

Remark 0/1: When these bits are reset to 0 or set to 1, other functions can be used at the same time as the one-shot pulse output function. For details, refer to Figures 6-2, 6-3, and 6-4.

Figure 6-28. Timing of One-Shot Pulse Output Operation with External Trigger (Clear & Start at Valid Edge of TI00 with Rising Edge Specified)



Caution 16-bit timer counter 0 (TM0) starts operating as soon as TMC02 and TMC03 are set to values other than 0, 0 (operation stop mode).

Remark N < M

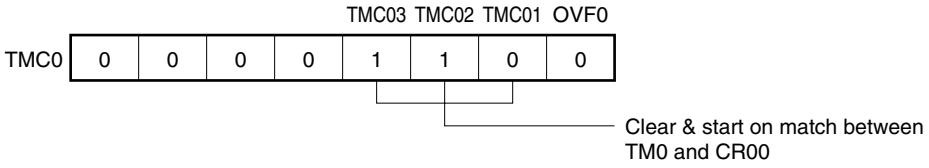
6.4.6 PPG output operation

16-bit timer/event counter 0 operates as a PPG (Programmable Pulse Generator) output when 16-bit timer mode control register 0 (TMC0) and capture/compare control register 0 (CRC0) are set as shown in Figure 6-29.

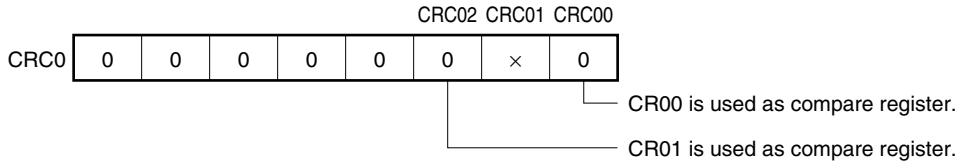
The PPG output pulse is a square wave and is output from the TO0/P31 pin. One cycle of this pulse is specified by the count value preset to 16-bit capture/compare register 00 (CR00), and the pulse width is specified by the count value preset to 16-bit capture/compare register 01 (CR01).

Figure 6-29. Setting of Control Register for PPG Output Operation

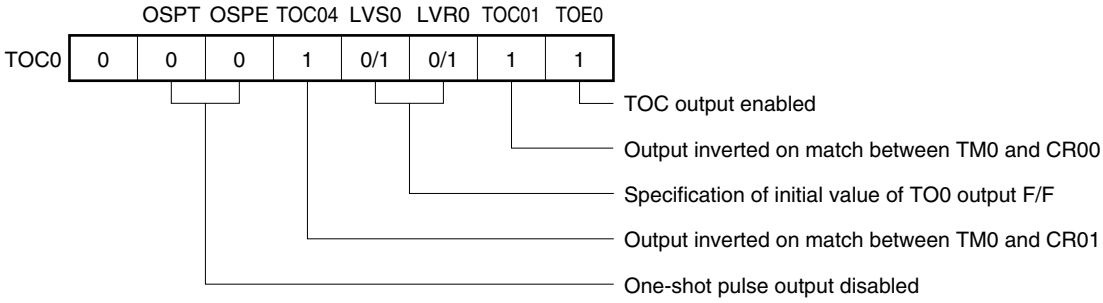
(a) 16-bit timer mode control register 0 (TMC0)



(b) Capture/compare control register 0 (CRC0)



(c) 16-bit timer output control register 0 (TOC0)



Cautions 1. Set CR00 and CR01 to a value in the following range.
0000H < CR01 < CR00 ≤ FFFFH

2. The cycle of the pulse generated by PPG output is (set value of CR00 + 1), and the duty factor is (set value of CR01 + 1)/(set value of CR00 + 1).

Remark ×: Don't care

Figure 6-30. Configuration of PPG Output

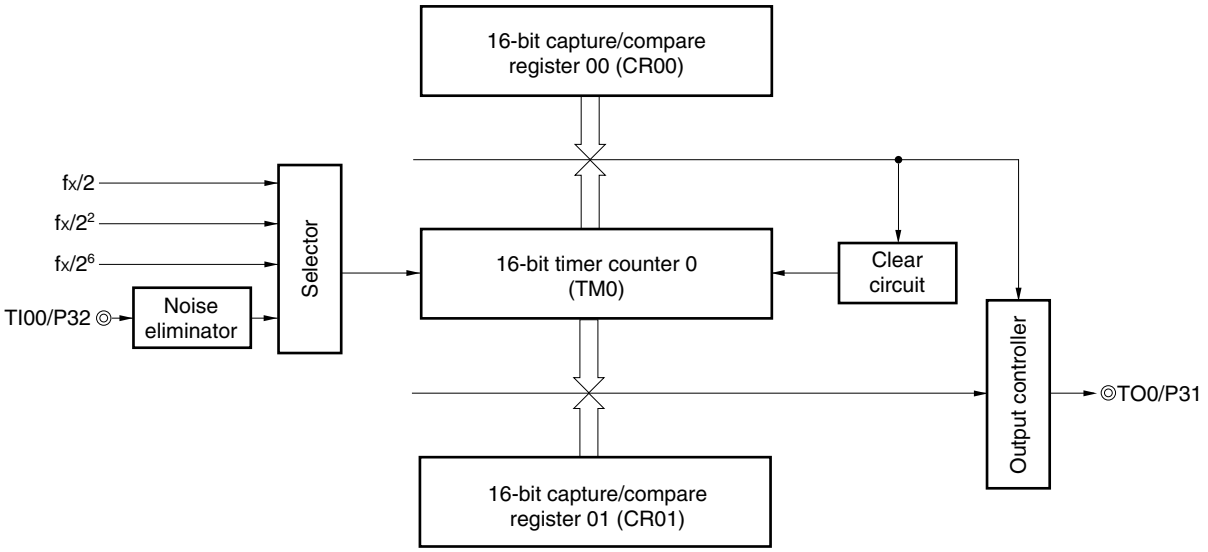
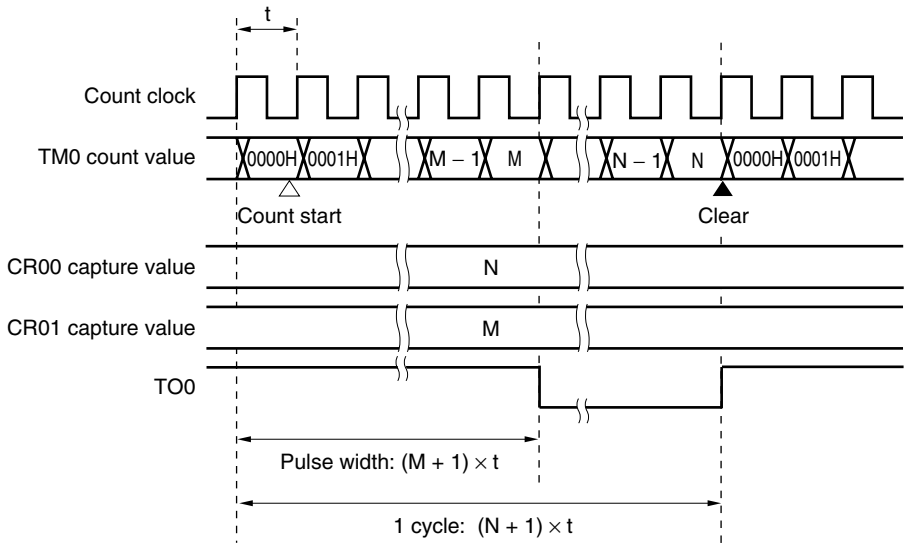


Figure 6-31. PPG Output Operation Timing



Remark 0000H < M < N ≤ FFFFH

★ 6.5 Program List

Caution The following sample program is shown as an example to describe the operation of semiconductor products and their applications. Therefore, when applying the following information to your devices, design the devices after performing evaluation on your own responsibility.

6.5.1 Interval timer

```
/************************************************************************/
/*                                                                      */
/*           Setting example of timer 0 interval timer mode             */
/*       Cycle set to 98 as intervalTM0 (at 6.3 MHz for 1 ms)           */
/*        Variable ppgdata prepared as rewrite data area                */
/*         : Cycle (if 0000, no change)                                 */
/*        ppgdata to be checked at every INTTM00, and changed if required.  */
/*        Therefore, if change is required, set the change data in ppgdata.  */
/*        When changed, ppgdata cleared to 0000.                        */
/*                                                                      */
/************************************************************************/
#pragma sfr
#pragma EI
#pragma DI
#define intervalTM0 98                  /* Cycle data to be set to CR00 */
#pragma interrupt INTTM00 intervalint rb2
        unsigned int ppgdata;           /* Data area to be set to timer 0 */

void main(void)
{
        PCC = 0x0;                      /* Set high-speed operation mode */
        ppgdata = 0;
                                        /* Set port */
                                        /* Set the following to output */
        P3 = 0b11111101;                /* Clear P31 */
        PM3.1 = 0;                      /* Set P31 as output */
                                        /* Set interrupt */
        TMMK00 = 0;                     /* Cancel INTTM00 interrupt mask */
                                        /* Set timer 0 */
        PRM0 = 0b00000010;              /* Count clock is fx/2^6 */
        CRC0 = 0b00000000;              /* Set CR00 and CR01 to compare register */
        CR00 = intervalTM0;             /* Set cycle initial value to CR00 */
        TOC0 = 0b00000111;              /* Invert on match with CR00, initial value L */
        TMC0 = 0b00001100;              /* Clear & start on match between TM0 and CR00 */
        EI();

        while(1);                       /* Loop as dummy here */
}

/* Timer 0 interrupt function */
void intervalint()
{
        unsigned int work;
/*****************************************************/
/*                                                   */
/* Define variables required for interrupt here      */
/*                                                   */
/*****************************************************/
        work = ppgdata;
        if (work != 0)
        {
                CR00 = work;
                ppgdata = 0;
                if (work == 0xffff)
                {
                        TMC0 = 0b00000000;     /* Stop timer */
                }
          }
/*********************************************************/
/*                                                       */
/* Describe processing required for interrupt below      */
/*                                                       */
/*********************************************************/
}
```

6.5.2 Pulse width measurement by free-running counter and one capture register

```
/************************************************************************/
/*                                                                      */
/*           Timer 0 operation sample                                   */
/*       Pulse width measurement example by free-running and CR01       */
/*        Measurement results up to 16 bits and not checked for errors  */
/*        data[0]: End flag                                             */
/*        data[1]: Measurement results (pulse width)                    */
/*        data[2]: Previous read value                                  */
/*                                                                      */
/************************************************************************/
#pragma sfr
#pragma EI
#pragma DI
#pragma interrupt INTTM01 intervalint rb2
        unsigned int data[3];           /* Data area */

void main(void)
{
        unsigned int length;
        PCC = 0x0;                      /* Set high-speed operation mode */
        data[0] = 0;
        data[1] = 0;
        data[2] = 0;
                                        /* Set port */
        PM3.2 = 1;                      /* Set P32 as input */
                                        /* Set interrupt */
        TMMK01 = 0;                     /* Cancel INTTM01 interrupt mask */
                                        /* Set timer 0 */
        PRM0 = 0b00110010;              /* Both rising and falling edges for TI00 */
                                        /* Count clock is fx/2^6              */
        CRC0 = 0b00000100;              /* Set CR01 to capture register */
        TMC0 = 0b00000100;              /* Start in free-run mode */
        EI();
        while(1){                       /* Dummy loop */
                while(data[0] == 0);    /* Wait for measurement completion */
                DI();                   /* Prohibit interrupt for exclusive operation */
                length = data[1];       /* Read measurement results */
                data[0] = 0;            /* Clear end flag */
                EI();                   /* Exclusive operation completed */
        }
}

/* Timer 0 interrupt function */
void intervalint()
{
        unsigned int work;
/*****************************************************/
/*                                                   */
/* Define variables required for interrupt here      */
/*                                                   */
/*****************************************************/
        work = CR01;                    /* Read capture value */
        data[1] = work - data[2];       /* Calculate and update interval */
        data[2] = work;                 /* Update read value */
        data[0] = 0xffff;               /* Set measurement completion flag*/

/*********************************************************/
/*                                                       */
/* Describe processing required for interrupt below      */
/*                                                       */
/*********************************************************/
}
```

6.5.3 Two-pulse-width measurement by free-running counter

```
/************************************************************************/
/*                                                                      */
/*          Timer 0 operation sample                                    */
/*       Two-pulse-width measurement sample by free-running             */
/*       Measurement results up to 16 bits and not checked for errors   */
/*         Result area at TI00 side                                     */
/*        data[0]: End flag                                             */
/*        data[1]: Measurement results (pulse width)                    */
/*        data[2]: Previous read value                                  */
/*         Result area at TI01 side                                     */
/*        data[3]: End flag                                             */
/*        data[4]: Measurement results (pulse width)                    */
/*        data[5]: Previous read value                                  */
/*                                                                      */
/************************************************************************/
#pragma sfr
#pragma EI
#pragma DI
#pragma interrupt INTTM00 intervalint rb2
#pragma interrupt INTTM01 intervalint2 rb2
        unsigned int data[6];           /* Data area */

void main(void)
{
        unsigned int length,length2;
        PCC = 0x0;                      /* Set high-speed operation mode */
        data[0] = 0;                    /* Clear data area */
        data[1] = 0;
        data[2] = 0;
        data[3] = 0;
        data[4] = 0;
        data[5] = 0;

                                        /* Set port */
        PM3.2 = 1;                      /* Set P32 as input */
        PM3.3 = 1;                      /* Set P33 as input */
                                        /* Set interrupt */
        TMMK01 = 0;                     /* Cancel INTTM01 interrupt mask */
        TMMK00 = 0;                     /* Cancel INTTM00 interrupt mask */
                                        /* Set timer 0 */
        PRM0 = 0b11110010;              /* Both rising and falling edges */
                                        /* Count clock is fx/2^6              */
        CRC0 = 0b00000101;              /* Set CR00 and CR01 to capture register */
        TMC0 = 0b00000100;              /* Start in free-run mode */
        EI();
        while(1){                       /* Dummy loop */
                if(data[0] != 0)        /* TI00 measurement completion check */
                {
                        TMMK01 = 1;             /* INTTM01 interrupt prohibited for
                                                   exclusive operation */
                        length = data[1];       /* Read measurement results */
                        data[0] = 0;            /* Clear end flag */
                        TMMK01 = 0;             /* Exclusive operation completed */
                }
                if(data[3] != 0)        /* TI01 measurement completion check */
                {
                        TMMK00 = 1;             /* INTTM00 interrupt prohibited for
                                                   exclusive operation */
                        length2 = data[4];      /* Read measurement results */
                        data[3] = 0;            /* Clear end flag */
                        TMMK00 = 0;             /* Exclusive operation completed */
                }
        }
}
```

```
/* INTTM00 interrupt function */
void intervalint()
{
        unsigned int work;
/*******************************************************/
/*                                                     */
/* Define variables required for interrupt here        */
/*                                                     */
/*******************************************************/
        work = CR00;                    /* Read capture value */
        data[4] = work - data[5];       /* Calculate and update interval */
        data[5] = work;                 /* Update read value */
        data[3] = 0xffff;               /* Set measurement completion flag*/

/*******************************************************/
/*                                                     */
/* Define variables required for interrupt below       */
/*                                                     */
/*******************************************************/
}
/* INTTM01 interrupt function */
void intervalint2()
{
        unsigned int work;
/*******************************************************/
/*                                                     */
/* Define variables required for interrupt here        */
/*                                                     */
/*******************************************************/
        work = CR01;                    /* Read capture value */
        data[1] = work - data[2];       /* Calculate and update interval */
        data[2] = work;                 /* Update read value */
        data[0] = 0xffff;               /* Set measurement completion flag*/

/*******************************************************/
/*                                                     */
/* Define variables required for interrupt below       */
/*                                                     */
/*******************************************************/
}
```

6.5.4 Pulse width measurement by restart

```
/************************************************************************/
/*                                                                      */
/*           Timer 0 operation sample                                   */
/*       Pulse width measurement example by restart                     */
/*       Measurement results up to 16 bits, not to be checked for errors  */
/*        data[0]: End flag                                             */
/*        data[1]: Measurement results (pulse width)                    */
/*        data[2]: Previous read value                                  */
/*                                                                      */
/************************************************************************/
#pragma sfr
#pragma EI
#pragma DI
#pragma interrupt INTTM01 intervalint rb2
        unsigned int data[3];           /* Data area */

void main(void)
{
        unsigned int length;
        PCC = 0x0;                      /* Set high-speed operation mode */
        data[0] = 0;
        data[1] = 0;
        data[2] = 0;
                                        /* Set port */
        PM3.2 = 1;                      /* Set P32 as input */
                                        /* Set interrupt */
        TMMK01 = 0;                     /* Cancel INTTM01 interrupt mask */
                                        /* Set timer 0 */
        PRM0 = 0b00110010;              /* Both rising and falling edges  */
                                        /* Count clock is fx/2^6        */
        CRC0 = 0b00000100;              /* Set CR01 to capture register   */
        TMC0 = 0b00001000;              /* Clear & start at TI00 valid edge */
        EI();
        while(1){                       /* Dummy loop */
                if(data[0] != 0)        /* Wait for TI00 measurement completion */
                {
                        TMMK01 = 1;             /* Prohibit INTTM01 for exclusive
                                                   operation */
                        length = data[1]+data[2];       /* Cycle calculation based on
                                                           measurement results */
                        data[0] = 0;            /* Clear end flag */
                        TMMK01 = 0;             /* Exclusive operation completed */
                }
        }
}

/* Timer 0 interrupt function */
void intervalint()
{
/*******************************************************/
/*                                                     */
/* Define variables required for interrupt here        */
/*                                                     */
/*******************************************************/
        data[2] = data[1];              /* Update old data */
        data[1] = CR01;                 /* Update read value */
        data[0] = 0xffff;               /* Set measurement completion flag*/

/*******************************************************/
/*                                                      */
/* Define variables required for interrupt below        */
/*                                                      */
/*******************************************************/
}
```

6.5.5 PPG output

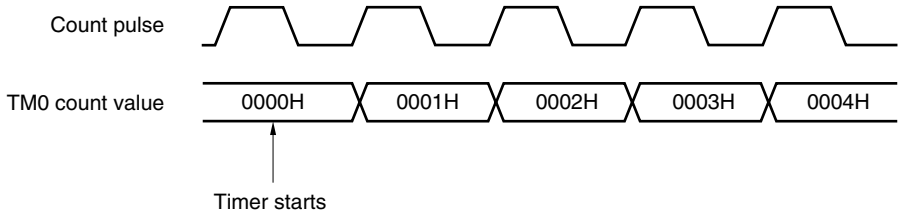
```
/*****************************************************************************/
/*                                                                           */
/*           Timer 0 PPG mode setting example                                */
/*       Cycle set to 98 as intervalTM0                                      */
/*       Active period set to 49 as active_time                              */
/*        Array ppgdata prepared as data area for rewriting                  */
/*        [0]: Active period (0000: no change, 0xffff: stop)                 */
/*        [1]: Cycle (0000: no change)                                       */
/*        ppgdata to be checked at every INTTM00, and changed if required.   */
/*        Therefore, if change is required, set the change data in ppgdata.  */
/*        When changed, ppgdata cleared to 0000.                             */
/*                                                                           */
/*****************************************************************************/
#pragma sfr
#pragma EI
#pragma DI
#define intervalTM0 98                  /* Cycle data to be set to CR00 */
#define active_time 49                  /* Initial value data of CR01 */
#pragma interrupt INTTM00 ppgint rb2
        unsigned int ppgdata[2];        /* Data area to be set to timer 0 */

void main(void)
{
        PCC = 0x0;                      /* Set high-speed operation mode */
        ppgdata[0] = 0;
        ppgdata[1] = 0;
                                        /* Set port */
        P3 = 0b11111101;                /* Clear P31 */
        PM3.1 = 0;                      /* Set P31 to output */
                                        /* Set interrupt */
        TMMK00 = 0;                     /* Cancel INTTM00 interrupt mask */
                                        /* Set timer 0 */
        PRM0 = 0b00000010;              /* Count clock is fx/2^6 */
        CRC0 = 0b00000000;              /* Set CR00 and CR01 to compare register */
        CR00 = intervalTM0;             /* Set initial value of cycle */
        CR01 = active_time;             /* Set initial value of active period */
        TOC0 = 0b00010111;              /* Inverted on match between CR00 and CR01,
                                           initial value L */
        TMC0 = 0b00001100;              /* Clear & start on match between TM0 and CR00 */
        EI();

        while(1);
}

/* Timer 0 interrupt function */
void ppgint()
{
        unsigned int work;
        work = ppgdata[0];
        if (work != 0)
        {
                CR01 = work;
                ppgdata[0] = 0;
                if (work == 0xffff)
                {
                        TMC0 = 0b00000000;      /* Stop timer */
                }
        }
        work = ppgdata[1];
        if (work != 0)
        {
                CR00 = work;
                ppgdata[1]=0;
        }
}
```

6.6 Notes on 16-Bit Timer/Event Counter 0

(1) Error on starting timer

An error of up to 1 clock occurs from when the timer is started until a match signal is generated. This is because 16-bit timer counter 0 (TM0) is started asynchronously to the count pulse.

Figure 6-32. Start Timing of 16-Bit Timer Counter



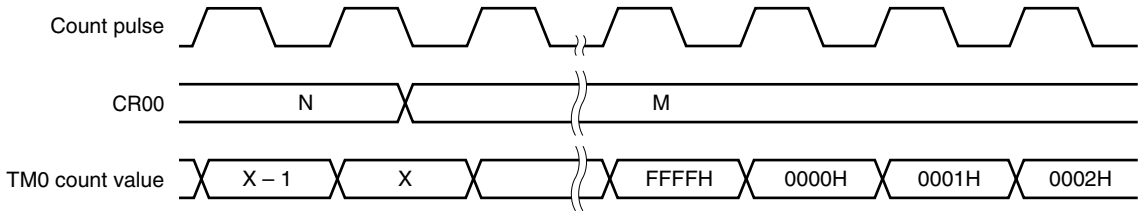
(2) Setting of 16-bit capture/compare register (in start & clear mode on match between TM0 and CR00)

Set 16-bit capture/compare registers 00 and 01 (CR00 and CR01) to a value other than 0000H (one pulse cannot be counted).

(3) Operation after changing value of compare register during timer count operation

If a new value written to 16-bit capture/compare register 00 (CR00) is less than the value of 16-bit timer counter 0 (TM0), TM0 continues counting, overflows, and starts counting again from 0. Therefore, if the new value of CR00 (M) is less than the old value (N), it is necessary to restart the timer after changing the value of CR00.

Figure 6-33. Timing After Changing Value of Compare Register During Timer Count Operation

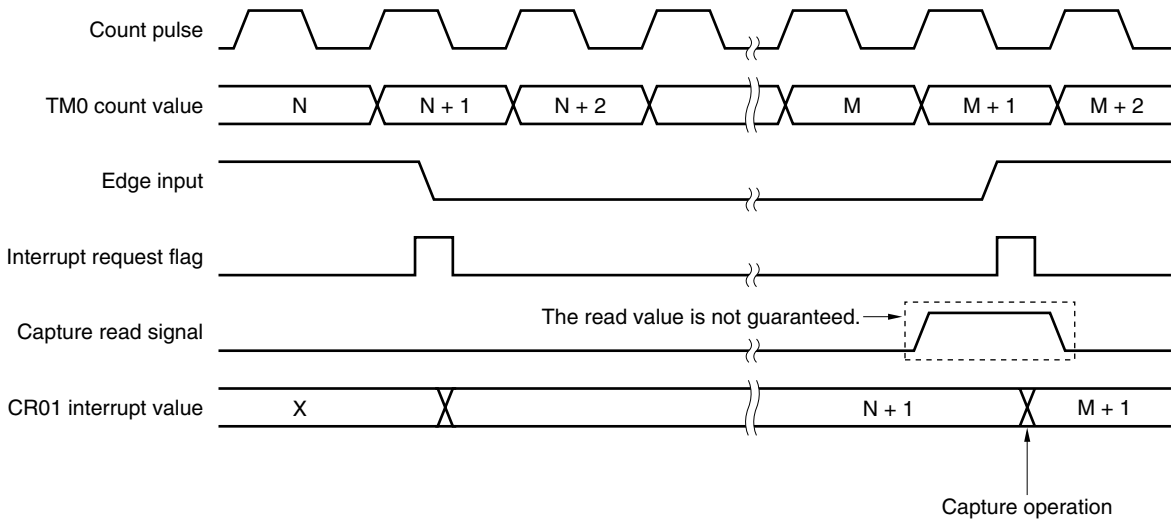


Remark N > X > M

(4) Data retention timing of capture register

When a capture trigger is input while a 16-bit capture/compare register (CR00/CR01) is being read, CR00/CR01 continues the normal capture operation, but the read value at this time is not guaranteed. However, the interrupt request flag (TMIF00/TMIF01) is set as a result of detecting the valid edge.

Figure 6-34. Data Retention Timing of Capture Register



(5) Setting of valid edge

Set the valid edge of the TI00/P32 pin after clearing bits 2 and 3 (TMC02 and TMC03) of 16-bit timer mode control register 0 to 0, 0, and stopping the timer operation. The valid edge is specified by using bits 4 and 5 (ES00 and ES01) of prescaler mode register 0 (PRM0).

(6) Re-triggering one-shot pulse

(a) One-shot pulse output with software trigger

Do not set OSPT to 1 while the one-shot pulse is being output. To output the one-shot pulse again, wait until the INTTM00 interrupt, which occurs on a match between TM0 and CR00, has occurred.

(b) One-shot pulse output with external trigger

The external trigger is ignored even if it is generated again while the one-shot pulse is being output.

★ **(c) One-shot pulse output function**

When using the software trigger for one-shot pulse output, fix the level of the TI00/P32 and TI01/P33 pins to either the high or low level. Otherwise, the external trigger will remain valid even when the software trigger is used, and the timer will be cleared and started when the level of the TI00/P32 or TI01/P33 pin changes, resulting in unexpected output of the pulse.

(7) Operation of OVF0 flag

(a) OVF0 flag setting

The OFV0 flag is set to 1 in the following case.

When one of clear & start mode on match between TM0 and CR00, clear & start mode on TI00 valid edge, or free-running mode is selected.

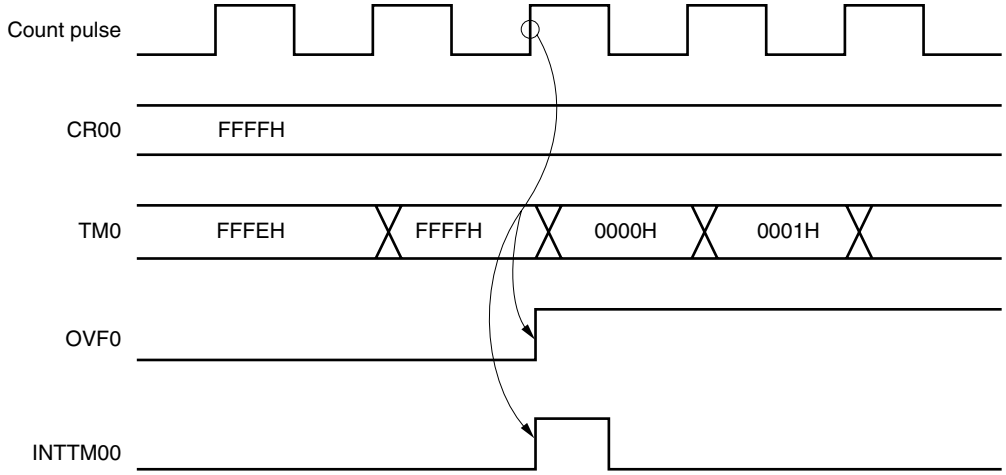
↓

CR00 is set to FFFFH.

↓

TM0 is counted up from FFFFH to 0000H.

Figure 6-35. Operation Timing of OVF0 Flag



(b) OVF0 flag clear

Even if the OVF0 flag is cleared before the next count clock is counted (before TM0 becomes 0001H) after TM0 has overflowed, the OVF0 flag is set again and the clear becomes invalid.

(8) Conflicting operations

(a) If the read period and capture trigger input conflict

If the read period and inputting a capture trigger conflict while 16-bit capture/compare registers 00 and 01 (CR00 and CR01) are used as capture registers, the capture operation takes precedence and the read data is undefined. However, the interrupt request flags (TMIF00/TMIF01) are set when the valid edge is detected.

(b) If the match timing of the write period and TM0 conflict

When 16-bit capture/compare registers 00 and 01 (CR00, CR01) are used as capture registers, because match detection cannot be performed correctly if the match timing of the write period and 16-bit timer counter 0 (TM0) conflict, do not write to CR00 and CR01 close to the match timing.

(9) Timer operation

<1> Even if 16-bit timer counter 0 (TM0) is read, its value is not captured to 16-bit capture/compare register 01 (CR01).

<2> While the timer is stopped, signals input to the TI00 and TI01 pins are not accepted regardless of the operating mode of the CPU.

<3> The one-shot pulse output operates correctly only in the free-running mode or the mode in which TM0 is cleared and started at the valid edge of TI00. In the mode in which TM0 is cleared and started on a match between TM0 and CR00, the one-shot operation cannot be performed because TM0 does not overflow.

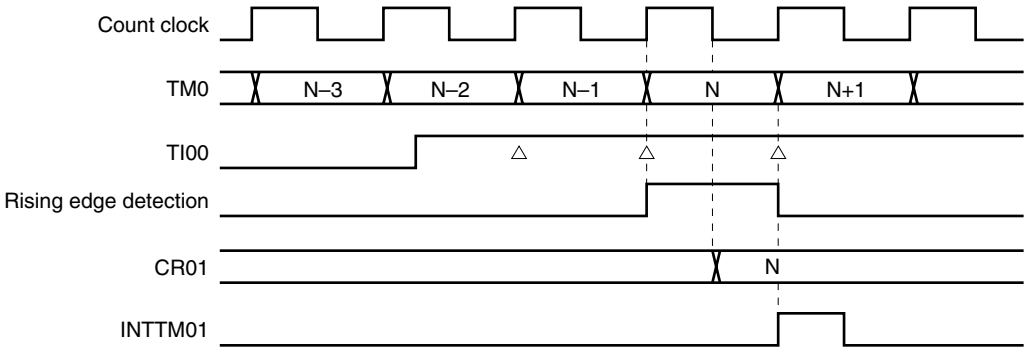
(10) Capture operation

<1> If the TI00 valid edge is specified as the count clock, a capture operation by the capture register specified as the trigger for TI00 is not possible.

<2> If both the rising and falling edges are selected as the valid edges of TI00, CR00 does not perform a capture operation.

<3> To ensure the reliability of the capture operation, the capture trigger requires a pulse longer than two cycles of the count clock selected by prescaler mode register 0 (PRM0).

Figure 6-36. CR01 Capture Operation with Rising Edge Specified



<4> The capture operation is performed at the fall of the count clock. Interrupt request input (INTTM00, INTTM01), however, occurs at the rise of the next count clock.

(11) Compare operation

<1> When a 16-bit timer capture/compare register (CR00/CR01) is overwritten during timer operation, a match interrupt may be generated or the clear operation may not be performed normally if that value is close to or larger than the timer value.

<2> The capture operation may not be performed for CR00/CR01 set to compare mode even if a capture trigger is input.

(12) Edge detection

<1> If the TI00 pin or the TI01 pin is high level immediately after system reset and the rising edge or both the rising and falling edges are specified as the valid edge of the TI00 pin or TI01 pin to enable 16-bit timer counter 0 (TM0) operation, a rising edge is detected immediately. Be careful when pulling up the TI00 pin or the TI01 pin. However, the rising edge is not detected at restart after the operation has been stopped.

<2> The sampling clock used to eliminate noise differs when the TI00 valid edge is used as the count clock and when it is used as a capture trigger. In the former case, the count clock is $f_X/2^3$, and in the latter case the count clock is selected by prescaler mode register 0 (PRM0). The capture operation is not performed until the valid edge is sampled and the valid level is detected twice, thus eliminating noise with a short pulse width.

★ (13) STOP mode setting

Stop the timer operation before setting STOP mode; otherwise the timer may malfunction when the main system clock starts.

CHAPTER 7 8-BIT TIMER/EVENT COUNTERS 50, 51

7.1 Functions of 8-Bit Timer/Event Counters 50, 51

8-bit timer/event counters 50 and 51 have the following two modes.

- Mode in which an 8-bit timer/event counter is used alone (single mode)
- Mode in which the two timer/event counters are connected in cascade (cascade connection mode with a resolution of 16 bits)

These two modes are explained below.

(1) Mode in which an 8-bit timer/event counter is used alone (single mode)

The timer/event counter operates as an 8-bit timer/event counter.
In this mode, the following functions can be used.

<1> Interval timer

Interrupt requests are generated at the preset interval.

- Number of counts: 1 to 256

<2> External event counter

The number of pulses with high/low level widths in a signal input externally can be measured.

<3> Square-wave output

A square wave with an arbitrary frequency can be output.

- Cycle: (1 × 2 to 256 × 2) × Count clock cycle

<4> PWM output

A pulse with an arbitrary duty ratio can be output.

- Cycle: Count clock × 256
- Duty ratio: Set value of compare register/256

(2) Mode in which the two timer/event counters are connected in cascade (cascade connection mode with a resolution of 16 bits)

By combining the two 8-bit timer/event counters, they can operate as a 16-bit timer/event counter.
In this mode, the following functions can be used.

- Interval timer with 16-bit resolution
- External event counter with 16-bit resolution
- Square-wave output with 16-bit resolution

Figures 7-1 and 7-2 show the block diagrams of 8-bit timer/event counters 50 and 51.

Figure 7-1. Block Diagram of 8-Bit Timer/Event Counter 50

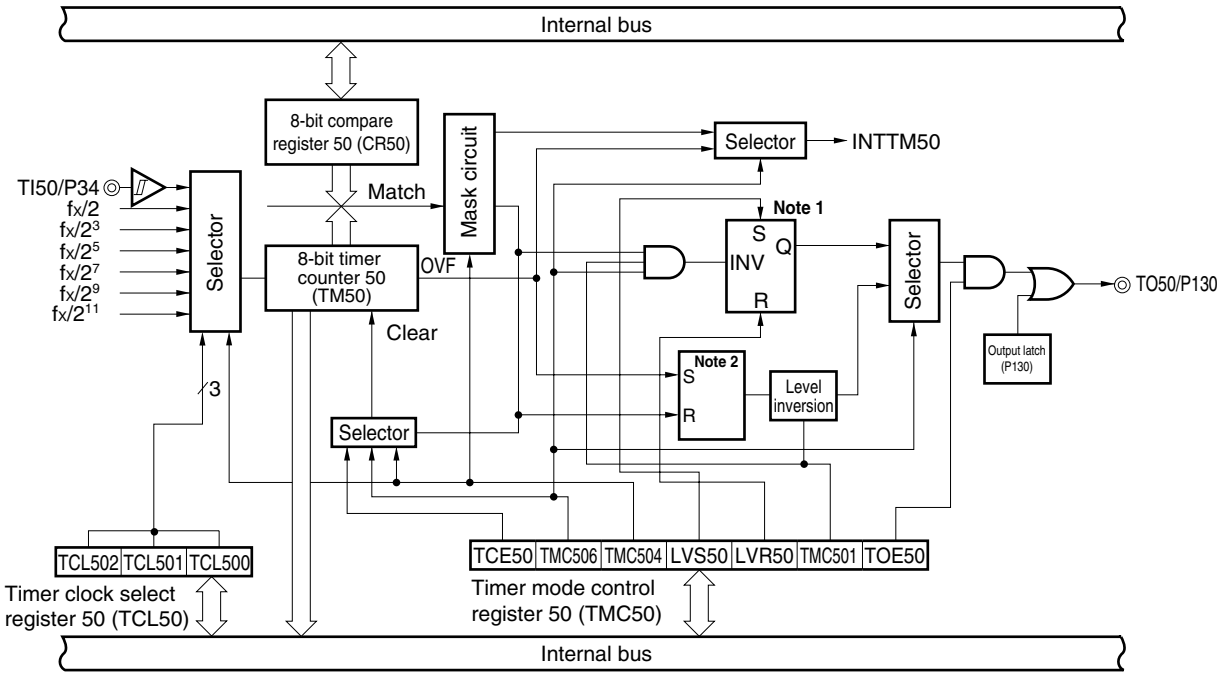
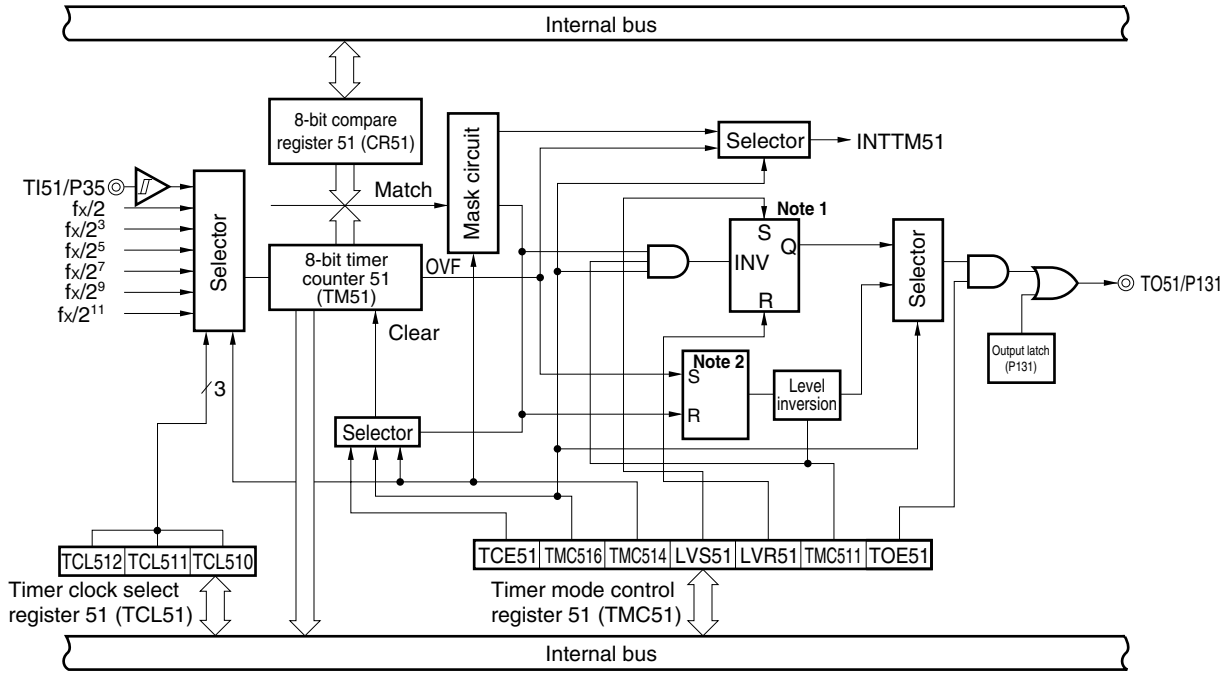


Figure 7-2. Block Diagram of 8-Bit Timer/Event Counter 51



Notes 1. Timer output F/F

2. PWM output F/F

7.2 Configuration of 8-Bit Timer/Event Counters 50, 51

8-bit timer/event counters 50 and 51 consist of the following hardware.

Table 7-1. Configuration of 8-Bit Timer/Event Counters 50 and 51

Item	Configuration
Timer counter	8-bit timer counters 50 and 51 (TM50 and TM51)
Register	8-bit compare registers 50 and 51 (CR50 and CR51)
Timer input	TI50, TI51
Timer output	TO50 and TO51
Control registers	• Timer clock select registers 50 and 51 (TCL50 and TCL51) • 8-bit timer mode control registers 50 and 51 (TMC50 and TMC51) • Port mode register 3 (PM3) • Port 13 (P13)

(1) 8-bit timer counters 50 and 51 (TM50 and TM51)

TM50 and TM51 are 8-bit read-only registers that count the count pulses.

The counter is incremented at the rising edge of the count clock.

When TM50 and TM51 are cascaded and used as a 16-bit timer, their values can be read by using a 16-bit memory manipulation instruction. However, because TM50 and TM51 are connected by the internal 8-bit bus, TM50 and TM51 are read separately in this order. Therefore, read the value of TM50 and TM51 when used as a 16-bit timer twice for comparison, taking changes in the value during counting into consideration.

If the count value is read while the timer is operating, stop input of the count clock**Note**, and read the count value at that point.

The count value is cleared to 00H in the following cases.

<1> When $\overline{\text{RESET}}$ is input

<2> When TCE5n is cleared

<3> Upon a match between TM5n and CR5n in the mode in which the timer is cleared and started on a match between TM5n and CR5n

★ **Note** An error may occur in the count. Select a count clock that has a high/low level longer than two cycles of the CPU clock.

Caution In cascade connection mode, the count value is reset to 0000H when TCE50 of the lowest timer is cleared.

Remark n = 0, 1

(2) 8-bit compare registers 50 and 51 (CR50 and CR51)

When CR5n is used as a compare register in other than PWM mode, the value set to CR5n is constantly compared with 8-bit timer counter 5n (TM5n) count value, and an interrupt request (INTTM5n) is generated if they match.

★ In PWM mode, the TO5n pin goes to the active level by the overflow of TM5n. When the values of TM5n and CR5n match, the TO5n pin goes to the inactive level.

It is possible to rewrite the value of CR5n within 00H to FFH during a count operation.

When TM50 and TM51 can be connected in cascade and used as a 16-bit timer, CR50 and CR51 operate as a 16-bit compare register. This register compares the count value with the register value, and if the values match, an interrupt request (INTTM50) is generated. The INTTM51 interrupt request is also generated at this time, so mask the INTTM51 interrupt request.

Set CR5n using an 8-bit memory manipulation instruction. $\overline{\text{RESET}}$ input makes CR5n undefined.

Caution In the cascade connection mode, stop the timer operation before setting data.

Remark n = 0, 1

7.3 Registers Controlling 8-Bit Timer/Event Counters 50, 51

The following four types of registers control 8-bit timer/event counters 50 and 51.

- Timer clock select registers 50 and 51 (TCL50 and TCL51)
- 8-bit timer mode control registers 50 and 51 (TMC50 and TMC51)
- Port mode register 3 (PM3)
- Port 13 (P13)

(1) Timer clock select registers 50 and 51 (TCL50 and TCL51)

These registers select the count clock of 8-bit timer/event counters 50 and 51 (TM50 and TM51) and the valid edge of the TI50 and TI51 inputs.

TCL50 and TCL51 are set by using an 8-bit memory manipulation instruction.

The values of these registers are cleared to 00H after reset.

Figure 7-3. Format of Timer Clock Select Register 50 (TCL50)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TCL50	0	0	0	0	0	TCL502	TCL501	TCL500	FF84H	00H	R/W

TCL502	TCL501	TCL500	Selection of count clock
0	0	0	Falling edge of TI50
0	0	1	Rising edge of TI50
0	1	0	$f_X/2$ (3.15 MHz)
0	1	1	$f_X/2^3$ (788 kHz)
1	0	0	$f_X/2^5$ (197 kHz)
1	0	1	$f_X/2^7$ (49.2 kHz)
1	1	0	$f_X/2^9$ (12.3 kHz)
1	1	1	$f_X/2^{11}$ (3.08 kHz)

Cautions 1. When changing the data of TCL50, be sure to stop the timer operation.

2. Be sure to reset bits 3 to 7 to "0".

Remarks 1. In the cascade connection mode, the only setting of the count clock in TCL50 is valid.

2. f_X: System clock oscillation frequency

3. ( ): f_X = 6.3 MHz

Figure 7-4. Format of Timer Clock Select Register 51 (TCL51)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TCL51	0	0	0	0	0	TCL512	TCL511	TCL510	FF87H	00H	R/W

TCL512	TCL511	TCL510	Selecttion of count clock
0	0	0	Falling edge of TI51
0	0	1	Rising edge of TI51
0	1	0	$f_X/2$ (3.15 MHz)
0	1	1	$f_X/2^3$ (788 kHz)
1	0	0	$f_X/2^5$ (197 kHz)
1	0	1	$f_X/2^7$ (49.2 kHz)
1	1	0	$f_X/2^9$ (12.3 kHz)
1	1	1	$f_X/2^{11}$ (3.08 kHz)

Cautions 1. When changing the data of TCL51, be sure to stop the timer operation.
2. Be sure to reset bits 3 to 7 to "0".

Remarks 1. In the cascade connection mode, the setting of TCL51 is invalid.
2. f_X: System clock oscillation frequency
3. ( ): f_X = 6.3 MHz

(2) 8-bit timer mode control registers 50 and 51 (TMC50 and TMC51)

TMC50 and TMC51 are registers that are used for the following.

<1> Controlling count operation of 8-bit timer counters 50 and 51 (TM50 and TM51)
<2> Selecting operation mode of 8-bit timer counters 50 and 51 (TM50 and TM51)
<3> Selecting single mode or cascade connection mode (TMC51 only)
<4> Setting status of timer output F/F (flip-flop)
<5> Controlling timer F/F or selecting active level in PWM (free-running) mode
<6> Controlling timer output

TMC50 and TMC51 can be set by using a 1-bit or 8-bit memory manipulation instruction.
They are cleared to 00H after reset.
Figures 7-5 and 7-6 show the formats of TMC50 and TMC51.

Figure 7-5. Format of 8-Bit Timer Mode Control Register 50 (TMC50)

Symbol	<7>	6	5	4	<3>	<2>	1	<0>	Address	After reset	R/W
TMC50	TCE50	TMC506	0	TMC504	LVS50	LVR50	TMC501	TOE50	FF85H	00H	R/W

TCE50	Control of count operation of TM50
0	Counter cleared to 0 and count operation disabled (prescaler disabled)
1	Count operation starts

TMC506	Selection of operating mode of TM50
0	Mode of clearing and starting TM50 on match between TM50 and CR50
1	PWM (free-running) mode

TMC504	Be sure to reset this bit to “0”.

LVS50	LVR50	Setting of status of timer output F/F
0	0	Not affected
0	1	Timer output F/F reset to 0
1	0	Timer output F/F set to 1
1	1	Setting prohibited

TMC501	Other than PWM mode (TMC506 = 0)	PWM mode (TMC506 = 1)
	Control of timer F/F	Selection of active level
0	Inversion operation disabled	Active high
1	Inversion operation enabled	Active low

TOE50	Control of timer output
0	Output (port mode) disabled
1	Output enabled

Caution Be sure to reset bit 4 (TMC504) to “0”.

Remarks 1. The PWM output becomes inactive when TCE50 = 0 in the PWM mode.

2. LVS50 and LVR50 are 0 when read after data has been set.

Figure 7-6. Format of 8-Bit Timer Mode Control Register 51 (TMC51)

Symbol	<7>	6	5	4	<3>	<2>	1	<0>	Address	After reset	R/W
TMC51	TCE51	TMC516	0	TMC514	LVS51	LVR51	TMC511	TOE51	FF88H	00H	R/W

TCE51	Control of count operation of TM51
0	Counter cleared to 0 and count operation disabled (prescaler disabled)
1	Count operation starts

TMC516	Selection of operating mode of TM51
0	Mode of clearing and starting TM51 on match between TM51 and CR51
1	PWM (free-running) mode

TMC514	Selection of single mode or cascade connection mode
0	Single mode
1	Cascade connection mode (connected to lower timer (TM50))

LVS51	LVR51	Setting of status of timer output F/F
0	0	Not affected
0	1	Timer output F/F reset to 0
1	0	Timer output F/F set to 1
1	1	Setting prohibited

TMC511	Other than PWM mode (TMC516 = 0)	PWM mode (TMC516 = 1)
	Control of timer F/F	Selection of active level
0	Inversion operation disabled	Active high
1	Inversion operation enabled	Active low

TOE51	Control of timer output
0	Output (port mode) disabled
1	Output enabled

Remarks 1. The PWM output becomes inactive when TCE51 = 0 in the PWM mode.

2. LVS51 and LVR51 are 0 when read after data has been set.

(3) Port mode register 3 (PM3)

This register sets port 3 I/O in 1-bit units.

When using the P34/TI50 pin for timer input, set PM34 to 1.

When using the P35/TI51 pin for timer input, set PM35 to 1.

When using the P130/TO50 pin for timer output, set the output latch of P130 to 0.

When using the P131/TO51 pin for timer output, set the output latch of P131 to 0.

PM3 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM3 to FFH.

Figure 7-7. Format of Port Mode Register 3 (PM3)

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

PM3n	Selection of P3n pin I/O mode (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

7.4 Operations of 8-Bit Timer/Event Counters 50, 51

7.4.1 Operation as interval timer (8-bit)

The 8-bit timer/event counter operates as an interval timer that repeatedly generates an interrupt request at the interval specified by the count value preset to 8-bit compare register 5n (CRn).

When the count value of 8-bit timer counter 5n (TM5n) matches the value set to CR5n, the value of TM5n is cleared to 0. TM5n continues counting and an interrupt request signal (INTTM5n) is generated.

The count clock of TM5n can be selected by using bits 0 to 2 (TCL5n0 to TCL5n2) of timer clock select register 5n (TCL5n).

[Setting]

<1> Set each register.
- TCL5n: Select a count clock.
- CR5n: Set a compare value.
- TMC5n: Select count operation stop and clear & start mode on match between TM5n and CR5n (TMC5n = 0000×××0B: × = don't care).

<2> The count operation is started when TEC5n is set to 1.

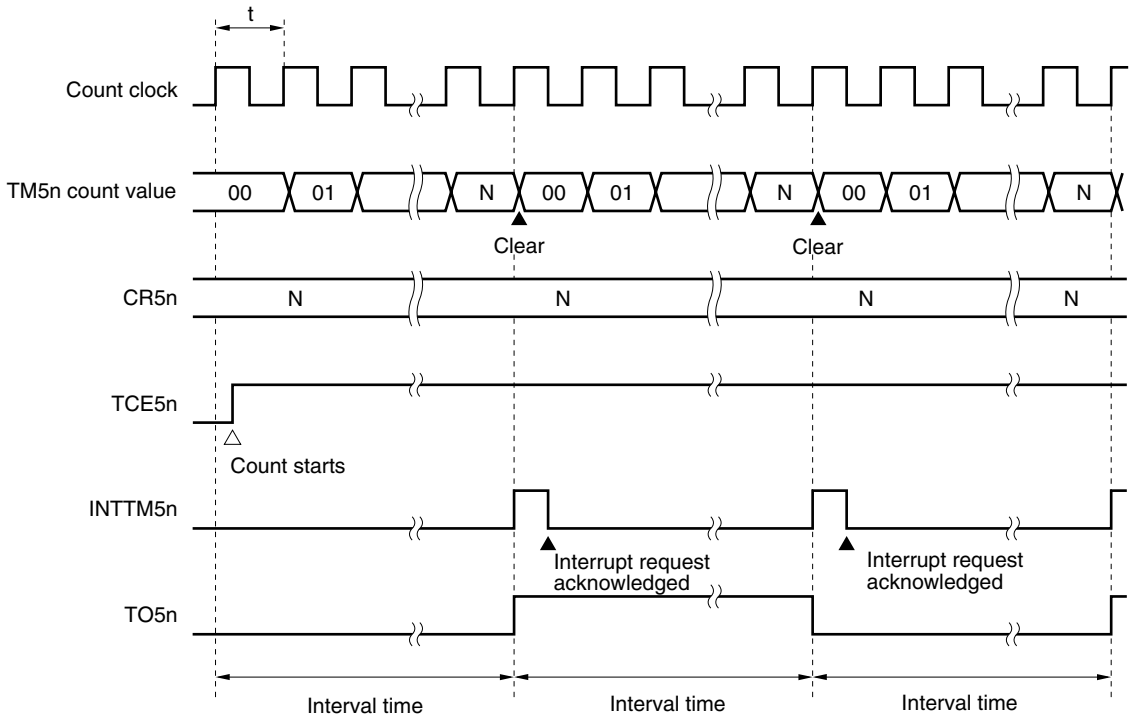
<3> INTTM5n is generated when the values of TM5n and CR5n match (TM5n is cleared to 00H).

<4> After that, INTTM5n is repeatedly generated at fixed intervals. To stop the count operation, clear TCE5n to 0.

Remark n = 0 or 1

Figure 7-8. Timing of Interval Timer Operation (1/3)

(a) Basic operation

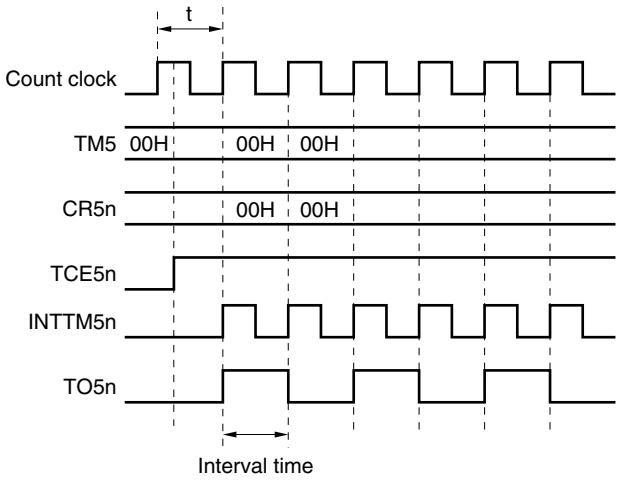


Remarks 1. Interval time = (N + 1) × t
N = 00H to FFH

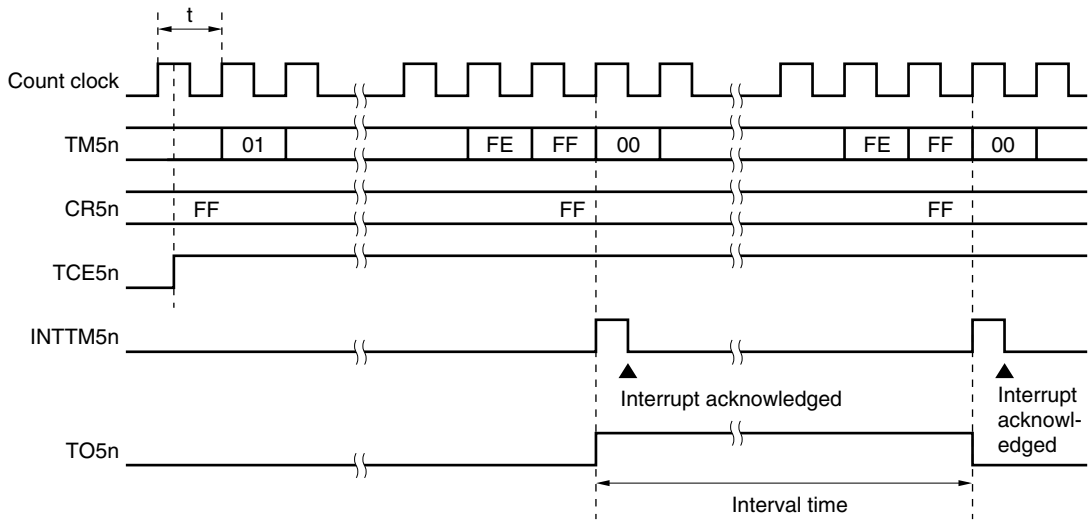
2. n = 0 or 1

Figure 7-8. Timing of Interval Timer Operation (2/3)

(b) When CR5n = 00H



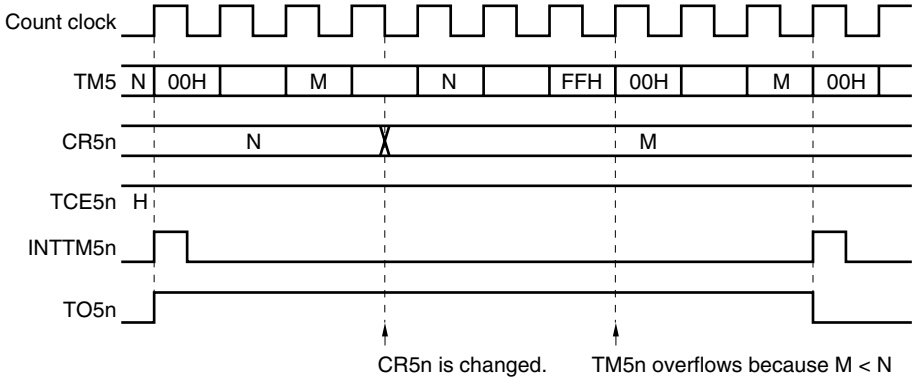
(c) When CR5n = FFH



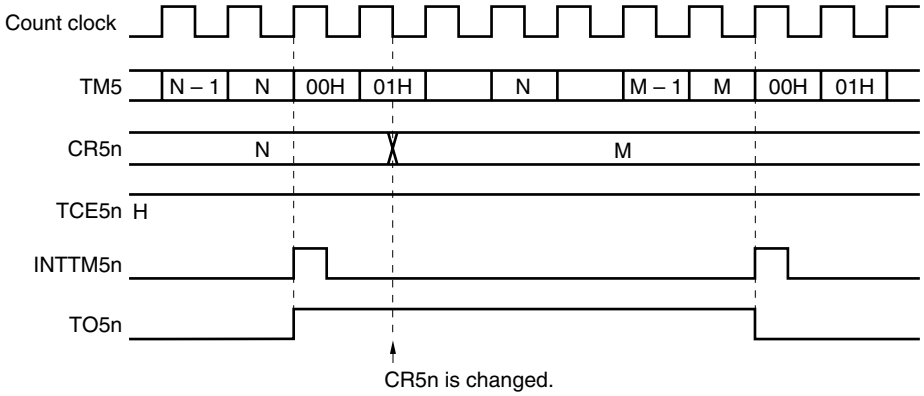
Remark n = 0 or 1

Figure 7-8. Timing of Interval Timer Operation (3/3)

(d) Operation when CR5n is changed (M < N)



(e) Operation when CR5n is changed (M > N)



Remark n = 0 or 1

7.4.2 Operation as external event counter

The external event counter counts the number of clock pulses input from an external source to the TI5n pin using 8-bit timer counter 5n (TM5n).

Each time the valid edge specified by timer clock select register 5n (TCL5n) is input to TI5n, the value of TM5n is incremented. Either the rising or falling edge can be selected as the valid edge.

When the count value of TM5n matches the value of 8-bit compare register 5n (CR5n), TM5n is cleared to 0, and an interrupt request signal (INTTM5n) is generated.

After that, each time the value of TM5n matches the value of CR5n, INTTM5n is generated.

[Setting]

<1> Set each register.

- TCL5n: Select rising or falling edge of TI5n input.
  Rising edge of TI5n → TCL5n = 00H
  Falling edge of TI5n → TCL5n = 01H
- CR5n: Set a compare value.
- TMC5n: Select count operation stop, clear & start mode on match between TM5n and CR5n, timer F/F inverted operation disable, timer output disable.
  (TMC5n = 0000 ××00B, × = don't care)

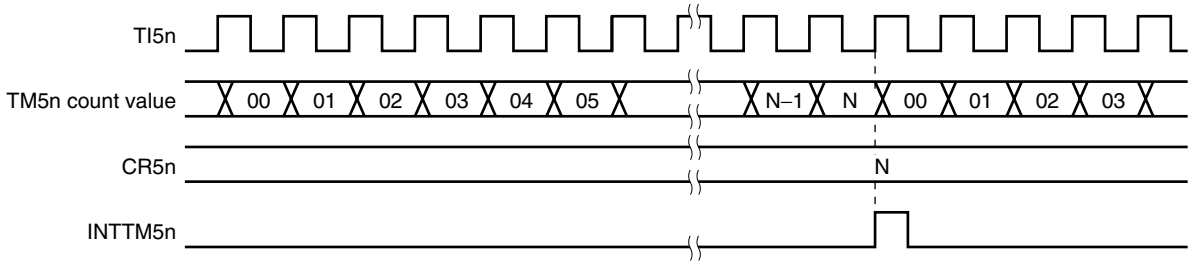
<2> When TCE5n = 1 is set, the number of pulses input from TI5n is counted.

<3> INTTM5n is generated when the values of TM5n and CR5n match (TM5n is cleared to 00H).

<4> After that, INTTM5n is generated each time when the values of TM5n and CR5n match.

Remark n = 0 or 1

Figure 7-9. Operation Timing of External Event Counter (with Rising Edge Specified)



Remarks 1. N = 00H to FFH

2. n = 0 or 1

7.4.3 Square-wave output operation (8-bit resolution)

The 8-bit timer/event counter (TM5n) can be used to output a square wave with any frequency at time intervals specified by the value preset to 8-bit compare register 5n (CR5n).

When bit 0 (TOE5n) of 8-bit timer mode control register 5n (TMC5n) is set to 1, the output status of TO5n is inverted at the interval specified by the count value preset to CR5n. In this way, a square wave (duty factor = 50%) of any frequency can be output.

[Setting]

<1> Set each register.

- Reset the port latches (P130 and P131) to “0”.
- TCL5n: Select a count clock.
- CR5n: Set a compare value.
- TMC5n: Select clear & start mode on match between TM5n and CR5n.

LVS5n	LVR5n	Setting of status of timer output F/F
1	0	High-level output
0	1	Low-level output

Enable inverting the timer F/F.

Enable the timer output → TOE5n = 1.

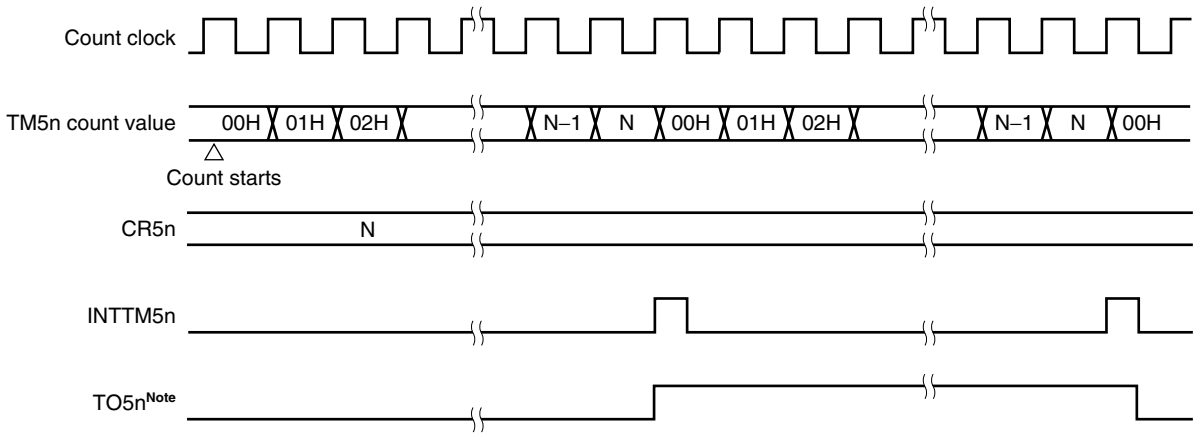
<2> When TCE5n is set to 1, the count operation is started.

<3> When the value of TM5n matches as the value of CR5n, the timer output F/F is inverted.
In addition, INTTM5n is generated, and TM5n is cleared to 00H.

<4> After that, the timer output F/F is inverted at fixed intervals, and a square wave is output from TO5n.

Remark n = 0 or 1

Figure 7-10. Timing of Square-Wave Output Operation



Note The initial value of TO5n output can be set by using bits 2 and 3 (LVR5n and LVS5n) of 8-bit timer mode control register 5n (TMC5n).

Remark n = 0 or 1

7.4.4 8-bit PWM output operation

The 8-bit timer/event counter can be used for PWM output when bit 6 (TMC5n6) of 8-bit timer mode control register 5n (TMC5n) is set to 1.

A pulse with the duty factor determined by the value set to 8-bit compare register 5n (CR5n) is output from TO5n.

Set the active level width of the PWM pulse to CR5n. The active level is selected by bit 1 (TMC5n) of TMC5n.

The count clock can be selected by bits 0 to 2 (TCL5n0 to TCL5n2) of timer clock select register n (TCL5n).

PWM output can be enabled or disabled by bit 0 (TOE5n) of TMC5n.

Caution The value of CR5n can be rewritten only once in once cycle in the PWM mode.

Remark n = 0 or 1

(1) Basic operation of PWM output

[Setting]

<1> Set each register.

- Reset the port latches (P130 and P131)[Note] to “0”.
- TCL5n: Select a count clock.
- CR5n: Set a compare value.
- TMC5n: Select count operation stop, PWM mode, timer output F/F not affected.

TMC5n1	Active level selection
0	Active high
1	Active low

Enable timer output.

(TMC5n = 01000001B or 01000011B)

<2> The count operation is started when TCE5n = 1 is set.

To stop the count operation, set TCE5n to 0.

Note 8-bit timer/event counter 50: P130

8-bit timer/event counter 51: P131

[Operation of PWM output]

<1> When the count operation is started, the PWM output (output from TO5n) remains inactive until an overflow occurs.

<2> When an overflow occurs, the active level is output. This active level is output until the value of CR5n matches the count value of 8-bit timer counter 5n (TM5n).

<3> The PWM output remains inactive after CR5n and the count value of TM5n have matched, until an overflow occurs again.

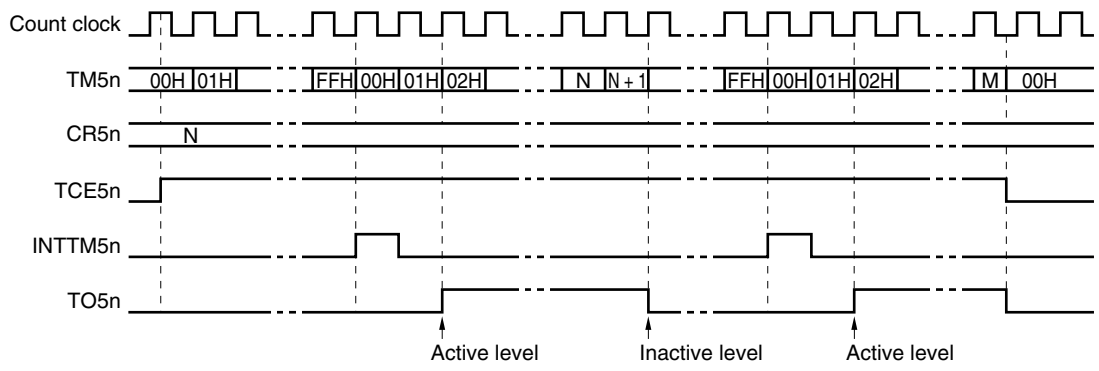
<4> After that, <2> and <3> are repeated until the count operation is stopped.

<5> When the count operation is stopped because TCE5n is cleared to 0, the PWM output becomes inactive.

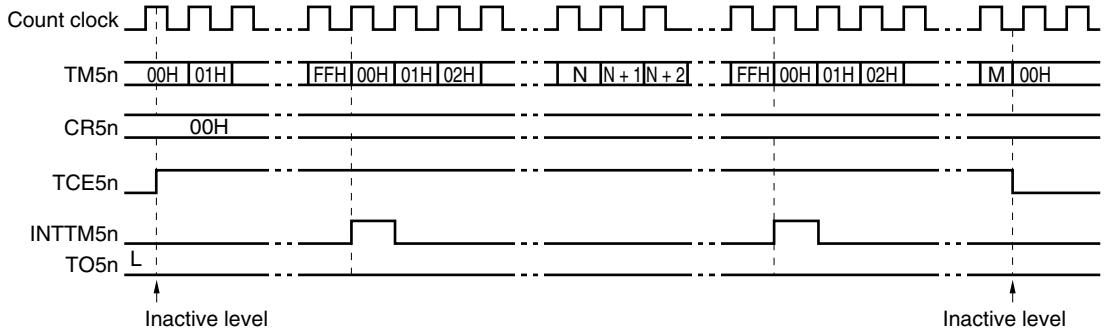
Remark n = 0 or 1

Figure 7-11. Operation Timing of PWM Output

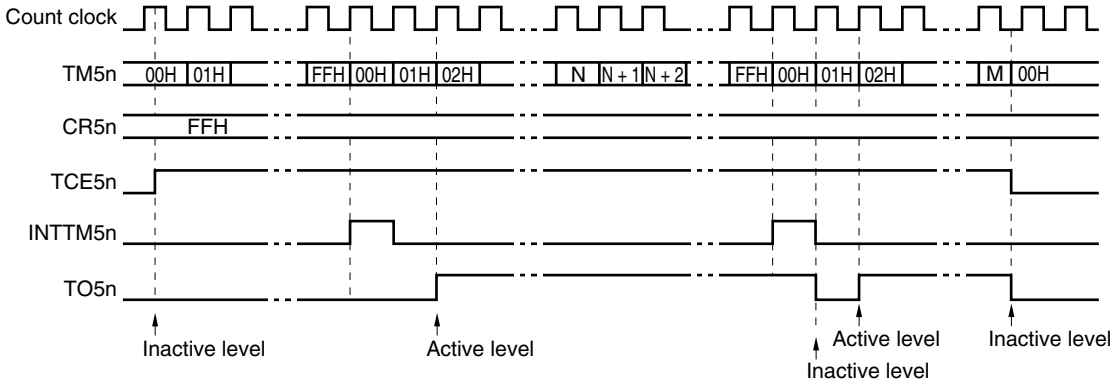
(a) Basic operation (when active level = H)



(b) When CR5n = 0



(c) When CR5n = FFH

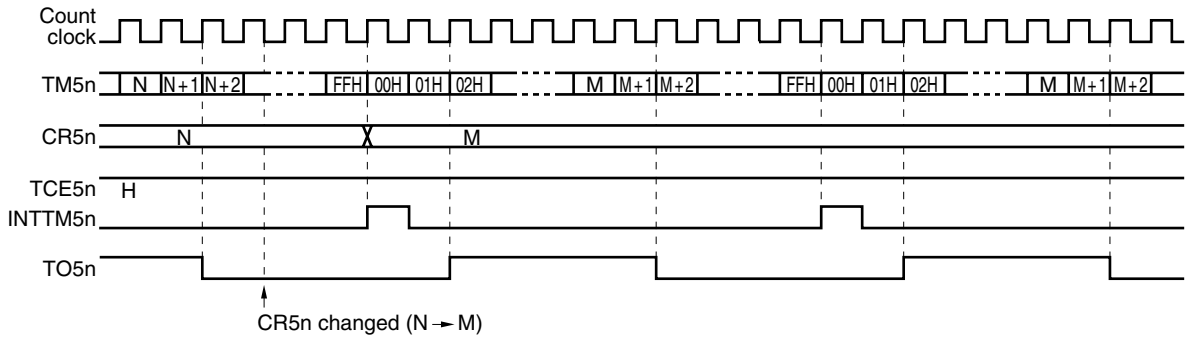


Remark n = 0 or 1

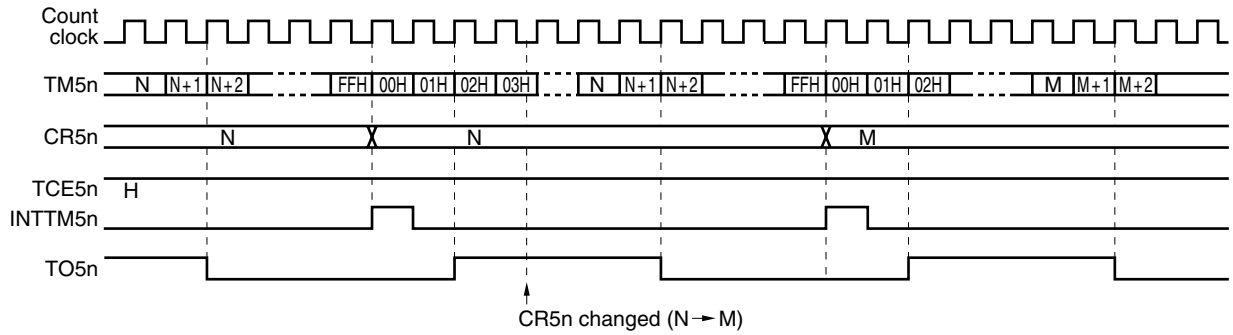
(2) Operation when CR5n is changed

Figure 7-12. Timing of Operation When CR5n Is Changed

(a) If value of CR5n is changed from N to M when TM5n > CR5n



(b) If value of CR5n is changed from N to M when TM5n < CR5n



Caution The value of CR5n can be changed only once in one cycle in the PWM mode.

Remark n = 0 or 1

7.4.5 Operation as interval timer (16-bit)

The 8-bit timer/event counters are used together in 16-bit timer/counter mode when bit 4 (TMC514) of 8-bit timer mode control register 51 (TM51) is set to 1.

In this mode, the 8-bit timer/event counters are used as a 16-bit interval timer that repeatedly generates an interrupt request at intervals specified by the count value preset to the 8-bit compare registers (CR50 and CR51).

At this time, CR50 serves as the lower 8 bits of the 16-bit compare register, and CR51 serves as the higher 8 bits.

[Setting]

<1> Set each register.

- TCL50: Select a count clock for TM50.
  The count clock for TM51, which is cascaded, does not have to be set.
- CR50 and CR51: Set compare values (each compare value can be set in a range of 00H to FFH).
- TMC50 and TMC51: Select clear & start mode on match between TM50 and CR50 (or between TM51 and CR51).

  (TM50 → TMC50 = 0000×××0B ×: don't care
  TM51 → TMC51 = 0001×××0B ×: don't care)

<2> The count operation is started by setting TCE51 of TMC51 to 1 first, and then TCE50 of TMC50 to 1.

<3> If the value of cascaded timer TM50 matches the value of CR50, INTTM50 of TM50 is generated (TM50 and TM51 are cleared to 00H).

<4> After that, INTTM50 is repeatedly generated at fixed intervals.

Cautions 1. Be sure to set the compare registers (CR50 and CR51) after stopping the operation of both timers (TCE50 = TCE51 = 0).

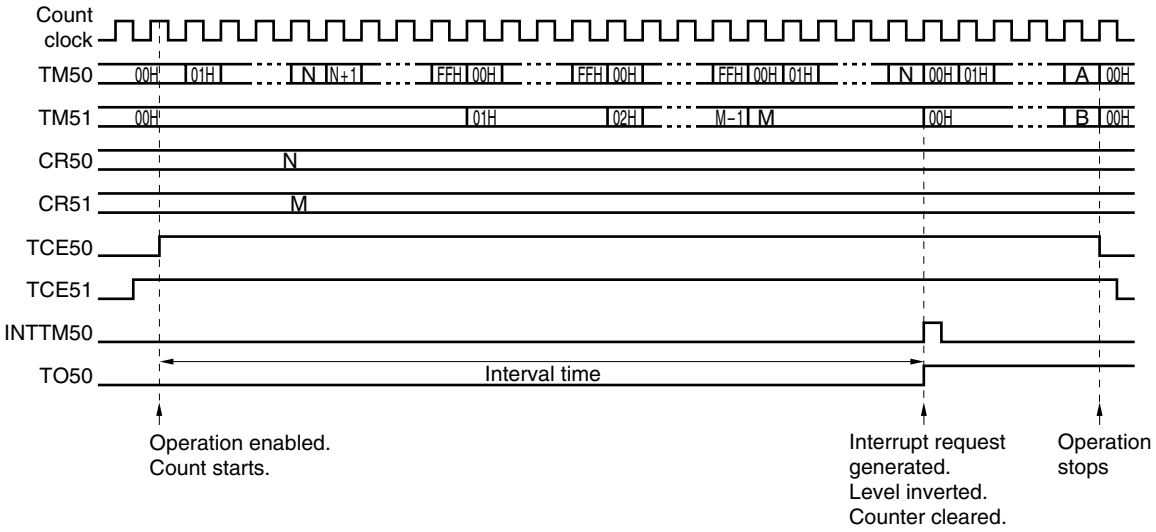
2. Even if the 8-bit timer/counters are cascaded, INTTM51 of TM51 is generated when the count value of TM51 matches CR51. Be sure to mask TM51 to disable this interrupt.

3. Set TCE50 and TCE51 in the order of TM51 then TM50.

4. Counting can be restarted or stopped just by setting or resetting TCE50 of TM50 to 1 or 0.

Figure 7-13 shows a timing example in the 16-bit cascade connection mode.

Figure 7-13. 16-Bit Cascade Connection Mode



★ 7.5 Program List

Caution The following sample program is shown as an example to describe the operation of semiconductor products and their applications. Therefore, when applying the following information to your devices, design the devices after performing evaluation on your own responsibility.

7.5.1 Interval timer (8-bit)

```
/*****************************************************************************/
/*                                                                           */
/*          Timer 50 operation sample                                        */
/*    Interval timer setting example (frequency change by interrupt processing)      */
/*     data[0]: Data set flag (value changed when other than 00)             */
/*     data[1]: Set data                                                     */
/*                                                                           */
/*****************************************************************************/
#pragma sfr
#pragma EI
#pragma DI
#pragma interrupt INTTM50 intervalint rb2
 unsigned char data[2];                 /* Data area */

void main(void)
{
        PCC = 0x0;                      /* Set high-speed operation mode */
        data[0] = 0;                    /* Clear data area */
        data[1] = 0;

                                        /* Set port        */
        P13 = 0b11111110;               /* Clear P130 when using TO50 */

                                        /* Set interrupt                 */
        TMMK50 = 0;                     /* Clear INTTM50 interrupt mask */
                                        /* Set timer 50                        */
        TMC50 = 0b00000111;             /* Clear & start mode, initial value L */
        CL50 = 0b00000101;              /* Both rising and falling edges */
                                        /* Count clock is fx/2^6 */
        CR50 = 98;                      /* Set interval to 1 ms as initial value */
        TCE50 = 1;                      /* Timer start */
        EI();
        while(1);                       /* Dummy loop */
}

/* INTTM50 interrupt function */
void intervalint()
{
        if(data[0] != 0)
        {
                CR50 = data[1];         /* Set new set value */
                data[0] = 0;            /* Clear request flag */
        }

}
```

7.5.2 External event counter

```
/************************************************************/
/*                                                          */
/*           Timer 50 operation sample                      */
/*       Event counter setting example                      */
/*        data: Count up flag                               */
/*                                                          */
/************************************************************/
#pragma sfr
#pragma EI
#pragma DI
#pragma interrupt INTTM50 intervalint rb2
        unsigned char data;             /* Data area */

void main(void)
{
        PCC = 0x0;                      /* Set high-speed operation mode */
        data = 0;                       /* Clear data area */

                                        /* Set port             */
        PM3.4 = 1;                      /* Set P34 to input */

                                        /* Set interrupt                  */
        TMMK50 = 0;                     /* Clear INTTM50 interrupt mask  */
                                        /* Set timer 50                      */
        TMC50 = 0b00000000;             /* Clear & start mode                */
        TCL50 = 0b00000001;             /* Specify rising edge of TI50       */
        CR50 = 0x10;                    /* Set N = 16 as initial value     */
        TCE50 = 1;                      /* Timer start                        */
        EI();

/************************************************************/
/*                                                          */
/*    Describe the processing to be executed                */
/*                                                          */
/************************************************************/

        while(data == 0);               /* Wait for count up */


/************************************************************/
/*                                                          */
/*    Describe the processing after count up below          */
/*                                                          */
/************************************************************/
}

/* INTTM50 interrupt function */
void intervalint()
{
        data = 0xff;                    /* Set count up flag   */
        TCE50 = 0;                      /* Timer stop          */
}
```

7.5.3 Interval timer (16-bit)

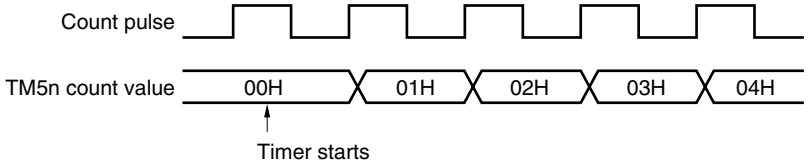
```
/************************************************************/
/*                                                          */
/*              Timer 5 operation sample                    */
/*               Cascade connection setting example         */
/*                                                          */
/************************************************************/
#pragma sfr
#pragma EI
#pragma DI
#define intervalTM5 98                  /* Cycle data to be set to CR */
#pragma interrupt INTTM50 ppgint rb2
        unsigned char ppgdata[2];       /* Data area to be set to timer 5 */

void main(void)
{
        int interval;
        interval = intervalTM5;
        PCC = 0x0;                      /* Select high-speed operation mode */
        ppgdata[0] = 0;                 /* Clear CR50 data */
        ppgdata[1] = 0;                 /* Clear CR51 data */
                                        /* Set port */
        P13 = 0b11111110;               /* Clear P130 when using TO50 */

                                        /* Set interrupt */
        TMMK50 = 0;                     /* Clear INTTM50 interrupt mask */
        TMMK51 = 1;                     /* Set INTTM51 interrupt mask */
                                        /* Set timer 5 */
        TCL50 = 0b00000101;             /* Count clock is fx/2^6 */
        CR50 = interval & 0xff;         /* Set lower compare register to CR50 */
        CR51 = interval >> 8;           /* Set higher compare register to CR51 */
        TMC50 = 0b00000111;             /* Inverted on match, initial value L */
        TMC51 = 0b00010000;             /* Cascade mode */
        TCE51 = 1;
        TCE50 = 1;                      /* Timer starts */
        EI();

        while(1);
}

/* Timer 5 interrupt function */
void ppgint()
{
        unsigned int work;
        work = ppgdata[0]+ppgdata[1]*0x100;
        if (work != 0)
        {
                TCE50 =0;
                CR51 = work >> 8;
                CR50 = work & 0xff;
                ppgdata[0] = 0;
                ppgdata[1] = 0;

                if (work != 0xffff)
                {
                        TCE50 = 1; /* Timer resumes */
                }
        }
}
```

7.6 Notes on 8-Bit Timer/Event Counters 50, 51

(1) Error on starting timer

An error of up to 1 clock occurs after the timer has been started until a match signal is generated. This is because 8-bit timer counter 5n (TM5n) is started asynchronously to the count pulse.

Figure 7-14. Start Timing of 8-Bit Timer Counter 5n



(2) STOP mode setting

Be sure to clear TCE5n to 0 to set the STOP status, except when TI5n input is selected. Otherwise, the timer may malfunction when the system clock starts oscillating.

(3) Reading TM5n (n = 0 or 1) during timer operation

When TM5n is read during operation, the count clock is temporarily stopped. Therefore, select a count clock with a high/low level longer than two cycles of the CPU clock. For example, when the CPU clock (f_{CPU}) is f_X, the count clock to be selected should be $f_X/4$ or less in order that TM5n can be read.

Remark n = 0 or 1

CHAPTER 8 BASIC TIMER

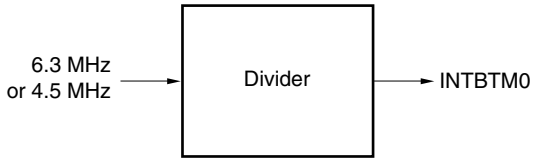
The basic timer is used for time management during program execution.

8.1 Function of Basic Timer

The basic timer generates an interrupt request signal (INTBTM0) at time intervals of 100 ms.

8.2 Configuration of Basic Timer

Figure 8-1. Block Diagram of Basic Timer



Caution With the μPD178078 and 178098A Subseries, a 4.5 MHz crystal resonator can be used in addition to the 6.3 MHz crystal resonator. If the system clock is used at a frequency of 4.5 MHz, set bit 0 (DTSCK0) of the DTS system clock select register (DTSCK) to 1 (refer to Note in 5.1).
Even if the system clock (6.3 or 4.5 MHz) is changed, the timing shown in Figure 8-2 Timing of Basic Timer Operation is not changed.
The first interrupt request signal (INTBTM0) after the DTSCK0 flag has been set is generated within 100 to 140 ms. The second signal and those that follow are generated at intervals of 100 ms.

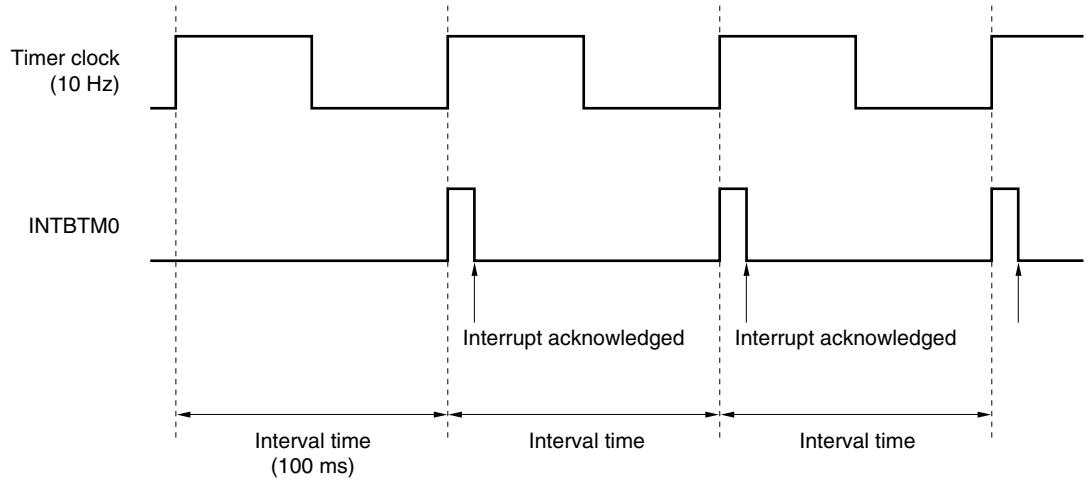
8.3 Operation of Basic Timer

An example of the operation of the basic timer is shown below.

In this example, the basic timer operates as an interval timer that repeatedly generates an interrupt at time intervals of 100 ms. An interrupt request signal (INTBTM0) is generated every 100 ms.

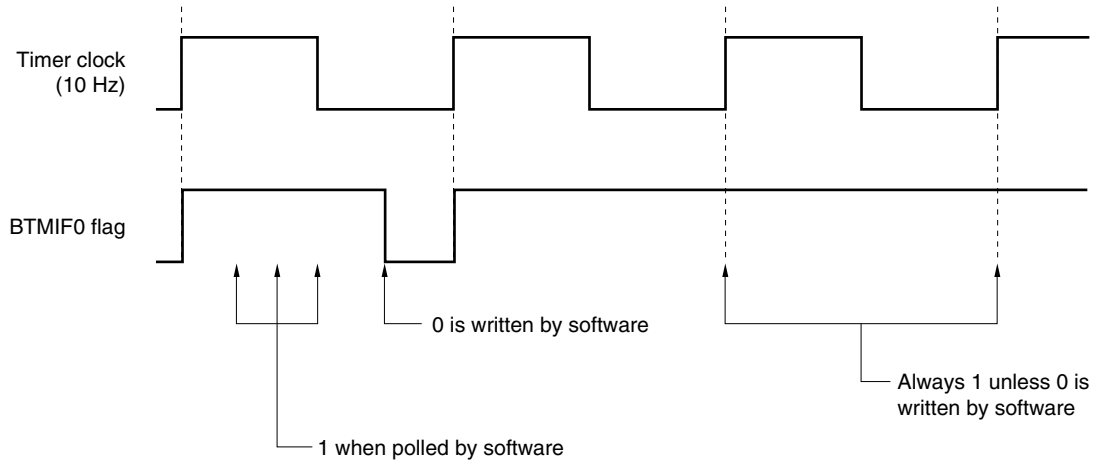
The timer clock frequency is 10 Hz.

Figure 8-2. Timing of Basic Timer Operation



By polling the interrupt request flag (BTMIF0) of this basic timer by software, time management can be carried out. Note that BTMIF0 is not a Read & Reset flag.

Figure 8-3. Operation Timing to Poll BTMIF0 Flag



For the registers controlling the basic timer, refer to **CHAPTER 18 INTERRUPT FUNCTIONS**.

CHAPTER 9 WATCHDOG TIMER

9.1 Watchdog Timer Functions

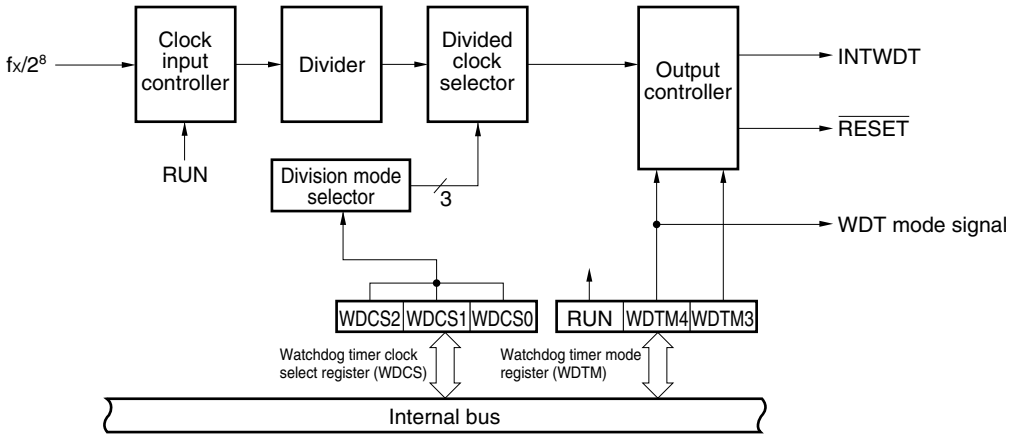
The watchdog timer has the following functions.

- Watchdog timer
- Interval timer

Caution Select the watchdog timer mode or the interval timer mode using the watchdog timer mode register (WDTM). (The watchdog timer and interval timer cannot be used simultaneously.)

Figure 9-1 shows a block diagram of the watchdog timer.

Figure 9-1. Block Diagram of Watchdog Timer



(1) Watchdog timer mode

An inadvertent program loop is detected. Upon detection of the inadvertent program loop, a non-maskable interrupt request or reset can be generated.

Table 9-1. Watchdog Timer Inadvertent Program Loop Detection Time

Inadvertent Program Loop Detection Time
$2^{12} \times 1/f_X$ (650 μs)
$2^{13} \times 1/f_X$ (1.30 ms)
$2^{14} \times 1/f_X$ (2.60 ms)
$2^{15} \times 1/f_X$ (5.20 ms)
$2^{16} \times 1/f_X$ (10.4 ms)
$2^{17} \times 1/f_X$ (20.8 ms)
$2^{18} \times 1/f_X$ (41.6 ms)
$2^{20} \times 1/f_X$ (166 ms)

Remark f_X: System clock oscillation frequency
( ): f_X = 6.3 MHz.

(2) Interval timer mode

Interrupt requests are generated at the preset time intervals.

Table 9-2. Interval Time

Interval Time
$2^{12} \times 1/f_X$ (650 μs)
$2^{13} \times 1/f_X$ (1.30 ms)
$2^{14} \times 1/f_X$ (2.60 ms)
$2^{15} \times 1/f_X$ (5.20 ms)
$2^{16} \times 1/f_X$ (10.4 ms)
$2^{17} \times 1/f_X$ (20.8 ms)
$2^{18} \times 1/f_X$ (41.6 ms)
$2^{20} \times 1/f_X$ (166 ms)

Remark f_X: System clock oscillation frequency
( ): f_X = 6.3 MHz.

9.2 Watchdog Timer Configuration

The watchdog timer consists of the following hardware.

Table 9-3. Configuration of Watchdog Timer

Item	Configuration
Control registers	Watchdog timer clock select register (WDCS) Watchdog timer mode register (WDTM)

9.3 Watchdog Timer Control Registers

The following two registers are used to control the watchdog timer.

- Watchdog timer clock select register (WDCS)
- Watchdog timer mode register (WDTM)

(1) Watchdog timer clock select register (WDCS)

This register sets the watchdog timer and overflow time of the interval timer.

WDCS is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets WDCS to 00H.

Figure 9-2. Format of Watchdog Timer Clock Select Register (WDCS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
WDCS	0	0	0	0	0	WDCS2	WDCS1	WDCS0	FF42H	00H	R/W

WDCS2	WDCS1	WDCS0	Watchdog timer/interval timer overflow time
0	0	0	2^{12}/fx (650 μs)
0	0	1	2^{13}/fx (1.30 ms)
0	1	0	2^{14}/fx (2.60 ms)
0	1	1	2^{15}/fx (5.20 ms)
1	0	0	2^{16}/fx (10.4 ms)
1	0	1	2^{17}/fx (20.8 ms)
1	1	0	2^{18}/fx (41.6 ms)
1	1	1	2^{20}/fx (166 ms)

Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 6.3 MHz.

(2) Watchdog timer mode register (WDTM)

This register sets the watchdog timer operating mode and enables/disables counting.

WDTM is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets WDTM to 00H.

Figure 9-3. Format of Watchdog Timer Mode Register (WDTM)

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
WDTM	RUN	0	0	WDTM4	WDTM3	0	0	0	FFF9H	00H	R/W

RUN	Count enable/disable
0	Count stop
1	Counter is cleared and counting starts.

WDTM4	WDTM3	Selection of watchdog timer operating mode[Note 2]
0	×	Interval timer mode[Note 3] (Maskable interrupt occurs upon generation of an overflow.)
1	0	Watchdog timer mode 1 (Non-maskable interrupt occurs upon generation of an overflow.)
1	1	Watchdog timer mode 2 (Reset operation is activated upon generation of an overflow.)

Notes 1. Once set to 1, RUN cannot be cleared to 0 by software. Therefore, use $\overline{\text{RESET}}$ input to clear RUN to 0.

2. Once set to 1, WDTM3 and WDTM4 cannot be cleared to 0 by software.

3. WDTM starts interval timer operation as soon as RUN is set to 1.

Caution When RUN is set to 1 so that the watchdog timer is cleared, the actual overflow time is up to 0.5% shorter than the time set by the watchdog timer clock select register (WDCS).

Remark ×: Don't care

9.4 Watchdog Timer Operations

9.4.1 Operation as watchdog timer

When bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 1, the watchdog timer operates to detect an inadvertent program loop.

The watchdog timer count clock (inadvertent program loop detection time interval) can be selected using bits 0 to 2 (WDCS0 to WDCS2) of watchdog timer clock select register 2 (WDCS). The watchdog timer starts counting when bit 7 (RUN) of WDTM is set to 1. After the watchdog timer starts counting, set RUN to 1 again within the set inadvertent program loop time interval to clear the watchdog timer and start counting again. If RUN is not set to 1 and the inadvertent program loop detection time elapses, a system reset or non-maskable interrupt request is generated according to the value of WDTM bit 3 (WDTM3).

The watchdog timer continues operating in the HALT mode but it stops in the STOP mode. Thus, set RUN to 1 before the STOP mode is set, clear the watchdog timer, and then execute the STOP instruction.

Caution The actual inadvertent program loop detection time may be shorter than the set time by up to 0.5%.

Table 9-4. Watchdog Timer Inadvertent Program Loop Detection Time

Inadvertent Program Loop Detection Time
$2^{12} \times 1/f_X$ (650 μs)
$2^{13} \times 1/f_X$ (1.30 ms)
$2^{14} \times 1/f_X$ (2.60 ms)
$2^{15} \times 1/f_X$ (5.20 ms)
$2^{16} \times 1/f_X$ (10.4 ms)
$2^{17} \times 1/f_X$ (20.8 ms)
$2^{18} \times 1/f_X$ (41.6 ms)
$2^{20} \times 1/f_X$ (166 ms)

Remarks 1. f_X: System clock oscillation frequency

2. ( ): f_X = 6.3 MHz.

9.4.2 Operation as interval timer

The watchdog timer operates as an interval timer that generates interrupt requests repeatedly at an interval of the preset count value when bit 3 (WDTM3) and bit 4 (WDTM4) of the watchdog timer mode register (WDTM) are set to 1 and 0, respectively.

The count clock (interval time) can be selected by using bits 0 to 2 (WDCS0 to WDCS2) of the watchdog timer clock select register (WDCS). By setting bit 7 (RUN) of WDTM to 1, the watchdog timer starts operating as an interval timer.

When the watchdog timer operates as interval timer, the interrupt mask flag (WDTMK) and priority specification flag (WDTPR) are validated and the maskable request interrupt (INTWDT) can be generated. Among the maskable interrupt requests, INTWDT has the highest default priority.

The interval timer continues operating in the HALT mode but it stops in STOP mode. Thus, set RUN to 1 before the STOP mode is set, clear the interval timer, and then execute the STOP instruction.

Cautions 1. Once bit 4 (WDTM4) of WDTM is set to 1 (with the watchdog timer mode selected), the interval timer mode is not set unless $\overline{\text{RESET}}$ is input.

2. The interval time just after being set by WDTM may be shorter than the set time by up to 0.5%.

Table 9-5. Interval Timer Interval Time

Interval Time
$2^{12} \times 1/f_X$ (650 μs)
$2^{13} \times 1/f_X$ (1.30 ms)
$2^{14} \times 1/f_X$ (2.60 ms)
$2^{15} \times 1/f_X$ (5.20 ms)
$2^{16} \times 1/f_X$ (10.4 ms)
$2^{17} \times 1/f_X$ (20.8 ms)
$2^{18} \times 1/f_X$ (41.6 ms)
$2^{20} \times 1/f_X$ (166 ms)

Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 6.3 MHz.

CHAPTER 10 BUZZER OUTPUT CONTROLLERS

10.1 Functions of Buzzer Output Controllers

The μPD178078 and 178098A Subseries have the following two types of buzzer output controllers.

- BEEP0
- BUZ

BEEP0 outputs a square wave of the buzzer frequency selected by BEEP frequency select register 0 (BEEPCL0) from the BEEP0/P36 pin.

BUZ outputs a square wave of the buzzer frequency selected by the clock output select register (CKS) from the BUZ/P37 pin.

Figures 10-1 and 10-2 show the block diagrams of BEEP0 and BUZ.

Figure 10-1. Block Diagram of BEEP0

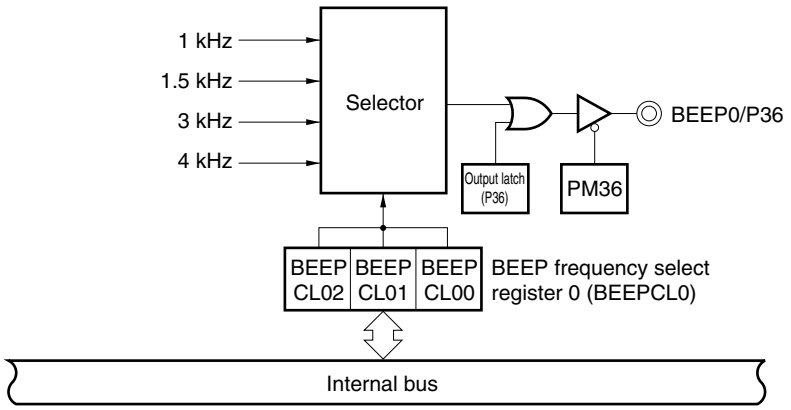
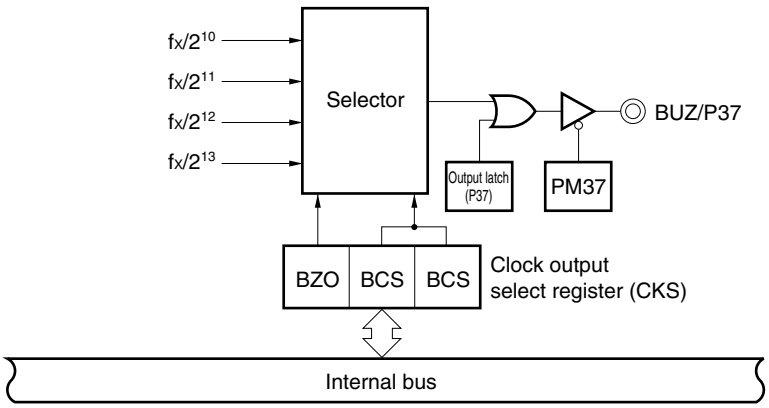


Figure 10-2. Block Diagram of BUZ



Remark fx: System clock oscillation frequency

10.2 Configuration of Buzzer Output Controllers

The buzzer output controllers consist of the following hardware.

Table 10-1. Configuration of Buzzer Output Controllers

(1) BEEP0

Item	Configuration
Control register	BEEP frequency select register 0 (BEEPCL0)

(2) BUZ

Item	Configuration
Control register	Clock output select register (CKS)

10.3 Registers Controlling Buzzer Output Controllers

10.3.1 BEEP0

BEEP0 is controlled by the following register.

- BEEP frequency select register 0 (BEEPCL0)

(1) BEEP frequency select register 0 (BEEPCL0)

This register selects the frequency of buzzer output.

BEEPCL0 is set by a 1-bit or 8-bit memory manipulation instruction.

This register is cleared to 00H after reset.

Figure 10-3. Format of BEEP Frequency Select Register 0 (BEEPCL0)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BEEPCL0	0	0	0	0	0	BEEPCL02	BEEPCL01	BEEPCL00	FF41H	00H	R/W

BEEPCL02	BEEPCL01	BEEPCL00	Selection of frequency of BEEP0 output
0	×	×	Buzzer output (port function) disabled
1	0	0	1 kHz
0	0	1	3 kHz
1	1	0	4 kHz
1	1	1	1.5 kHz

Cautions 1. The selected clock may not be correctly output during the period of 1 cycle immediately after the output clock has been changed.

2. The frequency of BEEP0 output (1 kHz, 1.5 kHz, 3 kHz, or 4 kHz) does not change even if the system clock (6.3 MHz or 4.5 MHz) is changed.

10.3.2 BUZ

BUZ is controlled by the following register.

- Clock output select register (CKS)

Figure 10-4. Format of Clock Output Select Register (CKS)

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CKS	BZOE	BCS1	BCS0	0	0	0	0	0	FF40H	00H	R/W

BZOE	BUZ output enable/disable
0	Low-level output
1	Buzzer output enabled

BCS1	BCS0	Selection of output clock of BUZ
0	0	$f_X/2^{10}$ (6.15 kHz)
0	1	$f_X/2^{11}$ (3.08 kHz)
1	0	$f_X/2^{12}$ (1.54 kHz)
1	1	$f_X/2^{13}$ (769 Hz)

f_X: System clock oscillation frequency

( ): f_X = 6.3 MHz

10.4 Operation of Buzzer Output Controllers

The buzzer frequency is output using the following procedure.

(1) BEEP0

<1> Select the buzzer output frequency by using bits 0 to 2 (BEEPCL00 to BEEPCL02) of BEEP frequency select register 0 (BEEPCL0).
<2> Reset the output latch of P36 to 0.
<3> Reset bit 6 (PM36) of port mode register 3 to 0 (set the output mode).

(2) BUZ

<1> Select the buzzer output frequency by using bits 5 and 6 (BCS0 and BCS1) of the clock output select register (CKS) (disable buzzer output).
<2> Set bit 7 (BZOE) of CKS to 1 and enable buzzer output.
<3> Reset the output latch of P37 to 0.
<4> Reset bit 7 (PM37) of port mode register 3 to 0 (set output mode).

CHAPTER 11 A/D CONVERTER

11.1 A/D Converter Functions

The A/D converter converts an analog input into a digital value. It consists of 8 channels (ANI0 to ANI7) with 8-bit resolution.

The conversion method is based on successive approximation and the conversion result is held in 8-bit A/D conversion result register 3 (ADCR3).

Conversion is started by setting A/D converter mode register 3 (ADM3).

Select one analog input channel from ANI0 to ANI7 and carry out A/D conversion.

When A/D conversion is completed, the next A/D conversion is started immediately. Each time an A/D conversion operation ends, an interrupt request (INTAD) is generated.

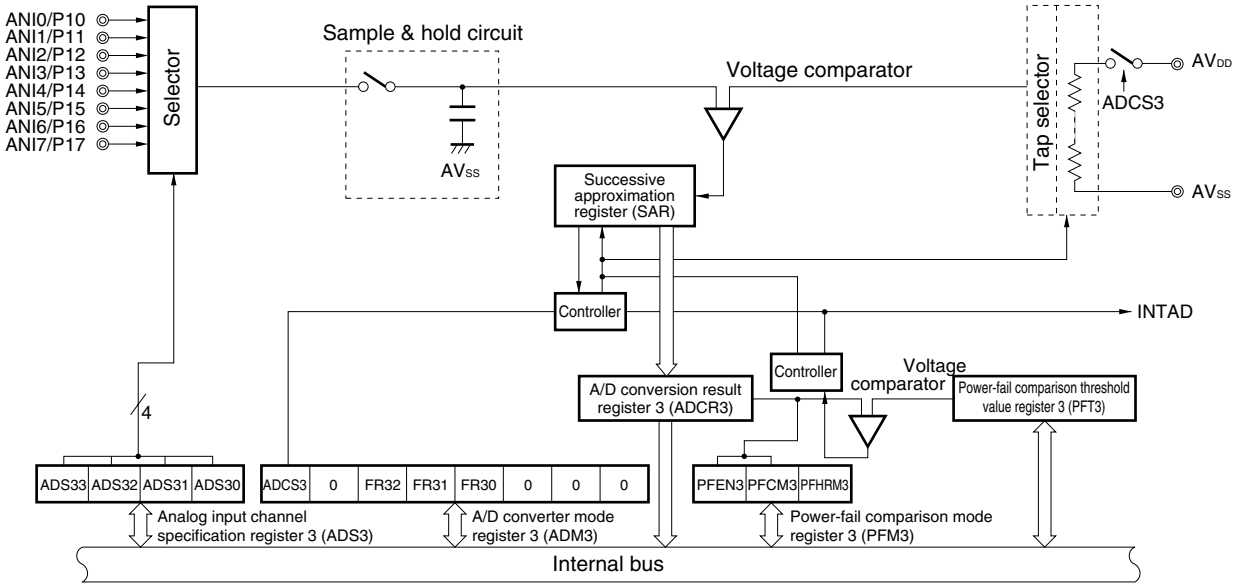
11.2 A/D Converter Configuration

The A/D converter consists of the following hardware.

Table 11-1. Configuration of A/D Converter

Item	Configuration
Analog input	8 channels (ANI0 to ANI7)
Control registers	A/D converter mode register 3 (ADM3) Analog input channel specification register 3 (ADS3) Power-fail comparison mode register 3 (PFM3)
Registers	Successive approximation register (SAR) A/D conversion result register 3 (ADCR3) Power-fail comparison threshold value register 3 (PFT3)

Figure 11-1. Block Diagram of A/D Converter



(1) Successive approximation register (SAR)

This register compares the analog input voltage value to the voltage tap (compare voltage) value applied from the series resistor string and holds the result from the most significant bit (MSB).

When up to the least significant bit (LSB) is set (end of A/D conversion), the SAR contents are transferred to the A/D conversion result register.

(2) A/D conversion result register 3 (ADCR3)

This register holds the A/D conversion result. Each time A/D conversion ends, the conversion result is loaded from the successive approximation register (SAR).

ADCR is read by an 8-bit memory manipulation instruction.

Reset input makes ADCR undefined.

Caution When data is written to A/D converter mode register 3 (ADM3) and analog input channel specification register 3 (ADS3), the contents of ADCR3 are undefined. Read the result of conversion after conversion has been completed and before writing data to ADM3 and ADS3. Otherwise, the correct conversion result may not be read.

(3) Power-fail comparison threshold value register 3 (PFT3)

This register sets the threshold value to be compared with the value of A/D conversion result register 3 (ADCR3).

PFT3 is read or written by using an 8-bit memory manipulation instruction.

(4) Sample & hold circuit

The sample & hold circuit samples the input signal of the analog input pin selected by the selector when A/D conversion starts, and holds the sampled analog input voltage value during A/D conversion.

(5) Voltage comparator

The voltage comparator compares the sampled analog input voltage to the series resistor string output voltage.

(6) Resistor string

The resistor string is connected between AV_{DD} and AV_{SS}, and generates a voltage to be compared to the analog input.

(7) ANI0 to ANI7 pins

These are 8-channel analog input pins used to input analog signals to undergo A/D conversion to the A/D converter.

Cautions 1. Use the ANI0 to ANI7 input voltages within the specified range. If a voltage of AV_{DD} or higher or AV_{SS} or lower is applied (even if within the absolute maximum ratings), the converted value of the corresponding channel becomes undefined and may adversely affect the converted values of other channels.

2. The analog input pins (ANI0 to ANI7) function alternately as input port pins (P10 to P17). When one of ANI0 to ANI7 is selected for A/D conversion, do not execute an input instruction to port 1; otherwise, the conversion resolution may drop.

If a digital pulse is applied to a pin adjacent to the pin being A/D converted, the expected A/D conversion value may not be obtained due to coupling noise. Do not apply a pulse to a pin adjacent to a pin being A/D converted.

(8) AV_{SS} pin

This is the ground pin of the A/D converter. Always use this pin at the same voltage as GND0, GND1, GND2, GNDPLL, and GNDPORT even when the A/D converter is not used.

(9) AV_{DD} pin

This is the analog power supply pin of the A/D converter. Always use this pin at the same voltage as $V_{DD}0$, V_{DD}PLL, and V_{DD}PORT even when the A/D converter is not used.

In the standby mode, the current flowing into the series resistor string can be reduced by stopping the conversion operation (by resetting bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) to 0).

11.3 Registers Controlling A/D Converter

The following three registers control the A/D converter.

- A/D converter mode register 3 (ADM3)
- Analog input channel specification register 3 (ADS3)
- Power-fail comparison mode register 3 (PFM3)

(1) A/D converter mode register 3 (ADM3)

This register selects the conversion time of the analog input to be converted and starts or stops the conversion operation.

ADM is set by a 1-bit or 8-bit memory manipulation instruction.

This register is cleared to 00H after reset.

Figure 11-2. Format of A/D Converter Mode Register 3 (ADM3)

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
ADM3	ADCS3	0	FR32	FR31	FR30	0	0	0	FF12H	00H	R/W

ADCS3	Control of A/D conversion operation
0	Conversion operation stopped
1	Conversion operation enabled

FR32	FR31	FR30	Selection of conversion time
0	0	0	288/fx (45.7 μs)
0	0	1	240/fx (38.0 μs)
0	1	0	192/fx (30.4 μs)
1	0	0	144/fx (22.8 μs)
1	0	1	120/fx (19.0 μs)
1	1	0	96/fx (15.2 μs)
Other than above			Setting prohibited

Cautions 1. The conversion result is undefined immediately after bit 7 (ADCS3) is set to 1.

2. To change the data of bits 3 to 5 (FR30 to FR32), stop the A/D conversion operation.

Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 6.3 MHz

(2) Analog input channel specification register 3 (ADS3)

This register specifies the input channel of the analog voltage to be converted.

ADS3 is set by an 8-bit memory manipulation instruction.

The value of this register is cleared to 00H after reset.

Figure 11-3. Format of Analog Input Channel Specification Register 3 (ADS3)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADS3	0	0	0	0	ADS33	ADS32	ADS31	ADS30	FF13H	00H	R/W

ADS33	ADS32	ADS31	ADS30	Specification of analog input channel
0	0	0	0	ANI0
0	0	0	1	ANI1
0	0	1	0	ANI2
0	0	1	1	ANI3
0	1	0	0	ANI4
0	1	0	1	ANI5
0	1	1	0	ANI6
0	1	1	1	ANI7
Other than above				Setting prohibited

(3) Power-fail comparison mode register 3 (PFM3)

PFM3 is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Figure 11-4. Format of Power-Fail Comparison Mode Register 3 (PFM3)

Symbol	<7>	<6>	<5>	4	3	2	1	0	Address	After reset	R/W
PFM3	PFEN3	PFCM3	PFHRM3	0	0	0	0	0	FF16H	00H	R/W

PFEN3	Power-fail comparison enable/disable
0	Power-fail comparison disabled
1	Power-fail comparison enabled

PFCM3	Selecttion of power-fail comparison mode
0	Interrupt request (INTAD) generated when ADCR3 ≥ PFT
1	Interrupt request (INTAD) generated when ADCR3 < PFT

Note PFHRM3	Selection of power-fail HALT repeat mode
0	Power-fail HALT repeat mode disabled
1	Power-fail HALT repeat mode disabled

Note When bit 5 (PFHRM3) is set to 1, power-fail comparison manipulation is enabled in the HALT mode, which means that A/D conversion is repeated until an interrupt request (INTAD) is generated (this bit is reset to 0 when INTAD is generated).

11.4 A/D Converter Operations

11.4.1 Basic operations of A/D converter

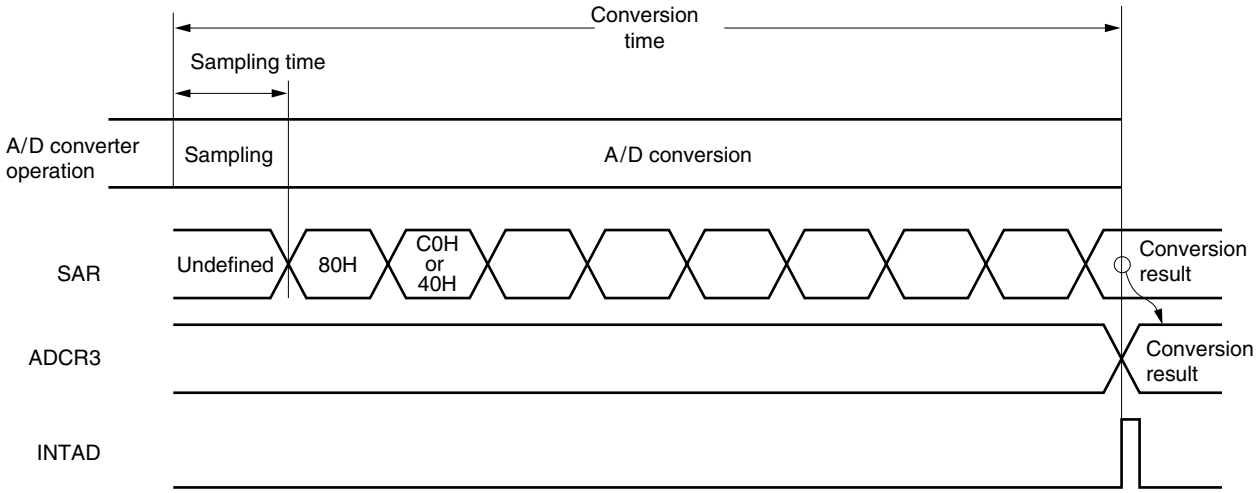
(1) Select one channel for A/D conversion using A/D converter analog input channel specification register 3 (ADS3).
(2) Sample the voltage input to the selected analog input channel using the sample & hold circuit.
(3) Sampling for the specified period of time sets the sample & hold circuit to the hold state so that the circuit holds the input analog voltage until the end of A/D conversion.
(4) Bit 7 of the successive approximation register (SAR) is set and the tap selector sets the series resistor string voltage tap to (1/2) AV_{DD}.
(5) The voltage difference between the series resistor string voltage tap and analog input is compared by a voltage comparator. If the analog input is greater than (1/2) AV_{DD}, the MSB of SAR remains set. If the input is smaller than (1/2) AV_{DD}, the MSB is reset.
(6) Next, bit 6 of SAR is automatically set and the operation proceeds to the next comparison. In this case, the series resistor string voltage tap is selected according to the preset value of bit 7 as described below.
   - Bit 7 = 1: (3/4) AV_{DD}
   - Bit 7 = 0: (1/4) AV_{DD}

   The voltage tap and analog input voltage are compared and bit 6 of SAR is manipulated with the result as follows.
   - Analog input voltage ≥ Voltage tap: Bit 6 = 1
   - Analog input voltage < Voltage tap: Bit 6 = 0

(7) Comparison of this sort continues up to bit 0 of SAR.
(8) Upon completion of the comparison of 8 bits, an effective digital resultant value remains in SAR and the resultant value is transferred to and latched in A/D conversion result register 3 (ADCR3).
   At the same time, the A/D conversion end interrupt request (INTAD) can also be generated.

Caution The value immediately after A/D conversion has been started is undefined.
Take appropriate measures, such as discarding the first conversion result by polling the A/D conversion end interrupt request (INTAD).

Figure 11-5. Basic Operation of A/D Converter



A/D conversion operations are performed continuously until bit 7 (ADCS3) of ADM is reset (0) by software.

If a write to ADM3 or ADS3 is performed during an A/D conversion operation, the conversion operation is initialized, and if the ADCS3 bit is set (1), conversion starts again from the beginning.

The value of ADCR3 is undefined after reset.

11.4.2 Input voltage and conversion results

The relationship between the analog input voltage input to the analog input pins (ANI0 to ANI7) and the A/D conversion result (the value stored in A/D conversion result register 3 (ADCR3)) is shown by the following expression.

$$ADCR3 = INT\left(\frac{V_{IN}}{AV_{DD}} \times 256 + 0.5\right)$$

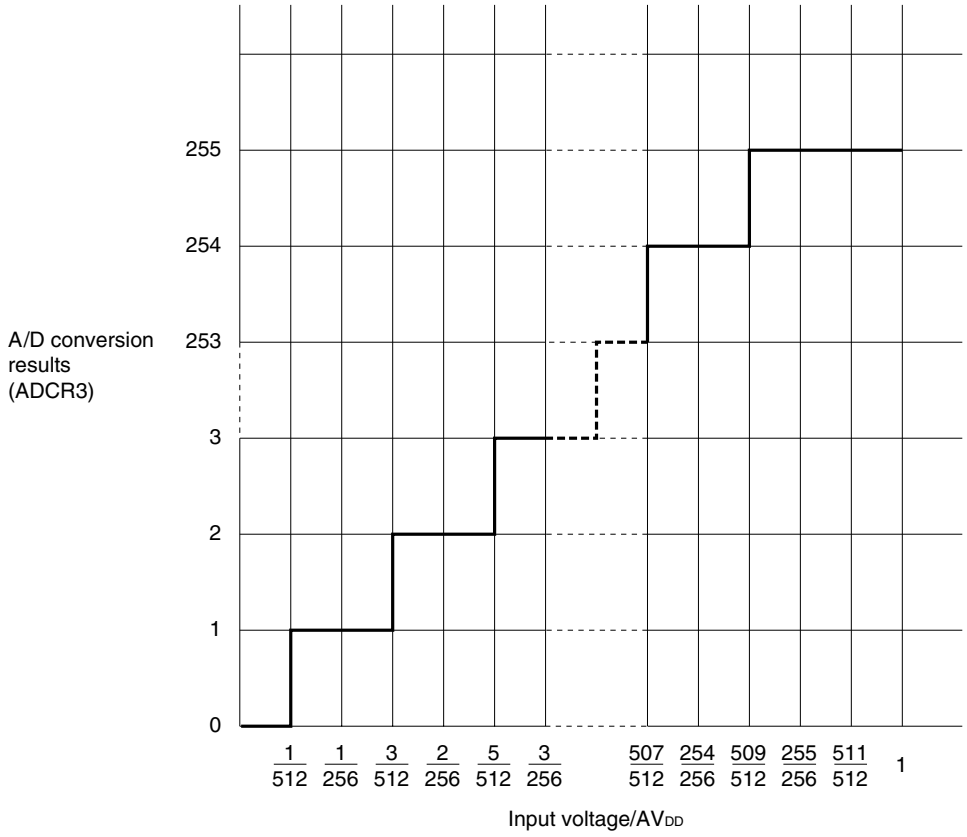
or

$$(ADCR3 - 0.5) \times \frac{AV_{DD}}{256} \leq V_{IN} < (ADCR3 + 0.5) \times \frac{AV_{DD}}{256}$$

Where, INT( ): Function which returns integer part of value in parentheses.
V_{IN}: Analog input voltage
AV_{DD}: AV_{DD} pin voltage
ADCR3: A/D conversion result register 3 (ADCR3) value

Figure 11-6 shows the relationship between the analog input voltage and the A/D conversion result.

Figure 11-6. Relationship Between Analog Input Voltage and A/D Conversion Result



11.4.3 A/D converter operating modes

The A/D converter has the following two modes.

- A/D conversion mode: In this mode, the voltage applied to the analog input pin selected from ANI0 to ANI7 is converted into a digital signal. The result of the A/D conversion is stored in A/D conversion result register 3 (ADCR3), and at the same time, an interrupt request signal (INTAD) is generated.
- Power-fail comparison mode: The digital value resulting from A/D conversion is compared with the value assigned to power-fail comparison threshold value register 3 (PFT3). If the result of the comparison matches the condition set by bit 6 (PFCM3) of power-fail comparison mode register 3 (PFM3), an interrupt request signal (INTAD) is generated.

(1) A/D conversion operation mode

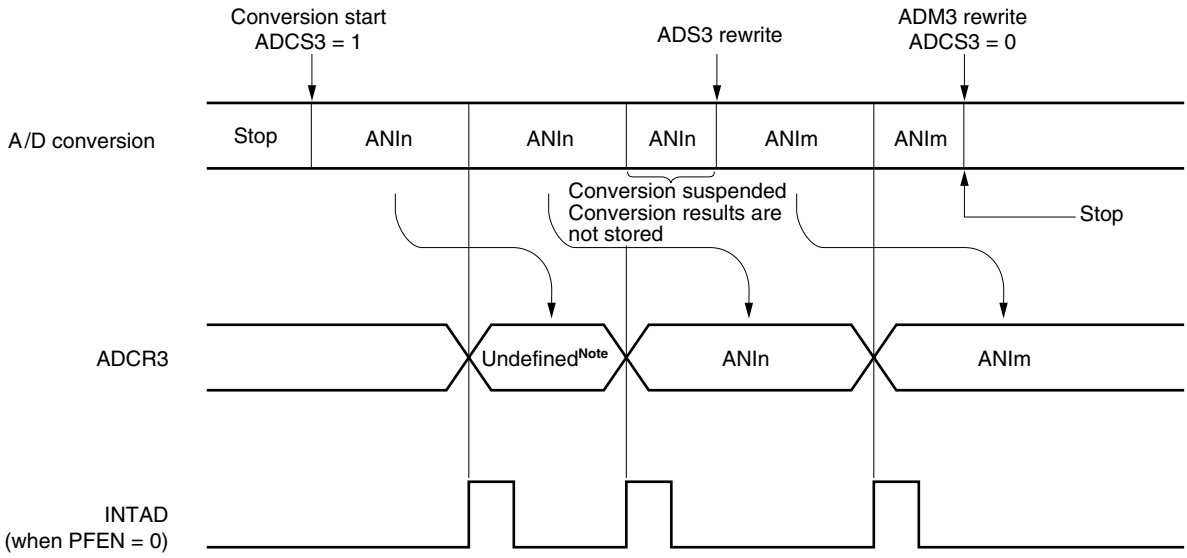
When bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) is set to 1, A/D conversion starts on the voltage applied to the analog input pins specified by bits 0 to 3 (ADS30 to ADS33) of ADS3.

At the end of the A/D conversion, the conversion result is stored in A/D conversion result register 3 (ADCR3) and the interrupt request signal (INTAD) is generated. After one A/D conversion operation is started and ends, the next A/D conversion operation starts immediately. The A/D conversion operation con-tinues repeatedly until new data is written to ADM3.

If data is written to ADCS3 again during A/D conversion, the converter suspends its A/D conversion operation and starts A/D conversion on the newly written data.

If data with ADCS3 set to 0 is written to ADM3 during A/D conversion, the A/D conversion operation stops immediately.

Figure 11-7. A/D Conversion Operation



Remarks 1. n = 0, 1, ... , 7

2. m = 0, 1, ... , 7

Note The conversion result is illegal immediately after bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) has been set to 1 (to enable conversion).

Caution Reset bit 5 (PFHRM3) of power-fail comparison mode register 3 (PFM3) to 0.

(2) Power-fail comparison mode

In the power-fail comparison mode, the digital value converted from an analog input is compared in units of 8 bits.

If the result of the comparison matches the condition set by bit 6 (PFCM3) of power-fail comparison mode register 3 (PFM3), an interrupt request (INTAD) is generated.

Note that the power-fail comparison mode can be used in the HALT mode. At this time, the HALT mode can be released by generating the interrupt request signal (INTAD) as a result of comparison (however, the A/D operation must be executed before the HALT instruction is executed).

To set the power-fail comparison mode, set bit 7 (PFEN3) of PFM3 to 1, set bit 6 (PFCM3) to the generation condition of INTAD, and assign the threshold value to be compared with the value of A/D conversion result register 3 (ADCR3) to power-fail comparison threshold value register 3 (PFT3).

By setting bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) to 1, the voltage applied to the analog input pin specified by ADS3 is converted into a digital signal. When A/D conversion has been completed, the result of the conversion is stored in ADCR3. This conversion result is compared with the value set in PFT3 and if the result of the comparison matches the condition set by bit 6 (PFCM3) of PFM3, an interrupt request signal (INTAD) is generated.

Figure 11-8. Power-Fail Comparison Threshold Value Register 3 (PFT3)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PFT3	PFT37	PFT36	PFT35	PFT34	PFT33	PFT32	PFT31	PFT30	FF15H	00H	R/W

Remark Bit 7 (PFT37) is the MSB, and bit 0 (PFT30) is the LSB.
For the setting value, refer to **11.4.2 Input voltage and conversion results**.

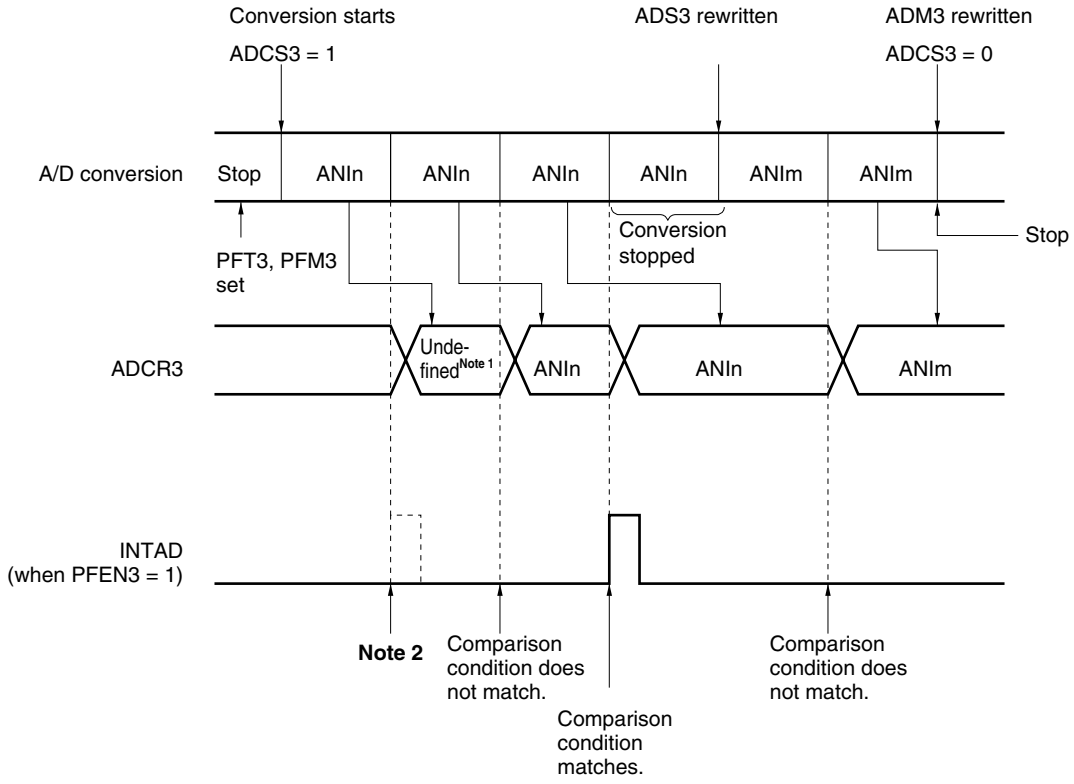
Cautions 1. In the power-fail comparison mode, the first result (A/D conversion result and interrupt request (INTAD)) of the A/D conversion (started by setting bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) to 1) is not correct.

2. When executing A/D conversion in the HALT mode using the power-fail HALT repeat mode, clear the interrupt request flag (ADIF) after the first conversion has been completed immediately after bit 7 (ADCS3) of ADM3 has been set to 1, and bit 5 (PFHRM3) of power-fail comparison mode register 3 (PFM3) has been set to 1, before executing the HALT instruction.

3. To set the power-fail comparison mode in the HALT mode, be sure to set bit 5 (PFHRM3) of PFM3 to 1 before executing the HALT instruction. Otherwise, comparison cannot be performed correctly because the conversion result in the HALT mode is not stored in A/D conversion result register 3 (ADCR3). If bit 5 (PFHRM3) of PFM3 is set in the normal operating mode (other than during HALT operation), the A/D conversion is not performed correctly. Therefore, be sure to clear this bit to 0 in the normal mode.

Figure 11-9. A/D Conversion Operation in Power-Fail Comparison Mode (1/3)

(1) In normal mode (other than HALT mode)



Notes 1. The conversion data is undefined immediately after bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) is set to 1 (to start conversion).

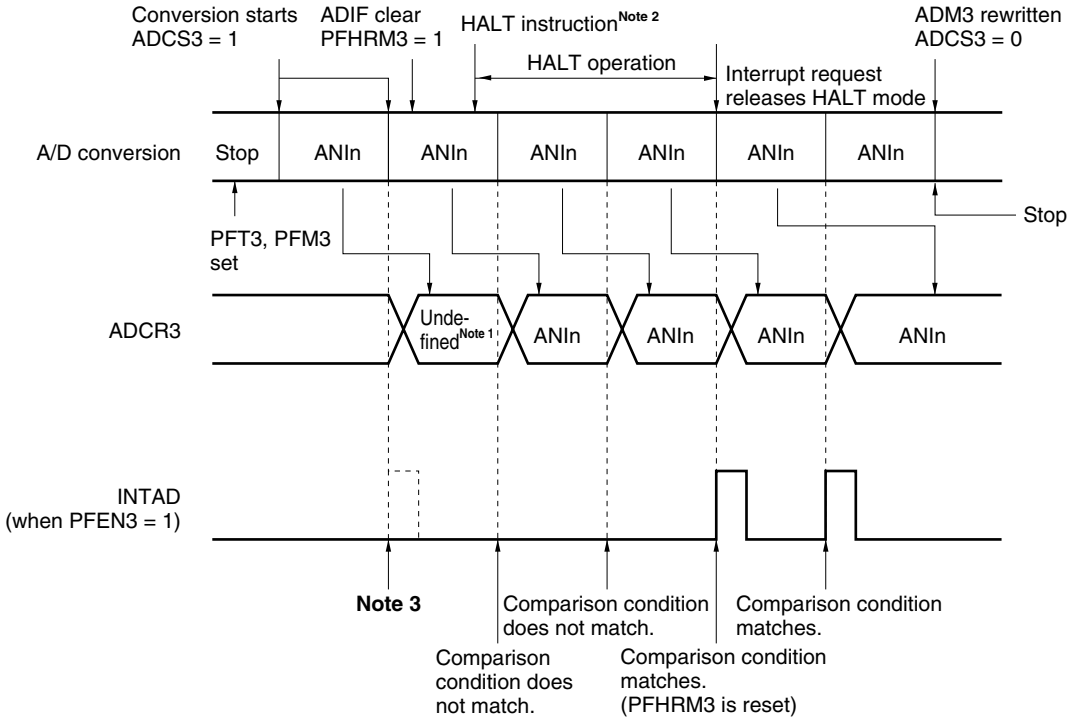
2. The first result of A/D conversion (A/D conversion result and interrupt request) is not correct. Do not use this result because there is a possibility that it will be determined that the comparison condition has matched even if it has not.

Caution Set power-fail comparison threshold value register 3 (PFT3) and power-fail comparison mode register 3 (PFM3) before starting conversion. Be sure to reset bit 5 (PFHRM3) of PFM3 to 0 (to disable HALT repeat mode setting).

Remark n = 0, 1, ... 7
m = 0, 1, ... 7

Figure 11-9. A/D Conversion Operation in Power-Fail Comparison Mode (2/3)

(2) In HALT repeat mode (when generation of interrupt (INTAD) is used to release HALT mode)



Notes 1. The conversion data is undefined immediately after bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) is set to 1 (to start conversion).

2. When executing A/D conversion in the HALT mode by using the power-fail comparison mode, clear the interrupt request flag (ADIF) after the first conversion has been completed immediately after bit 7 (ADCS3) of ADM3 has been set to 1, and bit 5 (PFHRM3) of power-fail comparison mode register 3 (PFM3) has been set to 1, before executing the HALT instruction.

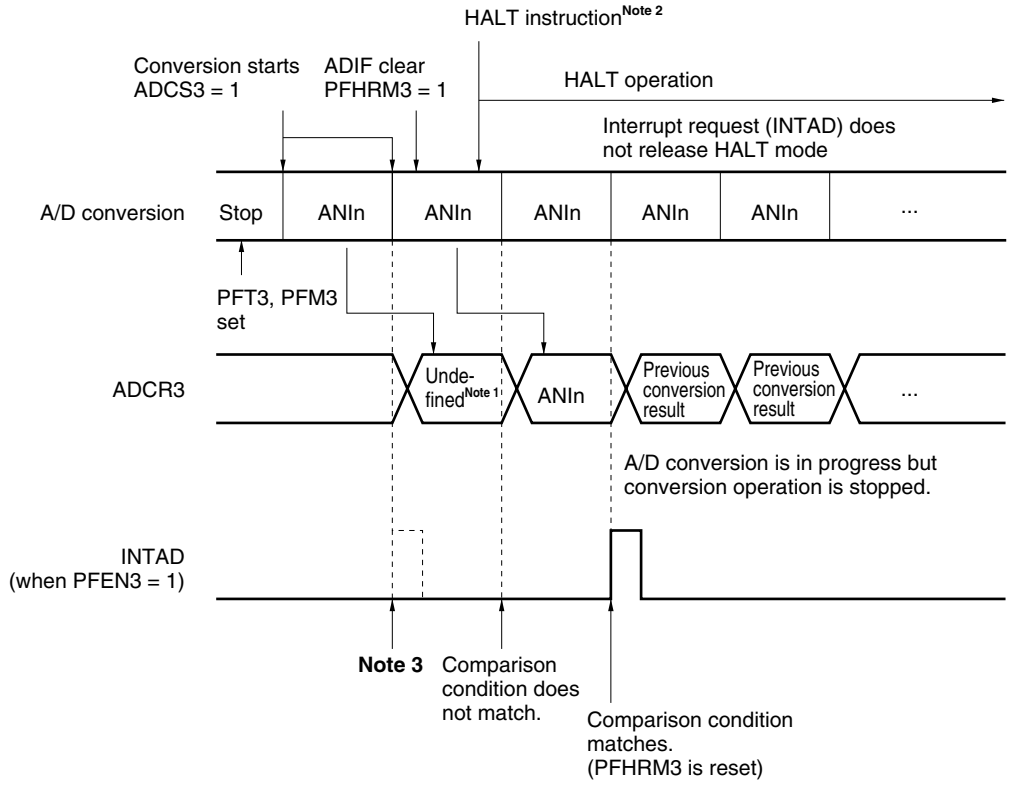
3. The first result of A/D conversion (A/D conversion result and interrupt request) is not correct. Do not use this result because there is a possibility that it will be determined that the comparison condition has matched even if it has not.

Caution Be sure to set bit 5 (PFHRM3) of PFM3 to 1 (to enable the HALT repeat mode setting).

Remark n = 0, 1, ... 7

Figure 11-9. A/D Conversion Operation in Power-Fail Comparison Mode (3/3)

(3) In HALT repeat mode (when generation of interrupt (INTAD) is not used to release HALT mode)



Notes 1. The conversion data is undefined immediately after bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) is set to 1 (to start conversion).

2. When executing A/D conversion in the HALT mode by using the power-fail HALT repeat mode, clear the interrupt request flag (ADIF) after the first conversion has been completed immediately after bit 7 (ADCS3) of ADM3 has been set to 1, and bit 5 (PFHRM3) of power-fail comparison mode register 3 (PFM3) has been set to 1, before executing the HALT instruction.

3. The first result of A/D conversion (A/D conversion result and interrupt request) is not correct. Do not use this result because there is a possibility that it will be determined that the comparison condition has matched even if it has not.

Caution Be sure to set bit 5 (PFHRM3) of PFM3 to 1 (to enable the HALT repeat mode setting).

Remark n = 0, 1, ... 7

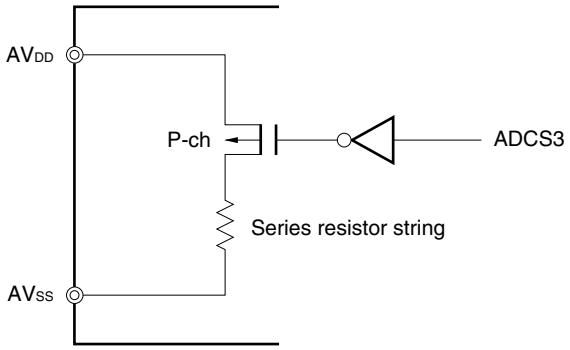
11.5 A/D Converter Cautions

(1) Current consumption in standby mode

The A/D converter is stopped in the standby mode. At this time, the current consumption can be reduced by resetting bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) to 0.

Figure 11-10 shows the circuit configuration of the series resistor string.

Figure 11-10. Circuit Configuration of Series Resistor String



(2) Input range of ANI0 to ANI7

The input voltages of ANI0 to ANI7 should be within the specification range. In particular, if a voltage of AV_{DD} or above or AV_{SS} or below is input (even if within the absolute maximum rating range), the conversion value for that channel will be undefined. The conversion values of the other channels may also be affected.

(3) Conflicting operations

<1> Conflict between writing A/D conversion result register 3 (ADCR3) on completion of conversion and reading ADCR3 by an instruction

Reading ADCR3 takes precedence. After ADCR3 has been read, a new conversion result is written to ADCR3.

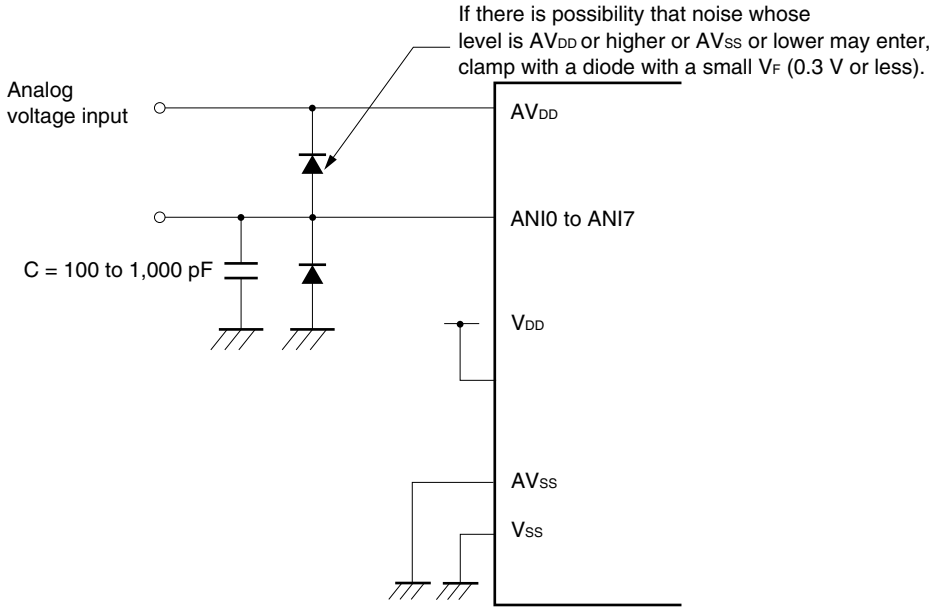
<2> Conflict between writing ADCR3 on completion of conversion and writing A/D converter mode register 3 (ADM3) or writing analog input channel specification register 3 (ADS3)

Writing ADM3 or ADS3 takes precedence. ADCR3 is not written. Nor is the A/D conversion end interrupt request signal (INTAD) generated.

★ **(4) Noise countermeasures**

In order to maintain 8-bit resolution, attention must be paid to noise on the AV_{DD} and ANI0 to ANI7 pins. Since the effect increases in proportion to the output impedance of the analog input source, it is recommended that a capacitor be connected externally as shown in Figure 11-11 in order to reduce noise.

Figure 11-11. Analog Input Pin Handling



(5) ANI0/P10 to ANI7/P17 pins

The analog input pins ANI0 to ANI7 also function as I/O port pins (port 1). When A/D conversion is performed with any of pins ANI0 to ANI7 selected, be sure not to execute an instruction that inputs data to port 1 while conversion is in progress, as this may reduce the conversion resolution.

Also, if digital pulses are applied to a pin adjacent to the pin in the process of A/D conversion, the expected A/D conversion value may not be obtainable due to coupling noise. Therefore, avoid applying pulses to pins adjacent to the pin undergoing A/D conversion.

★ **(6) Input impedance of ANI0 to ANI7 pins**

In this A/D converter, the internal sampling capacitor is charged and sampling is performed for approx. one tenth of the conversion time.

Since only the leakage current flows other than during sampling and the current for charging the capacitor also flows during sampling, the input impedance fluctuates and has no meaning.

To perform sufficient sampling, however, it is recommended to make the output impedance of the analog input source 10 kΩ or lower, or attach a capacitor of around 100 pF to the ANI0 to ANI7 pins (see Figure 11-11).

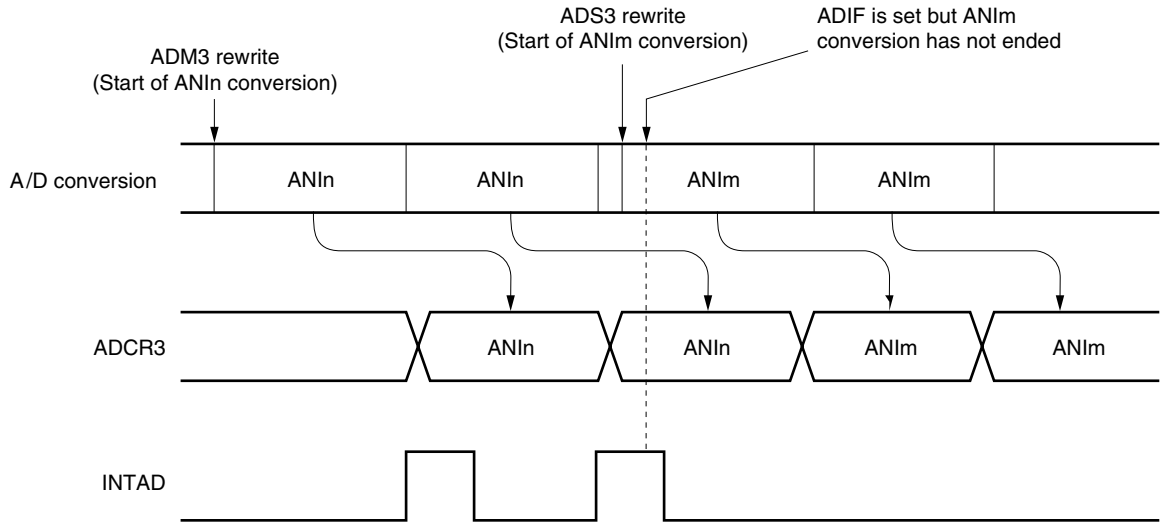
(7) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even if analog input channel specification register 3 (ADS3) is changed.

Caution is therefore required since, if the analog input pin is changed during A/D conversion, the A/D conversion result and conversion end interrupt request flag for the pre-change analog input may be set just before the ADS3 rewrite, and when ADIF is read immediately after the ADM rewrite, ADIF may be set despite the fact that the A/D conversion for the post-change analog input has not ended.

When A/D conversion is stopped and then resumed, clear the ADIF before it is resumed.

Figure 11-12. A/D Conversion End Interrupt Request Generation Timing



Remarks 1. n = 0, 1, ..., 7

2. m = 0, 1, ..., 7

(8) Conversion result immediately after starting A/D conversion

The first A/D conversion result value is undefined immediately after an A/D conversion operation has been started. Poll the A/D conversion end interrupt request (INTAD) and discard the first conversion result.

(9) Reading A/D conversion result register 3 (ADCR3)

If data is written to A/D converter mode register 3 (ADM3) and analog input channel specification register 3 (ADS3), the contents of ADCR3 are undefined. Read the conversion value before writing ADM3 and ADS3 after the conversion operation has been completed. Otherwise, the correct conversion result may not be read.

CHAPTER 12 OVERVIEW OF SERIAL INTERFACE

The serial interface differs between the μPD178096A and 178098A, and μPD178076, 178078, and 178F098. Table 12-1 shows the differences.

Table 12-1. Differences Between μPD178096A and 178098A, and μPD178076, 178078, and 178F098

Item	μPD178096A, 178098A	μPD178076, 178078, 178F098	Refer to:
SIO0	○	○	CHAPTER 13
SIO1	○	○	CHAPTER 14
SIO3	○	○	CHAPTER 15
UART0	—	○	CHAPTER 16

CHAPTER 13 SERIAL INTERFACE SIO0

13.1 Functions of Serial Interface SIO0

Serial interface SIO0 employs the following five modes.

- Operation stop mode
- 3-wire serial I/O mode
- SBI (serial bus interface) mode
- 2-wire serial I/O mode
- I^2C (Inter IC) bus mode**Note**

Note When using the I^2C bus mode (including when this mode is implemented by software without using the internal hardware), consult NEC Electronics when placing an order for a mask.

Caution Do not change the operation mode (3-wire serial I/O, SBI, 2-wire serial I/O, or I^2C bus) while the operation of serial interface SIO0 is enabled. To change the operation mode, stop the serial operation.

(1) Operation stop mode

This mode is used when serial transfer is not carried out to reduce power consumption.

(2) 3-wire serial I/O mode (MSB-/LSB-first selectable)

This mode is used for 8-bit data transfer using three lines, one each for the serial clock ($\overline{SCK0}$), serial output (SO0) and serial input (SI0). This mode enables simultaneous transmission/reception and therefore reduces the data transfer processing time.

The start bit of transferred 8-bit data is switchable between the MSB and LSB, so that devices can be connected regardless of their start bit recognition format.

This mode should be used when connecting with peripheral I/O devices or display controllers which incorporate a conventional clocked serial interface as is the case with the 75XL, 78K, and 17K Series.

(3) SBI (serial bus interface) mode (MSB-first)

This mode is used for 8-bit data transfer with two or more devices using two lines, one for the serial clock ($\overline{SCK0}$) and one for the serial data bus (SB0 or SB1).

In the SBI mode, transfer data is classified into “addresses”, “commands” and “data” for transmission/ reception, in conformance with serial bus format of NEC Electronics.

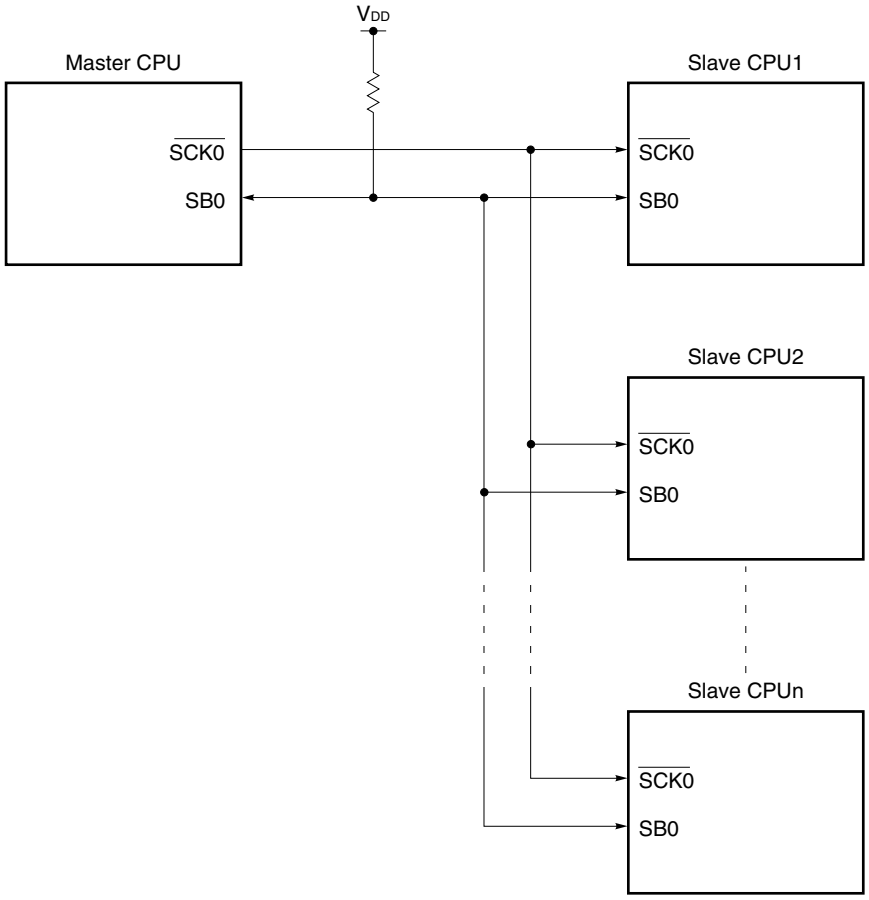
- Address: Data to select the target device for serial communication
- Command: Data to give instructions to the target device
- Data: Data actually transferred

In fact, transfer is started when the master device outputs an “address” to the serial bus to select one of the slave devices subject to communication. After that, “commands” and “data” are transmitted between the master device and slave device to implement serial communication. The receiver can automatically identify the received data as “addresses”, “commands”, or “data” by hardware.

This function enables the I/O ports to be used effectively and the application program serial interface control portions to be simplified.

In this mode, the wakeup function for handshake and the acknowledge and busy signal output function can also be used.

Figure 13-1. System Configuration Example of Serial Bus Interface (SBI)



(4) 2-wire serial I/O mode (MSB-first)

This mode is used for 8-bit data transfer using two lines, one for the serial clock ($\overline{SCK0}$) and one for the serial data bus (SB0 or SB1).

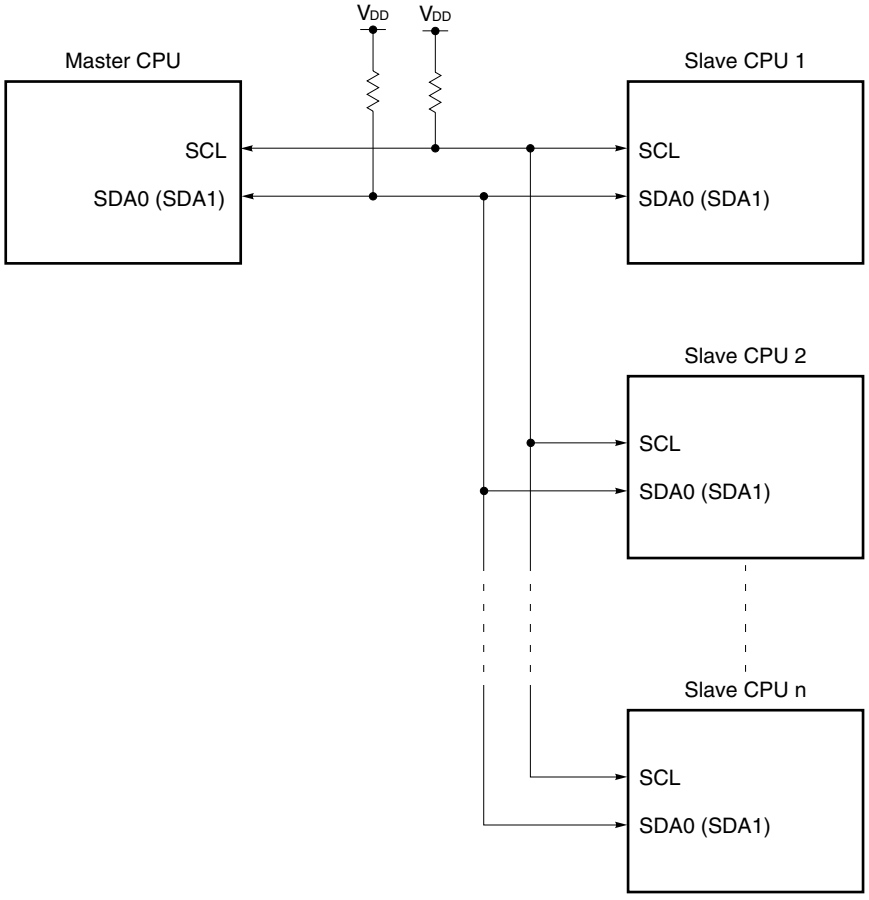
This mode enables handling of any of the possible data transfer formats by controlling the $\overline{SCK0}$ level and the SB0 or SB1 output level. Thus, the handshake line previously necessary for connection of two or more devices can be removed, resulting in an increased number of available I/O ports.

(5) I^2C bus mode (MSB-first)

This mode is used for 8-bit data transfer with two or more devices using two lines, one for the serial clock (SCL) and one for the serial data bus (SDA0 or SDA1).

This mode complies with the I^2C bus format. In this mode, the transmitter outputs three kinds of data onto the serial data bus: “start condition”, “data”, and “stop condition”. The receiver automatically distinguishes the received data as “start condition”, “data”, or “stop condition” by hardware.

Figure 13-2. Serial Bus Configuration Example Using I^2C Bus



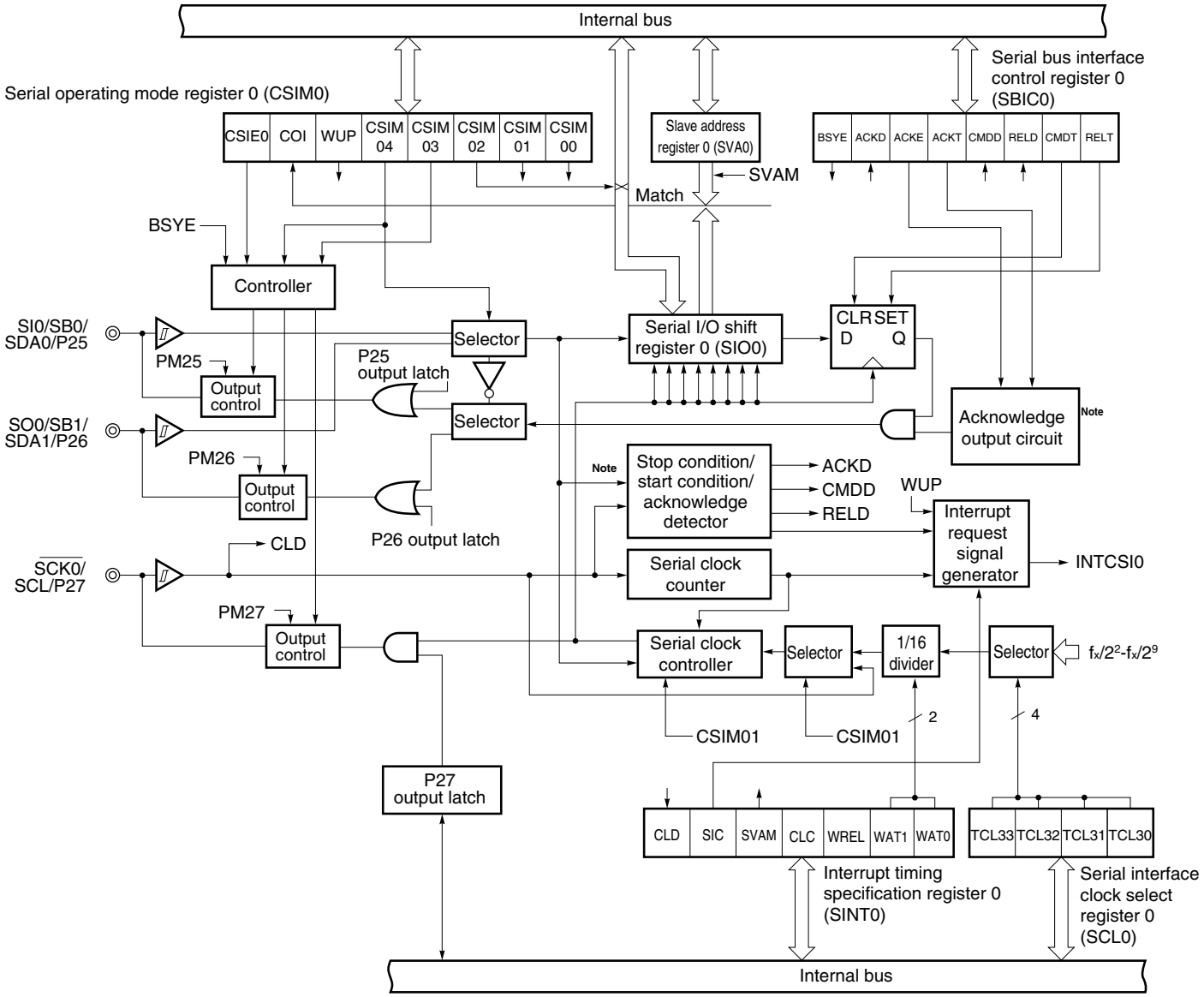
13.2 Configuration of Serial Interface SIO0

Serial interface SIO0 consists of the following hardware.

Table 13-1. Configuration of Serial Interface SIO0

Item	Configuration
Registers	Serial I/O shift register 0 (SIO0) Slave address register 0 (SVA0)
Control registers	Serial interface clock select register 0 (SCL0) Serial operating mode register 0 (CSIM0) Serial bus interface control register 0 (SBIC0) Interrupt timing specification register 0 (SINT0) Port mode register 2 (PM2) Port 2 (P2)

Figure 13-3. Block Diagram of Serial Interface SIO0



Note Example of operation in I^2C bus mode.

Remark The output control selects between CMOS output and N-ch open-drain output.

(1) Serial I/O shift register 0 (SIO0)

This is an 8-bit register used to carry out parallel/serial conversion and serial transmission/reception (shift operation) in synchronization with the serial clock.

SIO0 is set by an 8-bit memory manipulation instruction.

When bit 7 (CSIE0) of serial operating mode register 0 (CSIM0) is 1, writing data to SIO0 starts a serial operation.

In transmission mode, data written to SIO0 is output to the serial output (SO0) or serial data bus (SB0/SB1).

In reception mode, data is read from the serial input (SI0) or SB0/SB1 to SIO0.

Note that if the bus is driven in the SBI mode, 2-wire serial I/O mode, or I^2C bus mode, the bus pins must serve for both input and output. Thus, in the case of a device for reception, write FFH to SIO0 in advance (except when address reception is carried out by setting bit 5 (WUP) of CSIM0 to 1).

In the I^2C bus mode, set bit 7 (BSYE) of serial bus interface control register 0 (SBIC0) to 1.

In the SBI mode, the busy state can be cleared by writing data to SIO0. In this case, bit 7 (BSYE) of serial bus interface control register 0 (SBIC0) is not cleared to 0.

$\overline{\text{RESET}}$ input makes SIO0 undefined.

Caution In the I^2C bus mode, do not execute an instruction that writes to SIO0 while WUP (bit 5 of serial operating mode register 0 (CSIM0)) = 1. Data can be received when the wakeup function is being used (WUP = 1), even if such an instruction is not executed. For details of the wakeup function, refer to 13.4.5 (1) (c) Wakeup function.

(2) Slave address register 0 (SVA0)

This is an 8-bit register to set the slave address value for connection of a slave device to the serial bus.

SVA0 is set by an 8-bit memory manipulation instruction. It is not used in the 3-wire serial I/O mode.

The master device outputs a slave address for selection of a particular slave device to the connected slave devices. These two data (the slave address output from the master device and the SVA0 value) are compared by an address comparator. If they match, that slave device has been selected. In this case, bit 6 (COI) of serial operating mode register 0 (CSIM0) becomes 1.

The higher 7 bits of the slave address with the LSB masked can also be compared by setting the bit 4 (SVAM) of interrupt timing specification register 0 (SINT0).

If no match is detected in address reception, bit 2 (RELD) of serial bus interface control register 0 (SBIC0) is cleared to 0. By setting bit 5 (WUP) of CSIM0 to 1 in the SBI mode, the wakeup function can be used. In this case, an interrupt request signal (INTCSI0) is generated when the slave address output by the master matches the value of SVA0 (the interrupt request is also generated when a stop condition is detected). This interrupt request indicates that the master has requested communication.

Further, when SVA0 transmits data as a master or slave device in the SBI or 2-wire serial I/O mode, errors can be detected by using SVA0.

Reset input makes SVA0 undefined.

(3) SO0 latch

This latch holds the SI0/SB0/SDA0/P25 and SO0/SB1/SDA1/P26 pin levels. It can be directly controlled by software. In the SBI mode, this latch is set at the end of the 8th serial clock.

(4) Serial clock counter

This counter counts the serial clocks to be output and input during transmission/reception and to check whether 8-bit data has been transmitted/received.

(5) Serial clock controller

This circuit controls serial clock supply to serial I/O shift register 0 (SIO0). When the internal system clock is used, this circuit also controls clock output to the $\overline{\text{SCK0}}$/SCL/P27 pin.

(6) Interrupt request signal generator

This circuit controls interrupt request signal generation. It generates an interrupt request signal in the following cases.

- In the 3-wire serial I/O mode and 2-wire serial I/O mode
  This circuit generates an interrupt request signal every eight serial clocks.

- In the SBI mode
  When WUP**Note** is 0..... Generates an interrupt request signal every eight serial clocks.
  When WUP**Note** is 1..... Generates an interrupt request signal when the serial I/O shift register 0 (SIO0) value matches the slave address register 0 (SVA0) value after address reception.

  Note WUP is the wakeup function specification bit. It is bit 5 of serial operating mode register 0 (CSIM0).

- In the I^2C bus mode
  Generates an interrupt request as shown in Table 13-2.

(7) Output circuit and detector of control signals

These two circuits output and detect various control signals in the SBI mode.
They do not operate in the 3-wire serial I/O mode and 2-wire serial I/O mode.

- In the SBI mode
  Busy/acknowledge output circuit, bus release/command/acknowledge detector

- In the I^2C bus mode
  Acknowledge output circuit, stop condition/start condition/acknowledge detector

Table 13-2. Serial Interface SIO0 Interrupt Request Signal Generation

Serial Transfer mode	BSYE	WUP	WAT1	WAT0	ACKE	Description
I^2C bus mode (transmit)	0	0	1	0	0	An interrupt request signal is generated each time 8 serial clocks are counted (8-clock wait). Normally, during transmission, the settings WAT21, WAT0=1, 0, are not used. They are used only when wanting to coordinate the receive time and processing systematically using software. ACK information is generated by the receiving side, thus ACKE should be set to 0 (disabled).
			1	1	0	An interrupt request signal is generated each time 9 serial clocks are counted (9-clock wait). ACK information is generated by the receiving side, thus ACKE should be set to 0 (disabled).
	Other than above					Setting prohibited
I^2C bus mode (receive)	1	0	1	0	0	An interrupt request signal is generated each time 8 serial clocks are counted (8-clock wait). ACK information is output by manipulating ACKT by software after an interrupt is generated.
			1	1	0/1	An interrupt request signal is generated each time 9 serial clocks are counted (9-clock wait). To automatically generate ACK information, preset ACKE to 1 before transfer start. However, in the case of the master, set ACKE to 0 (disabled) before receiving the last data.
	1	1	1	1	1	After address is received, if the values of serial I/O shift register 0 (SI00) and slave address register 0 (SVA0) match, an interrupt request signal and a stop condition are generated. To automatically generate ACK information, preset ACKE to 1 (enable) before transfer start.
	Other than above					Setting prohibited

Remark BSYE: Bit 7 of the serial bus interface control register (SBIC)
ACKE: Bit 5 of the serial bus interface control register (SBIC)

13.3 Control Registers of Serial Interface SIO0

The following six registers are used to control serial interface SIO0.

- Serial interface clock select register 0 (SCL0)
- Serial operating mode register 0 (CSIM0)
- Serial bus interface control register 0 (SBIC0)
- Interrupt timing specification register 0 (SINT0)
- Port mode register 2 (PM2)
- Port 2

(1) Serial interface clock select register 0 (SCL0)

This register sets the serial clock of serial interface SIO0.

SCL0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SCL0 to 08H.

Figure 13-4. Format of Serial Interface Clock Select Register 0 (SCL0)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
SCL0	0	0	0	0	SCL03	SCL02	SCL01	SCL00	FF43H	08H	R/W

SCL03	SCL02	SCL01	SCL00	Selection of serial interface SIO0 serial clock
0	1	1	0	$f_X/2^2$ (Setting prohibited)
0	1	1	1	$f_X/2^3$ (788 kHz)
1	0	0	0	$f_X/2^4$ (394 kHz)
1	0	0	1	$f_X/2^5$ (197 kHz)
1	0	1	0	$f_X/2^6$ (98.4 kHz)
1	0	1	1	$f_X/2^7$ (49.2 kHz)
1	1	0	0	$f_X/2^8$ (24.6 kHz)
1	1	0	1	$f_X/2^9$ (12.3 kHz)
Other than above				Setting prohibited

Caution When rewriting SCL0 to other data, stop the serial transfer operation first.

Remark f_X: System clock oscillation frequency
( ): f_X = 6.3 MHz

(2) Serial operating mode register 0 (CSIM0)

This register sets the serial interface SIO0 serial clock, operating mode, operation enable/stop, and wakeup function, and displays the address comparator match signal.

CSIM0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM0 to 00H.

Caution Do not change the operating mode (3-wire serial I/O, 2-wire serial I/O, or SBI) while the operation of serial interface SIO0 is enabled. To change the operating mode, stop the serial operation.

Figure 13-5. Format of Serial Operating Mode Register 0 (CSIM0) (1/2)

Symbol	<7>	<6>	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM0	CSIE0	COI	WUP	CSIM04	CSIM03	CSIM02	CSIM01	0	FF60H	00H	R/W Note 1

R/W	CSIM01	Selection of serial interface SIO0 clock
	0	Input clock to $\overline{\text{SCK0}}$/SCL/P27 pin from off-chip
	1	Clock specified by bits 0 to 3 of serial interface clock select register 0 (SCL0)

R/W

CSIM 04	CSIM 03	CSIM 02	PM25	P25	PM26	P26	PM27	P27	Operating mode	Start bit	SIO/SB0/SDA0/P25 pin function	SO0/SB1/SDA1/P26 pin function	$\overline{\text{SCK0}}$/SCL/P27 pin function
0	×	0	Note 2 1	Note 2 ×	0	0	0	1	3-wire serial I/O mode	MSB	SIO Note 2 (Input)	SO0 (CMOS output)	$\overline{\text{SCK0}}$ (CMOS I/O)
		1								LSB			
1	0	0	Note 3 ×	Note 3 ×	0	0	0	1	SBI mode	MSB	P25 (CMOS I/O)	SB1 (N-ch open-drain I/O)	$\overline{\text{SCK0}}$ (CMOS I/O)
		1	0	0	Note 3 ×	Note 3 ×	0	1			SB0 (N-ch open-drain I/O)	P26 (CMOS I/O)	
1	1	0	Note 3 ×	Note 3 ×	0	0	0	1	2-wire serial I/O mode or I^2C bus mode	MSB	P25 (CMOS I/O)	SB1/SDA1 (N-ch open-drain I/O)	$\overline{\text{SCK0}}$/SCL (N-ch open-drain I/O)
		1	0	0	Note 3 ×	Note 3 ×	0	1			SB0/SDA0 (N-ch open-drain I/O)	P26 (CMOS I/O)	

(Continued)

Caution When using $\overline{\text{SCK0}}$ or SCL, set P27 to 1. If P27 is set to 0, it always outputs a low level.

Notes 1. Bit 6 (COI) is a read-only bit.

2. Can be used as P25 (CMOS input/output) when used only for transmission.

3. Can be used freely as a port function.

Remark ×: Don't care

PM××: Port mode register

P××: Output latch of port

Figure 13-5. Format of Serial Operating Mode Register 0 (CSIM0) (2/2)

R/W	WUP	Control of wake up function[Note 1]
	0	Interrupt request signal generated with each serial transfer in any mode
	1	• Interrupt request signal generated when the address received after bus release (when CMDD = RELD = 1) matches the slave address register 0 data in SBI mode • Interrupt request signal generated when the address received after start condition detected (CMDD = 1) matches the slave address register 0 data in I^2C bus mode

R	COI	Slave address comparison result flag[Note 2]
	0	Slave address register 0 and serial I/O shift register 0 data do not match
	1	Slave address register 0 and serial I/O shift register 0 data match

R/W	CSIE0	Control of serial interface SIO0 operation
	0	Operation stopped
	1	Operation enabled

Notes 1. When using the wakeup function (WUP = 1), set bit 5 (SIC) of interrupt timing specification register 0 (SINT0) as follows. Do not execute a write instruction to serial I/O shift register 0 (SIO0) while WUP = 1.

When the wakeup function is used in SBI mode: SIC = 0

When the wakeup function is used in I^2C bus mode: SIC = 1

2. When CSIE0 = 0, COI is 0.

(3) Serial bus interface control register 0 (SBIC0)

This register sets the serial bus interface SIO0 operation and display the status.

SBIC0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SBIC0 to 00H.

Figure 13-6. Format of Serial Bus Interface Control Register 0 (SBIC0) (1/3)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SBIC0	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	FF61H	00H	R/W[Note]

R/W		
R/W	RELT	Used for bus release signal output in SBI mode and stop condition signal output. When RELT = 1, SO latch is set to 1. After SO latch setting, automatically cleared to 0. Also cleared to 0 when CSIE0 = 0.

R/W		
R/W	CMDT	Used for command signal output in SBI mode and start condition signal output. When CMDT = 1, SO latch is cleared to (0). After SO latch clearance, automatically cleared to 0. Also cleared to 0 when CSIE0 = 0.

R	RELD	Detection of bus release (SBI mode)/stop condition (I^2C bus mode)
Clear conditions (RELD = 0)		Set conditions (RELD =1)
• When transfer start instruction is executed • If SIO0 and SVA0 values do not match in address reception • When CSIE0 = 0 • When reset input is applied		• When bus release signal (REL) or stop condition is detected

R	CMDD	Detection of command (SBI mode)/start condition (I^2C bus mode)
Clear conditions (CMDD = 0)		Set conditions (CMDD = 1)
• When transfer start instruction is executed • When bus release signal (REL) or stop condition is detected • When CSIE0 = 0 • When reset input is applied		• When command signal (CMD) or stop condition is detected

Note Bits 2, 3, and 6 (RELD, CMDD and ACKD) are read-only bits.

Remarks 1. Bits 0, 1, and 4 (RELD, CMDT, and ACKT) are 0 when read after data setting.

2. CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

Figure 13-6. Format of Serial Bus Interface Control Register 0 (SBIC0) (2/3)

(a) SBI mode

R/W

ACKT	The acknowledge signal is output in synchronization with the falling edge of $\overline{SCK0}$ just after execution of the instruction to be set to 1, and after acknowledge signal output, automatically cleared to 0. Used as ACKE = 0. Also cleared to 0 upon start of serial interface transfer or when CSIE0 = 0.

R/W

ACKE	Control of acknowledge signal output	
0	Acknowledge signal automatic output disabled (output with ACKT enabled)	
1	Before completion of transfer	The acknowledge signal is output in synchronization with the 9th falling edge of $\overline{SCK0}$ (automatically output when ACKE = 1).
	After completion of transfer	The acknowledge signal is output in synchronization with the falling edge of $\overline{SCK0}$ just after execution of the instruction to be set to 1 (automatically output when ACKE = 1). However, not automatically cleared to 0 after acknowledge signal output.

R

ACKD	Detection of acknowledge	
Clear conditions (ACKD = 0)		Set conditions (ACKD = 1)
• Falling edge of $\overline{SCK0}$ immediately after the busy mode is released after executing the transfer start instruction • When CSIE0 = 0 • When reset input is applied		• When acknowledge signal ($\overline{ACK}$) is detected at the rising edge of $\overline{SCK0}$ clock after completion of transfer

R/W

BSYE Note	Synchronization of busy signal output control
0	Busy signal output in synchronization with the falling edge of $\overline{SCK0}$ clock just after execution of the instruction to be cleared to 0 disabled.
1	Busy signal output at the falling edge of $\overline{SCK0}$ clock following the acknowledge signal.

Note The busy mode can be canceled by the start of serial interface transfer or address signal reception. However, the BSYE flag is not cleared to 0.

Remark CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

Figure 13-6. Format of Serial Bus Interface Control Register 0 (SBIC0) (3/3)

(b) I^2C bus mode

R/W

ACKT	SDA0 (SDA1) is set to low just after execution of the instruction to be set to 1 before the next SCL falling edge. Used for generating an $\overline{ACK}$ signal by software if the 8-clock wait mode is selected. Cleared to 0 if CSIE = 0 when a transfer by the serial interface is started.

R/W

ACKE	Control of acknowledge signal output[Note 1]
0	Acknowledge signal automatic output disabled. (However, output with ACKT is enabled) Used for reception when 8-clock wait mode is selected or for transmission. [Note 2]
1	Acknowledge signal automatic output enabled. Acknowledge signal output in synchronization with the falling edge of the 9th SCL clock cycle (automatically output when ACKE = 1). However, not automatically cleared to 0 after acknowledge signal output. Used in reception with 9-clock wait mode selected.

R

ACKD	Detection of acknowledge	
Clear conditions (ACKD = 0)		Set conditions (ACKD = 1)
• When transfer start instruction is executed • When CSIE0 = 0 • When reset input is applied		• When acknowledge signal ($\overline{ACK}$) is detected at the rising edge of SCL clock after completion of transfer

R/W

[Note 3] BSYE	Control of N-ch open-drain output for transmission in I^2C bus mode[Note 4]
0	Output enabled (transmission)
1	Output disabled (reception)

Notes 1. This setting should be performed before transfer.

2. If 8-clock wait mode is selected, the acknowledge signal at reception must be output using ACKT.

3. The busy mode can be canceled by the start of serial interface transfer or the reception of an address signal. However, the BSYE flag is not cleared to 0.

4. When using the wakeup function, be sure to set BSYE to 1.

Remark CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

(4) Interrupt timing specification register 0 (SINT0)

This register sets the bus release interrupt and address mask functions and displays the P27/$\overline{SCK0}$/SCL pin level status.

SINT0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SINT0 to 00H.

Figure 13-7. Format of Interrupt Timing Specification Register 0 (SINT0) (1/2)

Symbol	7	<6>	<5>	<4>	<3>	<2>	1	0	Address	After reset	R/W
SINT0	0	CLD	SIC	SVAM	CLC	WREL	WAT1	WAT0	FF63H	00H	R/W[Note 1]

R/W

WAT1	WAT0	Control of wait and interrupt
0	0	Interrupt servicing request generated at rising edge of 8th $\overline{SCK0}$ clock (keeping clock output in high impedance state).
0	1	Setting prohibited
1	0	Used in I^2C bus mode (8-clock wait). Interrupt servicing request generated at rising edge of 8th SCL clock. (In the case of a master device, SCL output is made low to enter a wait state after 8 clock pulses are output. In the case of a slave device, SCL output is made low to request a wait state after 8 clock pulses are input.)
1	1	Used in I^2C bus mode (9-clock wait). Interrupt servicing request generated at rising edge of 9th SCL clock . (In the case of a master device, SCL output is made low to enter a wait state after 9 clock pulses are output. In the case of a slave device, SCL output is made low to request a wait state after 9 clock pulses are input.)

R/W

WREL	Control of wait sate cancellation
0	Wait state has already been cancelled.
1	Cancel wait state. Automatically cleared to 0 when the state is cancelled. (Used to cancel wait state by means of WAT0 and WAT1.)

R/W

CLC	Control of clock level[Note 2]
0	Used in I^2C bus mode. Output level of SCL pin made low unless serial transfer is being performed.
1	Used in I^2C bus mode. SCL pin enters high-impedance state unless serial transfer is being performed (except for clock line, which is kept high). Used to enable master device to generate start condition and stop condition signals.

Notes 1. Bit 6 (CLD) is a read-only bit.

2. When not using the I^2C mode, set CLC to 0.

Figure 13-7. Format of Interrupt Timing Specification Register 0 (SINT0) (2/2)

R/W

SVAM	SVA0 bits to be used as slave address
0	Bits 0 to 7
1	Bits 7 to 0

R/W

SIC	Selection of INTCSI0 interrupt source[Note 1]
0	CSIIF0 is set to 1 upon termination of serial interface SIO0 transfer
1	CSIIF0 is set to 1 upon bus release detection (SBI mode), stop condition detection (I^2C bus mode), or termination of serial interface SIO0 transfer

R

CLD	P27/$\overline{SCK0}$/SCL pin level[Note 2]
0	Low level
1	High level

Notes 1. When using wakeup function in the I^2C mode, set SIC to 1.
When using wakeup function in the SBI mode, set SIC to 0.

2. When CSIE0 = 0, CLD is 0.

Remark SVA0: Slave address register 0
CSIIF: Interrupt request flag corresponding to INTCSI0
CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

13.4 Operations of Serial Interface SIO0

The following five operating modes are available for serial interface SIO0.

- Operation stop mode
- 3-wire serial I/O mode
- SBI mode
- 2-wire serial I/O mode
- I^2C (Inter IC) bus mode

13.4.1 Operation stop mode

Serial transfer is not carried out in the operation stop mode.

Serial I/O shift register 0 (SIO0) does not carry out shift operation either and thus it can be used as an ordinary 8-bit register.

In the operation stop mode, the P25/SI0/SB0/SDA0, P26/SO0/SB1/SDA1 and P27/$\overline{SCK0}$/SCL pins can be used as ordinary I/O ports.

(1) Register setting

The operation stop mode is set using serial operating mode register 0 (CSIM0).

CSIM0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM0 to 00H.

Symbol	<7>	<6>	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM0	CSIE0	COI	WUP	CSIM04	CSIM03	CSIM02	CSIM01	0	FF60H	00H	R/W

R/W

CSIE0	Control of serial interface SIO0 operation
0	Operation stopped
1	Operation enabled

13.4.2 3-wire serial I/O mode operation

The 3-wire serial I/O mode is effective for connection of peripheral I/O units and display controllers which incorporate a conventional clocked serial interface as is the case with the 75XL, 78K, and 17K Series.

Communication is carried out with three lines, one each for the serial clock ($\overline{SCK0}$), serial output (SO0), and serial input (SI0).

(1) Register setting

The 3-wire serial I/O mode is set using serial operating mode register 0 (CSIM0), serial bus interface control register 0 (SBIC0), port mode register 2 (PM2), and port 2 (P2).

(a) Serial operating mode register 0 (CSIM0)

CSIM0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM0 to 00H.

Symbol	<7>	<6>	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM0	CSIE0	COI	WUP	CSIM04	CSIM03	CSIM02	CSIM01	0	FF60H	00H	R/W Note 1

R/W

CSIM01	Selection of serial interface SIO0 clock
0	Clock Input to $\overline{SCK0}$/SCL/P27 pin from off-chip
1	Clock specified by bits 0 to 3 of serial interface clock select register 0 (SCL0)

R/W

CSIM 04	CSIM 03	CSIM 02	PM25	P25	PM26	P26	PM27	P27	Operating mode	Start bit	SIO/SB0/SDA0/P25 pin function	SO0/SB1/SDA1/P26 pin function	$\overline{SCK0}$/SCL/P27 pin function
0	×	0	1 Note 2	× Note 2	0	0	0	1	3-wire serial I/O mode	MSB	SI0 Note 2 (Input)	SO0 (CMOS output)	$\overline{SCK0}$ (CMOS I/O)
		1								LSB			
1	0	SBI mode (refer to **13.4.3 SBI mode operation**).											
1	1	2-wire serial I/O mode (refer to **13.4.4 2-wire serial I/O mode operation**) or I^2C bus mode (refer to **13.4.5 I^2C bus mode operation**).											

R/W

WUP	Control of wakeup function
0	Interrupt request signal generated with each serial transfer in any mode
1	Setting prohibited Note 3

R/W

CSIE0	Control of serial interface SIO0 operation
0	Operation stopped
1	Operation enabled

Caution When using $\overline{SCK0}$ or SCL, set P27 to 1. If P27 is set to 0, it always outputs a low level.

Notes 1. Bit 6 (COI) is a read-only bit.

2. Can be used as P25 (CMOS I/O) when used only for transmission.

3. Be sure to set WUP to 0 when the 3-wire serial I/O mode is selected.

Remark ×: Don't care

PM××: Port mode register

P××: Output latch of port

(b) Serial bus interface control register 0 (SBIC0)

SBIC0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SBIC0 to 00H.

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SBIC0	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	FF61H	00H	R/W

R/W		
	RELT	When RELT = 1, SO latch is set to 1. After SO latch setting, automatically cleared to 0. Also cleared to 0 when CSIE0 = 0.

R/W		
	CMDT	When CMDT = 1, SO latch is cleared to 0. After SO latch clearance, automatically cleared to 0. Also cleared to 0 when CSIE0 = 0.

Remark CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

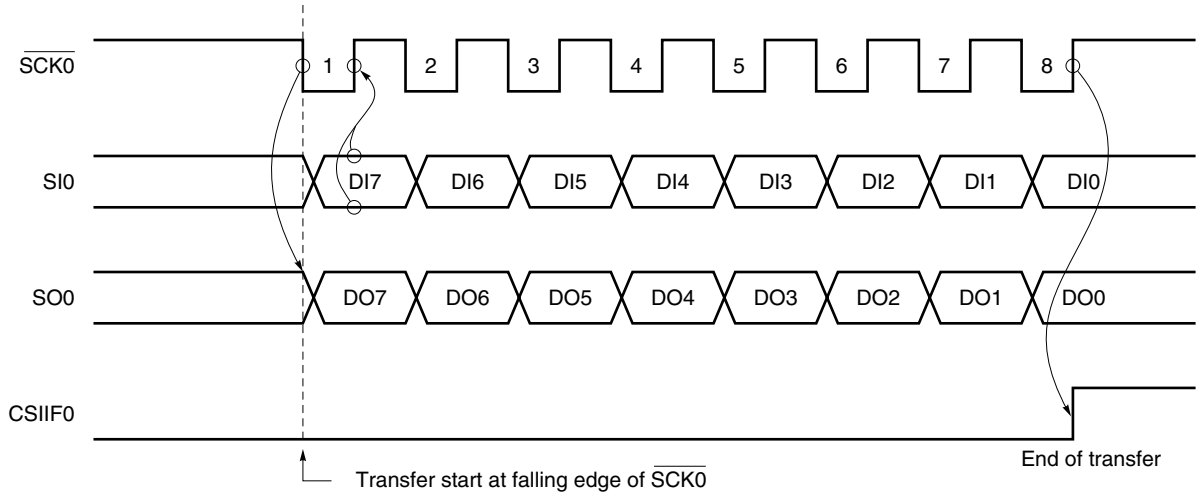
(2) Communication operation

The 3-wire serial I/O mode is used for data transmission/reception in 8-bit units. Bit-wise data transmission/reception is carried out in synchronization with the serial clock.

The shift operation of serial I/O shift register 0 (SIO0) is carried out at the falling edge of the serial clock ($\overline{SCK0}$). The transmitted data is held in the SO0 latch and is output from the SO0 pin. The received data input to the SI0 pin is latched in SIO0 at the rising edge of $\overline{SCK0}$.

Upon termination of 8-bit transfer, the SIO0 operation stops automatically and the interrupt request flag (CSIIF0) is set.

Figure 13-8. 3-Wire Serial I/O Mode Timing



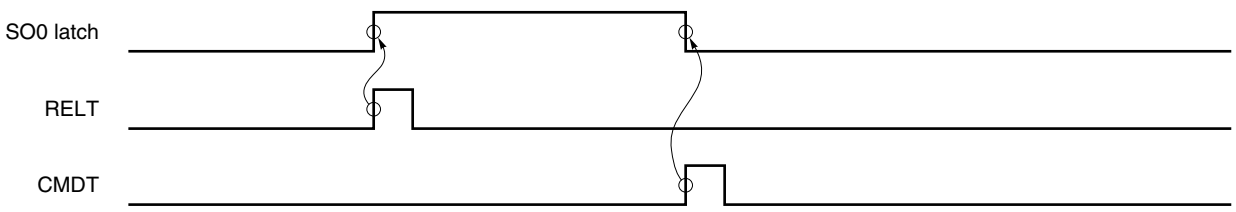
The SO0 pin is a CMOS output pin and outputs the current SO0 latch status. Thus, the SO0 pin output status can be manipulated by setting bit 0 (RELT) and bit 1 (CMDT) of serial bus interface control register 0 (SBIC0). However, do not carry out this manipulation during serial transfer.

Control the $\overline{SCK0}$ pin output level in the output mode (internal system clock mode) by manipulating the P27 output latch (refer to **13.4.8 $\overline{SCK0}$/SCL/P27 pin output manipulation**).

(3) Other signals

Figure 13-9 shows the RELT and CMDT operations.

Figure 13-9. RELT and CMDT Operations



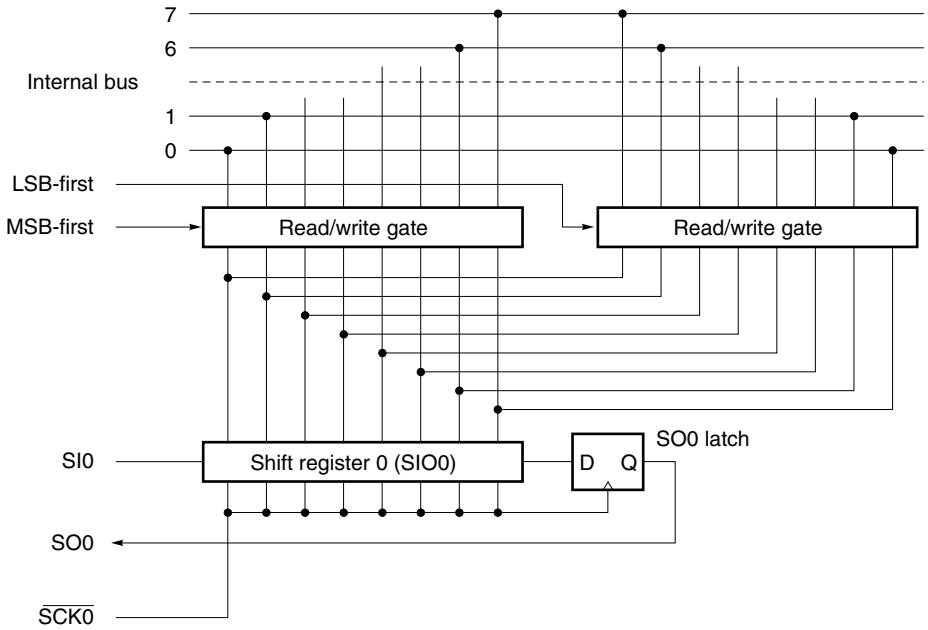
(4) MSB/LSB switching as the start bit

In the 3-wire serial I/O mode, transfer can be selected to start from the MSB or LSB.

Figure 13-10 shows the configuration of serial I/O shift register 0 (SIO0) and the internal bus. As shown in the figure, MSB/LSB can be read/written in reverse form.

MSB/LSB switching as the start bit can be specified by bit 2 (CSIM02) of serial operating mode register 0 (CSIM0).

Figure 13-10. Circuit for Switching Transfer Bit Order



Start bit switching is realized by switching the bit order for data written to SIO0. The SIO0 shift order remains unchanged.

Thus, switching between MSB-first and LSB-first must be performed before writing data to the shift register.

(5) Transfer start

Serial transfer is started by setting transfer data to serial I/O shift register 0 (SIO0) when the following two conditions are satisfied.

- Serial interface SIO0 operation control bit (CSIE0) = 1.
- Internal serial clock is stopped or $\overline{SCK0}$ is high level after 8-bit serial transfer.

Caution If CSIE0 is set to "1" after writing data to SIO0, transfer does not start.

Remark CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

Upon termination of 8-bit transfer, serial transfer automatically stops and the interrupt request flag (CSIIF0) is set.

13.4.3 SBI mode operation

SBI (Serial Bus Interface) is a high-speed serial interface that complies with the NEC Electronics serial bus format.

SBI uses a single master device and employs the clocked serial I/O format with the addition of a bus configuration function. This function enables devices to communicate using only two lines. Thus, when configuring a serial bus with two or more microcontrollers and peripheral ICs, the number of ports to be used and the number of wires on the board can be decreased.

The master device outputs three kinds of data to slave devices on the serial data bus: “addresses” to select the device to communicate with, “commands” to instruct the selected device, and “data”, which is the data actually sent.

The slave device can identify the received data as “address”, “command”, or “data” by hardware. This function enables the application program that controls the serial interface SIO0 to be simplified.

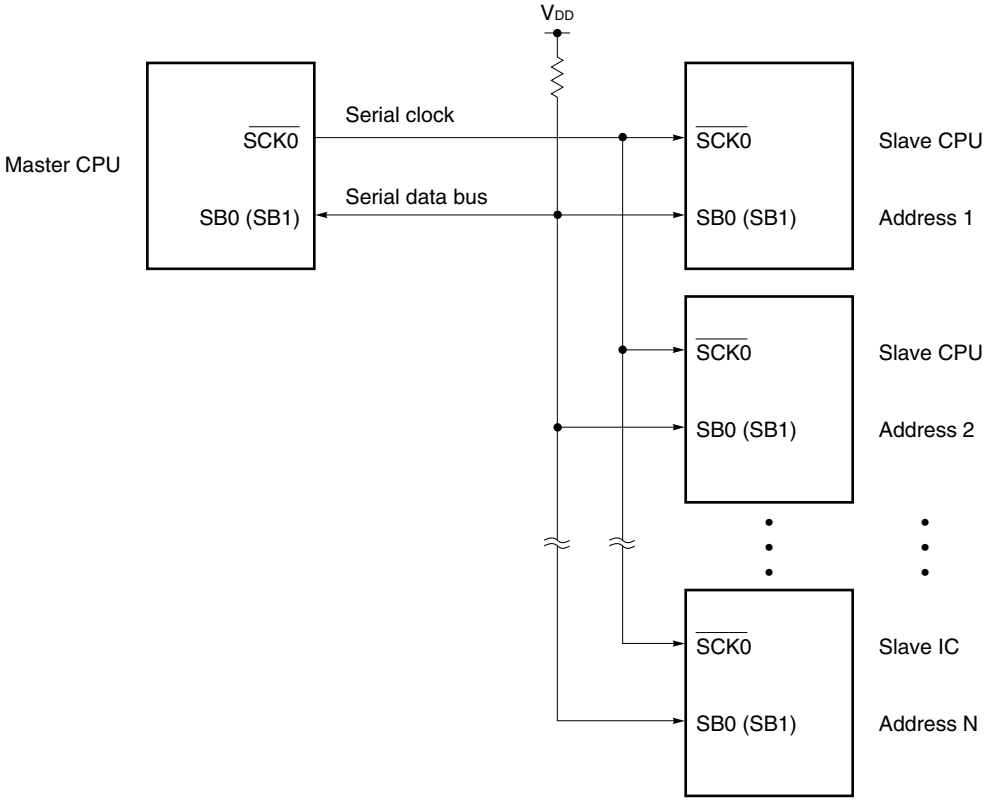
The SBI function is incorporated into various devices including 75XL Series and 78K Series devices.

Figure 13-11 shows a serial bus configuration example when a CPU having a serial interface compliant with SBI and peripheral ICs are used.

In SBI, the SB0 (SB1) serial data bus pin is an open-drain output pin and therefore the serial data bus line behaves in the same way as a wired-AND configuration. In addition, a pull-up resistor must be connected to the serial data bus line.

When using the SBI mode, refer to **(d)** in **(11) SBI mode cautions** described later.

Figure 13-11. Example of Serial Bus Configuration with SBI



Caution When exchanging the master CPU/slave CPU, a pull-up resistor is necessary for the serial clock line ($\overline{\text{SCK0}}$) as well because serial clock line ($\overline{\text{SCK0}}$) I/O switching is carried out asynchronously between the master and slave CPUs.

(1) SBI functions

In the conventional serial I/O format, when a serial bus is configured by connecting two or more devices, many ports and wiring are necessary to provide chip select signals to identify commands and data, and to judge the busy state, because only the data transfer function is available. If these operations are to be controlled by software, the software will become heavily loaded.

In SBI, a serial bus can be configured with two signal lines, one for the serial clock $\overline{\text{SCK0}}$ and one for the serial data bus SB0 (SB1). Thus, use of SBI leads to a reduction in the number of microcontroller ports and the amount of wiring and routing on the board.

The SBI functions are described below.

(a) Address/command/data identification function

Serial data is distinguished as addresses, commands, and data.

(b) Chip select function by address transmission

The master executes slave chip selection by address transmission.

(c) Wakeup function

The slave can easily judge address reception (chip select judgment) with the wakeup function (which can be set/reset by software).

When the wakeup function is set, the interrupt request signal (INTCSI0) is generated upon reception of a match address.

Thus, when communication is executed with two or more devices, the CPU of devices other than the selected slave device can operate independent of serial communications.

(d) Acknowledge signal ($\overline{\text{ACK}}$) control function

The acknowledge signal is controlled to check serial data reception.

(e) Busy signal ($\overline{\text{BUSY}}$) control function

The busy signal is controlled to report the slave busy state.

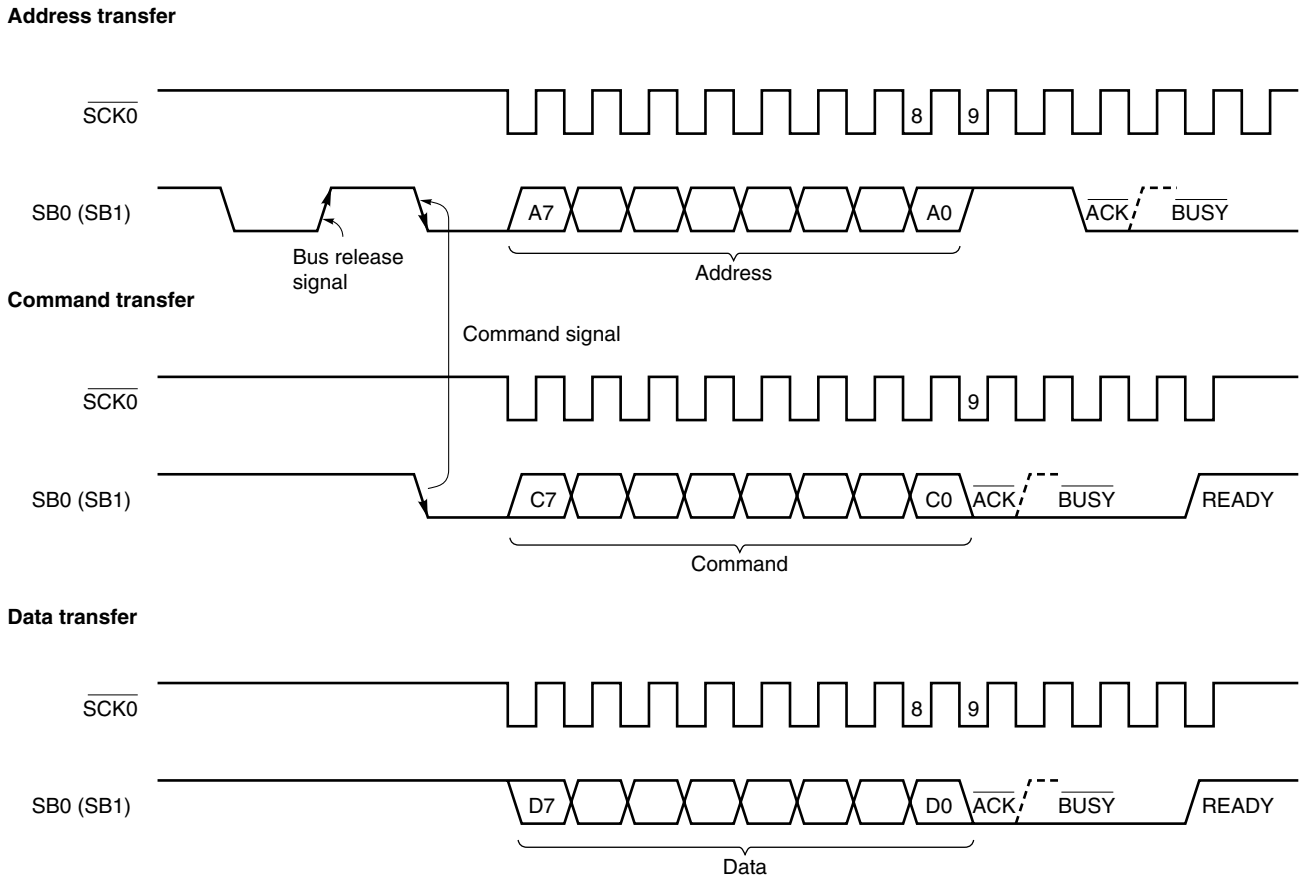
(2) SBI definition

The SBI serial data format and the signals to be used are defined as follows.

Serial data to be transferred with SBI consists of three kinds of data: “address”, “command”, and “data”.

Figure 13-12 shows the address, command, and data transfer timing.

Figure 13-12. SBI Transfer Timing



Remark The dotted line indicates the READY status.

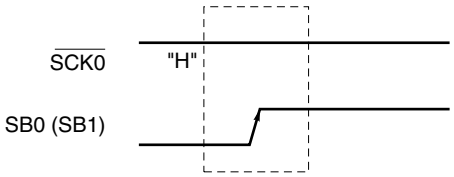
The bus release signal and the command signal are output by the master device. $\overline{\text{BUSY}}$ is output by the slave device. $\overline{\text{ACK}}$ can be output by either the master or slave device (normally, the 8-bit data receiver outputs $\overline{\text{ACK}}$). Serial clocks continue to be output by the master device from 8-bit data transfer start to $\overline{\text{BUSY}}$ release.

(a) Bus release signal (REL)

The bus release signal occurs when the SB0 (SB1) line changes from low level to high level when the $\overline{SCK0}$ line is high level (without serial clock output).

This signal is output by the master device.

Figure 13-13. Bus Release Signal



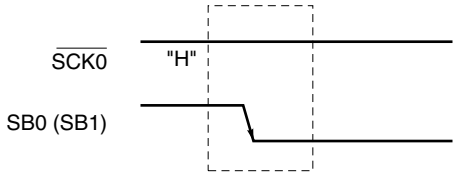
The bus release signal indicates that the master device is going to transmit an address to the slave device. The slave device incorporates hardware to detect the bus release signal.

Caution A transition of the SB0 (SB1) pin from low to high is recognized as a bus release signal when the $\overline{SCK0}$ line is high. If the change timing of the bus is shifted due to the influence of board capacitance, data that is transmitted may be identified as a bus release signal by mistake. Exercise care when wiring.

(b) Command signal (CMD)

The command signal occurs when the SB0 (SB1) line changes from high level to low level when the $\overline{SCK0}$ line is high level (without serial clock output). This signal is output by the master device.

Figure 13-14. Command Signal



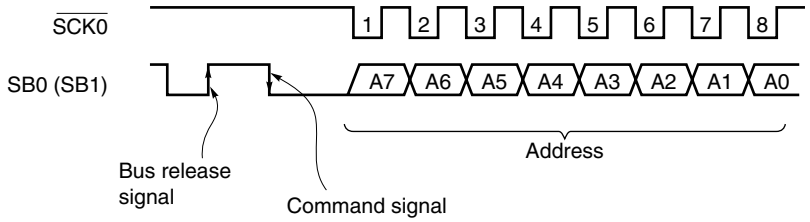
The command signal indicates that the master is going to transmit a command to the slave (however, the command signal following the bus release signal indicates that an address is to be transmitted). The slave device incorporates hardware to detect the command signal.

Caution A transition of the SB0 (SB1) pin from high to low is recognized as a command signal when the $\overline{SCK0}$ line is high. If the change timing of the bus is shifted due to the influence of board capacitance, data that is transmitted may be identified as a command signal by mistake. Exercise care when wiring.

(c) Address

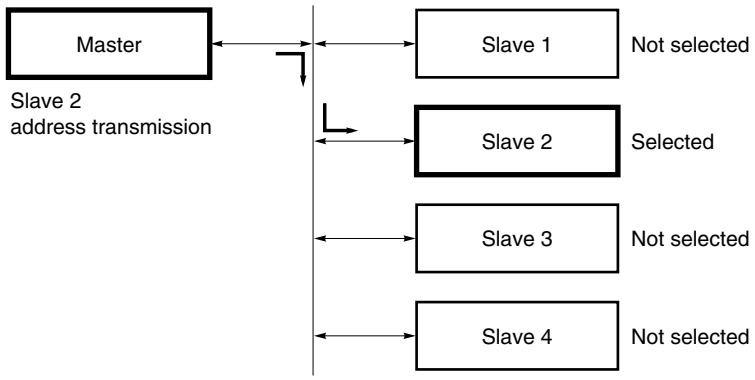
An address is 8-bit data that the master device outputs to the slave devices connected to the bus line in order to select a particular slave device.

Figure 13-15. Addresses



Eight-bit data following the bus release and command signals is defined as an “address”. In the slave device, this condition is detected by hardware and whether or not the 8-bit data matches that slave’s specification number (slave address) is checked by hardware. If the 8-bit data matches the slave address, that slave device has been selected. After that, communication with the master device continues until a release instruction is received from the master device.

Figure 13-16. Slave Selection by Address



(d) Command and data

The master device transmits commands to and transmits/receives data to/from the slave device selected by address transmission.

Figure 13-17. Commands

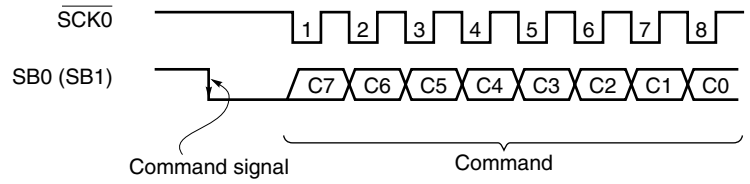
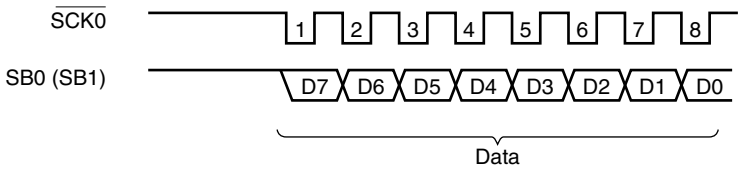


Figure 13-18. Data



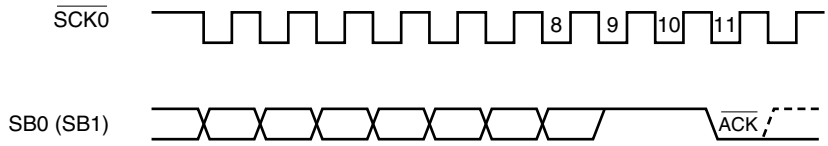
Eight-bit data following a command signal is defined as “command” data. Eight-bit data without a command signal is defined as “data”. Command and data operation procedures can be determined by the user according to their communication specifications.

(e) Acknowledge signal ($\overline{ACK}$)

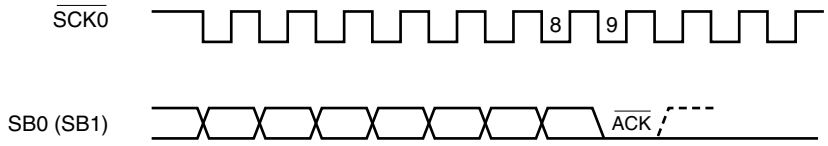
The acknowledge signal is used to check serial data reception between the transmitter and receiver.

Figure 13-19. Acknowledge Signal

[When output in synchronization with 11th $\overline{SCK0}$ clock]



[When output in synchronization with 9th $\overline{SCK0}$ clock]



The acknowledge signal is one-shot pulse generated at the falling edge of $\overline{SCK0}$ after 8-bit data transfer. It can be positioned anywhere and can be synchronized with any $\overline{SCK0}$ clock.

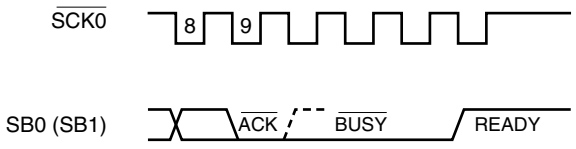
After 8-bit data transmission, the transmitter checks whether the receiver has returned the acknowledge signal. If the acknowledge signal is not returned for the preset period of time after data transmission, it can be judged that data reception has not been carried out correctly.

(f) Busy signal ($\overline{\text{BUSY}}$) and ready signal (READY)

The $\overline{\text{BUSY}}$ signal is used to report to the master device that the slave device is not ready for data transmission/reception.

The READY signal is intended to report to the master device that the slave device is ready for data transmission/reception.

Figure 13-20. $\overline{\text{BUSY}}$ and READY Signals



In SBI, the slave device notifies the master device of the busy state by setting the SB0 (SB1) line to the low level.

$\overline{\text{BUSY}}$ signal output follows acknowledge signal output from the master or slave device. It is set/reset at the falling edge of $\overline{\text{SCK0}}$. When the $\overline{\text{BUSY}}$ signal is reset, the master device automatically terminates the output of the $\overline{\text{SCK0}}$ serial clock.

When the $\overline{\text{BUSY}}$ signal is reset and the READY signal is set, the master device can start the next transfer.

Caution In the SBI mode, SBI outputs the $\overline{\text{BUSY}}$ signal after the $\overline{\text{BUSY}}$ clear instruction has been executed until the next serial clock falls. If WUP is set to 1 by mistake during this period, $\overline{\text{BUSY}}$ will not be cleared. Before setting WUP to 1, therefore, clear $\overline{\text{BUSY}}$, and make sure that the SB0 (SB1) pin has gone high.

(3) Register setting

The SBI mode is set using serial operating mode register 0 (CSIM0), serial bus interface control register 0 (SBIC0), interrupt timing specification register 0 (SINT0), port mode register 2 (PM2), and port 2 (P2).

(a) Serial operating mode register 0 (CSIM0)

CSIM0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM0 to 00H.

Symbol	<7>	<6>	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM0	CSIE0	COI	WUP	CSIM04	CSIM03	CSIM02	CSIM01	0	FF60H	00H	R/W[Note 1]

R/W

CSIM01	Selection of serial interface SIO0 clock
0	Clock input to $\overline{SCK0}$/SCL/P27 pin from off-chip
1	Clock specified by bits 0 to 3 of serial interface clock select register 0 (SCL0)

R/W

CSIM 04	CSIM 03	CSIM 02	PM25	P25	PM26	P26	PM27	P27	Operating mode	Start bit	SI0/SB0/SDA0/P25 pin function	SO0/SB1/SDA1/P26 pin function	$\overline{SCK0}$/SCL/P27 pin function
0	×	3-wire serial I/O mode (refer to **13.4.2 3-wire serial I/O mode operation**).											
1	0	0	Note 2 ×	Note 2 ×	0	0	0	1	SBI mode	MSB	P25 (CMOS I/O)	SB1 (N-ch open-drain I/O)	$\overline{SCK0}$ (CMOS I/O)
		1	0	0	Note 2 ×	Note 2 ×	0	1			SB0 (N-ch open-drain I/O)	P26 (CMOS I/O)	
1	1	2-wire serial I/O mode (refer to **13.4.4 2-wire serial I/O mode operation**) or I^2C bus mode (refer to **13.4.5 I^2C bus mode operation**).											

R/W

WUP	Control of wakeup function[Note 3]
0	Interrupt request signal generated with each serial transfer in any mode
1	Interrupt request signal generated when the address received after bus release (when CMDD = RELD = 1) matches the slave address register 0 data in SBI mode

R

COI	Slave address comparison result flag[Note 4]
0	Slave address register 0 and serial I/O shift register 0 data do not match
1	Slave address register 0 and serial I/O shift register 0 data match

R/W

CSIE0	Control of serial interface SIO0 operation
0	Operation stopped
1	Operation enabled

Caution When using $\overline{SCK0}$ or SCL, set P27 to 1. If P27 is set to 0, it always outputs a low level.

Notes 1. Bit 6 (COI) is a read-only bit.

2. Can be used freely as a port.

3. Clear bit 5 (SIC) of interrupt timing specification register 0 (SINT0) to 0 when using the wake-up function in the SBI mode (WUP=1).

4. When CSIE0 = 0, COI is 0.

Remark ×: Don't care

PM××: Port mode register

P××: Output latch of port

(b) Serial bus interface control register 0 (SBIC0)

SBIC0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SBIC0 to 00H.

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SBIC0	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	FF61H	00H	R/W[Note]

R/W

RELT	Used for bus release signal output. When RELT = 1, SO latch is set to (1). After SO latch setting, automatically cleared to (0). Also cleared to 0 when CSIE0 = 0.

R/W

CMDT	Used for command signal output. When CMDT = 1, SO latch is cleared to (0). After SO latch clearance, automatically cleared to (0). Also cleared to 0 when CSIE0 = 0.

R

RELD	Detection of bus release	
Clear conditions (RELD = 0)		Set conditions (RELD = 1)
• When transfer start instruction is executed • If SIO0 and SVA0 values do not match in address reception • When CSIE0 = 0 • When reset input is applied		• When bus release signal (REL) is detected

R

CMDD	Detection of command	
Clear conditions (CMDD = 0)		Set conditions (CMDD = 1)
• When transfer start instruction is executed • When bus release signal (REL) is detected • When CSIE0 = 0 • When reset input is applied		• When command signal (CMD) is detected

R/W

ACKT	The acknowledge signal is output in synchronization with the falling edge of $\overline{SCK0}$ just after execution of the instruction to be set to (1) and, after acknowledge signal output, is automatically cleared to (0). Used as ACKE = 0. Also cleared to (0) upon start of serial interface transfer or when CSIE0 = 0.

R/W

ACKE	Control of acknowledge signal output	
0	Acknowledge signal automatic output disabled (output with ACKT enabled)	
1	Before completion of transfer	The acknowledge signal is output in synchronization with the 9th falling edge of $\overline{SCK0}$ (automatically output when ACKE = 1).
	After completion of transfer	The acknowledge signal is output in synchronization with the falling edge of $\overline{SCK0}$ just after execution of the instruction to be set to 1 (automatically output when ACKE = 1). However, not automatically cleared to 0 after acknowledge signal output.

(Continued)

Note Bits 2, 3, and 6 (RELD, CMDD and ACKD) are read-only bits.

Remarks 1. Bits 0, 1, and 4 (RELT, CMDT, and ACKT) are 0 when read after data setting.

2. CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

R

ACKD	Detection of acknowledge	
Clear conditions (ACKD = 0)		Set conditions (ACKD = 1)
• When $\overline{SCK0}$ falls immediately after the busy mode is released after the transfer start instruction is executed. • When CSIE0 = 0 • When reset input is applied		• When the acknowledge signal ($\overline{ACK}$) is detected at the rising edge of the $\overline{SCK0}$ clock after completion of transfer

R/W

Note BSYE	Control of busy signal output synchronization
0	Busy signal output in synchronization with the falling edge of $\overline{SCK0}$ clock just after execution of the instruction to be cleared to (0) (makes READY status) disabled.
1	Busy signal output at the falling edge of $\overline{SCK0}$ clock following the acknowledge signal.

Note Busy mode can be cleared by the start of serial interface transfer. However, the BSYE flag is not cleared to 0.

Remark CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

(c) Interrupt timing specification register 0 (SINT0)

SINT0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SINT0 to 00H.

Symbol	7	<6>	<5>	<4>	<3>	<2>	1	0	Address	After reset	R/W
SINT0	0	CLD	SIC	SVAM	CLC	WREL	WAT1	WAT0	FF63H	00H	R/W[Note 1]

R/W

SVAM	SVA0 bits to be used as slave address
0	Bits 0 to 7
1	Bits 7 to 0

R/W

SIC	Selection of INTCSI0 interrupt source[Note 2]
0	CSIIF0 is set upon termination of serial interface SIO0 transfer
1	CSIIF0 is set upon bus release detection or termination of serial interface SIO0 transfer

R

CLD	$\overline{\text{SCK0}}$/SCL/P27 pin level[Note 3]
0	Low level
1	High level

Notes 1. Bit 6 (CLD) is a read-only bit.

2. When using the wakeup function in the SBI mode, set SIC to 0.

3. When CSIE0 = 0, CLD is 0.

Caution Be sure to set bits 0 to 3 to 0 when using SBI mode.

Remark SVA0: Slave address register 0

CSIIF: Interrupt request flag corresponding to INTCSI0

CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

(4) Signals

Figures 13-21 to 13-26 show the signals and flag operations in SBI. Table 13-3 lists the signals in SBI.

Figure 13-21. RELT, CMDT, RELD, and CMDD Operations (Master)

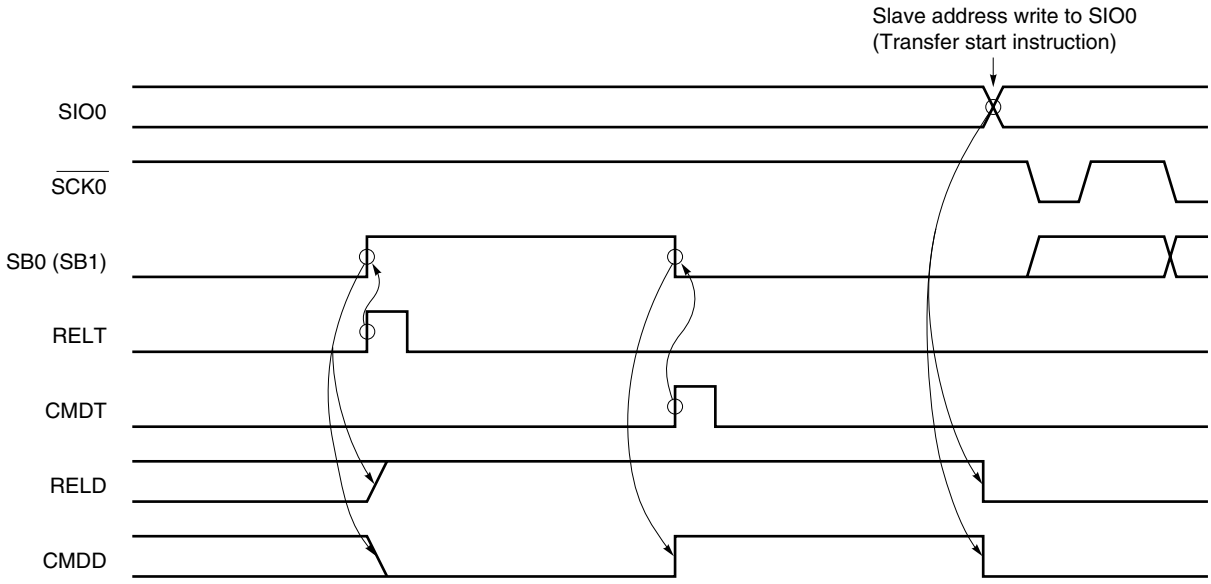


Figure 13-22. RELD and CMDD Operations (Slave)

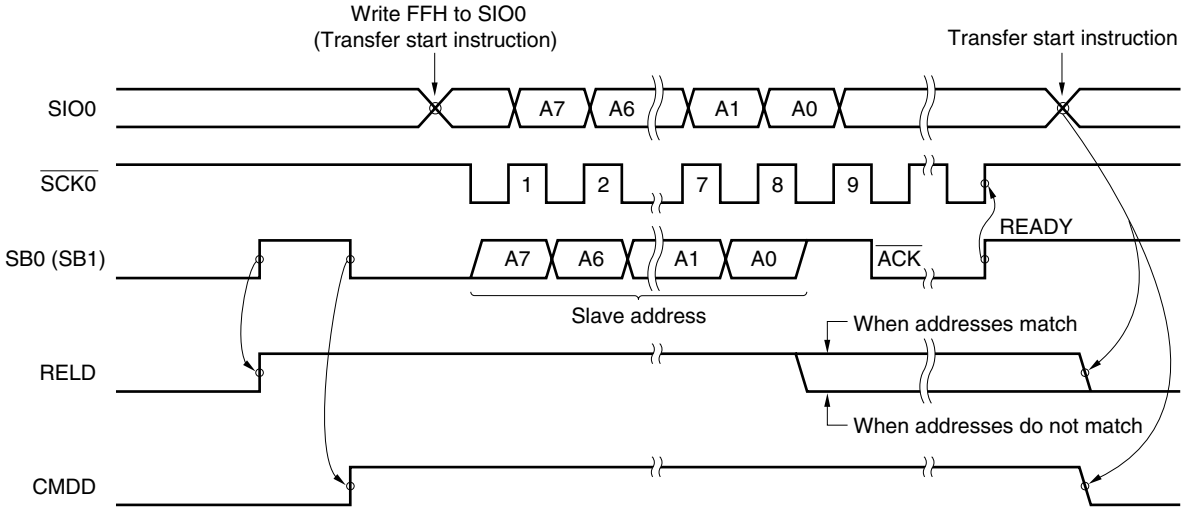
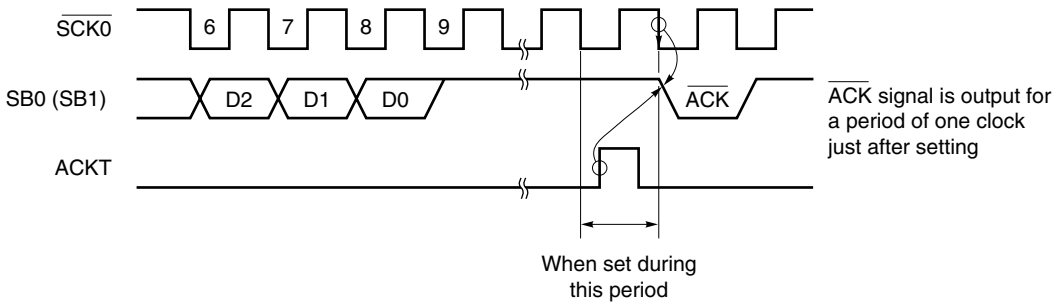


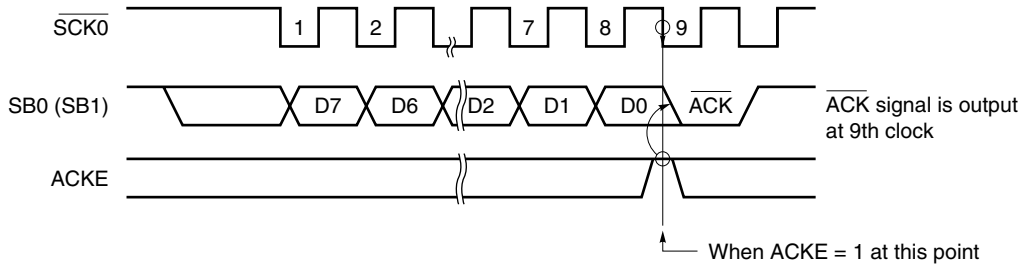
Figure 13-23. ACKT Operation



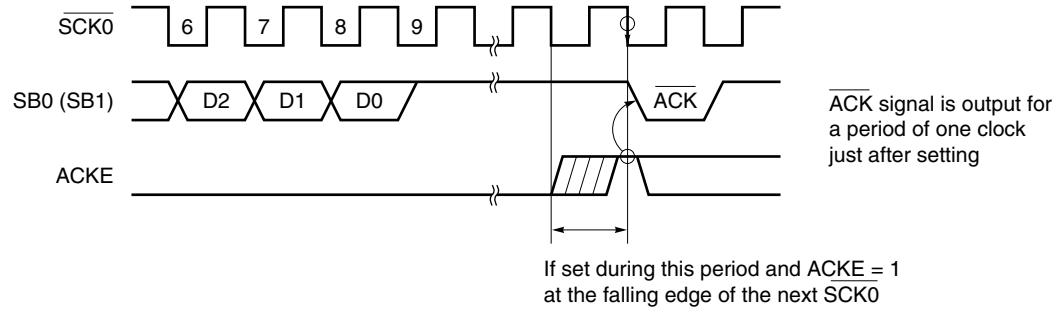
Caution Do not set ACKT before termination of transfer.

Figure 13-24. ACKE Operations

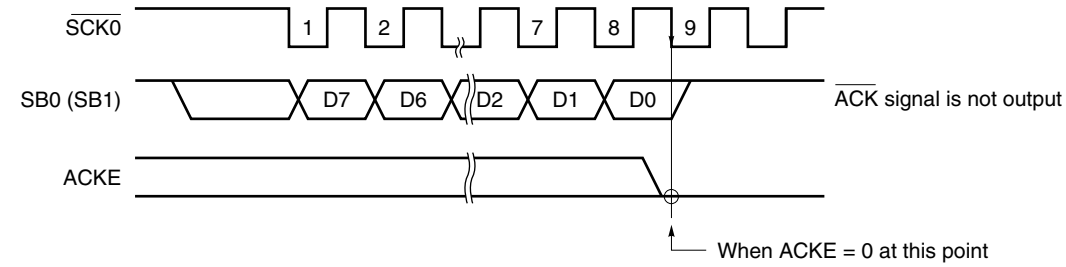
(a) When ACKE = 1 upon completion of transfer



(b) When set after completion of transfer



(c) When ACKE = 0 upon completion of transfer



(d) When "ACKE = 1" period is short

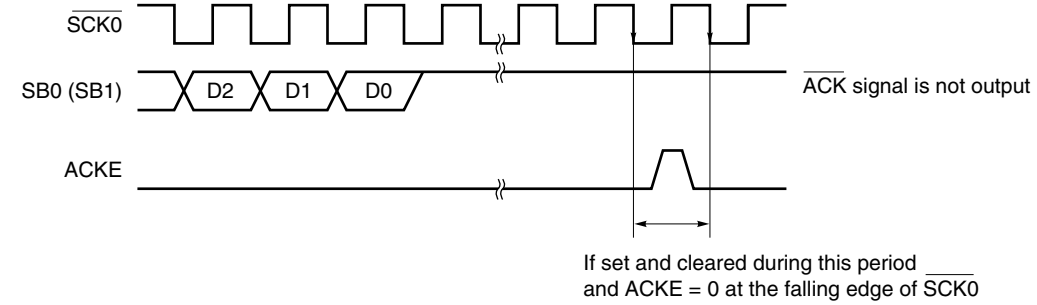
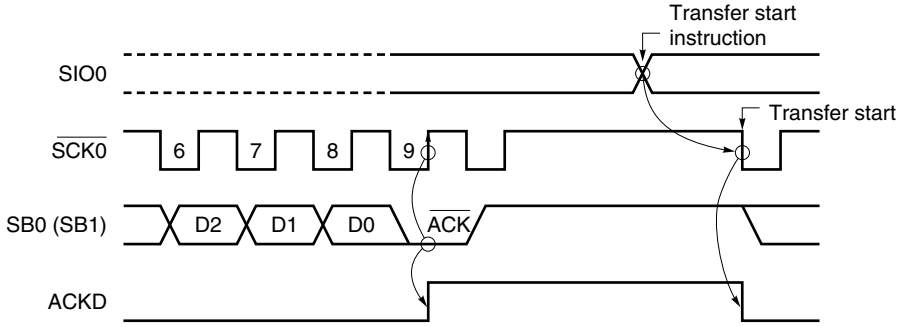
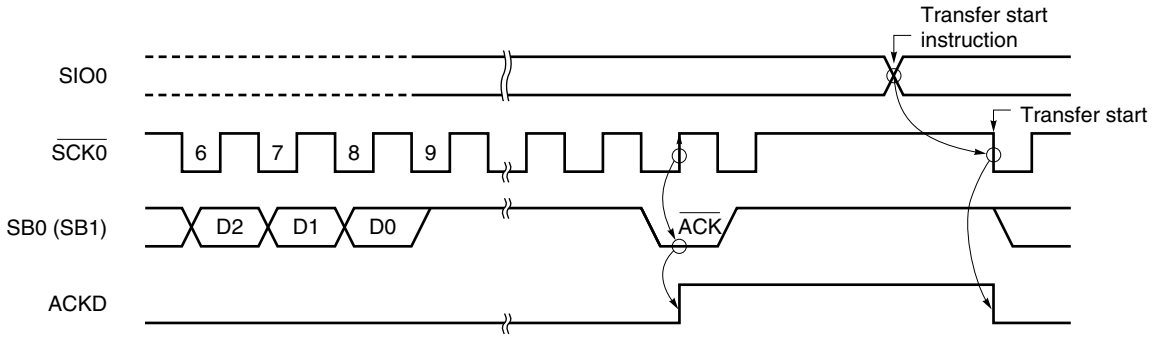


Figure 13-25. ACKD Operations

(a) When $\overline{ACK}$ signal is output at 9th clock of $\overline{SCK0}$



(b) When $\overline{ACK}$ signal is output after 9th clock of $\overline{SCK0}$



(c) Clear timing when transfer start is set during BUSY

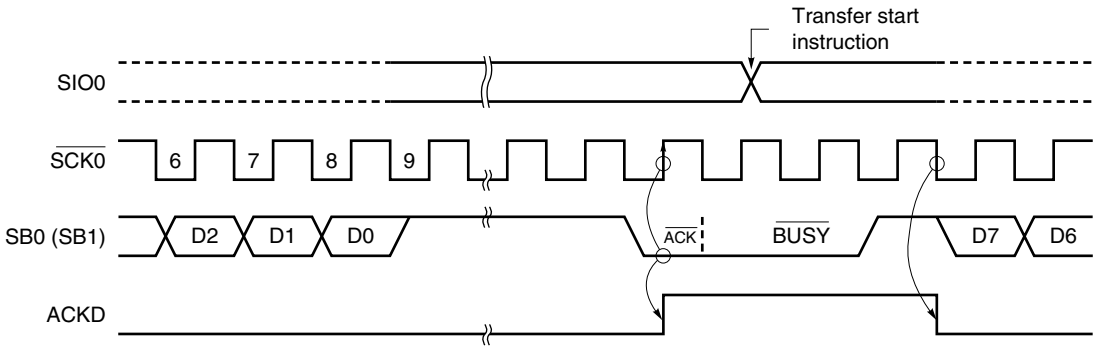


Figure 13-26. BSYE Operation

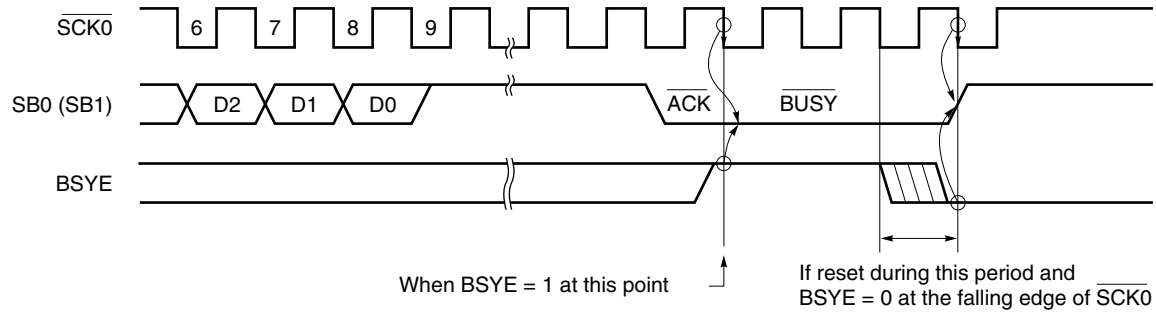


Table 13-3. Signals in SBI Mode (1/2)

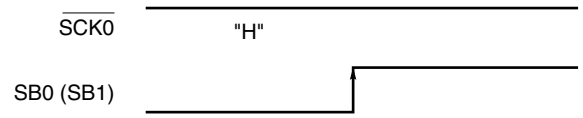
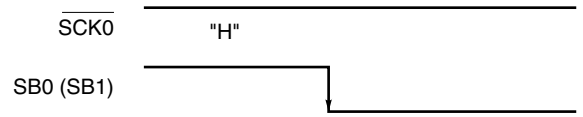
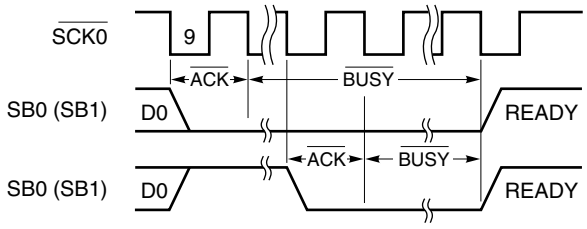
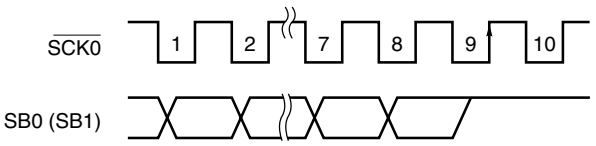
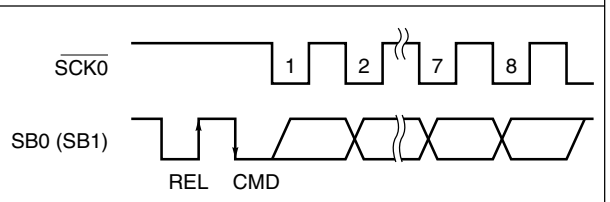
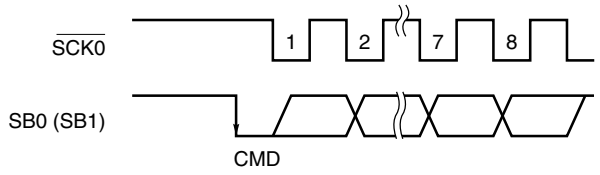
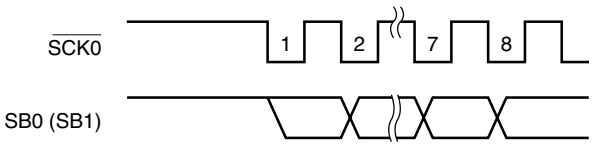
Signal Name	Output Device	Definition	Timing Chart	Output Condition	Effect on Flag	Meaning of Signal
Bus release signal (REL)	Master	SB0 (SB1) rising edge when $\overline{SCK0}$ = 1		RELT set	• RELD set • CMDD cleared	CMD signal is output to indicate that transmit data is an address.
Command signal (CMD)	Master	SB0 (SB1) falling edge when $\overline{SCK0}$ = 1		CMDT set	CMDD set	i) Transmit data is an address after REL signal output. ii) REL signal is not output and transmit data is a command.
Acknowledge signal ($\overline{ACK}$)	Master/ slave	Low-level signal to be output to SB0 (SB1) during one-clock period of $\overline{SCK0}$ after completion of serial reception	[Synchronous BUSY output]	①ACKE = 1 ②ACKT set	ACKD set	Completion of reception
Busy signal ($\overline{BUSY}$)	Slave	[Synchronous BUSY signal] Low-level signal to be output to SB0 (SB1) following acknowledge signal		BSYE = 1	—	Serial receive disabled because of processing
Ready signal (READY)	Slave	High-level signal to be output to SB0 (SB1) before serial transfer start and after completion of serial transfer		①BSYE = 0 ②Execution of instruction for data write to SIO0 (transfer start instruction)	—	Serial receive enabled

Table 13-3. Signals in SBI Mode (2/2)

Signal Name	Output Device	Definition	Timing Chart	Output Condition	Effects on Flag	Meaning of Signal
Serial clock ($\overline{SCK0}$)	Master	Synchronous clock to output address/command/ data, $\overline{ACK}$ signal, synchro- nous $\overline{BUSY}$ signal, etc. Address/command/data are transferred with the first eight synchronous clocks.		When CSIE0 = 1, execution of instruction for data write to SIO0 (serial transfer start instruction)**Note 2**	CSIIF0 set (rising edge of 9th clock of $\overline{SCK0}$)**Note 1**	Timing of signal output to serial data bus
Address (A7 to A0)	Master	8-bit data to be transferred in synchronization with $\overline{SCK0}$ after output of REL and CMD signals				Address value of slave device on the serial bus
Command (C7 to C0)	Master	8-bit data to be transferred in synchronization with $\overline{SCK0}$ after output of only CMD signal without REL signal output				Instructions and messages to the slave device
Data (D7 to D0)	Master/ slave	8-bit data to be transferred in synchronization with $\overline{SCK0}$ without output of REL and CMD signals				Numeric values to be processed by slave or master device

Notes 1. When WUP = 0, CSIIF0 is set at the rising edge of the 9th clock of $\overline{SCK0}$.
When WUP = 1, an address is received. CSIIF0 is set only when the address matches the slave address register 0 (SVA0) value.

2. In the $\overline{BUSY}$ state, transfer starts after the READY state is set.

(5) Pin configuration

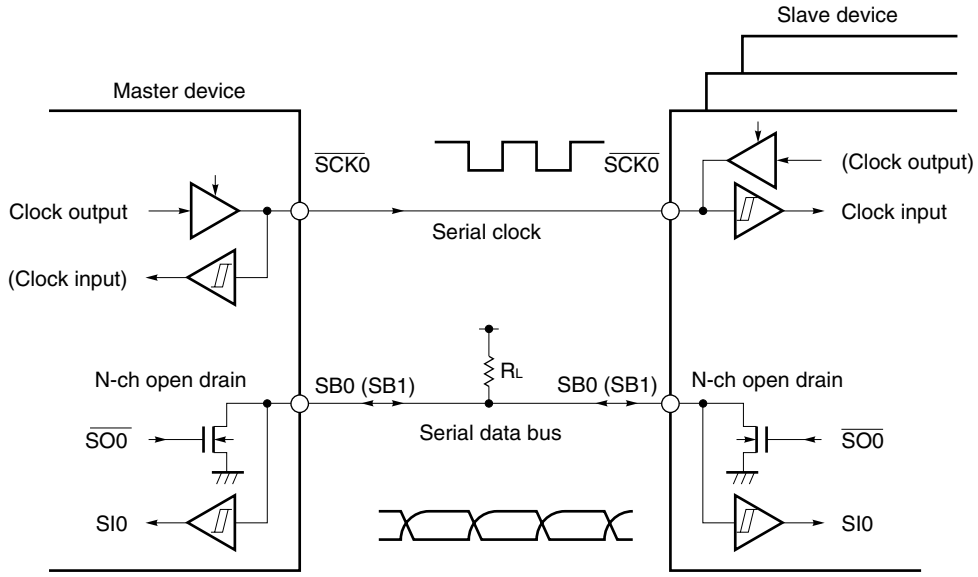
The serial clock pin ($\overline{SCK0}$) and serial data bus pin SB0 (SB1) have the following configurations.

(a) $\overline{SCK0}$ ..... Serial clock I/O pin
<1> Master ... CMOS and push-pull output
<2> Slave ..... Schmitt input

(b) SB0 (SB1) .... Serial data I/O alternate-function pin
Both master and slave devices have an N-ch open-drain output and a Schmitt input.

Because the serial data bus line has an N-ch open-drain output, an external pull-up resistor is necessary.

Figure 13-27. Pin Configuration



Caution Because the N-ch open-drain output must be made high impedance when data is received, write FFH to SIO0 in advance. The N-ch open-drain output can be made high impedance throughout transfer. However, when the wakeup function specification bit (WUP) = 1, the N-ch open-drain output is always made high impedance. Thus, it is not necessary to write FFH to SIO0.

(6) Address match detection method

In the SBI mode, a particular slave device is selected by address communication from the master device and communication is started.

Address match detection is executed by hardware. CSIIF0 is set in the wakeup state (WUP = 1) only when the address transmitted from the master device matches the value set to slave address register 0 (SVA0).

Cautions 1. Slave selection/non-selection is detected by matching of the slave address received after bus release (RELD = 1).
For this match detection, the match interrupt request (INTCSI0) of the address generated with WUP = 1 is normally used. Thus, execute selection/non-selection detection using the slave address when WUP = 1.
2. When detecting selection/non-selection without using an interrupt request with WUP = 0, do so by means of transmission/reception of a command preset by the program instead of using the address match detection method.

(7) Error detection

In the SBI mode, the serial bus SB0 (SB1) status being transmitted is fetched into the destination device, that is, serial I/O shift register 0 (SIO0). Thus, transmit errors can be detected in the following way.

(a) Method of comparing SIO0 data before transmission to that after transmission

In this case, if two data differ from each other, a transmit error is judged to have occurred.

(b) Method of using slave address register 0 (SVA0)

Transmit data is set to both SIO0 and SVA0 and is transmitted. After termination of transmission, the COI bit (match signal coming from the address comparator) of serial operating mode register 0 (CSIM0) is tested. If “1”, normal transmission is judged to have been carried out. If “0”, a transmit error is judged to have occurred.

(8) Communication operation

In the SBI mode, the master device normally selects one slave device as the communication target from among two or more devices by outputting an “address” to the serial bus.

After the communication target device has been determined, commands and data are transmitted/received and serial communication is realized between the master and slave device.

Figures 13-28 to 13-31 show the data communication timing charts.

The shift operation of serial I/O shift register 0 (SIO0) is carried out at the falling edge of the serial clock ($\overline{SCK0}$). Transmit data is latched into the SO0 latch and is output MSB-first from the SB0/P25 or SB1/P26 pin. Receive data input to the SB0 (or SB1) pin at the rising edge of $\overline{SCK0}$ is latched into SIO0.

Figure 13-28. Address Transmission from Master Device to Slave Device (WUP = 1)

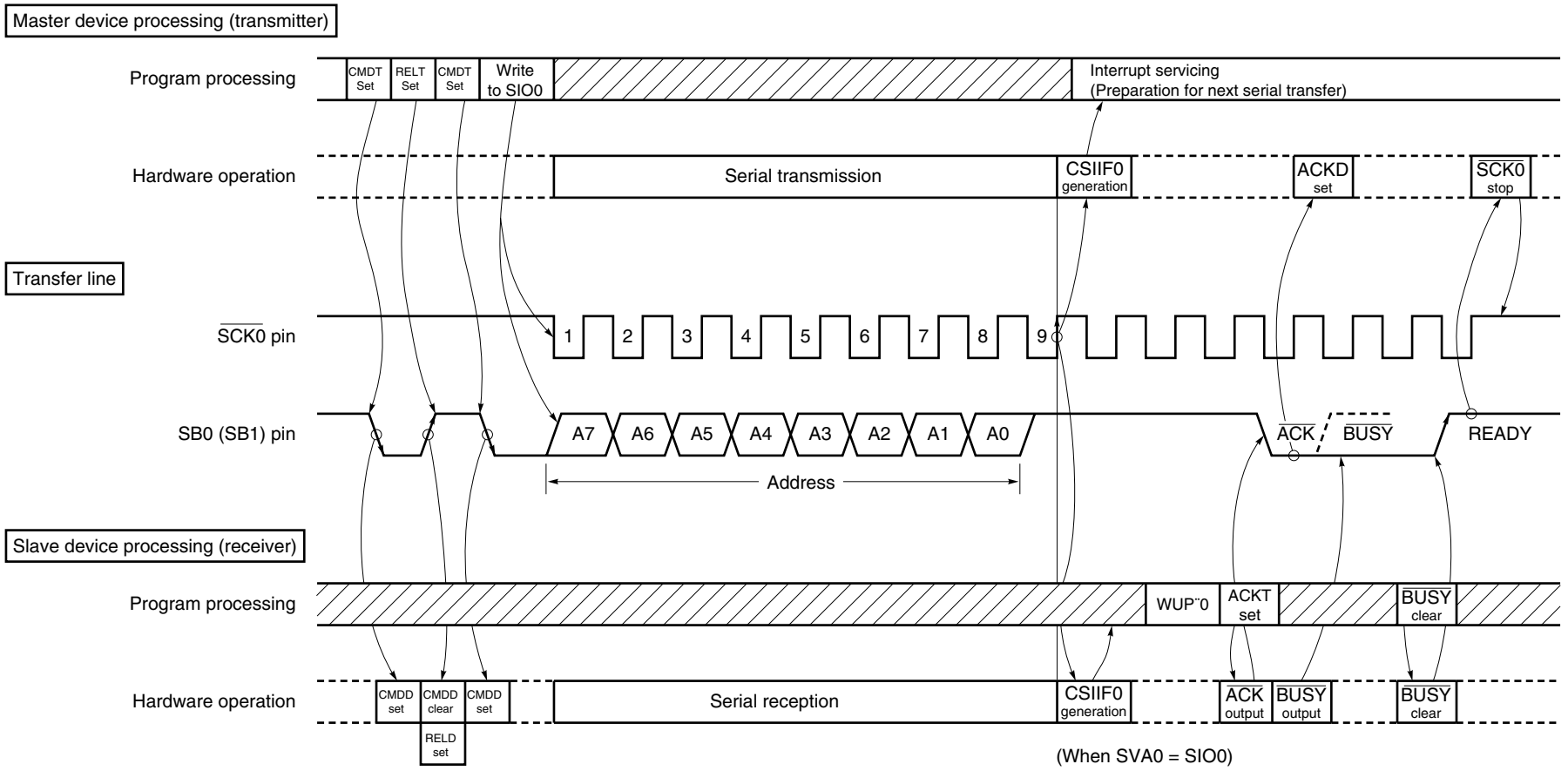


Figure 13-29. Command Transmission from Master Device to Slave Device

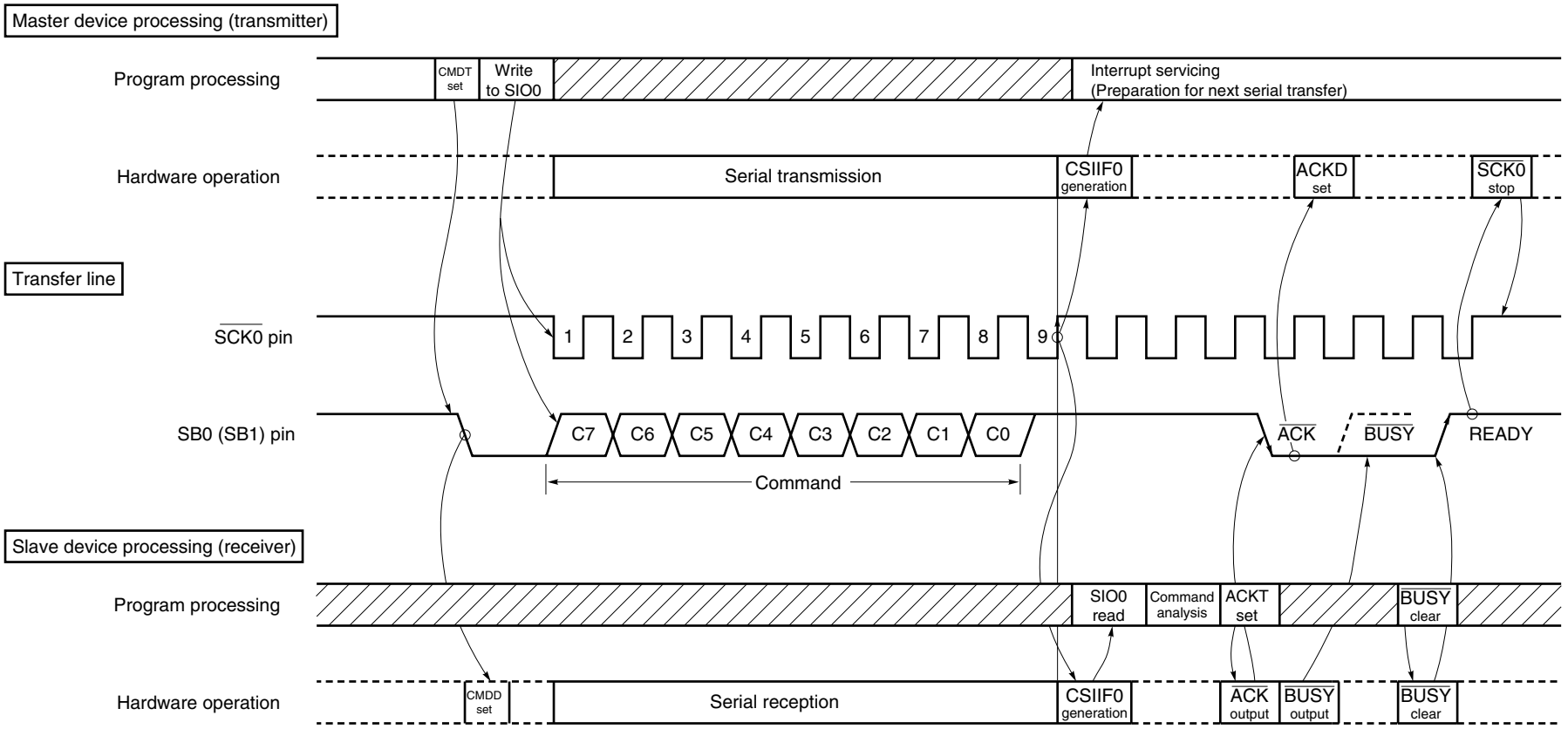


Figure 13-30. Data Transmission from Master Device to Slave Device

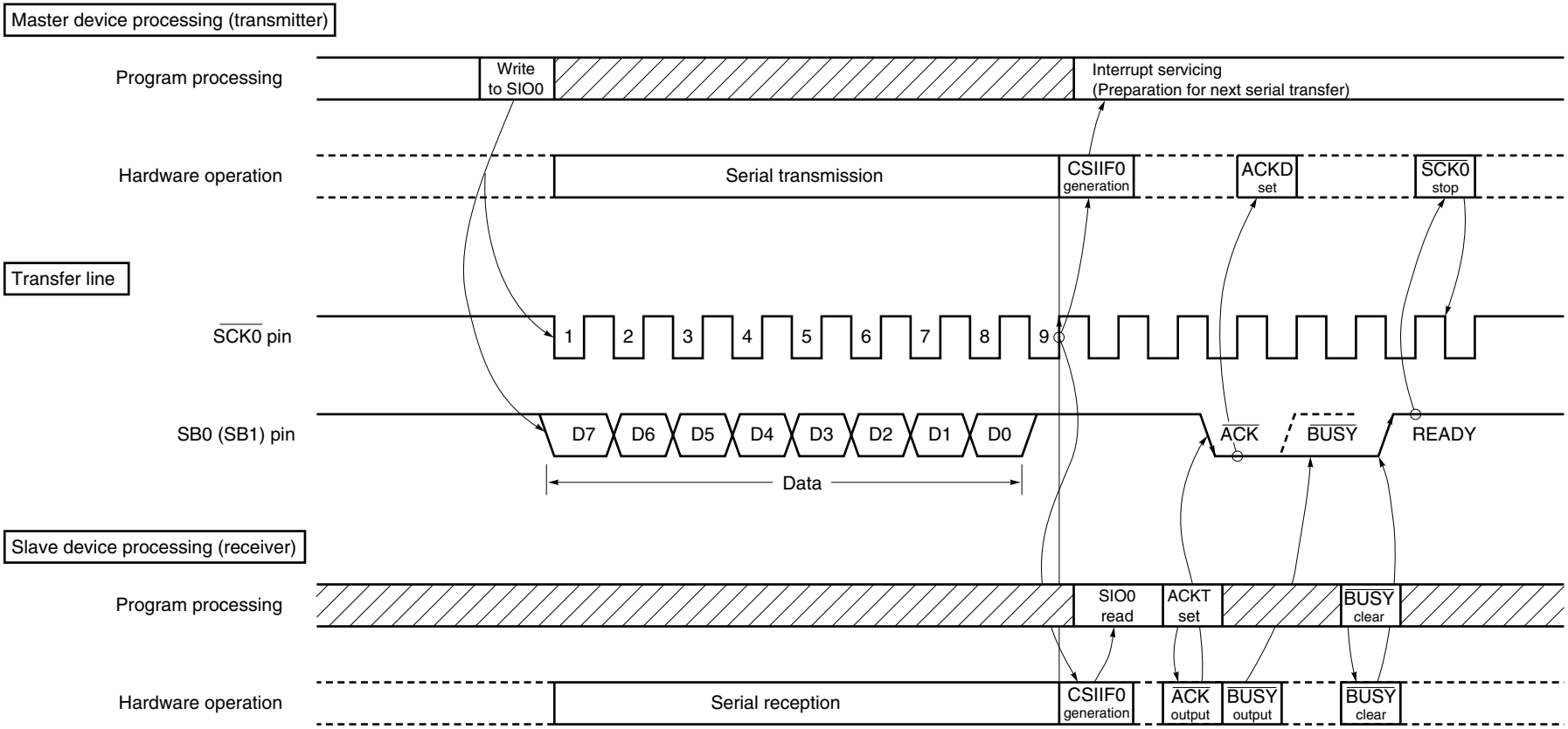
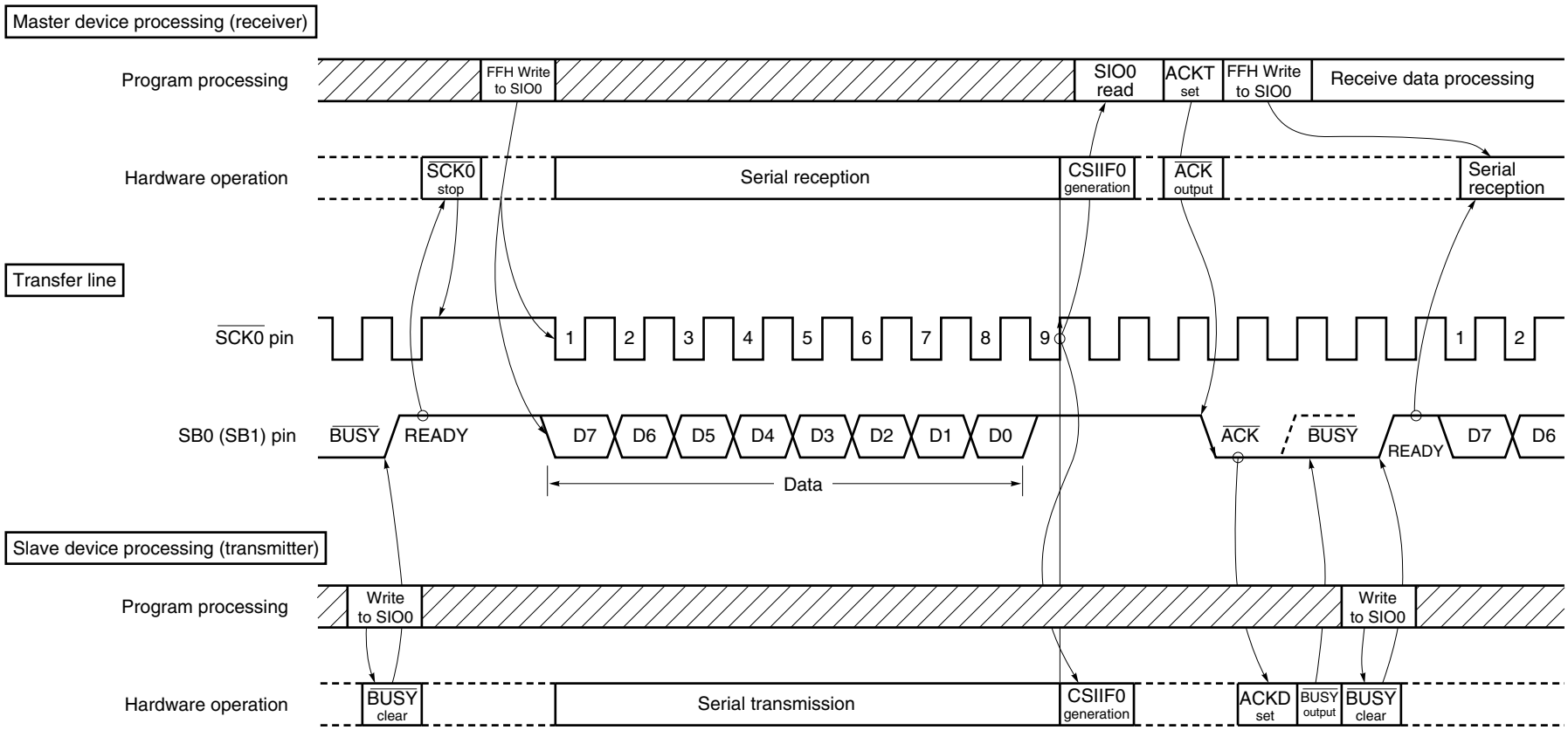


Figure 13-31. Data Transmission from Slave Device to Master Device



(9) Transfer start

Serial transfer is started by setting transfer data to serial I/O shift register 0 (SIO0) when the following two conditions are satisfied.

- Serial interface SIO0 operation control bit (CSIE0) = 1
- Internal serial clock is stopped or $\overline{SCK0}$ is at high level after 8-bit serial transfer.

Cautions 1. If CSIE0 is set to "1" after data is written to SIO0, transfer does not start.

2. Because the N-ch open-drain output must be made high impedance for data reception, write FFH to SIO0 in advance.
However, when the wakeup function specification bit (WUP) = 1, the N-ch open-drain output is always made high impedance. Thus, it is not necessary to write FFH to SIO0.

3. If data is written to SIO0 when the slave is busy, the data is not lost.
When the busy state is cleared and the SB0 (or SB1) input is set to the high level (READY) state, transfer starts.

Upon termination of 8-bit transfer, serial transfer automatically stops and the interrupt request flag (CSIIF0) is set.

Be sure to set the pins used to input/output data after input of the RESET signal and before serial transfer of 1 byte as follows.

<1> Set the output latches of P25 and P26 to 1.
<2> Set bit 0 (RELT) of serial bus interface control register 0 (SBIC0) to 1.
<3> Clear the output latches of P25 and P26, which were set to 1 above, to 0.

(10) Method used to judge busy state of a slave

When the device is in the master mode, use the method below to judge whether the slave device is in the busy state or not.

<1> Detect acknowledge signal ($\overline{ACK}$) or interrupt request signal generation.
<2> Set the port mode register PM25 (or PM26) of the SB0/P25 (or SB1/P26) pin to the input mode.
<3> Read out the pin state (when the pin level is high, the READY state is set).

After the detection of the READY state, set the port mode register to 0 and return to the output mode.

(11) SBI mode cautions

(a) Slave selection/non-selection is detected by match detection of the slave address received after bus release (RELD = 1).
For this match detection, the match interrupt (INTCSI0) of the address generated with WUP = 1 is normally used. Thus, execute selection/non-selection detection using the slave address when WUP = 1.

(b) When detecting selection/non-selection without using an interrupt with WUP = 0, do so by means of transmission/reception of a command preset by the program instead of using the address match detection method.

(c) In the SBI mode, SBI outputs the $\overline{BUSY}$ signal after the $\overline{BUSY}$ clear instruction has been executed until the next serial clock falls. If WUP is set to 1 by mistake during this period, $\overline{BUSY}$ will not be cleared. Before setting WUP to 1, therefore, clear $\overline{BUSY}$, and make sure that the SB0 (SB1) pin has gone high.

(d) For pins which are to be used for data I/O, be sure to carry out the following settings before serial transfer of the 1st byte after reset input.

<1> Set the P25 and P26 output latches to 1.
<2> Set bit 0 (RELT) of the serial bus control register to 1.
<3> Reset the P25 and P26 output latches from 1 to 0.

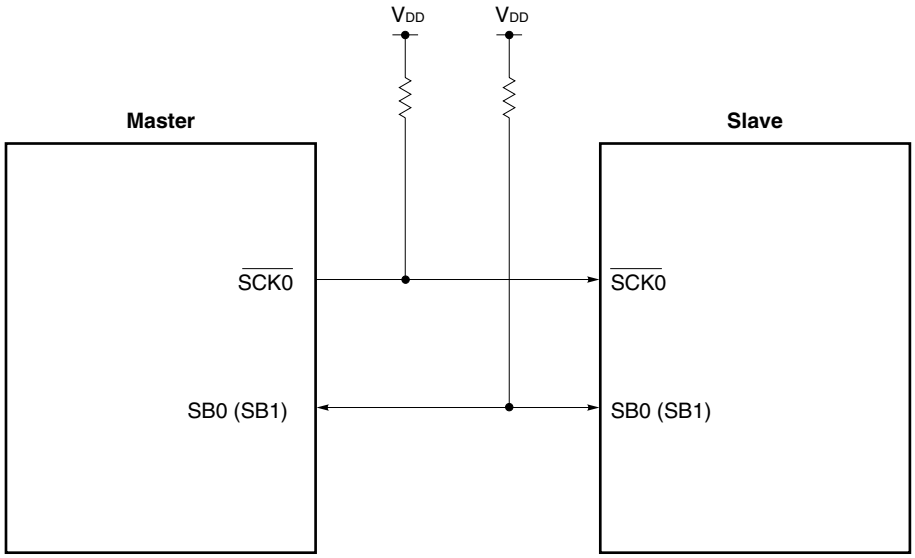
(e) A transition of the SB0 (SB1) pin from low to high or high to low is recognized as a bus release signal or a command signal when the $\overline{SCK0}$ line is high. If the change timing of the bus is shifted due to the influence of board capacitance, data that is transmitted may be identified as a bus release signal or a command signal by mistake. Exercise care when wiring.

13.4.4 2-wire serial I/O mode operation

The 2-wire serial I/O mode can cope with any communication format by program.

Communication is basically carried out with two lines, one for the serial clock ($\overline{SCK0}$) and one for serial data I/O (SB0 or SB1).

Figure 13-32. Serial Bus Configuration Example Using 2-Wire Serial I/O Mode



(1) Register setting

The 2-wire serial I/O mode is set using serial operating mode register 0 (CSIM0), serial bus interface control register 0 (SBIC0), interrupt timing specification register 0 (SINT0), port mode register 2 (PM2), and port 2 (P2).

(a) Serial operating mode register 0 (CSIM0)

CSIM0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM0 to 00H.

Symbol	<7>	<6>	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM0	CSIE0	COI	WUP	CSIM04	CSIM03	CSIM02	CSIM01	0	FF61H	00H	R/W[Note 1]

R/W

CSIM01	Selection of serial interface SIO0 clock selection
0	Clock input to $\overline{\text{SCK0}}$/SCL/P27 pin from off-chip
1	Clock specified by bits 0 to 3 of serial interface clock select register 0 (SCL0)

R/W

CSIM 04	CSIM 03	CSIM 02	PM25	P25	PM26	P26	PM27	P27	Operating mode	Start bit	SIO/SB0/SDA0/P25 pin function	SO0/SB1/SDA1/P26 pin function	$\overline{\text{SCK0}}$/SCL/P27 pin function
0	×	3-wire serial I/O mode (refer to **13.4.2 3-wire serial I/O mode operation**)											
1	0	SBI mode (Refer to **13.4.3 SBI mode operation**)											
1	1	0	Note 2	Note 2	0	0	0	1	2-wire serial I/O mode or I^2C bus mode (Refer to **13.4.5**)	MSB	P25 (CMOS I/O)	SB1/SDA1 (N-ch open-drain I/O)	$\overline{\text{SCK0}}$/SCL (N-ch open-drain I/O)
		1	0	0	Note 2	Note 2	0	1			SB0/SDA0 (N-ch open-drain I/O)	P26 (CMOS I/O)	

R/W

WUP	Control of wakeup function
0	Interrupt request signal generated with each serial transfer in any mode
1	Setting prohibited[Note 3]

R

COI	Slave address comparison result flag [Note 4]
0	Slave address register 0 and serial I/O shift register 0 data do not match
1	Slave address register 0 and serial I/O shift register 0 data match

R/W

CSIE0	Control of serial interface SIO0 operation
0	Operation stopped
1	Operation enabled

Caution When using $\overline{\text{SCK0}}$, set P27 to 1. If P27 is set to 0, it always outputs a low level.

Notes 1. Bit 6 (COI) is a read-only bit.

2. Can be used freely as a port.

3. Be sure to set WUP to 0 in the 2-wire serial I/O mode.

4. When CSIE0 = 0, COI is 0.

Remark ×: Don't care

PM××: Port mode register

P××: Output latch of port

(b) Serial bus interface control register 0 (SBIC0)

SBIC0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SBIC0 to 00H.

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SBIC0	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	FF61H	00H	R/W

R/W

RELT	When RELT = 1, SO latch is set to 1. After SO latch setting, automatically cleared to 0. Also cleared to 0 when CSIE0 = 0.

R/W

CMDT	When CMDT = 1, SO latch is cleared to 0. After SO latch clearance, automatically cleared to 0. Also cleared to 0 when CSIE0 = 0.

(c) Interrupt timing specification register 0 (SINT0)

SINT0 is set with a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SINT0 to 00H.

Symbol	7	<6>	<5>	<4>	<3>	<2>	1	0	Address	After reset	R/W
SINT0	0	CLD	SIC	SVAM	CLC	WREL	WAT1	WAT0	FF63H	00H	R/W[Note 1]

R

CLD	$\overline{SCK0}$/SCL/P27 pin level[Note 2]
0	Low level
1	High level

Caution Be sure to set bits 0 to 3 to 0 when using in 2-wire serial I/O mode.

Notes 1. Bit 6 (CLD) is a read-only bit.

2. When CSIE0 = 0, CLD is 0.

Remark CSIE0: Bit 7 of serial operating mode register 0 (CSIM0)

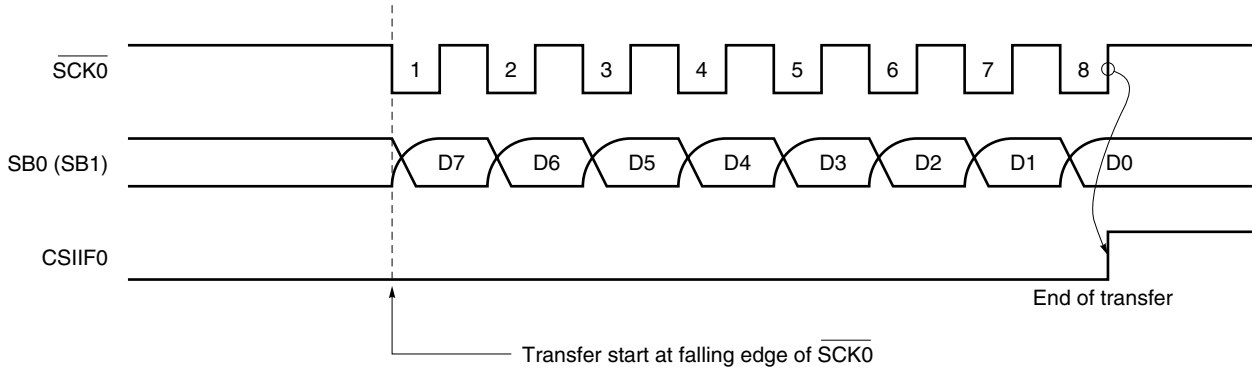
(2) Communication operation

The 2-wire serial I/O mode is used for data transmission/reception in 8-bit units. Data transmission/reception is carried out bit-wise in synchronization with the serial clock.

The shift operation of serial I/O shift register 0 (SIO0) is carried out in synchronization with the falling edge of the serial clock ($\overline{SCK0}$). The transmit data is held in the SO0 latch and is output from the SB0/P25 (or SB1/P26) pin on an MSB-first basis. The receive data input from the SB0 (or SB1) pin is latched into the shift register at the rising edge of $\overline{SCK0}$.

Upon termination of 8-bit transfer, the shift register operation stops automatically and the interrupt request flag (CSIIF0) is set.

Figure 13-33. 2-Wire Serial I/O Mode Timing



The SB0 (or SB1) pin specified for the serial data bus is an N-ch open-drain I/O and thus it must be externally connected to a pull-up resistor. Because it is necessary to make the N-ch open-drain output high impedance for data reception, write FFH to SIO0 in advance.

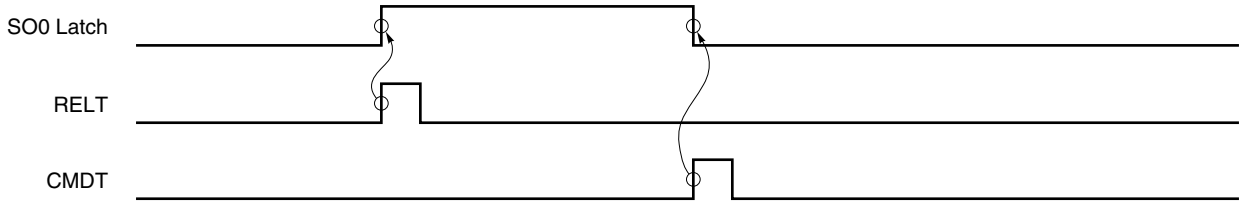
The SB0 (or SB1) pin generates the SO0 latch status and thus the SB0 (or SB1) pin output status can be manipulated by setting bit 0 (RELT) and bit 1 (CMDT) of serial bus interface control register 0 (SBIC0). However, do not carry out this manipulation during serial transfer.

Control the $\overline{SCK0}$ pin output level in the output mode (internal system clock mode) by manipulating the P27 output latch (refer to **13.4.8 $\overline{SCK0}$/SCL/P27 pin output manipulation**).

(3) Signals

Figure 13-34 shows the RELT and CMDT operations.

Figure 13-34. RELT and CMDT Operations



(4) Transfer start

Serial transfer is started by setting transfer data to serial I/O shift register 0 (SIO0) when the following two conditions are satisfied.

- Serial interface SIO0 operation control bit (CSIE0) = 1
- Internal serial clock is stopped or $\overline{SCK0}$ is at high level after 8-bit serial transfer.

Cautions 1. If CSIE0 is set to "1" after data is written to SIO0, transfer does not start.

2. Because the N-ch open-drain output must be made high impedance for data reception, write FFH to SIO0 in advance.

Upon termination of 8-bit transfer, serial transfer automatically stops and the interrupt request flag (CSIIF0) is set.

(5) Error detection

In the 2-wire serial I/O mode, the serial bus SB0 (SB1) status being transmitted is fetched into the destination device, that is, serial I/O shift register 0 (SIO0). Thus, a transmit error can be detected in the following way.

(a) Method of comparing SIO0 data before transmission to that after transmission

In this case, if the two data differ from each other, a transmit error is judged to have occurred.

(b) Method of using slave address register 0 (SVA0)

Transmit data is set to both SIO0 and SVA0 and is transmitted. After termination of transmission, the COI bit (match signal coming from the address comparator) of serial operating mode register 0 (CSIM0) is tested. If "1", normal transmission is judged to have been carried out. If "0", a transmit error is judged to have occurred.

13.4.5 I^2C bus mode operation

The I^2C bus mode is provided for when communication operations are performed between a single master device and multiple slave devices. This mode configures a serial bus that includes only a single master device, and is based on the clocked serial I/O format with the addition of bus configuration functions, which allows the master device to communicate with a number of (slave) devices using only two lines: a serial clock (SCL) line and a serial data bus (SDA0 or SDA1) line. Consequently, when the user plans to configure a serial bus which includes multiple microcontrollers and peripheral devices, using this configuration results in reduction of the required number of port pins and on-board wiring.

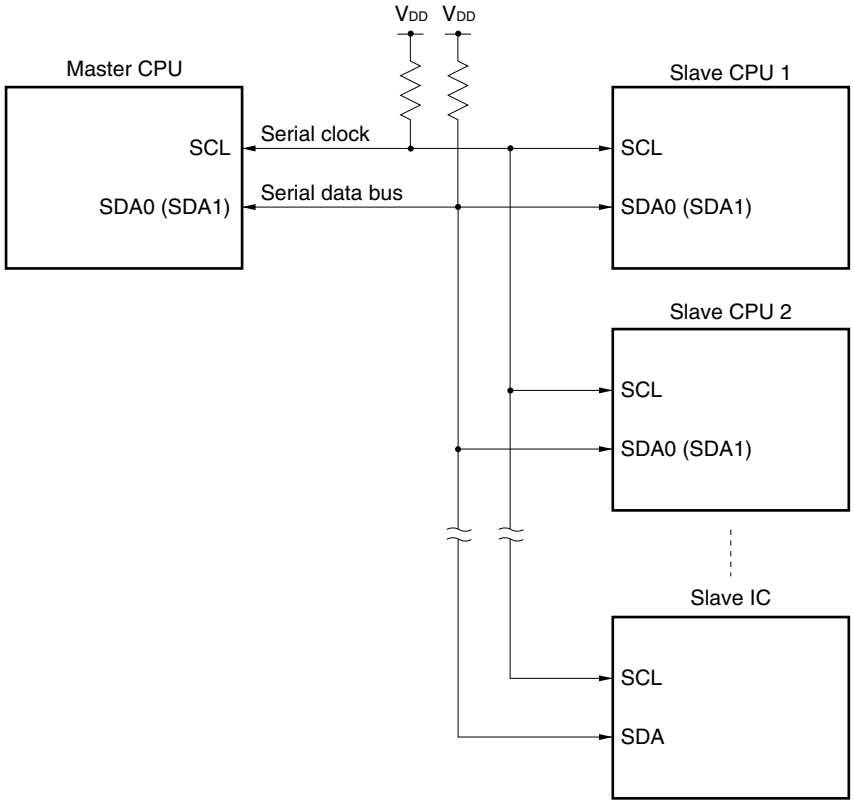
In the I^2C bus specification, the master sends start condition, data, and stop condition signals to slave devices via the serial data bus, while slave devices automatically detect and distinguish the type of signals using the signal detection function incorporated as hardware. This simplifies I^2C bus control sections in the application program.

An example of a serial bus configuration is shown in Figure 13-35. The system below is composed of CPUs and peripheral ICs having serial interface hardware that complies with the I^2C bus specification.

Note that pull-up resistors are required to connect to both serial clock line and serial data bus line, because N-ch open-drain outputs are used for the serial clock pin (SCL) and the serial data bus pin (SDA0 or SDA1) on the I^2C bus.

The signals used in the I^2C bus mode are described in Table 13-4.

Figure 13-35. Serial Bus Configuration Example Using I^2C Bus



(1) I^2C bus mode functions

In the I^2C bus mode, the following functions are available.

(a) Automatic identification of serial data

Slave devices automatically detect and identify start condition, data, and stop condition signals sent via the serial data bus.

(b) Chip selection by specifying device address

The master device can select a specific slave device connected to the I^2C bus and communicate with it by sending in advance the address data corresponding to the destination device.

(c) Wakeup function

When the received address matches the value of slave address register 0 (SVA0), the slave device internally generates an interrupt signal (also generated when a stop condition is detected). Therefore, CPUs other than the selected slave device on the I^2C bus can perform independent operations during serial communication.

(d) Acknowledge signal ($\overline{ACK}$) control function

The master device and a slave device send and receive acknowledge signals to confirm that serial communication has been executed normally.

(e) Wait signal ($\overline{WAIT}$) control function

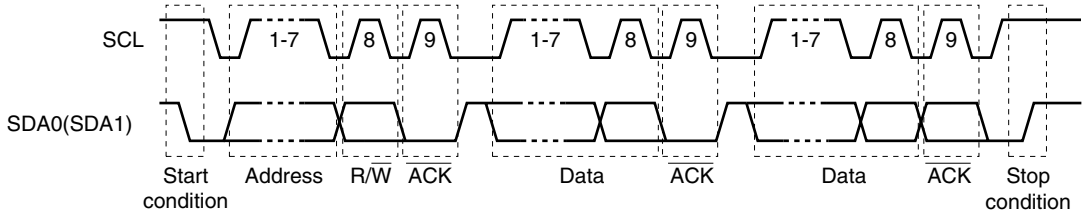
When a slave device is preparing for data transmission or reception and requires more waiting time, the slave device outputs a wait signal on the bus to inform the master device of the wait status.

(2) I^2C bus definition

This section describes the format of serial data communication and the functions of the signals used in the I^2C bus mode.

First, the transfer timing of the start condition, data, and stop condition signals, which are output onto the signal data bus of the I^2C bus, is shown in Figure 13-36.

Figure 13-36. I^2C Bus Serial Data Transfer Timing



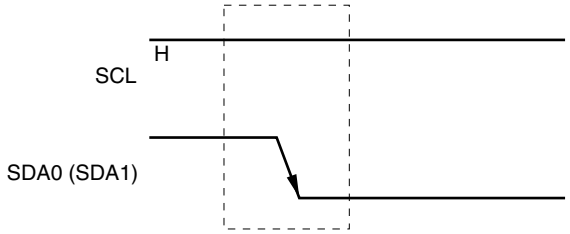
The start condition, slave address, and stop condition signals are output by the master. The acknowledge signal ($\overline{ACK}$) is output by either the master or the slave device (normally by the device which has received the 8-bit data that was sent). A serial clock (SCL) is continuously supplied from the master device.

(a) Start condition

When the SDA0 (SDA1) pin level is changed from high to low while the SCL pin is high, this transition is recognized as the start condition signal. This start condition signal, which is created using the SCL and SDA0 (or SDA1) pins, is output from the master device to slave devices to initiate a serial transfer. Refer to **13.4.6 Cautions on use of I²C bus mode**, for details of the start condition output.

The start condition signal is detected by hardware incorporated in slave devices.

Figure 13-37. Start Condition

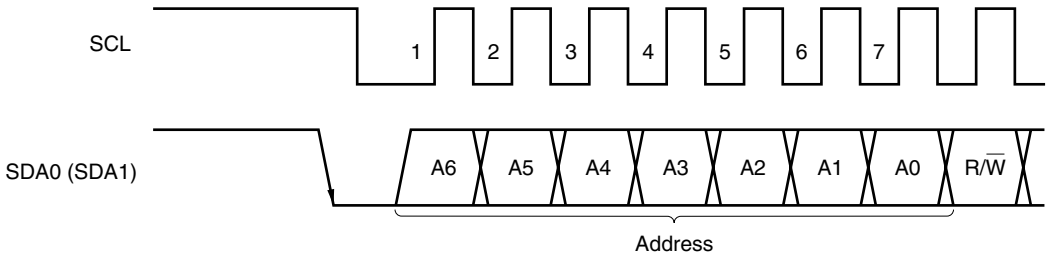


(b) Address

The 7 bits following the start condition signal are defined as an address.

The 7-bit address data is output by the master device to specify a specific slave from among those connected to the bus line. Each slave device on the bus line must therefore have a different address. Therefore, after a slave device detects the start condition, it compares the 7-bit address data received and the data of slave address register 0 (SVA0). After the comparison, only the slave device in which the data matches becomes the communication partner, and subsequently performs communication with the master device until the master device sends a start condition or stop condition signal.

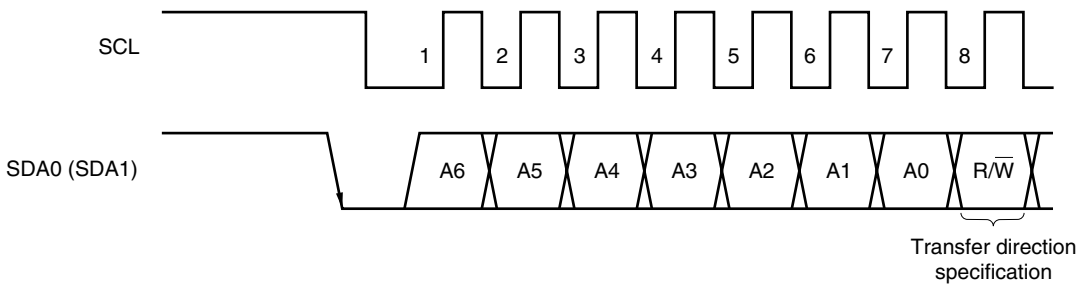
Figure 13-38. Address



(c) Transfer direction specification

The 1 bit that follows the 7-bit address data sent from the master device, is defined as the transfer direction specification bit. If this bit is 0, it is the master device which will send data to the slave. If it is 1, it is the slave device which will send data to the master.

Figure 13-39. Transfer Direction Specification

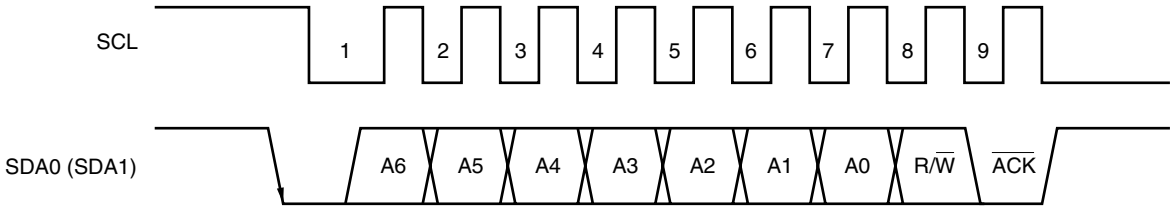


(d) Acknowledge signal ($\overline{ACK}$)

The acknowledge signal indicates that the transferred serial data has definitely been received. This signal is used between the transmitting side and receiving side devices for confirmation of correct data transfer. In principle, the receiving side device returns an acknowledge signal to the transmitting device each time it receives 8-bit data. The only exception is when the receiving side is the master device and the 8-bit data is the last transfer data; the master device outputs no acknowledge signal in this case.

The transmitting side that has transferred 8-bit data waits for the acknowledge signal which will be sent from the receiving side. If the transmitting side device receives the acknowledge signal, which means a successful data transfer, it proceeds to the next processing. If this signal is not sent back from the slave device, this means that the data sent has not been received by the slave device, and therefore the master device outputs a stop condition signal to terminate subsequent transmissions.

Figure 13-40. Acknowledge Signal

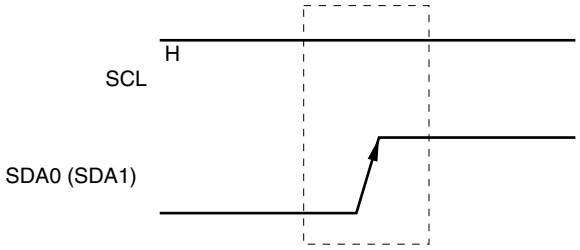


(e) Stop condition

If the SDA0 (SDA1) pin level changes from low to high while the SCL pin is high, this transition is defined as a stop condition signal.

The stop condition signal is output from the master to the slave device to terminate a serial transfer. The stop condition signal is detected by hardware incorporated in the slave device.

Figure 13-41. Stop Condition



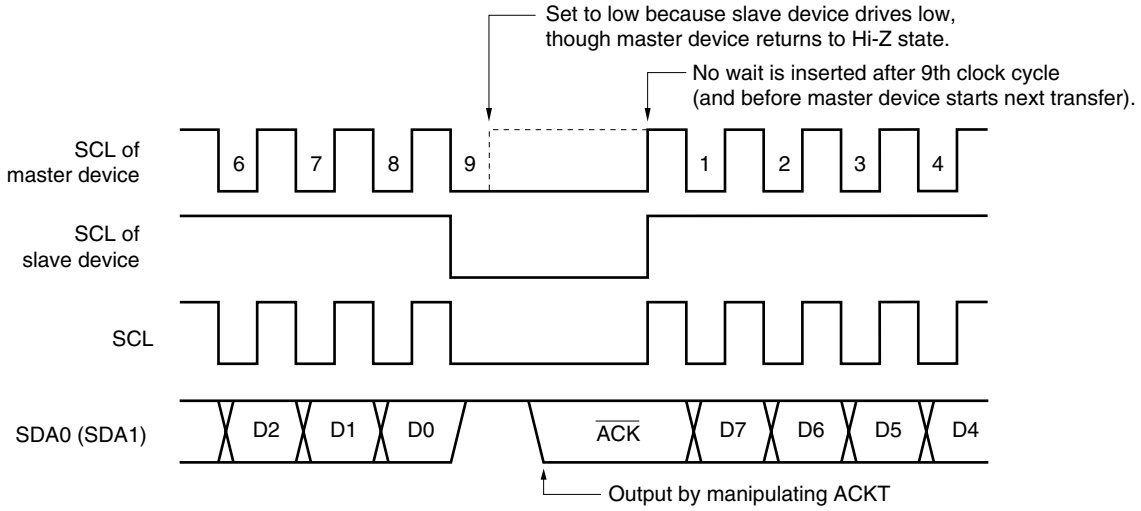
(f) Wait signal ($\overline{\text{WAIT}}$)

The wait signal is output by a slave device to inform the master device that the slave device is in wait state due to preparing for transmitting or receiving data.

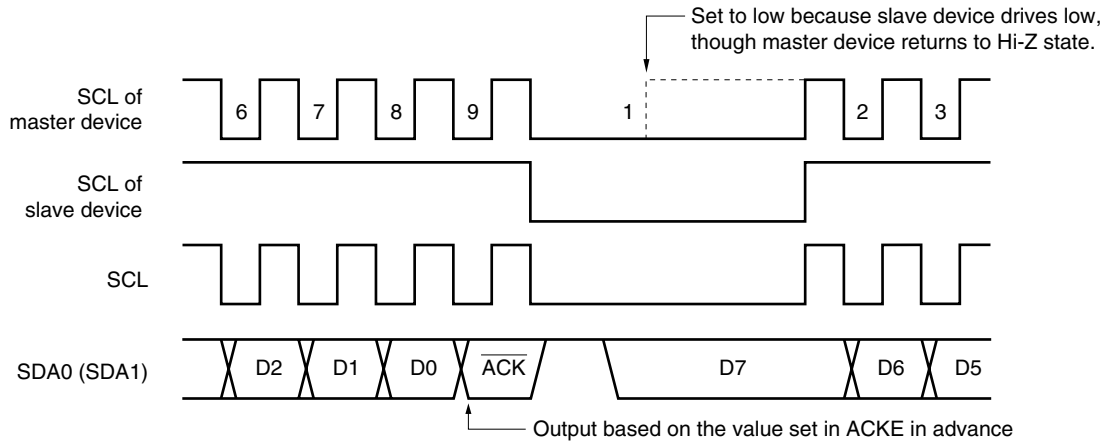
During the wait state, the slave device continues to output the wait signal by keeping the SCL pin low to delay subsequent transfers. When the wait state is released, the master device can start the next transfer. For the releasing operation of slave devices, refer to **13.4.6 Cautions on use of I^2C bus mode**.

Figure 13-42. Wait Signal

(a) Wait of 8 clock cycles



(b) Wait of 9 clock cycles



(3) Register setting

The I^2C mode is set by serial operating mode register 0 (CSIM0), serial bus interface control register 0 (SBIC0), interrupt timing specification register 0 (SINT0), port mode register 2 (PM2), and port 2 (P2).

(a) Serial operating mode register 0 (CSIM0)

CSIM0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM0 to 00H.

Symbol	<7>	<6>	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM0	CSIE0	COI	WUP	CSIM04	CSIM03	CSIM02	CSIM01	0	FF60H	00H	R/W[Note 1]

R/W	CSIM01	Selection of serial interface SIO0 clock
	0	Clock input from off-chip to $\overline{\text{SCK0}}$/SCL/P27 pin
	1	Clock specified by bits 0 to 3 of serial interface clock select register 0 (SCL0)

R/W	CSIM 04	CSIM 03	CSIM 02	PM25	P25	PM26	P26	PM27	P27	Operating mode	Start bit	SI0/SB0/SDA0/ P25 pin function	SO0/SB1/SDA1/ P26 pin function	$\overline{\text{SCK0}}$/SCL/P27 pin function
	0	×	3-wire serial I/O mode (refer to **13.4.2 3-wire serial I/O mode operation**)											
	1	0	SBI mode (refer to **13.4.3 SBI mode operation**)											
	1	1	0	× Note 2	× Note 2	0	0	0	1	2-wire serial I/O (refer to **13.4.4**) or I^2C bus mode	MSB	P25 (CMOS I/O)	SB1/SDA1 (N-ch open-drain I/O)	$\overline{\text{SCK0}}$/SCL (N-ch open-drain I/O)
			1	0	0	× Note 2	× Note 2	0	1			SB0/SDA0 (N-ch open-drain I/O)	P26 (CMOS I/O)	

R/W	WUP	Control of wakeup function[Note 3]
	0	Interrupt request signal generated with each serial transfer in any mode
	1	In I^2C bus mode, interrupt request signal is generated when the address data received after start condition detection (when CMDD = 1) matches the data in slave address register 0.

R	COI	Slave address comparison result flag[Note 4]
	0	Slave address register 0 and serial I/O shift register 0 data do not match
	1	Slave address register 0 and serial I/O shift register 0 data match

R/W	CSIE0	Control of serial interface SIO0 operation
	0	Operation stopped
	1	Operation enabled

Caution When using SCL, set P27 to 1. If P27 is set to 0, it always outputs a low level.

Notes 1. Bit 6 (COI) is a read-only bit.

2. Can be used freely as a port.

3. When using the wakeup function in the I^2C bus mode (WUP = 1), set bit 5 (SIC) of interrupt timing specification register 0 (SINT0) to 1. Do not execute a write instruction to serial I/O shift register 0 (SIO0) while WUP = 1.

4. When CSIE0 = 0, COI is 0.

Remark ×: Don't care
PM××: Port mode register
P××: Output latch of port

(b) Serial bus interface control register 0 (SBIC0)

SBIC0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SBIC0 to 00H.

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SBIC0	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	FF61H	00H	R/W Note

R/W		
R/W	RELT	Use for stop condition signal output. When RELT = 1, SO latch is set to 1. After SO latch setting, automatically cleared to 0. Also cleared to 0 when CSIE0 = 0.

R/W		
R/W	CMDT	Use for start condition signal output. When CMDT = 1, SO latch is cleared to 0. After clearing SO latch, automatically cleared to 0. Also cleared to 0 when CSIE0 = 0.

R	RELD	Detection of stop condition
	Clear conditions (RELD = 0)	Setting conditions (RELD = 1)
	• When transfer start instruction is executed • If SIO0 and SVA0 values do not match in address reception • When CSIE0 = 0 • When reset input is applied	• When stop condition is detected

R	CMDD	Detects Start condition
	Clear conditions (CMDD = 0)	Setting conditions (CMDD = 1)
	• When transfer start instruction is executed • When stop condition is detected • When CSIE0 = 0 • When reset input is applied	• When start condition is detected

R/W		
R/W	ACKT	SDA0 (SDA1) is set to low level after the instruction to be set to 1 is executed (ACKT = 1) before the next SCL falling edge. Used for generating an $\overline{ACK}$ signal by software if the 8-clock wait mode is selected. Cleared to 0 if CSIE = 0 when a transfer by the serial interface is started.

(Continued)

Note Bits 2, 3, and 6 (RELD, CMDD, ACKD) are read-only bits.

R/W

ACKE	Control of acknowledge signal automatic output[Note 1]
0	Disabled (with ACKT enabled). Used when receiving data in the 8-clock wait mode or when transmitting data.[Note 2]
1	Enabled. After completion of transfer, the acknowledge signal is output in synchronization with the 9th falling edge of the SCL clock (automatically output when ACKE = 1). However, not automatically cleared to 0 after acknowledge signal output. Used for reception when the 9-clock wait mode is selected.

R

ACKD	Detection of acknowledge
Clear conditions (ACKD = 0)	Set conditions (ACKD = 1)
• When transfer start instruction is executed • When CSIE0 = 0 • When reset input is applied	• When acknowledge signal is detected at rising edge of SCL clock after completion of transfer

R/W

BSYE[Note 3]	Control of N-ch open-drain output for transmission in I^2C bus mode[Note 4]
0	Output enabled (transmission)
1	Output disabled (reception)

Notes 1. This setting must be performed prior to transfer start.

2. In the 8-clock wait mode, use ACKT for output of the acknowledge signal after normal data reception.

3. The busy mode can be released by the start of a serial interface transfer or reception of an address signal. However, the BSYE flag is not cleared.

4. When using the wakeup function, be sure to set BSYE to 1.

(c) Interrupt timing specification register 0 (SINT0)

SINT0 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets SINT0 to 00H.

Symbol	7	<6>	<5>	<4>	<3>	<2>	1	0	Address	After reset	R/W
SINT0	0	CLD	SIC	SVAM	CLC	WREL	WAT1	WAT0	FF63H	00H	R/W Note 1

R/W	WAT1	WAT0	Control of wait and interrupt
	0	0	Setting prohibited Note 2
	0	1	
	1	0	Used in I^2C bus mode (8-clock wait) An interrupt servicing request is generated on the rise of the 8th SCL clock cycle. (In the case of a master device, the SCL pin is driven low after output of 8 clock cycles, to enter the wait state. In the case of a slave device, the SCL pin is driven low after input of 8 clock cycles, to request the wait state.)
	1	1	Used in I^2C bus mode (9-clock wait) An interrupt servicing request is generated on the rise of the 9th SCL clock cycle. (In the case of a master device, the SCL pin is driven low after output of 9 clock cycles, to enter the wait state. In the case of a slave device, the SCL pin is driven low after input of 9 clock cycles, to request the wait state.)

R/W	WREL	Control of wait release
	0	Indicates that the wait state has been released.
	1	Release the wait state. Automatically cleared to 0 after releasing the wait state. This bit is used to release the wait state set by means of WAT0 and WAT1.

R/W	CLC	Control of clock level
	0	Used in I^2C bus mode. In cases other than serial transfer, SCL pin output is driven low.
	1	Used in I^2C bus mode. In cases other than serial transfer, SCL pin output is set to high impedance. (The clock line is held high.) Used by the master device to generate the start condition and stop condition signals.

R/W	SVAM	SVA0 bits used as slave address
	0	Bits 0 to 7
	1	Bits 7 to 0

R/W	SIC	Selection of INTCSI0 interrupt source Note 3
	0	CSIIF0 is set to 1 after end of serial interface SIO0 transfer.
	1	CSIIF0 is set to 1 after end of serial interface SIO0 transfer or when stop condition is detected.

R	CLD	$\overline{SCK0}$/SCL/P27 pin level Note 4
	0	Low level
	1	High level

Notes 1. Bit 6 (CLD) is read-only.

2. When the I^2C bus mode is used, be sure to set WAT0 and WAT1 to 1 and 0, or 1 and 1, respectively.

3. When using the wakeup function in I^2C mode, be sure to set SIC to 1.

4. When CSIE0 = 0, CLD is 0.

Remark SVA0: Slave address register 0

(4) Signals

A list of signals in the I^2C bus mode is given in Table 13-4.

Table 13-4. Signals in I^2C Bus Mode

Signal Name	Description	
Start condition	Definition:	SDA0 (SDA1) falling edge when SCL is high Note 1
	Function:	Indicates that serial communication will start and subsequent data is address data.
	Signaled by:	Master
	Signaled when:	CMDT is set.
	Affected flag(s):	CMDD (is set).
Stop condition	Definition:	SDA0 (SDA1) rising edge when SCL is high Note 1
	Function:	Indicates end of serial transmission.
	Signaled by:	Master
	Signaled when:	RELT is set.
	Affected flag(s):	RELD (is set) and CMDD (is cleared).
Acknowledge signal (ACK)	Definition:	Low level of SDA0 (SDA1) pin during one SCL clock cycle after serial reception
	Function:	Indicates completion of reception of 1 byte.
	Signaled by:	Master or slave
	Signaled when:	ACKT is set with ACKE = 1.
	Affected flag(s):	ACKD (is set).
Wait ($\overline{\text{WAIT}}$)	Definition:	Low-level signal output to SCL
	Function:	Indicates state in which serial reception is not possible.
	Signaled by:	Slave
	Signaled when:	WAT1, WAT0 = 1x.
	Affected flag(s):	None
Serial clock (SCL)	Definition:	Synchronization clock for output of various signals
	Function:	Serial communication synchronization signal.
	Signaled by:	Master
	Signaled when:	**Note 2**
	Affected flag(s):	CSIIF0 Note 3
Address (A6 to A0)	Definition:	7-bit data synchronized with SCL immediately after start condition signal
	Function:	Indicates address value for specification of slave on serial bus.
	Signaled by:	Master
	Signaled when:	**Note 2**
	Affected flag(s):	CSIIF0 Note 3
Transfer direction (R/$\overline{\text{W}}$)	Definition:	1-bit data output in synchronization with SCL after address output
	Function:	Indicates whether data transmission or reception is to be performed.
	Signaled by:	Master
	Signaled when:	**Note 2**
	Affected flag(s):	CSIIF0 Note 3
Data (D7 to D0)	Definition:	8-bit data synchronized with SCL, not immediately after start condition
	Function:	Contains data to be actually sent.
	Signaled by:	Master or slave
	Signaled when:	**Note 2**
	Affected flag(s):	CSIIF0 Note 3

Notes 1. The level of the serial clock can be controlled by CLC of interrupt timing specification register 0 (SINT0).

2. An instruction to write data to SIO0 is executed when CSIE0 = 1 (serial transfer start directive). In the wait state, the serial transfer operation will be started after the wait state is released.

3. If the 8-clock wait is selected when WUP = 0, CSIIF0 is set at the rising edge of the 8th clock cycle of SCL. If the 9-clock wait is selected when WUP = 0, CSIIF0 is set at the rising edge of the 9th clock cycle of SCL. If WUP = 1, CSIIF0 is set when an address is received and the address matches the slave address register 0 (SVA0) value, and when a stop condition is detected.

(5) Pin configuration

The configurations of the serial clock pin (SCL) and the serial data bus pin SDA0 (SDA1) are shown below.

(a) SCL

Serial clock I/O alternate-function pin.

<1> Master N-ch open-drain output

<2> Slave Schmitt input

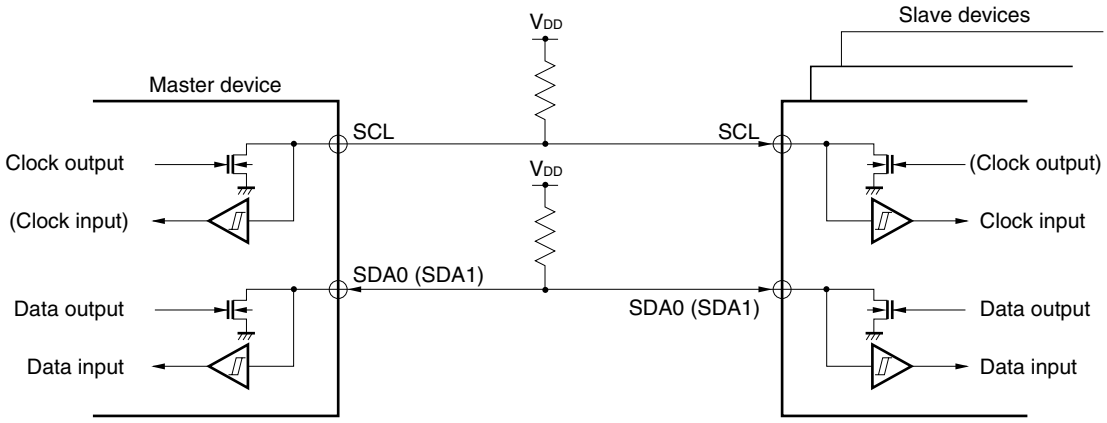
(b) SDA0 (SDA1)

Serial data I/O alternate-function pin.

Uses N-ch open-drain output and Schmitt-input buffers for both master and slave devices.

Note that pull-up resistors are required to connect to both serial clock line and serial data bus line, because open-drain buffers are used for the serial clock pin (SCL) and the serial data bus pin (SDA0 or SDA1) on the I^2C bus.

Figure 13-43. Pin Configuration



Caution Because it is necessary to make an N-ch open-drain output high impedance while data is being received, set bit 6 (BSYE) of serial bus interface control register 0 (SBIC0) to 1 in advance and write FFH to serial I/O shift register 0 (SIO0).

When the wakeup function is used (when bit 5 (WUP) of serial the serial operating mode register 0 (CSIM0) is set), do not write FFH to SIO0 before reception. The N-ch open-drain output is always high impedance even if FFH is not written to SIO0.

(6) Address match detection method

In the I^2C mode, the master can select a specific slave device by sending slave address data.

Address match detection is performed automatically by the slave device hardware. A slave device address is compared with the slave address sent from the master device. If they match and the wakeup state (WUP) bit is then 1, the interrupt request flag (CSIIF0) is set (it is also set when a stop condition is detected). When using the wakeup function (WUP = 1), set SIC to 1.

Caution Whether a slave is selected or not depends on detection of match of the data (address) received after the start condition.

To detect this match, an address match detection interrupt (INTCSI0) that occurs when WUP = 1, is normally used. Therefore, to enable detection of whether a slave is selected or not, be sure that WUP = 1.

(7) Error detection

In the I^2C bus mode, transmit error detection can be performed by the following methods because the serial bus SDA0 (SDA1) status during transmission is also taken into serial I/O shift register 0 (SIO0) of the transmitting device.

(a) Comparison of SIO0 data before and after transmission

In this case, a transmit error is judged to have occurred if the two data values are different.

(b) Using the slave address register 0 (SVA0)

Transmit data is set in SIO0 and SVA0 before transmission is performed. After transmission, the COI bit (match signal from the address comparator) of serial operating mode register 0 (CSIM0) is tested: "1" indicates normal transmission, and "0" indicates a transmit error.

(8) Communication operation

In the I^2C bus mode, the master selects the slave device communicate with from among multiple devices by outputting address data onto the serial bus.

After the slave address data, the master sends the R/W bit, which indicates the data transfer direction, and starts serial communication with the selected slave device.

Data communication timing charts are shown in Figures 13-44 and 13-45.

In the transmitting device, serial I/O shift register 0 (SIO0) shifts transmit data to the SO latch in synchronization with the falling edge of the serial clock (SCL), the SO0 latch outputs the data on an MSB-first basis from the SDA0 or SDA1 pin to the receiving device.

In the receiving device, the data input from the SDA0 or SDA1 pin is taken into serial I/O shift register 0 (SIO0) in synchronization with the rising edge of SCL.

Figure 13-44. Data Transmission from Master to Slave (Both Master and Slave Selected 9-Clock Wait) (1/3)

(a) Start condition to address

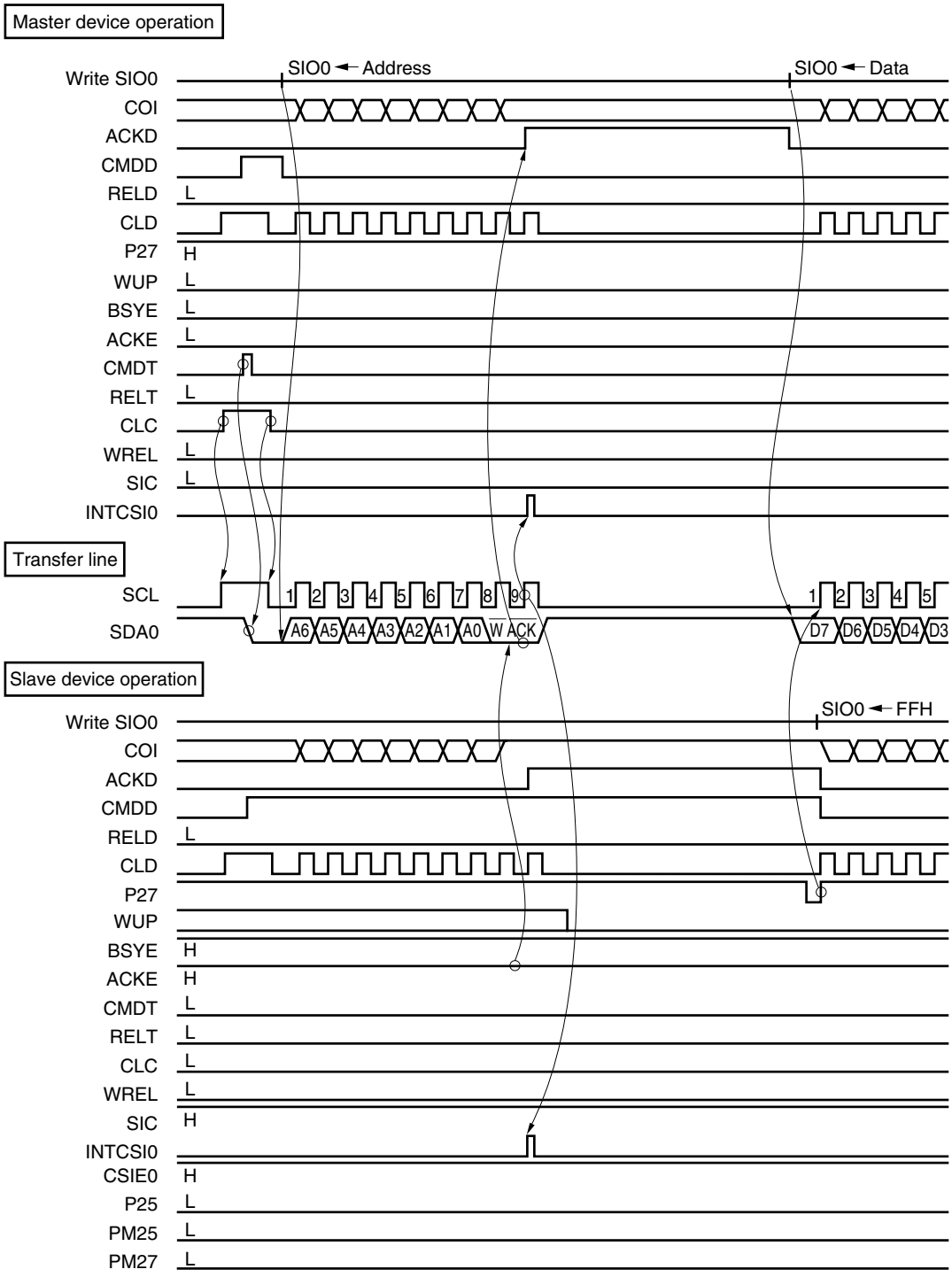


Figure 13-44. Data Transmission from Master to Slave (Both Master and Slave Selected 9-Clock Wait) (2/3)

(b) Data

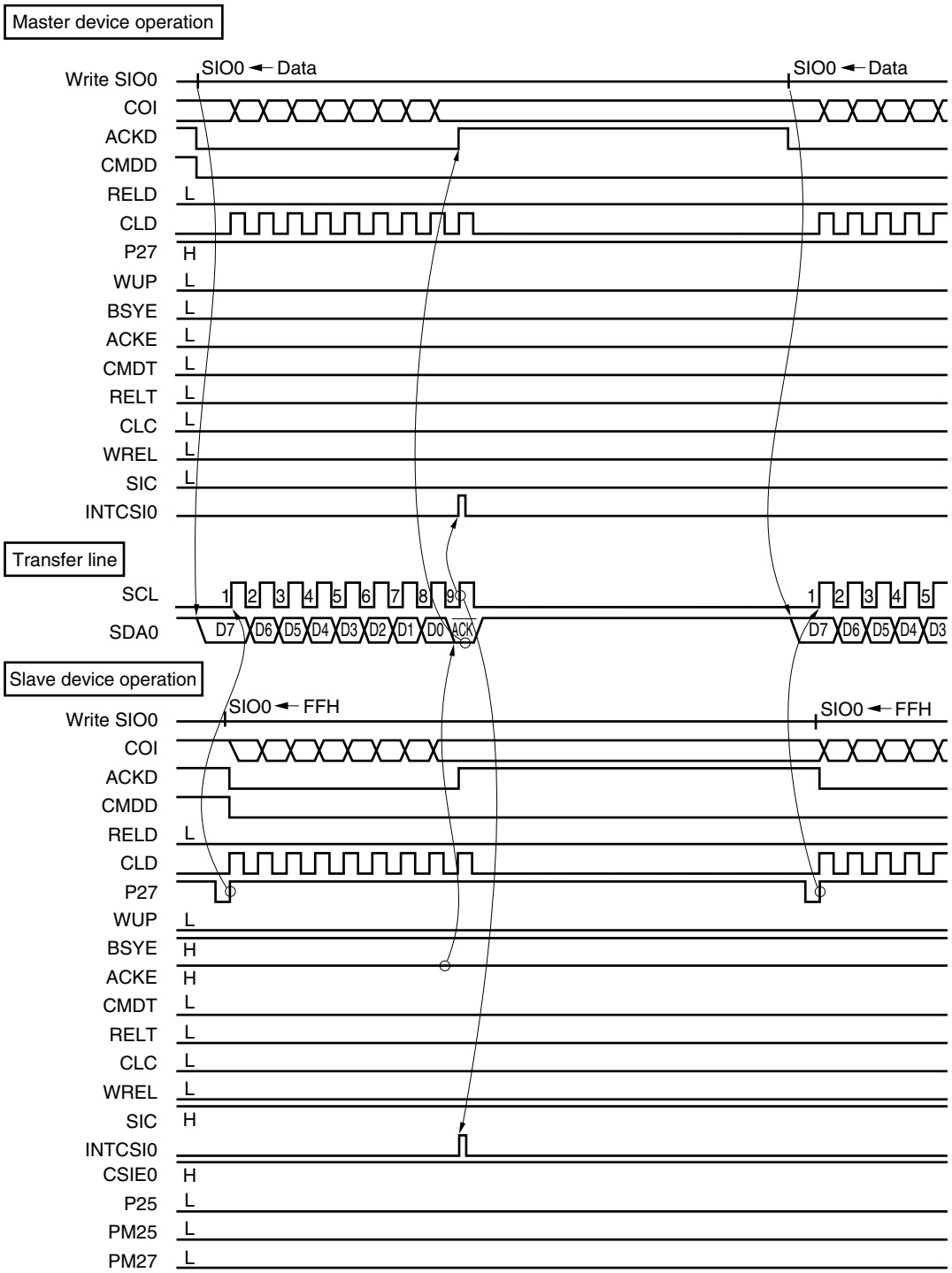


Figure 13-44. Data Transmission from Master to Slave (Both Master and Slave Selected 9-Clock Wait) (3/3)

(c) Stop condition

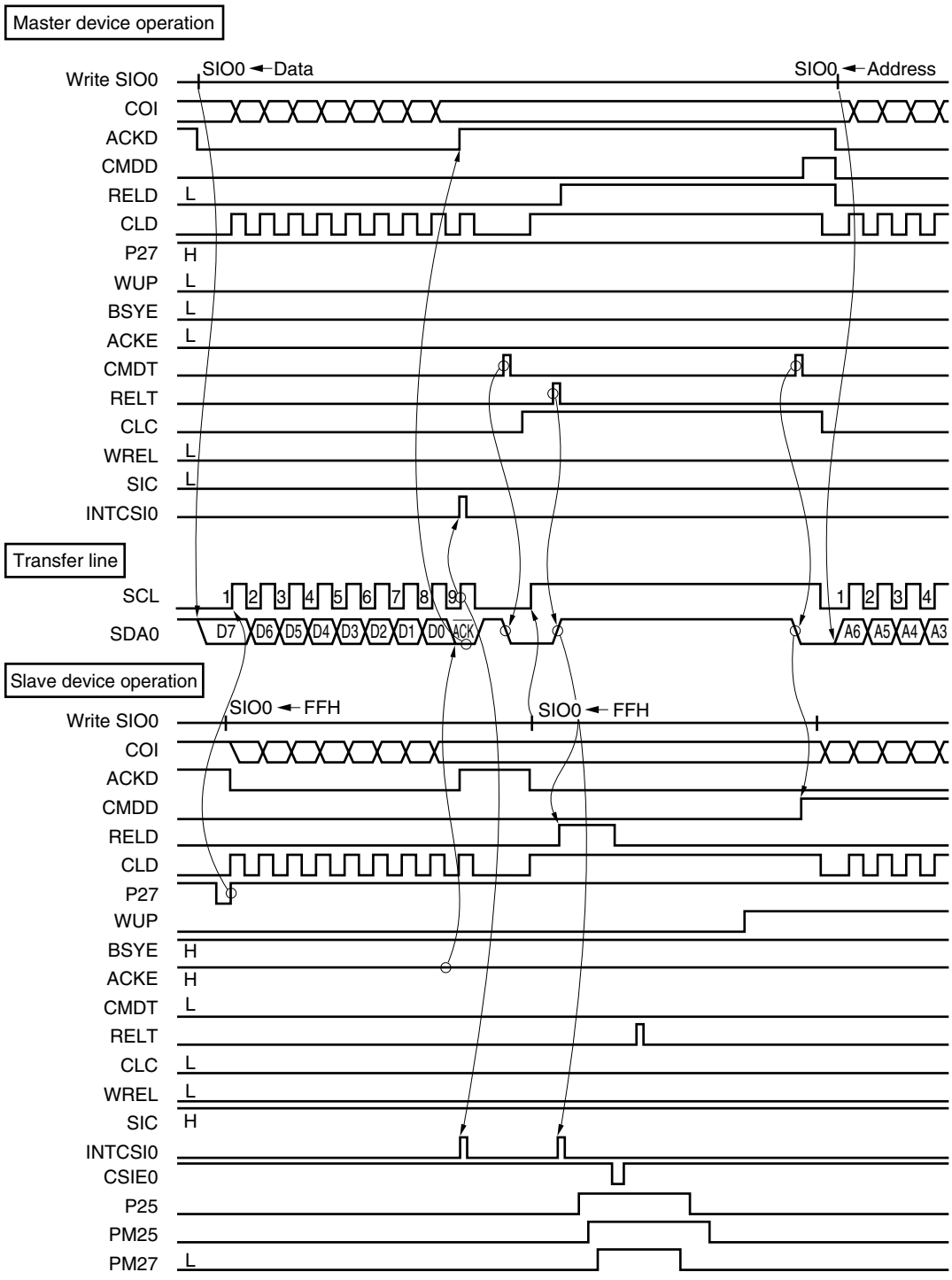


Figure 13-45. Data Transmission from Slave to Master (Both Master and Slave Selected 9-Clock Wait) (1/3)

(a) Start condition to address

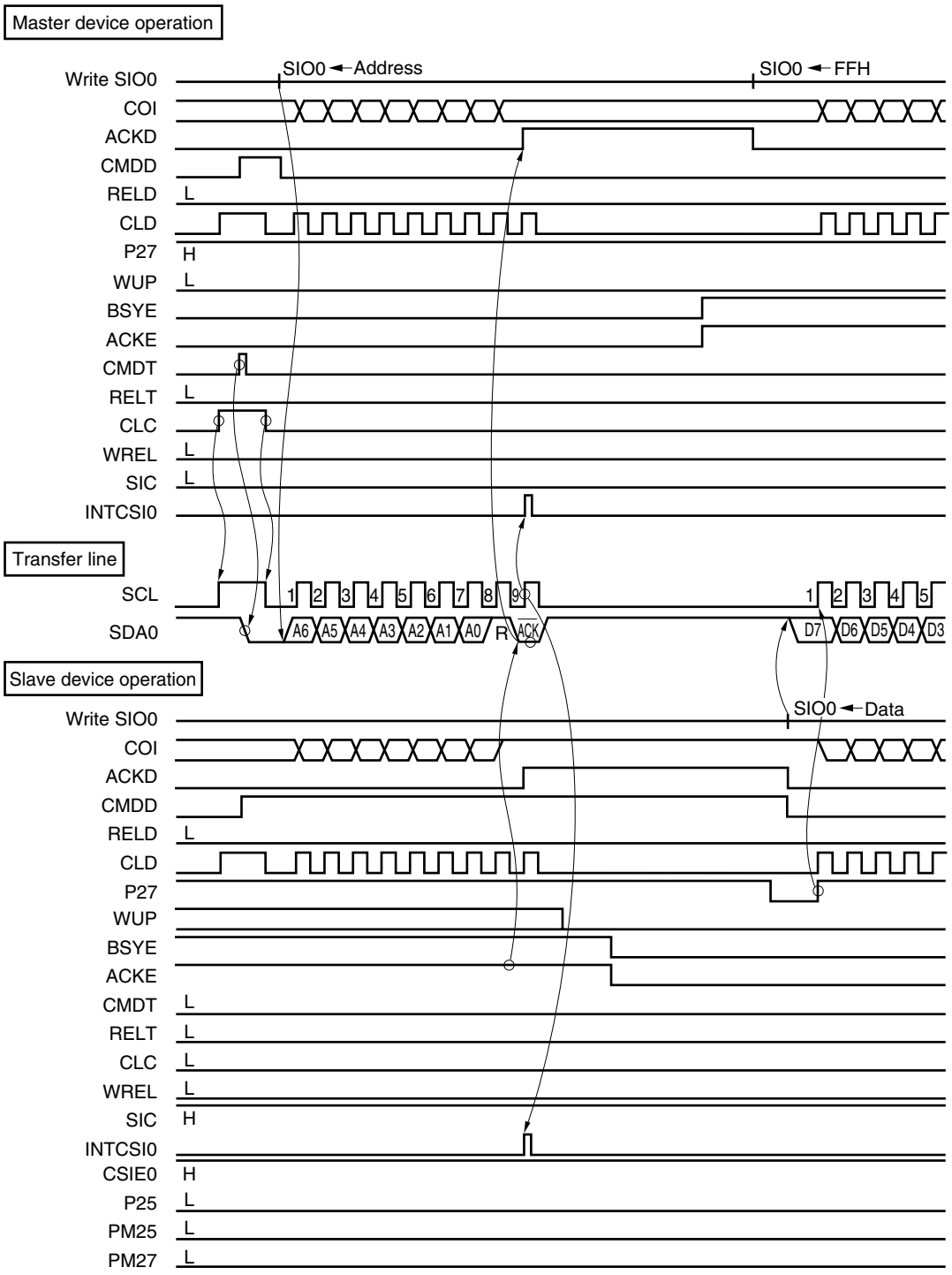


Figure 13-45. Data Transmission from Slave to Master (Both Master and Slave Selected 9-Clock Wait) (2/3)

(b) Data

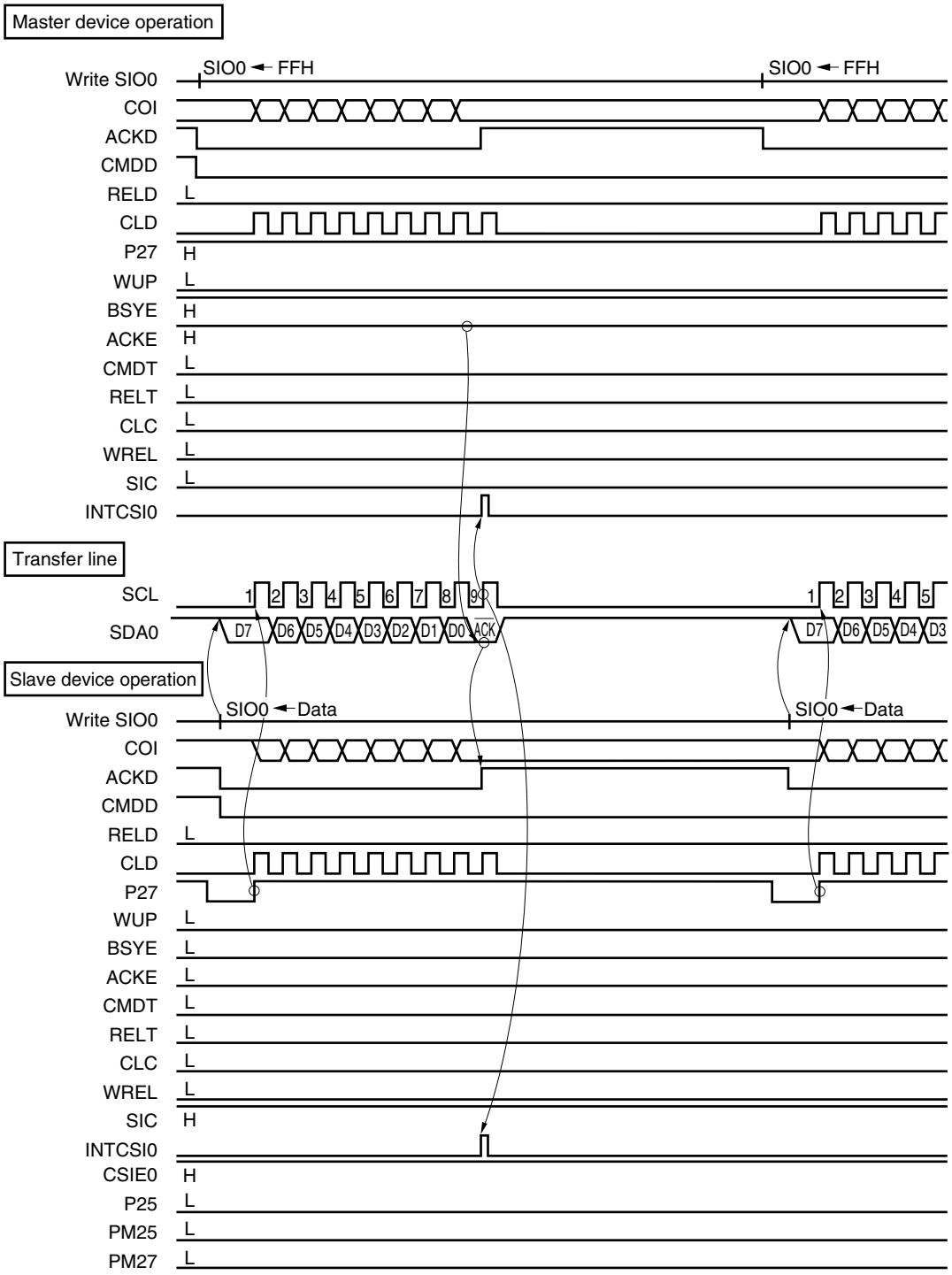
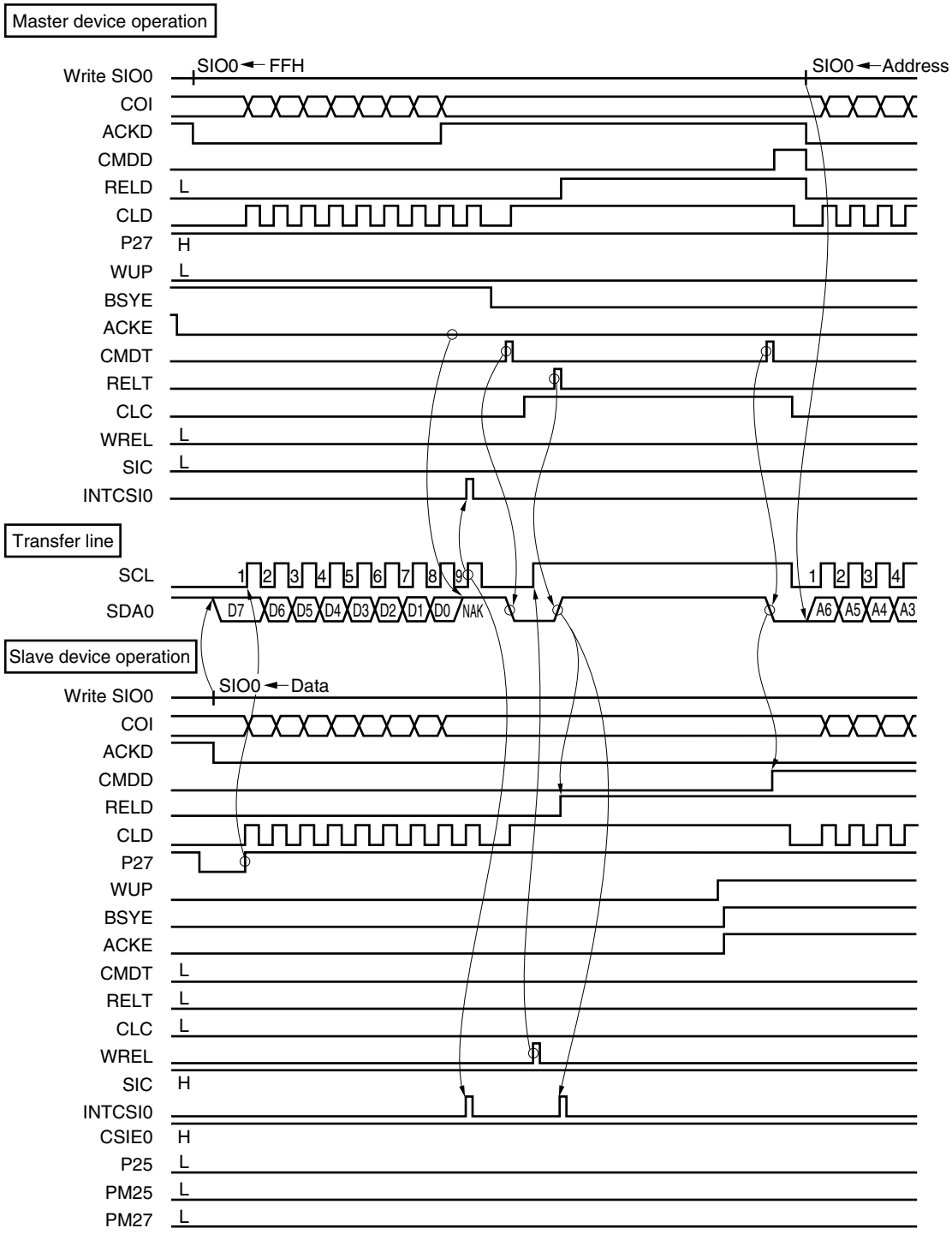


Figure 13-45. Data Transmission from Slave to Master (Both Master and Slave Selected 9-Clock Wait) (3/3)

(c) Stop condition



(9) Start of transfer

A serial transfer is started by setting transfer data in serial I/O shift register 0 (SIO0) if the following two conditions are satisfied.

- The serial interface SIO0 operation control bit (CSIE0) = 1.
- After an 8-bit serial transfer, the internal serial clock is stopped or SCL is low.

Cautions 1. Be sure to set CSIE0 to 1 before writing data to SIO0. Setting CSIE0 to 1 after writing data to SIO0 will not start a transfer operation.

2. Because the N-ch open-drain output must be made high impedance during data reception, set bit 7 (BSYE) of serial bus interface control register 0 (SBIC0) to 1 before writing FFH to SIO0.

When the wakeup function is used (when bit 5 (WUP) of serial operating mode register 0 (CSIM0) is set), do not write FFH to SIO0 before reception. The N-ch open-drain output is always high impedance even if FFH is not written to SIO0 .

3. If data is written to SIO0 while the slave is in the wait state, that data is held. The transfer is started when SCL is output after the wait state is cleared.

When an 8-bit data transfer ends, serial transfer is stopped automatically and the interrupt request flag (CSIIF0) is set.

13.4.6 Cautions on use of I^2C bus mode

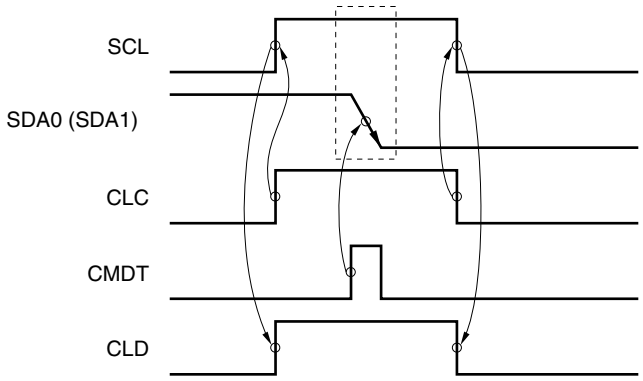
(1) Start condition output (master)

The SCL pin normally outputs a low-level signal when no serial clock is output. It is necessary to change the SCL pin to high in order to output a start condition signal. Set CLC of interrupt timing specification register 0 (SINT0) to 1 to drive the SCL pin high.

After setting CLC, clear CLC to 0 and return the SCL pin to low. If CLC remains 1, no serial clock is output.

If it is the master device which outputs the start condition and stop condition signals, confirm that CLD is set to 1 after setting CLC to 1; a slave device may have set SCL to low (wait state).

Figure 13-46. Start Condition Output



(2) Slave wait release

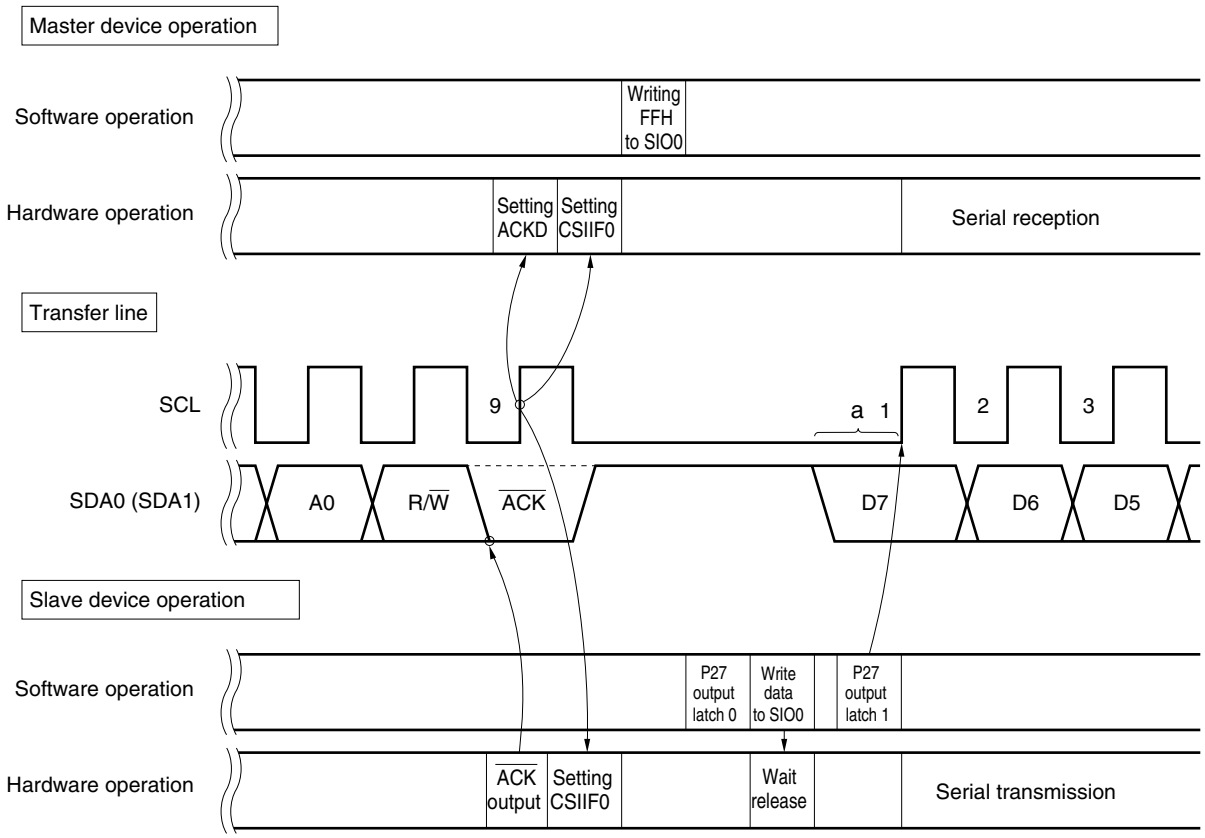
The slave wait release operation is performed by setting the WREL flag or executing an SIO0 write instruction. If the slave sends data, the wait is immediately released by execution of an SIO0 write instruction and the clock rises without the start transmission bit being output on the data line. Therefore, as shown in Figure 13-47, data should be transmitted by manipulating the P27 output latch via the program. At this time, control the low-level width ("a" in Figure 13-47) of the first serial clock at the timing used for setting the P27 output latch to 1 after execution of an SIO0 write instruction.

In addition, if the acknowledge signal from the master is not output (if data transmission from the slave is completed), set the WREL flag of SINT0 to 1 and release the wait.

If the slave receives data, after execution of an SIO0 write instruction, it is not necessary to manipulate the P27 output latch because the data to be received has already been output on the data line even if the wait is released.

For the timing of these operations, see Figures 13-44 and 13-45.

Figure 13-47. Slave Wait Release (Transmission)



(3) Reception completion of slave

During processing of reception completion by a slave device, confirm the statuses of CMDD and COI (if CMDD = 1). This procedure is necessary to use the wakeup function normally. If an uncertain amount of data is sent from the master device, the slave device cannot determine whether the start condition signal or the data will be sent from the master. This may disable use of the wakeup function.

13.4.7 Restrictions in using I^2C bus mode

The following restrictions must be observed when using the μPD178078, 178098A Subseries.

- **Restrictions when using as slave device in I^2C bus mode**

Devices: μPD178076, 178078, 178096A, 178098A, 178F098
IE-178098-NS-EM1

Description: If the wakeup function is executed (by setting the WUP flag (bit 5 of serial operating mode register 0 (CSIM0)) to 1) in the serial transfer status**Note**, data between another slave device and the master is identified as an address. If that data matches the slave address of the μPD178078, 178098A Subseries, therefore, the μPD178078, 178098A Subseries participates in communication, destroying the communicated data.

Note The serial transfer status is the status after serial I/O shift register 0 (SIO0) has been written until the interrupt request flag (CSIIF0) is set to 1 due to the end of serial transfer.

Preventive measures: The above problem can be avoided by modifying the program.
Before executing the wakeup function, execute the program shown below that clears the serial transfer status. When executing the wakeup function, do not write an instruction that writes to SIO0. Data can be received during execution of the wakeup function even if such an instruction is not executed.
This program clears the serial transfer status. To clear the serial transfer status, serial interface SIO0 must be stopped once (by clearing the CSIE0 flag (bit 7 of serial operating mode register 0 (CSIM0)) to 0). If serial interface SIO0 is stopped in the I^2C bus mode, however, the SCL pin outputs a high level and SDA0 (SDA1) pin outputs a low level. Consequently, communication of the I^2C bus may be affected. Therefore, this program makes the SCL and SDA0 (SDA1) pins go into a high-impedance state to prevent the I^2C bus from being affected.
Note that, in this example, the SDA0 (/P25) pin is used as the serial data I/O pin.
If the SDA1 (/P26) pin is used as the serial data I/O pin, take P2.5 and PM2.5 in the program below as P2.6 and PM2.6.
For the timing of each signal when this program is executed, refer to Figure 13-44.

- **Example of program for clearing serial transfer status**

```
SET1   P2.5      ; (1)
SET1   PM2.5     ; (2)
SET1   PM2.7     ; (3)
CLR1   CSIE0     ; (4)
SET1   CSIE0     ; (5)
SET1   RELT      ; (6)
CLR1   PM2.7     ; (7)
CLR1   P2.5      ; (8)
CLR1   PM2.5     ; (9)
```

(1) Instruction (5) prevents the SDA0 pin from outputting a low level when the I^2C bus mode is restored. The SDA0 pin goes into a high-impedance state.
(2) Instruction (4) sets the P25 (/SDA0) pin in the input mode to prevent the SDA0 line from being affected when the port mode is restored. The input mode is set when instruction (2) is executed.
(3) Instruction (4) sets the P27 (/SCL) pin in the input mode to prevent the SCL line from being affected when the port mode is restored. The input mode is set when instruction (3) is executed.
(4) The mode is changed from the I^2C bus mode to the port mode.
(5) The mode is changed from the port mode to the I^2C bus mode.
(6) Instruction (8) prevents the SDA0 pin from outputting a low level.
(7) Because the P27 pin must be set in the output mode in the I^2C bus mode, the P27 pin is set in the output mode.
(8) Because the output latch of the P25 pin must be set to 0 in the I^2C bus mode, the output latch of the P25 pin is set to 0.
(9) Because the P25 pin must be set in the output mode in the I^2C bus mode, the P25 pin is set in the output mode.

Remark RELT: Bit 0 of serial bus interface control register 0 (SBIC0)

13.4.8 $\overline{\text{SCK0}}$/SCL/P27 pin output manipulation

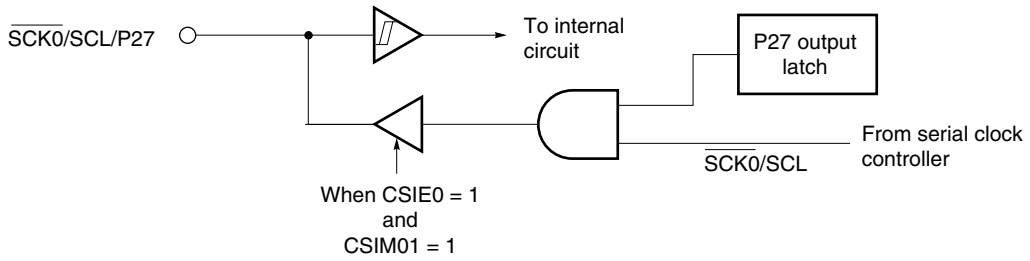
Because the $\overline{\text{SCK0}}$/SCL/P27 pin incorporates an output latch, static output is also possible by software in addition to normal serial clock output.

P27 output latch manipulation enables any value of $\overline{\text{SCK0}}$/SCL to be set by software. (The SI0/SB0/SDA0 and SO0/SB1/SDA1 pins are controlled by the RELT and CMDT bits of SBIC0.)

The procedure for manipulating $\overline{\text{SCK0}}$/SCL/P27 pin output is described below.

(1) Set serial operating mode register 0 (CSIM0) (the $\overline{\text{SCK0}}$/SCL pin is enabled for serial operation in the output mode). $\overline{\text{SCK0}}$ = 1 and SCL = 0 with serial transfer suspended.
(2) Manipulate the P27 output latch with a bit manipulation instruction.

Figure 13-48. $\overline{\text{SCK0}}$/SCL/P27 Pin Configuration



CHAPTER 14 SERIAL INTERFACE SIO1

14.1 Functions of Serial Interface SIO1

Serial interface SIO1 employs the following three modes.

- Operation stop mode
- 3-wire serial I/O mode
- 3-wire serial I/O mode with automatic transmit/receive function

(1) Operation stop mode

This mode is used when serial transfer is not carried out to reduce power consumption.

(2) 3-wire serial I/O mode (MSB-/LSB-first selectable)

This mode is used for 8-bit data transfer using three lines, one each for the serial clock ($\overline{\text{SCK1}}$), serial output (SO1) and serial input (SI1).

The 3-wire serial I/O mode enables simultaneous transmission/reception and so decreases the data transfer processing time.

Since the start bit of 8-bit data to undergo serial transfer is switchable between MSB and LSB, connection is enabled with either start bit device.

The 3-wire serial I/O mode is effective for connection of peripheral I/O units and display controllers which incorporate a conventional clocked serial interface such as the 75XL, 78K and 17K Series.

(3) 3-wire serial I/O mode with automatic transmit/receive function (MSB-/LSB-first selectable)

This mode has an automatic transmit/receive function in addition to the functions in **(2)** above.

The automatic transmit/receive function is used to transmit/receive data with a maximum of 32 bytes. This function enables the hardware to transmit/receive data to/from an OSD (On Screen Display) device and a device with a built-in display controller/driver independently of the CPU, thus alleviating the software load.

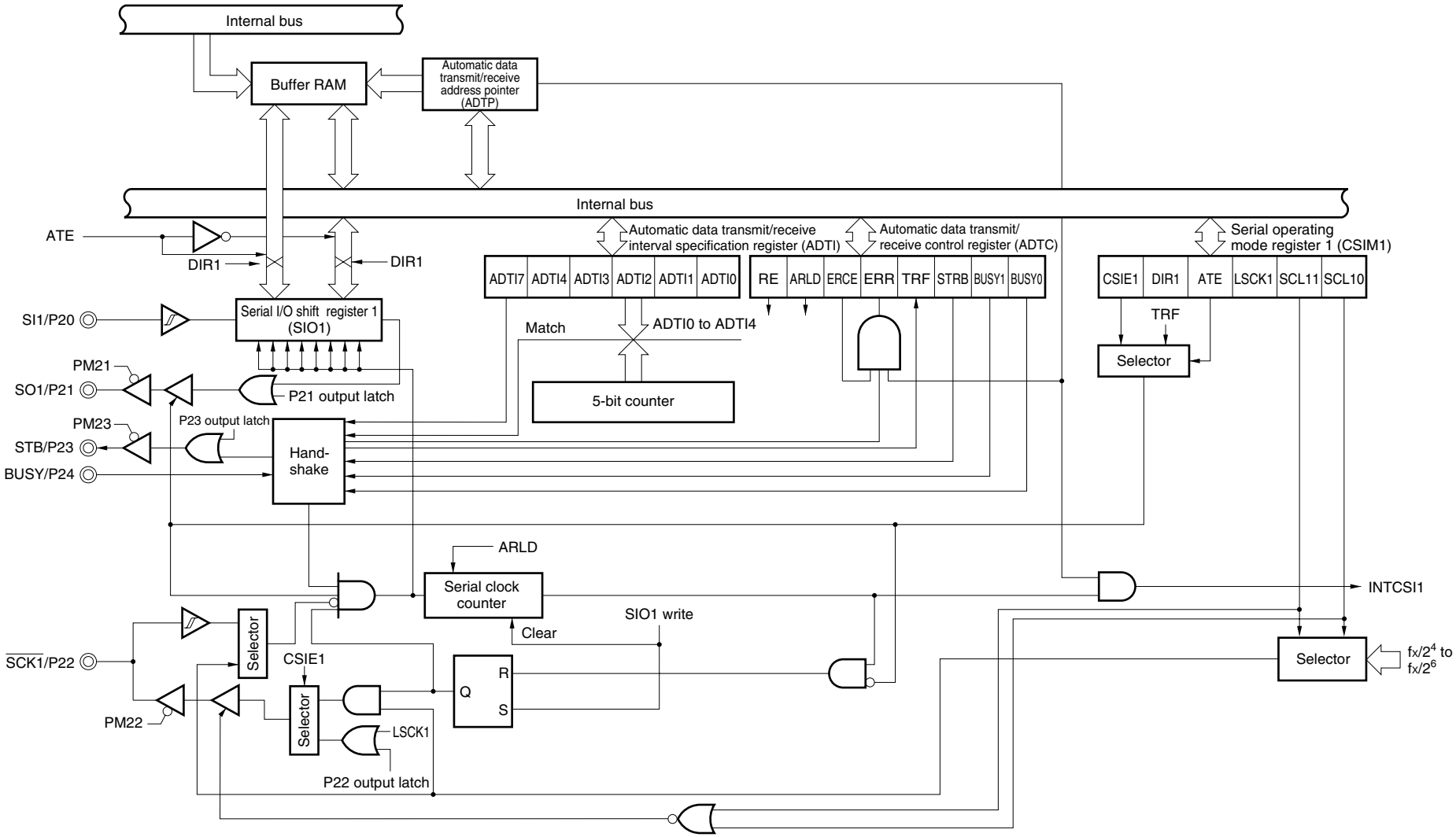
14.2 Configuration of Serial Interface SIO1

Serial interface SIO1 consists of the following hardware.

Table 14-1. Configuration of Serial Interface SIO1

Item	Configuration
Registers	Serial I/O shift register 1 (SIO1) Automatic data transmit/receive address pointer (ADTP)
Control registers	Serial operating mode register 1 (CSIM1) Automatic data transmit/receive control register (ADTC) Automatic data transmit/receive interval specification register (ADTI) Port mode register 2 (PM2) Port 2 (P2)

Figure 14-1. Block Diagram of Serial Interface SIO1



(1) Serial I/O shift register 1 (SIO1)

This is an 8-bit register used to carry out parallel/serial conversion and serial transmission/reception (shift operation) in synchronization with the serial clock.

SIO1 is set by an 8-bit memory manipulation instruction.

When the value in bit 7 (CSIE1) of serial operating mode register 1 (CSIM1) is 1, writing data to SIO1 starts a serial operation.

In transmission, data written to SIO1 is output to the serial output (SO1). In reception, data is read from the serial input (SI1) to SIO1.

Reset input makes SIO1 undefined.

Caution Do not write data to SIO1 while the automatic transmit/receive function is activated.

(2) Automatic data transmit/receive address pointer (ADTP)

This register stores the value of (transmit data byte –1) while the automatic transmit/receive function is activated. As data is transferred/received, it is automatically decremented.

ADTP is set by an 8-bit memory manipulation instruction. The higher 3 bits must be set to 0.

Reset input sets ADTP to 00H.

Caution Do not write data to ADTP while the automatic transmit/receive function is activated.

(3) Serial clock counter

This counter counts the serial clocks to be output and input during transmission/reception to check whether 8-bit data has been transmitted/received.

14.3 Control Registers of Serial Interface SIO1

The following five registers are used to control serial interface SIO1.

- Serial operating mode register 1 (CSIM1)
- Automatic data transmit/receive control register (ADTC)
- Automatic data transmit/receive interval specification register (ADTI)
- Port mode register 2 (PM2)
- Port 2 (P2)

(1) Serial operating mode register 1 (CSIM1)

This register sets the serial interface SIO1 serial clock, operating mode, operation enable/stop and automatic transmit/receive operation enable/stop.

CSIM1 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM1 to 00H.

Figure 14-2. Format of Serial Operating Mode Register 1 (CSIM1)

Symbol	<7>	6	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM1	CSIE1	DIR1	ATE	LSCK1	0	0	SCL11	SCL10	FF68H	00H	R/W

CSIE1	Enable/disable of operation of serial interface SIO1		
	Shift register operation	Serial counter	Port[Note 1]
0	Operation stopped	Cleared	Port function
1	Operation enabled	Count operation enabled	Serial function + port function

DIR1	Specification of first bit of serial transfer data
0	MSB
1	LSB

ATE	Selection of operating mode of serial interface SIO1
0	3-wire serial I/O mode
1	3-wire serial I/O mode with automatic transmit/receive function

LSCK1	Chip enable control of $\overline{SCK1}$ pin
0	$\overline{SCK1}$ is used as port (P22) when CSIE1 = 0. $\overline{SCK1}$ is used for clock output when CSIE1 = 1.
1	$\overline{SCK1}$ is fixed to high level when CSIE1 = 0. $\overline{SCK1}$ is used for clock output when CSIE1 = 1.

SCL11	SCL10	Selection of serial clock of serial interface SIO1
0	0	External clock input to $\overline{SCK1}$ pin[Note 2]
0	1	$f_X/2^4$ (394 kHz)
1	0	$f_X/2^5$ (197 kHz)
1	1	$f_X/2^6$ (98.4 kHz)

Notes 1. When CSIE1 = 0 (SIO1 operation stop status), the P20/SI1, P21/SO1, P22/$\overline{SCK1}$, P23/STB, and P24/BUSY pins can be used as port pins.

2. When external clock input is selected by clearing SCL11 and SCL10 to 0, 0, clear bits 2 (STRB) and 1 (BUSY1) of the automatic data transmit/receive control register (ADTC) to 0, 0.

Remark ( ): f_X = 6.3 MHz

(2) Automatic data transmit/receive control register (ADTC)

This register sets automatic receive enable/disable, the operating mode, strobe output enable/disable, busy input enable/disable and displays automatic transmit/receive execution.

ADTC is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets ADTC to 00H.

Figure 14-3. Format of Automatic Data Transmit/Receive Control Register (ADTC)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
ADTC	RE	ARLD	ERCE	ERR	TRF	STRB	BUSY1	BUSY0	FF6AH	00H	R/W Note 1

R/W

BUSY1	BUSY0	Control of busy input
0	×	Not using busy input Note 2
1	0	Busy input enabled (active high)
1	1	Busy input enabled (active low)

R/W

STRB	Control of strobe output
0	Strobe output disabled Note 3
1	Strobe output enabled

R

TRF	Status of automatic transmit/receive function Note 4
0	End of automatic transmission/reception detected. (This bit is set to 0 upon suspension of automatic transmission/reception or when ARLD = 0.)
1	Automatic transmission/reception in progress. (This bit is set to 1 when data is written to SIO1.)

R

ERR	Error detection of automatic transmit/receive function
0	No error (This bit is set to 0 when data is written to SIO1.)
1	Error occurred

R/W

ERCE	Error check control of automatic transmit/ receive function
0	Error check disabled
1	Error check enabled (only when BUSY1 = 1)

R/W

ARLD	Operating mode selection of automatic transmit/ receive function
0	Single operating mode
1	Repetitive operating mode

R/W

RE	Receive control of automatic transmit/receive function
0	Receive disabled Note 5
1	Receive enabled

Notes 1. Bits 3 and 4 (TRF and ERR) are read-only bits.

2. When BUSY1 is reset to 0, P24 (CMOS I/O) is used even when bit 7 (CSIE1) of serial operating mode register 1 (CSIM1) is set to 1.

3. When STRB is reset to 0, P23 (CMOS I/O) is used even when bit 7 (CSIE1) of CSIM1 is set to 1.

4. When an interrupt is acknowledged, interrupt request flag CSIIF1 is cleared. Therefore, use TRF, instead of CSIIF1, to identify the completion of automatic transmission/reception.

5. When RE is reset to 0, P20 (CMOS I/O) is used even when bit 7 (CSIE1) of CSIM1 is set to 1.

Caution When an external clock input is selected with bits 0 and 1 (SCL11 and SCL10) of CSIM1 set to 0, set bits 2 and 1 (STRB and BUSY1) of ADTC to 0, 0.

Remark ×: Don't care

(3) Automatic data transmit/receive interval specification register (ADTI)

This register sets the automatic data transmit/receive function data transfer interval.

ADTI is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets ADTI to 00H.

Figure 14-4. Format of Automatic Data Transmit/Receive Interval Specification Register (ADTI) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADTI	ADTI7	0	0	ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	FF6BH	00H	R/W

ADTI7	Control of data transfer interval
0	No control of interval by ADTI Note 1
1	Control of interval by ADTI (ADTI0 to ADTI4)

ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	Control of data transfer interval (f_X = 6.3 MHz, f_{SCK} = 394 kHz) Note 2
0	0	0	0	0	5.08 μs + 0.5/f_{SCK}
0	0	0	0	1	
0	0	0	1	0	7.61 μs + 0.5/f_{SCK}
0	0	0	1	1	10.2 μs + 0.5/f_{SCK}
0	0	1	0	0	12.7 μs + 0.5/f_{SCK}
0	0	1	0	1	15.2 μs + 0.5/f_{SCK}
0	0	1	1	0	17.8 μs + 0.5/f_{SCK}
0	0	1	1	1	20.3 μs + 0.5/f_{SCK}
0	1	0	0	0	22.8 μs + 0.5/f_{SCK}
0	1	0	0	1	25.4 μs + 0.5/f_{SCK}
0	1	0	1	0	27.9 μs + 0.5/f_{SCK}
0	1	0	1	1	30.5 μs + 0.5/f_{SCK}
0	1	1	0	0	33.0 μs + 0.5/f_{SCK}
0	1	1	0	1	35.5 μs + 0.5/f_{SCK}
0	1	1	1	0	38.1 μs + 0.5/f_{SCK}
0	1	1	1	1	40.6 μs + 0.5/f_{SCK}

(Continued)

Figure 14-4. Format of Automatic Data Transmit/Receive Interval Specification Register (ADTI) (2/2)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADTI	ADTI7	0	0	ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	FF6BH	00H	R/W

ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	Specification of data transfer interval (f_X = 6.3 MHz, f_{SCK} = 394 kHz)[Note]
1	0	0	0	0	43.1 μs + 0.5/f_{SCK}
1	0	0	0	1	45.7 μs + 0.5/f_{SCK}
1	0	0	1	0	48.2 μs + 0.5/f_{SCK}
1	0	0	1	1	50.8 μs + 0.5/f_{SCK}
1	0	1	0	0	53.3 μs + 0.5/f_{SCK}
1	0	1	0	1	55.8 μs + 0.5/f_{SCK}
1	0	1	1	0	58.4 μs + 0.5/f_{SCK}
1	0	1	1	1	60.9 μs + 0.5/f_{SCK}
1	1	0	0	0	63.5 μs + 0.5/f_{SCK}
1	1	0	0	1	66.0 μs + 0.5/f_{SCK}
1	1	0	1	0	68.5 μs + 0.5/f_{SCK}
1	1	0	1	1	71.1 μs + 0.5/f_{SCK}
1	1	1	0	0	73.6 μs + 0.5/f_{SCK}
1	1	1	0	1	76.1 μs + 0.5/f_{SCK}
1	1	1	1	0	78.6 μs + 0.5/f_{SCK}
1	1	1	1	1	81.2 μs + 0.5/f_{SCK}

Notes 1. The interval time is 2/f_{SCK}.

2. The data transfer interval time is found from the following expressions (n: Value set to ADTI0 to ADTI4).

<1> n = 0

$$\text{Interval time} = \frac{2}{f_{SCK}} + \frac{0.5}{f_{SCK}}$$

<2> n = 1 to 31

$$\text{Interval time} = \frac{n+1}{f_{SCK}} + \frac{0.5}{f_{SCK}}$$

Cautions 1. Do not write ADTI during operation of the automatic data transmit/receive function.

2. Be sure to set bits 5 and 6 to 0.

3. When controlling the interval time of automatic transmit/receive data transfer by using ADTI, busy control is invalid. (Refer to 14.4.3 (4) (a) Busy control option.)

Remark f_X: System clock oscillation frequency

f_{SCK}: Serial clock frequency

(4) Port mode register 2 (PM2)

PM2 is a register that sets input/output of port 2 in 1-bit units.

When using the P21/SO1 pin as a serial data output, set PM21 and the output latch of P21 to 0.

When using the P22/$\overline{\text{SCK1}}$ pin as a clock output, set PM22 and the output latch of P22 to 0.

When using the P25/STB pin as a strobe output, set PM25 and the output latch of P25 to 0.

When using the P20/SI1 pin as a serial data input, the P22/$\overline{\text{SCK1}}$ pin as a clock input, and the P24/BUSY pin as a busy input, set PM20, PM22, and PM24 to 1. At this time, the output latches of P20, P22, and P24 can be either 0 or 1.

PM2 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets the value of PM2 to FFH.

Figure 14-5. Format of Port Mode Register 2 (PM2)

Address: FF22H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20

PM2n	I/O mode selection of P2n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

14.4 Operations of Serial Interface SIO1

The following three operating modes are available for serial interface SIO1.

- Operation stop mode
- 3-wire serial I/O mode
- 3-wire serial I/O mode with automatic transmit/receive function

14.4.1 Operation stop mode

Serial transfer is not carried out in the operation stop mode. Serial I/O shift register 1 (SIO1) does not carry out shift operations either, and thus it can be used as an ordinary 8-bit register.

In the operation stop mode, the P20/SI1, P21/SO1, P22/$\overline{\text{SCK1}}$, P23/STB and P24/BUSY pins can be used as ordinary I/O ports.

(1) Register setting

The operation stop mode is set using serial operating mode register 1 (CSIM1).

CSIM1 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM1 to 00H.

Symbol	<7>	6	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM1	CSIE1	DIR1	ATE	LSCK1	0	0	SCL11	SCL10	FF68H	00H	R/W

CSIE1	Enable/disable of operation of serial interface SIO1		
	Shift register operation	Serial counter	Port[Note 1]
0	Operation stopped	Cleared	Port function
1	Operation enabled	Count operation enabled	Serial function + port function

Note When CSIE1 = 0 (SIO1 operation stop status), the P20/SI1, P21/SO1, P22/$\overline{\text{SCK1}}$, P23/STB, and P24/BUSY pins can be used as port pins.

14.4.2 3-wire serial I/O mode operation

The 3-wire serial I/O mode is effective for connection of peripheral I/O units and display controllers that incorporate a conventional synchronous serial interface such as the 75XL, 78K and 17K Series.

Communication is carried out with three lines, one each for the serial clock ($\overline{SCK1}$), serial output (SO1), and serial input (SI1).

(1) Register setting

The 3-wire serial I/O mode is set using serial operating mode register 1 (CSIM1), port mode register 2 (PM2), and port 2 (P2).

(a) Serial operating mode register 1 (CSIM1)

CSIM1 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM1 to 00H.

Symbol	<7>	6	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM1	CSIE1	DIR1	ATE	LSCK1	0	0	SCL11	SCL10	FF68H	00H	R/W

CSIE1	Enable/disable of operation of serial interface SIO1		
	Shift register operation	Serial counter	Port[Note 1]
0	Operation stopped	Cleared	Port function
1	Operation enabled	Count operation enabled	Serial function + port function

DIR1	Start bit
0	MSB
1	LSB

ATE	Selection of operating mode of serial interface SIO1
0	3-wire serial I/O mode
1	3-wire serial I/O mode with automatic transmit/receive function

LSCK1	Chip enable control of $\overline{SCK1}$ pin
0	$\overline{SCK1}$ is used as port (P22) when CSIE1 = 0. $\overline{SCK1}$ is used for clock output when CSIE1 = 1.
1	$\overline{SCK1}$ is fixed to high level when CSIE1 = 0. $\overline{SCK1}$ is used for clock output when CSIE1 = 1.

SCL11	SCL10	Selection of serial clock of serial interface SIO1
0	0	External clock input to $\overline{SCK1}$ pin[Note 2]
0	1	$f_X/2^4$ (394 kHz)
1	0	$f_X/2^5$ (197 kHz)
1	1	$f_X/2^6$ (98.4 kHz)

Notes 1. When CSIE1 = 0 (SIO1 operation stop status), the P20/SI1, P21/SO1, P22/$\overline{\text{SCK1}}$, P23/STB, and P24/BUSY pins can be used as port pins.

2. When external clock input is selected by clearing SCL11 and SCL10 to 0, 0, clear bits 2 (STRB) and 1 (BUSY1) of the automatic data transmit/receive control register (ADTC) to 0, 0.

Remark ( ): fx = 6.3 MHz

(b) Port mode register 2 (PM2)

PM2 is a register that sets input/output of port 2 in 1-bit units.

When using the P21/SO1 pin as a serial data output, set PM21 and the output latch of P21 to 0.

When using the P22/$\overline{\text{SCK1}}$ pin as a clock output, set PM22 and the output latch of P22 to 0.

When using the P25/STB pin as a strobe output, set PM25 and the output latch of P25 to 0.

When using the P20/SI1 pin as a serial data input, the P22/$\overline{\text{SCK1}}$ pin as a clock input, and the P24/BUSY pin as a busy input, set PM20, PM22, and PM24 to 1. At this time, the output latches of P20, P22, and P24 can be either 0 or 1.

PM2 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets the value of PM2 to FFH.

Address: FF22H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20

PM2n	I/O mode selection of P2n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

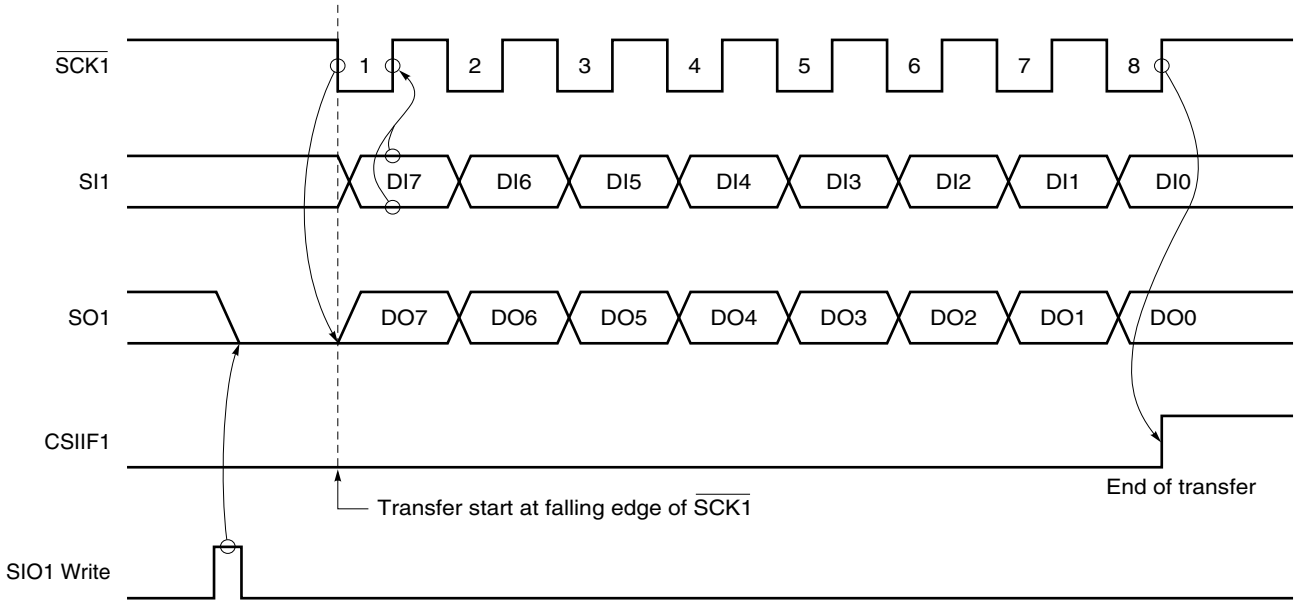
(2) Communication operation

The 3-wire serial I/O mode is used for data transmission/reception in 8-bit units. Bit-wise data transmission/reception is carried out in synchronization with the serial clock.

The shift operation of serial I/O shift register 1 (SIO1) is carried out at the falling edge of the serial clock ($\overline{SCK1}$). The transmit data is held in the SO1 latch and is output from the SO1 pin. The receive data input to the SI1 pin is latched into SIO1 at the rising edge of $\overline{SCK1}$.

Upon termination of 8-bit transfer, the SIO1 operation stops automatically and the interrupt request flag (CSIIF1) is set.

Figure 14-6. 3-Wire Serial I/O Mode Timing



Caution The SO1 pin becomes low level by SIO1 write.

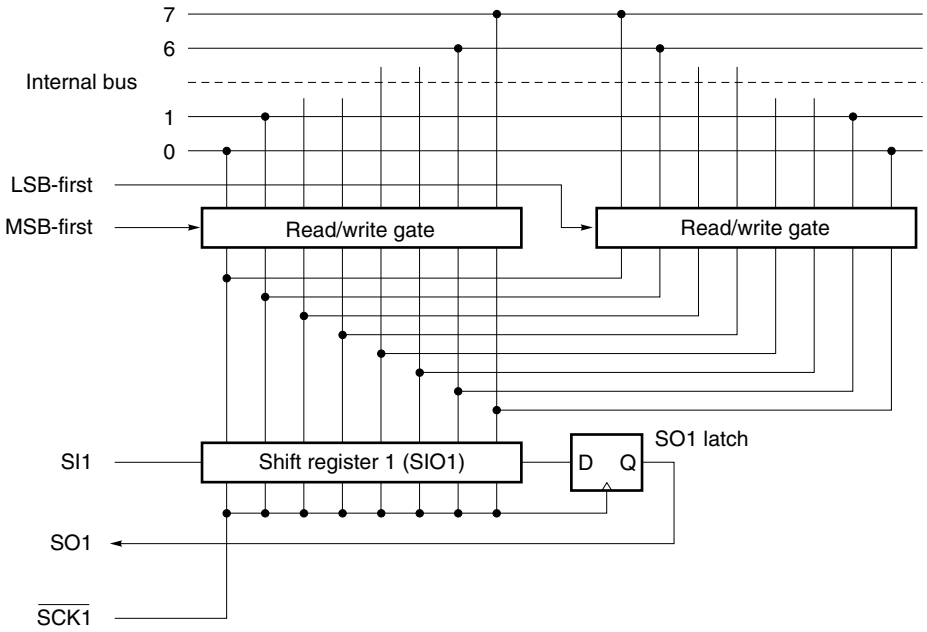
(3) MSB/LSB switching as the start bit

In the 3-wire serial I/O mode, transfer can be selected to start from the MSB or LSB.

Figure 14-7 shows the configuration of serial I/O shift register 1 (SIO1) and the internal bus. As shown in the figure, MSB/LSB can be read/written in reverse form.

MSB/LSB switching as the start bit can be specified by bit 6 (DIR1) of serial operating mode register 1 (CSIM1).

Figure 14-7. Circuit for Switching Transfer Bit Order



Start bit switching is realized by switching the bit order for data written to SIO1. The SIO1 shift order remains unchanged.

Thus, switching between MSB-first and LSB-first must be performed before writing data to the shift register.

(4) Transfer start

Serial transfer is started by setting transfer data to serial I/O shift register 1 (SIO1) when the following two conditions are satisfied.

- Serial interface SIO1 operation control bit (bit 7 (CSIE1) of serial operating mode register 1 (CSIM1)) = 1
- Internal serial clock is stopped or SCK1 is a high level after 8-bit serial transfer.

Caution If CSIE1 is set to 1 after data is written to SIO1, transfer does not start.

Upon termination of 8-bit transfer, serial transfer automatically stops and the interrupt request flag (CSIIF1) is set.

14.4.3 3-wire serial I/O mode operation with automatic transmit/receive function

This 3-wire serial I/O mode is used for transmission/reception of a maximum of 32 bytes of data without the use of software. Once transfer is started, the set number of bytes of data prestored in the RAM can be transmitted, and the set number of bytes data can be received and stored in the RAM.

Handshake signals (STB and BUSY) are supported by hardware to transmit/receive data continuously, facilitating connection of an OSD (On Screen Display) LSI and peripheral LSIs including an LCD controller/driver.

(1) Register setting

The 3-wire serial I/O mode with automatic transmit/receive function is set using serial operating mode register 1 (CSIM1), the automatic data transmit/receive control register (ADTC), automatic data transmit/receive interval specification register (ADTI), port mode register 2 (PM2), and port 2 (P2).

(a) Serial operating mode register 1 (CSIM1)

CSIM1 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets CSIM1 to 00H.

Symbol	<7>	6	<5>	4	3	2	1	0	Address	After reset	R/W
CSIM1	CSIE1	DIR1	ATE	LSCK1	0	0	SCL11	SCL10	FF68H	00H	R/W

CSIE1	Enable/disable of operation of serial interface SIO1		
	Shift register operation	Serial counter	Port[Note 1]
0	Operation stopped	Cleared	Port function
1	Operation enabled	Count operation enabled	Serial function + port function

DIR1	Start bit
0	MSB
1	LSB

ATE	Selection of operating mode of serial interface SIO1
0	3-wire serial I/O mode
1	3-wire serial I/O mode with automatic transmit/receive function

LSCK1	Chip enable control of $\overline{SCK1}$ pin
0	$\overline{SCK1}$ is used as port (P22) when CSIE1 = 0. $\overline{SCK1}$ is used for clock output when CSIE1 = 1.
1	$\overline{SCK1}$ is fixed to high level when CSIE1 = 0. $\overline{SCK1}$ is used for clock output when CSIE1 = 1.

SCL11	SCL10	Selection of serial clock of serial interface SIO1
0	0	External clock input to $\overline{SCK1}$ pin[Note 2]
0	1	$f_X/2^4$ (394 kHz)
1	0	$f_X/2^5$ (197 kHz)
1	1	$f_X/2^6$ (98.4 kHz)

Notes 1. When CSIE1 = 0 (SIO1 operation stop status), the P20/SI1, P21/SO1, P22/$\overline{SCK1}$, P23/STB, and P24/BUSY pins can be used as port pins.

2. When external clock input is selected by clearing SCL11 and SCL10 to 0, 0, clear bits 2 (STRB) and 1 (BUSY1) of the automatic data transmit/receive control register (ADTC) to 0, 0.

(b) Automatic data transmit/receive control register (ADTC)

ADTC is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets ADTC to 00H.

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
ADTC	RE	ARLD	ERCE	ERR	TRF	STRB	BUSY1	BUSY0	FF6AH	00H	R/W[Note 1]

R/W

BUSY1	BUSY0	Control of busy input
0	×	Not using busy input[Note 2]
1	0	Busy input enabled (active high)
1	1	Busy input enabled (active low)

R/W

STRB	Control of strobe output
0	Strobe output disabled[Note 3]
1	Strobe output enabled

R

TRF	Status of automatic transmit/receive function[Note 4]
0	End of automatic transmission/reception detected. (This bit is set to 0 upon suspension of automatic transmission/reception or when ARLD = 0.)
1	Automatic transmission/reception in progress. (This bit is set to 1 when data is written to SIO1.)

R

ERR	Error detection of automatic transmit/receive function
0	No error (This bit is set to 0 when data is written to SIO1)
1	Error occurred

R/W

ERCE	Error check control of automatic transmit/ receive function
0	Error check disabled
1	Error check enabled (only when BUSY1 = 1)

R/W

ARLD	Operating mode selection of automatic transmit/ receive function
0	Single operating mode
1	Repetitive operating mode

R/W

RE	Receive control of automatic transmit/receive function
0	Receive disabled[Note 5]
1	Receive enabled

Notes 1. Bits 3 and 4 (TRF and ERR) are read-only bits.

2. When BUSY1 is reset to 0, P24 (CMOS I/O) is used even when bit 7 (CSIE1) of serial operating mode register 1 (CSIM1) is set to 1.

3. When STRB is reset to 0, P23 (CMOS I/O) is used even when bit 7 (CSIE1) of CSIM1 is set to 1.

4. When an interrupt is acknowledged, interrupt request flag CSIIF1 is cleared. Therefore, use TRF, instead of CSIIF1, to identify the completion of automatic transmission/reception.

5. When RE is reset to 0, P20 (CMOS I/O) is used even when bit 7 (CSIE1) of CSIM1 is set to 1.

Caution When an external clock input is selected with bit 1 (CSIM11) of serial operating mode register 1 (CSIM1) set to 0, set STRB and BUSY1 of ADTC to 0, 0 (when an external clock is input, handshake control cannot be performed).

Remark ×: Don't care

(c) Automatic data transmit/receive interval specification register (ADTI)

This register sets the automatic data transmit/receive function data transfer interval.

ADTI is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets ADTI to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADTI	ADTI7	0	0	ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	FF6BH	00H	R/W

ADTI7	Control of data transfer interval
0	No control of interval by ADTI Note 1
1	Control of interval by ADTI (ADTI0 to ADTI4)

ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	Specification of data transfer interval (f_X = 6.3 MHz, f_{SCK} = 394 kHz)Note 2
0	0	0	0	0	5.08 μs + 0.5/f_{SCK}
0	0	0	0	1	
0	0	0	1	0	7.61 μs + 0.5/f_{SCK}
0	0	0	1	1	10.2 μs + 0.5/f_{SCK}
0	0	1	0	0	12.7 μs + 0.5/f_{SCK}
0	0	1	0	1	15.2 μs + 0.5/f_{SCK}
0	0	1	1	0	17.8 μs + 0.5/f_{SCK}
0	0	1	1	1	20.3 μs + 0.5/f_{SCK}
0	1	0	0	0	22.8 μs + 0.5/f_{SCK}
0	1	0	0	1	25.4 μs + 0.5/f_{SCK}
0	1	0	1	0	27.9 μs + 0.5/f_{SCK}
0	1	0	1	1	30.5 μs + 0.5/f_{SCK}
0	1	1	0	0	33.0 μs + 0.5/f_{SCK}
0	1	1	0	1	35.5 μs + 0.5/f_{SCK}
0	1	1	1	0	38.1 μs + 0.5/f_{SCK}
0	1	1	1	1	40.6 μs + 0.5/f_{SCK}

(Continued)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADTI	ADTI7	0	0	ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	FF6BH	00H	R/W

ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	Specification of data transfer interval (f_X = 6.3 MHz, f_{SCK} = 394 kHz)Note 2
1	0	0	0	0	43.1 μs + 0.5/f_{SCK}
1	0	0	0	1	45.7 μs + 0.5/f_{SCK}
1	0	0	1	0	48.2 μs + 0.5/f_{SCK}
1	0	0	1	1	50.8 μs + 0.5/f_{SCK}
1	0	1	0	0	53.3 μs + 0.5/f_{SCK}
1	0	1	0	1	55.8 μs + 0.5/f_{SCK}
1	0	1	1	0	58.4 μs + 0.5/f_{SCK}
1	0	1	1	1	60.9 μs + 0.5/f_{SCK}
1	1	0	0	0	63.5 μs + 0.5/f_{SCK}
1	1	0	0	1	66.0 μs + 0.5/f_{SCK}
1	1	0	1	0	68.5 μs + 0.5/f_{SCK}
1	1	0	1	1	71.1 μs + 0.5/f_{SCK}
1	1	1	0	0	73.6 μs + 0.5/f_{SCK}
1	1	1	0	1	76.1 μs + 0.5/f_{SCK}
1	1	1	1	0	78.6 μs + 0.5/f_{SCK}
1	1	1	1	1	81.2 μs + 0.5/f_{SCK}

Notes 1. The interval time is 2/f_{SCK}.

2. The data transfer interval time is found from the following expressions (n: Value set to ADTI0 to ADTI4).

<1> n = 0

$$\text{Interval time} = \frac{2}{f_{SCK}} + \frac{0.5}{f_{SCK}}$$

<2> n = 1 to 31

$$\text{Interval time} = \frac{n+1}{f_{SCK}} + \frac{0.5}{f_{SCK}}$$

Cautions 1. Do not write ADTI during operation of the automatic data transmit/receive function.

2. Be sure to set bits 5 and 6 to 0.

3. When controlling the interval time of automatic transmit/receive data transfer by using ADTI, busy control is invalid. (Refer to 14.4.3 (4) (a) Busy control option.)

Remark f_X: System clock oscillation frequency

f_{SCK}: Serial clock frequency

(d) Port mode register 2 (PM2)

PM2 is a register that sets input/output of port 2 in 1-bit units.

When using the P21/SO1 pin as a serial data output, set PM21 and the output latch of P21 to 0.

When using the P22/$\overline{\text{SCK1}}$ pin as a clock output, set PM22 and the output latch of P22 to 0.

When using the P25/STB pin as a strobe output, set PM25 and the output latch of P25 to 0.

When using the P20/SI1 pin as a serial data input, the P22/$\overline{\text{SCK1}}$ pin as a clock input, and the P24/BUSY pin as a busy input, set PM20, PM22, and PM24 to 1. At this time, the output latches of P20, P22, and P24 can be either 0 or 1.

PM2 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets the value of PM2 to FFH.

Address: FF22H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20

PM2n	I/O mode selection of P2n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

(2) Automatic transmit/receive data setting

(a) Transmit data setting

<1> Write transmit data from the least significant address FAC0H of buffer RAM (up to FADFH). The transmit data should be in the order of higher address to lower address.
<2> Set the value obtained by subtracting 1 from the number of transmit data bytes to the automatic data transmit/receive address pointer (ADTP).

(b) Automatic transmit/receive mode setting

<1> Set bit 7 (CSIE1) and bit 5 (ATE) of serial operating mode register 1 (CSIM1) to 1.
<2> Set bit 7 (RE) of the automatic data transmit/receive control register (ADTC) to 1.
<3> Set the data transmit/receive interval in the automatic data transmit/receive interval specification register (ADTI).
<4> Write any value to serial I/O shift register 1 (SIO1) (transfer start trigger).

Caution Writing any value to SIO1 orders the start of automatic transmit/receive operation; the written value has no meaning.

The following operations are automatically carried out when (a) and (b) are carried out.

- After the buffer RAM data specified by ADTP is transferred to SIO1, transmission is carried out (start of automatic transmission/reception).
- The received data is written to the buffer RAM address specified by ADTP.
- ADTP is decremented and the next data transmission/reception is carried out. Data transmission/reception continues until the ADTP decremental output becomes 00H and address FAC0H data is output (end of automatic transmission/reception).
- When automatic transmission/reception is terminated, bit 3 (TRF) of ADTC is cleared to 0.

(3) Communication operation

(a) Basic transmit/receive mode

This transmit/receive mode is the same as the 3-wire serial I/O mode in which the specified number of data are transmitted/received in 8-bit units.

Serial transfer is started when any data is written to serial I/O shift register 1 (SIO1) while bit 7 (CSIE1) of serial operating mode register 1 (CSIM1) is set to 1.

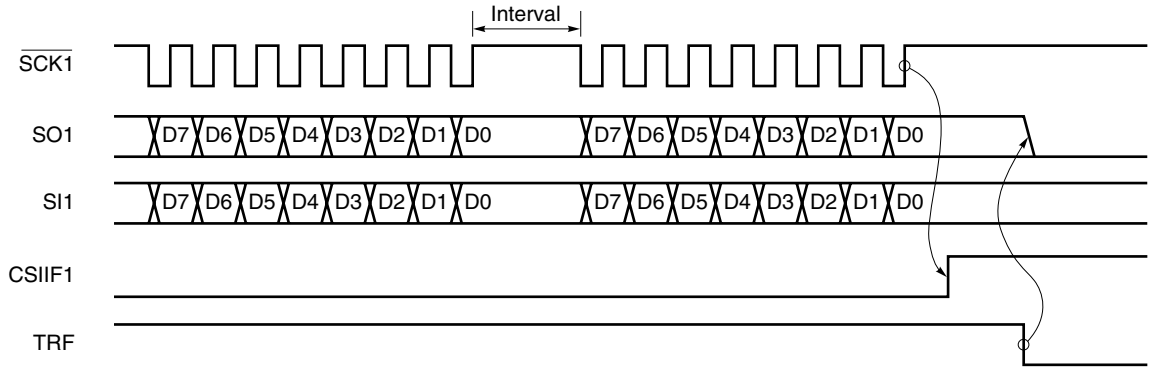
Upon completion of transmission of the last byte, the interrupt request flag (CSIIF1) is set. The termination of automatic transmission/reception should be checked by using bit 3 (TRF) of the automatic data transmit/receive control register (ADTC), not by CSIIF1 because the CSIIF1 interrupt request flag is cleared if an interrupt is acknowledged.

If busy control and strobe control are not executed, the P23/STB and P24/BUSY pins can be used as normal I/O ports.

Figure 14-8 shows the basic transmit/receive mode operation timing, and Figure 14-9 shows the operation flowchart.

Figure 14-10 shows the buffer RAM operation in 6-byte transmission.

Figure 14-8. Basic Transmit/Receive Mode Operation Timing

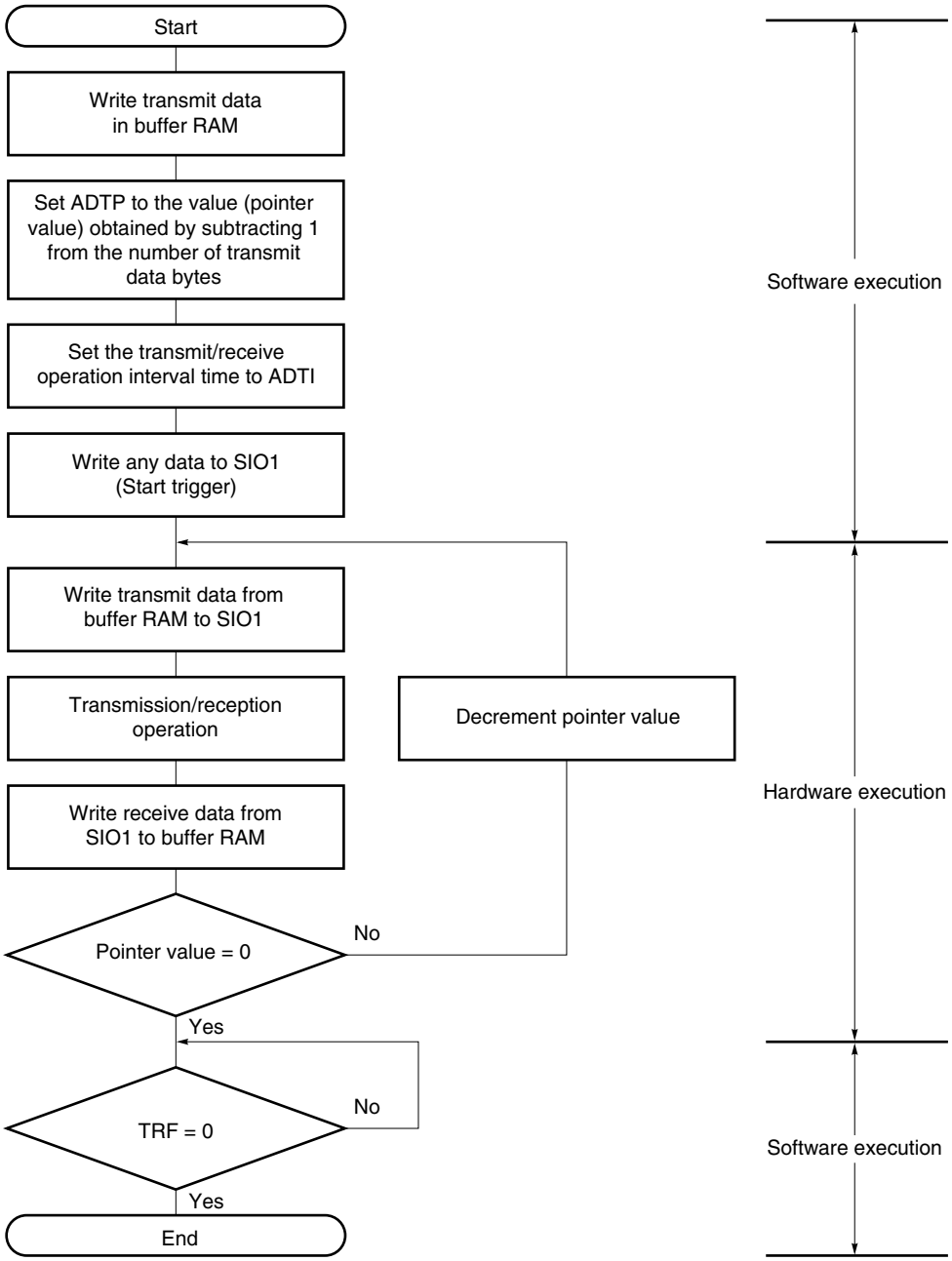


Cautions 1. Because, in the basic transmit/receive mode, the automatic transmit/receive function writes/reads data to/from the buffer RAM after 1-byte transmission/reception, an interval is inserted until the next transmission/reception. As the buffer RAM write/read is performed at the same time as CPU processing, the maximum interval is dependent upon the CPU processing and the value of the automatic data transmit/receive interval specification register (ADTI) (refer to (6) Automatic data transmit/receive interval).

2. When TRF is cleared, the SO1 pin becomes low level.

Remark CSIIF: Interrupt request flag
TRF: Bit 3 of the automatic data transmit/receive control register (ADTC)

Figure 14-9. Basic Transmit/Receive Mode Flowchart



ADTP: Automatic data transmit/receive address pointer
ADTI: Automatic data transmit/receive interval specification register
SIO1: Serial I/O shift register 1
TRF: Bit 3 of the automatic data transmit/receive control register (ADTC)

In 6-byte transmission/reception (bit 6 (ARLD) and bit 7 (RE) of the automatic data transmit/receive control register (ADTC) = 0 and 1, respectively) in basic transmit/receive mode, the buffer RAM operates as follows.

(i) Before transmission/reception (refer to Figure 14-10 (a))

After any data has been written to SIO1 (start trigger: this data is not transferred), transmit data 1 (T1) is transferred from the buffer RAM to SIO1. When transmission of the first byte is completed, receive data 1 (R1) is transferred from SIO1 to the buffer RAM, and the automatic data transmit/receive address pointer (ADTP) is decremented. Then transmit data 2 (T2) is transferred from the buffer RAM to SIO1.

(ii) 4th byte transmit/receive point (refer to Figure 14-10 (b))

Transmission/reception of the third byte is completed, and transmit data 4 (T4) is transferred from the buffer RAM to SIO1. When transmission of the fourth byte is completed, receive data 4 (R4) is transferred from SIO1 to the buffer RAM, and ADTP is decremented.

(iii) Completion of transmission/reception (refer to Figure 14-10 (c))

When transmission of the sixth byte is completed, receive data 6 (R6) is transferred from SIO1 to the buffer RAM, and the interrupt request flag (CSIIF1) is set (INTCSI1 generation).

Figure 14-10. Buffer RAM Operation in 6-Byte Transmission/Reception (in Basic Transmit/Receive Mode) (1/2)

(a) Before transmission/reception

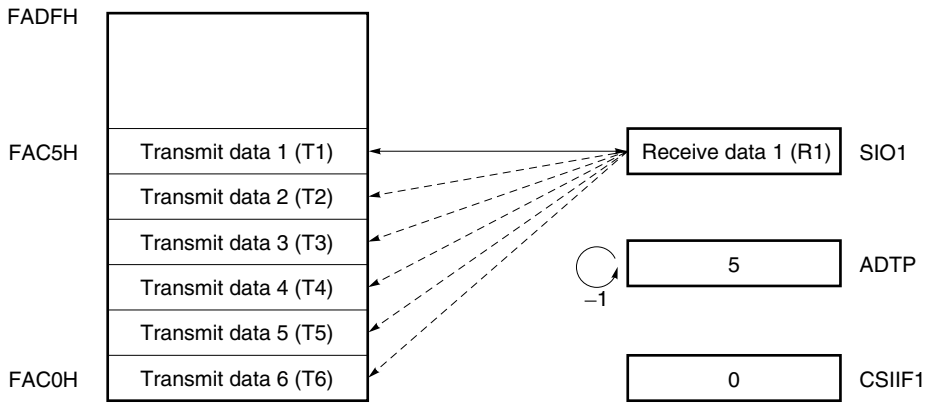
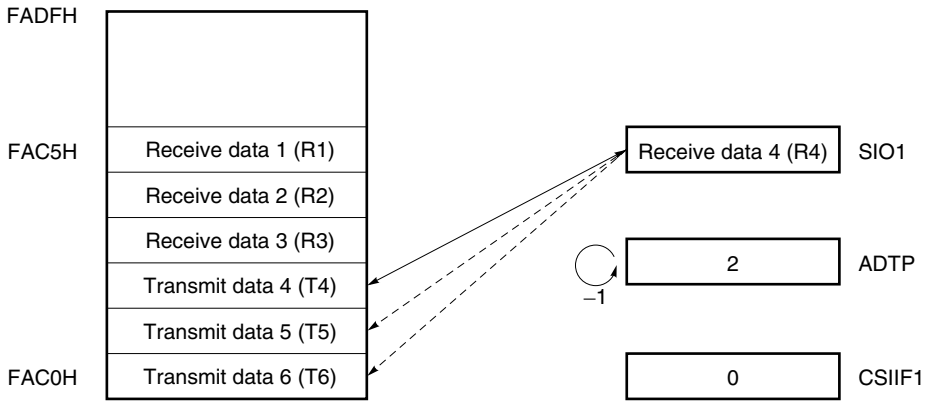
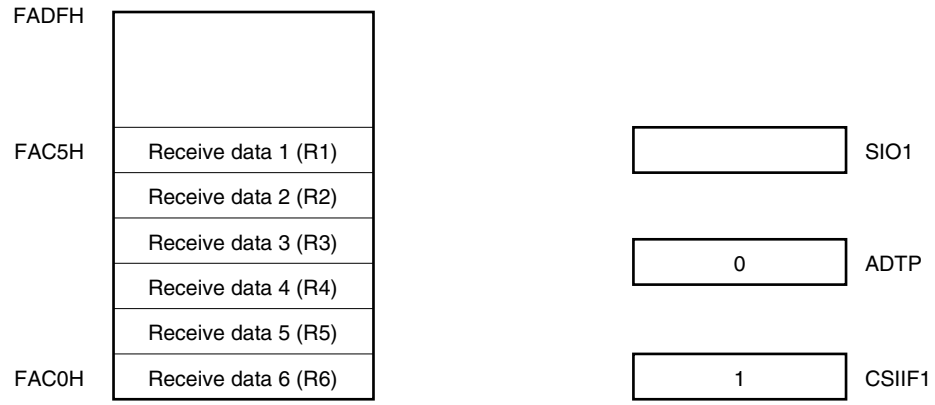


Figure 14-10. Buffer RAM Operation in 6-Byte Transmission/Reception (in Basic Transmit/Receive Mode) (2/2)

(b) 4th byte transmission/reception



(c) Completion of transmission/reception



(b) Basic transmit mode

In this mode, the specified number of 8-bit unit data are transmitted.

Serial transfer is started when any data is written to serial I/O shift register 1 (SIO1) while bit 7 (CSIE1) of serial operating mode register 1 (CSIM1) is set to 1, and bit 7 (RE) of the automatic data transmit/receive control register (ADTC) is set to 0.

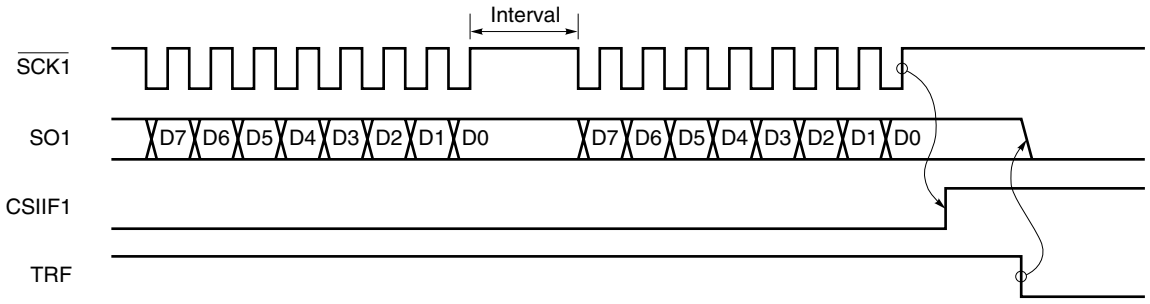
Upon completion of transmission of the last byte, the interrupt request flag (CSIIF1) is set. The termination of automatic transmission/reception should be checked by using bit 3 (TRF) of the automatic data transmit/receive control register (ADTC), not by CSIIF1.

If a receive operation, busy control, and strobe control are not executed, the P20/SI1, P23/STB and P24/BUSY pins can be used as normal I/O pins.

Figure 14-11 shows the basic transmit mode operation timing, and Figure 14-12 shows the operation flowchart.

Figure 14-13 shows the buffer RAM operation when repeatedly transmitting 6 bytes.

Figure 14-11. Basic Transmit Mode Operation Timing



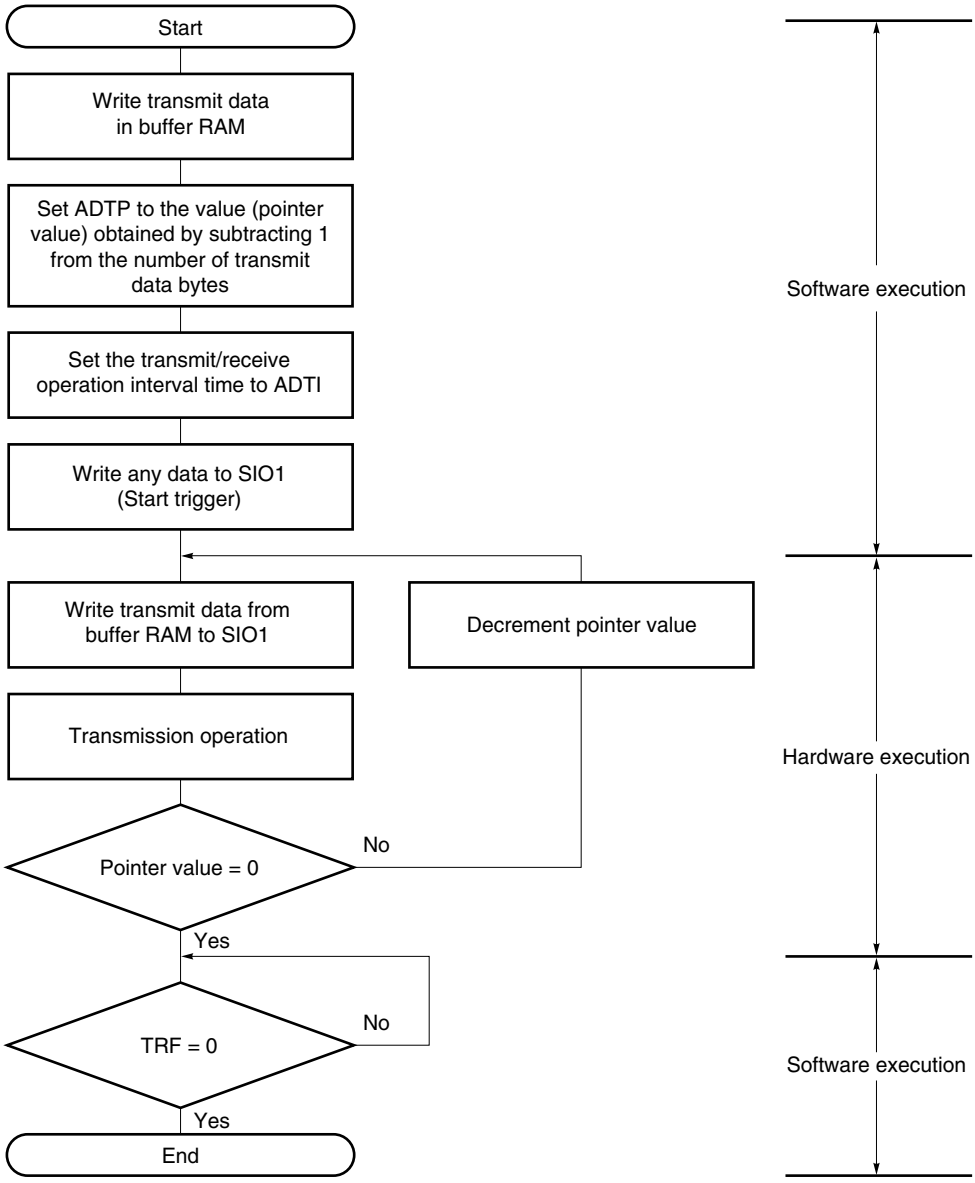
Cautions 1. Because, in the basic transmit mode, the automatic transmit/receive function reads data from the buffer RAM after 1-byte transmission, an interval is inserted until the next transmission. As the buffer RAM read is performed at the same time as CPU processing, the maximum interval is dependent upon the CPU processing and the value of the automatic data transmit/receive interval specification register (ADTI) (refer to (6) Automatic data transmit/receive interval).

2. When TRF is cleared, the SO1 pin becomes low level.

Remark CSIIF: Interrupt request flag

TRF: Bit 3 of the automatic data transmit/receive control register (ADTC)

Figure 14-12. Basic Transmit Mode Flowchart



ADTP: Automatic data transmit/receive address pointer
ADTI: Automatic data transmit/receive interval specification register
SIO1: Serial I/O shift register 1
TRF: Bit 3 of the automatic data transmit/receive control register (ADTC)

In 6-byte transmission (bit 6 (ARLD) and bit 7 (RE) of the automatic data transmit/receive control register (ADTC) are 0) in basic transmit mode, the buffer RAM operates as follows.

(i) Before transmission (refer to Figure 14-13 (a))

After any data has been written to SIO1 (start trigger: this data is not transferred), transmit data 1 (T1) is transferred from the buffer RAM to SIO1. When transmission of the first byte is completed, ADTP is decremented. Then transmit data 2 (T2) is transferred from the buffer RAM to SIO1.

(ii) 4th byte transmission point (refer to Figure 14-13 (b))

Transmission of the third byte is completed, and transmit data 4 (T4) is transferred from the buffer RAM to SIO1. When transmission of the fourth byte is completed, ADTP is decremented.

(iii) Completion of transmission/reception (refer to Figure 14-13 (c))

When transmission of the sixth byte is completed, the interrupt request flag (CSIIF1) is set (INTCSI1 generation).

Figure 14-13. Buffer RAM Operation in 6-Byte Transmission (in Basic Transmit Mode) (1/2)

(a) Before transmission

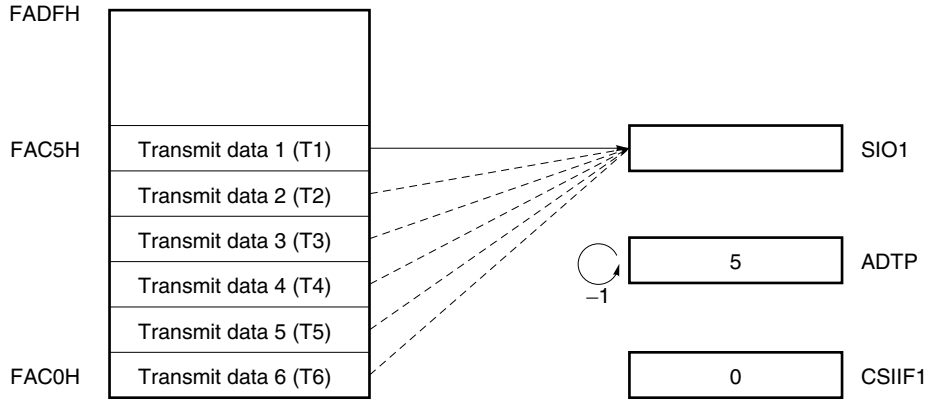
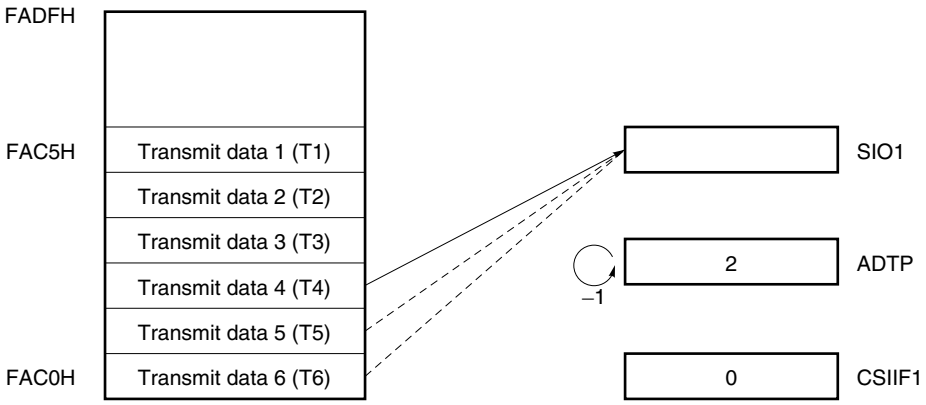
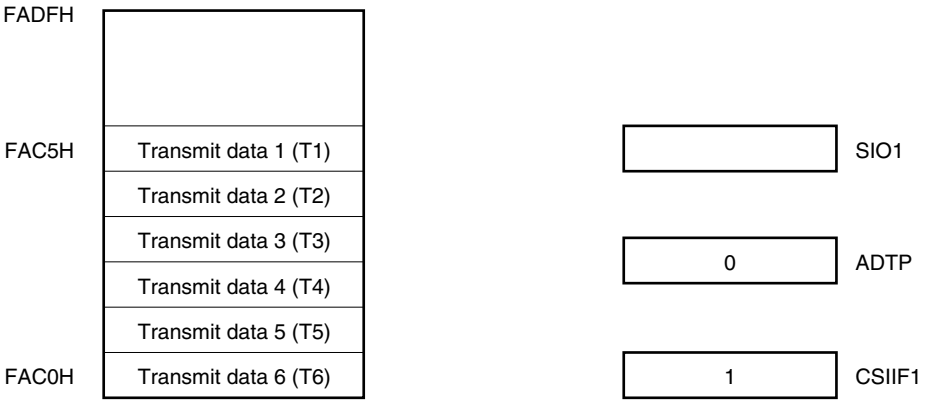


Figure 14-13. Buffer RAM Operation in 6-Byte Transmission (in Basic Transmit Mode) (2/2)

(b) 4th byte transmission point



(c) Completion of transmission/reception



(c) Repeat transmit mode

In this mode, data stored in the buffer RAM is transmitted repeatedly.

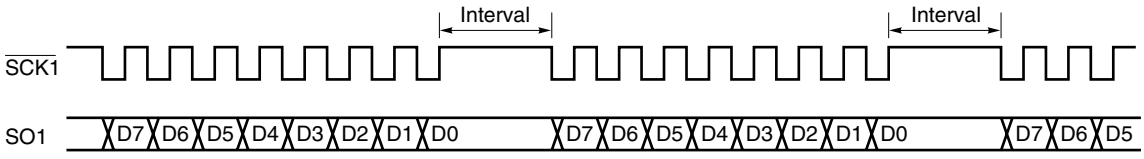
Serial transfer is started by writing any data to serial I/O shift register 1 (SIO1) when bit 7 (CSIE1) of serial operating mode register 1 (CSIM1) is set to 1, and bit 7 (RE) of the automatic data transmit/receive control register (ADTC) is set to 0.

Unlike the basic transmit mode, after the last byte (data in address FAC0H) has been transmitted, the interrupt request flag (CSIIF1) is not set, the value when the transmission was started is set in the automatic data transmit/receive address pointer (ADTP) again, and the buffer RAM contents are transmitted again.

When a reception operation, busy control, and strobe control are not performed, the P20/SI1, P23/STB and P24/BUSY pins can be used as ordinary I/O pins.

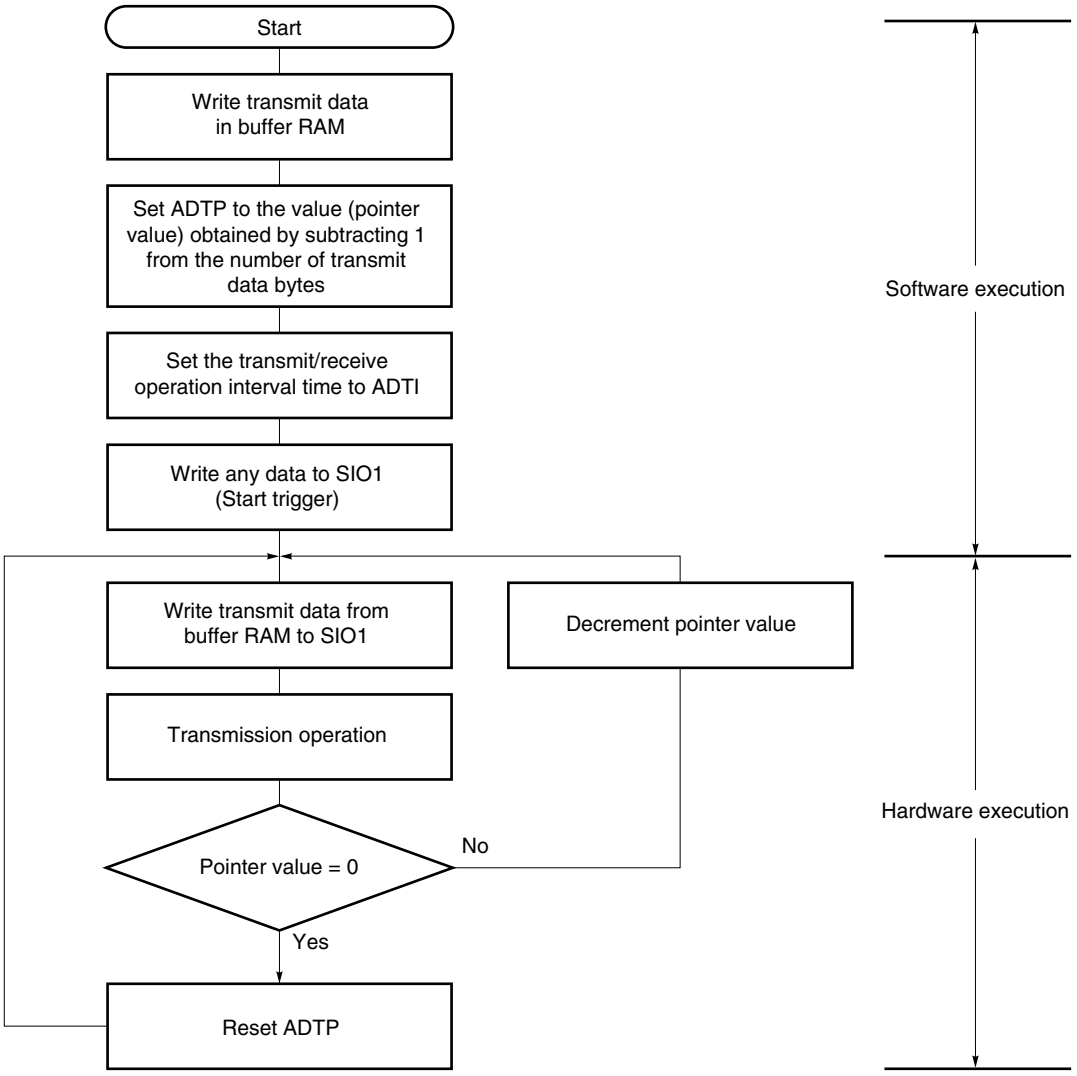
The repeat transmit mode operation timing is shown in Figure 14-14, and the operation flowchart in Figure 14-15.

Figure 14-14. Repeat Transmit Mode Operation Timing



Caution Since, in the repeat transmit mode, a read is performed on the buffer RAM after the transmission of one byte, the interval is included in the period up to the next transmission. As the buffer RAM read is performed at the same time as CPU processing, the maximum interval is dependent upon the CPU processing and the value of the automatic data transmit/receive interval specification register (ADTI) (refer to (6) Automatic data transmit/receive interval).

Figure 14-15. Repeat Transmit Mode Flowchart



ADTP: Automatic data transmit/receive address pointer
ADTI: Automatic data transmit/receive interval specification register
SIO1: Serial I/O shift register 1

In 6-byte transmission (bit 6 (ARLD) and bit 7 (RE) of the automatic data transmit/receive control register (ADTC) are 1 and 0, respectively) in repeat transmit mode, the buffer RAM operates as follows.

(i) Before transmission (refer to Figure 14-16 (a))

After any data has been written to SIO1 (start trigger: this data is not transferred), transmit data 1 (T1) is transferred from the buffer RAM to SIO1. When transmission of the first byte is completed, ADTP is decremented. Then transmit data 2 (T2) is transferred from the buffer RAM to SIO1.

(ii) Upon completion of transmission of 6 bytes (refer to Figure 14-16 (b))

When transmission of the sixth byte is completed, the interrupt request flag (CSIIF1) is not set. The previous pointer value is assigned to ADTP.

(iii) 7th byte transmission point (refer to Figure 14-16 (c))

Transmit data 1 (T1) is transferred from the buffer RAM to SIO1 again. When transmission of the first byte is completed, ADTP is decremented. Then transmit data 2 (T2) is transferred from the buffer RAM to SIO1.

Figure 14-16. Buffer RAM Operation in 6-Byte Transmission (in Repeat Transmit Mode) (1/2)

(a) Before transmission

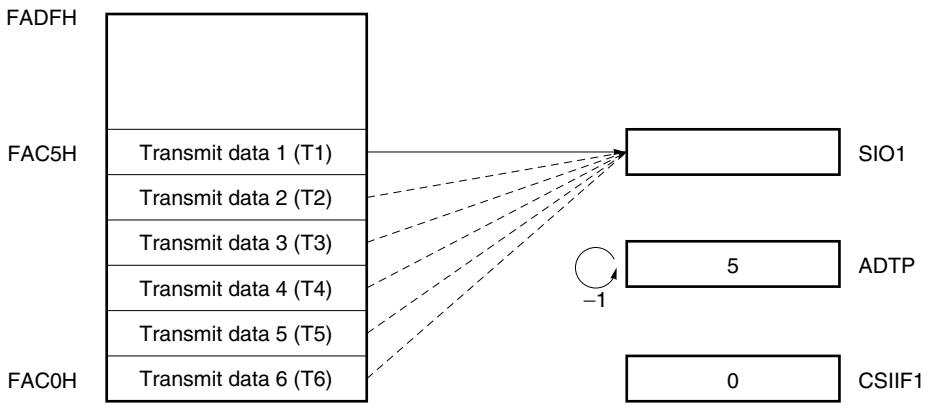
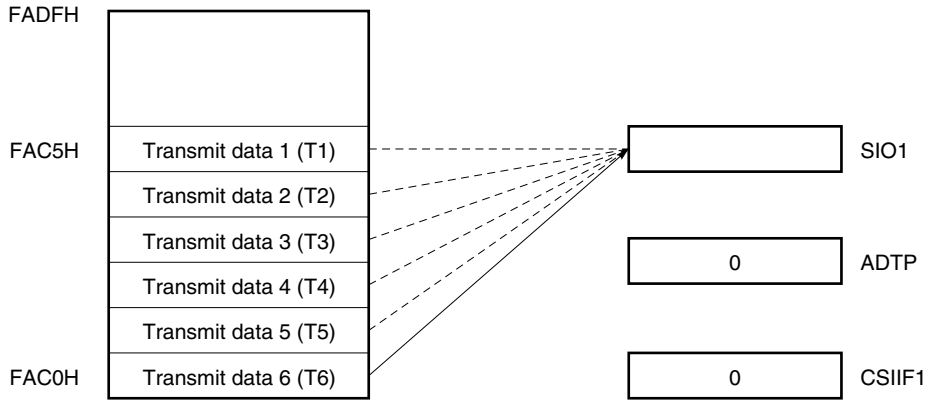
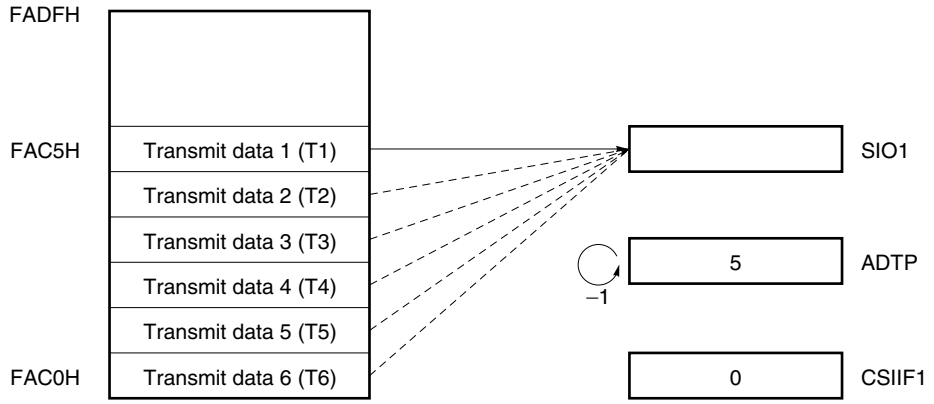


Figure 14-16. Buffer RAM Operation in 6-Byte Transmission (in Repeat Transmit Mode) (2/2)

(b) Upon completion of transmission of 6 bytes



(c) 7th byte transmission point



(d) Automatic transmission/reception suspension and restart

Automatic transmission/reception can be temporarily suspended by setting bit 7 (CSIE1) of serial operating mode register 1 (CSIM1) to 0.

If 8-bit data transfer is in progress, the transmission/reception is not suspended if bit 7 (CSIE1) is set to 0. It is suspended upon completion of 8-bit data transfer.

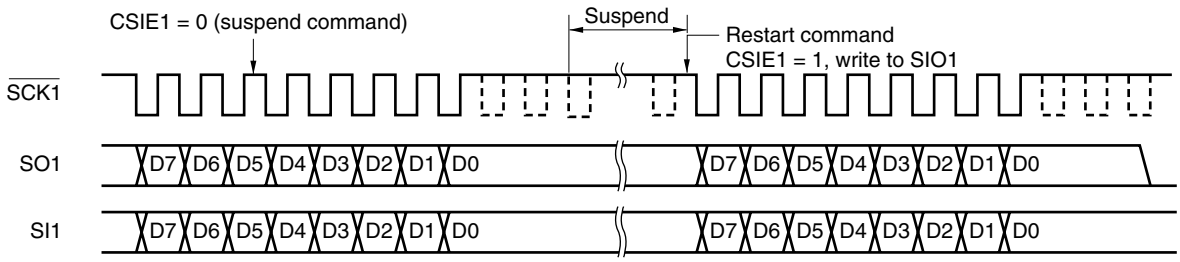
When suspended, bit 3 (TRF) of the automatic data transmit/receive control register (ADTC) is set to 0 after transfer of the 8th bit, and all the port pins that function alternately as serial interface pins (P20/SI1, P21/SO1, P22/$\overline{\text{SCK1}}$, P23/STB and P24/BUSY) are set to the port mode.

During restart of transmission/reception, the remaining data can be transferred by setting CSIE1 to 1 and writing any data to serial I/O shift register 1 (SIO1).

Cautions 1. If the HALT instruction is executed during automatic transmission/reception, transfer is suspended and the HALT mode is set if 8-bit data transfer is in progress.

2. When suspending automatic transmission/reception, do not change the operating mode to 3-wire serial I/O mode while TRF = 1.

Figure 14-17. Automatic Transmission/Reception Suspension and Restart



CSIE1: Bit 7 of serial operating mode register 1 (CSIM1)

(4) Synchronization control

Busy control and strobe control are functions used to synchronize transmission/reception between the master device and a slave device.

By using these functions, a shift in bits being transmitted or received can be detected.

(a) Busy control option

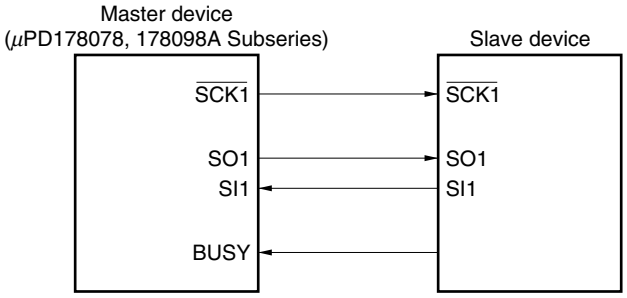
Busy control is a function used to keep the serial transmission/reception by the master device waiting while the busy signal output by a slave device to the master is active.

When using this busy control option, the following conditions must be satisfied.

- Bit 5 (ATE) of serial operating mode register 1 (CSIM1) is set to 1.
- Bit 1 (BUSY1) of the automatic data transmit/receive control register (ADTC) is set to 1.

Figure 14-18 shows the system configuration of the master device and a slave device when the busy control option is used.

Figure 14-18. System Configuration When Busy Control Option Is Used



The master device inputs the busy signal output by the slave device to the BUSY/P24 pin. The master device samples the input busy signal in synchronization with the falling of the serial clock. Even if the busy signal becomes active while 8-bit data is being transmitted or received, transmission/reception by the master is not kept waiting. If the busy signal is active at the rising edge of the serial clock 2 clocks after completion of transmission/reception of the 8-bit data, the busy input becomes valid. After that, the master transmission/reception is kept waiting while the busy signal is active.

The active level of the busy signal is set by bit 0 (BUSY0) of ADTC.

BUSY0 = 0: Active high

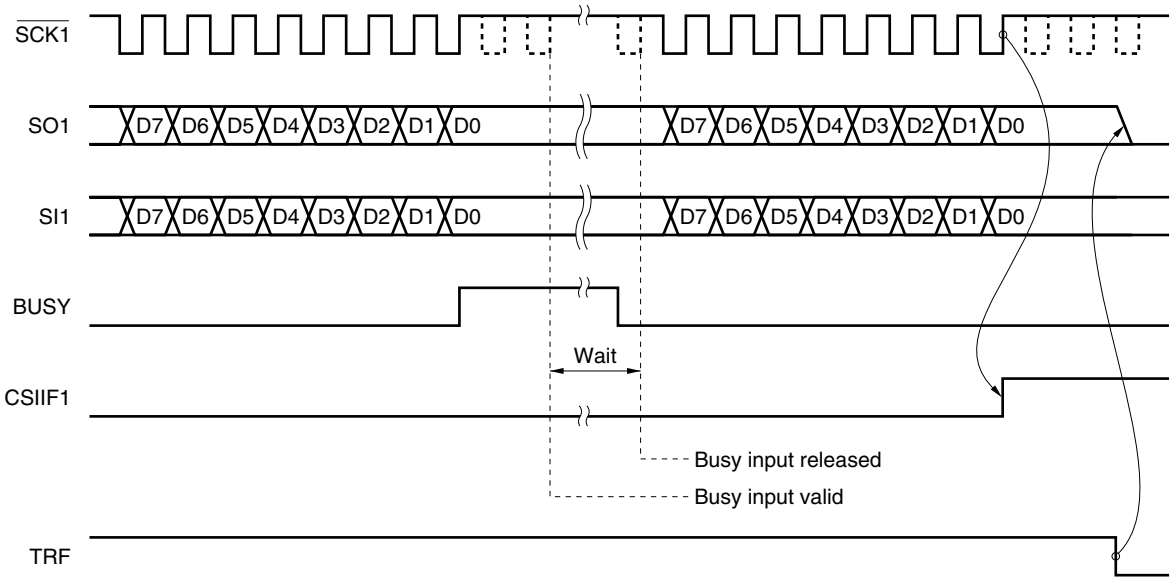
BUSY1 = 1: Active low

When using the busy control option, select the internal clock as the serial clock. Control by the busy signal cannot be implemented with the external clock.

Figure 14-19 shows the operation timing when the busy control option is used.

Caution Busy control cannot be used simultaneously with the interval time control function of the automatic data transmit/receive interval specification register (ADTI). If used, busy control is invalid.

Figure 14-19. Operation Timing When Busy Control Option Is Used (When BUSY0 = 0)



Caution If TRF is cleared, the SO1 pin goes low.

Remark CSIIF1: Interrupt request flag
TRF: Bit 3 of the automatic data transmit/receive control register (ADTC)

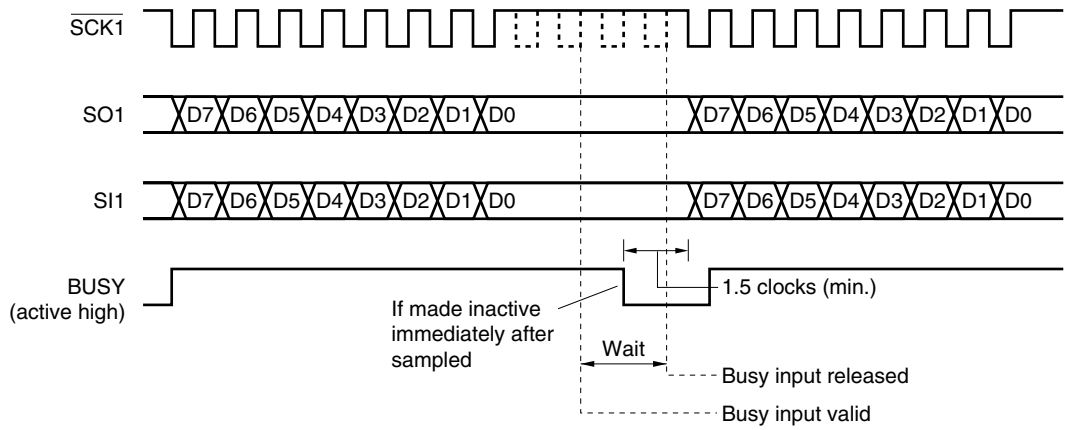
When the busy signal becomes inactive, waiting is released. If the sampled busy signal is inactive, transmission/reception of the next 8-bit data is started at the falling edge of the next clock.

Because the busy signal is asynchronous to the serial clocks, it takes up to 1 clock until the busy signal, even if made inactive by the slave, is sampled. It takes 0.5 clocks until data transfer is started after the busy signal was sampled.

To accurately release waiting, the slave must keep the busy signal inactive for at least the duration of 1.5 clocks.

Figure 14-20 shows the timing of the busy signal and wait release. This figure shows an example where the busy signal is active as soon as transmission/reception has been started.

Figure 14-20. Busy Signal and Wait Release (When BUSY0 = 0)



(b) Strobe control option

Strobe control is a function used to synchronize data transmission/reception between the master and a slave device. The master device outputs the strobe signal from the STB/P23 pin when 8-bit transmission/reception has been completed. By this signal, the slave device can determine the timing of the end of data transmission. Therefore, synchronization is established even if a bit shift occurs due to noise on the serial clock, and transmission of the next byte is not affected by the bit shift.

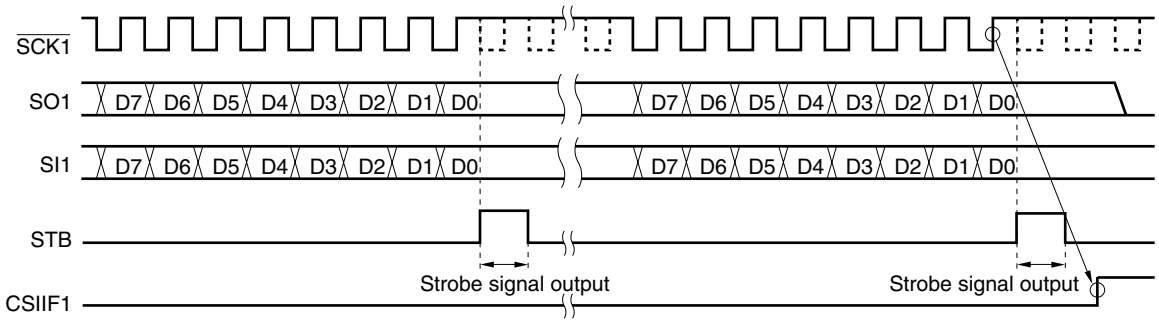
To use the strobe control option, the following conditions must be satisfied.

- Bit 5 (ATE) of serial operating mode register 1 (CSIM1) is set to 1.
- Bit 2 (STRB) of the automatic data transmit/receive control register (ADTC) is set to 1.

A strobe signal is output from the STB/P23 pin for the duration of 1 clock in synchronization with the falling of the ninth serial clock.

Figure 14-21 shows the operation timing when the strobe control option is used.

Figure 14-21. Operation Timing When Strobe Control Option Is Used



(c) Busy & strobe control option

Usually, the busy control and strobe control options are simultaneously used as handshake signals. In this case, the strobe signal is output from the STB/P23 pin, and the BUSY/P24 pin is sampled, and transmission/reception can be kept waiting while the busy signal is input.

When the strobe control option is not used, the P23/STB pin can be used as a normal I/O port pin.

To use the busy & strobe control option, the following conditions must be satisfied.

- Bit 5 (ATE) of serial operating mode register 1 (CSIM1) is set to 1.
- Bit 2 (STRB) and bit 1 (BUSY1) of the automatic data transmit/receive control register (ADTC) are set to 1.

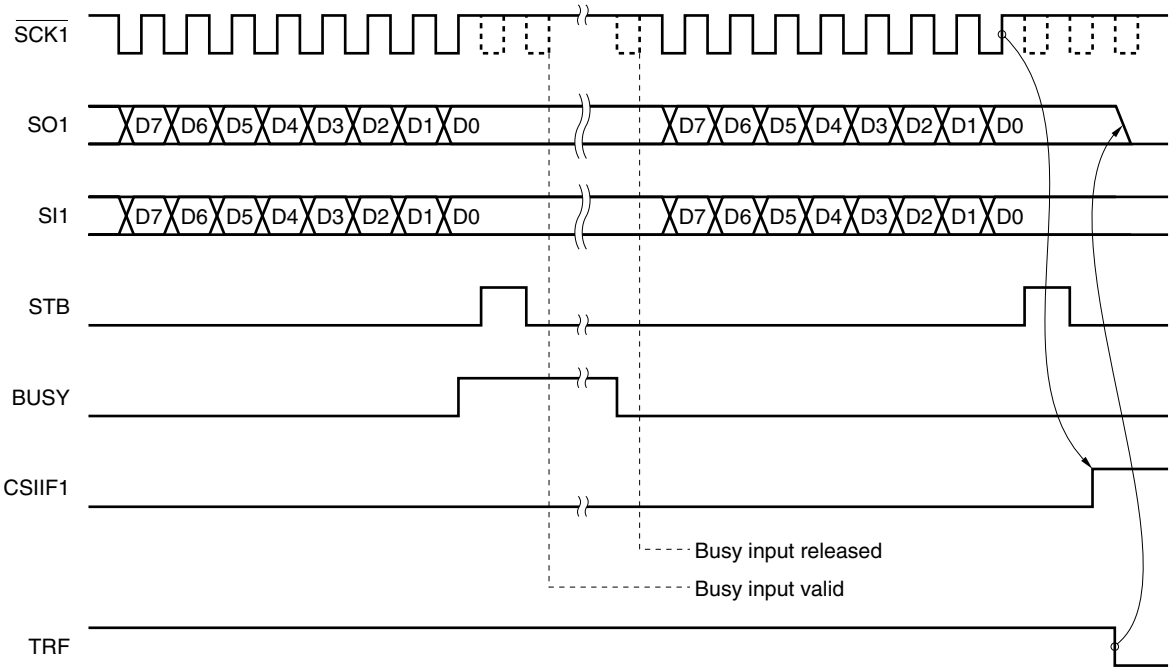
The active level of the busy signal is set by bit 0 (BUSY0) of ADTC.

BUSY0 = 0: Active high
BUSY1 = 1: Active low

Figure 14-22 shows the operation timing when the busy & strobe control option is used.

When the strobe control option is used, the interrupt request flag (CSIIF1) that is set on completion of transmission/reception is set after the strobe signal is output.

Figure 14-22. Operation Timing When Busy & Strobe Control Option Is Used (When BUSY0 = 0)



Caution When TRF is cleared, the SO1 pin goes low.

Remark CSIIF1: Interrupt request flag
TRF: Bit 3 of the automatic data transmit/receive control register (ADTC)

(d) Bit shift detection by busy signal

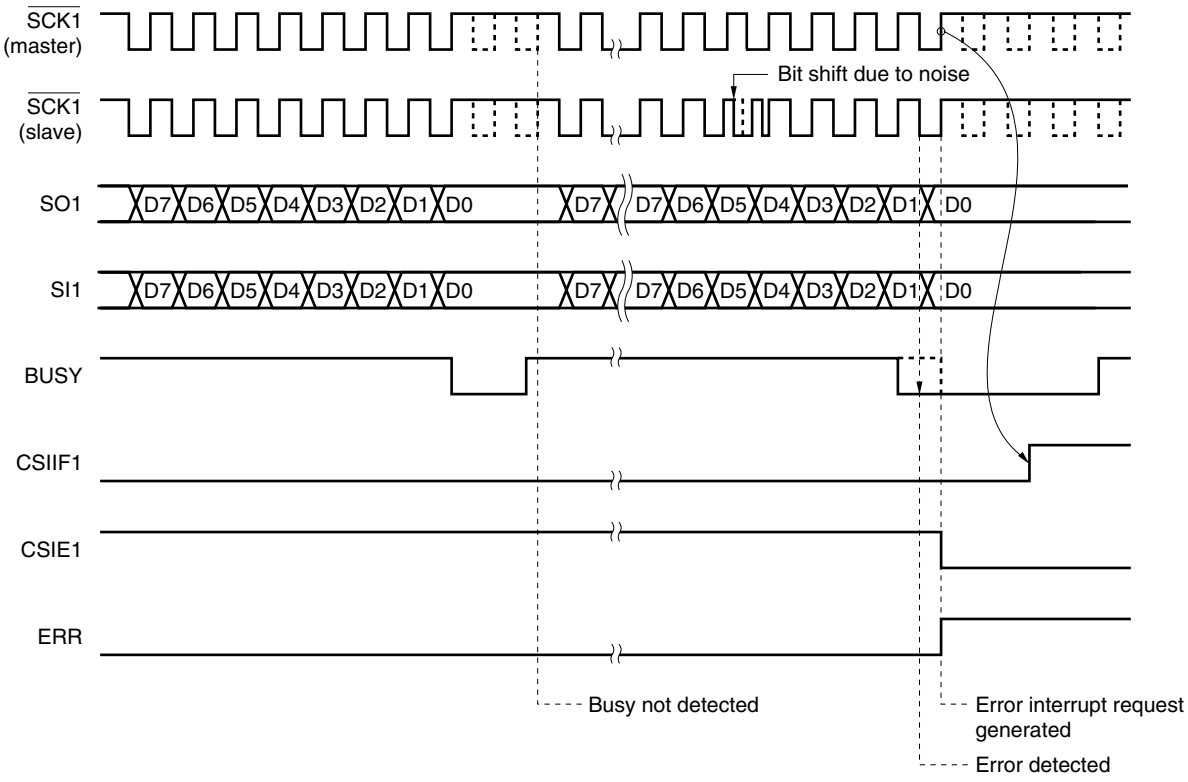
During automatic transmission/reception, a bit shift of the serial clock of the slave device may occur because noise is superimposed on the serial clock signal output by the master device. Unless the strobe control option is used at this time, the bit shift affects transmission of the next byte. In this case, the master can detect the bit shift by checking the busy signal during transmission by using the busy control option. A bit shift is detected by using the busy signal as follows.

The slave outputs the busy signal after the rising of the eighth serial clock during data transmission/reception (to not keep transmission/reception waiting by the busy signal at this time, make the busy signal inactive within 2 clocks).

The master samples the busy signal in synchronization with the falling of the leading side of the serial clock. If a bit shift has not occurred, all the eight serial clocks that have been sampled are inactive. If the sampled serial clocks are active, it is assumed that a bit shift has occurred, and error processing is executed (by setting bit 4 (ERR) of the automatic transmit/receive control register (ADTC) to 1).

Figure 14-23 shows the operation timing of the bit shift detection function by the busy signal.

Figure 14-23. Operation Timing of Bit Shift Detection Function by Busy Signal (When BUSY0 = 1)



CSIIF1: Interrupt request flag

CSIE1: Bit 7 of serial operating mode register 1 (CSIM1)

ERR: Bit 4 of the automatic data transmit/receive control register (ADTC)

(5) Timing of interrupt request signal generation

The interrupt signal is generated in synchronization with the timing shown in Table 14-2.

Table 14-2. Timing of Interrupt Request Signal Generation

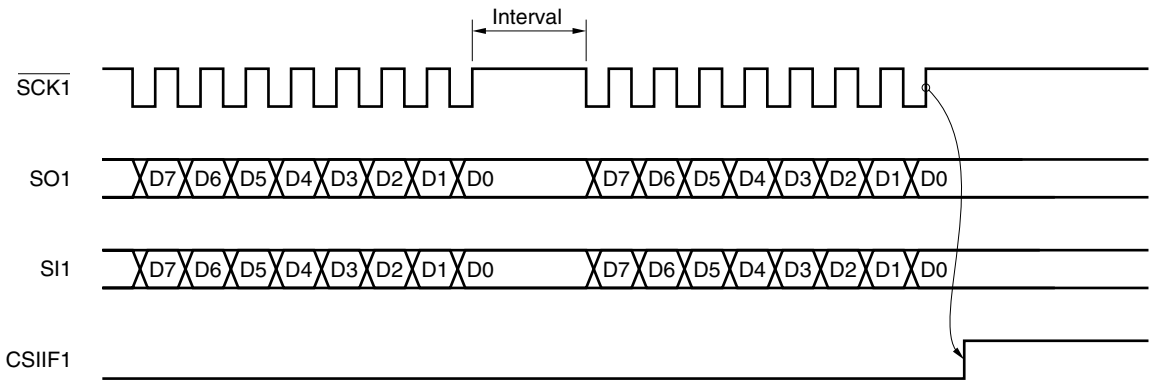
Operating Mode		Timing of Interrupt Request Signal
Single mode	Master mode	10th serial clock at end of transfer
	Slave mode	8th serial clock at end of transfer
Repeat transmit mode		Not generated
If bit shift occurs during transmission/reception		8th serial clock

(6) Interval time of automatic transmission/reception

Because read/write to/from the buffer RAM using the automatic transmit/receive function is performed asynchronously to the CPU processing, the interval time is dependent on the CPU processing at the timing of the eighth rising of the serial clock and the set value of the automatic data transmit/receive interval specification register (ADTI). Whether the interval time is dependent on ADTI is selected by setting bit 7 (ADTI7) of ADTI. If ADTI7 is reset to 0, the interval time is 2/f_{SCK}. If ADTI7 is set to 1, the interval time determined by the set contents of ADTI or the interval time by the CPU processing is selected, whichever is greater. Figure 14-24 shows the interval time of automatic transmission/reception.

Remark f_{SCK}: Serial clock frequency

Figure 14-24. Interval Time of Automatic Transmission/Reception



The following expression must be satisfied to access the buffer RAM.

1 transfer cycle ≤ Read access + Write access + CPU buffer RAM access

In the case of a “high-speed CPU & low-speed SCK[Note]”, the interval time is not necessary. In the case of a “low-speed CPU & high-speed SCK[Note]”, the interval time is necessary.
In this case, make sure that a sufficient interval time elapses, by using the automatic data transmit/receive interval specification register (ADTI), so that the above expression is satisfied.

Note The speeds of the CPU clock and SCK differ depending on the type of CPU core.

15.1 Function of Serial Interface SIO3

Serial interface SIO3 has the following two modes.

(1) Operation stop mode

This mode is used when serial transfer is not performed. For details, refer to **15.4.1**.

(2) 3-wire serial I/O mode (MSB-first)

In this mode, 8-bit data is transferred by using three lines: a serial clock ($\overline{SCK3}$), serial output (SO3), and serial input (SI3).

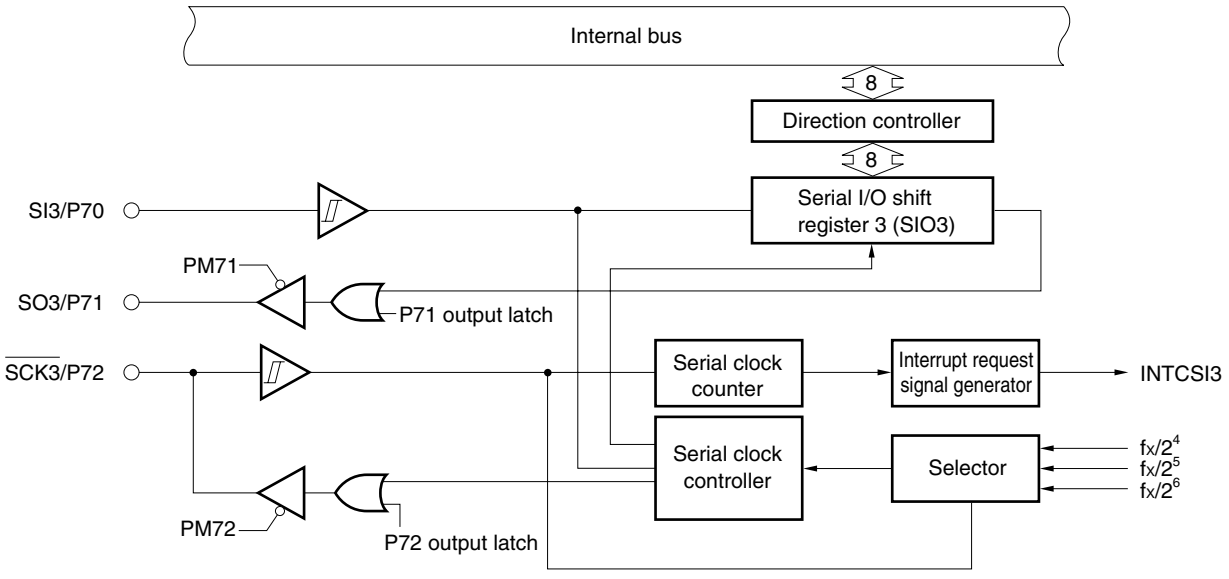
Because transmission and reception can be executed simultaneously in this mode, the processing time of data transfer can be shortened.

The first bit of the 8-bit data to be transferred is the MSB.

The 3-wire serial I/O mode is useful for connecting a peripheral I/O or display controller with a clocked serial interface. For details, refer to **15.4.2**.

Figure 15-1 shows the block diagram of serial interface SIO3.

Figure 15-1. Block Diagram of Serial Interface SIO3



15.2 Configuration of Serial Interface SIO3

The serial interface SIO3 consists of the following hardware.

Table 15-1. Configuration of Serial Interface SIO3

Item	Configuration
Register	Serial I/O shift register 3 (SIO3)
Control registers	Serial operating mode register 3 (CSIM3) Port mode register (PM7) Port 7 (P7)

(1) Serial I/O shift register 3 (SIO3)

This 8-bit register converts parallel data into serial data and transmits or receives serial data (shift operation) in synchronization with the serial clock.

SIO3 is set by an 8-bit memory manipulation instruction.

A serial operation is started by writing or reading data to or from SIO3 when bit 7 (CSIE3) of serial operating mode register 3 (CSIM3) is 1.

Data written to SIO3 is output to a serial output line (SO3) for transmission.

Data is read to SIO3 from a serial input line (SI3) for reception.

The value of this register is undefined after reset.

Caution Do not execute an access to SIO3 during transfer other than a transfer start trigger access (read operations are disabled when MODE0 = 0, and write operations are disabled when MODE0 = 1).

15.3 Registers Controlling Serial Interface SIO3

The following three registers control serial interface SIO3.

- **Serial operating mode register 3 (CSIM3)**
- **Port mode register 7 (PM7)**
- **Port 7 (P7)**

(1) Serial operating mode register 3 (CSIM3)

This register selects the serial clock of SIO3 and the operating mode, and enables or disables the operation.

CSIM3 is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Figure 15-2. Format of Serial Operating Mode Register 3 (CSIM3)

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM3	CSIE3	0	0	0	0	MODE	SCL31	SCL30	FF6FH	00H	R/W

CSIE3	Enable/disable of SIO3 operation		
	Shift register operation	Serial counter	Port
0	Operation disabled	Cleared	Port function[Note 1]
1	Operation enabled	Counter operation enabled	Serial function + port function[Note 2]

MODE	Transfer operating mode flag		
	Operating mode	Transfer start trigger	SO3 output
0	Transmit or transmit/receive mode	SIO3 write	Serial output
1	Receive only mode	SIO3 read	Fixed to low level[Note 3]

SCL31	SCL30	Selection of clock
0	0	External clock input to $\overline{SCK3}$
0	1	$f_X/2^4$ (394 kHz)
1	0	$f_X/2^5$ (197 kHz)
1	1	$f_X/2^6$ (98.4 kHz)

Notes 1. The SI3, SO3, and $\overline{SCK3}$ pins can be used as port pins when CSIE3 = 0 (when SIO3 operation is stopped).

2. When CSIE3 = 1 (when SIO3 operation is enabled), the SI3 pin can be used as a port pin if only the transmission function is used, and the SO3 pin can be used as a port pin in the receive mode.

3. This pin can be used as a port pin (P71).

Remarks 1. f_X: System clock oscillation frequency

2. ( ): f_X = 6.3 MHz

(2) Port mode register 7 (PM7)

PM7 is a register that sets input/output of port 7 in 1-bit units.

When using the P71/SO3 pin as a serial data output, set PM71 and the output latch of P71 to 0.

When using the P72/$\overline{\text{SCK3}}$ pin as a clock output, set PM72 and the output latch of P72 to 0.

When using the P70/SI3 pin as a serial data input and the P72/$\overline{\text{SCK3}}$ pin as a clock input, set PM70 and PM72 to 1. At this time, the output latches of P70 and P72 can be either 0 or 1.

PM7 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets the value of PM7 to FFH.

Figure 15-3. Format of Port Mode Register 7 (PM7)

Address: FF27H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70

PM7n	I/O mode selection of P7n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

15.4 Operation of Serial Interface SIO3

This section explains the two modes of serial interface SIO3.

15.4.1 Operation stop mode

In this mode, serial transfer is not performed.

The P70/SI3, P71/SO3, and P72/$\overline{SCK3}$ pins can be used as ordinary I/O port pins.

(1) Register setting

The operation stop mode is set by using serial operating mode register 3 (CSIM3).

CSIM3 is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM3	CSIE3	0	0	0	0	MODE	SCL31	SCL30	FF6FH	00H	R/W

CSIE3	Enable/disable of SIO3 operation		
	Shift register operation	Serial counter	Port
0	Operation disabled	Cleared	Port function[Note 1]
1	Operation enabled	Count operation enabled	Serial function + port function[Note 2]

Notes 1. The SI3, SO3, and $\overline{SCK3}$ pins can be used as port pins when CSIE3 = 0 (when SIO3 operation is stopped).

2. When CSIE3 = 1 (when SIO3 operation is enabled), the SI3 pin can be used as a port pin if only the transmission function is used, and the SO3 pin can be used as a port pin in the receive mode.

15.4.2 3-wire serial I/O mode

The 3-wire serial I/O mode is useful for connecting a peripheral I/O or display controller equipped with a clocked serial interface.

In this mode, communication is executed by using three lines: a serial clock ($\overline{SCK3}$), serial output (SO3), and serial input (SI3).

(1) Register setting

The 3-wire serial I/O mode is set by using serial operating mode register 3 (CSIM3).

(a) Serial operation mode register 3 (CSIM3)

This register is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM3	CSIE3	0	0	0	0	MODE	SCL31	SCL30	FF6FH	00H	R/W

CSIE3	Enable/disable of SIO3 operation		
	Shift register operation	Serial counter	Port
0	Operation disabled	Cleared	Port function[Note 1]
1	Operation enabled	Counter operatio enabled	Serial function + port function[Note 2]

MODE	Transfer operation mode flag		
	Operating mode	Transfer start trigger	SO3 output
0	Transmit or transmit/receive mode	SIO3 write	Serial output
1	Receive-only mode	SIO3 read	Fixed to low level[Note 3]

SCL31	SCL30	Selection of clock
0	0	External clock input to $\overline{SCK3}$
0	1	$f_X/2^4$ (394 kHz)
1	0	$f_X/2^5$ (197 kHz)
1	1	$f_X/2^6$ (98.4 kHz)

Notes 1. The SI3, SO3, and $\overline{SCK3}$ pins can be used as port pins when CSIE3 = 0 (when SIO3 operation is stopped).

2. When CSIE3 = 1 (when SIO3 operation is enabled), the SI3 pin can be used as a port pin if only the transmission function is used, and the SO3 pin can be used as a port pin in the receive mode.

3. This pin can be used as a port pin (P71).

Remarks 1. f_X: System clock oscillation frequency

2. ( ): f_X = 6.3 MHz

(b) Port mode register 7 (PM7)

PM7 is a register that sets input/output of port 7 in 1-bit units.

When using the P71/SO3 pin as a serial data output, set PM71 and the output latch of P71 to 0.

When using the P72/$\overline{\text{SCK3}}$ pin as a clock output, set PM72 and the output latch of P72 to 0.

When using the P70/SI3 pin as a serial data input and the P72/$\overline{\text{SCK3}}$ pin as a clock input, set PM70 and PM72 to 1. At this time, the output latches of P70 and P72 can be either 0 or 1.

PM7 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets the value of PM7 to FFH.

Address: FF27H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM7	PM77	PM76	PM75	PM74	PM73	PM72	PM71	PM70

PM7n	I/O mode selection of P7n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

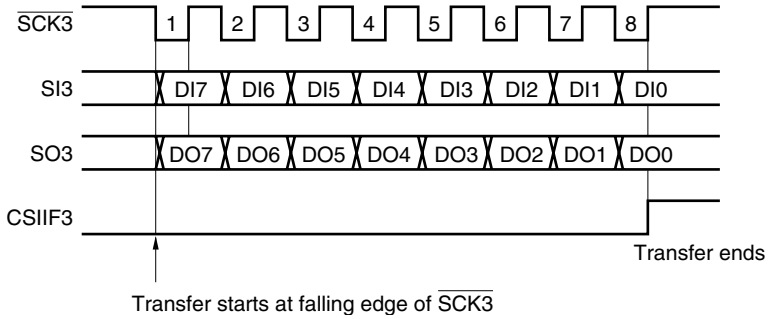
(2) Communication operation

In the 3-wire serial I/O mode, data is transmitted or received in 8-bit units. Data is transmitted or received in synchronization with the serial clock.

The shift operation of serial I/O shift register 3 (SIO3) is performed at the falling edge of the serial clock ($\overline{SCK3}$). The transmit data is held in SO3 and is output from the SO3 pin. The receive data input to the SI3 pin is latched to SIO3 at the falling edge of the serial clock.

When 8-bit data has been transferred, the operation of SIO3 is automatically stopped, and the interrupt request flag (CSIIF3) is set.

Figure 15-4. Timing in 3-Wire Serial I/O Mode



(3) Starting transfer

Serial transfer is started by writing (or reading) transfer data to serial I/O shift register 3 (SIO3) when the following conditions are satisfied.

- Operation control bit of SIO3 (bit 7 (CSIE3) of serial operation mode register 3 (CSIM3)) = 1
- If the internal serial clock is stopped or $\overline{SCK3}$ is high after transfer of 8-bit serial data
- Transmit/receive mode
  Transfer is started if SIO3 is written when bit 7 (CSIE3) of CSIM3 = 1, and bit 2 (MODE) = 0
- Receive mode
  Transfer is started if SIO3 is read when bit 7 (CSIE3) of CSIM3 = 1, and bit 2 (MODE) = 1

Caution Serial transfer is not started even if 1 is written to CSIE3 after data is written to SIO3.

On completion of transfer of the 8-bit data, serial transfer is automatically stopped, and the interrupt request flag (CSIIF3) is set.

16.1 Functions of Serial Interface UART0

Serial interface UART0 has the following two modes.

(1) Operation stop mode

In this mode, serial transfer is not performed.

For details, refer to **16.4.1**.

(2) Asynchronous serial interface (UART) mode

This mode is used to transmit or receive 1-byte data following a start bit. Full duplex operation can be executed in this mode.

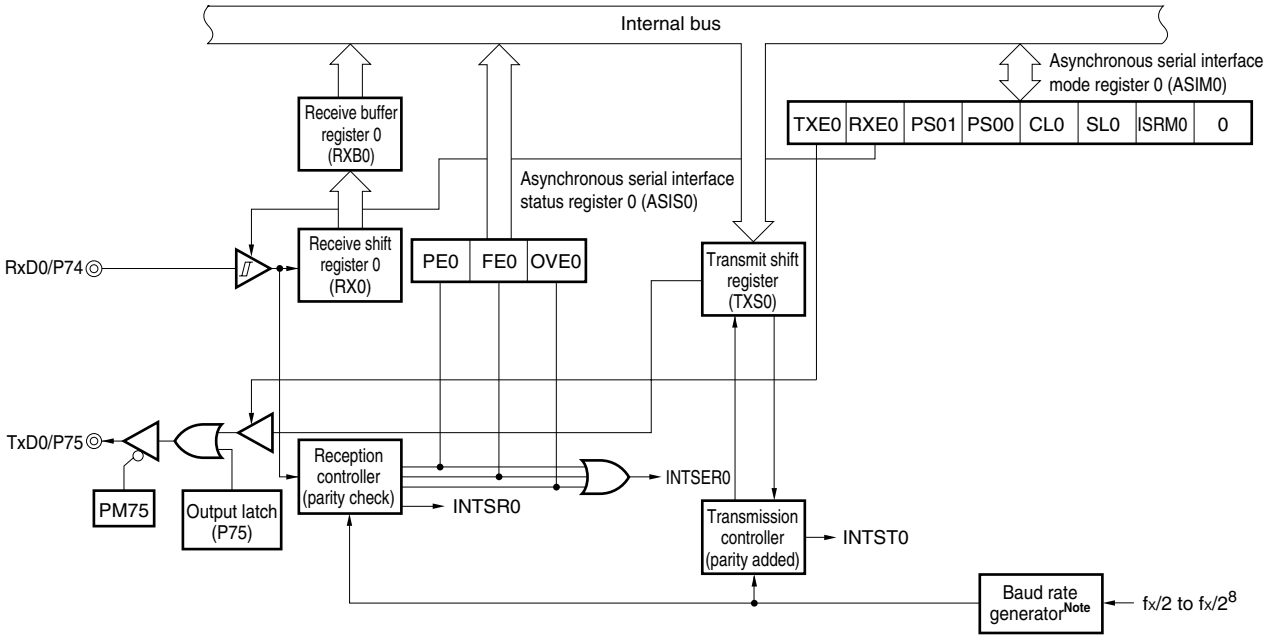
Because a UART-dedicated baud rate generator is provided, communication can be executed at a wide range of baud rates.

Moreover, MIDI standard baud rate (31.25 kbps) can also be generated by using the UART-dedicated baud rate generator.

For details, refer to **16.4.2**.

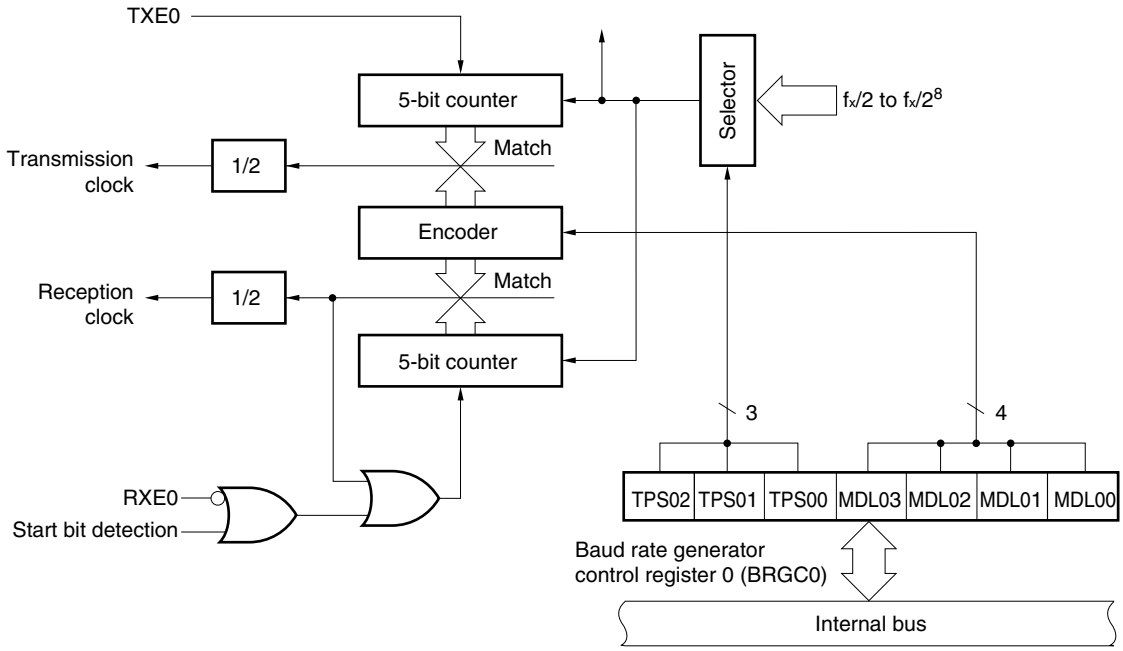
Figure 16-1 shows the block diagram of serial interface UART0.

Figure 16-1. Block Diagram of Serial Interface UART0



Note For the configuration of the baud rate generator, refer to Figure 16-2.

★ **Figure 16-2. Block Diagram of Baud Rate Generator**



Remark TXED: Bit 7 of asynchronous serial interface mode register 0 (ASIM0)
RXED: Bit 6 of asynchronous serial interface mode register 0 (ASIM0)

16.2 Configuration of Serial Interface UART0

Serial interface UART0 consists of the following hardware.

Table 16-1. Configuration of Serial Interface UART0

Item	Configuration
Registers	Transmit shift register 0 (TXS0) Receive shift register 0 (RX0) Receive buffer register 0 (RXB0)
Control registers	Asynchronous serial interface mode register 0 (ASIM0) Asynchronous serial interface status register 0 (ASIS0) Baud rate generator control register 0 (BRGC0) Port mode register 7 (PM7) Port 7 (P7)

(1) Transmit shift register 0 (TXS0)

This register stores transmit data. The data written to TXS0 is transmitted as serial data.

When the data length is specified to be 7 bits, bits 0 to 6 of the data written to TXS0 are transferred as transmit data. The transmission operation is started by writing data to TXS0.

TXS0 is written by an 8-bit memory manipulation instruction, but it cannot be read.

The value of this register is set to FFH after reset.

Caution Do not write data to TXS0 during transmission.
Because TXS0 and receive buffer register 0 (RXB0) are assigned to the same address, if this address is read, the value of RXB0 is read.

(2) Receive shift register 0 (RX0)

This register converts the serial data input to the RxD0 pin into parallel data. When 1 byte of data has been received, the receive data is transferred to receive buffer register 0 (RXB0).

RXB0 cannot be directly manipulated by software.

(3) Receive buffer register 0 (RXB0)

This register holds receive data. Each time 1 byte of data has been received, new receive data is transferred from receive shift register 0 (RX0).

If the data length is specified to be 7 bits, the receive data is transferred to bits 0 to 6 of RXB0, and the MSB of RXB0 is always 0.

RXB0 can be read by an 8-bit memory manipulation instruction, but it cannot be written.

The value of this register is set to FFH after reset.

Caution Because RXB0 and transmit shift register 0 (TXS0) are assigned to the same address, if data is written to this address, the value is written to TXS0.

(4) Transmission controller

This circuit controls the transmission operation by appending a start bit, parity bit, and stop bit to the data written to transmit shift register 0 (TXS0) according to the contents of asynchronous serial interface mode register 0 (ASIM0).

(5) Reception controller

This circuit controls the reception operation according to the contents of asynchronous serial interface mode register 0 (ASIM0). It also checks for errors such as a parity error during reception and, if an error is detected, writes a value identifying the error to asynchronous serial interface status register 0 (ASIS0).

16.3 Registers Controlling Serial Interface UART0

The following five registers control serial interface UART0.

- Asynchronous serial interface mode register 0 (ASIM0)
- Asynchronous serial interface status register 0 (ASIS0)
- Baud rate generator control register 0 (BRGC0)
- Port mode register 7 (PM7)
- Port 7 (P7)

(1) Asynchronous serial interface mode register 0 (ASIM0)

This 8-bit register controls the serial transfer operation of serial interface UART0.

ASIM0 is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Figure 16-3 shows the format of ASIM0.

Figure 16-3. Format of Asynchronous Serial Interface Mode Register 0 (ASIM0)

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0 Note	FF5AH	00H	R/W

TXE0	RXE0	Operating mode	Function of RXD0/P74 pin	Function of TXD0/P75 pin
0	0	Operation stop	Port function (P74)	Port function (P75)
0	1	UART mode (reception only)	Serial function (RXD0)	
1	0	UART mode (transmission only)	Port function (P74)	Serial function (TXD0)
1	1	UART mode (transmission/reception)	Serial function (RXD0)	

PS01	PS00	Specification of parity bit
0	0	No parity
0	1	0 parity always appended during transmission. Parity not checked during reception (parity error does not occur).
1	0	Odd parity
1	1	Even parity

CL0	Specification of character length
0	7 bits
1	8 bits

SL0	Specification of stop bit length of transmit data
0	1 bit
1	2 bits

ISRM0	Control of reception completion interrupt in case of error
0	Reception completion interrupt request generated in case of error
1	Reception completion interrupt request not generated in case of error

Note Be sure to reset bit 0 of ASIM0 to 0.

Cautions 1. Before changing the operation mode, stop the serial transmission/reception operation.

2. Set port mode register 7 (PM7) and the output latch of port 7 (P7) as follows in the UART mode.

Reset the output latch to 0.

- **During reception**
  Set the P74/RXD0 pin to the input mode (PM74 = 1).
- **During transmission**
  Set the P75/TXD0 pin to the output mode (PM75 = 0, P75 = 0).
- **During transmission/reception**
  Set P74 to the input mode and P75 to the output mode.

(2) Asynchronous serial interface status register 0 (ASIS0)

This register indicates the type of error when a reception error occurs in the UART mode.

ASIS0 is set by an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Figure 16-4. Format of Asynchronous Serial Interface Register 0 (ASIS0)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ASIS0	0	0	0	0	0	PE0	FE0	OVE0	FF5BH	00H	R

PE0	Parity error flag
0	No parity error
1	Parity error occurred (if parity of transmit data did not match).

FE0	Framing error flag
0	No framing error
1	Framing error occurred[Note 1] (if stop bit was not detected).

OVE0	Overrun error flag
0	No overrun error
1	Overrun error occurred[Note 2] (if next reception operation was completed before data was read from receive buffer register 0).

Notes 1. Even when the stop bit length is set to 2 bits by bit 2 (SL0) of asynchronous serial interface mode register 0 (ASIM0), only 1 stop bit is detected during reception.

2. Be sure to read receive buffer register 0 (RXB0) if an overrun error has occurred. Until RXB0 is read, the overrun error will persistently occur each time data is received.

(3) Baud rate generator control register 0 (BRGC0)

This register sets the serial clock of serial interface UART0.

BRGC0 is set by an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Figure 16-5 shows the format of BRGC0.

Figure 16-5. Format of Baud Rate Generator Control Register 0 (BRGC0)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00	FF5CH	00H	R/W

TPS02	TPS01	TSP00	Selection of source clock of 5-bit counter	n
0	0	0	$f_X/2$ (3.15 MHz)	1
0	0	1	$f_X/2^2$ (1.58 MHz)	2
0	1	0	$f_X/2^3$ (788 kHz)	3
0	1	1	$f_X/2^4$ (394 kHz)	4
1	0	0	$f_X/2^5$ (197 kHz)	5
1	0	1	$f_X/2^6$ (98.4 kHz)	6
1	1	0	$f_X/2^7$ (49.2 kHz)	7
1	1	1	$f_X/2^8$ (24.6 kHz)	8

MDL03	MDL02	MDL01	MDL00	Selection of input clock of baud rate generator	k
0	0	0	0	$f_{SCK}/16$	0
0	0	0	1	$f_{SCK}/17$	1
0	0	1	0	$f_{SCK}/18$	2
0	0	1	1	$f_{SCK}/19$	3
0	1	0	0	$f_{SCK}/20$	4
0	1	0	1	$f_{SCK}/21$	5
0	1	1	0	$f_{SCK}/22$	6
0	1	1	1	$f_{SCK}/23$	7
1	0	0	0	$f_{SCK}/24$	8
1	0	0	1	$f_{SCK}/25$	9
1	0	1	0	$f_{SCK}/26$	10
1	0	1	1	$f_{SCK}/27$	11
1	1	0	0	$f_{SCK}/28$	12
1	1	0	1	$f_{SCK}/29$	13
1	1	1	0	$f_{SCK}/30$	14
1	1	1	1	Setting prohibited	—

Caution If data is written to BRGC0 during communication, the output of the baud rate generator is disturbed and communication cannot be executed normally. Therefore, do not write BRGC0 during communication.

Remark f_{SCK}: Source clock of 5-bit counter
n: Value set by TPS00 to TPS02 ($1 \leq n \leq 8$)
k: Value set by MDL00 to MDL03 ($0 \leq k \leq 14$)
( ): f_X = 6.3 MHz

16.4 Operation of Serial Interface UART0

This section explains the two modes of serial interface UART0.

16.4.1 Operation stop mode

In this mode, serial transfer is not performed, and the pins used for the serial interface can be used as ordinary port pins.

(1) Register setting

The operation stop mode is set using asynchronous serial interface mode register 0 (ASIM0).

ASIM0 is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0 Note	FF5AH	00H	R/W

TXE0	RXE0	Operating mode	Function of RXD0/P74 pin	Function of TXD0/P75 pin
0	0	Operation stop	Port function (P74)	Port function (P75)
0	1	UART mode (reception only)	Serial function (RXD0)	
1	0	UART mode (transmission only)	Port function (P74)	Serial function (TXD0)
1	1	UART mode (transmission/reception)	Serial function (RXD0)	

Note Be sure to reset bit 0 of ASIM0 to 0.

Caution Before changing the operating mode, stop the serial transmission/reception operation.

16.4.2 Asynchronous serial interface (UART) mode

This mode is used to transmit or receive 1-byte data following a start bit. Full duplex operation can be executed in this mode.

Because a UART-dedicated baud rate generator is provided, communication can be executed at a wide range of baud rates.

Moreover, the baud rate of the MIDI standard (31.25 kbps) can also be generated by using the UART-dedicated baud rate generator.

(1) Register setting

The UART mode is set using asynchronous serial interface mode register 0 (ASIM0), asynchronous serial interface status register 0 (ASIS0), baud rate generator control register 0 (BRGC0), port mode register 7 (PM7), and port 7 (P7).

(a) Asynchronous serial interface mode register 0 (ASIM0)

ASIM0 is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	0 Note	FF5AH	00H	R/W

TXE0	RXE0	Operating mode	Function of RXD0/P74 pin	Function of TXD0/P75 pin
0	0	Operation stop	Port function (P74)	Port function (P75)
0	1	UART mode (reception only)	Serial function (RXD0)	
1	0	UART mode (transmission only)	Port function (P74)	Serial function (TXD0)
1	1	UART mode (transmission/reception)	Serial function (RXD0)	

PS01	PS00	Specifion of parity bit
0	0	No parity
0	1	0 parity always appended during transmission. Parity not checked during reception (parity error does not occur).
1	0	Odd parity
1	1	Even parity

CL0	Specification of character length
0	7 bits
1	8 bits

SL0	Specification of stop bit length of transmit data
0	1 bit
1	2 bits

ISRM0	Control of reception completion interrupt in case of error
0	Reception completion interrupt request generated in case of error
1	Reception completion interrupt request not generated in case of error

Note Be sure to reset bit 0 of ASIM0 to 0.

Cautions 1. Before changing the operation mode, stop the serial transmission/reception operation.

2. Set port mode register 7 (PM7) and the output latch of port 7 (P7) as follows in the UART mode.

- **During reception**
  Set P74 (RXD0) to the input mode (PM74 = 1).
- **During transmission**
  Set P75 (TXD0) to the output mode (PM75 = 0, P75 = 0).
- **During transmission/reception**
  Set P74 to the input mode and P75 to the output mode.

(b) Asynchronous serial interface status register 0 (ASIS0)

ASIS0 is set by an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H after reset.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ASIS0	0	0	0	0	0	PE0	FE0	OVE0	FF5BH	00H	R

PE0	Parity error flag
0	No parity error
1	Parity error occurred (if parity of transmit data did not match).

FE0	Framing error flag
0	No framing error
1	Framing error occurred[Note 1] (if stop bit was not detected).

OVE0	Overrun error flag
0	No overrun error
1	Overrun error occurred[Note 2] (if next reception operation was completed before data was read from receive buffer register 0).

Notes 1. Even when the stop bit length is set to 2 bits by bit 2 (SL0) of asynchronous serial interface mode register 0 (ASIM0), only 1 stop bit is detected during reception.

2. Be sure to read receive buffer register 0 (RXB0) if an overrun error has occurred. Until RXB0 is read, the overrun error will persistently occur each time data is received.

(c) Baud rate generator control register 0 (BRGC0)

BRGC0 is set by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGC0 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00	FF5CH	00H	R/W

TPS02	TPS01	TPS00	Selection of source clock of 5-bit counter	n
0	0	0	$f_X/2$ (3.15 MHz)	1
0	0	1	$f_X/2^2$ (1.58 MHz)	2
0	1	0	$f_X/2^3$ (788 kHz)	3
0	1	1	$f_X/2^4$ (394 kHz)	4
1	0	0	$f_X/2^5$ (197 kHz)	5
1	0	1	$f_X/2^6$ (98.4 kHz)	6
1	1	0	$f_X/2^7$ (49.2 kHz)	7
1	1	1	$f_X/2^8$ (24.6 kHz)	8

MDL03	MDL02	MDL01	MDL00	Selection of input clock of baud rate generator	k
0	0	0	0	$f_{SCK}/16$	0
0	0	0	1	$f_{SCK}/17$	1
0	0	1	0	$f_{SCK}/18$	2
0	0	1	1	$f_{SCK}/19$	3
0	1	0	0	$f_{SCK}/20$	4
0	1	0	1	$f_{SCK}/21$	5
0	1	1	0	$f_{SCK}/22$	6
0	1	1	1	$f_{SCK}/23$	7
1	0	0	0	$f_{SCK}/24$	8
1	0	0	1	$f_{SCK}/25$	9
1	0	1	0	$f_{SCK}/26$	10
1	0	1	1	$f_{SCK}/27$	11
1	1	0	0	$f_{SCK}/28$	12
1	1	0	1	$f_{SCK}/29$	13
1	1	1	0	$f_{SCK}/30$	14
1	1	1	1	Setting prohibited	—

Caution If data is written to BRGC0 during communication, the output of the baud rate generator is disturbed and communication cannot be executed normally. Therefore, do not write BRGC0 during communication.

Remark f_{SCK}: Source clock of 5-bit counter

n: Value set by TPS00 to TPS02 ($1 \leq n \leq 8$)

k: Value set by MDL00 to MDL03 ($0 \leq k \leq 14$)

( ): fx = 6.3 MHz

(2) Baud rate

The baud rate can be calculated by the following expression.

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1}\ (k + 16)}\ [\text{bps}]$$

f_X: System clock oscillation frequency

n: Value set by TPS00 to TPS02 ($1 \leq n \leq 8$) (Refer to Figure 16-5.)

k: Value set by MDL00 to MDL03 ($0 \leq k \leq 14$) (Refer to Figure 16-5.)

Table 16-2. BRGC0 Set Value for Each Baud Rate

Target Baud Rate [bps]	f_X = 6.3 MHz		f_X = 4.5 MHz	
	BRGC0	Error (%)	BRGC0	Error (%)
300	—	—	7DH	1.02
600	75H	–2.34	6DH	1.02
1200	65H	–2.34	5DH	1.02
2400	55H	–2.34	4DH	1.02
4800	45H	–2.34	3DH	1.02
9600	35H	–2.34	2DH	1.02
19200	25H	–2.34	1DH	1.02
31250	19H	0.8	12H	0.0
38400	15H	–2.34	0DH	1.02

Remark Baud rate = $f_X/\{2^{n+1} \times (k + 16)\}$

f_X: System clock oscillation frequency

n: Value set by TPS00 to TPS02 ($1 \leq n \leq 8$)

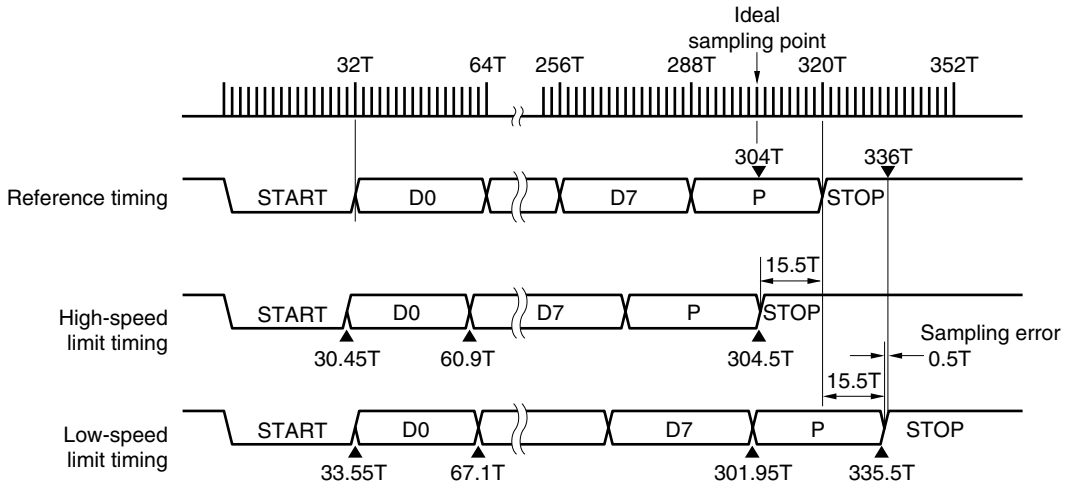
k: Value set by MDL00 to MDL03 ($0 \leq k \leq 14$)

- Permissible error range of baud rate

  The error of the baud rate depends on the number of bits in one frame and the division ratio of the 5-bit counter [1/(16 + k)].

  Figure 16-5 shows an example of the permissible error.

Figure 16-6. Permissible Error of Baud Rate Allowing for Sampling Error (k = 0)



$$\text{Permissible baud rate error (where K = 0)} = \frac{\pm 15.5}{320} \times 100 = 4.8438\ (\%)$$

★ **Caution The above permissible error value is the calculated value based on the ideal sampling point. When designing, take the error of the start bit detection time into consideration and add margins.**

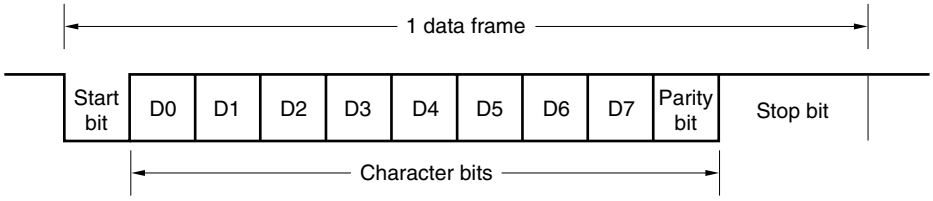
Remark T: Source clock cycle of 5-bit counter

(3) Basic operation

(a) Data format

Figure 16-7 shows the format of the transmit/receive data.

Figure 16-7. Format of Transmit/Receive Data of Asynchronous Serial Interface



One data frame consists of the following bits.

- Start bit ..... 1 bit
- Character bits .... 7 or 8 bits (LSB-first)
- Parity bit..... Even parity/odd parity/0 parity/no parity
- Stop bit ..... 1 or 2 bits

The character bit length, parity, and stop bit length of one data frame are selected by asynchronous serial interface mode register 0 (ASIM0).

If the character bit length is specified to be 7 bits, only the lower 7 bits (bits 0 to 6) are valid. The most significant bit (bit 7) is ignored during transmission, and it is “0” during reception.

The serial transfer rate is set by baud rate generator control register 0 (BGRC0).

If a receive error occurs in the serial data, the error can be identified by reading the status of asynchronous serial interface status register 0 (ASIS0).

(b) Type and operation of parity

A parity bit is used to detect a bit error in the communication data. Usually, the same type of parity bit is used on both the transmitter and receiver sides. With even parity or odd parity, an error of 1 bit (or odd-number error) can be detected. Errors cannot be detected when 0 parity or no parity is specified.

(i) Even parity

- **During transmission**

  The parity bit is set so that the number of bits in the transmit data, plus the parity bit, that are "1" is even. The value of the parity bit is as follows.

  If the number of bits in transmit data that are "1" is odd: 1
  If the number of bits in transmit data that are "1" is even: 0

- **During reception**

  The number of bits in the receive data, including the parity bit, that are "1" are counted. If the number of bits is odd, a parity error occurs.

(ii) Odd parity

- **During transmission**

  The parity bit is set so that the number of bits in the transmit data, plus the parity bit, that are "1" is odd. The value of the parity bit is as follows.

  If the number of bits in transmit data that are "1" is odd: 0
  If the number of bits in transmit data that are "1" is even: 1

- **During reception**

  The number of bits in the receive data, including the parity bit, that are "1" are counted. If the number of bits is even, a parity error occurs.

(iii) 0 parity

The parity bit is cleared to 0 during transmission, regardless of the transmit data.
The parity bit is not checked during reception. Therefore, a parity error does not occur regardless of whether the parity bit is 0 or 1.

(iv) No parity

A parity bit is not appended to the transmit data.
During reception, data is received assuming that it has no parity bit. Because no parity bit is used, parity errors do not occur.

(c) Transmission

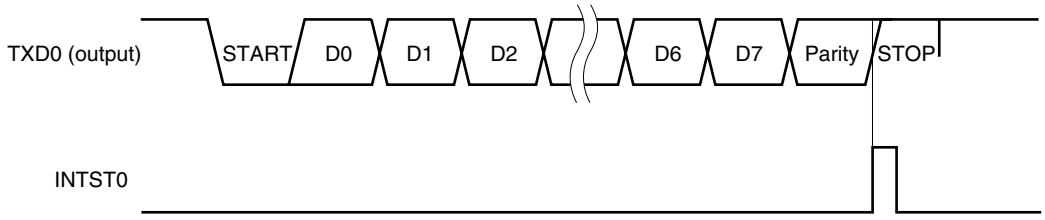
The transmission operation is started when transmit data is written to transmit shift register 0 (TXS0). The start bit, parity bit, and stop bit are automatically appended.

When the data in TXS0 is shifted out and TXS0 becomes empty as a result of starting transmission, a transmission completion interrupt request (INTST0) is generated.

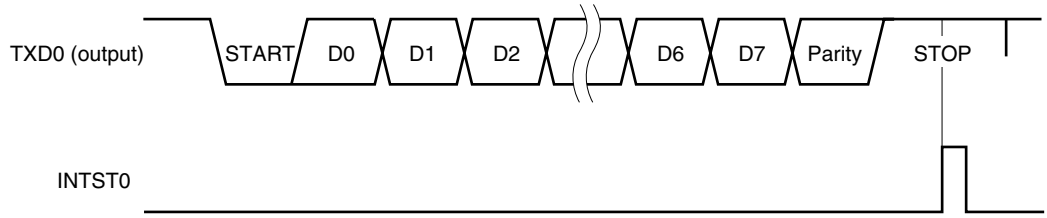
Figure 16-8 shows the timing of the transmission completion interrupt.

Figure 16-8. Generation Timing of Transmission Completion Interrupt Request of Asynchronous Serial Interface

(i) Stop bit length: 1



(ii) Stop bit length: 2



Caution Do not change the contents of asynchronous serial interface mode register 0 (ASIM0) during transmission. If the contents of the ASIM0 register are changed during transmission, subsequent transmissions may not be able to be performed (the normal status can be restored by $\overline{\text{RESET}}$ input).

Whether transmission is in progress can be checked by software, by using the transmission completion interrupt request (INTST0) or the interrupt request flag (STIF0) that is set by INTST0.

(d) Reception

Reception is enabled when bit 6 (RXE0) of asynchronous serial interface mode register 0 (ASIM0) is set to 1, and then the input to the RXD0 pin is sampled.

The RXD0 pin is sampled with the serial clock specified by ASIM0.

When the RXD0 pin goes low, the 5-bit counter of the baud rate generator starts counting. When a time of half the set baud rate has elapsed, the start timing signal of data sampling is output. If the RXD0 pin is sampled with this start timing signal again and is found to be low level, the pin level is recognized as a start bit. The 5-bit counter is initialized, counting is started, and the data is sampled. If character data, a parity bit, and 1 stop bit are detected after the start bit, reception of one frame of data is completed. When reception of one frame of data has been completed, the receive data in the shift register is transferred to receive buffer register 0 (RXB0), and a reception completion interrupt request (INTSR0) is generated.

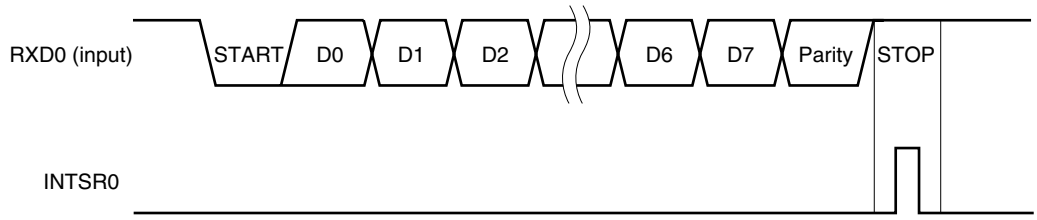
Even if an error occurs, the receive data in which the error occurred is transferred to RXB0. If bit 1 (ISRM0) of ASIM0 is cleared to 0 when the error occurred, INTSR0 is generated (refer to Figure 16-10).

INTSR0 is not generated if the ISRM0 bit is set to 1.

If the RXE0 bit is reset to 0 during reception, reception is immediately stopped. At this time, the contents of RXB0 and ASIS0 are not affected, nor are INTSR0 and INTSER0 generated.

Figure 16-9 shows the timing of generating the reception completion interrupt request of the asynchronous serial interface.

Figure 16-9. Timing of Generation of Reception Completion Interrupt of Asynchronous Serial Interface



Caution Be sure to read receive buffer register 0 (RXB0) even if a reception error occurs. Otherwise, an overrun error occurs when the next data is received, and the reception error status persists.

(e) Reception error

Three types of errors: a parity error, framing error, and overrun error, may occur during reception. If the error flag in asynchronous serial interface status register 0 (ASIS0) is set as a result of receiving data, the reception error interrupt request (INTSER0) is generated. This interrupt occurs before the reception completion interrupt request (INTSR0). Table 16-3 shows the causes of reception errors.

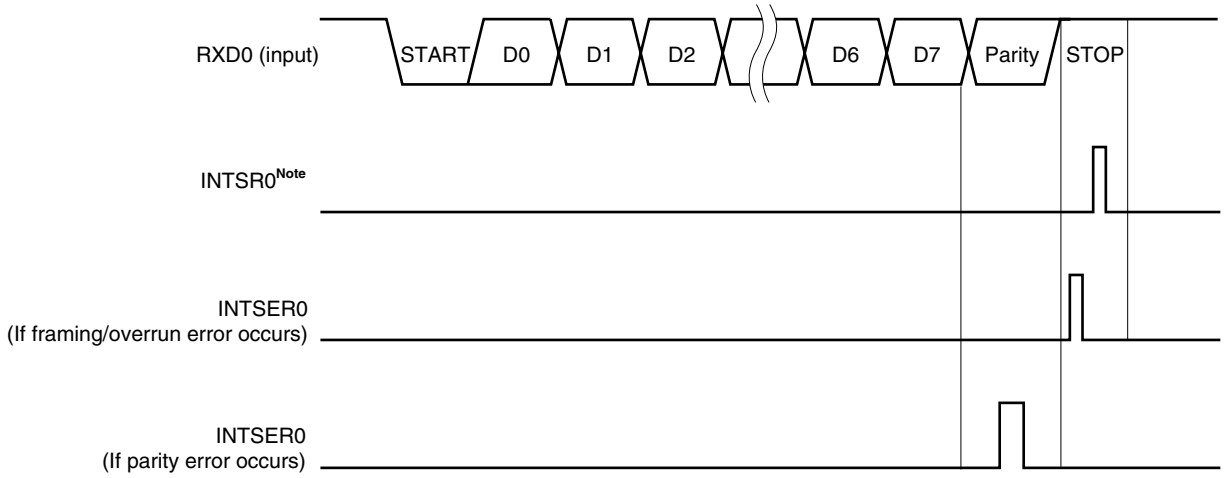
Which error has occurred during reception can be identified by reading the contents of ASIS0 in the reception error processing (INTSER0) (refer to Table 16-3 and Figure 16-10).

The contents of ASIS0 are reset when receive buffer register 0 (RXB0) is read or the next data is received (if an error occurs in the next data, the corresponding error flag is set).

Table 16-3. Causes of Reception Errors

Reception Error	Cause	Value of ASIS0
Parity error	Parity specified for transmission does not match parity of receive data.	04H
Framing error	Stop bit is not detected.	02H
Overrun error	Reception of next data is completed before data is read from receive buffer register 0.	01H

Figure 16-10. Reception Error Timing



Note If a reception error occurs while the ISRM0 bit is set to 1, INTSR0 does not occur.

Cautions 1. The contents of asynchronous serial interface status register 0 (ASIS0) are reset to 0 when receive buffer register 0 (RXB0) is read or the next data is received. To identify the error, be sure to read ASIS0 before reading RXB0.

2. Be sure to read receive buffer register 0 (RXB0) even if a reception error occurs. Otherwise, an overrun error occurs when the next data is received, and the reception error status persists.

CHAPTER 17 IEBus CONTROLLER (μPD178096A, 178098A, 178F098 ONLY)

The μPD178098A Subseries (μPD178096A, 178098A, 178F098) incorporates an IEBus controller.

IEBus (Inter Equipment Bus) is a small-scale digital data transfer system that transfers data between units. To implement IEBus by using the μPD178098A Subseries, an external IEBus driver and receiver are necessary because they are not provided.

The internal IEBus controller of the μPD178098A Subseries is of negative logic.

17.1 IEBus Controller Functions

17.1.1 Communication protocol of IEBus

The communication protocol of the IEBus is as follows.

(1) Multitask mode

All the units connected to the IEBus can transfer data to the other units.

(2) Broadcast communication function

Communication between one unit and multiple units can be performed as follows.

- Group-unit broadcast communication: Broadcast communication to group units
- All-unit broadcast communication: Broadcast communication to all units

(3) Effective transfer rate

The effective transfer rate is in mode 1 (the μPD178098A Subseries does not support modes 0 and 2 for the effective transfer rate).

- Mode 1: Approx. 18 Kbps (when f_X = 6.3 MHz)

Caution Different modes must not be mixed on one IEBus.

(4) Communication mode

Data transfer is executed in half-duplex asynchronous communication mode.

(5) Access control: CSMA/CD (Carrier Sense Multiple Access with Collision Detection)

The priority of the IEBus is as follows.

<1> Broadcast communication takes precedence over individual communication (communication from one unit to another).

<2> The lower master address takes precedence.

(6) Communication scale

The communication scale of IEBus is as follows.

- Number of units: 50 MAX.
- Cable length: 150 m MAX. (when twisted pair cable is used)

Caution The communication scale in an actual system differs depending on the characteristics of the cables, etc., constituting the IEBus driver/receiver and IEBus.

17.1.2 Determination of bus mastership (arbitration)

An operation to occupy the bus is performed when a unit connected to the IEBus controls the other units. This operation is called arbitration.

When two or more units simultaneously start transmission, arbitration is used to grant one of the units permission to occupy the bus.

Because only one unit is granted the bus mastership as a result of arbitration, the priority conditions of the bus are predetermined as follows.

Caution The bus mastership is released if communication is aborted.

(1) Priority by communication type

Broadcast communication (communication from one unit to multiple units) takes precedence over normal communication (communication from one unit to another).

(2) Priority by master address

If the communication type is the same, communication with the lower master address takes precedence.

A master address consists of 12 bits, with unit 000H having the highest priority and unit FFFH having the lowest priority.

17.1.3 Communication mode

Although the IEBus has three communication modes each having a different transfer rate, the μPD178098A Subseries supports only communication mode 1. The transfer rate and the maximum number of transfer bytes in one communication frame in communication mode 1 are as shown in Table 17-1.

Table 17-1. Transfer Rate and Maximum Number of Transfer Bytes in Communication Mode 1

Communication Mode	Maximum Number of Transfer Bytes (Bytes/Frame)	Effective Transfer Rate (Kbps)[Note]
1	32	Approx. 18

Note The effective transfer rate when the maximum number of transfer bytes is transmitted (when f_X = 6.3 MHz).

Select the communication mode (mode 1) for each unit connected to the IEBus before starting communication. If the communication mode of the master unit and that of the partner unit (slave unit) are not the same, communication is not executed correctly.

17.1.4 Communication address

With the IEBus, each unit is assigned a specific 12-bit address. This communication address consists of the following identification numbers.

- Higher 4 bits: Group number (number to identify the group to which each unit belongs)
- Lower 8 bits: Unit number (number to identify each unit in a group)

17.1.5 Broadcast communication

Normally, transmission or reception is performed between the master unit and its partner slave unit on a one-to-one basis. During broadcast communication, however, two or more slave units exist and the master unit executes transmission to these slave units. Because multiple slave units exist, the slave units do not return an acknowledge signal during communication.

Whether broadcast communication or normal communication is to be executed is selected by the broadcast bit (for this bit, refer to **17.1.6 (2) Broadcast bit**).

Broadcast communication is classified into two types: group-unit broadcast communication and all-unit broadcast communication. Group-unit broadcast and all-unit broadcast are identified by the value of the slave address (for the slave address, refer to **17.1.6 (4) Slave address field**).

(1) Group-unit broadcast communication

Broadcast communication is performed with the units in a group identified by the group number indicated by the higher 4 bits of the communication address.

(2) All-unit broadcast communication

Broadcast communication is performed with all the units, regardless of the value of the group number.

★

17.1.6 Transfer format of IEBus

Caution The logic on the IEBus I/O pin of the μPD178098A Subseries and the logic of the IEBus protocol (data on the IEBus) are inverted values. The following describes the case of the IEBus protocol.
<Example: Start bit>
• μPD178098A: High level
• IEBus protocol: Low level

Figure 17-1 shows the transfer signal format of the IEBus.

Figure 17-1. IEBus Transfer Signal Format

	Header		Master address field		Slave address field			Control field			Telegraph length field			Data field					
Frame format	Start bit	Broad-cast bit	Master address bit	P	Slave address bit	P	A	Control bit	P	A	Tele-graph length bit	P	A	Data bit	P	A	Data bit	P	A

Remarks 1. P: Parity bit, A: $\overline{\text{ACK}}$/NACK bit

2. The master station ignores the acknowledge bit during broadcast communication.

(1) Start bit

The start bit is a signal that informs the other units of the start of data transfer.

The unit that is to start data transfer outputs a low-level signal (start bit) for a specific time, and then starts outputting the broadcast bit.

If another unit has already output its start bit when one unit is to output the start bit, this unit does not output the start bit but waits for completion of output of the start bit by the other unit. When the output of the start bit by the other unit is complete, the unit starts outputting the broadcast bit in synchronization with the completion of the start bit output by the other unit.

The units other than the one that has started communication detect this start bit, and enter the reception status.

(2) Broadcast bit

This bit indicates whether the master selects one slave (individual communication) or multiple slaves (Broadcast communication) as the other party of communication.

When the broadcast bit is 0, it indicates broadcast communication; when it is 1, individual communication is indicated. broadcast communication is classified into two types: group-unit communication and all-unit communication. These communication types are identified by the value of the slave address (for the slave address, refer to **17.1.6 (4) Slave address field**).

Because two or more slave units exist in the case of broadcast communication, the acknowledge bit in each field subsequent to the master address field is not returned.

If two or more units start transmitting a communication frame at the same time, broadcast communication takes precedence over individual communication, and wins in arbitration.

If one unit occupies the bus as the master, the value set to the broadcast request flag (ALLRQ) of IEBus control register 0 (BCR0) is output.

(3) Master address field

The master address field is output by the master to inform a slave of the master's address.

The configuration of the master address field is as shown in Figure 17-2.

If two or more units start transmitting the broadcast bit at the same time, the master address field is used to make a judgment of arbitration.

The data output in the master address field is compared with the data on the bus each time one bit is output. If the master address output in the master address field is found to be different from the data on the bus as a result of comparison, it is assumed that the master has lost in arbitration. As a result, the master stops transmission and enters the reception status.

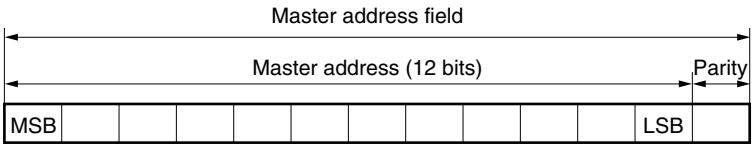
Because the IEBus is configured of wired AND, the unit having the minimum master address of the units participating in arbitration (arbitration masters) wins in arbitration.

After a 12-bit master address has been output, only one unit remains in the transmission status as the master unit.

Next, this master unit outputs a parity bit, determines the master address of other units, and starts outputting a slave address field.

If one unit occupies the bus as the master, the address set by the IEBus unit address register (UAR) is output.

Figure 17-2. Master Address Field



(4) Slave address field

The master outputs the address of the unit with which it is to communicate.

Figure 17-3 shows the configuration of the slave address field.

A parity bit is output after a 12-bit slave address has been transmitted in order to prevent a wrong slave address from being received by mistake. Next, the master unit detects an acknowledge signal from the slave unit to confirm that the slave unit exists on the bus. When the master has detected the acknowledge signal, it starts outputting the control field. During broadcast communication, however, the master does not detect the acknowledge bit but starts outputting the control field.

The slave unit outputs the acknowledge signal if its slave address matches and if the slave detects that the parities of both the master address and slave address are even. The slave unit judges that the master address or slave address has not been correctly received and does not output the acknowledge signal if the parities are odd. At this time, the master unit is in the standby (monitor) status, and communication ends.

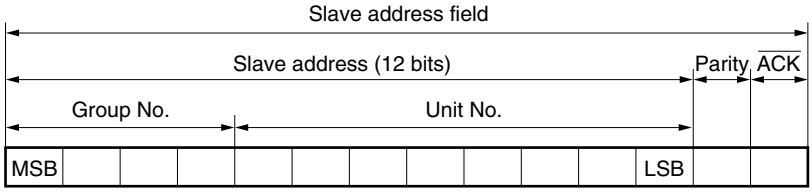
During broadcast communication, the slave address is used to identify group-unit broadcast or all-unit broadcast, as follows.

If slave address is FFFH: All-unit broadcast communication
If slave address is other than FFFH: Group-unit broadcast communication

Remark The group No. during group-unit broadcast communication is the value of the higher 4 bits of the slave address.

If one unit occupies the bus as the master, the address set by the IEBus slave address register (SAR) is output.

Figure 17-3. Slave Address Field



(5) Control field

The master outputs the operation it requires the slave to perform, by using this field.

The configuration of the control field is as shown in Figure 17-4.

If the parity following the control bit is even and if the slave unit can execute the function required by the master unit, the slave unit outputs an acknowledge signal and starts outputting the telegraph length field. If the slave unit cannot execute the function required by the master unit even if the parity is even, or if the parity is odd, the slave unit does not output the acknowledge signal, and returns to the standby (monitor) status.

The master unit starts outputting the telegraph field after confirming the acknowledge signal.

If the master cannot confirm the acknowledge signal, the master unit enters the standby status, and communication ends. During broadcast communication, however, the master unit does not confirm the acknowledge signal, and starts outputting the telegraph length field.

Table 17-2 shows the contents of the control bits.

Table 17-2. Contents of Control Bits

Bit 3[Note 1]	Bit 2	Bit 1	Bit 0	Function
0	0	0	0	Read slave status
0	0	0	1	Undefined
0	0	1	0	Undefined
0	0	1	1	Read data and lock[Note 2]
0	1	0	0	Read lock address (lower 8 bits)[Note 3]
0	1	0	1	Read lock address (higher 4 bits)[Note 3]
0	1	1	0	Read slave status and unlock[Note 2]
0	1	1	1	Read data
1	0	0	0	Undefined
1	0	0	1	Undefined
1	0	1	0	Write command and lock[Note 2]
1	0	1	1	Write data and lock[Note 2]
1	1	0	0	Undefined
1	1	0	1	Undefined
1	1	1	0	Write command
1	1	1	1	Write data

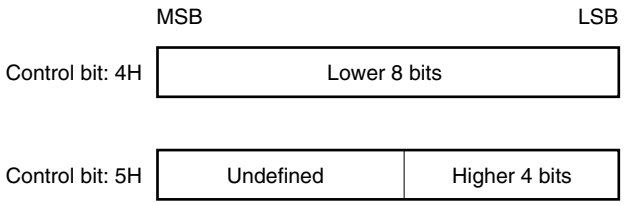
Notes 1. The telegraph length bit of the telegraph length field and data transfer direction of the data field change as follows depending on the value of bit 3 (MSB).

If bit 3 is '1': Transfer from master unit to slave unit

If bit 3 is '0': Transfer from slave unit to master unit

2. This is a control bit that specifies locking or unlocking (refer to **17.1.7 (4) Locking and unlocking**).

3. The lock address is transferred in 1-byte (8-bit) units and is configured as follows.



If the control bit received from the master unit is not as shown in Table 17-3, the unit locked by the master unit rejects acknowledging the control bit, and does not output the acknowledge bit.

Table 17-3. Control Field for Locked Slave Unit

Bit 3	Bit 2	Bit 1	Bit 0	Function
0	0	0	0	Read slave status
0	1	0	0	Read lock address (lower 8 bits)
0	1	0	1	Read lock address (higher 4 bits)

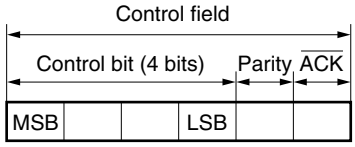
★ Moreover, units for which lock is not set by the master unit reject acknowledgment and do not output an acknowledge bit when the control data shown in Table 17-4 is acknowledged.

Table 17-4. Control Field for Unlocked Slave Unit

Bit 3	Bit 2	Bit 1	Bit 0	Function
0	1	0	0	Read lock address (lower 8 bits)
0	1	0	1	Read lock address (higher 4 bits)

If one unit occupies the bus as the master, the value set to the IEBus control data register (CDR) is output.

Figure 17-4. Control Field



★ **Table 17-5. Acknowledge Signal Output Conditions of Control Field**

(a) If received control data is AH, BH, EH, or FH

Communication Target (SLVRQ) Slave Specification = 1 No Specification = 0	Lock Status (LOCK) Lock = 1 Unlock = 0	Master Unit Identification (Match with PAR) Lock Request Unit = 1 Other = 0	Slave Transmission Enable (ENSLVTX)	Slave Reception Enable (ENSLVRX)	Received Control Data			
					AH	BH	EH	FH
1	0	don't care	don't care	1	√			
	1	1						
Other than above					×			

(b) If received control data is 0H, 3H, 4H, 5H, 6H, or 7H

Communication Target (SLVRQ) Slave Specification = 1 No Specification = 0	Lock Status (LOCK) Lock = 1 Unlock = 0	Master Unit Identification (Match with PAR) Lock Request Unit = 1 Other = 0	Slave Transmission Enable (ENSLVTX)	Slave Reception Enable (ENSLVRX)	Received Control Data					
					0H	3H	4H	5H	6H	7H
1	0	don't care	0	don't care	√	×	×	×	√	×
			1		√	√	×	×	√	√
	1	0	don't care		√	×	√	√	×	×
		1	0		√	×	√	√	√	×
			1		√	√	√	√	√	√
Other than above					×					

Caution If the received control data is other than the data shown in Table 17-5, × ($\overline{\text{ACK}}$ is not returned) is unconditionally assumed.

Remarks 1. √: $\overline{\text{ACK}}$ is returned.
×: $\overline{\text{ACK}}$ is not returned.

2. ENSLVTX: Bit 4 of IEBus control register 0 (BCR0)
ENSLVRX: Bit 3 of IEBus control register 0 (BCR0)
LOCK: Bit 2 of the IEBus unit status register (USR)
SLVRQ: Bit 6 of the IEBus unit status register (USR)
PAR: IEBus partner address register

(6) Telegraph length field

This field is output by the transmission side to inform the reception side of the number of bytes of the transmit data.

The configuration of the telegraph length field is as shown in Figure 17-5.

Table 17-6 shows the relationship between the telegraph length bit and the number of transmit data.

Figure 17-5. Telegraph Length Field

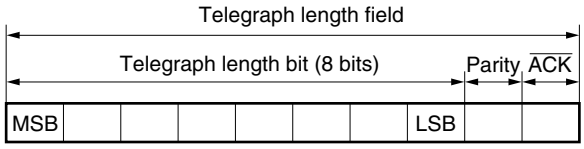


Table 17-6. Contents of Telegraph Length Bit

Telegraph Length Bit (Hex)	Number of Transmit Data Bytes
01H	1 byte
02H	2 bytes
FFH	255 bytes
00H	256 bytes

The operation of the telegraph length field differs depending on whether the master transmits data (when control bit 3 is 1) or receives data (when control bit 3 is 0).

(a) When master transmits data

The telegraph length bit and parity bit are output by the master unit. When the slave unit detects that the parity is even, it outputs the acknowledge signal, and starts outputting the data field. During broadcast communication, however, the slave unit does not output the acknowledge signal.

If the parity is odd, the slave unit judges that the telegraph length bit has not been correctly received, does not output the acknowledge signal, and returns to the standby (monitor) status. At this time, the master unit also returns to the standby status, and communication ends.

(b) When master receives data

The telegraph length bit and parity bit are output by the slave unit and the synchronization signals of bits are output by the master unit. If the master unit detects that the parity bit is even, it outputs the acknowledge signal.

If the parity bit is odd, the master unit judges that the telegraph length bit has not been correctly received, does not output the acknowledge signal, and returns to the standby (monitor) status. At this time, the slave unit also returns to the standby status, and communication ends.

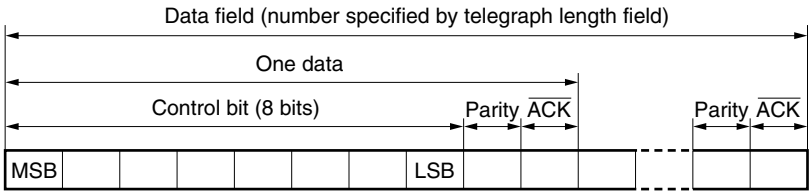
(7) Data field

This is data output by the transmission side.

The master unit transmits to or receives data from a slave unit by using the data field.

The configuration of the data field is as shown below.

Figure 17-6. Data Field



Following the data bit, the parity bit and acknowledge bit are respectively output by the master unit and slave unit.

Use broadcast communication only for when the master unit transmits data. At this time, the acknowledge bit is ignored.

The operation differs as follows depending on whether the master transmits or receives data.

(a) When master transmits data

When the master unit writes data to a slave unit, the master unit transmits the data bit and parity bit to the slave unit. If the parity is even and the receive data is not stored in the IEBus data register (DR) when the slave unit has received the data bit and parity bit, the slave unit outputs an acknowledge signal. If the parity is odd or if the receive data is stored in the IEBus data register (DR), the slave unit rejects receiving the data, and does not output the acknowledge signal.

If the slave unit does not output the acknowledge signal, the master unit transmits the same data again. This operation continues until the master detects the acknowledge signal from the slave unit, or the data exceeds the maximum number of transmit bytes.

If there is more data and the maximum number of transmit bytes is not exceeded when the parity is even and when the slave unit outputs the acknowledge signal, the master unit transmits the next data.

During broadcast communication, the slave unit does not output the acknowledge signal, and the master ★ unit transfers 1 byte of data at a time. If the parity is odd or the DR register is storing receive data after the slave unit has received the data bit and parity bit during broadcast communication, the slave unit judges that reception has not been performed correctly, and stops reception.

(b) When master receives data

When the master unit reads data from a slave unit, the master unit outputs a sync signal corresponding to all the read bits.

The slave unit outputs the contents of the data and parity bits to the bus in response to the sync signal from the master unit.

The master unit reads the data and parity bits output by the slave unit, and checks the parity.

If the parity is odd, or if the DR register is storing a receive data, the master unit rejects accepting the data, and does not output the acknowledge signal. If the maximum number of transmit bytes is within the value that can be transmitted in one communication frame, the master unit repeats reading the same data.

If the parity is even and the DR register is not storing a receive data, the master unit accepts the data and returns the acknowledge signal. If the maximum number of transmit bytes is within the value that can be transmitted in one frame, the master unit reads the next data.

★ **Caution Do not operate master reception in broadcast communication, because the slave unit cannot be defined and data transfer cannot be performed correctly.**

(8) Parity bit

The parity bit is used to check that if the transmit data has no error.

The parity bit is appended to each data of the master address, slave address, control, telegraph length, and data bits.

The parity bit is even parity. If the number of bits in data that are '1' is odd, the parity bit is '1'. If the number of bits in the data that are '1' is even, the parity bit is '0'.

(9) Acknowledge bit

During normal communication (communication from one unit to another), an acknowledge bit is appended to the following locations to check that the data has been correctly received.

- End of slave address field
- End of control field
- End of telegraph length field
- End of data field

The definition of the acknowledge bit is as follows.

- 0: Indicates that the transmit data is recognized ($\overline{\text{ACK}}$).
- 1: Indicates that the transmit data is not recognized (NACK).

During broadcast communication, however, the contents of the acknowledge bit are ignored.

(a) Acknowledge bit at end of slave address field

The acknowledge bit at the end of the slave address field serves as NACK in any of the following cases, and transmission is stopped.

- If the parity of the master address bit or slave address bit is incorrect
- If a timing error (error in bit format) occurs
- If a slave unit does not exist

(b) Acknowledge bit at end of control field

The acknowledge bit at the end of the control field serves as NACK in any of the following cases, and transmission is stopped.

- If the parity of the control bit is incorrect
- If control bit 3 is '1' (write operation) when the slave reception enable flag (ENSLVRX) is not set (1) (refer to **17.4.2 (1) IEBus control register 0 (BCR0)**)
- If the control bit indicates reading of data (3H or 7H) when the slave transmission enable flag (ENSLVTX) is not set (1) (refer to **17.4.2 (1) IEBus control register 0 (BCR0)**)
- If a unit other than the one that set locking requests control bits 3H, 6H, 7H, AH, BH, EH, or FH when locking is set
- If the control bit indicates reading of a lock address (4H or 5H) even when locking is not set
- If a timing error occurs
- If the control bit is undefined

★ **Cautions 1. Even when the slave transmission enable flag (ENSLVTX) is not set (1), $\overline{\text{ACK}}$ may be returned if control data is received (refer to Table 17-5).**

2. Even when the slave reception enable flag (ENSLVRX) is not set (1), NACK is always returned by the acknowledge bit in the control field if data/command writing control data is acknowledged.

Slave reception can be disabled (communication stopped) by the ENSLVRX flag only in the case of individual communication. In the case of broadcast communication, communication is maintained and the data request interrupt (INTIE1) or IEBus end interrupt (INTIE2) is generated.

(c) Acknowledge bit at end of telegraph length field

The acknowledge bit at the end of the telegraph length field serves as NACK in any of the following cases, and transmission is stopped.

- If the parity of the telegraph length bit is incorrect
- If a timing error occurs

(d) Acknowledge bit at end of data field

The acknowledge bit at the end of the data field serves as NACK in any of the following cases, and transmission is stopped.

- If the parity of the data bit is incorrect[Note]
- If a timing error occurs after the preceding acknowledge bit has been transmitted
- If receive data is stored in the IEBus data register (DR) and no more data can be received[Note]

★ **Note** In this case, when the communication executed is individual communication, if the maximum number of transmit bytes is within the value that can be transmitted in one frame, the transmission side executes transmission of that data field again. For broadcast communication, the transmission side does not execute transmission again, a communication error occurs on the reception side and reception stops.

17.1.7 Transfer data

(1) Slave status

The master unit can learn why the slave unit did not return the acknowledge bit ($\overline{ACK}$) by reading the slave status. The slave status is determined according to the result of the last communication the slave unit has executed. All the slave units can supply information on the slave status.

The configuration of the slave status is shown below.

Figure 17-7. Bit Configuration of Slave Status

MSB LSB

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Bit 0[Note 1]	Meaning
0	Transmit data is not written in IEBus data register (DR)
1	Transmit data is written in IEBus data register (DR)

Bit 1[Note 2]	Meaning
0	Receive data is not stored in IEBus data register (DR)
1	Receive data is stored in IEBus data register (DR)

Bit 2	Meaning
0	Unit is not locked
1	Unit is locked

Bit 3	Meaning
0	Fixed to 0

Bit 4[Note 3]	Meaning
0	Slave transmission is stopped
1	Slave transmission is ready

Bit 5	Meaning
0	Fixed to 0

Bit 7	Bit 6	Meaning	
0	0	Mode 0	Indicates the highest mode supported by unit[Note 4].
0	1	Mode 1	
1	0	Mode 2	
1	1	Not used	

Notes 1. After reset: Bit 0 is set to 1.

2. The receive buffer size is 1 byte.

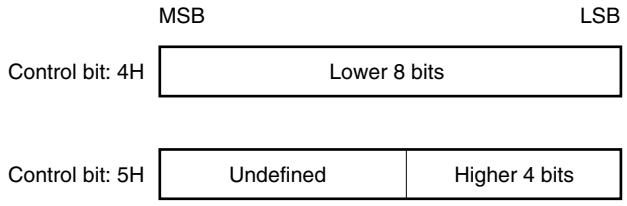
3. When the μPD178098A Subseries serves as a slave unit, this bit corresponds to the status indicated by bit 4 (ENSLVTX) of IEBus control register 0 (BCR0).

4. When the μPD178098A Subseries serves as a slave unit, bits 7 and 6 are fixed to '0' and '1' (mode 1), respectively.

(2) Lock address

When the lock address is read (control bit: 4H or 5H), the address (12 bits) of the master unit that has issued the lock instruction is configured in 1-byte units as shown below and read.

Figure 17-8. Configuration of Lock Address



(3) Data

If the control bit indicates reading of data (3H or 7H), the data in the data buffer of the slave unit is read by the master unit.

If the control bit indicates writing of data (BH or FH), the data received by the slave unit is processed according to the operation rule of that slave unit.

(4) Locking and unlocking

The lock function is used when a message is transferred in two or more communication frames.

The unit that is locked does not receive data from units other than the one that has locked the unit (either individual or broadcast communication).

A unit is locked or unlocked as follows.

(a) Locking

If the communication frame is completed without succeeding to transmit or receive data of the number of bytes specified by the telegraph length bit after the telegraph length field has been transmitted or received ($\overline{ACK}$ = 0) by the control bit that specifies locking (3H, AH, or BH), the slave unit is locked by the master unit. At this time, the bit (bit 2) in the byte indicating the slave status is set to '1'.

(b) Unlocking

After transmitting or receiving data of the number of data bytes specified by the telegraph length bit in one communication frame by the control bit that has specified locking (3H, AH, or BH), or the control bit that has specified unlocking (6H), the slave unit is unlocked by the master unit. At this time, the bit related to locking (bit 2) in the byte indicating the slave status is reset to '0'.

Locking or unlocking is not performed during broadcast communication.

The locking and unlocking conditions are shown below.

★ **(c) Lock setting conditions**

Control Data	Broadcast Communication		Individual Communication	
	Communication End	Frame End	Communication End	Frame End
3H, 6H[Note]			Not locked	Lock set
AH, BH	Not locked	Not locked	Not locked	Lock set
0H, 4H, 5H, EH, FH	Not locked	Not locked	Not locked	Not locked

★ **(d) Lock release conditions (while locked)**

Control Data	Broadcast Communication from Lock Request Unit		Individual Communication from Lock Request Unit	
	Communication End	Frame End	Communication End	Frame End
3H, 6H[Note]			Unlocked	Remains locked
AH, BH	Unlocked	Unlocked	Unlocked	Remains locked
0H, 4H, 5H, EH, FH	Remains locked	Remains locked	Remains locked	Remains locked

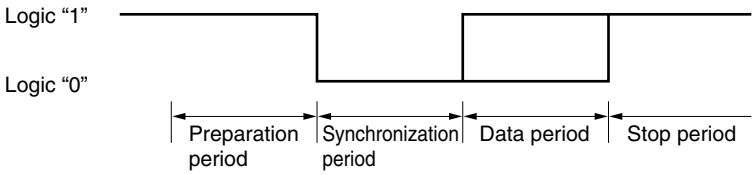
Note The frame end of control data 6H (slave status read/unlock) occurs when the parity in the data field is odd, and when the acknowledge signal from the IEBus unit is repeated up to the maximum number of transfer bytes without being output.

17.1.8 Bit format

★ **Caution The logic on the IEBus I/O pin of the μPD178098A Subseries and the logic of the IEBus protocol (data on the IEBus) are inverted values. The following describes the case of the IEBus protocol.**
<Example: Start bit>
• μPD178098A: High level
• IEBus protocol: Low level

The format of the bits constituting the communication frame of the IEBus is shown below.

Figure 17-9. Bit Format of IEBus



Preparation period: First low-level (logic “1”) period
Synchronization period: Next high-level (logic “0”) period
Data period: Period indicating value of bit
Stop period: Last low-level (logic “1”) period

The synchronization period and data period are almost equal to each other in length.

The IEBus synchronizes each bit. The specifications on the time of the entire bit and the time related to the period allocated to that bit differ depending on the type of the transmit bit, or whether the unit is the master unit or a slave unit.

★ The master and slave units monitor whether each period (preparation period, synchronization period, data period, and stop period) is output for specified time while they are communicating. If a period is not output for the specified time, the master and slave units report a timing error, immediately terminate communication and enter the standby status.

17.2 Simple IEBus Controller

The μPD178098A Subseries has a newly developed IEBus controller. The functions of this IEBus controller are limited compared with the conventional IEBus interface functions of the existing models (provided in the 78K/0 Series).

Table 17-7 compares the conventional IEBus interface functions of the existing models with the simple IEBus interface functions of the μPD178098A Subseries.

Table 17-7. Comparison of Existing and Simple IEBus Interface Functions

Item	Conventional Functions (IEBus of 78K/0)	Simple Version (IEBus Incorporated in μPD178098A Subseries)
Communication mode	Modes 0, 1, and 2	Fixed to mode 1
Internal system clock	fx = 6.0 (6.29) MHz	fx = 6.3 MHz[Note 1]
Internal buffer size	Transmit buffer: 33 bytes (FIFO) Receive buffer: 40 bytes (FIFO) Up to 4 frames can be received.	Transmit/receive data register
CPU processing	Communication start preprocessing (data setting) Setting and management of each communication status Writing data to transmit buffer Reading data from receive buffer	Communication start preprocessing (data setting) Setting and management of each communication status 1-byte data write processing 1-byte data read processing Management of transmission such as slave status Management of multiple frames, master request reprocessing
Hardware processing	Bit processing (modulation/demodulation, error detection) Field processing (generation/management) Arbitration result detection Parity processing (generation/error detection) Automatic return of $\overline{ACK}$/NACK Automatic data reprocessing Automatic master reprocessing[Note 2] Transmission processing such as automatic slave status transmission Multiple-frame reception processing	Bit processing (modulation/demodulation, error detection) Field processing (generation/management) Arbitration result detection Parity processing (generation/error detection) Automatic return of $\overline{ACK}$/NACK Automatic data transmission reprocessing

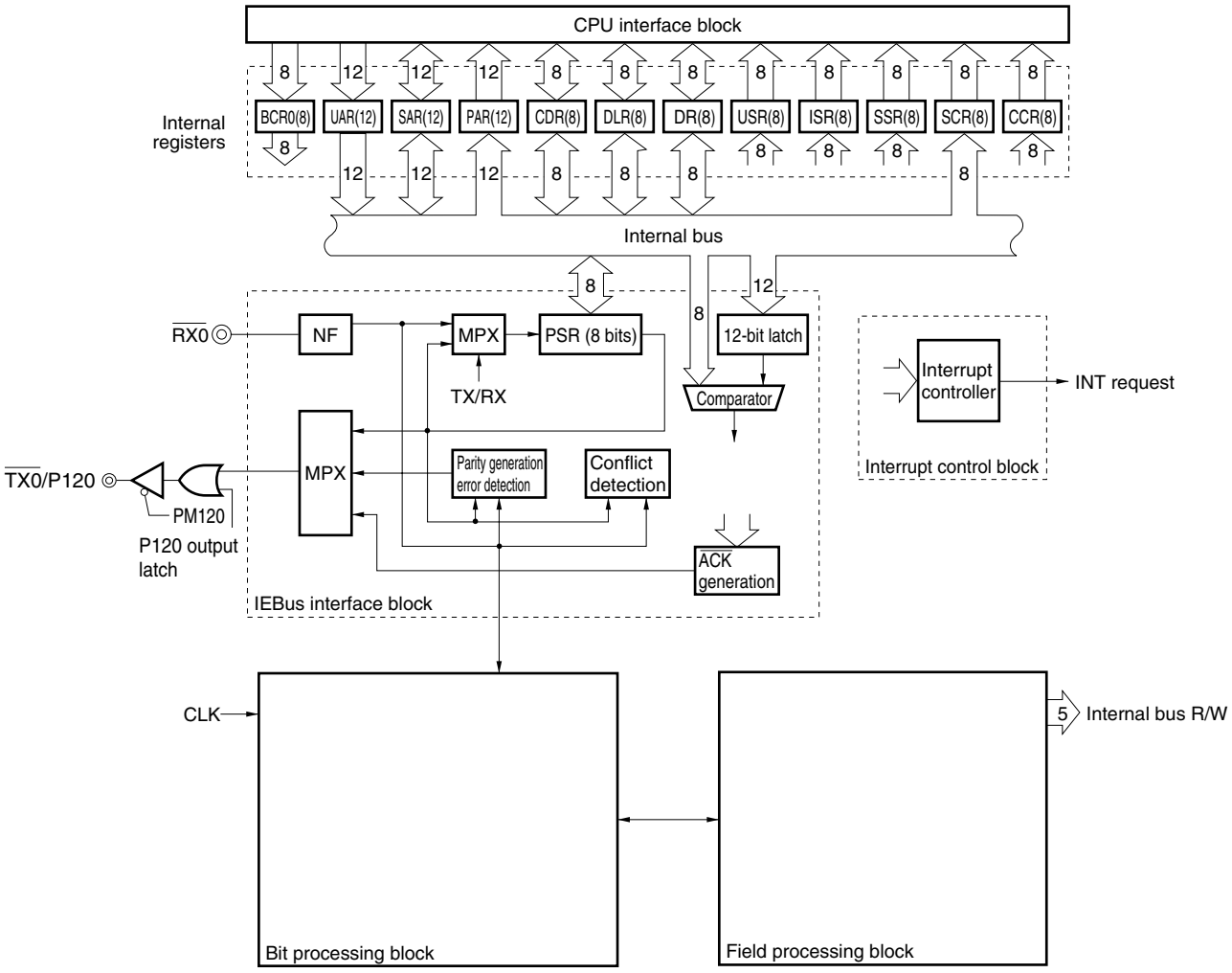
★ **Notes 1.** The μPD178098A Subseries supports the IEBus controller when fx = 6.3 MHz. When fx = 4.5 MHz, the IEBus controller is not supported.

2. Automatic master reprocessing: After generating the master request, if the master request is cancelled by arbitration, etc., the bus is released and the master automatically re-issues the master request.

17.3 IEBus Controller Configuration

The block diagram of the IEBus controller is shown below.

Figure 17-10. IEBus Controller Block Diagram



(1) Hardware configuration and functions

The IEBus mainly consists of the following six internal blocks.

- CPU interface block
- Interrupt control block
- Internal registers
- Bit processing block
- Field processing block
- IEBus interface block

(a) CPU interface block

This is a control block that interfaces between the CPU and the IEBus.

(b) Interrupt control block

This control block transfers interrupt request signals from the IEBus to the CPU.

(c) Internal registers

These registers set data to the control registers and fields that control the IEBus (for the internal registers, refer to **17.4 Internal Registers of IEBus Controller**).

(d) Bit processing block

This block generates and disassembles bit timing, and mainly consists of a bit sequence ROM, 8-bit preset timer, and comparator.

(e) Field processing block

This block generates each field in the communication frame, and mainly consists of a field sequence ROM, 4-bit down counter, and comparator.

(f) IEBus interface block

This is the interface block for an external driver/receiver, and mainly consists of a noise filter, shift register, conflict detector, parity detector, parity generator, and $\overline{\text{ACK}}$/NACK generator.

17.4 Internal Registers of IEBus Controller

17.4.1 Internal register list

Table 17-8. Internal Registers of IEBus Controller

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After
				1 Bit	8 Bits	16 Bits	Reset
FFB0H	IEBus control register 0	BCR0	R/W	√	√	–	00H
FFB2H	IEBus unit address register	UAR		–	–	√	0000H
FFB4H	IEBus slave address register	SAR		–	–	√	
FFB6H	IEBus partner address register	PAR	R	–	–	√	
FFB8H	IEBus control data register	CDR	R/W	–	√	–	01H
FFB9H	IEBus telegraph length register	DLR		–	√	–	
FFBAH	IEBus data register	DR		–	√	–	00H
FFBBH	IEBus unit status register	USR	R	√	√	–	
FFBCH	IEBus interrupt status register	ISR	R/W	√	√	–	
FFBDH	IEBus slave status register	SSR	R	√	√	–	41H
FFBEH	IEBus communication success counter	SCR		–	√	–	01H
FFBFH	IEBus transmit counter	CCR		–	√	–	20H

Cautions 1. The above registers are mapped to the SFR space.

2. Registers UAR, SAR, and PAR must be manipulated in word units.

3. Instructions in Read Modify Write mode (such as XCH and ROL4) cannot be used for DR, CDR, DLR, and ISR.

4. When using the IEBus, set port mode register 12 (PM12) and the output latch of port 12 (P12) as follows.

Set the P120/$\overline{TX0}$ pin in the output mode (PM120 = 0, P120 = 0)

Set the P121/$\overline{RX0}$ pin in the input mode (PM121 = 1)

17.4.2 Description of internal registers

The internal registers incorporated in the IEBus controller are described below.

(1) IEBus control register 0 (BCR0)

Figure 17-11. Format of IEBus Control Register 0 (BCR0)

After reset: 00H R/W Address: FFB0H

	<7>	<6>	<5>	<4>	<3>	2	1	0
BCR0	ENIEBUS	MSTRQ	ALLRQ	ENSLVTX	ENSLVRX	0	0	0

ENIEBUS	Communication enable flag
0	IEBus unit stopped
1	IEBus unit active

MSTRQ	Master request flag
0	IEBus unit not requested as master
1	IEBus unit requested as master

ALLRQ	Broadcast request flag
0	Individual communication requested
1	Broadcast communication requested

ENSLVTX	Slave transmission enable flag
0	Slave transmission disabled
1	Slave transmission enabled

ENSLVRX	Slave reception enable flag
0	Slave reception disabled
1	Slave reception enabled

★ **Cautions 1. While the IEBus is operating as the master, writing to the BCR0 register (including bit manipulation instructions) is disabled until either the end of that communication or frame, or until communication is stopped by the occurrence of an arbitration-loss communication error. Master requests cannot therefore be nested. However, if the IEBus is specified as a slave while a master request is being held pending, the BCR0 register can be written to at the end of communication to clear the communication end/frame end flag. This is also the case when communication has been forcibly stopped (ENIEBUS flag = 0).**

2. If a bit manipulation instruction for the BCR0 register conflicts with a hardware reset of the MSTRQ flag, the BCR0 register may not operate normally. The following countermeasures are recommended in this case.

- **Because the hardware reset is instigated in the acknowledgment period of the slave address field, be sure to observe Caution 1 of (b) Master request flag (MSTRQ) below.**
- **Be sure to observe the caution above regarding writing to the BCR0 register.**

(a) Communication enable flag (ENIEBUS)...Bit 7

<Set/reset conditions>

Set: By software

Reset: By software

★ **Caution Before setting the ENIEBUS flag, make the following setting.**

- **Set the interrupt enabled (EI) status and enable the interrupt servicing of INTIE2 (IEMK2 = 0).**
- **Set the IEBus unit address register (UAR)**

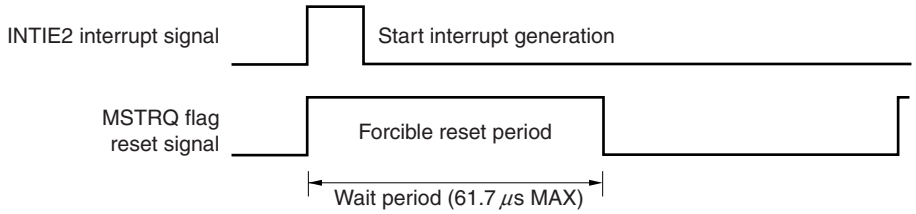
★ **(b) Master request flag (MSTRQ)...Bit 6**

<Set/reset conditions>

Set: By software

Reset: By hardware, at the end of the arbitration period. Because the reset signal is generated in the ACK period of the slave address field, if a MSTRQ flag setting instruction is sent in this period, it will be invalid.

Cautions 1. The master request should be resent by software following a loss in arbitration. When resending the master request in this case, set (1) the MSTRQ flag after securing the required wait period. This flag is unable to be set (1) before the end of this wait period.



2. When a master request has been sent and bus mastership acquired, do not set the MSTRQ, ENSLVTX, or ENSLVRX flag until the end of communication (i.e. the ISR register's communication end/frame end flag is set (1)) as setting these flags disables interrupt request generation. However, these flags can be set if communication has been aborted.

(c) Broadcast request flag (ALLRQ)...Bit 5

<Set/reset conditions>

Set: By software

Reset: By software

Caution When requesting broadcast communication, always set the ALLRQ flag, then the MSTRQ flag.

(d) Slave transmission enable flag (ENSLVTX)...Bit 4

<Set/reset conditions>

Set: By software

Reset: By software

★ **Cautions 1. Clear the ENSLVTX flag before setting the MSTRQ flag when making a master request. If a slave transmission request is sent in slave mode when the ENSLVTX flag is unset, NACK in the control field will be returned. Moreover, when returning to an enabled state from a disabled state, transmission becomes valid from the next frame.**

2. If the controller receives control data for data/control writing (3H, 7H) when the ENSLVTX flag is unset, NACK will be returned via the acknowledge bit of the control field.

3. The ENSLVTX flag will be set and the status interrupt (INTIE2) will be generated when the control data (0H, 4H, 5H, 6H) of a slave status request is returned, even if the ENSLVTX flag is in the reset status. At this time, the data returned via the acknowledge bit of the control field ($\overline{ACK}$ or NACK) depends on the status of the local unit and the received control data.

(e) Slave reception enable flag (ENSLVRX)...Bit 3

<Set/reset conditions>

Set: By software

Reset: By software

★ **Caution If the ENSLVRX flag is reset when the IEBus is busy with other CPU processing, NACK will be returned via the acknowledge bit of the control field, making it possible to disable slave reception. Note that resetting this flag only disables individual communication, not broadcast communication. If the received slave address matches the unit address during individual communication, however, the start interrupt (INTIE2) is generated. If CPU processing has priority (neither reception nor transmission occurs), be sure to stop the IEBus unit by resetting the ENIEBUS flag. Note also that when returning to an enabled state from a disabled state, transmission becomes valid from the next frame.**

(2) IEBus unit address register (UAR)

This register sets the local address of an IEBus unit. This register must be always set before starting communication.

The unit address (12 bits) is set to bits 11 to 0.

Figure 17-12. Format of IEBus Unit Address Register (UAR)

After reset: 0000H R/W Address: FFB2H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UAR	0	0	0	0												

(3) IEBus slave address register (SAR)

When a master request is issued, the value of this register is reflected in the value of the transmit data in the slave address field. This register must always be set before starting communication.

The slave address (12 bits) is set to bits 11 to 0.

Figure 17-13. Format of IEBus Slave Address Register (SAR)

After reset: 0000H R/W Address: FFB4H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAR	0	0	0	0												

(4) IEBus partner address register (PAR)

(a) Slave unit

The value of the receive data in the master address field (address of the master unit) is written to this register.

If a request “4H” to read the lock address (lower 8 bits) is received from the master, the CPU must read the value of this register, and write it to the lower 8 bits of the IEBus data register (DR).

If a request “5H” to read the lock address (higher 4 bits) is received from the master, the CPU must read the value of this register and write the data of the higher 4 bits to DR.

The partner address (12 bits) is set to bits 11 to 0.

Figure 17-14. Format of IEBus Partner Address Register (PAR)

After reset: 0000H R Address: FFB6H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PAR	0	0	0	0												

(5) IEBus control data register (CDR)

(a) Master unit

The data of the lower 4 bits is reflected in the data transmitted in the control field. When a master request is issued, this register must be set in advance before starting communication.

(b) Slave unit

The data received in the control field is written to the lower 4 bits.

When the status transmission flag (STATUSF) of the IEBus interrupt status register (ISR) is set, an interrupt (INTIE2) is issued, and each processing should be performed by software, according to the value of the lower 4 bits of CDR.

Figure 17-15. Format of IEBus Control Data Register (CDR)

After reset: 01H R/W Address: FFB8H

	7	6	5	4	3	2	1	0
CDR	0	0	0	0	CDR3	CDR2	CDR1	CDR0

CDR3	CDR2	CDR1	CDR0	Function
0	0	0	0	Read slave status
0	0	0	1	Undefined
0	0	1	0	Undefined
0	0	1	1	Read data and lock
0	1	0	0	Read lock address (lower 8 bits)
0	1	0	1	Read lock address (higher 4 bits)
0	1	1	0	Read slave status and unlock
0	1	1	1	Read data
1	0	0	0	Undefined
1	0	0	1	Undefined
1	0	1	0	Write command and lock
1	0	1	1	Write data and lock
1	1	0	0	Undefined
1	1	0	1	Undefined
1	1	1	0	Write command
1	1	1	1	Write data

Cautions 1. Because the slave unit must judge whether the received data is a "command" or "data", it must read the value of this register after completing communication.

2. Instructions in Read Modify Write mode (such as XCH and ROL4) cannot be used for CDR.

3. If the master unit sets an undefined value, NACK is returned from the slave unit, and communication is aborted. During broadcast communication, however, the master unit continues communication without recognizing $\overline{\text{ACK}}$/NACK; therefore, make sure not to set an undefined value to this register during broadcast communication.

★ **4. In the case of defeat in a bus conflict and a slave status request is received from the unit that won, the IEBus telegraph length register (DLR) is fixed to "01H". Therefore, when a re-request of the master follows, the appointed telegraph length must be set to DLR.**

★ **(c) Slave status return operation**

When the IEBus receives a request to transfer from master to slave status (control data: 0H, 6H) or a lock address request (4H, 5H), whether $\overline{ACK}$ in the control field is returned or not depends on the status of the IEBus unit.

(1) If 0H or 6H control data was received in the unlocked state → $\overline{ACK}$ returned
(2) If 4H or 5H control data was received in the unlocked state → $\overline{ACK}$ not returned
(3) If 0H, 4H, 5H or 6H control data was received in the locked state from the unit that sent the lock request → $\overline{ACK}$ returned
(4) If 0H, 4H, or 5H control data was received in the locked state from other than the unit that sent the lock request → $\overline{ACK}$ returned
(5) If 6H control data was received in the locked state from other than the unit that sent the lock request → $\overline{ACK}$ not returned

In all of the above cases, the acknowledgment of a slave status or lock address request will cause the STATUSF flag (bit 4 of the ISR register) to be set and the status interrupt request (INTIE2) to be generated. The generation timing is at the end of the control field parity bit (at the start of the $\overline{ACK}$ bit). However, if $\overline{ACK}$ is not returned, a NACK error is generated after the $\overline{ACK}$ bit, and communication is terminated.

Figure 17-16. Interrupt Generation Timing (for (1), (3), and (4))

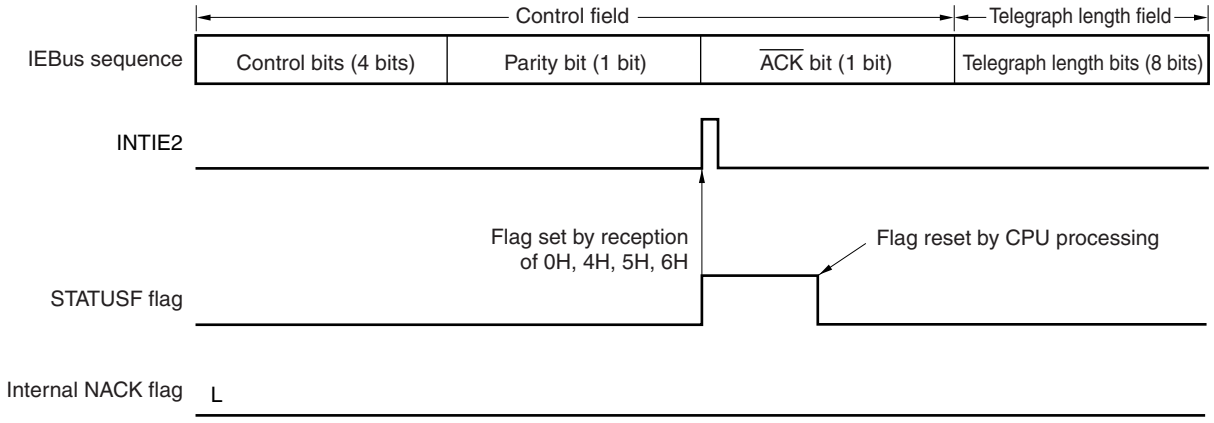
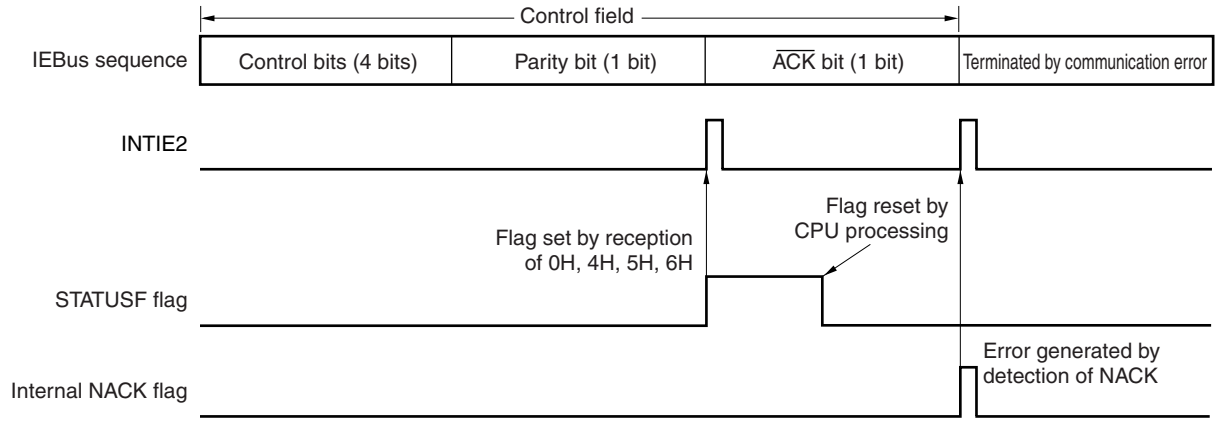
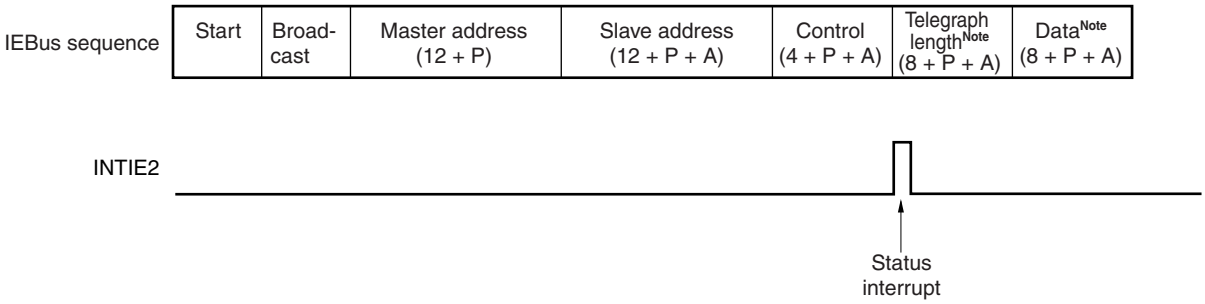


Figure 17-17. Interrupt Generation Timing (for (2) and (5))



Because in (4) and (5) the communication was from other than the unit that sent the lock request while the IEBus was in the locked state, the start or communication complete interrupt (INTIE2) is not generated, even if the IEBus unit is the communication target. The STATUSF flag (bit 4 of the IEBus interrupt status register (ISR)) is set and the status interrupt request (INTIE2) generated, however, if a slave status or lock address request is acknowledged. Note that even if the same control data is received while the IEBus is in the locked state, the interrupt generation timing for INTIE2 differs depending on whether the master unit (3) or another unit (4) is requesting the locked state.

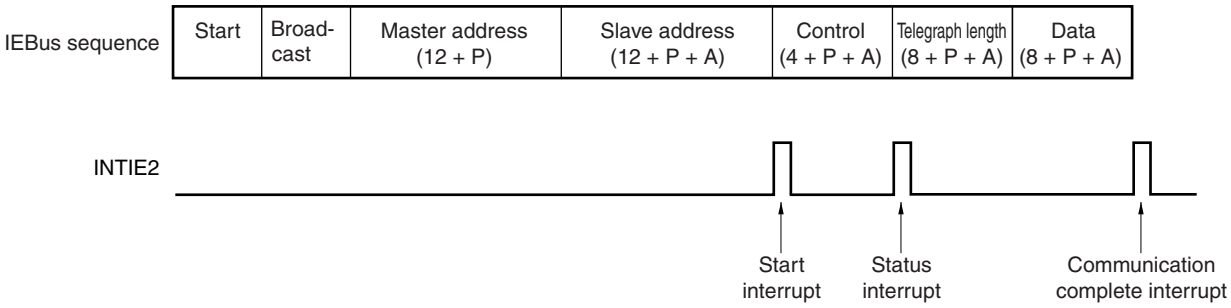
Figure 17-18. Timing of INTIE2 Interrupt Generation in Locked State (for (4) and (5))



Note The telegraph length and data modes are not set in the case of (5) because $\overline{ACK}$ is not returned.

Remark P: Parity bit, A: $\overline{ACK}$/NACK bit

Figure 17-19. Timing of INTIE2 Interrupt Generation in Locked State (for (3))



Remark P: Parity bit, A: $\overline{ACK}$/NACK bit

(6) IEBus telegraph length register (DLR)

(a) Transmission unit (master transmission, slave transmission)

The data of this register is reflected in the data transmitted in the telegraph length field and indicates the number of bytes of the transmit data. This register must be set in advance before transmission.

(b) Reception unit (master reception, slave reception)

The receive data in the telegraph length field transmitted from the transmission unit is written to this register.

★ **Remark** The IEBus telegraph length register consists of a write register and a read register. Consequently, data written to this register cannot be read as is. The data that can be read is the data received during IEBus communication.

Figure 17-20. Format of IEBus Telegraph Length Register (DLR)

After reset: 01H R/W Address: FFB9H

	7	6	5	4	3	2	1	0
DLR								

Bit								Setting value	Number of communication data bytes
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	01H	1 byte
0	0	0	0	0	0	1	0	02H	2 bytes
:	:	:	:	:	:	:	:	:	:
0	0	1	0	0	0	0	0	20H	32 bytes
:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	1	1	FFH	255 bytes
0	0	0	0	0	0	0	0	00H	256 bytes

Cautions 1. If the master issues a request "0H, 4H, 5H, or 6H" to transmit a slave status and lock address (higher 4 bits, lower 8 bits), the contents of this register are set to "01H" by hardware; therefore, the CPU does not have to set this register.

★ **2. In the case of defeat in a bus conflict and a slave status request is received from the unit that won, the IEBus telegraph length register (DLR) is fixed to "01H". Therefore, when a re-request of the master follows, the appointed telegraph length must be set to DLR.**

3. Instructions in Read Modify Write mode (such as XCH and ROL4) cannot be used for DLR.

★ **(7) IEBus data register (DR)**

The IEBus data register (DR) sets the communication data. The communication data (8 bits) is set to bits 7 to 0.

Remark The IEBus data register consists of a write register and a read register. Consequently, data written to this register cannot be read as is. The data that can be read is the data received during IEBus communication.

(a) Transmission unit

The data (1 byte) written to the IEBus data register (DR) is stored in the shift register of the IEBus. It is then output from the most significant bit, and an interrupt (INTIE1) is issued to the CPU each time 1 byte has been transmitted. If NACK is received after 1-byte data has been transmitted during individual transfer, the next data is not transferred from DR to the shift register, and the same data is retransmitted. At this time, INTIE1 is not generated.

INTIE1 is issued when the IEBus interface shift register stores the IEBus data register (DR) value. However, when the last byte and 32nd byte (the last byte of 1 communication frame) are stored in the shift register, INTIE1 is not issued.

(b) Reception unit

One byte of the data received by the shift register of the IEBus interface block is stored to this register. Each time 1 byte has been correctly received, an interrupt (INTIE1) is issued.

Figure 17-21. Format of IEBus Data Register (DR)

After reset: 00H R/W Address: FFBAH

	7	6	5	4	3	2	1	0
DR								

Cautions 1. If the next data is not set in time while the transmission unit is set, an underrun occurs, and a communication error interrupt (INTIE2) occurs, stopping transmission.

2. When the IEBus is a receiving unit, if the reading of the data is too late for the next data reception timing, the unit will enter the overrun state. At this time, during individual communication reception, NACK will be returned at the acknowledge bit of the data field, and the master unit will be requested to retransmit the data. If an overrun error occurs during broadcast communication reception, the communication error interrupt (INTIE2) is generated.

3. Instructions in Read Modify Write mode (such as XCH and ROL4) cannot be used for DR.

(8) IEBus unit status register (USR)

Figure 17-22. Format of IEBus Unit Status Register (USR)

After reset: 00H R Address: FFBBH

	7	<6>	<5>	<4>	<3>	<2>	1	0
USR	0	SLVRQ	ARBIT	ALLTRNS	ACK	LOCK	0	0

SLVRQ	Slave request flag
0	No request from master to slave
1	Request from master to slave

ARBIT	Arbitration result flag
0	Arbitration win
1	Arbitration loss

ALLTRNS	Broadcast communication flag
0	Individual communication status
1	Broadcast communication status

ACK	$\overline{\text{ACK}}$ transmission flag
0	NACK transmitted
1	$\overline{\text{ACK}}$ transmitted

LOCK	Lock status flag
0	Unit unlocked
1	Unit locked

(a) Slave request flag (SLVRQ)...Bit 6

A flag indicating whether there has been a slave request from the master.

★ <Set/reset conditions>

Set: When the unit is requested as a slave (if the received slave address and unit UAR match during individual communication reception, or if the higher 4 bits of the received slave address match or if the received slave address is FFFH during broadcast communication reception), this flag is set by hardware when the acknowledge period of the slave address field starts.

Reset: This flag is reset by hardware when the unit is not requested as a slave. The reset timing is the same as the set timing. If the unit is requested as a slave immediately after communication has been correctly received (when the SLVRQ bit is set), and if a parity error occurs in the slave address field for that communication, the flag is not reset.

(b) Arbitration result flag (ARBIT)...Bit 5

A flag that indicates the result of arbitration.

<Set/reset conditions>

Set: When the data output by the IEBus unit during the arbitration period does not match the bus line data after the master request.

Reset: By the start bit timing.

★ **Cautions 1. The timing at which the arbitration result flag (ARBIT) is reset differs depending on whether the unit outputs a start bit.**

- **If start bit is output: The flag is reset at the output start timing.**
- **If start bit is not output: The flag is reset at the detection timing of the start bit (approx. 160 μs after output)**

2. The flag is reset at the detection timing of the start bit if the other unit outputs the start bit earlier and the local unit does not output the start bit after the master request.

★ **(c) Broadcast communication flag (ALLTRNS)...Bit 4**

A flag that indicates whether the unit is performing broadcast communication. The contents of the flag are updated in the broadcast field of each frame.

Except for initialization (reset) by system reset, the set/reset conditions vary depending on the receive data of the broadcast field bit.

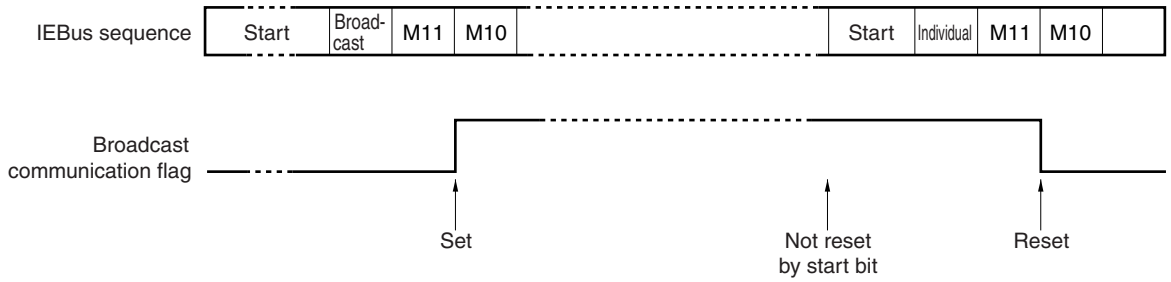
<Set/reset conditions>

Set: When “Broadcast” is received by the broadcast field

Reset: When “individual” is received by the broadcast field, or upon the input of a system reset.

Caution The broadcast communication flag is updated regardless of whether the IEBus is the communication target or not.

Figure 17-23. Example of broadcast Communication Flag Operation



(d) $\overline{\text{ACK}}$ transmission flag ($\overline{\text{ACK}}$)...Bit 3

A flag that indicates whether $\overline{\text{ACK}}$ has been transmitted in the $\overline{\text{ACK}}$ period of the $\overline{\text{ACK}}$ field when the IEBus is a receiving unit. The contents of the flag are updated in the $\overline{\text{ACK}}$ period of each frame. However, if the internal circuit is initialized by the occurrence of a parity error, etc., the contents are not updated in the $\overline{\text{ACK}}$ period of that field.

★ **(e) Lock status flag (LOCK)...Bit 2**

A flag that indicates whether the unit is locked.

<Set/reset conditions>

Set: When the communication end flag (ENDTRNS) goes low level and the frame end flag (ENDFRAM) goes high level after receipt of a lock specification (3H, 6H, AH, BH) in the control field.

Reset: When the communication enable flag (ENIEBUS) is cleared.
When the communication end flag is set after receipt of a lock release (3H, 6H, AH, BH) in the control field.

Caution Lock specification/release is not possible in broadcast communication. In the lock status, individual communication from a unit other than the one that requests locking is not acknowledged. However, even communication from a unit other than the one that requests locking is acknowledged as long as the communication is a slave status request.

Remark ENDTRNS: Bit 3 of the IEBus interrupt status register (ISR)
ENDFRAM: Bit 2 of the IEBus interrupt status register (ISR)
ENIEBUS: Bit 7 of IEBus control register 0 (BCR0)

★ **(9) IEBus interrupt status register (ISR)**

This register indicates the status when the IEBus issues an interrupt. ISR is read to generate an interrupt, after which the specified interrupt servicing is carried out.

Reset the ISR register after reading it. Until it is reset, the INTIE2 interrupt signal is not generated (nor held pending).

To reset the ISR register, reset each flag, satisfying the reset conditions in Table 17-9.

Table 17-9. Reset Conditions of Flags in ISR Register

Flag Name	Reset Condition	Processing Example
IEERR, STARTF, STATUSF	Byte write operation of ISR register. Any value can be written.	ISR = 00H, etc.
ENDTRNS, ENDFRAM	Set MSTRQ, ENSLVTX, or ENSLVRX flag.	BCR0 register = 88H or ENSLVTX = 1, etc.

Caution Even if 0 is written to the ENDTRNS or ENDFRAM flag by accessing the ISR register, these flags are not reset. Reset them as described above.

Remark MSTRQ: Bit 6 of IEBus control register 0 (BCR0)
ENSLVTX: Bit 4 of IEBus control register 0 (BCR0)
ENSLVRX: Bit 3 of IEBus control register 0 (BCR0)

Figure 17-24. Format of IEBus Interrupt Status Register (ISR)

After reset: 00H R/W Address: FFBCH

	7	<6>	<5>	<4>	<3>	<2>	1	0
ISR	0	IEERR	STARTF	STATUSF	ENDTRNS	ENDFRAM	0	0

IEERR	Communication error flag (during communication)
0	No communication error
1	Communication error

STARTF	Start interrupt flag
0	Start interrupt does not occur
1	Start interrupt occurs

STATUSF	Status transmission flag (slave)
0	No slave status/lock address (higher 4 bits, lower 8 bits) transmission request
1	Slave status/lock address (higher 4 bits, lower 8 bits) transmission request

ENDTRNS	Communication end flag
0	Communication does not end after the number of bytes set in the telegraph length field have been transferred
1	Communication ends after the number of bytes set in the telegraph length field have been transferred

ENDFRAM	Frame end flag
0	The frame (transfer of the maximum number of bytes (32 bytes) prescribed by mode 1) did not end
1	The frame (transfer of the maximum number of bytes (32 bytes) prescribed by mode 1) ended

Caution Each of IEERR, STARTF, STATUSF, ENDTRNS, and ENDFRAM are generation triggers for the interrupt request signal (INTIE2) (see Figure 17-28 Configuration of Interrupt Control Block). Because of this, if any one of these interrupt triggers has been set, no new interrupt will be generated by a subsequent trigger. Clear the flag of the interrupt source by the interrupt servicing program before the next interrupt occurs.

(a) Communication error flag (IEERR)...Bit 6

A flag that indicates the detection of an error during communication.

<Set/reset conditions>

Set: The flag is set if a timing error, parity error (except in the data field), NACK reception (except in the data field), underrun error, or overrun error (that occurs during broadcast communication reception) occurs.

Reset: By software

(b) Start interrupt flag (STARTF)...Bit 5

A flag that indicates whether an interrupt was in the ACK period of the slave address field.

<Set/reset conditions>

Set: In the slave address field, upon a master request. When the IEBus is a slave unit, this flag is set upon a request from the master (only if it was a slave request in the locked state from the unit requesting a lock).

Reset: By software

(c) Status transmission flag (STATUSF)...Bit 4

A flag indicating that the transmission status is either the master to slave status, or the lock address (higher 4 bits, lower 8 bits), when the IEBus is a slave unit.

<Set/reset conditions>

Set: When 0H, 4H, 5H, or 6H is received in the control field from the master when the IEBus is a slave unit.

Reset: By software

(d) Communication end flag (ENDTRNS)...Bit 3

A flag that indicates whether communication ends after the number of bytes set in the telegraph length field have been transferred.

<Set/reset conditions>

Set: When the value of the IEBus communication success counter (SCR) is 0.

Reset: When the MSTRQ, ENSLVTX, or ENSLVRX flag of IEBus control register 0 (BCR0) is set.

(e) Frame end flag (ENDFRAM)...Bit 2

A flag that indicates whether communication ends after the maximum number of bytes (32 bytes) prescribed by mode 1 have been transferred.

<Set/reset conditions>

Set: When the value of the IEBus transmit counter (CCR) is 0.

Reset: When the MSTRQ, ENSLVTX, or ENSLVRX flag of IEBus control register 0 (BCR0) is set.

(f) Communication error triggers

- **Timing error**

  Occurrence conditions: Occurs if the high/low level width of the communication bit has shifted from the prescribed value.

  Remark: The respective prescribed values are set in the bit processing block and monitored by the internal 8-bit timer. An interrupt is generated when a timing error occurs.

- **Parity error**

  Occurrence conditions: Occurs if the generated parity and the received parity in each field do not match when the IEBus is a receiving unit.

  Remark: During individual communication, an interrupt is generated if a parity error occurs in a field other than the data field.
  During broadcast communication, an interrupt is generated even if a parity error occurs in the data field.

  Restriction: If there is a slave request that has lost in arbitration to a broadcast request, no interrupt is generated, even if a parity error occurs.

- **NACK reception**

  Occurrence conditions: This error occurs when NACK is received during the $\overline{\text{ACK}}$ period in each of the slave address, control, and telegraph length fields during individual communication, regardless of whether the unit is the master or a slave unit.
  A NACK reception only occurs in individual communication. $\overline{\text{ACK}}$ and NACK are not discriminated in broadcast communication.

  Remark: An interrupt is generated if NACK is received in a field other than the data field.

- **Underrun error**

  Occurrence conditions: Occurs during data transmission if there was insufficient time to write the next transmit data to the IEBus data register (DR) before $\overline{\text{ACK}}$ reception.

  Remark: An interrupt is generated if an underrun occurs.

- **Overrun error**

  Occurrence conditions: The data interrupt request (INTIE1) that stores each byte of data in the IEBus data register (DR) is generated, and the DR register is read by software. An overrun error occurs if this reading processing is late and its timing becomes that of the next data reception.

  Remark: In individual communication reception, an acknowledgment is not returned in the ACK period of this data, resulting in the retransmission of the data by the transmitting unit. Consequently, the IEBus transfer counter (CCR) is decremented, whereas the IEBus communication success counter (SCR) is not. In broadcast communication reception, reception is stopped by the occurrence of a communication error interrupt request (INTIE2), at which time the DR register is not updated. The STATRX flag (bit 1 of the SSR register) also remains set (1) without generating INTIE1. The overrun state is released at the timing of the next data reception following the reading of DR.

(g) Overrun error - supplementary details

(i) When the frame ends in the overrun state during individual communication reception

If the DR register is not read after entering the overrun state and the retransmitted data reaches the maximum number of bytes (32 bytes), the frame end interrupt (INTIE2) is generated. The overrun state is maintained until the DR register is read after the end of the frame.

(ii) If the next reception is started in the case of (i) above, or if the next reception is started without the DR register being read after the final data has been received, regardless of whether the communication is broadcast or individual

Even if communication to the IEBus unit starts in the overrun state, the cause of the overrun, NACK, is not returned in the ACK period of the slave address, control, or telegraph length field (the DR register is not updated). If the next communication is not to the IEBus unit, the DR register is not updated until it is read. Because the IEBus unit is not a communication target, the data interrupt (INTIE1) and communication error interrupt (INTIE2) are not generated.

(iii) If the next transmission occurs in the overrun state

The data to be transmitted next in the overrun state can be no more than 2 bytes long.

Because the data request interrupt (INTIE1) is not generated, the transmit data cannot be set, resulting in an underrun error. Therefore, clear the overrun status before starting transmission.

(iv) Overrun state release

The overrun state can only be released by reading the DR register or by a system reset. Therefore, be sure to read DR in the communication error interrupt servicing program.

(10) IEBus slave status register (SSR)

This register indicates the communication status of the slave unit. After receiving a slave status transmission request from the master, the CPU reads this register, and writes the slave status to the IEBus data register (DR) to transmit the slave status. At this time, the telegraph length is automatically set to “01H”, so setting of the IEBus telegraph length register (DLR) is not required (because it is preset by hardware).

Bits 6 and 7 indicate the highest mode supported by the unit, and are fixed to “01H” (mode 1).

Figure 17-25. Format of IEBus Slave Status Register (SSR)

After reset: 41H R Address: FFBDH

	7	6	5	<4>	3	<2>	<1>	<0>
SSR	0	1	0	STATSLV	0	STATLOCK	STATRX	STATTX

STATSLV	Slave transmission status flag
0	Slave transmission stops
1	Slave transmission enabled

STATLOCK	Lock status flag
0	Unlock status
1	Lock status

STATRX	DR receive status
0	Receive data not stored in DR
1	Receive data stored in DR

STATTX	DR transmit status
0	Transmit data not stored in DR
1	Transmit data stored in DR

(a) Slave transmission status flag (STATSLV)...Bit 4

Reflects the contents of the slave transmission enable flag.

(b) Lock status flag (STATLOCK)...Bit 2

Reflects the contents of the locked flag.

(c) DR reception status (STATRX)...Bit 1

This flag indicates the DR reception state.

(d) DR transmission status (STATTX)...Bit 0

This flag indicates the DR transmission state.

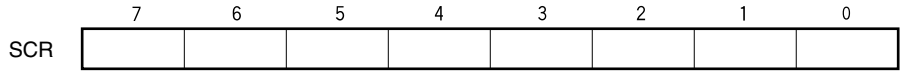
(11) IEBus communication success counter (SCR)

The IEBus communication success counter (SCR) indicates the number of remaining communication bytes.

This register reads the count value of the counter in the value set by the IEBus telegraph length register (DLR) is decremented by $\overline{ACK}$ in the data field. When the count value has reached "00H", the communication end flag (ENDTRNS) of the IEBus interrupt status register (ISR) is set.

Figure 17-26. Format of IEBus Communication Success Counter (SCR)

After reset: 01H R Address: FFBEH



Bit								Setting value	Remaining number of communication data bytes
7	6	5	4	3	2	1	0		
0	0	0	0	0	0	0	1	01H	1 byte
0	0	0	0	0	0	1	0	02H	2 bytes
:	:	:	:	:	:	:	:	:	:
0	0	1	0	0	0	0	0	20H	32 bytes
:	:	:	:	:	:	:	:	:	:
1	1	1	1	1	1	1	1	FFH	255 bytes
0	0	0	0	0	0	0	0	00H	0 bytes (end of communication) or 256 bytes[Note]

Note The actual hard counter consists of 9 bits. When "00H" is read, it cannot be judged whether the remaining number of communication data bytes is 0 (end of communication) or 256. Therefore, either the communication end flag is used, or if "00H" is read when the first interrupt occurs at the beginning of communication, the remaining number of communication data bytes is judged to be 256.

(12) IEBus transmit counter (CCR)

The IEBus transmit counter (CCR) indicates the number of remaining bytes of the communication byte number specified in the communication mode.

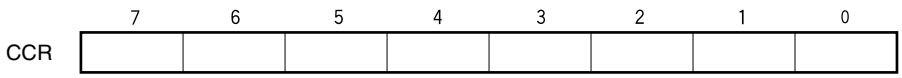
Bits 7 to 0 of the IEBus transmit counter (CCR) indicate the number of transfer bytes.

This register reads the count value of the counter that is preset to the maximum number of transmitted bytes (32 bytes) per frame specified in mode 1. Whereas SCR (IEBus communication success counter) is decremented during normal communication ($\overline{ACK}$), CCR is decremented when 1 byte has been communicated, regardless of whether $\overline{ACK}$ or NACK. When the count value has reached "00H", the frame end flag (ENDFRAM) of the IEBus interrupt status register (ISR) is set.

The maximum number of transfer bytes of the preset value of mode 1 per frame is 20H (32 bytes).

Figure 17-27. Format of IEBus Transmit Counter (CCR)

After reset: 20H R Address: FFBFH



17.5 Interrupt Operations of IEBus Controller

★ 17.5.1 Interrupt control block

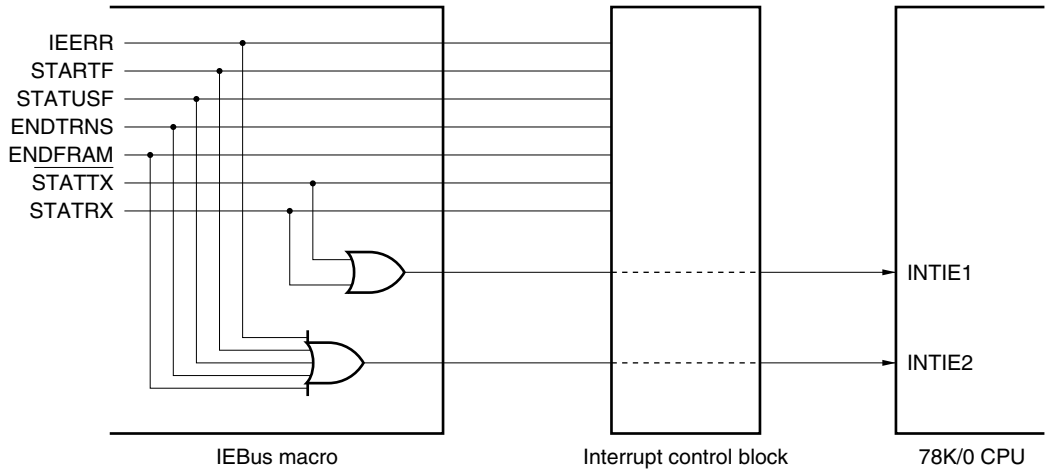
The interrupt request signals are shown below.

<1>	Communication error	IEERR
<2>	Start interrupt	STARTF
<3>	Status communication	STATUSF
<4>	End of communication	ENDTRNS
<5>	End of frame	ENDFRAM
<6>	Transmit data write request	$\overline{\text{STATTX}}$
<7>	Receive data read request	STATRX

<1> to <5> of the above interrupt requests are assigned to the IEBus interrupt status register (ISR). For details, refer to **Table 17-10 Interrupt Source List**.

The configuration of the interrupt control block is illustrated below.

Figure 17-28. Configuration of Interrupt Control Block



Caution OR output of IEERR, STARTF, STATUSF, ENDTRNS, and ENDFRAM is treated as a vectored interrupt request signal (INTIE2).

17.5.2 Interrupt source list

The interrupt sources are listed below.

Table 17-10. Interrupt Source List

Interrupt Source		Condition of Generation		CPU Processing After Generation of Interrupt	Remark
		Unit	Field		
Communication error	Timing error	Master/slave	All fields	Undo communication processing	Communication error is OR output of timing error, parity error, NACK reception, underrun error, and overrun error
	Parity error	Reception	Other than data (individual)		
			All fields (Broadcast)		
	NACK reception	Reception (Transmission)	Other than data (individual)		
	Underrun error	Transmission	Data		
	Overrun error	Reception	Data (Broadcast)		
Start interrupt		Master	Slave/address	Slave request judgment Copnflict judgment (If lost, remaster processing) Communication preparation processing	Interrupt always occurs if conflict lost when master request is issued.
		Slave	Slave/address	Slave request judgment Communication preparation processing	Generated only when slave request is issued.
Status transmission		Slave	Control	Refer to transmission processing example such as slave status.	Interrupt occurs regardless of slave transmission enable flag. Interrupt occurs if NACK is returned in the control field.
End of communication		Transmission	Data	End processing by software	Set if SCR is cleared to 0
		Reception	Data	End processing by software Receive data processing	
End of frame		Transmission	Data	Retransmission preparation processing	Set if CCR is cleared to 0
		Reception	Data	Re-reception preparation processing	
Transmit data write		Transmission	Data	Transmit data write processing by software	Set after transfer of transmission data to internal shift register. This does not occur when the last data is transferred.
Receive data read		Reception	Data	Receive data read processing by software	Set after normal data reception

★ 17.5.3 Communication error source processing list

The following table shows the occurrence conditions of the communication errors (timing error, NACK reception, overrun error, underrun error, and parity error), error processing by the internal IEBus controller, and examples of processing by software.

Table 17-11. Communication Error Source Processing List (1/3)

		Timing Error			
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	If bit specification timing is not correct			
	Location of occurrence	Other than data field	Data field	Other than data field	Data field
Broadcast communication	Hardware processing	• Reception stops. • INTIE2 occurs • To start bit waiting status **Remark** Communication between other units does not end.		• Transmission stops. • INTIE2 occurs • To start bit waiting status	
	Software processing	• Error processing (such as retransmission request)		• Error processing (such as retransmission request)	
Individual communication	Hardware processing	• Reception stops. • INTIE2 occurs. • NACK is returned. • To start bit waiting status		• Transmission stops. • INTIE2 occurs. • To start bit waiting status	
	Software processing	• Error processing (such as retransmission request)		• Error processing (such as retransmission request)	

		NACK Reception				
Occurrence condition	Unit status	Reception		Transmission		
	Occurrence condition	Unit NACK transmission		NACK reception		
	Location of occurrence	Other than data field	Data field	Other than data field	Data field	NACK reception of data of 32nd byte
Broadcast communication	Hardware processing	–	–	–	–	–
	Software processing	–	–	–	–	–
Individual communication	Hardware processing	• Reception stops. • INTIE2 occurs. • To start bit waiting status	• INTIE2 does not occur. • Data retransmitted by other unit is received.	• Transmission stops. • INTIE2 occurs. • To start bit waiting status	• INTIE2 does not occur. • Retrans-mission processing	• INTIE2 occurs[Note]. • To start bit waiting status
	Software processing	• Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)

Note Both ISR.6 (IEERR) and ISR.2 (ENDFRAM) are set to 1.
To reset them, satisfy the conditions in Table 17-9.

Table 17-11. Communication Error Source Processing List (2/3)

		Overrun Error		Underrun Error	
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	DR cannot be read in time before the next data is received.		DR cannot be written in time before the next data is transmitted.	
	Location of occurrence	Other than data field	Data field	Other than data field	Data field
Broadcast communication	Hardware processing	–	• Reception stops. • INTIE2 occurs. • To start bit waiting status **Remarks 1.** Communication between other units does not end. **2.** Data cannot be received until the overrun status is cleared.	–	• Transmission stops. • INTIE2 occurs. • To start bit waiting status
	Software processing	–	• DR is read and overrun status is cleared. • Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)
Individual communication	Hardware processing	–	• INTIE2 does not occur. • NACK is returned. • Data is retransmitted from other unit. **Remark** Data cannot be received until overrun status is cleared.	–	• Transmission stops. • INTIE2 occurs. • To start bit waiting status
	Software processing	–	• DR is read and overrun status is cleared. • Error processing (such as retransmission request)	–	• Error processing (such as retransmission request)

Table 17-11. Communication Error Source Processing List (3/3)

		Parity Error			
Occurrence condition	Unit status	Reception		Transmission	
	Occurrence condition	Received data and received parity do not match.		–	
	Location of occurrence	Other than data field	Data field	Other than data field	Data field
Broadcast communication	Hardware processing	• Reception stops. • INTIE2 occurs. • To start bit waiting status **Remark** Communication between other units does not end.		–	–
	Software processing	• Error processing (such as retransmission request)		–	–
Individual communication	Hardware processing	• Reception stops. • INTIE2 occurs. • To start bit waiting status	• Reception does not stop. • INTIE2 does not occur. • NACK is returned. • Data retransmitted by other unit is received.	–	–
	Software processing	• Error processing (such as retransmission request)	–	–	–

17.6 Interrupt Generation Timing and Main CPU Processing

17.6.1 Master transmission

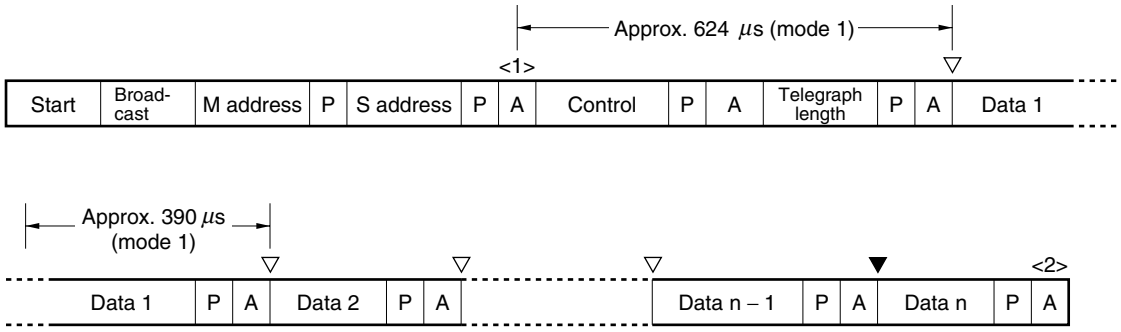
Initial preparation processing:

Set a unit address, slave address, control data, telegraph length, and the first byte of the transmit data.

Communication start processing:

Set the bus control register (enable communication, master request, and slave reception).

Figure 17-29. Master Transmission



<1> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of slave request → Slave reception processing
(See **17.6.1 (1) Slave reception processing**)

↓

Judgment of conflict result → Remaster request processing

<2> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Recommunication processing
(See **17.6.1 (3) Recommunication processing**)

Remarks 1. ▽: Interrupt (INTIE1) occurrence (See **17.6.1 (2) Interrupt (INTIE1) occurrence**)
The transmit data of the second and subsequent bytes are written to the IEBus data register (DR) by software. At this time, the data transfer direction is RAM (memory) → SFR (peripheral).

2. ▼: An interrupt (INTIE1) does not occur.

3. n = Final number of data bytes

(1) Slave reception processing

If a slave reception request is confirmed during vectored interrupt servicing, change the data transfer direction from RAM (memory) → SFR (peripheral) to SFR (peripheral) → RAM (memory) by software until the first data is received. The maximum pending period of this data transfer direction changing processing is about 1040 μs in communication mode 1.

(2) Interrupt (INTIE1) occurrence

If NACK is received from the slave in the data field, an interrupt (INTIE1) is not issued to the CPU, and the same data is retransmitted by hardware.

If the transmit data is not written in time during the period of writing the next data, a communication error interrupt occurs due to occurrence of underrun, and communication ends midway.

(3) Recommunication processing

The vectored interrupt servicing in <2> of Figure 17-29 judges whether the data has been correctly transmitted within one frame. If the data has not been correctly transmitted (if the number of data to be transmitted in one frame could not be transmitted), the data must be retransmitted in the next frame, or the remainder of the data must be transmitted.

17.6.2 Master reception

★ Before performing master reception, it is necessary to notify the slave unit of slave transmission. Therefore, more than two communication frames are necessary for master reception.

The slave unit prepares the transmit data, sets (1) the slave transmission enable flag (ENSLVTX), and waits.

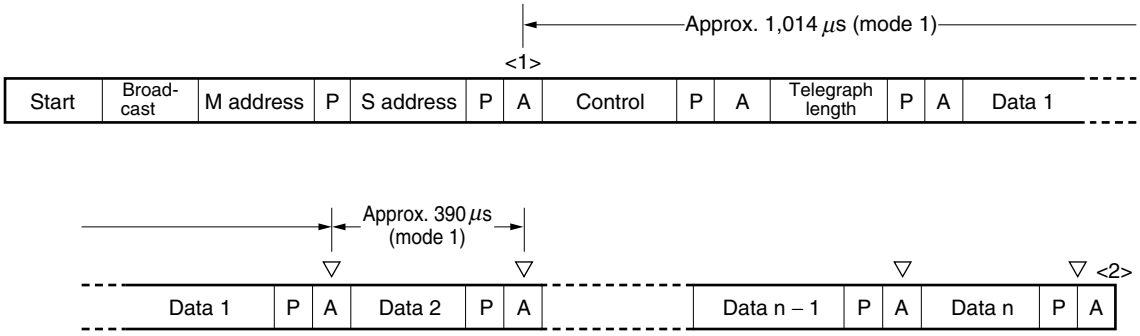
Initial preparation processing:

Set a unit address, slave address, and control data.

Communication start processing:

Set the bus control register (enable communication and master request).

Figure 17-30. Master Reception



<1> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing
↓
Judgment of slave request → Slave reception processing
↓
Judgment of conflict result → Remaster request processing

<2> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing
↓
Judgment of end of communication → End of communication processing
↓
Judgment of end of frame → Frame end processing (See **17.6.2 (2) Frame end processing**)

Remarks 1. ▽: Interrupt (INTIE1) occurrence (See **17.6.2 (1) Interrupt (INTIE1) occurrence**)
The receive data stored in the IEBus data register (DR) is read by software.
At this time, the data transfer direction is SFR (peripheral) → RAM (memory).

2. n = Final number of data bytes

(1) Interrupt (INTIE1) occurrence

If NACK is transmitted (hardware processing) in the data field, an interrupt (INTIE1) is not issued to the CPU, and the same data is retransmitted from the slave.

If the receive data is not read by the time the next data is received, the hardware automatically transmits NACK.

(2) Frame end processing

The vectored interrupt servicing in <2> of Figure 17-30 judges whether the data has been correctly received within one frame. If the data has not been correctly received (if the number of data to be received in one frame could not be received), a request to retransmit the data must be made to the slave in the next communication frame.

17.6.3 Slave transmission

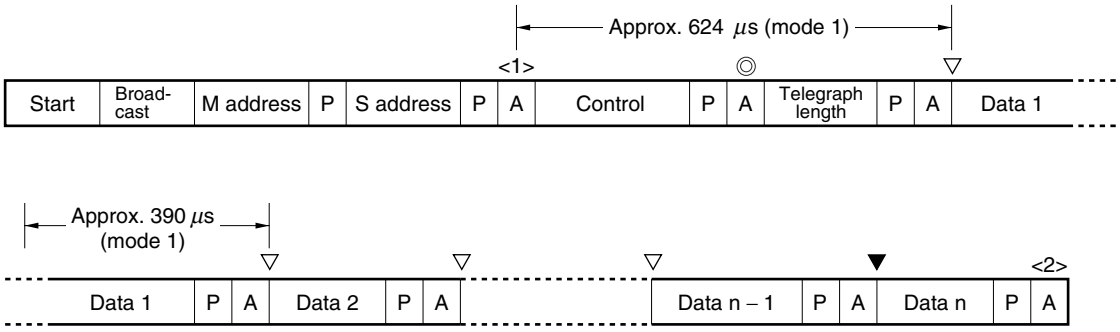
Initial preparation processing:

Set a unit address, telegraph length, and the first byte of the transmit data.

Communication start processing:

Set the bus control register (enable communication, slave transmission, and slave reception).

Figure 17-31. Slave Transmission



<1> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of slave request

<2> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Frame end processing (See **17.6.3 (2) Frame end processing**)

Remarks 1. ▽: Interrupt (INTIE1) occurrence (See **17.6.3 (1) Interrupt (INTIE1) occurrence**). The transmit data of the second and subsequent bytes are written to the IEBus data register (DR) by software. At this time, the data transfer direction is RAM (memory) → SFR (peripheral).

2. ▼: An interrupt (INTIE1) does not occur.

3. ◎: An interrupt (INTIE2) occurs only when 0H, 4H, 5H, or 6H is received in the control field in the slave status (for the slave status return operation in the locked state, refer to **17.4.2 (5) IEBus control data register (CDR)**).

4. n = Final number of data bytes

(1) Interrupt (INTIE1) occurrence

If NACK is received from the master in the data field, an interrupt (INTIE1) is not issued to the CPU, and the same data is retransmitted by hardware.

If the transmit data is not written in time during the period of writing the next data, a communication error interrupt occurs due to occurrence of underrun, and communication is abnormally ended.

(2) Frame end processing

The vectored interrupt servicing in <2> of Figure 17-31 judges whether the data has been correctly transmitted within one frame. If the data has not been correctly transmitted (if the number of data to be transmitted in one frame could not be transmitted), the data must be retransmitted in the next frame, or the remainder of the data must be transmitted.

17.6.4 Slave reception

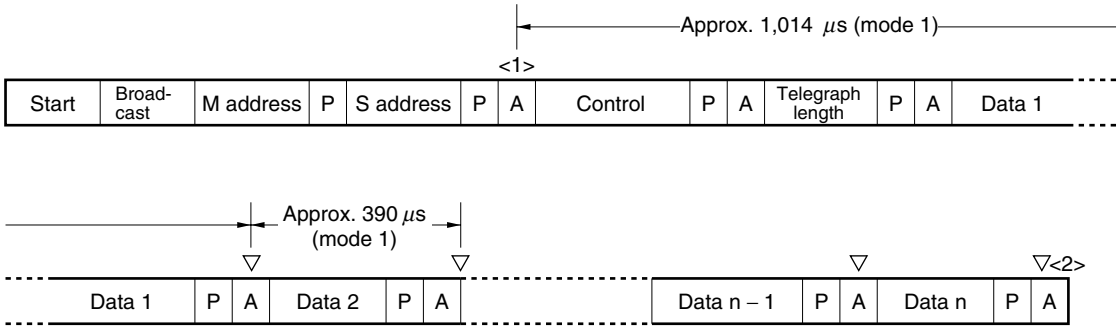
Initial preparation processing:

Set a unit address.

Communication start processing:

Set the bus control register (enable communication, disable slave transmission, and enable slave reception).

Figure 17-32. Slave Reception



<1> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of slave request → Slave processing

<2> Interrupt (INTIE2) occurrence

Judgment of occurrence of error → Error processing

↓

Judgment of end of communication → End of communication processing

↓

Judgment of end of frame → Frame end processing (See **17.6.4 (2) Frame end processing**)

Remarks 1. ▽: Interrupt (INTIE1) occurrence (See **17.6.4 (1) Interrupt (INTIE1) occurrence**).
The receive data stored in the IEBus data register (DR) is read by software.
At this time, the data transfer direction is SFR (peripheral) → RAM (memory).

2. n = Final number of data bytes

(1) Interrupt (INTIE1) occurrence

If NACK is transmitted in the data field, an interrupt (INTIE1) is not issued to the CPU, and the same data is retransmitted from the master.

If the receive data is not read by the time the next data is received, NACK is automatically transmitted.

(2) Frame end processing

The vectored interrupt servicing in <2> of Figure 17-32 judges whether the data has been correctly received within one frame.

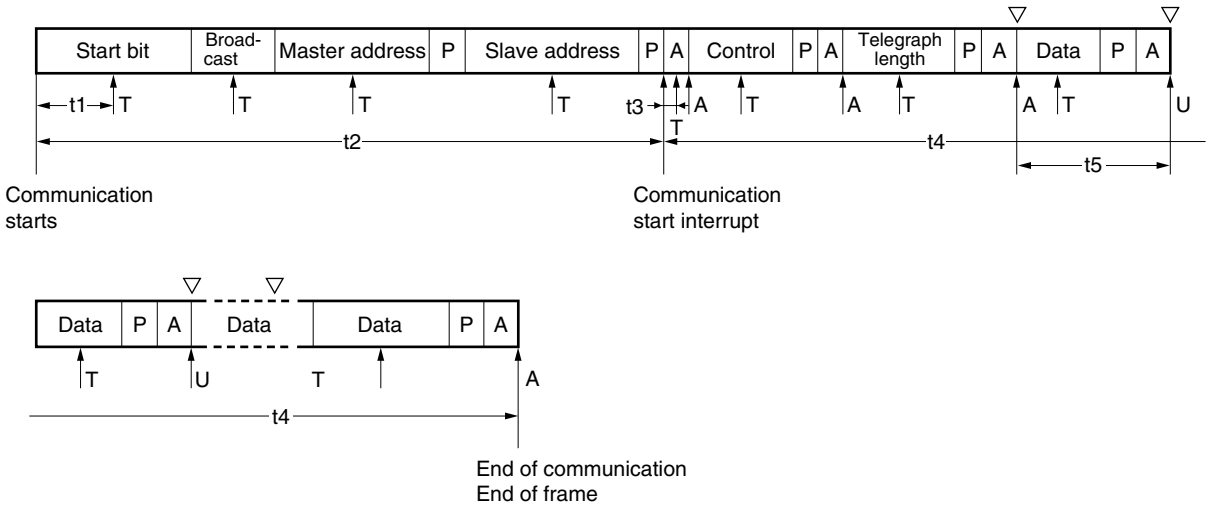
17.6.5 Interval of occurrence of interrupt for IEBus control

Each control interrupt must occur at each point of communication and perform the necessary processing until the next interrupt occurs. Therefore, the CPU must control the IEBus control block, taking the shortest time of this interrupt into consideration.

The locations at which the following error interrupts may occur are indicated by ↑ in the field where it may occur. ↑ does not mean that the interrupt occurs at each of the points indicated by ↑. If an error interrupt (timing error, parity error, NACK reception, underrun error, or overrun error) occurs, the IEBus internal circuit is initialized. As a result, subsequent interrupts do not occur in that communication frame.

(1) Master transmission

Figure 17-33. Master Transmission (Interval of Interrupt Occurrence)



Remarks 1. T: Timing error
A: NACK reception
U: Underrun error
▽: Data set interrupt (INTIE1)

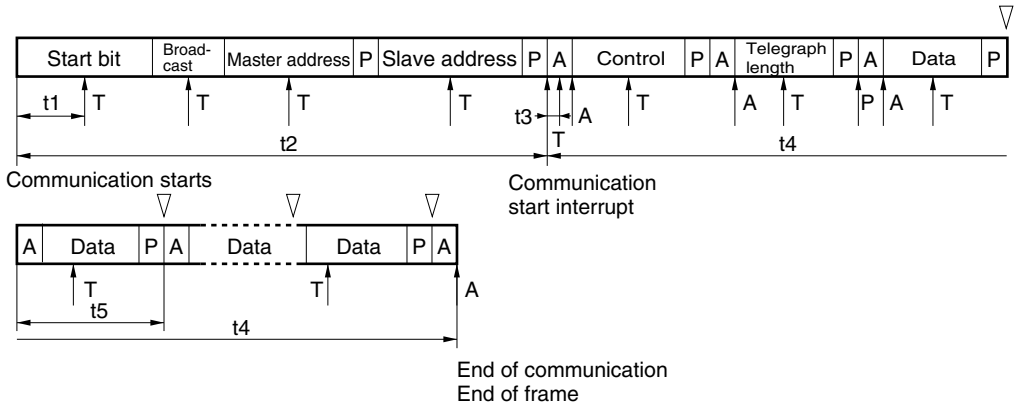
2. End of frame occurs at the end of 32-byte data.

(IEBus: 6.29 MHz operation)

Item	Symbol	MIN.	Unit
Communication starts - timing error	t1	Approx. 93	μs
Communication starts - communication start interrupt	t2	Approx. 1,282	μs
Communication start interrupt - timing error	t3	Approx. 15	μs
Communication start interrupt - end of communication	t4	Approx. 1,012	μs
Transmission data request interrupt interval	t5	Approx. 375	μs

(2) Master reception

Figure 17-34. Master Reception (Interval of Interrupt Occurrence)



Remarks 1. T: Timing error
P: Parity error
A: NACK reception
▽: Data set interrupt (INTIE1)

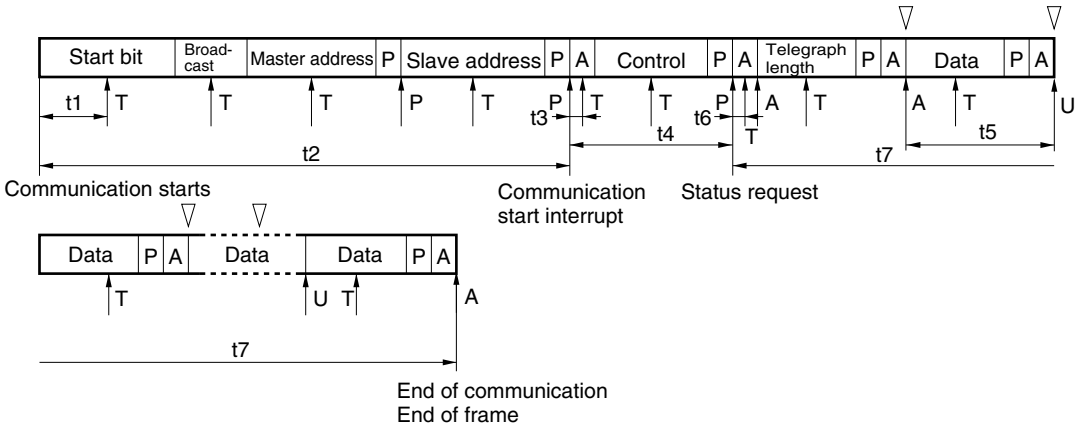
2. End of frame occurs at the end of 32-byte data.

(IEBus: 6.29 MHz operation)

Item	Symbol	MIN.	Unit
Communication starts - timing error	t1	Approx. 93	μs
Communication starts - communication start interrupt	t2	Approx. 1,282	μs
Communication start interrupt - timing error	t3	Approx. 15	μs
Communication start interrupt - end of communication	t4	Approx. 1,012	μs
Receive data read interval	t5	Approx. 375	μs

(3) Slave transmission

Figure 17-35. Slave Transmission (Interval of Interrupt Occurrence)



Remarks 1. T: Timing error
P: Parity error
A: NACK reception
U: Underrun error
▽: Data set interrupt (INTIE1)

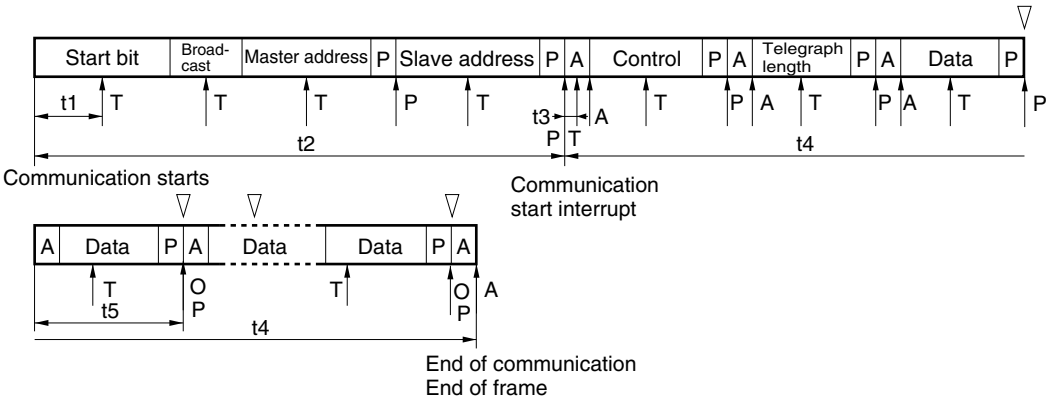
2. End of frame occurs at the end of 32-byte data.

(IEBus: 6.29 MHz operation)

Item	Symbol	MIN.	Unit
Communication starts - timing error	t1	Approx. 96	μs
Communication starts - communication start interrupt	t2	Approx. 1,192	μs
Communication start interrupt - timing error	t3	Approx. 15	μs
Communication start interrupt - status request	t4	Approx. 225	μs
Transmission data request interrupt interval	t5	Approx. 375	μs
Status request - timing error	t6	Approx. 15	μs
Status request - end of communication	t7	Approx. 787	μs

(4) Slave reception

Figure 17-36. Slave Reception (Interval of Interrupt Occurrence)



Remarks 1. T: Timing error
P: Parity error
A: NACK reception
O: Overrun error
▽: Data set interrupt (INTIE1)

2. End of frame occurs at the end of 32-byte data.

(IEBus: 6.29 MHz operation)

Item	Symbol	MIN.	Unit
Communication starts - timing error	t1	Approx. 96	μs
Communication starts - communication start interrupt	t2	Approx. 1,192	μs
Communication start interrupt - timing error	t3	Approx. 15	μs
Communication start interrupt - end of communication	t4	Approx. 1,012	μs
Receive data read interval	t5	Approx. 375	μs

CHAPTER 18 INTERRUPT FUNCTIONS

18.1 Interrupt Function Types

The following three types of interrupt functions are used.

(1) Non-maskable interrupt

This interrupt is acknowledged unconditionally even if interrupts are disabled. It does not undergo interrupt priority control and is given top priority over all other interrupt requests.

It generates a standby release signal.

One interrupt source from the watchdog timer is incorporated as a non-maskable interrupt.

(2) Maskable interrupts

These interrupts undergo mask control. Maskable interrupts can be divided into a high-priority interrupt group and a low-priority interrupt group by setting the priority specification flag register (PR).

Multiple interrupt servicing can be applied to low-priority interrupts when high-priority interrupts are generated.

If two or more interrupts with the same priority are generated simultaneously, each interrupt is serviced according to a predetermined priority (refer to Table 18-1).

A standby release signal is generated.

Maskable interrupts are provided in each product as follows.

- μPD178076, 178078 Internal: 13, external: 8
- μPD178096A, 178098A Internal: 12, external: 8
- μPD178F098 Internal: 15, external: 8

(3) Software interrupt

This is a vectored interrupt generated by executing the BRK instruction. It is acknowledged even in an disabled interrupt state. The software interrupt does not undergo interrupt priority control.

18.2 Interrupt Sources and Configuration

The μPD178076 and 178078 have a total of 22 interrupt sources, including non-maskable, maskable, and software interrupts. The μPD178096A and 178098A have a total of 21 interrupt sources, and the μPD178F098 has a total of 24 sources.

Remark Either a non-maskable interrupt or a maskable interrupt (internal) can be selected as the interrupt source of the watchdog timer (INTWDT).

Table 18-1. Interrupt Sources (1/6)

(1) μPD178076, 178078 (1/2)

Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	–	INTWDT	Overflow of watchdog timer (when watchdog timer mode 1 is selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Overflow of watchdog timer (when interval timer mode is selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTP4			000EH	
	6	INTP5			0010H	
	7	INTP6			0012H	
	8	INTP7			0014H	
	9	INTCSI0	End of transfer by serial interface SIO0	Internal	0016H	(B)
	10	INTCSI1	End of transfer by serial interface SIO1		0018H	
	11	INTCSI3	End of transfer by serial interface SIO3		001AH	
	12	INTTM50	Generation of match signal of 8-bit timer/event counter 50		001CH	
	13	INTTM51	Generation of match signal of 8-bit timer/event counter 51		001EH	
	14	INTSER0	Reception error of serial interface UART0		0020H	
	15	INTSR0	End of reception by serial interface UART0		0022H	
	16	INTST0	End of transmission by serial interface UART0		0024H	
	17	INTBTM0	Generation of coincidence signal of basic timer BTM0		0026H	
	18	INTTM00	Generation of signal indicating match between 16-bit timer counter 0 (TM0) and capture/compare register 00 (CR00) (when CR00 is used as compare register)		0028H	
			Detection of input edge of TI00/P32 pin (when CR00 is used as capture register)	External		(D)

Notes 1. If two or more maskable interrupts occur at the same time, they are acknowledged or held pending according to their default priorities. A default priority of 0 is the highest, while 22 is the lowest.

2. (A) to (E) under the heading Basic Configuration Type correspond to (A) to (E) in Figure 18-1.

Table 18-1. Interrupt Sources (2/6)

(1) μPD178076, 178078 (2/2)

Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Maskable	19	INTTM01	Generation of signal indicating match between 16-bit timer counter 0 (TM0) and capture/compare register 01 (CR01) (when CR01 is used as compare register)	Internal	002AH	(B)
			Detection of input edge of TI01/P33 pin (when CR01 is used as capture register)	External		(D)
	20	–	–	–	Note 3	–
	21	–	–		Note 3	
	22	INTAD	End of conversion by A/D converter	Internal	0030H	(B)
Software	–	BRK	Execution of BRK instruction	–	003EH	(E)

Notes 1. If two or more maskable interrupts occur at the same time, they are acknowledged or held pending according to their default priorities. A default priority of 0 is the highest, while 22 is the lowest.

2. (A) to (E) under the heading Basic Configuration Type correspond to (A) to (E) in Figure 18-1.

3. There are no interrupt sources corresponding to vector addresses 002CH and 002EH.

Table 18-1. Interrupt Sources (3/6)

(2) μPD178096A, 178098A (1/2)

Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	–	INTWDT	Overflow of watchdog timer (when watchdog timer mode 1 is selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Overflow of watchdog timer (when interval timer mode is selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTP4			000EH	
	6	INTP5			0010H	
	7	INTP6			0012H	
	8	INTP7			0014H	
	9	INTCSI0	End of transfer by serial interface SIO0	Internal	0016H	(B)
	10	INTCSI1	End of transfer by serial interface SIO1		0018H	
	11	INTCSI3	End of transfer by serial interface SIO3		001AH	
	12	INTTM50	Generation of match signal of 8-bit timer/event counter 50		001CH	
	13	INTTM51	Generation of match signal of 8-bit timer/event counter 51		001EH	
	14	–	–	–	Note 3	–
	15	–	–		Note 3	
	16	–	–		Note 3	
	17	INTBTM00	Generation of match signal of basic timer BTM0	Internal	0026H	(B)
	18	INTTM00	Generation of signal indicating match between 16-bit timer counter 0 (TM0) and capture/compare register 00 (CR00) (when CR00 is used as compare register)		0028H	
			Detection of input edge of TI00/P32 pin (when CR00 is used as capture register)	External		(D)

Notes 1. If two or more maskable interrupts occur at the same time, they are acknowledged or held pending according to their default priorities. A default priority of 0 is the highest, while 22 is the lowest.

2. (A) to (E) under the heading Basic Configuration Type correspond to (A) to (E) in Figure 18-1.

3. There are no interrupt sources corresponding to vector addresses 0020H, 0022H, and 0024H.

Table 18-1. Interrupt Sources (4/6)

(2) μPD178096A, 178098A (2/2)

Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Maskable	19	INTTM01	Generation of signal indicating match between 16-bit timer counter 0 (TM0) and capture/compare register 01 (CR01) (when CR01 is used as compare register)	Internal	002AH	(B)
			Detection of input edge of TI01/P33 pin (when CR01 is used as capture register)	External		(D)
	20	INTIE1	IEBus0 data access request	Internal	002CH	(B)
	21	INTIE2	IEBus0 communication error and start/end of communication		002EH	
	22	INTAD	End of conversion by A/D converter AD1		0030H	(B)
Software	–	BRK	Execution of BRK instruction	–	003EH	(E)

Notes 1. If two or more maskable interrupts occur at the same time, they are acknowledged or held pending according to their default priorities. A default priority of 0 is the highest, while 22 is the lowest.

2. (A) to (E) under the heading Basic Configuration Type correspond to (A) to (E) in Figure 18-1.

Table 18-1. Interrupt Sources (5/6)

(3) μPD178F098 (1/2)

Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	–	INTWDT	Overflow of watchdog timer (when watchdog timer mode 1 is selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Overflow of watchdog timer (when interval timer mode is selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTP4			000EH	
	6	INTP5			0010H	
	7	INTP6			0012H	
	8	INTP7			0014H	
	9	INTCSI0	End of transfer by serial interface SIO0	Internal	0016H	(B)
	10	INTCSI1	End of transfer by serial interface SIO1		0018H	
	11	INTCSI3	End of transfer by serial interface SIO3		001AH	
	12	INTTM50	Generation of match signal of 8-bit timer/event counter 50		001CH	
	13	INTTM51	Generation of match signal of 8-bit timer/event counter 51		001EH	
	14	INTSER0	Reception error of serial interface UART0		0020H	
	15	INTSR0	End of reception by serial interface UART0		0022H	
	16	INTST0	End of transmission by serial interface UART0		0024H	
	17	INTBTM0	Generation of match signal of basic timer BTM0		0026H	
	18	INTTM00	Generation of signal indicating match between 16-bit timer counter 0 (TM0) and capture/compare register 00 (CR00) (when CR00 is used as compare register)		0028H	
			Detection of input edge of TI00/P32 pin (when CR00 is used as capture register)	External		(D)

Notes 1. If two or more maskable interrupts occur at the same time, they are acknowledged or held pending according to their default priorities. A default priority of 0 is the highest, while 22 is the lowest.

2. (A) to (E) under the heading Basic Configuration Type correspond to (A) to (E) in Figure 18-1.

Table 18-1. Interrupt Sources (6/6)

(3) μPD178F098 (2/2)

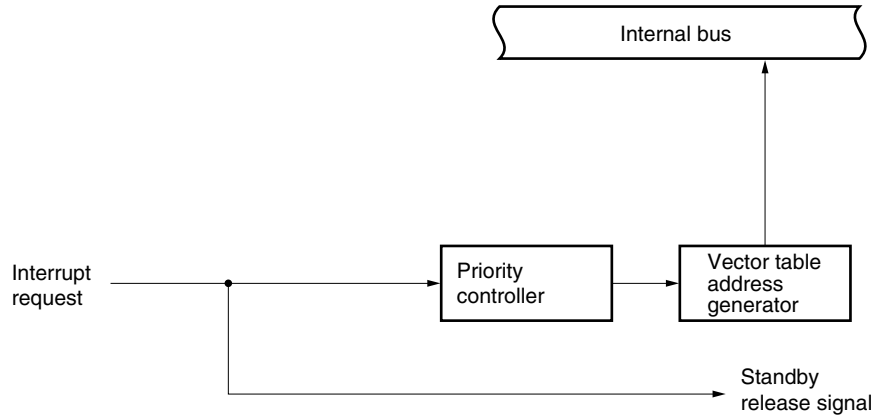
Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Maskable	19	INTTM01	Generation of signal indicating match between 16-bit timer counter 0 (TM0) and capture/compare register 01 (CR01) (when CR01 is used as compare register)	Internal	002AH	(B)
			Detection of input edge of TI01/P33 pin (when CR01 is used as capture register)	External		(D)
	20	INTIE1	IEBus0 data access request	Internal	002CH	(B)
	21	INTIE2	IEBus0 communication error and start/end of communication		002EH	
	22	INTAD	End of conversion by A/D converter AD1		0030H	(B)
Software	–	BRK	Execution of BRK instruction	–	003EH	(E)

Notes 1. If two or more maskable interrupts occur at the same time, they are acknowledged or held pending according to their default priorities. A default priority of 0 is the highest, while 22 is the lowest.

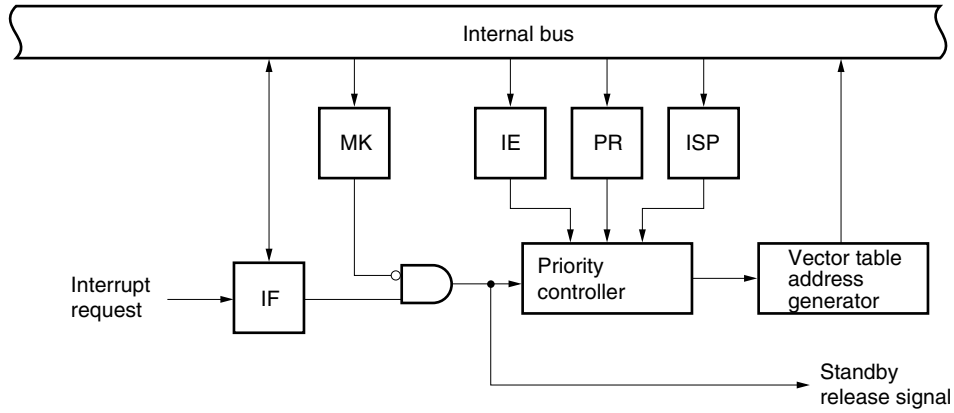
2. (A) to (E) under the heading Basic Configuration Type correspond to (A) to (E) in Figure 18-1.

Figure 18-1. Basic Configuration of Interrupt Function (1/2)

(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt

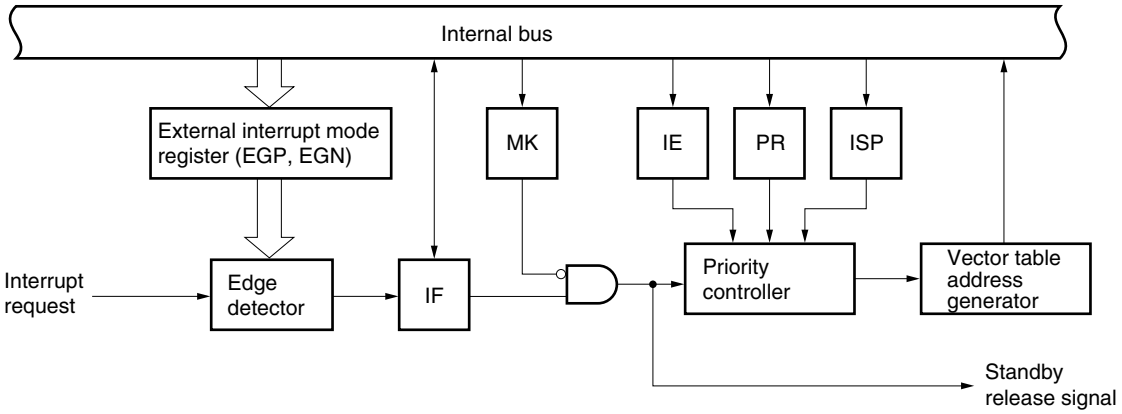
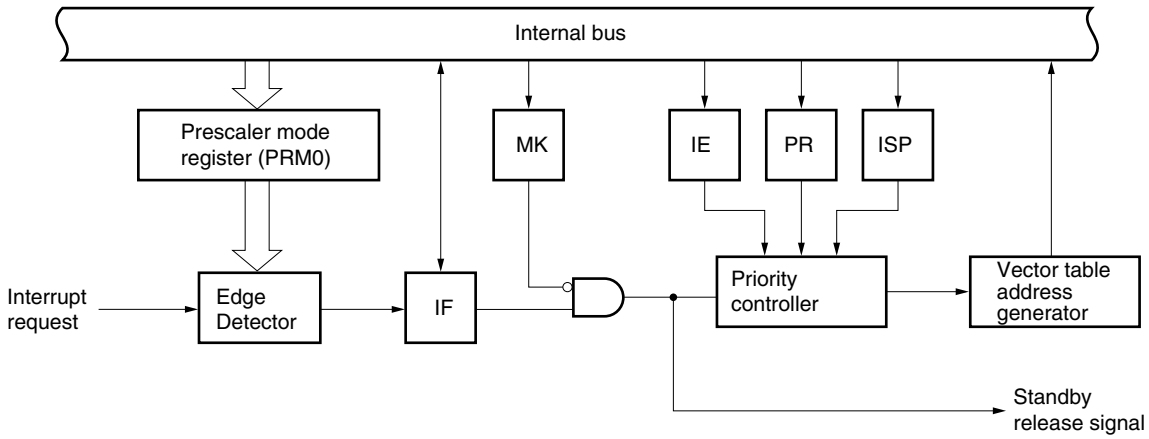
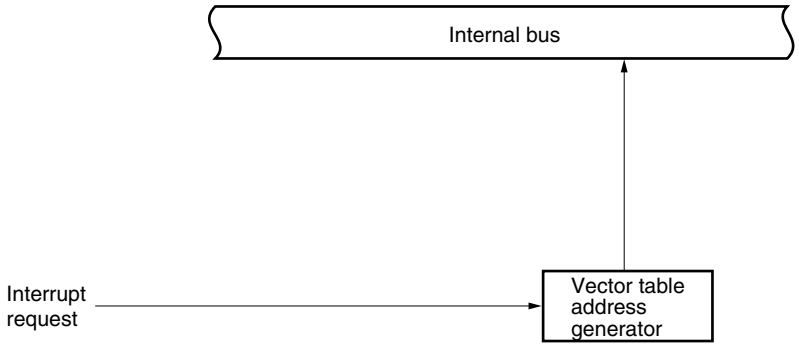


Figure 18-1. Basic Configuration of Interrupt Function (2/2)

(D) External maskable interrupt (INTTM00, INTTM01)



(E) Software interrupt



Remark IF: Interrupt request flag
IE: Interrupt enable flag
ISP: In-service priority flag
MK: Interrupt mask flag
PR: Priority specification flag

18.3 Interrupt Function Control Registers

The following six types of registers are used to control the interrupt functions.

- Interrupt request flag register (IF0L, IF0H, IF1L)
- Interrupt mask flag register (MK0L, MK0H, MK1L)
- Priority specification flag register (PR0L, PR0H, PR1L)
- External interrupt rising edge enable register (EGP)
- External interrupt falling edge enable register (EGN)
- Program status word (PSW)

Table 18-2 lists the interrupt request flags, interrupt mask flags, and priority specification flags corresponding to interrupt request sources.

Table 18-2. Flags Corresponding to Interrupt Request Sources

Interrupt Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specification Flag	
		Register		Register		Register
INTWDT	WDTIF	IF0L	WDTMK	MK0L	WDTPR	PR0L
INTP0	PIF0		PMK0		PPR0	
INTP1	PIF1		PMK1		PPR1	
INTP2	PIF2		PMK2		PPR2	
INTP3	PIF3		PMK3		PPR3	
INTP4	PIF4		PMK4		PPR4	
INTP5	PIF5		PMK5		PPR5	
INTP6	PIF6		PMK6		PPR6	
INTP7	PIF7	IF0H	PMK7	MK0H	PPR7	PR0H
INTCSI0	CSIIF0		CSIMK0		CSIPR0	
INTCSI1	CSIIF1		CSIMK1		CSIPR1	
INTCSI3	CSIIF3		CSIMK3		CSIPR3	
INTTM50	TMIF50		TMMK50		TMPR50	
INTTM51	TMIF51		TMMK51		TMPR51	
INTSER0[Note 1]	SERIF0[Note 1]		SERMK0[Note 1]		SERPR0[Note 1]	
INTSR0[Note 1]	SRIF0[Note 1]		SRMK0[Note 1]		SRPR0[Note 1]	
INTST0[Note 1]	STIF0[Note 1]	IF1L	STMK0[Note 1]	MK1L	STPR0[Note 1]	PR1L
INTBTM0	BTMIF0		BTMMK0		BTMPR0	
INTTM00	TMIF00		TMMK00		TMPR00	
INTTM01	TMIF01		TMMK01		TMPR01	
INTIE1[Note 2]	IEIF1[Note 2]		IEMK1[Note 2]		IEPR1[Note 2]	
INTIE2[Note 2]	IEIF2[Note 2]		IEMK2[Note 2]		IEPR2[Note 2]	
INTAD	ADIF		ADMK		ADPR	

Notes 1. μPD178076, 178078, 178F098 only

2. μPD178096A, 178098A, 178F098 only

(1) Interrupt request flag registers (IF0L, IF0H, IF1L)

The interrupt request flags are set to 1 when the corresponding interrupt request is generated or an instruction is executed. They are cleared to 0 when an instruction is executed upon acknowledgment of an interrupt request or upon application of reset input.

IF0L, IF0H and IF1L are set by a 1-bit or 8-bit memory manipulation instruction. If IF0L and IF0H are used as the 16-bit register IF0, use a 16-bit memory manipulation instruction for setting.

Reset input sets these registers to 00H.

Figure 18-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
IF0L	PIF6	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0	WDTIF	FFE0H	00H	R/W
	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
IF0H	SRIF0 Note 1	SERIF0 Note 1	TMIF51	TMIF50	CSIIF3	CSIIF1	CSIIF0	PIF7	FFE1H	00H	R/W
	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
IF1L	0	ADIF	IEIF2 Note 2	IEIF1 Note 2	TMIF01	TMIF00	BTMIF0	STIF0 Note 1	FFE2H	00H	R/W

××ID×	Interrupt request flag
0	No interrupt request signal
1	Interrupt request signal is generated; Interrupt request state

Notes 1. These bits are provided in the μPD178076, 178078, and 178F098 only. Be sure to reset these bits to 0 in the μPD178096A and 178098A.

2. These bits are provided in the μPD178096A, 178098A, and 178F098 only. Be sure to reset these bits to 0 in the μPD178076 and 178078.

Cautions 1. The WDTIF flag is R/W enabled only when the watchdog timer is used as an interval timer. If the watchdog timer is used in watchdog timer mode 1, set the WDTIF flag to 0.

2. To operate the timers, serial interface, and A/D converter after the standby mode has been released, clear the interrupt request flag, because the interrupt request flag may be set by noise.

(2) Interrupt mask flag registers (MK0L, MK0H, MK1L)

The interrupt mask flags are used to enable/disable the corresponding maskable interrupt servicing and to set standby clear enable/disable.

MK0L, MK0H and MK1L are set by a 1-bit or 8-bit memory manipulation instruction. If MK0L and MK0H are used as the 16-bit register MK0, use a 16-bit memory manipulation instruction for setting.

Reset input sets these registers to FFH.

Figure 18-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
MK0L	PMK6	PMK5	PMK4	PMK3	PMK2	PMK1	PMK0	WDTMK	FFE4H	FFH	R/W
	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
MK0H	SRMK0 Note 1	SERMK0 Note 1	TMMK51	TMMK50	CSIMK3	CSIMK1	CSIMK0	PMK7	FFE5H	FFH	R/W
	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
MK1L	1	ADMK	IEMK2 Note 2	IEMK1 Note 2	TMMK01	TMMK00	BTMMK0	STMK0 Note 1	FFE6H	FFH	R/W

××MK×	Control of interrupt servicing
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Notes 1. These bits are provided in the μPD178076, 178078, and 178F098 only. Be sure to reset these bits to 0 in the μPD178096A and 178098A.

2. These bits are provided in the μPD178096A, 178098A, and 178F098 only. Be sure to reset these bits to 0 in the μPD178076 and 178078.

Cautions 1. If the WDTMK flag is read when the watchdog timer is used in watchdog timer mode 1, the MK0 value becomes undefined.

2. Because port 0 has an alternate function as an external interrupt request input, when the output level is changed by specifying the output mode of the port function, an interrupt request flag is set. Therefore, the interrupt mask flag should be set to 1 before using the output mode.

(3) Priority specification flag registers (PR0L, PR0H, PR1L)

The priority specification flags are used to set the corresponding maskable interrupt priority order.

PR0L, PR0H and PR1L are set by a 1-bit or 8-bit memory manipulation instruction. If PR0L and PR0H are used as the 16-bit register PR0, use a 16-bit memory manipulation instruction for setting.

Reset input sets these registers to FFH.

Figure 18-4. Format of Priority Specification Flag Registes (PR0L, PR0H, PR1L)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
PR0L	PPR6	PPR5	PPR4	PPR3	PPR2	PPR1	PPR0	WDTPR	FFF8H	FFH	R/W
	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
PR0H	SRPR0 Note 1	SERPR0 Note 1	TMPR51	TMPR50	CSIPR3	CSIPR1	CSIPR0	PPR7	FFE9H	FFH	R/W
	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
PR1L	1	ADPR	IEPR2 Note 2	IEPR1 Note 2	TMPR01	TMPR00	BTMPR0	STPR0 Note 1	FFEAH	FFH	R/W

××PR×	Priority level selection
0	High priority level
1	Low priority level

Notes 1. These bits are provided in the μPD178076, 178078, and 178F098 only. Be sure to set these bits to 1 in the μPD178096A and 178098A.

2. These bits are provided in the μPD178096A, 178098A, and 178F098 only. Be sure to set these bits to 1 in the μPD178076 and 178078.

Caution When the watchdog timer is used in watchdog timer mode 1, set the WDTPR flag to 1.

(4) External interrupt rising edge enable register (EGP), external interrupt falling edge enable register (EGN)

These registers set the valid edge for INTP0 to INTP7.

EGP and EGN are set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets these registers to 00H.

Figure 18-5. Format of External Interrupt Rising Edge Enable Register (EGP) and External Interrupt Falling Edge Enable Register (EGN)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
EGP	EGP7	EGP6	EGP5	EGP4	EGP3	EGP2	EGP1	EGP0	FF48H	00H	R/W
	7	6	5	4	3	2	1	0			
EGN	EGN7	EGN6	EGN5	EGN4	EGN3	EGN2	EGN1	EGN0	FF49H	00H	R/W

EGPn	EGNn	Selection of INTPn pin valid edge (n = 0 to 7)
0	0	Interrupts disabled
0	1	Falling edge
1	0	Rising edge
1	1	Both falling and rising edges

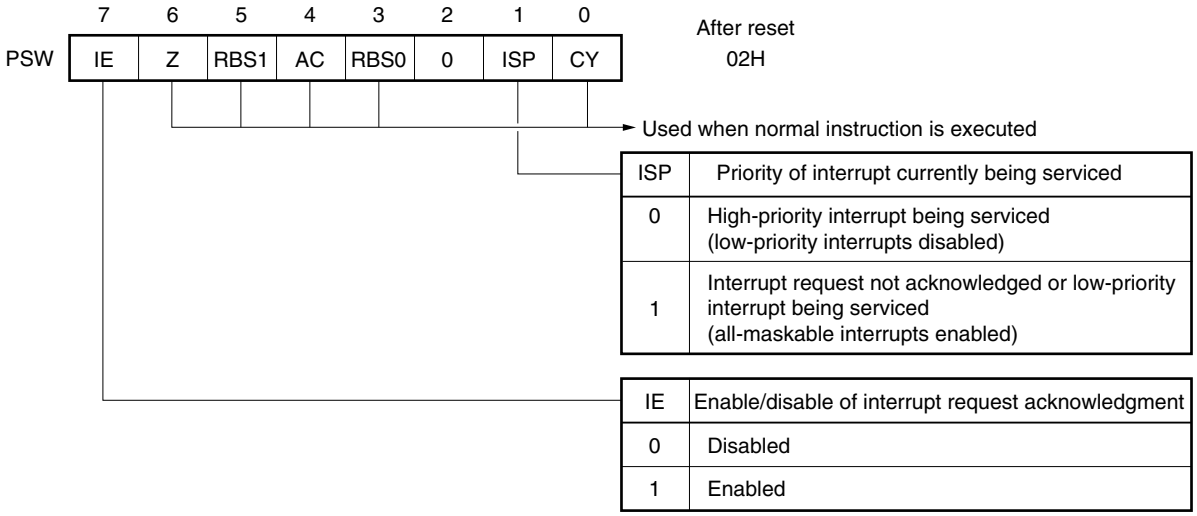
(5) Program status word (PSW)

The program status word is a register used to hold the instruction execution result and the current status of an interrupt request. The IE flag to set maskable interrupt enable/disable and the ISP flag to control nesting interrupt are mapped in the PSW.

Besides 8-bit unit read/write, this register can carry out operations via bit manipulation instructions and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged, if the BRK instruction is executed, the contents of the PSW are automatically saved into a stack and the IE flag is reset to 0. If a maskable interrupt request is acknowledged, the contents of the priority specification flag of the acknowledged interrupt are transferred to the ISP flag. The acknowledged interrupt is also saved into the stack by the PUSH PSW instruction and restored from the stack by the RETI, RETB, and POP PSW instructions.

Reset input sets the PSW to 02H.

Figure 18-6. Configuration of Program Status Word



ISP	Priority of interrupt currently being serviced
0	High-priority interrupt being serviced (low-priority interrupts disabled)
1	Interrupt request not acknowledged or low-priority interrupt being serviced (all-maskable interrupts enabled)

IE	Enable/disable of interrupt request acknowledgment
0	Disabled
1	Enabled

18.4 Interrupt Servicing Operations

18.4.1 Non-maskable interrupt request acknowledgment operation

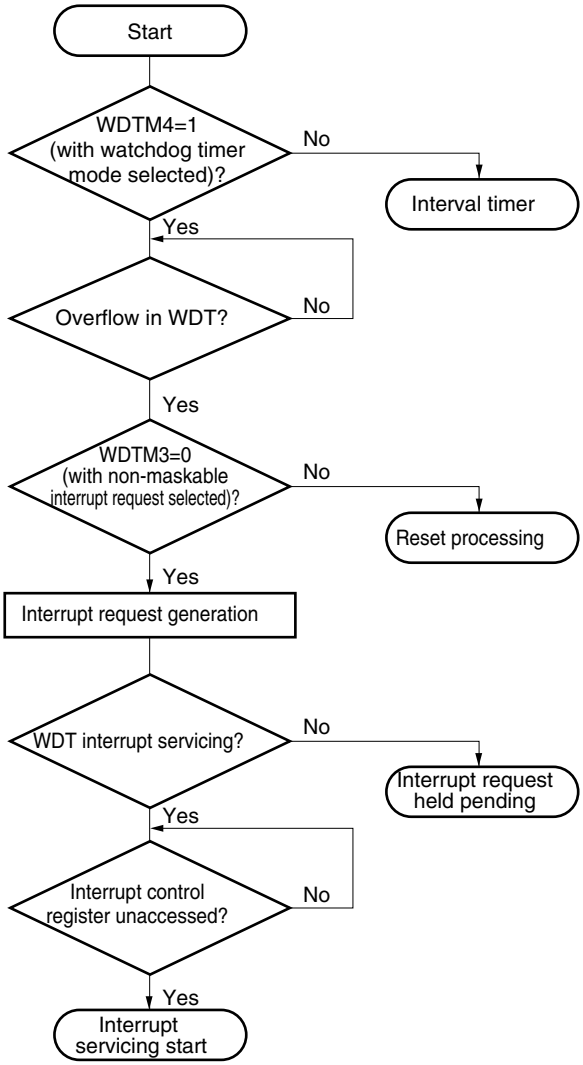
A non-maskable interrupt request is unconditionally acknowledged even if in an interrupt request acknowledgment disabled state. It does not undergo interrupt priority control and has the highest priority of all interrupts.

If a non-maskable interrupt request is acknowledged, the acknowledged interrupt is saved in the stack, the program status word (PSW), and the program counter (PC), in that order, the IE and ISP flags are reset to 0, and the vector table contents are loaded into the PC and branched.

A new non-maskable interrupt request generated during execution of a non-maskable interrupt servicing program is acknowledged after execution of the current non-maskable interrupt servicing program has finished (following RETI instruction execution) and one main routine instruction is executed. If a new non-maskable interrupt request is generated twice or more during non-maskable interrupt servicing program execution, only one non-maskable interrupt request is acknowledged after termination of the non-maskable interrupt servicing program execution.

Figure 18-7 shows the flowchart from generation of a non-maskable interrupt request to acknowledgment. Figure 18-8 shows the timing of acknowledging the non-maskable interrupt request, and Figure 18-9 shows the operation performed if a more than one non-maskable interrupt request occurs.

Figure 18-7. Flowchart from Generation of Non-Maskable Interrupt Request to Acknowledgment



WDTM: Watchdog timer mode register
WDT: Watchdog timer

Figure 18-8. Non-Maskable Interrupt Request Acknowledgment Timing

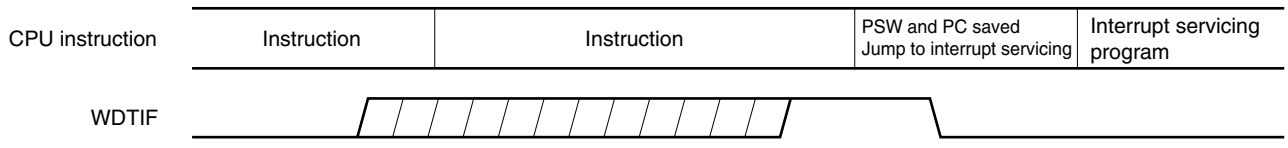
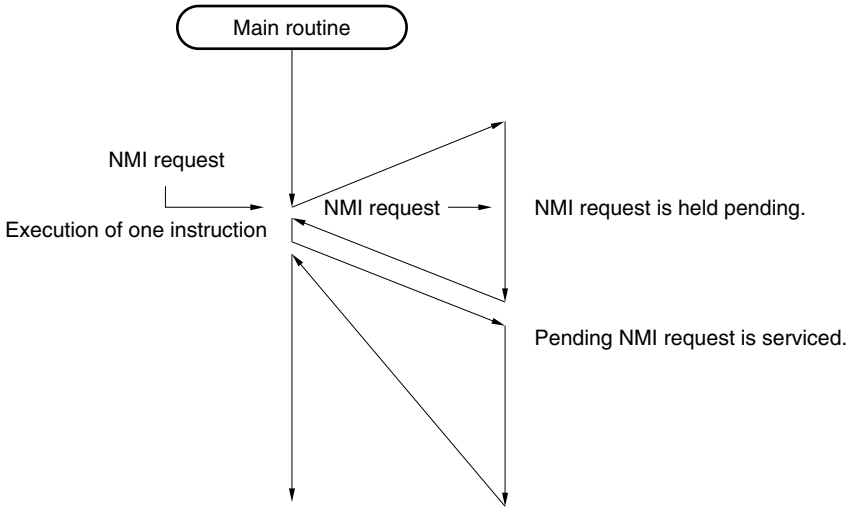
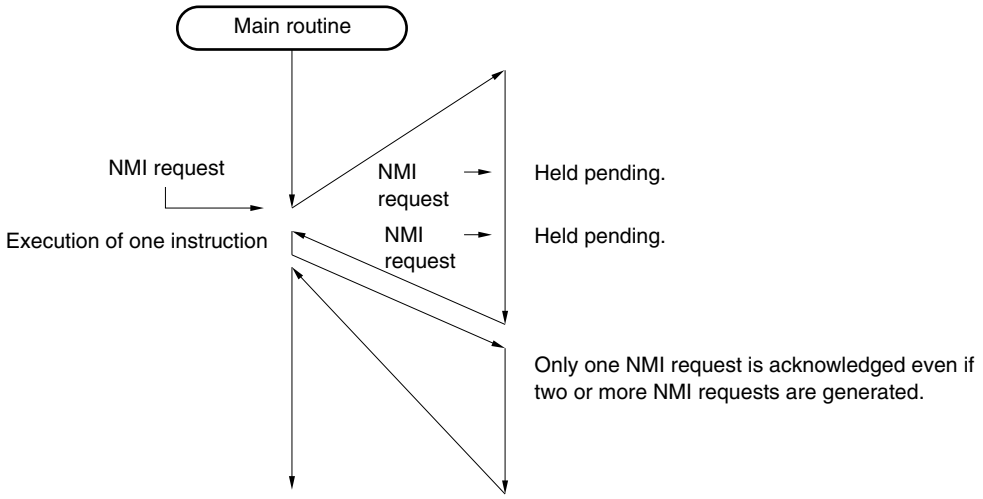


Figure 18-9. Non-Maskable Interrupt Request Acknowledgment Operation

(a) If a new non-maskable interrupt request is generated during non-maskable interrupt servicing program execution



(b) If two non-maskable interrupt requests are generated during non-maskable interrupt servicing program execution



18.4.2 Maskable interrupt request acknowledgment operation

A maskable interrupt request becomes acknowledgeable when an interrupt request flag is set to 1 and the mask (MK) flag of the interrupt request is cleared to 0. A vectored interrupt request is acknowledged in an interrupt enabled state (with the IE flag set to 1). However, a low-priority interrupt request is not acknowledged during high-priority interrupt servicing (with the ISP flag reset to 0).

The wait times from maskable interrupt request generation to interrupt request servicing are as follows.

For the interrupt acknowledgment timing, refer to Figures 18-11 and 18-12.

Table 18-3. Times from Maskable Interrupt Request Generation to Interrupt Servicing

	Minimum Time	Maximum Time **Note**
When ××PR× = 0	7 clocks	32 clocks
When ××PR× = 1	8 clocks	33 clocks

Note If an interrupt request is generated just before a divide instruction, the wait time is maximized.

Remark 1 clock : $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

If two or more maskable interrupt requests are generated simultaneously, the request specified as having a priority by the priority specification flag is acknowledged first. If two or more requests are specified as having the same priority by the priority specification flag, the default priorities apply.

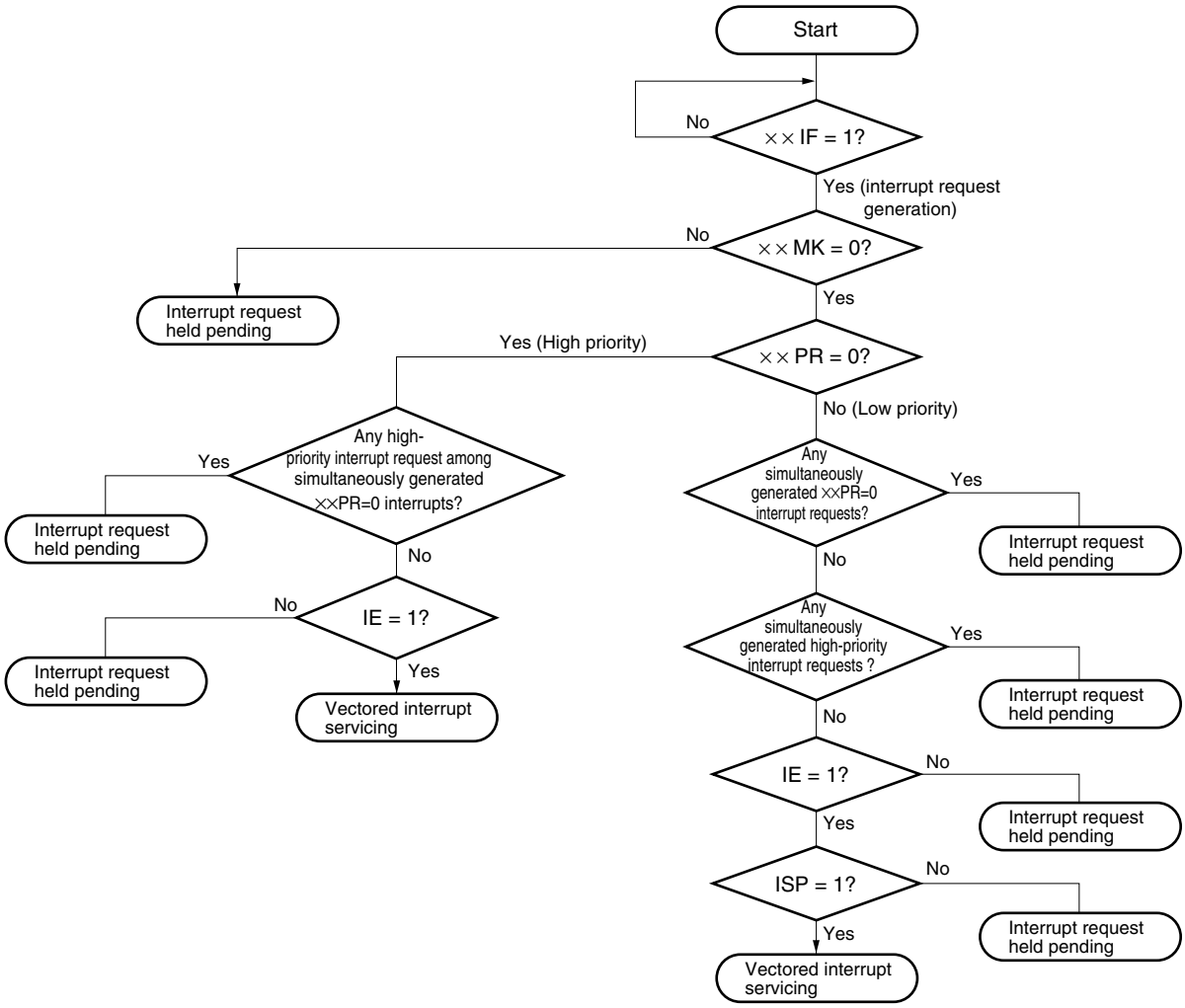
Any pending interrupt requests are acknowledged when they become acknowledgeable.

Figure 18-10 shows the interrupt request acknowledgment algorithms.

If a maskable interrupt request is acknowledged, the acknowledged interrupt request is saved in the stack, the program status word (PSW), and the program counter (PC), in that order, the IE flag is reset to 0, and the acknowledged interrupt priority specification flag contents are transferred to the ISP flag. Further, the vector table data determined for each interrupt request is loaded into the PC and branched.

Return from the interrupt is possible using the RETI instruction.

Figure 18-10. Interrupt Request Acknowledgment Processing Algorithm



××IF: Interrupt request flag

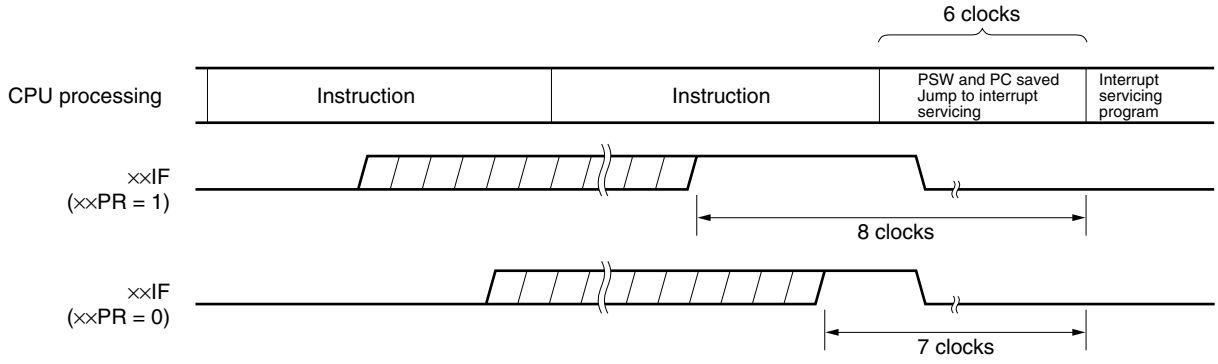
××MK: Interrupt mask flag

××PR: Priority specification flag

IE: Flag controlling acknowledgment of maskable interrupt request (1 = Enabled, 0 = Disabled)

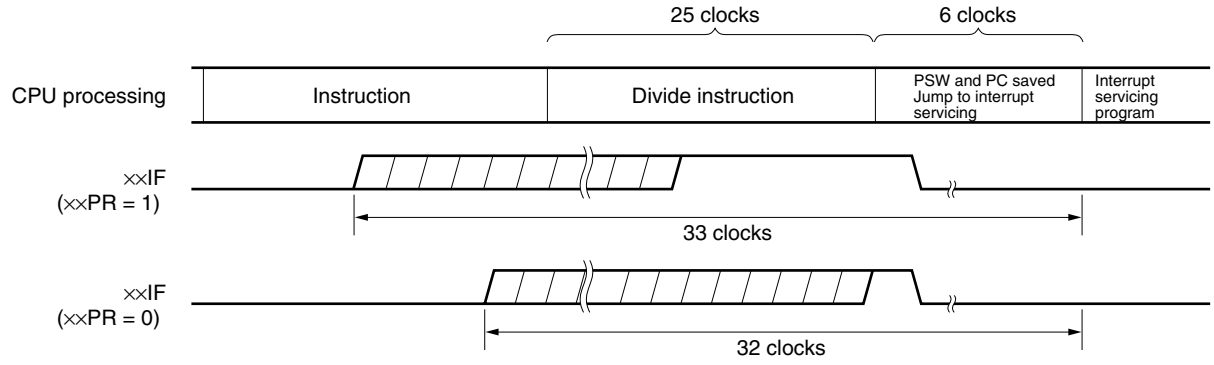
ISP: Flag indicating priority of interrupt currently being serviced (0 = Interrupt with high priority being serviced, 1 = Interrupt request is not acknowledged, or interrupt with low priority is being serviced)

Figure 18-11. Interrupt Request Acknowledgment Timing (Minimum Time)



Remark 1 clock: $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

Figure 18-12. Interrupt Request Acknowledgment Timing (Maximum Time)



Remark 1 clock: $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

18.4.3 Software interrupt request acknowledgment operation

A software interrupt request is acknowledged by BRK instruction execution. Software interrupts cannot be disabled.

If a software interrupt request is acknowledged, it is saved in the stack, the program status word (PSW), and the program counter (PC), in that order, the IE flag is reset to 0, and the contents of the vector tables (003EH and 003FH) are loaded into the PC and branched.

Return from a software interrupt is possible using the RETB instruction.

Caution Do not use the RETI instruction for returning from a software interrupt.

18.4.4 Multiple servicing interrupt

Acknowledgment of an interrupt request while another interrupt is being serviced is called multiple interrupt servicing.

Multiple interrupt servicing does not take place unless interrupts (except the non-maskable interrupt) are enabled to be acknowledged (IE = 1). Acknowledgment of another interrupt request is disabled (IE = 0) when one interrupt has been acknowledged. Therefore, to enable multiple interrupt servicing, the EI flag must be set to 1 during interrupt servicing, to enable other interrupts.

Multiple interrupt servicing may not occur even when interrupts are enabled. This is controlled by the priorities of the interrupts. Although two types of priorities, default priority and programmable priority, may be assigned to an interrupt, multiple interrupt servicing is controlled by using the programmable priority.

If an interrupt with the same priority level as or a higher priority than the interrupt currently being serviced occurs, that interrupt can be acknowledged and serviced. If an interrupt with a priority lower than that of the interrupt currently being serviced occurs, that interrupt cannot be acknowledged and serviced.

An interrupt that is not acknowledged and serviced because it is disabled or it has a low priority is held pending. This interrupt is acknowledged after servicing of the current interrupt has been completed and one instruction of the main routine has been executed.

Multiple interrupt servicing is not enabled while the non-maskable interrupt is being serviced.

Table 18-4 shows the interrupts to which multiple interrupt servicing can be applied, and Figure 18-13 shows an example of multiple interrupt servicing.

Table 18-4. Interrupt Requests Enabled for Multiple Interrupt Servicing

Interrupt Request / Interrupt being serviced		Non-maskable Interrupt Request	Maskable Interrupt Request			
			PR = 0		PR = 1	
			IE = 1	IE = 0	IE = 1	IE = 0
Non-maskable interrupt		D	D	D	D	D
Maskable interrupt	ISP = 0	E	E	D	D	D
	ISP = 1	E	E	D	E	D
Software interrupt servicing		E	E	D	E	D

Remarks 1. E: Multiple interrupt servicing enabled

2. D: Multiple interrupt servicing disabled

3. ISP and IE are the flags contained in the PSW

ISP = 0: An interrupt with higher priority is being serviced

ISP = 1: An interrupt request is not acknowledged or an interrupt with lower priority is being serviced

IE = 0: Interrupt request acknowledgment is disabled

IE = 1: Interrupt request acknowledgment is enabled

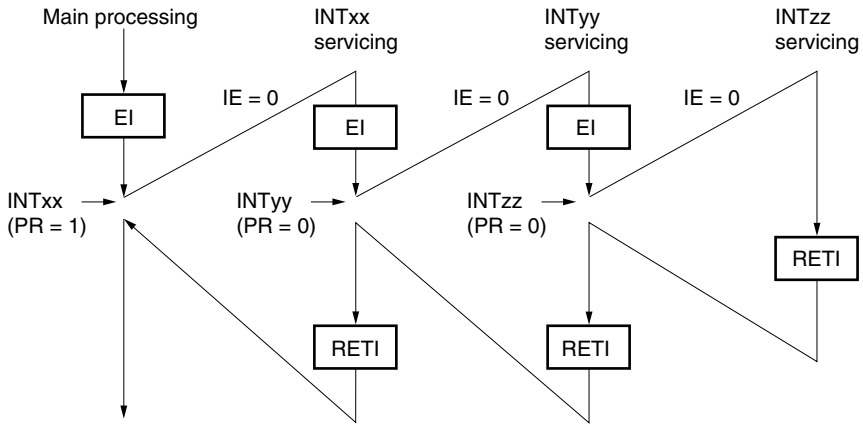
4. PR is a flag contained in PR0L and PR0R.

PR = 0: Higher priority level

PR = 1: Lower priority level

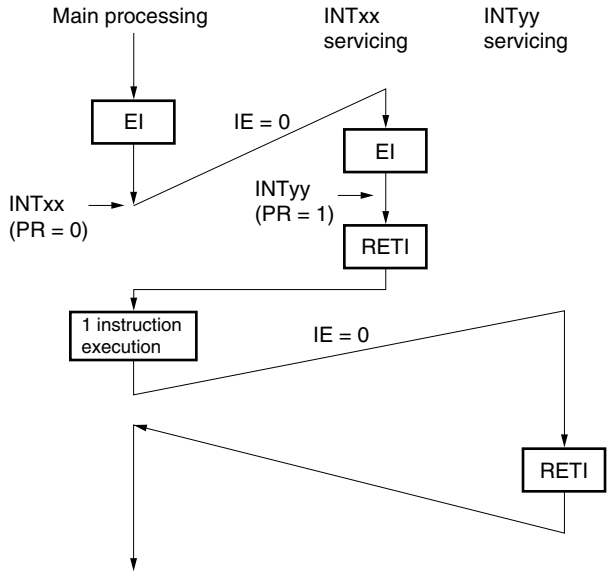
Figure 18-13. Example of Multiple Interrupt Servicing (1/2)

Example 1. Example where multiple interrupt servicing takes place two times



Two interrupt requests, INTyy and INTzz, are acknowledged while interrupt INTxx is being serviced, and multiple interrupt servicing takes place. Before each interrupt request is acknowledged, the EI instruction is always executed, and interrupts are enabled.

Example 2. Example where multiple interrupt servicing does not take place because of priority control



Interrupt request INTyy that is generated while interrupt INTxx is being serviced is not acknowledged because its priority is lower than that of INTxx, and therefore, multiple interrupt servicing does not take place. The INTyy request is held pending, and is acknowledged after one instruction of the main routine has been executed.

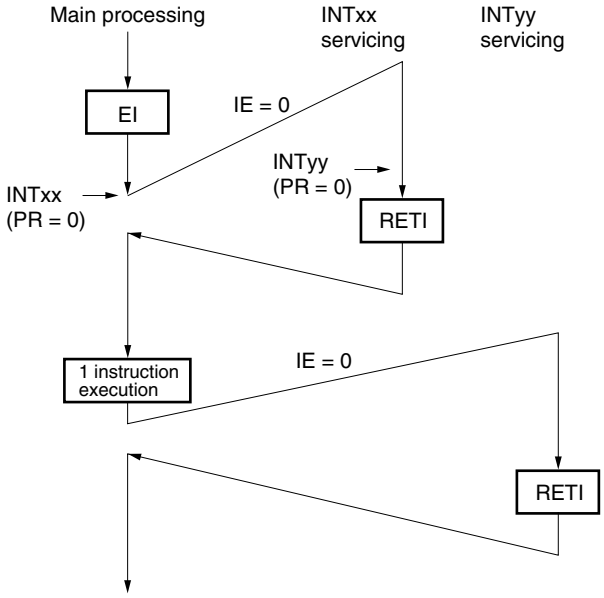
PR = 0: High-priority level

PR = 1: Low-priority level

IE = 0: Acknowledging interrupt requests is disabled.

Figure 18-13. Example of Multiple Interrupt Servicing (2/2)

Example 3. Example where multiple interrupt servicing does not take place because interrupts are not enabled



Because interrupts are not enabled (EI instruction is not issued) in the servicing of interrupt INTxx, interrupt request INTyy is not acknowledged, and multiple interrupt servicing does not take place. The INTyy request is held pending, and is acknowledged after one instruction of the main routine has been executed.

PR = 0: High priority level
IE = 0: Acknowledging interrupts is disabled.

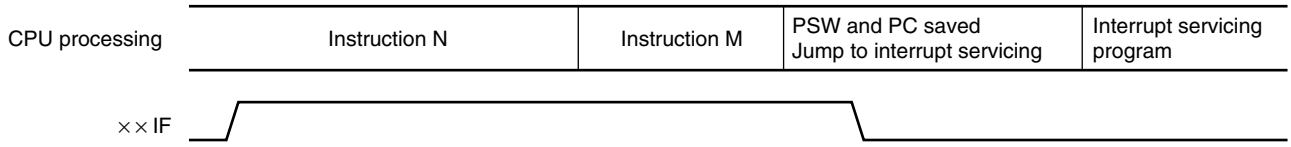
18.4.5 Pending interrupt requests

Even if an interrupt request is generated, the following instructions hold it pending.

- MOV PSW, #byte
- MOV A, PSW
- MOV PSW, A
- MOV1 PSW.bit, CY
- MOV1/AND1/OR1/XOR1 CY, PSW.bit
- SET1/CLR1 PSW.bit
- RETB
- RETI
- PUSH PSW
- POP PSW
- BT/BF/BTCLR PSW.bit, $addr16
- EI
- DI
- Instructions manipulating IF0L, IF0H, IF1L, MK0L, MK0H, MK1L, PR0L, PR0H, PR1L, EGP, and EGN registers

Caution Because the IE flag is cleared to 0 by the software interrupt (caused by execution of the BRK instruction), a maskable interrupt request is not acknowledged even if it occurs while the BRK instruction is being executed. However, the non-maskable interrupt is acknowledged.

Figure 18-14. Pending Interrupt Requests



Remarks 1. Instruction N: Instruction that holds interrupt request pending

2. Instruction M: Instruction that does not hold interrupt request pending

3. Operation of ××IF is not affected by value of ××PR.

CHAPTER 19 PLL FREQUENCY SYNTHESIZER

19.1 Function of PLL Frequency Synthesizer

The PLL (Phase Locked Loop) frequency synthesizer is used to lock the frequency in the MF (Middle Frequency), HF (High Frequency), and VHF (Very High Frequency) ranges to a specific frequency by means of phase difference comparison.

The PLL frequency synthesizer divides the frequency of the signal input from the VCOL or VCOH pin by using a programmable divider, and outputs the phase difference between the frequency of this signal and the reference frequency from the EO0 and EO1 pins.

The following two types of input pins and five frequency division modes are used.

(1) Direct division (MF) mode

The VCOL pin is used.

The VCOH pin is set in the status specified by bit 3 (VCOHDMD) of the PLL mode select register (PLLMD).

(2) Pulse swallow (HF) mode

The VCOL pin is used.

The VCOH pin is set in the status specified by bit 3 (VCOHDMD) of PLLMD.

(3) Pulse swallow (VHF) mode

The VCOH pin is used.

The VCOL pin is set in the status specified by bit 2 (VCOLDMD) of PLLMD.

(4) VCOL and VCOH pin disable

The VCOL and VCOH pins are set in the status specified by bits 2 (VCOLDMD) and 3 (VCOHDMD) of PLLMD.

At this time, the phase comparator, reference frequency generator, and charge pump operate.

(5) PLL disable

The PLL disabled status is set by the PLL reference mode register (PLLRF).

The VCOH and VCOL pins are set in the status specified by bits 2 (VCOLDMD) and 3 (VCOHDMD) of PLLMD.

The EO0 and EO1 pins go into a high-impedance state.

At this time, all the internal PLL operations are stopped.

These division modes are selected by using the PLL mode select register (PLLMD).

The division value (N value) is set to the programmable divider by using the PLL data register. Frequency division in each of the above modes is carried out according to the value (N value) set to the programmable divider.

Table 19-1 shows the division modes, input pins used (VCOL pin or VCOH pin), and the value that can be set to the programmable divider.

Table 19-1. Division Mode, Input Pin, and Division Value

Division Mode	Pin Used	Value That Can Be Set
Direct division (MF)	VCOL	32 to $2^{12} - 1$
Pulse swallow (HF)	VCOL	1024 to $2^{17} - 1$
Pulse swallow (VHF)	VCOH	1024 to $2^{17} - 1$

Caution For the frequencies that can actually be input, and the input amplitude, refer to CHAPTER 25 ELECTRICAL SPECIFICATIONS.

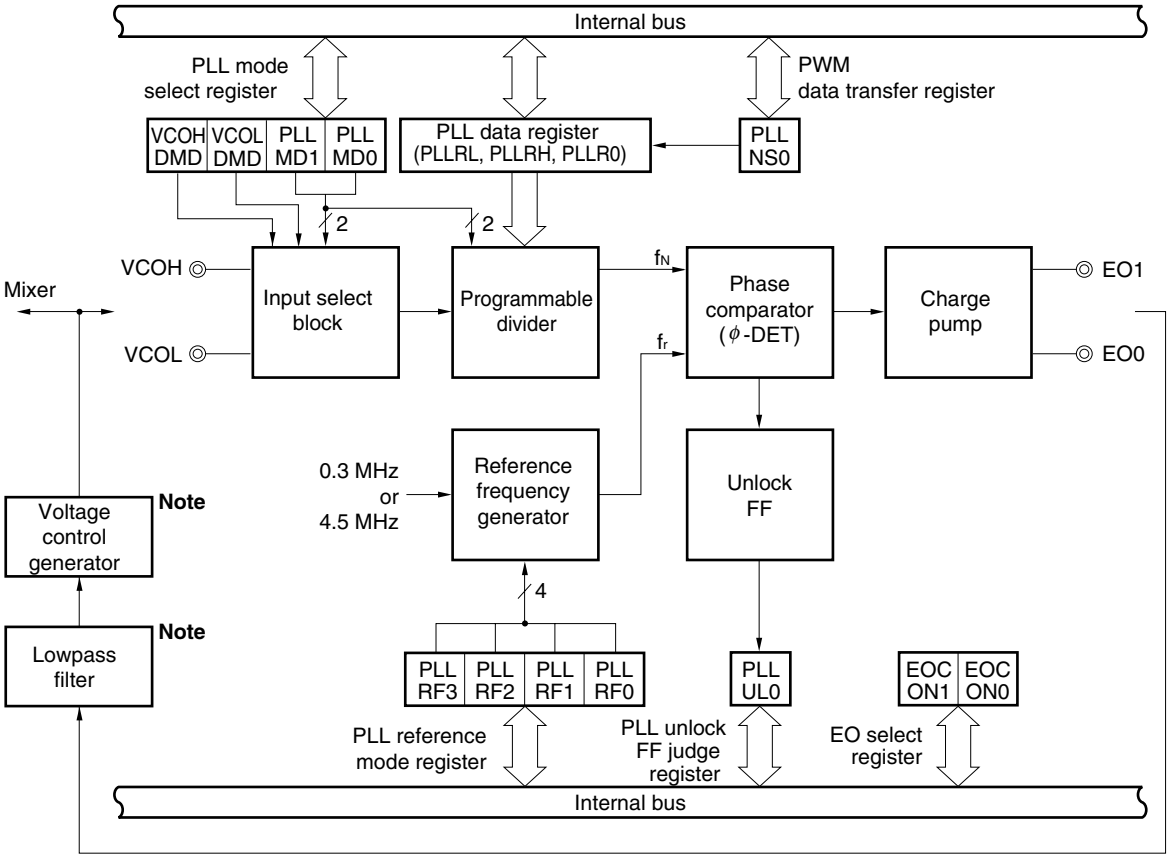
19.2 Configuration of PLL Frequency Synthesizer

The PLL frequency synthesizer consists of the following hardware.

Table 19-2. Configuration of PLL Frequency Synthesizer

Item	Configuration
Data registers	PLL data register L (PLLRL) PLL data register H (PLLRH) PLL data register 0 (PLLR0)
Control registers	PLL mode select register (PLLMD) PLL reference mode register (PLLRF) PLL unlock FF judge register (PLLUL) PLL data transfer register (PLLNS)

Figure 19-1. Block Diagram of PLL Frequency Synthesizer



Note External circuit

(1) PLL data register L (PLLRL), PLL data register H (PLLRH), and PLL data register 0 (PLLR0)

These registers set the division value of the PLL frequency synthesizer. The division value of the PLL frequency synthesizer is made up of 17 bits. The higher 16 bits of this value are set by PLL data register L (PLLRL) and PLL data register H (PLLRH). The higher 16 bits can also be set by the PLL data register (PLLR). The least significant bit is set by bit 7 (PLLSCN) of PLL data register 0 (PLLR0).

The contents of these registers are undefined after reset. These registers hold the current values in the STOP and HALT modes.

(2) Input select block

The input select block consists of the VCOL and VCOH pins, and input amplifiers of the respective pins.

(3) Programmable divider

The programmable divider consists of two modulus prescalers, a programmable counter (12 bits), a swallow counter (5 bits), and a division mode select switch.

(4) Reference frequency generator

The reference frequency generator consists of a divider that generates the reference frequency f_r of the PLL frequency synthesizer, and a multiplexer.

(5) Phase comparator

The phase comparator (ϕ-DET) compares the phase of the divided frequency output f_N of the programmable divider with that of the reference frequency output f_r of the reference frequency generator, and outputs an up request signal ($\overline{UP}$) and down request signal ($\overline{DW}$).

(6) Unlock FF

The unlock FF detects the unlock status of the PLL frequency synthesizer from the up request signal ($\overline{UP}$) and down request signal ($\overline{DW}$) of the phase comparator (ϕ-DET).

(7) Charge pump

The charge pump outputs the result of the output of the phase comparator from the error out pins (EO0 and EO1 pins).

19.3 Registers Controlling PLL Frequency Synthesizer

The PLL frequency synthesizer is controlled by the following four registers.

- PLL mode select register (PLLMD)
- PLL reference mode register (PLLRF)
- PLL unlock FF judge register (PLLUL)
- PLL data transfer register (PLLNS)

(1) PLL mode select register (PLLMD)

This register selects the input pin and division mode of the PLL frequency synthesizer.

PLLMD is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is set to 00H after reset.

In the STOP mode, only bits 3 and 2 (VCOHDMD and VCOLDMD) retain the previous value. Bits 1 and 0 (PLLMD1 and PLLMD0) are reset to 0.

In the HALT mode, this register holds the value immediately before the HALT mode was set.

Figure 19-2. Format of PLL Mode Select Register (PLLMD)

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	After reset	R/W
PLLMD	0	0	0	0	VCOHDMD	VCOLDMD	PLLMD1	PLLMD0	FFA0H	00H	R/W

VCOH DMD	Selection of disable status of VCOH pin
0	Connected to pull-down resistor.
1	High-impedance status

VCOL DMD	Selection of disable status of VCOL pin
0	Connected to pull-down resistor.
1	High-impedance status

PLLMD1	PLLMD0	Selection of division mode of PLL frequency synthesizer and VCO input pin
0	0	VCOL and VCOH pins disabled[Note]
0	1	Direct division (VCOL pin and MF mode)
1	0	Pulse swallow (VCOH pin and VHF mode)
1	1	Pulse swallow (VCOL pin and HF mode)

Note This does not mean that the PLL is disabled. The VCOH and VCOL pins become the status specified by bit 3 (VCOHDMD) and bit 2 (VCOLDMD). The EO0 and EO1 pins go low.

Remark Bits 4 to 7 are fixed to 0 by hardware.

(2) PLL reference mode register (PLLRF)

This register selects the reference frequency f_r of the PLL frequency synthesizer and sets the disabled status of the PLL frequency synthesizer.

PLLRF is set by 1-bit or 8-bit memory manipulation instruction.

The value of this register is set to 0FH after reset and in the STOP mode.

In the HALT mode, this register holds the value immediately before the HALT mode was set.

Figure 19-3. Format of PLL Reference Mode Register (PLLRF)

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	After reset	R/W
PLLRF	0	0	0	0	PLLRF3	PLLRF2	PLLRF1	PLLRF0	FFA1H	0FH	R/W

PLLRF3	PLLRF2	PLLRF1	PLLRF0	Settings of reference frequency f_r of PLL frequency synthesizer
0	0	0	0	50 kHz
0	0	0	1	25 kHz
0	0	1	0	12.5 kHz
0	0	1	1	9 kHz
0	1	0	0	1 kHz
0	1	0	1	3 kHz
0	1	1	0	10 kHz
0	1	1	1	Setting prohibited
1	×	×	×	PLL disable**Note**

Note When PLL disable is selected, the status of the VCOL, VCOH, EO0, and EO1 pins are as follows.

VCOH, VCOL pins: Status specified by bit 3 (VCOHDMD) and bit 2 (VCOLDMD) of the PLL mode select register (PLLMD).

EO0, EO1 pins: High-impedance state

Remark Bits 4 to 7 are fixed to 0 by hardware.

×: Don't care

(3) PLL unlock FF judge register (PLLUL)

This register detects whether the PLL frequency synthesizer is in the unlock status.

Because this register is an R&RESET register, it is reset to 0 after it has been read.

The value of this register is set to 0×H[Note 1] after reset.

In the STOP and HALT modes, this register holds the value immediately before the STOP or HALT mode was set.

Figure 19-4. Format of PLL Unlock FF Judge Register (PLLUL)

Symbol	7	6	5	4	3	2	1	<0>	Address	After reset	R/W
PLLUL	0	0	0	0	0	0	0	PLLUL0	FFA2H	0×H[Note 1]	R[Note 2]

PLLUL0	Detection of status of unlock FF
0	Unlock FF = 0: PLL lock status
1	Unlock FF = 1: PLL unlock status

Notes 1. The value of bit 0 (PLLUL0) after reset differs depending on the type of reset that has been executed (refer to the table below).

2. Bit 0 (PLLUL0) is R&Reset.

		7	6	5	4	3	2	1	0
After reset	Power-on clear	0	0	0	0	0	0	0	Undefined
	Watchdog timer	↓	↓	↓	↓	↓	↓	↓	Held
	$\overline{\text{RESET}}$ input	↓	↓	↓	↓	↓	↓	↓	Held
STOP mode		↓	↓	↓	↓	↓	↓	↓	Held
HALT mode		↓	↓	↓	↓	↓	↓	↓	Held

Remark Bits 1 to 7 are fixed to 0 by hardware.

(4) PLL data transfer register (PLLNS)

This register transfers the values of the PLL data registers (PLLRL, PLLRH, and PLLR0) to the programmable counter and swallow counter.

The value of this register is 00H after reset and in the STOP mode.

In the HALT mode, this register holds the previous value immediately before the HALT mode was set.

Figure 19-5. Format of PLL Data Transfer Register (PLLNS)

Symbol	7	6	5	4	3	2	1	<0>	Address	After reset	R/W
PLLNS	0	0	0	0	0	0	0	PLLNS0	FFA3H	00H	W

PLLNS0	Transfer of value of PLL data register to programmable counter and swallow counter
0	Not transferred
1	Transferred

Remark Bits 1 to 7 are fixed to 0 by hardware.

19.4 Operation of PLL Frequency Synthesizer

19.4.1 Operation of each block of PLL frequency synthesizer

(1) Operation of input select block and programmable divider

The input select block and programmable divider select the input pin and division mode of the PLL frequency synthesizer and divide the frequency in the selected division mode, according to the setting of the PLL mode select register (PLLMD).

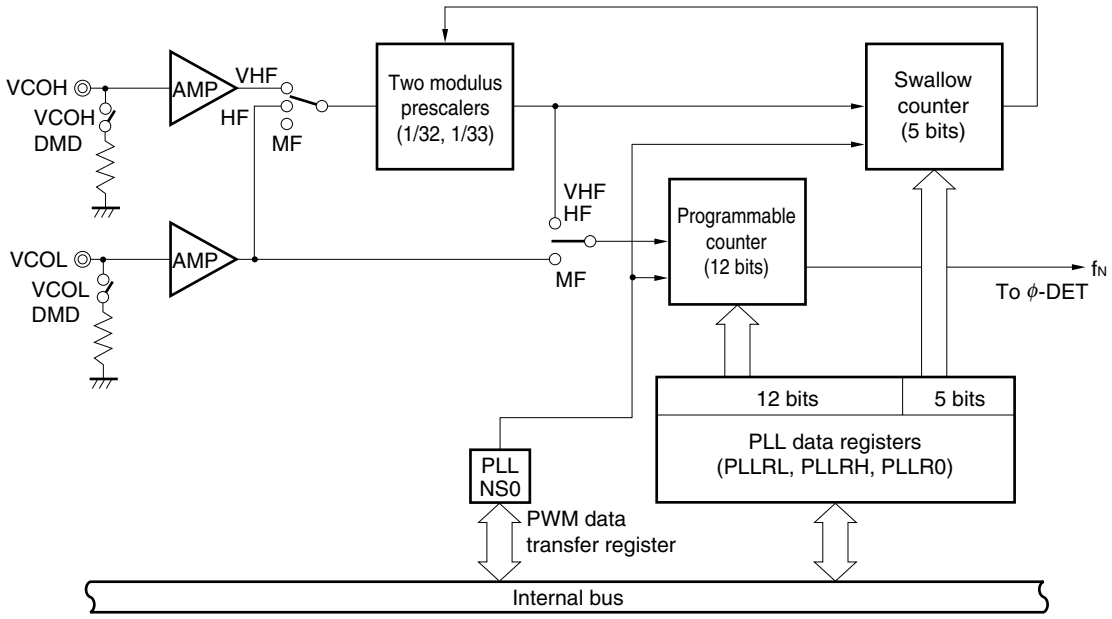
The programmable counter (12 bits) and pulse swallow counter (5 bits) are binary counters.

The division value (N value) is set to the programmable counter and swallow counter by the PLL data registers (PLLRL, PLLRH, and PLLR0).

When the N value has been transferred to the programmable counter and swallow counter, frequency division is performed in the selected division mode according to the status of bit 0 (PLLNS0) of the PLL data transfer register.

Figure 19-6 shows the configuration of the input select block and programmable divider.

Figure 19-6. Configuration of Input Select Block and Programmable Divider



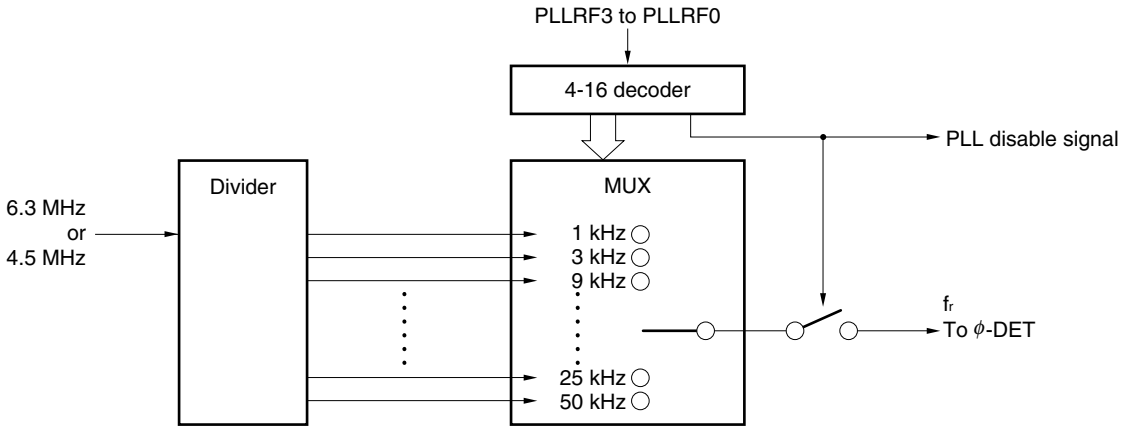
(2) Operation of reference frequency generator

The reference frequency generator divides the 6.3 MHz or 4.5 MHz output of the crystal resonator and generates seven types of reference frequency f_r for the PLL frequency synthesizer.

Reference frequency f_r is selected by the PLL reference mode register (PLLRF).

Figure 19-7 shows the configuration of the reference frequency generator.

Figure 19-7. Reference Frequency Generator Configuration



(3) Operation of phase comparator (ϕ-DET)

Figure 19-8 shows the configuration of the phase comparator (ϕ-DET), charge pump, and unlock FF.

The phase comparator (ϕ-DET) compares the phase of the divided frequency f_N of the programmable divider with that of the reference frequency f_r of the reference frequency generator, and outputs an up request signal, $\overline{UP}$, or a down request signal, $\overline{DW}$.

If the divided frequency f_N is lower than the reference frequency f_r, the up request signal is output. If f_N is higher than f_r, the down request signal is output.

Figure 19-9 shows the relationships between the reference frequency f_r, divided frequency f_N, up request signal $\overline{UP}$, and down request signal $\overline{DW}$.

When the PLL is disabled, neither the up nor the down request signal is output.

The up and down request signals are input to the charge pump and unlock FF.

Figure 19-8. Phase Comparator, Charge Pump, and Unlock FF Configuration

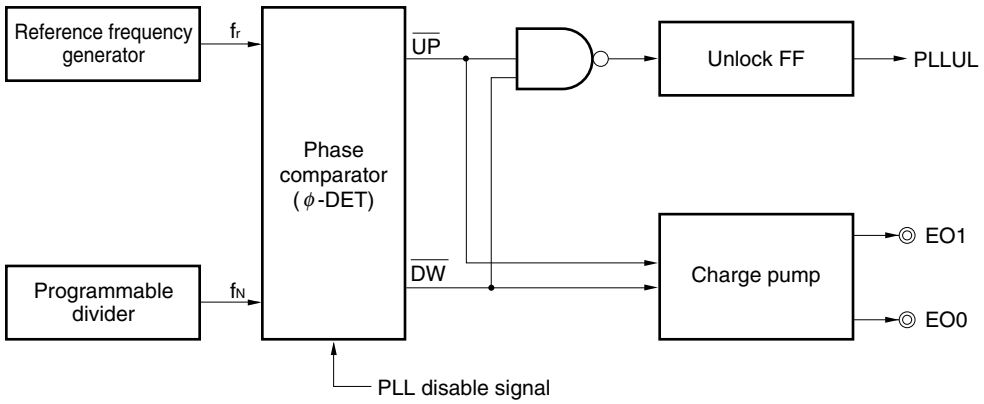
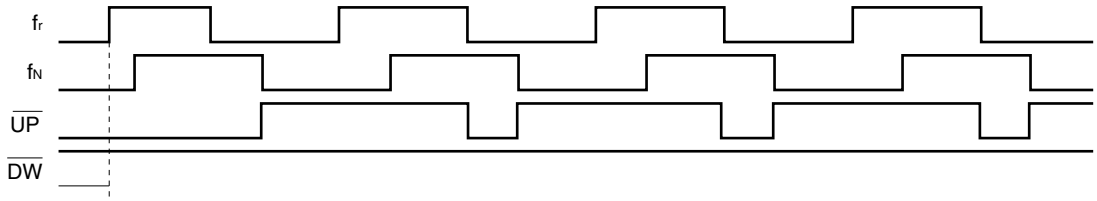
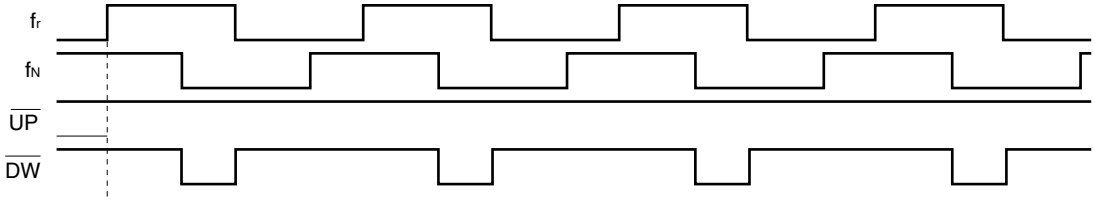


Figure 19-9. Relationship Between f_r, f_N, $\overline{UP}$, and $\overline{DW}$

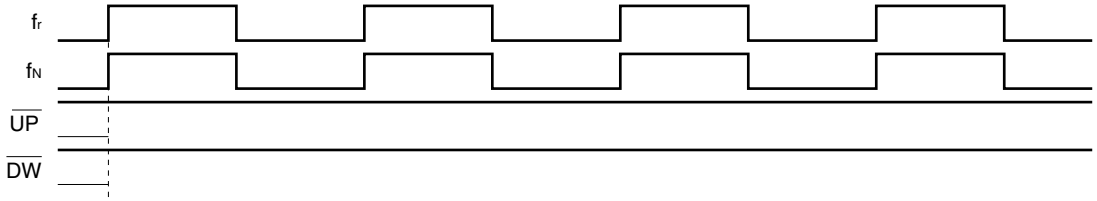
(a) If f_N advances f_r in phase



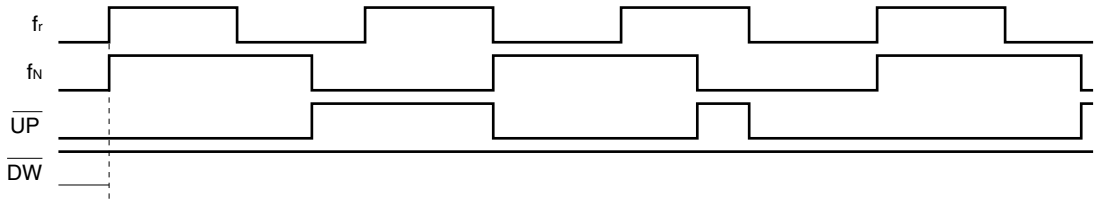
(b) If f_N advances f_r in phase



(c) If f_N and f_r are in phase



(d) If f_N is lower than f_r



(4) Operation of charge pump

The charge pump outputs the result of the up request ($\overline{UP}$) or down request ($\overline{DW}$) signal from the phase comparator (ϕ-DET) from the error out pins (EO0 and EO1 pins). Table 19-3 shows the output signals.

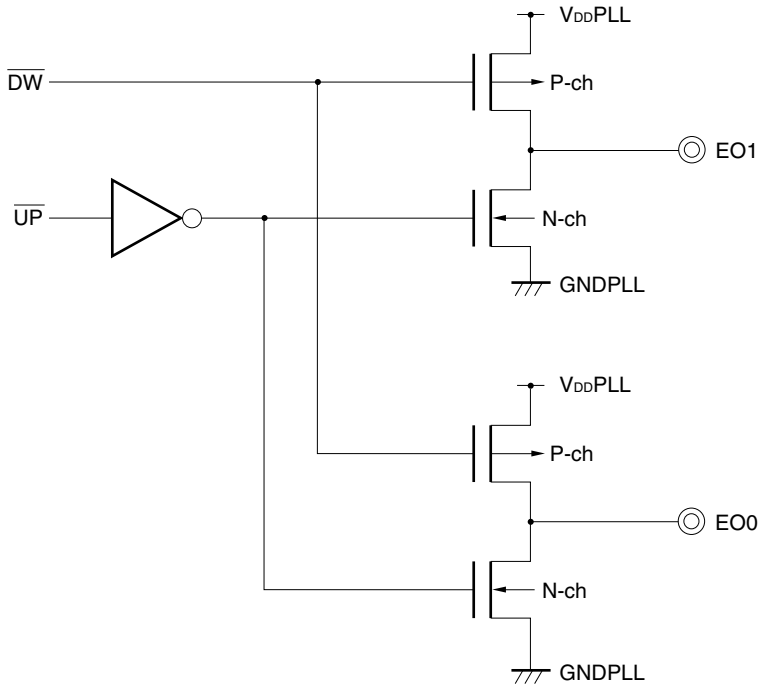
★ The EO0 and EO1 pins are voltage-driven type pins.

Figure 19-10 shows the configuration of the error out pins.

Table 19-3. Error Out Output Signal

Relationship Between Divided Frequency f_N and Reference Frequency f_r	Error Out Output Signal
When $f_r > f_N$	Low level
When $f_r < f_N$	High level
When $f_r = f_N$	Floating (high impedance)

Figure 19-10. Error Out Pin Configuration



(5) Operation of unlock FF

The unlock FF detects the unlock status of the PLL frequency synthesizer.

It detects the unlock status of the PLL frequency synthesizer from the up request signal $\overline{UP}$ and down request signal $\overline{DW}$ of the phase comparator (ϕ-DET).

Because either of the up request or down request signal outputs a low level in the unlock status, the unlock status can be detected by using this low-level signal.

The status of the unlock FF is detected by bit 0 (PLLUL0) of the PLL unlock FF judge register (PLLUL).

The unlock FF is set at the cycle of the reference frequency f_r selected at that time.

The PLL unlock FF judge register is reset when its contents have been read.

To read the PLL unlock FF judge register, therefore, it must be read at a cycle longer than the cycle ($1/f_r$) of the reference frequency.

19.4.2 Operation to set N value of PLL frequency synthesizer

The division value (N value) is set to the programmable counter (12 bits) and swallow counter (5 bits) by the PLL data registers (PLLRL, PLLRH, and PLLR0).

When the N value has been transferred to the programmable counter and swallow counter by bit 0 (PLLNS0) of the PLL data transfer register (PLLNS), frequency division is carried out in the selected division mode.

Examples of setting the N value in the respective division modes (MF, HF, and VHF) are shown below.

(1) Direct division mode (MF)

(a) Calculating division value N (value set to PLL data register)

$$N = \frac{f_{VCOL}}{f_r}$$

where, f_{VCOL}: Input frequency of V_{COL} pin

f_r: Reference frequency

(b) Example of setting PLL data register

An example of setting the PLL data register to receive broadcast stations in the following MW band is shown below.

Receive frequency: 1422 kHz (MW band)

Reference frequency: 9 kHz

Intermediate frequency: 450 kHz

Division value N is calculated as follows:

$$N = \frac{f_{VCOL}}{f_r} = \frac{1422 + 450}{9} = 208 \text{ (decimal)}$$

$$= 0D0H \text{ (hexadecimal)}$$

Data is set to the PLL data registers (PLLR and PLLR0) as follows.

PLLR																PLLR0							
PLLRH								PLLRL								←PLLSCN							
b7	b6	b5	b4	b3	b2	b1	b0	b7	b6	b5	b4	b3	b2	b1	b0	b7	b6	b5	b4	b3	b2	b1	b0
b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0							
Programmable counter value												Don' t care				Fixed to 0							
0	0	0	0	1	1	0	1	0	0	0	0												
0				D				0															

After setting the above PLL data registers (PLLR and PLLR0), data must be transferred to the programmable counter by setting bit 0 (PLLNS0) of the PLL data transfer register (PLLNS).

(2) Pulse swallow mode (HF)

(a) Calculating division value N (value set to PLL data register)

$$N = \frac{f_{VCOL}}{f_r}$$

where, f_{VCOL}: Input frequency of V_{COL} pin

f_r: Reference frequency

(b) Example of setting PLL data register

An example of setting the PLL data register to receive broadcast stations in the following SW band is shown below.

Receive frequency: 25.50 MHz (SW band)

Reference frequency: 10 kHz

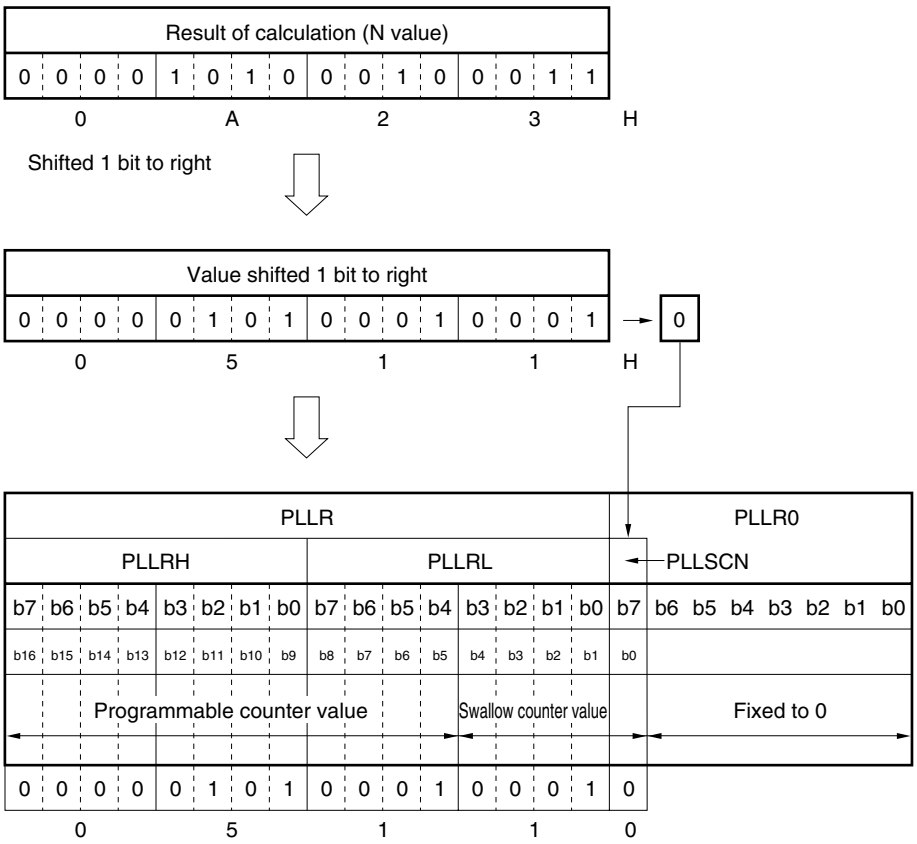
Intermediate frequency: 450 kHz

Division value N is calculated as follows:

$$N = \frac{f_{VCOL}}{f_r} = \frac{25500 + 450}{10} = 2595 \text{ (decimal)}$$

$$= 0A23H \text{ (hexadecimal)}$$

Because the least significant bit of division value N must be set to bit 7 (PLLSCN) of PLL data register 0 (PLLR0), data must be set by shifting the result of the above calculation 1 bit to the right.
Data is set to the PLL data registers (PLLR and PLLR0) as follows.



After setting the above PLL data registers (PLLR and PLLR0), data must be transferred to the programmable counter and swallow counter by setting bit 0 (PLLNS0) of the PLL data transfer register (PLLNS).
In this example, a value of half the N value is set to the higher 16 bits of the PLL data register (PLLR) by shifting the N value resulting from calculation 1 bit to the right.
If the N value is calculated as follows with the least significant bit of the N value in PLLSCN fixed to 0, the result of the calculation (N_{PLLR}) can be set to the PLL data register (PLLR) as is.
If the calculation result is set in this way, however, the input frequency (f_{VCOL}) is $2 \times f_r$ (reference frequency) of the set value N_{PLLR}.

$$N_{PLLR} = \frac{f_{VCOL}}{2f_r}$$

(3) Pulse swallow mode (VHF)

(a) Calculating division value N (value set to PLL data register)

$$N = \frac{f_{VCOH}}{f_r}$$

where, f_{VCOH}: Input frequency of VCOH pin

f_r: Reference frequency

(b) Example of setting PLL data register

An example of setting the PLL data register to receive broadcast stations in the following FM band is shown below.

Receive frequency: 100.0 MHz (FM band)

Reference frequency: 50 kHz

Intermediate frequency: +10.7 MHz

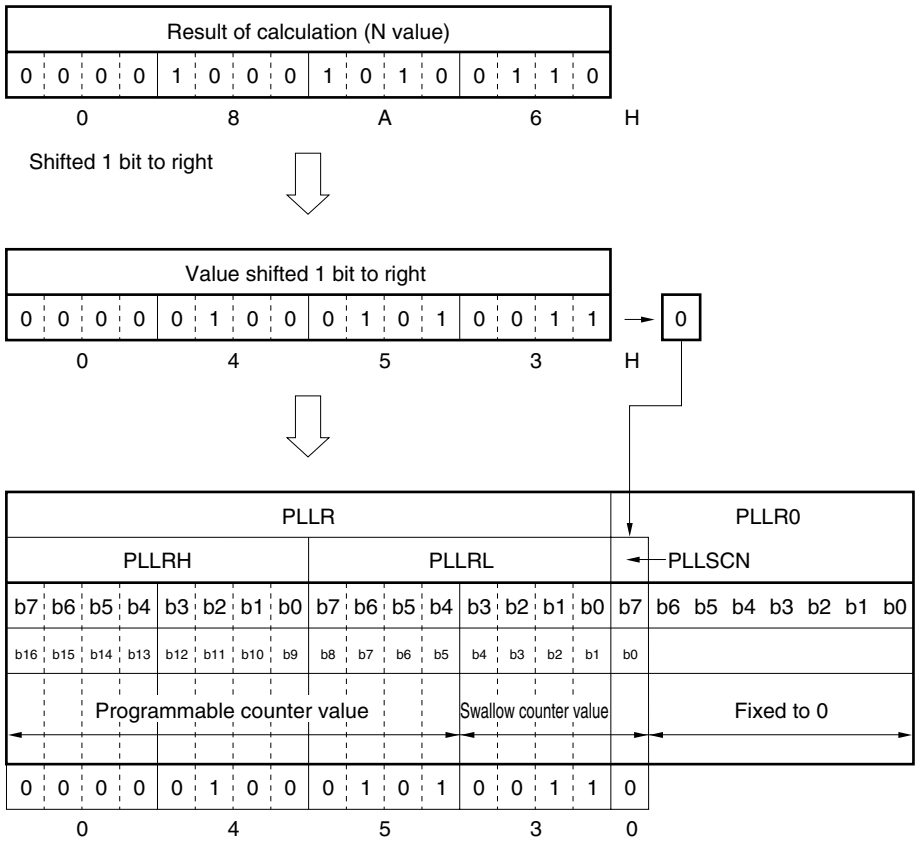
Division value N is calculated as follows.

$$N = \frac{f_{VCOH}}{f_r} = \frac{100.0 + 10.7}{0.05} = 2214 \text{ (decimal)}$$

$$= 08A6H \text{ (hexadecimal)}$$

Because the least significant bit of division value N must be set to PLL data register 0 (PLLR0), data must be set by shifting the value calculated by the above expression 1 bit to the right.

Data is set to the PLL data registers (PLLR and PLLR0) as follows.



After setting the above PLL data registers (PLLR and PLLR0), data must be transferred to the programmable counter and swallow counter by setting bit 0 (PLLNS0) of the PLL data transfer register (PLLNS).

In this example, a value of half the N value is set to the higher 16 bits of the PLL data register (PLLR) by shifting the N value resulting from calculation 1 bit to the right.

If the N value is calculated as follows with the least significant bit of the N value in PLLSCN fixed to 0, the result of the calculation (N_{PLLR}) can be set to the PLL data register (PLLR) as is.

If the calculation result is set in this way, however, the input frequency (f_{VCOH}) is $2 \times f_r$ (reference frequency) of the set value N_{PLLR}.

$$N_{PLLR} = \frac{f_{VCOH}}{2f_r}$$

19.5 PLL Disable Status

The PLL frequency synthesizer can be stopped (PLL disable status) by performing any of the following settings while the PLL frequency synthesizer is operating.

- Setting the value of bit 3 (PLLRF3) of the PLL reference mode register (PLLRF) to 1 to set the PLL disable status
- Setting STOP mode using the STOP instruction
- Setting the reset using the reset function

The following table shows the operation of each block and the status of each register in the PLL disable status.

Table 19-4. Operation of Each Block and Register Status in PLL Disable Status

Block/Register	Status in PLL Disable Status
VCOL and VCOH pins	Status set in bit 3 (VCOHDMD) and bit 2 (VCOLDMD) of PLLMD
Programmable divider	Division stops
Reference frequency generator	Output stops
Phase comparator	Output stops
EO0 and EO1 pin	High impedance
PLL mode select register	Retains value on execution of write instruction
PLL data register	
PLL unlock FF judge register	

19.6 Notes on PLL Frequency Synthesizer

- **Notes on using PLL frequency synthesizer**
  Because the input pins (VCOL and VCOH pins) of the PLL frequency synthesizer are provided with an AC amplifier, cut the DC component of the input signal by connecting a capacitor to the input pins in series.
  The potential of the selected input pin is intermediate (about $1/2V_{DD}$). An unselected input pin becomes the status set in bit 3 (VCOHDMD) and bit 2 (VCOLDMD) of the PLL mode select register (PLLMD).
  For the frequencies that can actually be input and the input amplitude, refer to **CHAPTER 25 ELECTRICAL SPECIFICATIONS**.

CHAPTER 20 FREQUENCY COUNTER

20.1 Function of Frequency Counter

The frequency counter counts the intermediate frequency (IF) of a tuner.

It counts the intermediate frequency input to the FMIFC or AMIFC pin for a specific time (1 ms, 4 ms, 8 ms, or open) with a 16-bit counter. The count value of the frequency counter is stored in the IF counter register.

For the range of the frequency that can be input to the FMIFC and AMIFC pins, refer to **CHAPTER 25 ELECTRICAL SPECIFICATIONS**.

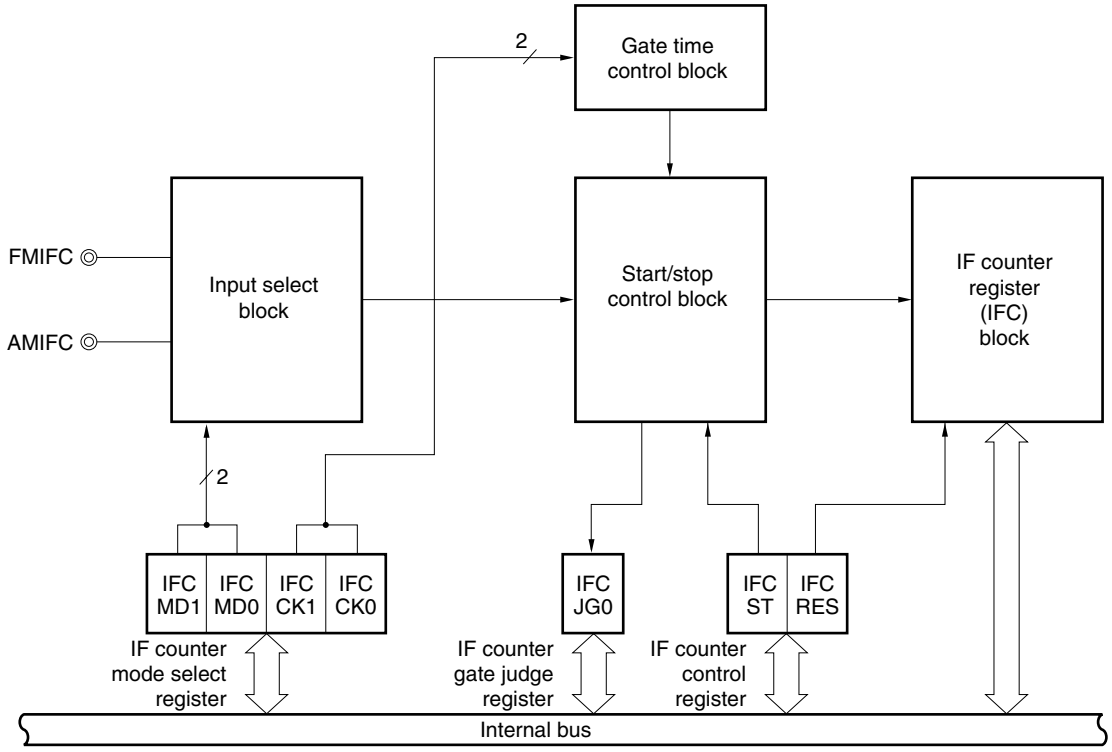
20.2 Configuration of Frequency Counter

The frequency counter consists of the following hardware.

Table 20-1. Configuration of Frequency Counter

Item	Configuration
Counter register	IF counter register (IFC)
Control registers	IF counter mode select register (IFCMD) IF counter control register (IFCR) IF counter gate judge register (IFCJG)

Figure 20-1. Frequency Counter Block Diagram



(1) IF counter input select block

The IF counter input select block selects the pin to be used from the FMIFC and AMIFC pins, and the count mode.

(2) Gate time control block

The gate time control block sets the gate time (count time).

(3) Start/stop control block

The start/stop control block sets the count start of the IF counter register and detects the end of counting.

(4) IF counter register block

The IF counter register block is a 16-bit register that counts up the frequency input in the set gate time. The count value is stored in the IF counter register (IFCR). When the count value reaches FFFFH, the IF counter register holds FFFFH at the next input, and stops counting. The value of this register is reset to 0000H after reset or in the STOP mode. In the HALT mode, it holds the current count value.

20.3 Registers Controlling Frequency Counter

The frequency counter is controlled by the following three registers.

- IF counter mode select register (IFCMD)
- IF counter control register (IFCCR)
- IF counter gate judge register (IFCJG)

(1) IF counter mode select register (IFCMD)

This register selects the input pin of the frequency counter, and selects the mode and gate time (count time).

This register is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is reset to 00H after reset or in the STOP mode.

In the HALT mode, this register holds the value immediately before the HALT mode was set.

Figure 20-2. Format of IF Counter Mode Select Register (IFCMD)

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	After reset	R/W
IFCMD	0	0	0	0	IFCMD1	IFCMD0	IFCCK1	IFCCK0	FFA9H	00H	R/W

IFCMD1	IFCMD0	Selection of frequency counter pin and mode
0	0	FMIFC AMIFC pins disabled[Note 1]
0	1	AMIFC pin, AMIF count mode[Note 2]
1	0	FMIFC pin, FMIF count mode[Note 2]
1	1	FMIFC pin, AMIF count mode[Note 2]

IFCCK1	IFCCK0	Selection of gate time
0	0	1 ms
0	1	4 ms
1	0	8 ms
1	1	Open

Notes 1. The FMIFC and AMFIC pins are used as port pins.

2. When using the AMIFC/P101 and FMIFC/P102 pins to input signals to the frequency counter, set PM101 and PM102 to 1.

Caution Any pin not selected by IFCMD is automatically set in the port mode.

Remark Bits 4 to 7 are fixed to 0 by hardware.

(2) IF counter control register (IFCCR)

This register sets the count start of the IF counter register and clears the IF counter register.

IFCCR is set by a 1-bit or 8-bit memory manipulation instruction.

The value of this register is reset to 00H after reset and in the STOP mode.

In the HALT mode, this register holds the value immediately before the HALT mode was set.

Figure 20-3. Format of IF Counter Control Register (IFCCR)

Symbol	7	6	5	4	3	2	<1>	<0>	Address	After reset	R/W
IFCCR	0	0	0	0	0	0	IFCST	IFCRES	FFACH	00H	W

IFCST	Count start of IF counter register
0	Nothing is affected
1	Counting starts

IFCRES	Clearance of data of IF counter register
0	Nothing is affected
1	Data of IF counter register cleared

Remark Bits 2 to 7 are fixed to 0 by hardware.

(3) IF counter gate judge register (IFCJG)

This register detects opening/closing of the gate of the frequency counter.

The value of this register is reset to 00H after reset and in the STOP mode.

In the HALT mode, this register holds the value immediately before the HALT mode was set.

Figure 20-4. Format of IF Counter Gate Judge Register (IFCJG)

Symbol	7	6	5	4	3	2	1	<0>	Address	After reset	R/W
IFCJG	0	0	0	0	0	0	0	IFCJG0	FFABH	00H	R

IFCJG0	Detection of opening/closing of gate of frequency counter
0	Gate is closed
1	• If gate time is set to other than open Status until gate is closed after IFCST has been set to 1 • If gate time is set to open Status where gate is open as soon as it has been set to be opened

Remark Bits 1 to 7 are fixed to 0 by hardware.

Caution IFCJG0 remains set even if the IF counter register overflows and stops counting, until the set gate time expires.

20.4 Operation of Frequency Counter

(1) Select the input pin, mode, and gate time by using the IF counter mode select register (IFCMD). Figure 20-5 shows a block diagram of input pin and mode selection.

(2) Set bit 0 (IFCRES) of the IF counter control register (IFCCR) to 1, and clear the data of the IF counter register.

(3) Set bit 1 (IFCST) of the IF counter control register (IFCCR) to 1.

(4) The gate is opened only for the set gate time from when the 1 kHz internal signal rises after IFCST is set. If the gate time is set to open, the gate is opened as soon as it has been specified to be opened.
Bit 0 (IFCJG0) of the IF counter gate judge register (IFCJG) is automatically set to 1 as soon as IFCST has been set to 1.
When the gate time has expired, bit 0 (IFCJG0) of the IF counter gate judge register (IFCJG) is automatically cleared to 0. If it is specified that the gate be open, however, IFCJG0 is not automatically cleared. In this case, set a gate time. Figure 20-6 shows the gate timing of the frequency counter.

(5) The IF counter register counts the frequency input to the selected FMIFC or FMIFC pin while the gate is open. If the FMIFC pin is used in the FMIF count mode, however, the input frequency is divided by half before it is counted input to the selected FMIFC or AMIFC pin while the gate is open.

The relationship between count value x (decimal), the input frequencies (f_{FMIFC} and f_{AMIFC}), and the gate time (T_{GATE}) is shown below.

- FMIF count mode (FMIFC pin)

$$f_{FMIFC} = \frac{x}{T_{GATE}} \times 2 \text{ (kHz)}$$

- AMIF count mode (FMIFC or AMIFC pin)

$$f_{AMIFC} = \frac{x}{T_{GATE}} \text{ (kHz)}$$

Figure 20-5. Block Diagram of Input Pin and Mode Selection

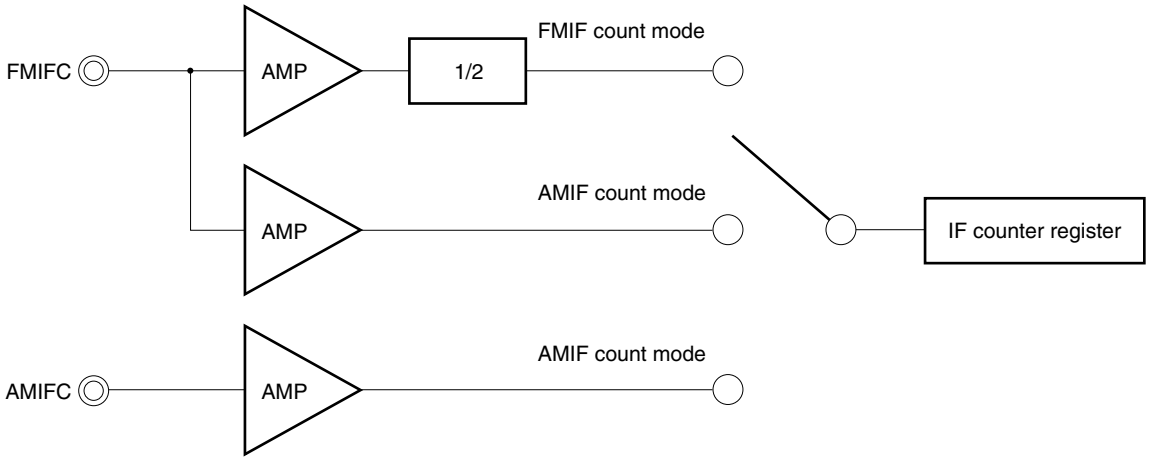
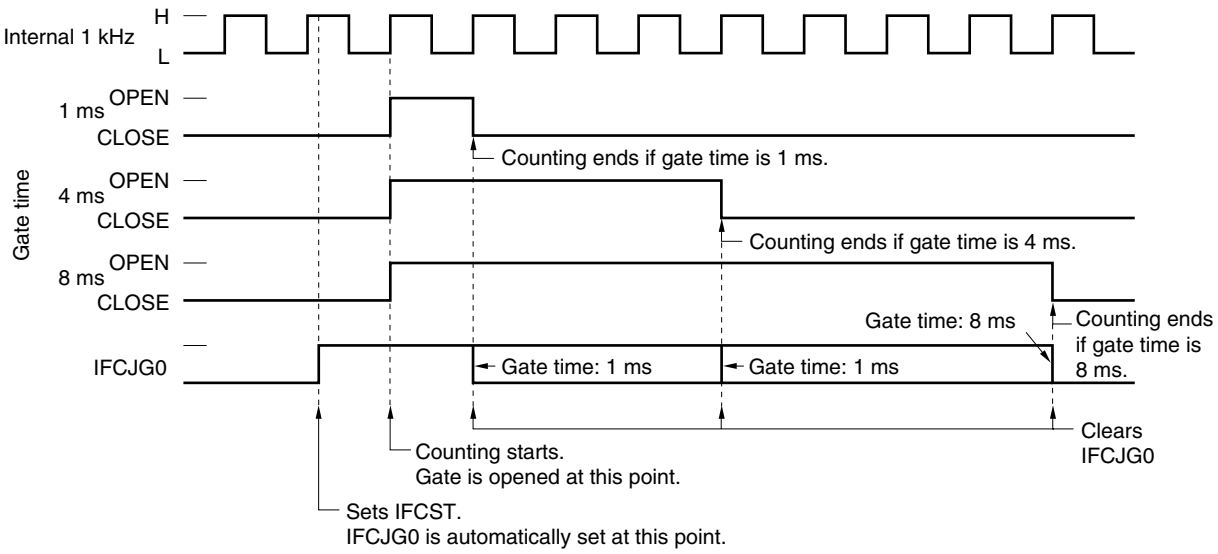
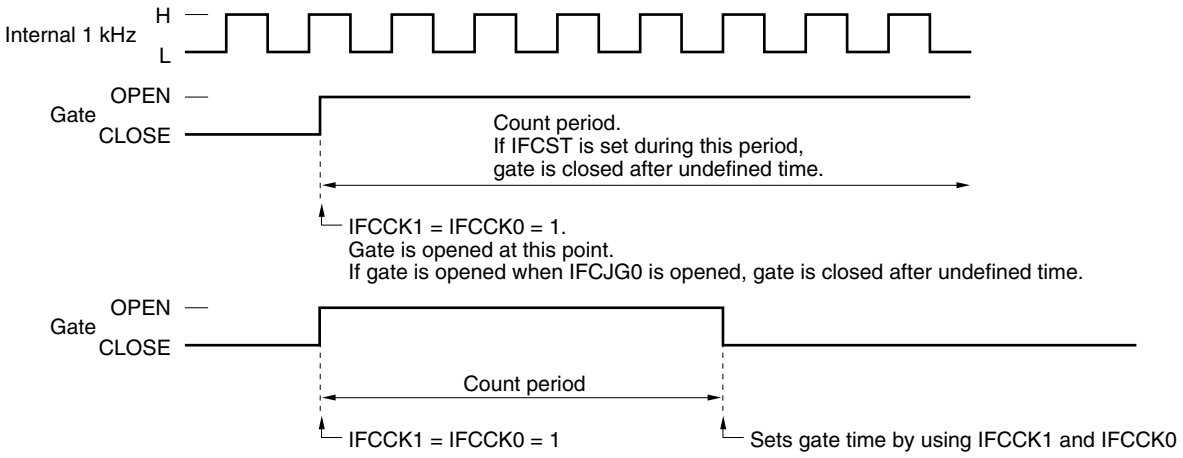


Figure 20-6. Gate Timing of Frequency Counter

(a) If gate time is set to 1, 4, or 8 ms



(b) If gate is set to be open



Caution If counting is started by using IFCST while this gate is open, the gate is closed after an undefined time. To open the gate, therefore, do not set IFCST to 1.

Remark IFCST: Bit 1 of the IF counter control register (IFCCR)
IFCJG0: Bit 0 of the IF counter gate judge register (IFJG)
IFCCK1, IFCCK0: Bits 1 and 0 of the IF counter mode select register (IFCMD)

20.5 Notes on Frequency Counter

(1) Notes on using frequency counter

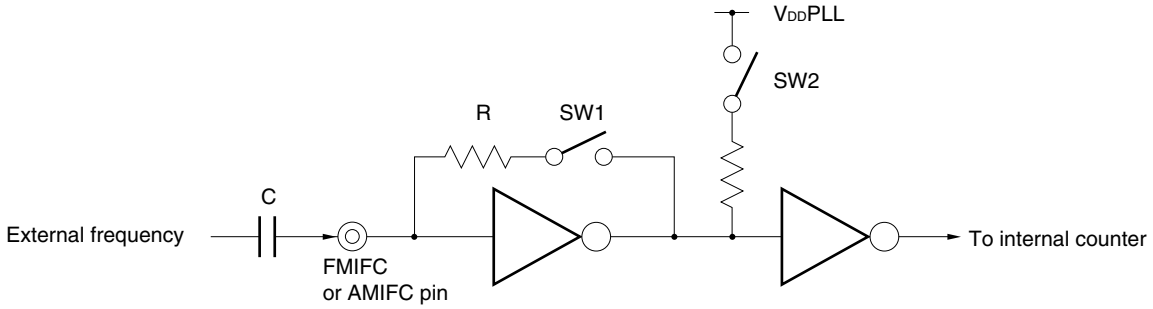
Because signals are input to the frequency counter from an input pin (FMIFC or AMIFC pin) with an AC amplifier as shown in Figure 20-7, cut the DC component of the input signals by using capacitor C.

If the FMIFC or AMIFC pin is selected by the IF counter mode select register, switch SW1 turns ON, and switch SW2 turns OFF. As a result, the voltage on the pin is about 1/2V_{DD}.

Unless the voltage has risen to a sufficient intermediate level at this time, counting may not be performed normally because the AC amplifier is not in the normal operating range.

Therefore, make sure that sufficient wait time elapses after a pin has been selected and before counting is started (IFCST = 1).

Figure 20-7. Frequency Counter Input Pin Circuit



(2) Notes in HALT mode

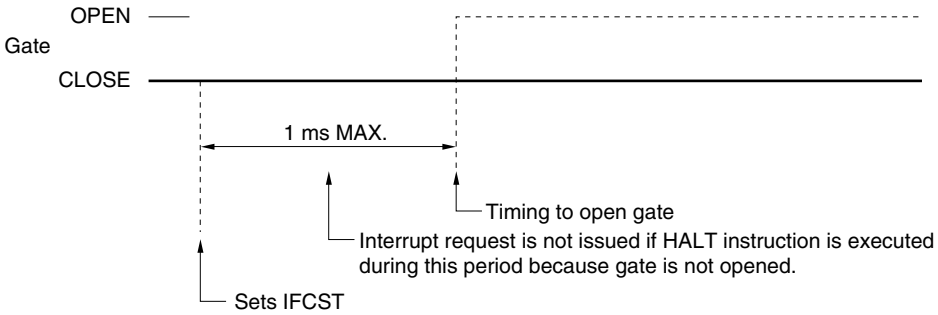
The FMIFC and AMIFC pins hold the status immediately before the HALT mode was set.

To release the HALT mode by using the interrupt of the frequency counter at this time, the following point must be noted.

The gate will not be opened if the HALT instruction is executed after counting has been started by IFCST before the gate is actually opened.

Therefore, wait for at least 1 ms before executing the HALT instruction.

Figure 20-8. Gate Status When HALT Instruction Is Executed



(3) Error of frequency counter

The error of the frequency counter includes the error of the gate time and the count error.

(1) Error of gate time

The gate time of the frequency counter is created by dividing 6.3 MHz.

Therefore, if 6.3 MHz is shifted “+x” ppm, the gate time is also shifted “–x” ppm.

(2) Count error

The frequency counter counts the frequency at the rising edge of the input signal.

If a high level is input to the pin when the gate is opened, therefore, one excess pulse is counted. When the gate is closed, however, counting is not affected by the status of the pin.

Therefore, the count error is “maximum + 1”.

CHAPTER 21 STANDBY FUNCTION

21.1 Standby Function and Configuration

21.1.1 Standby function

The standby function is designed to decrease the power consumption of the system. The following two modes are available.

(1) HALT mode

HALT instruction execution sets the HALT mode. The HALT mode stops the CPU operation clock, but the system clock oscillator continues oscillating. In this mode, the current consumption cannot be decreased as much as in the STOP mode. The HALT mode is effective for restarting immediately upon interrupt request generation and to carry out intermittent operations such as in watch applications.

Although the CPU stops operating, the peripheral functions can operate. To lower the current consumption, therefore, stop all unnecessary circuits before executing the HALT instruction.

(2) STOP mode

STOP instruction execution sets the STOP mode. In the STOP mode, the system clock oscillator stops and the whole system stops. The CPU current consumption can be considerably decreased in this mode.

Data memory low-voltage hold (down to V_{DD} = 2.3 V) is possible. Thus, the STOP mode is effective for holding data memory contents with ultra-low current consumption.

If the supply voltage drops below 2.3 V, the system is reset by means of power-on clear reset. For reset, refer to **CHAPTER 22 RESET FUNCTION**.

Because this mode can be released upon interrupt request generation, it enables intermittent operations to be carried out.

However, because a wait time is necessary to secure the oscillation stabilization time after the STOP mode is released, select the HALT mode if it is necessary to start processing immediately upon interrupt request generation.

All the functions stop operating in this mode.

Some registers of the PLL frequency synthesizer and frequency counter are reset, but the other functions are stopped with their current status held.

Cautions 1. When proceeding to the STOP mode, be sure to stop the peripheral hardware operations before executing the STOP instruction.

2. The following sequence is recommended for power consumption reduction of the A/D converter: first clear bit 7 (ADCS3) of ADM3 to 0 to stop the A/D conversion operation, then execute the HALT or STOP instruction.

21.1.2 Standby function control register

A wait time from when the STOP mode is released upon interrupt request generation until the oscillation stabilizes is controlled by the oscillation stabilization time select register (OSTS).

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

Reset input sets OSTS to 04H.

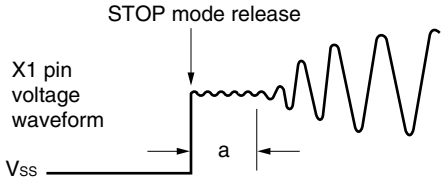
Figure 21-1. Format of Oscillation Stabilization Time Select Register (OSTS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

OSTS2	OSTS1	OSTS0	Selection of oscillation stabilization time when STOP mode is released
0	0	0	2^{12}/fx (650 μs)
0	0	1	2^{14}/fx (2.60 ms)
0	1	0	2^{15}/fx (5.20 ms)
0	1	1	2^{16}/fx (10.4 ms)
1	0	0	2^{17}/fx (20.8 ms)
Other than above			Setting prohibited

Remark fx: System clock oscillation frequency
( ): fx = 6.3 MHz

Caution The wait time when the STOP mode is released does not include the time required for the clock oscillation to start after the STOP mode has been released (see "a" in the figure below), regardless of whether the mode has been released by the $\overline{\text{RESET}}$ signal or an interrupt request.



21.2 Standby Function Operations

21.2.1 HALT mode

(1) HALT mode setting and operating status

The HALT mode is set by executing the HALT instruction.

The operating status in the HALT mode is described below.

Table 21-1. HALT Mode Operating Status

Item		Status
Clock generator		System clock oscillates. Clock supply to CPU stopped.
CPU		Stops operating.
Port		Hold status before HALT mode was set.
16-bit timer/event counter 0		Hold operation before HALT mode was set and can operate.
8-bit timer/event counters 50, 51		
Basic timer		
Watchdog timer		
Buzzer output controller		
A/D converter		Holds operation performed when HALT mode was set. However, comparison cannot be performed correctly in A/D conversion operation mode. In power-fail comparison mode, operation is as follows depending on setting of bit 5 (PFHRM3) of power-fail comparison mode register 3 (PFM3): { PFHRM3 = 0: Comparison cannot be performed normally. { PFHRM3 = 1: Power-fail comparison operation can be performed.
Serial interface	SIO0, SIO3, UART0[Note 1]	Hold operation performed when HALT mode was set and can operate.
	SIO1	Holds operation performed when HALT mode was set. However, transfer is continued with erroneous data if serial clock is supplied in automatic transfer mode.
IEBus controller[Note 2]		Hold operation before HALT mode was set and can operate.
External interrupt		
PLL frequency synthesizer		
Frequency counter		Holds operation performed before HALT mode was set. However, operation is not performed correctly even though it is continued.
Power-on clear circuit		Reset when voltage of less than 3.5 V is detected.

Notes 1. μPD178076, 178078, and 178F098 only.

2. μPD178096A, 178098A, and 178F098 only.

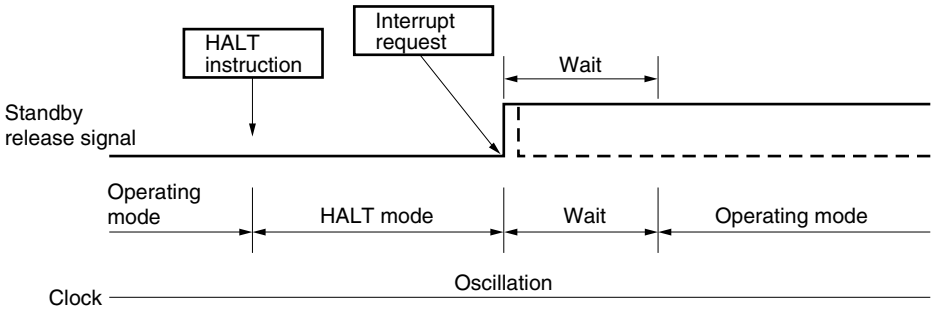
(2) HALT mode release

The HALT mode can be released by the following three sources.

(a) Release upon unmasked interrupt request generation

When an unmasked interrupt request is generated, the HALT mode is released. If interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If disabled, the next address instruction is executed.

Figure 21-2. HALT Mode Release upon Interrupt Generation



Remarks 1. The broken lines indicate the case when the interrupt request which has released the standby status is acknowledged.

2. The wait time is as follows:

- When vectored interrupt servicing is carried out: 8 to 9 clocks
- When vectored interrupt servicing is not carried out: 2 to 3 clocks

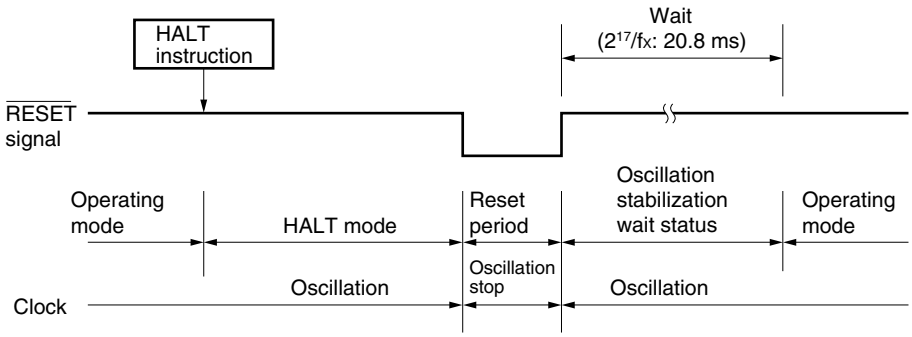
(b) Release upon non-maskable interrupt request generation

When a non-maskable interrupt is generated, the HALT mode is released and vectored interrupt servicing is carried out regardless of whether interrupt acknowledgment is enabled or disabled.

(c) Release by $\overline{\text{RESET}}$ input

If a $\overline{\text{RESET}}$ signal is input, the HALT mode is released. As is the case with a normal reset operation, the program is executed after branch to the reset vector address.

Figure 21-3. HALT Mode Release by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 6.3 MHz

Table 21-2. Operation After HALT Mode Release

Release Source	MK××	PR××	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt servicing execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt servicing execution
	1	×	×	×	HALT mode held
Non-maskable interrupt request	–	–	×	×	Interrupt servicing execution
$\overline{\text{RESET}}$ input	–	–	×	×	Reset processing

Remark ×: Don't care

21.2.2 STOP mode

(1) STOP mode setting and operating status

The STOP mode is set by executing the STOP instruction.

Cautions 1. When the STOP mode is set, the X1 pin is pulled down to GND0, and the X2 pin is internally pulled up to V_{DD} to minimize the leakage current of the crystal oscillator.

2. Because the interrupt request signal is used to release the standby mode, if there is an interrupt source with the interrupt request flag set and the interrupt mask flag reset, the standby mode is immediately released if set. Thus, the STOP mode is reset to the HALT mode immediately after execution of the STOP instruction, and the operating mode is set after the wait set using the oscillation stabilization time select register (OSTS).

The operating status in the STOP mode is described below.

Table 21-3. STOP Mode Operating Status

Item	Status
Clock generator	System clock stopped. Clock supply to CPU stopped.
CPU	Stops operating.
Ports	Hold status before STOP mode was set.
16-bit timer/event counter 0	Stop operating and cannot operate.
8-bit timer/event counters 50, 51	
Basic timer	
Watchdog timer	
Buzzer output controller	
A/D converter	
Serial interface SIO0, SIO1, SIO3, UART0[Note 1]	
IEBus controller[Note 2]	
External interrupt	Can operate.
PLL frequency synthesizer	Stop operating and cannot operate.
Frequency counter	
Power-on clear circuit	$\overline{\text{RESET}}$ generated when 2.3 V or less is detected.

Notes 1. μPD178076, 178078, and 178F098 only.

2. μPD178096A, 178098A, and 178F098 only.

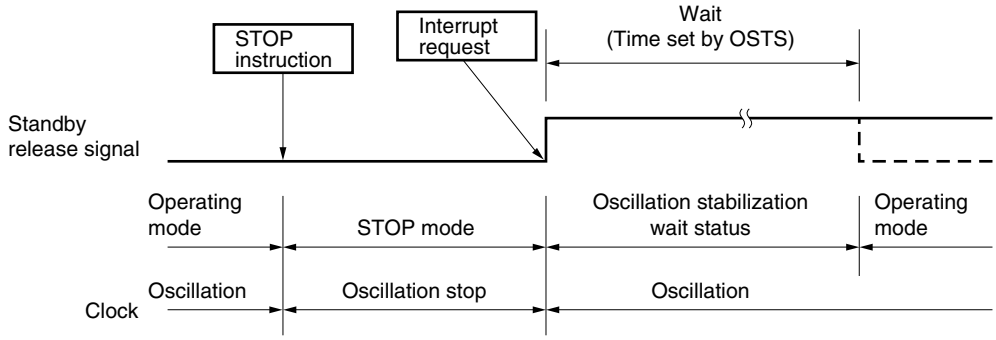
(2) STOP mode release

The STOP mode can be released by the following two sources.

(a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the STOP mode is released. If interrupt request acknowledgment is enabled after the lapse of the oscillation stabilization time, vectored interrupt servicing is carried out. If interrupt request acknowledgment is disabled, the next address instruction is executed.

Figure 21-4. STOP Mode Release by Interrupt Request Generation

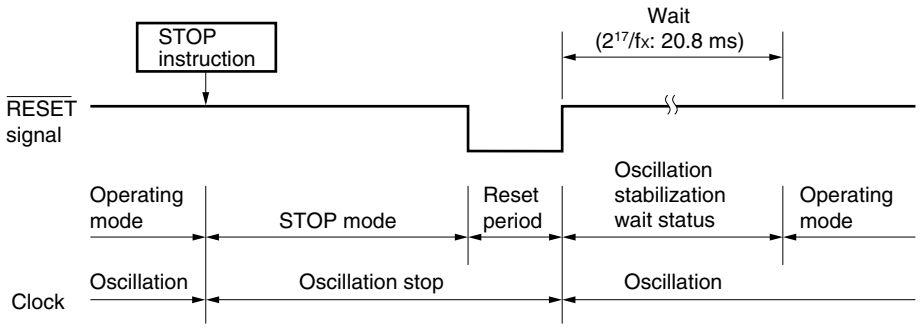


Remark The broken lines indicate the case when the interrupt request which has released the standby status is acknowledged.

(b) Release by $\overline{\text{RESET}}$ input

If a $\overline{\text{RESET}}$ signal is input, the STOP mode is released, and after the lapse of oscillation stabilization time, the reset operation is carried out.

Figure 21-5. Release by STOP Mode $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 6.3 MHz

Table 21-4. Operation After STOP Mode Release

Release Source	MK××	PR××	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt servicing execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt servicing execution
	1	×	×	×	STOP mode held
$\overline{\text{RESET}}$ input	–	–	×	×	Reset processing

Remark ×: Don't care

CHAPTER 22 RESET FUNCTION

22.1 Reset Function

The following three operations are available to generate the reset signal.

(1) External reset input by $\overline{\text{RESET}}$ pin
(2) Internal reset by inadvertent program loop time detection of watchdog timer
(3) Internal reset by power-on clear (POC)

(1) External reset input by $\overline{\text{RESET}}$ pin

When a low level is input to the $\overline{\text{RESET}}$ pin, the device is reset, and each hardware unit enters the status shown in Table 22-1. While the reset signal is input and during the oscillation stabilization time immediately after the $\overline{\text{RESET}}$ signal has been deasserted, each pin goes into a high-impedance state (however, the P130 through P137 pins go low, and the VCOH and VCOL pins are pulled down).

The $\overline{\text{RESET}}$ signal is deasserted when a high level is input to the $\overline{\text{RESET}}$ pin, and program execution is started after the oscillation stabilization time (2^{17}/fx) has elapsed.

(2) Internal reset by inadvertent program loop time detection of watchdog timer

Reset is effected and each hardware unit enters the status shown in Table 22-1 when the watchdog timer overflows. While reset is effected and during the oscillation stabilization time immediately after the effect of reset has been cleared, each pin goes into a high-impedance state (however, the P130 to P137 pins go low, and the VCOH and VCOL pins are pulled down).

Reset by the watchdog timer is cleared immediately after reset has been effected, and program execution is started after the oscillation stabilization time (2^{17}/fx) has elapsed.

(3) Internal reset by power-on clear (POC)

Reset is effected by means of power-on clear under the following conditions.

- If supply voltage is less than 3.5 V**Note** on power application
- If supply voltage drops to less than 2.3 V**Note** in STOP mode
- If supply voltage drops to less than 3.5 V**Note** (including in HALT mode)

When these power-on clear reset conditions are satisfied, reset is effected, and each hardware unit enters the status shown in Table 22-1. While the reset signal is input and during the oscillation stabilization time immediately after the reset signal has been deasserted, each pin goes into a high-impedance state (the P130 to P137 pins go low, however).

Reset by power-on clear is cleared if the supply voltage rises beyond a specific level, and program execution is started after the oscillation stabilization time (2^{17}/fx) has elapsed.

Note These voltage values are maximum values. Actually, reset is effected at a voltage lower than these.

Cautions 1. For an external reset, input a low level to the $\overline{\text{RESET}}$ pin for 10 μs or more.

2. During reset input, system clock oscillation remains stopped.

3. When the STOP mode is released by $\overline{\text{RESET}}$ input, the STOP mode register contents are held during reset input. However, the I/O port pins become high-impedance. Output port pins (P130 to P137) become low level regardless of the previous status.

Figure 22-1. Reset Function Block Diagram

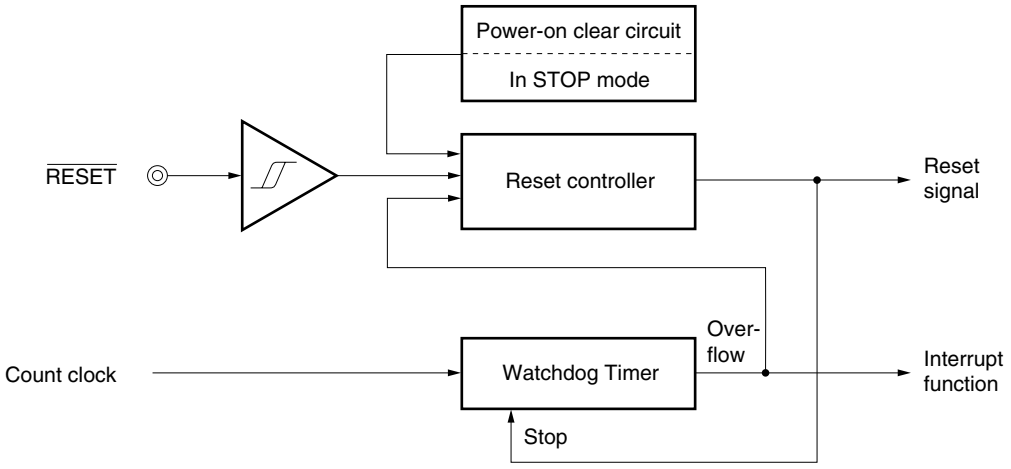
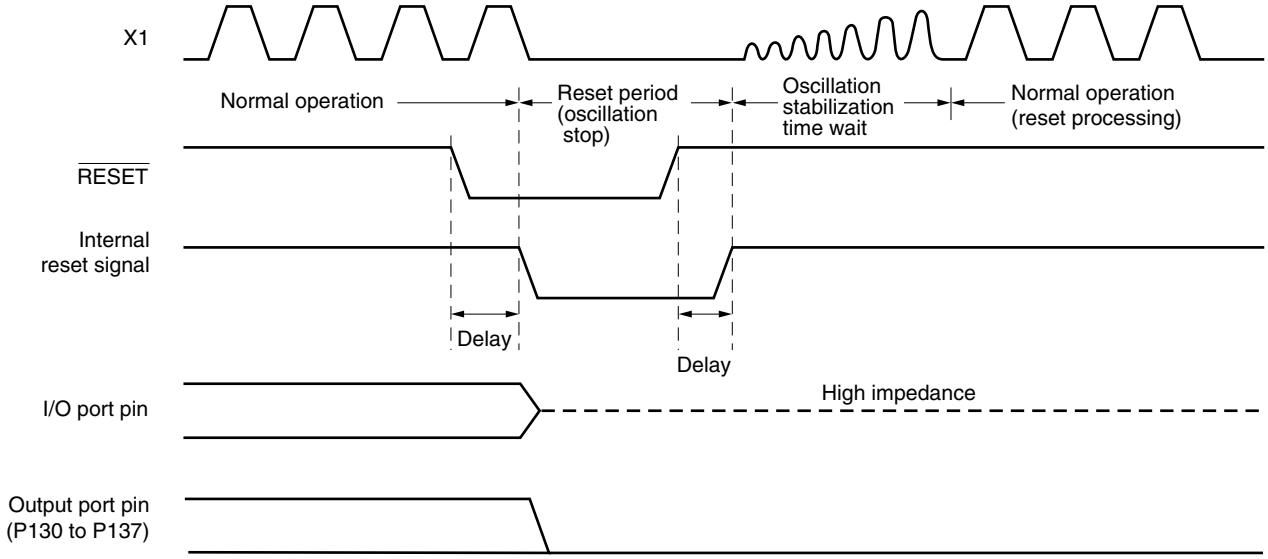


Figure 22-2. Timing of Reset by $\overline{\text{RESET}}$ Input

(a) In normal operating mode



(b) In STOP mode

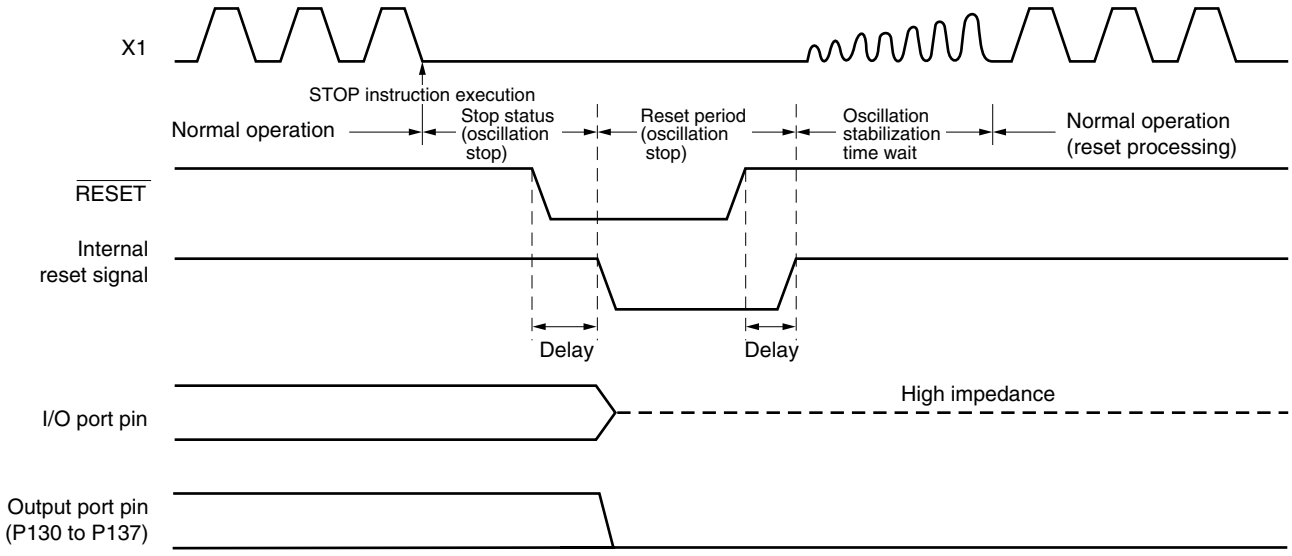


Figure 22-3. Timing of Reset due to Watchdog Timer Overflow

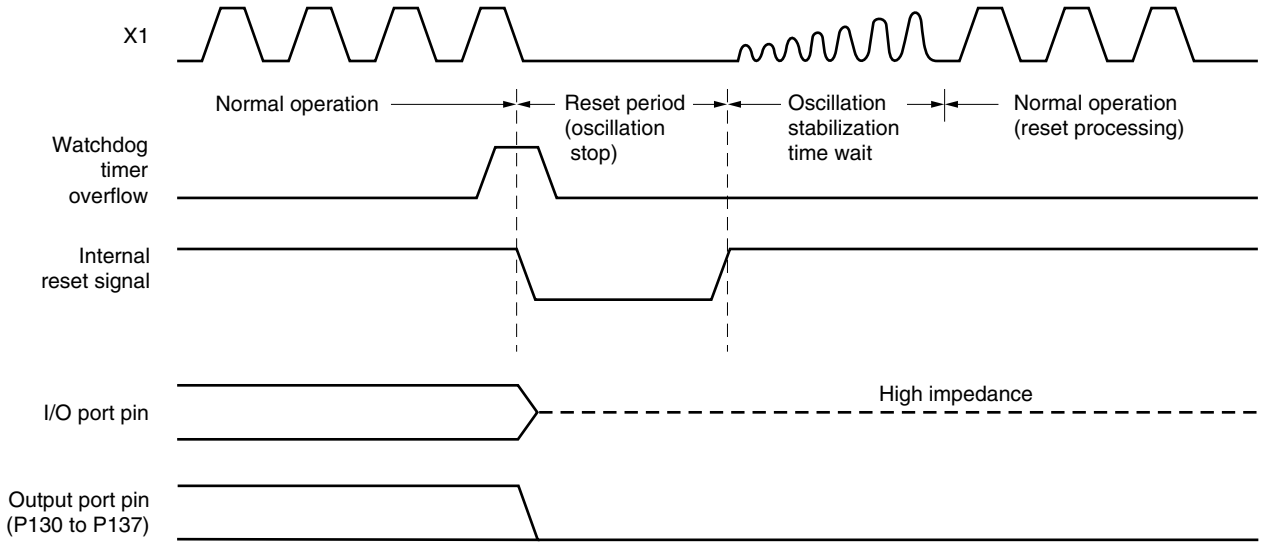
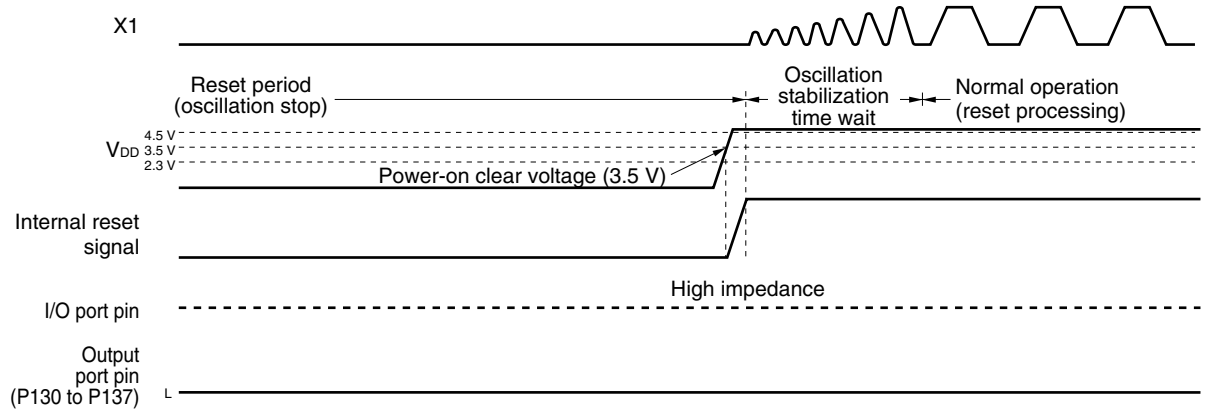
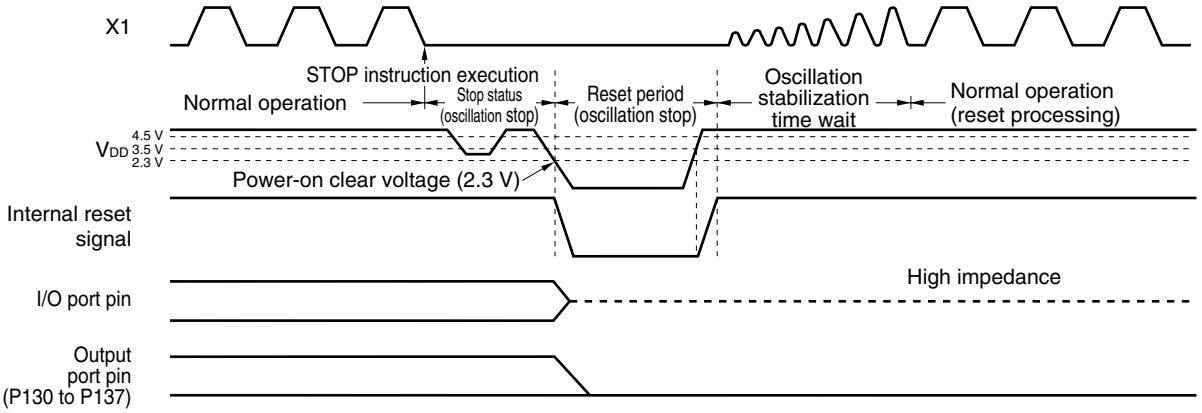


Figure 22-4. Timing of Reset by Power-on Clear

(a) At Power-on



(b) In STOP mode



(c) In normal operating mode (including HALT mode)

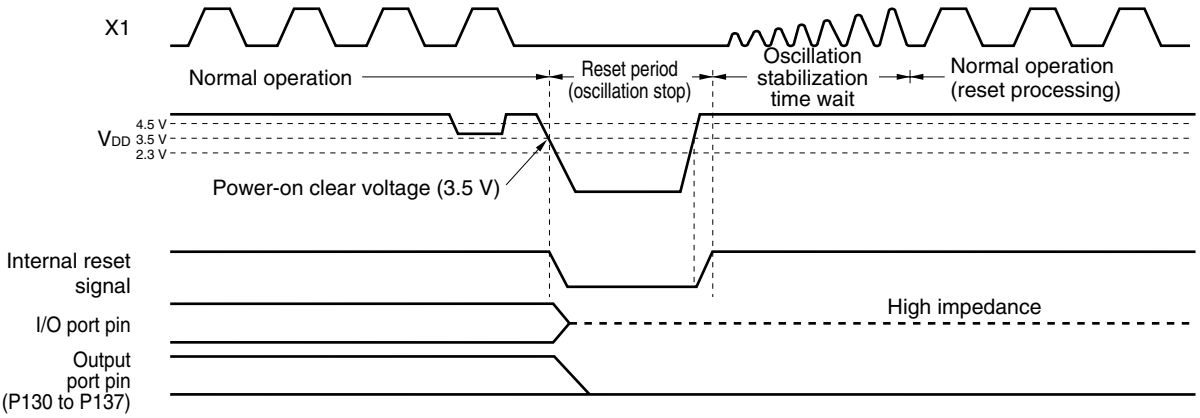


Table 22-1. Hardware Status After Reset (1/3)

Hardware		Status After Reset
Program counter (PC)[Note 1]		Contents of reset vector table (0000H, 0001H) are set.
Stack pointer (SP)		Undefined
Program status word (PSW)		Undefined
RAM	Data memory	Undefined[Note 2]
	General-purpose register	Undefined[Note 2]
Ports (output latches)	Ports 0, 2, 3, 7, 10, 12, 13 (P0, P2, P3, P7, P10, P12, P13)	00H
	Ports 4 to 6 (P4 to P6)	Undefined
Port mode registers (PM0, PM2 to PM7, PM10, PM12)		FFH
Processor clock control register (PCC)		04H
Oscillation stabilization time select register (OSTS)		04H
Memory size select register (IMS)		CFH[Note 3]
Internal expansion RAM size select register (IXS)		0CH[Note 4]
16-bit timer/event counter 0	Timer register (TM0)	0000H
	Capture/compare registers 00, 01 (CR00, CR01)	Undefined
	Mode control register 0 (TMC0)	00H
	Prescaler mode register 0 (PRM0)	00H
	Capture/compare control register 0 (CRC0)	00H
	Output control register 0 (CRC0)	00H
8-bit timer/event counters 50, 51	Counters 50, 51 (TM50, TM51)	00H
	Compare registers 50, 51 (CR50, CR51)	Undefined
	Clock select registers 50, 51 (TCL50, TCL51)	00H
	Mode control registers 50, 51 (TMC50, TMC51)	00H

Notes 1. During reset input or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.

2. The status before reset is held even after reset in the standby mode.

3. The initial value is CFH. Set the following value to this register according to the product.
μPD178076, 178096A: CCH
μPD178078, 178098A: CFH
μPD178F098: Value corresponding to mask ROM version

★ **4.** The initial value is 0CH. Set the following value to this register according to the product.
μPD178076, 178096A: 0AH
μPD178078, 178098A: 08H
μPD178F098: Value corresponding to mask ROM version

Table 22-1. Hardware Status After Reset (2/3)

Hardware			Status After Reset
Watchdog timer		Clock select register (WDCS)	00H
		Mode register (WDTM)	00H
Buzzer output controller	BEEP0	Frequency select register 0 (BEEPCL0)	00H
	BUZ	Clock output control register (CKS)	00H
Serial interface	SIO0	Shift register 0 (SIO0)	Undefined
		Slave address register 0 (SVA0)	Undefined
		Clock select register 0 (SCL0)	08H
		Operating mode register 0 (CSIM0)	00H
		Serial bus interface control register 0 (SBIC0)	00H
		Interrupt timing specification register 0 (SINT0)	00H
	SIO1	Shift register 1 (SIO1)	Undefined
		Operating mode register 1 (CSIM1)	00H
		Automatic data transmit/receive address pointer register (ADTP)	00H
		Automatic data transmit/receive control register (ADTC)	00H
		Automatic data transmit/receive transfer interval specification register (ADTI)	00H
	SIO3	Shift register 3 (SIO3)	Undefined
		Operating mode register 3 (CSIM3)	00H
	UART0[Note 1]	Asynchronous serial interface mode register 0 (ASIM0)	00H
		Asynchronous serial interface status register 0 (ASIS0)	00H
		Baud rate generator control register 0 (BRGC0)	00H
		Transmit shift register 0/receive buffer register 0 (TXS0, RXB0)	FFH
IEBus controller[Note 2]		IEBus control register 0 (BCR0)	00H
		IEBus unit address register (UAR)	0000H
		IEBus slave address register (SAR)	0000H
		IEBus other unit address register (PAR)	0000H
		IEBus control data register (CDR)	01H
		IEBus telegraph length register (DLR)	01H
		IEBus data register (DR)	00H
		IEBus unit status register (USR)	00H
		IEBus interrupt status register (ISR)	00H
		IEBus slave status register (SSR)	41H
		IEBus communication success counter (SCR)	01H
		IEBus transmission counter (CCR)	20H

Notes 1. μPD178076, 178078, and 178F098 only

2. μPD178096A, 178098A, and 178F098 only

Table 22-1. Hardware Status After Reset (3/3)

Hardware		Status After Reset
A/D converter	Mode register 3 (ADM3)	00H
	A/D conversion result register 3 (ADCR3)	Undefined
	Analog input channel specification register 3 (ADS3)	00H
	Power-fail comparison mode register 3 (PFM3)	00H
	Power-fail comparison threshold value register 3 (PFT3)	00H
Interrupt	Request flag registers (IF0L, IF0H, and IF1L)	00H
	Mask flag registers (MK0L, MK0H, and MK1L)	FFH
	Priority specification flag registers (PR0L, PR0H, and PR1L)	FFH
	External interrupt rising edge enable register (EGP)	00H
	External interrupt falling edge enable register (EGN)	00H
PLL frequency synthesizer	PLL mode select register (PLLMD)	00H
	PLL reference mode register (PLLRF)	0FH
	PLL unlock FF judge register (PLLUL)	Held**Note 1**
	PLL data registers (PLLRH, PLLRL, and PLLR0)	Undefined
	PLL data transfer control register (PLLNS)	00H
Frequency counter	IF counter mode select register (IFCMD)	00H
	IF counter gate judge register (IFCJG)	00H
	IF counter control register (IFCCR)	00H
	IF counter data register (IFCR)	0000H
Power-on clear	POC status register (POCS)	Held**Note 2**
	VM45 control register (VM45C)	00H

Notes 1. Undefined only at power-on clear reset

2. 03H only at power-on clear reset

22.2 Power Failure Detection Function

If reset is effected by means of power-on clear, bit 0 (POCM) of the POC status register (POCS) is set to 1. If reset is effected by the $\overline{\text{RESET}}$ pin or the watchdog timer, however, POCM holds the previous status.

A power failure status can be detected by detecting POCM after reset by power-on clear has been released (after program execution has been started from address 0000H).

Figure 22-5. Format of POC Status Register (POCS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
POCS	0	0	0	0	0	0	VM45	POCM	FF1BH	Held[Note]	R&Reset

POCM	Detection of power-on clear occurrence status
0	Power-on clear does not occur
1[Note]	Reset is effected by power-on clear

Note The value of this register is set to 03H only when reset is effected by power-on clear. It is not reset by the $\overline{\text{RESET}}$ pin or watchdog timer.

Remark The values of the special-function registers, other than POCS, are the same as the values after a reset effected by means of power-on clear.

22.3 4.5 V Voltage Detection Function

This function is used to detect a voltage drop on the V_{DD} pin below 4.5 V (4.5 V ± 0.3 V). If the voltage on the V_{DD} pin drops below 4.5 V (4.5 V ± 0.3 V), bit 1 (VM45) of the POC status register (POCS) is set. At the same time, this status can be monitored by the VM45/P30 pin. Therefore, the power to the other peripheral units can be controlled when a voltage of less than 4.5 V is detected.

Note, however, that this 4.5 V voltage detection function does not cause an internal reset.

Figure 22-6. Format of POC Status Register (POCS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
POCS	0	0	0	0	0	0	VM45	POCM	FF1BH	Held[Note]	R&Reset

VM45	Detection of voltage level of V_{DD} pin
0	Not detected if V_{DD} pin is less than 4.5 V (4.3 ± 0.3 V)
1	Detected if V_{DD} pin is less than 4.5 V (4.3 ± 0.3 V)

Note The value of this register is set to 03H only at power-on clear reset, and is not reset by the $\overline{\text{RESET}}$ pin or watchdog timer.

Remark The values of the special-function registers other than POCS after reset are the same as the values at power-on clear.

The status detected (not detected) by VM45 can be output to (monitored by) the VM45/P30 pin under control of the VM45 control register (VM45C).

Figure 22-7. Format of VM45 Control Register (VM45C)

Symbol	7	6	5	4	3	2	<1>	<0>	Address	After reset	R/W
VM45C	0	0	0	0	0	0	VM45C1	VM45C0	FF19H	00H	R/W

VM45C1	Enable/disable of output of VM45 (V_{DD} 4.5 V monitor)
0	Output of VM45 to VM45/P30 pin disabled (port function)
1[Note]	Output of VM45 to VM45/P30 pin enabled

VM45C0	Selection of output of VM45 (V_{DD} 4.5 V monitor)
0	High level output
1	Low level output

Note When the VM45/P30 pin is used for VM45 output, reset the output latches of bit 0 (PM30) of PM3 and bit 0 (P30) of P3 to 0.

CHAPTER 23 μPD178F098

The μPD178F098 is provided with a flash memory to/from which data can be written/erased with the device mounted on the board. The differences between the flash memory version (μPD178F098) and mask ROM versions (μPD178076, 178078, 178096A, and 178098A) are shown in Table 23-1.

Table 23-1. Differences Between μPD178F098 and Mask ROM Versions

Item		μPD178F098	μPD178076, 178078	μPD178096A, 178098A
Internal memory	ROM structure	Flash memory	Mask ROM	
	ROM capacity	60 KB	μPD178076: 48 KB μPD178078: 60 KB	μPD178096A: 48 KB μPD178098A: 60 KB
	Expansion RAM capacity	2048 bytes	μPD178076: 1024 bytes μPD178078: 2048 bytes	μPD178096A: 1024 bytes μPD178098A: 2048 bytes
Internal ROM capacity selected by memory size select register (IMS)		Equivalent to mask ROM version	μPD178076: 0AH μPD178078: 08H	μPD178096A: 0AH μPD178098A: 08H
Internal expansion RAM capacity selected by internal expansion RAM size select register (IXS)		Equivalent to mask ROM version	μPD178076: CCH μPD178078: CFH	μPD178096A: CCH μPD178098A: CFH
Serial interface		4 channels (SIO0, SIO1, SIO3, UART0)		3 channels (SIO0, SIO1, SIO3)
IEBus controller		Provided	Not provided	Provided
IC pin		Not provided	Provided	
V_{PP} pin		Provided	Not provided	

23.1 Memory Size Select Register

The internal memory capacity of the μPD178F098 can be changed using the memory size select register (IMS). By using this register, the memory of the μPD178F098 can be mapped in the same manner as a mask ROM version with a different internal memory capacity.

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

This register is set to CFH after reset.

Be sure to set IMS to CCH or CFH.

Figure 23-1. Format of Memory Size Select Register (IMS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IMS	RAM2	RAM1	RAM0	0	ROM3	ROM2	ROM1	ROM0	FFF0H	CFH	R/W

RAM2	RAM1	RAM0	Selection of internal high-speed RAM capacity
1	1	0	1024 bytes
Other than above			Setting prohibited

RAM3	RAM2	RAM1	RAM0	Selection of internal ROM capacity
1	1	0	0	48 KB
1	1	1	1	60 KB
Other than above				Setting prohibited

Table 23-2 shows the setting of IMS to perform the same memory mapping as that of a mask ROM version.

Table 23-2. Set Value of Memory Size Select Register

Target Version	Set Value of IMS
μPD178076, 178096A	CCH
μPD178078, 178098A	CFH

23.2 Internal Expansion RAM Size Select Register

The internal RAM expansion capacity of the μPD178F098 can be changed using the internal expansion RAM size select register (IXS). By using this register, the memory of the μPD178F098 can be mapped in the same manner as a mask ROM version with a different internal expansion RAM capacity.

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

This register is set to 0CH after reset.

Be sure to set IXS to 0AH or 08H.

Figure 23-2. Format of Internal Expansion RAM Size Select Register (IXS)

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
IXS	0	0	0	IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	FFF4H	0CH	R/W

IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	Selects internal expansion RAM capacity
0	1	0	0	0	2048 bytes
0	1	0	1	0	1024 bytes
Other than above					Setting prohibited

Table 23-3 shows the setting of IXS to perform the same memory mapping as that of a mask ROM version.

Table 23-3. Set Value of Internal Expansion RAM Size Select Register

Target Version	Set Value of IXS
μPD178076, 178096A	0AH
μPD178078, 178098A	08H

★ 23.3 Flash Memory Features

Flash memory programming is performed by connecting a dedicated flash programmer (Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4)) to the target system with the flash memory mounted on the target system (on-board write). A flash memory writing adapter (program adapter), which is a target board used exclusively for programming, is also provided.

Remark FL-PR3, FL-PR4, and the program adapter are products of Naito Densei Machida Mfg. Co., Ltd. (TEL +81-45-475-4191).

Programming using flash memory has the following advantages.

- Software can be modified after the microcontroller is solder-mounted on the target system.
- Distinguishing software facilities low-quantity, varied model production
- Easy data adjustment when starting mass production

23.3.1 Programming environment

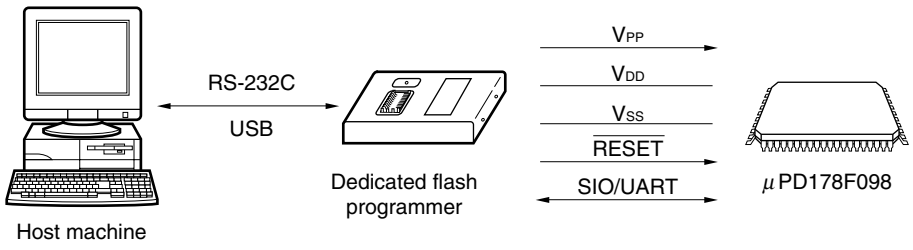
The following shows the environment required for μPD178F098 flash memory programming.

When Flashpro III (part no. FL-PR3, PG-FP3) or Flashpro IV (part no. FL-PR4, PG-FP4) is used as a dedicated flash programmer, a host machine is required to control the dedicated flash programmer. Communication between the host machine and flash programmer is performed via RS-232C/USB (Rev. 1.1).

For details, refer to the manuals for Flashpro III/Flashpro IV.

Remark USB is supported by Flashpro IV only.

Figure 23-3. Environment for Writing Program to Flash Memory



23.3.2 Communication mode

Use the communication mode shown in Table 23-4 to perform communication between the dedicated flash programmer and μPD178F098.

Table 23-4. Communication Mode List

Communication Mode	TYPE Setting[Note 1]					Pins Used	Number of V_{PP} Pulses
	COMM PORT	SIO Clock	CPU CLOCK	Flash Clock	Multiple Rate		
3-wire serial I/O	SIO ch-0 (3 wired, sync.)	100 Hz to 1.25 MHz[Note 2]	Optional	1 to 5 MHz[Note 2]	1.0	P72/$\overline{SCK3}$ P71/SO3 P70/SI3	0
UART (UART0)	UART ch-0 (Async.)	4,800 to 76,800 bps[Notes 2, 3]	Optional	1 to 5 MHz[Note 2]	1.0	P75/TxD0 P74/RxD0	8

Notes 1. Selection items for TYPE settings on the dedicated flash programmer (Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4)).

2. The possible setting range differs depending on the voltage. For details, see **CHAPTER 25 ELECTRICAL SPECIFICATIONS**.

3. Because factors other than the baud rate error, such as the signal waveform slew, also affect UART communication, thoroughly evaluate the slew as well as the baud rate error.

Figure 23-4. Communication Mode Selection Format

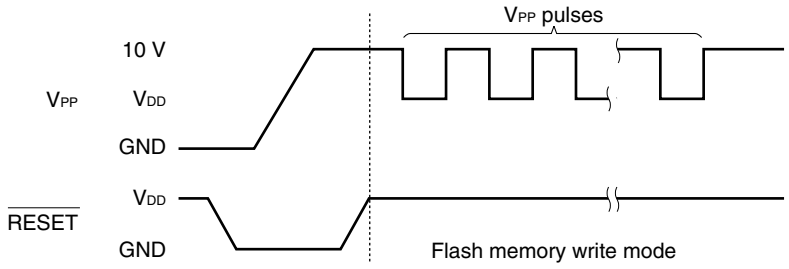
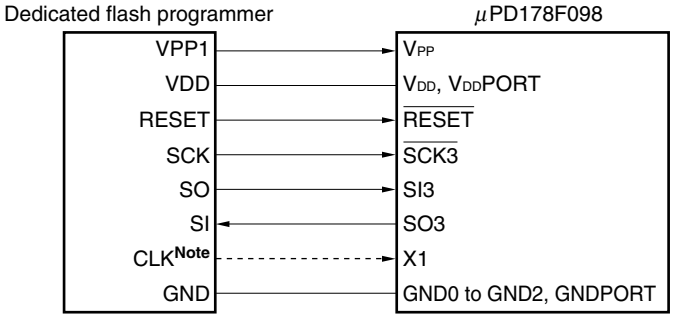
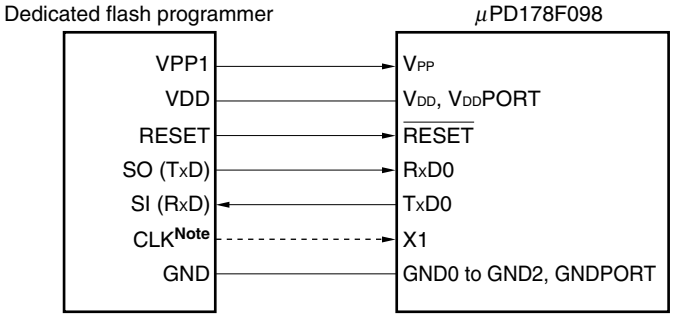


Figure 23-5. Example of Connection with Dedicated Flash Programmer

(a) 3-wire serial I/O



(b) UART



Note Connect this pin when the system clock is supplied from the dedicated flash programmer. If a resonator is already connected to the X1 pin, do not connect to the CLK pin.

Caution The V_{DD} and V_{DD}PORT pins, even if already connected to the power supply, must be connected to the VDD pin of the dedicated flash programmer. When using the power supply connected to the V_{DD} and V_{DD}PORT pins, supply voltage before starting programming.

When Flashpro III/Flashpro IV is used as a dedicated flash programmer, the following signals are generated for the μPD178F098. For details, refer to the manual of Flashpro III/Flashpro IV.

Table 23-5. Pin Connection List

Signal Name	I/O	Pin Function	Pin Name	3-Wire Serial I/O	UART
VPP1	Output	Write voltage	V_{PP}	◎	◎
VPP2	–	–	–	×	×
VDD	I/O	V_{DD} voltage generation/ voltage monitoring	V_{DD}, V_{DD}PORT	◎Note	◎Note
GND	–	Ground	GND0 to GND2, GNDPORT	◎	◎
CLK	Output	Clock output	X1	○	○
RESET	Output	Reset signal	$\overline{\text{RESET}}$	◎	◎
SI (RxD)	Input	Reception signal	SO3/TxD0	◎	◎
SO (TxD)	Output	Transmit signal	SI3/RxD0	◎	◎
SCK	Output	Transfer clock	$\overline{\text{SCK3}}$	◎	×
HS	Input	Handshake signal	–	×	×

Note V_{DD} voltage must be supplied before programming is started.

Remark ◎: Pin must be connected.
○: If the signal is supplied on the target board, pin does not need to be connected.
×: Pin does not need to be connected.

23.3.3 On-board pin handling

When performing programming on the target system, provide a connector on the target system to connect the dedicated flash programmer.

An on-board function that allows switching between normal operation mode and flash memory programming mode may be required in some cases.

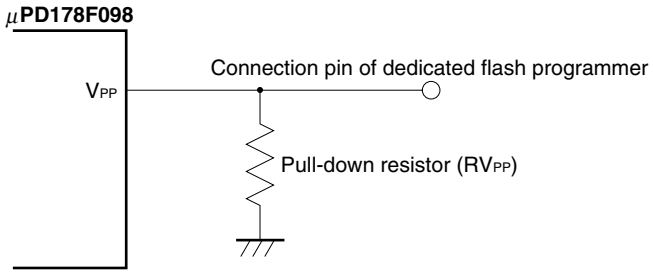
<V_{PP} pin>

In normal operation mode, input 0 V to the V_{PP} pin. In flash memory programming mode, a write voltage of 10.0 V (TYP.) is supplied to the V_{PP} pin, so perform the following.

(1) Connect a pull-down resistor (RV_{PP} = 10 kΩ) to the V_{PP} pin.
(2) Use the jumper on the board to switch the V_{PP} pin input to either the programmer or directly to GND.

A V_{PP} pin connection example is shown below.

Figure 23-6. V_{PP} Pin Connection Example



<Serial interface pins>

The following shows the pins used by the serial interface.

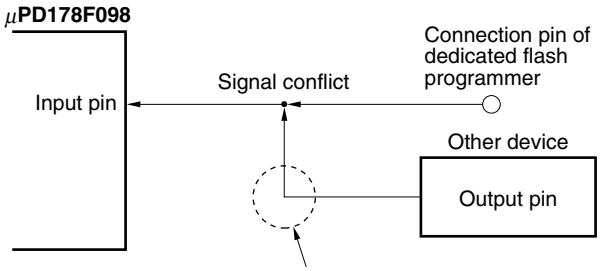
Serial Interface	Pins Used
3-wire serial I/O	SI3, SO3, $\overline{SCK3}$
UART	RxD0, TxD0

When connecting the dedicated flash programmer to a serial interface pin that is connected to another device on-board, signal conflict or abnormal operation of the other device may occur. Care must therefore be taken with such connections.

(1) Signal conflict

If the dedicated flash programmer (output) is connected to a serial interface pin (input) that is connected to another device (output), a signal conflict occurs. To prevent this, isolate the connection with the other device or set the other device to the output high-impedance status.

Figure 23-7. Signal Conflict (Input Pin of Serial Interface)

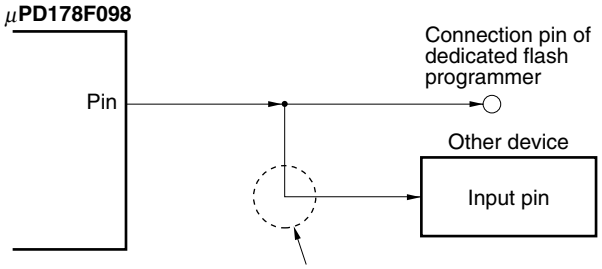


In the flash memory programming mode, the signal output by another device and the signal sent by the dedicated flash programmer conflict, therefore, isolate the signal of the other device.

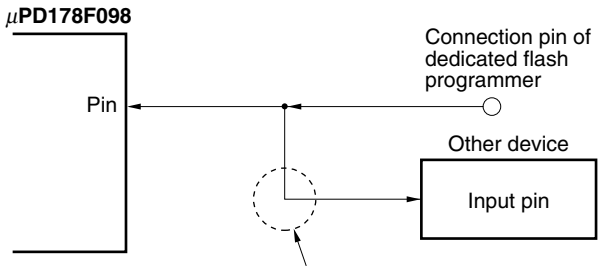
(2) Abnormal operation of other device

If the dedicated flash programmer (output or input) is connected to a serial interface pin (input or output) that is connected to another device (input), a signal is output to the device, and this may cause an abnormal operation. To prevent this abnormal operation, isolate the connection with the other device or set so that the input signals to the other device are ignored.

Figure 23-8. Abnormal Operation of Other Device



If the signal output by the μPD178F098 affects another device in the flash memory programming mode, isolate the signals of the other device.

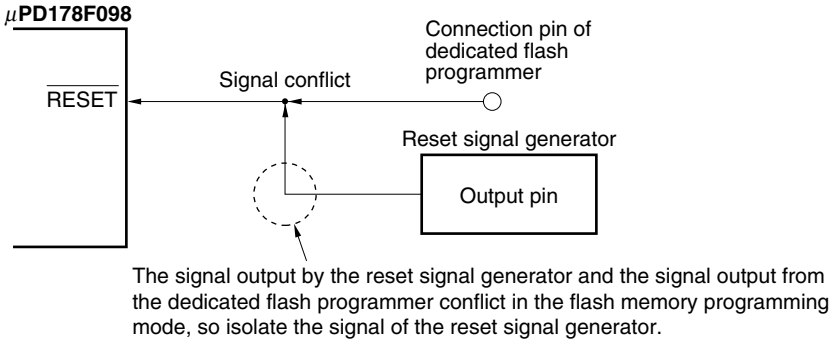


If the signal output by the dedicated flash programmer affects another device in the flash memory programming mode, isolate the signals of the other device.

<$\overline{\text{RESET}}$ pin>

If the reset signal of the dedicated flash programmer is connected to the $\overline{\text{RESET}}$ pin connected to the reset signal generator on-board, a signal conflict occurs. To prevent this, isolate the connection with the reset signal generator. If the reset signal is input from the user system in the flash memory programming mode, a normal programming operation cannot be performed. Therefore, do not input reset signals from other than the dedicated flash programmer.

Figure 23-9. Signal Conflict ($\overline{\text{RESET}}$ Pin)



<Port pins>

When the μPD178F098 enters the flash memory programming mode, all the pins other than those that communicate in flash memory programming are in the same status as immediately after reset.

If the external device does not recognize initial statuses such as the output high-impedance status, therefore, connect the external device to V_{DD} or V_{SS} via a resistor.

<Oscillator>

When using the on-board clock, connect X1 and X2 as required in the normal operation mode.

When using the clock output of the flash programmer, connect it directly to X1, disconnecting the oscillator on-board, and leave the X2 pin open.

<Power supply>

To use the power output from the flash programmer, connect the V_{DD} and V_{DD}PORT pins to VDD of the flash programmer, and the GND0 to GND2 and GNDPORT pins to GND of the flash programmer.

To use the on-board power supply, make connections that accord with the normal operation mode. However, because the voltage is monitored by the flash programmer, be sure to connect VDD of the flash programmer.

Supply the same power as in the normal operation mode to the other power supply pins (AV_{DD}, AV_{SS}, V_{DD}PLL, and GNDPLL).

23.3.4 Connection of adapter for flash writing

The following figures show examples of the recommended connection when the adapter for flash writing is used.

Figure 23-10. Wiring Example for Flash Writing Adapter in 3-Wire Serial I/O Mode

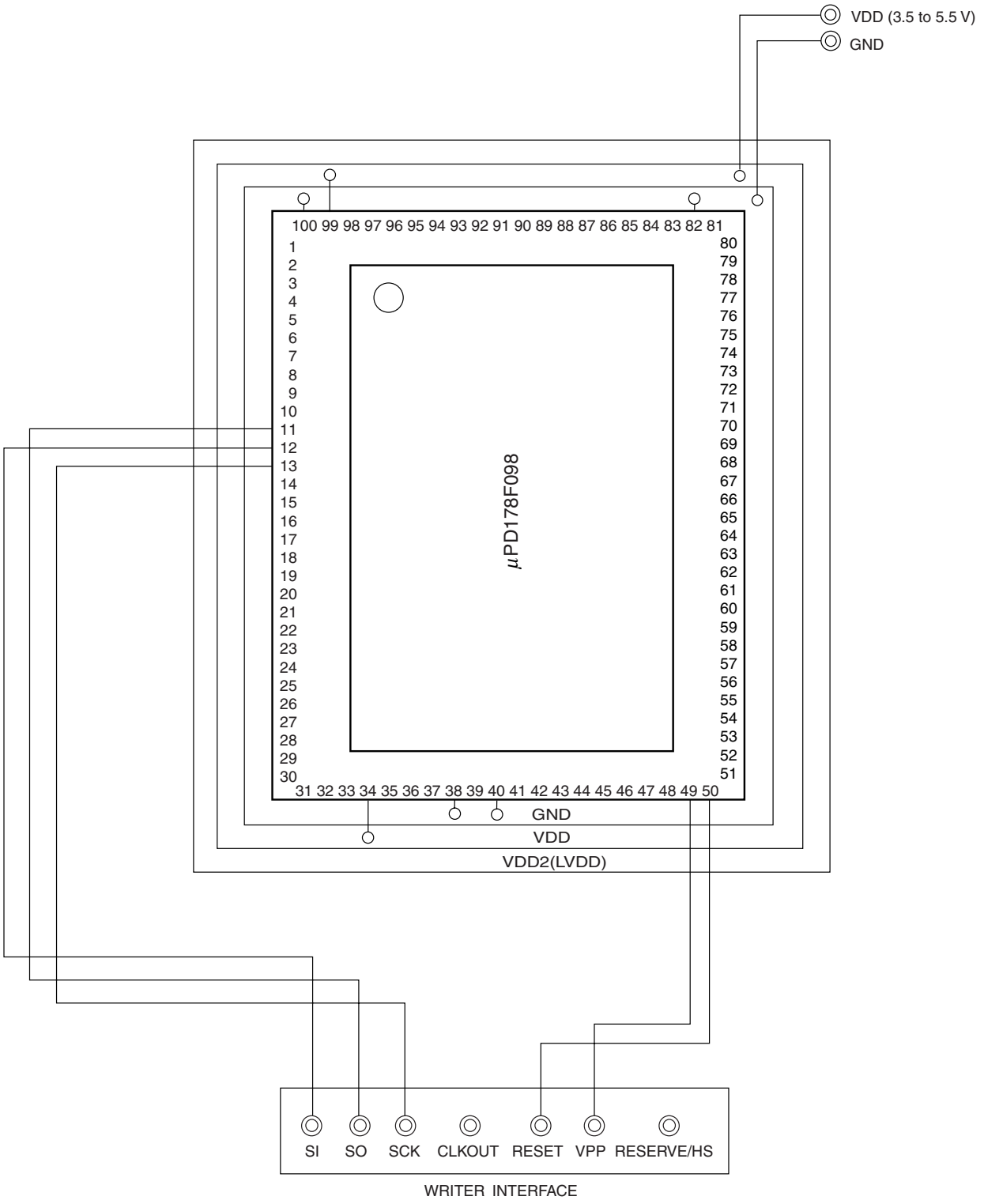
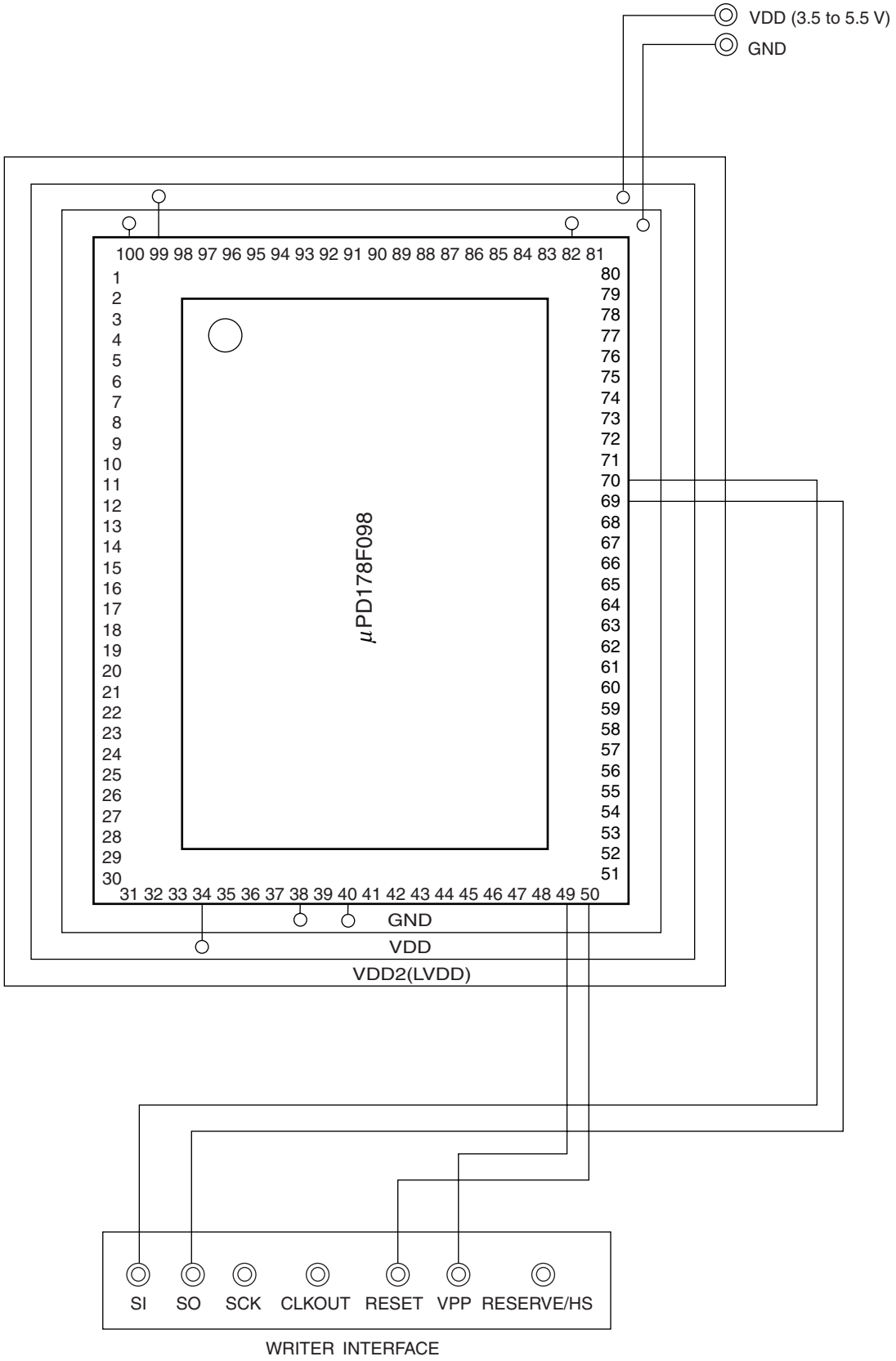


Figure 23-11. Wiring Example for Flash Writing Adapter in UART



CHAPTER 24 INSTRUCTION SET

This chapter describes each instruction set of the μPD178078 and 178098A Subseries as list tables. For details of the operation and operation code of each instruction, refer to the separate document **78K/0 Series Instruction User's Manual (U12326E)**.

24.1 Conventions

24.1.1 Operand identifiers and description method

Operands are written in the “Operand” column of each instruction in accordance with the description of the instruction operand identifiers (refer to the assembler specifications for details). When there are two or more descriptions, select one of them. Uppercase letters and the symbols #, !, $, and [ ] are keywords and must be written as they are. Each symbol has the following meaning.

- #: Immediate data specification
- !: Absolute address specification
- $: Relative address specification
- [ ]: Indirect address specification

In the case of immediate data, write an appropriate numeric value or a label. When using a label, be sure to write the #, !, $, and [ ] symbols.

For the operand register symbols, r and rp, either function names (X, A, C, etc.) or absolute names (names in parentheses in the table below, R0, R1, R2, etc.) can be used.

Table 24-1. Operand Symbols and Descriptions

Symbol	Description
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7),
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special-function register symbol**Note**
sfrp	Special-function register symbol (16-bit manipulatable register, even addresses only)**Note**
saddr	FE20H to FF1FH Immediate data or labels
saddrp	FE20H to FF1FH Immediate data or labels (even addresses only)
addr16	0000H to FFFFH Immediate data or labels (Only even addresses for 16-bit data transfer instructions)
addr11	0800H to 0FFFH Immediate data or labels
addr5	0040H to 007FH Immediate data or labels (even addresses only)
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
RBn	RB0 to RB3

Note Addresses from FFD0H to FFDFH cannot be accessed with these operands.

Remark For special-function register symbols, refer to **Table 3-4 Special-Function Register List**.

24.1.2 Description of “operation” column

A:	A register; 8-bit accumulator
X:	X register
B:	B register
C:	C register
D:	D register
E:	E register
H:	H register
L:	L register
AX:	AX register pair; 16-bit accumulator
BC:	BC register pair
DE:	DE register pair
HL:	HL register pair
PC:	Program counter
SP:	Stack pointer
PSW:	Program status word
CY:	Carry flag
AC:	Auxiliary carry flag
Z:	Zero flag
RBS:	Register bank select flag
IE:	Interrupt request enable flag
NMIS:	Non-maskable interrupt servicing flag
( ):	Memory contents indicated by address or register contents in parentheses
$\times_H$, $\times_L$:	Higher 8 bits and lower 8 bits of 16-bit register
$\wedge$:	Logical product (AND)
$\vee$:	Logical sum (OR)
$\veebar$:	Exclusive logical sum (exclusive OR)
——:	Inverted data
addr16:	16-bit immediate data or label
jdisp8:	Signed 8-bit data (displacement value)

24.1.3 Description of “flag operation” column

(Blank):	Not affected
0:	Cleared to 0
1:	Set to 1
$\times$:	Set/cleared according to the result
R:	Previously saved value is restored

24.2 Operation List

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit data transfer	**MOV**	r, #byte	2	4	–	r ← byte	
		saddr, #byte	3	6	7	(saddr) ← byte	
		sfr, #byte	3	–	7	sfr ← byte	
		A, r **Note 3**	1	2	–	A ← r	
		r, A **Note 3**	1	2	–	r ← A	
		A, saddr	2	4	5	A ← (saddr)	
		saddr, A	2	4	5	(saddr) ← A	
		A, sfr	2	–	5	A ← sfr	
		sfr, A	2	–	5	sfr ← A	
		A, !addr16	3	8	9	A ← (addr16)	
		!addr16, A	3	8	9	(addr16) ← A	
		PSW, #byte	3	–	7	PSW ← byte	× × ×
		A, PSW	2	–	5	A ← PSW	
		PSW, A	2	–	5	PSW ← A	× × ×
		A, [DE]	1	4	5	A ← (DE)	
		[DE], A	1	4	5	(DE) ← A	
		A, [HL]	1	4	5	A ← (HL)	
		[HL], A	1	4	5	(HL) ← A	
		A, [HL + byte]	2	8	9	A ← (HL + byte)	
		[HL + byte], A	2	8	9	(HL + byte) ← A	
		A, [HL + B]	1	6	7	A ← (HL + B)	
		[HL + B], A	1	6	7	(HL + B) ← A	
		A, [HL + C]	1	6	7	A ← (HL + C)	
		[HL + C], A	1	6	7	(HL + C) ← A	
	XCH	A, r **Note 3**	1	2	–	A ↔ r	
		A, saddr	2	4	6	A ↔ (saddr)	
		A, sfr	2	–	6	A ↔ sfr	
		A, !addr16	3	8	10	A ↔ (addr16)	
		A, [DE]	1	4	6	A ↔ (DE)	
		A, [HL]	1	4	6	A ↔ (HL)	
		A, [HL + byte]	2	8	10	A ↔ (HL + byte)	
		A, [HL + B]	2	8	10	A ↔ (HL + B)	
		A, [HL + C]	2	8	10	A ↔ (HL + C)	

Notes 1. When the internal high-speed RAM area is accessed or an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed.

3. Except "r = A"

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit data transfer	**MOVW**	rp, #word	3	6	–	rp ← word			
		saddrp, #word	4	8	10	(saddrp) ← word			
		sfrp, #word	4	–	10	sfrp ← word			
		AX, saddrp	2	6	8	AX ← (saddrp)			
		saddrp, AX	2	6	8	(saddrp) ← AX			
		AX, sfrp	2	–	8	AX ← sfrp			
		sfrp, AX	2	–	8	sfrp ← AX			
		AX, rp **Note 3**	1	4	–	AX ← rp			
		rp, AX **Note 3**	1	4	–	rp ← AX			
		AX, !addr16	3	10	12	AX ← (addr16)			
		!addr16, AX	3	10	12	(addr16) ← AX			
	XCHW	AX, rp **Note 3**	1	4	–	AX ↔ rp			
8-bit operation	**ADD**	A, #byte	2	4	–	A, CY ← A + byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte	×	×	×
		A, r **Note 4**	2	4	–	A, CY ← A + r	×	×	×
		r, A	2	4	–	r, CY ← r + A	×	×	×
		A, saddr	2	4	5	A, CY ← A + (saddr)	×	×	×
		A, !addr16	3	8	9	A, CY ← A + (addr16)	×	×	×
		A, [HL]	1	4	5	A, CY ← A + (HL)	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A + (HL + byte)	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A + (HL + B)	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A + (HL + C)	×	×	×
	ADDC	A, #byte	2	4	–	A, CY ← A + byte + CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte + CY	×	×	×
		A, r **Note 4**	2	4	–	A, CY ← A + r + CY	×	×	×
		r, A	2	4	–	r, CY ← r + A + CY	×	×	×
		A, saddr	2	4	5	A, CY ← A + (saddr) + CY	×	×	×
		A, !addr16	3	8	9	A, CY ← A + (addr16) + CY	×	×	×
		A, [HL]	1	4	5	A, CY ← A + (HL) + CY	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A + (HL + byte) + CY	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A + (HL + B) + CY	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A + (HL + C) + CY	×	×	×

Notes 1. When the internal high-speed RAM area is accessed or an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Only when rp = BC, DE or HL

4. Except "r = A"

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	**SUB**	A, #byte	2	4	–	A, CY ← A – byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte	×	×	×
		A, r Note 3	2	4	–	A, CY ← A – r	×	×	×
		r, A	2	4	–	r, CY ← r – A	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr)	×	×	×
		A, !addr16	3	8	9	A, CY ← A – (addr16)	×	×	×
		A, [HL]	1	4	5	A, CY ← A – (HL)	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A – (HL + byte)	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A – (HL + B)	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A – (HL + C)	×	×	×
	SUBC	A, #byte	2	4	–	A, CY ← A – byte – CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte – CY	×	×	×
		A, r Note 3	2	4	–	A, CY ← A – r – CY	×	×	×
		r, A	2	4	–	r, CY ← r – A – CY	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr) – CY	×	×	×
		A, !addr16	3	8	9	A, CY ← A – (addr16) – CY	×	×	×
		A, [HL]	1	4	5	A, CY ← A – (HL) – CY	×	×	×
		A, [HL + byte]	2	8	9	A, CY ← A – (HL + byte) – CY	×	×	×
		A, [HL + B]	2	8	9	A, CY ← A – (HL + B) – CY	×	×	×
		A, [HL + C]	2	8	9	A, CY ← A – (HL + C) – CY	×	×	×
	AND	A, #byte	2	4	–	A ← A ∧ byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr) ∧ byte	×		
		A, r Note 3	2	4	–	A ← A ∧ r	×		
		r, A	2	4	–	r ← r ∧ A	×		
		A, saddr	2	4	5	A ← A ∧ (saddr)	×		
		A, !addr16	3	8	9	A ← A ∧ (addr16)	×		
		A, [HL]	1	4	5	A ← A ∧ [HL]	×		
		A, [HL + byte]	2	8	9	A ← A ∧ [HL + byte]	×		
		A, [HL + B]	2	8	9	A ← A ∧ [HL + B]	×		
		A, [HL + C]	2	8	9	A ← A ∧ [HL + C]	×		

Notes 1. When the internal high-speed RAM area is accessed or an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except "r = A"

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	**OR**	A, #byte	2	4	–	A ← A∨byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr)∨byte	×		
		A, r Note 3	2	4	–	A ← A∨r	×		
		r, A	2	4	–	r ← r∨A	×		
		A, saddr	2	4	5	A ← A∨(saddr)	×		
		A, !addr16	3	8	9	A ← A∨(addr16)	×		
		A, [HL]	1	4	5	A ← A∨(HL)	×		
		A, [HL + byte]	2	8	9	A ← A∨(HL + byte)	×		
		A, [HL + B]	2	8	9	A ← A∨(HL + B)	×		
		A, [HL + C]	2	8	9	A ← A∨(HL + C)	×		
	XOR	A, #byte	2	4	–	A ← A⊻byte	×		
		saddr, #byte	3	6	8	(saddr) ← (saddr)⊻byte	×		
		A, r Note 3	2	4	–	A ← A⊻r	×		
		r, A	2	4	–	r ← r⊻A	×		
		A, saddr	2	4	5	A ← A⊻(saddr)	×		
		A, !addr16	3	8	9	A ← A⊻(addr16)	×		
		A, [HL]	1	4	5	A ← A⊻(HL)	×		
		A, [HL + byte]	2	8	9	A ← A⊻(HL + byte)	×		
		A, [HL + B]	2	8	9	A ← A⊻(HL + B)	×		
		A, [HL + C]	2	8	9	A ← A⊻(HL + C)	×		
	CMP	A, #byte	2	4	–	A – byte	×	×	×
		saddr, #byte	3	6	8	(saddr) – byte	×	×	×
		A, r Note 3	2	4	–	A – r	×	×	×
		r, A	2	4	–	r – A	×	×	×
		A, saddr	2	4	5	A – (saddr)	×	×	×
		A, !addr16	3	8	9	A – (addr16)	×	×	×
		A, [HL]	1	4	5	A – (HL)	×	×	×
		A, [HL + byte]	2	8	9	A – (HL + byte)	×	×	×
		A, [HL + B]	2	8	9	A – (HL + B)	×	×	×
		A, [HL + C]	2	8	9	A – (HL + C)	×	×	×

Notes 1. When the internal high-speed RAM area is accessed or an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

3. Except "r = A"

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit operation	**ADDW**	AX, #word	3	6	–	AX, CY ← AX + word	×	×	×
	SUBW	AX, #word	3	6	–	AX, CY ← AX – word	×	×	×
	CMPW	AX, #word	3	6	–	AX – word	×	×	×
Multiply/ divide	**MULU**	X	2	16	–	AX ← A × X			
	DIVUW	C	2	25	–	AX (Quotient), C (Remainder) ← AX ÷ C			
Increment/ decrement	**INC**	r	1	2	–	r ← r + 1	×	×	
		saddr	2	4	6	(saddr) ← (saddr) + 1	×	×	
	DEC	r	1	2	–	r ← r – 1	×	×	
		saddr	2	4	6	(saddr) ← (saddr) – 1	×	×	
	INCW	rp	1	4	–	rp ← rp + 1			
	DECW	rp	1	4	–	rp ← rp – 1			
Rotate	**ROR**	A, 1	1	2	–	(CY, A_7 ← A_0, A_{m-1} ← A_m) × 1 time			×
	ROL	A, 1	1	2	–	(CY, A_0 ← A_7, A_{m+1} ← A_m) × 1 time			×
	RORC	A, 1	1	2	–	(CY ← A_0, A_7 ← CY, A_{m-1} ← A_m) × 1 time			×
	ROLC	A, 1	1	2	–	(CY ← A_7, A_0 ← CY, A_{m+1} ← A_m) × 1 time			×
	ROR4	[HL]	2	10	12	A_{3-0} ← $(HL)_{3-0}$, $(HL)_{7-4}$ ← A_{3-0}, $(HL)_{3-0}$ ← $(HL)_{7-4}$			
	ROL4	[HL]	2	10	12	A_{3-0} ← $(HL)_{7-4}$, $(HL)_{3-0}$ ← A_{3-0}, $(HL)_{7-4}$ ← $(HL)_{3-0}$			
BCD adjust	**ADJBA**		2	4	–	Decimal Adjust Accumulator after Addition	×	×	×
	ADJBS		2	4	–	Decimal Adjust Accumulator after Subtract	×	×	×
Bit manipulate	**MOV1**	CY, saddr.bit	3	6	7	CY ← (saddr.bit)			×
		CY, sfr.bit	3	–	7	CY ← sfr.bit			×
		CY, A.bit	2	4	–	CY ← A.bit			×
		CY, PSW.bit	3	–	7	CY ← PSW.bit			×
		CY, [HL].bit	2	6	7	CY ← (HL).bit			×
		saddr.bit, CY	3	6	8	(saddr.bit) ← CY			
		sfr.bit, CY	3	–	8	sfr.bit ← CY			
		A.bit, CY	2	4	–	A.bit ← CY			
		PSW.bit, CY	3	–	8	PSW.bit ← CY	×	×	
		[HL].bit, CY	2	6	8	(HL).bit ← CY			

Notes 1. When the internal high-speed RAM area is accessed or an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Bit manipulate	**AND1**	CY, saddr.bit	3	6	7	CY ← CY ∧ (saddr.bit)			×
		CY, sfr.bit	3	–	7	CY ← CY ∧ sfr.bit			×
		CY, A.bit	2	4	–	CY ← CY ∧ A.bit			×
		CY, PSW.bit	3	–	7	CY ← CY ∧ PSW.bit			×
		CY, [HL].bit	2	6	7	CY ← CY ∧ (HL).bit			×
	OR1	CY, saddr.bit	3	6	7	CY ← CY ∨ (saddr.bit)			×
		CY, sfr.bit	3	–	7	CY ← CY ∨ sfr.bit			×
		CY, A.bit	2	4	–	CY ← CY ∨ A.bit			×
		CY, PSW.bit	3	–	7	CY ← CY ∨ PSW.bit			×
		CY, [HL].bit	2	6	7	CY ← CY ∨ (HL).bit			×
	XOR1	CY, saddr.bit	3	6	7	CY ← CY ⩛ (saddr.bit)			×
		CY, sfr.bit	3	–	7	CY ← CY ⩛ sfr.bit			×
		CY, A.bit	2	4	–	CY ← CY ⩛ A.bit			×
		CY, PSW. bit	3	–	7	CY ← CY ⩛ PSW.bit			×
		CY, [HL].bit	2	6	7	CY ← CY ⩛ (HL).bit			×
	SET1	saddr.bit	2	4	6	(saddr.bit) ← 1			
		sfr.bit	3	–	8	sfr.bit ← 1			
		A.bit	2	4	–	A.bit ← 1			
		PSW.bit	2	–	6	PSW.bit ← 1	×	×	×
		[HL].bit	2	6	8	(HL).bit ← 1			
	CLR1	saddr.bit	2	4	6	(saddr.bit) ← 0			
		sfr.bit	3	–	8	sfr.bit ← 0			
		A.bit	2	4	–	A.bit ← 0			
		PSW.bit	2	–	6	PSW.bit ← 0	×	×	×
		[HL].bit	2	6	8	(HL).bit ← 0			
	SET1	CY	1	2	–	CY ← 1			1
	CLR1	CY	1	2	–	CY ← 0			0
	NOT1	CY	1	2	–	CY ← $\overline{CY}$			×

Notes 1. When the internal high-speed RAM area is accessed or an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Call/return	**CALL**	!addr16	3	7	–	(SP – 1) ← (PC + 3)$_H$, (SP – 2) ← (PC + 3)$_L$, PC ← addr16, SP ← SP – 2			
	CALLF	!addr11	2	5	–	(SP – 1) ← (PC + 2)$_H$, (SP – 2) ← (PC + 2)$_L$, PC$_{15-11}$ ← 00001, PC$_{10-0}$ ← addr11, SP ← SP – 2			
	CALLT	[addr5]	1	6	–	(SP – 1) ← (PC + 1)$_H$, (SP – 2) ← (PC + 1)$_L$, PC$_H$ ← (00000000, addr5 + 1), PC$_L$ ← (00000000, addr5), SP ← SP – 2			
	BRK		1	6	–	(SP – 1) ← PSW, (SP – 2) ← (PC + 1)$_H$, (SP – 3) ← (PC + 1)$_L$, PC$_H$ ← (003FH), PC$_L$ ← (003EH), SP ← SP – 3, IE ← 0			
	RET		1	6	–	PC$_H$ ← (SP + 1), PC$_L$ ← (SP), SP ← SP + 2			
	RETI		1	6	–	PC$_H$ ← (SP + 1), PC$_L$ ← (SP), PSW ← (SP + 2), SP ← SP + 3, NMIS ← 0	R	R	R
	RETB		1	6	–	PC$_H$ ← (SP + 1), PC$_L$ ← (SP), PSW ← (SP + 2), SP ← SP + 3	R	R	R
Stack manipulate	**PUSH**	PSW	1	2	–	(SP – 1) ← PSW, SP ← SP – 1			
		rp	1	4	–	(SP – 1) ← rp$_H$, (SP – 2) ← rp$_L$, SP ← SP – 2			
	POP	PSW	1	2	–	PSW ← (SP), SP ← SP + 1	R	R	R
		rp	1	4	–	rp$_H$ ← (SP + 1), rp$_L$ ← (SP), SP ← SP + 2			
	MOVW	SP, #word	4	–	10	SP ← word			
		SP, AX	2	–	8	SP ← AX			
		AX, SP	2	–	8	AX ← SP			
Unconditional branch	**BR**	!addr16	3	6	–	PC ← addr16			
		$addr16	2	6	–	PC ← PC + 2 + jdisp8			
		AX	2	8	–	PC$_H$ ← A, PC$_L$ ← X			
Conditional branch	**BC**	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if CY = 1			
	BNC	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if CY = 0			
	BZ	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if Z = 1			
	BNZ	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if Z = 0			

Notes 1. When the internal high-speed RAM area is accessed or an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to the internal ROM program.

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag
				Note 1	Note 2		Z AC CY
Conditional branch	**BT**	saddr.bit, $addr16	3	8	9	PC ← PC + 3 + jdisp8 if(saddr.bit) = 1	
		sfr.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if sfr.bit = 1	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 1	
		PSW.bit, $addr16	3	–	9	PC ← PC + 3 + jdisp8 if PSW.bit = 1	
		[HL].bit, $addr16	3	10	11	PC ← PC + 3 + jdisp8 if (HL).bit = 1	
	BF	saddr.bit, $addr16	4	10	11	PC ← PC + 4 + jdisp8 if(saddr.bit) = 0	
		sfr.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if sfr.bit = 0	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 0	
		PSW.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if PSW. bit = 0	
		[HL].bit, $addr16	3	10	11	PC ← PC + 3 + jdisp8 if (HL).bit = 0	
	BTCLR	saddr.bit, $addr16	4	10	12	PC ← PC + 4 + jdisp8 if(saddr.bit) = 1 then reset(saddr.bit)	
		sfr.bit, $addr16	4	–	12	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit	
		PSW.bit, $addr16	4	–	12	PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit	× × ×
		[HL].bit, $addr16	3	10	12	PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit	
	DBNZ	B, $addr16	2	6	–	B ← B – 1, then PC ← PC + 2 + jdisp8 if B ≠ 0	
		C, $addr16	2	6	–	C ← C –1, then PC ← PC + 2 + jdisp8 if C ≠ 0	
		saddr. $addr16	3	8	10	(saddr) ← (saddr) – 1, then PC ← PC + 3 + jdisp8 if(saddr) ≠ 0	
CPU control	**SEL**	RBn	2	4	–	RBS1, 0 ← n	
	NOP		1	2	–	No Operation	
	EI		2	–	6	IE ← 1(Enable Interrupt)	
	DI		2	–	6	IE ← 0(Disable Interrupt)	
	HALT		2	6	–	Set HALT Mode	
	STOP		2	6	–	Set STOP Mode	

Notes 1. When the internal high-speed RAM area is accessed or an instruction with no data access

2. When an area except the internal high-speed RAM area is accessed

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the PCC register.

2. This clock cycle applies to the internal ROM program.

24.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, ROR4, ROL4, PUSH, POP, DBNZ

1st Operand \ 2nd Operand	#byte	A	r[Note]	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL + byte] [HL + B] [HL + C]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV ADD ADDC SUB SUBC AND OR XOR CMP											INC DEC
B, C											DBNZ		
sfr	MOV	MOV											
saddr	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV									DBNZ		INC DEC
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											ROR4 ROL4
[HL + byte] [HL + B] [HL + C]		MOV											
X													MULU
C													DIVUW

Note Except r = A

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

1st Operand \ 2nd Operand	#word	AX	rp[Note]	sfrp	saddrp	!addr16	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW[Note]						INCW DECW PUSH POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
!addr16		MOVW						
SP	MOVW	MOVW						

Note Only when rp = BC, DE, HL

(3) Bit manipulation instructions

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

1st Operand \ 2nd Operand	A.bit	sfr.bit	saddr.bit	PSW.bit	[HL].bit	CY	$addr16	None
A.bit						MOV1	BT BF BTCLR	SET1 CLR1
sfr.bit						MOV1	BT BF BTCLR	SET1 CLR1
saddr.bit						MOV1	BT BF BTCLR	SET1 CLR1
PSW.bit						MOV1	BT BF BTCLR	SET1 CLR1
[HL].bit						MOV1	BT BF BTCLR	SET1 CLR1
CY	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1			SET1 CLR1 NOT1

(4) Call instructions/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

1st Operand \ 2nd Operand	AX	!addr16	!addr11	[addr5]	$addr16
Basic instruction	BR	CALL BR	CALLF	CALLT	BR BC BNC BZ BNZ
Compound instruction					BT BF BTCLR DBNZ

(5) Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

Absolute Maximum Ratings (T_A = 25°C)

Parameter	Symbol	Conditions		Rating	Unit
Supply voltage	V_{DD}			–0.3 to +6.0	V
	V_{DD}PORT			–0.3 to V_{DD} + 0.3 **Note 1**	V
	AV_{DD}			–0.3 to V_{DD} + 0.3 **Note 1**	V
	V_{DD}PLL			–0.3 to V_{DD} + 0.3 **Note 1**	V
	V_{PP}	μPD178F098 only, **Note 2**		–0.3 to +10.5	V
Input voltage	V_I			–0.3 to V_{DD} + 0.3	V
Output voltage	V_O	Excluding P130 to P137		–0.3 to V_{DD} + 0.3	V
Output breakdown voltage	V_{BDS}	P130 to P137	N-ch open drain	16	V
Analog input voltage	V_{AN}	P10 to P17	Analog input pin	–0.3 to V_{DD} + 0.3	V
High-level output current	I_{OH}	Per pin		–8	mA
		Total of P00, P01, P20 to P27, P50 to P57, and P70 to P73		–15	mA
		Total of P02 to P07, P30 to P37, P40 to P47, P60 to P67, P74 to P77, and P120 to P124		–15	mA
		Total of P100 to P102		–10	mA
Low-level output current	I_{OL} **Note 3**	Per pin	Peak value	16	mA
			r.m.s	8	mA
		Total of P00, P01, P20 to P27, P50 to P57, and P70 to P73	Peak value	30	mA
			r.m.s	15	mA
		Total of P02 to P07, P30 to P37, P40 to P47, P60 to P67, P74 to P77, P120 to P124, and P130 to P137	Peak value	30	mA
			r.m.s	15	mA
		Total of P100 to 102	Peak value	20	mA
			r.m.s	10	mA
Operating temperature	T_A	During normal operation		–40 to +85	°C
		During flash memory programming		10 to 40	°C
Storage temperature	T_{stg}			–55 to +125	°C

(Notes are explained on the next page.)

Caution If the rated value of even one of the above parameters is exceeded even momentarily, the quality of the product may be degraded. The absolute maximum ratings, therefore, are the values exceeding which the product may be physically damaged. Be sure to use the product with these ratings never being exceeded.

Remark Unless otherwise specified, the characteristics of alternate-function pins are the same as those of port pins.

Notes 1. Keep the voltage at V_{DD}PORT, AV_{DD}, and V_{DD}PLL the same as that at the V_{DD} pin.

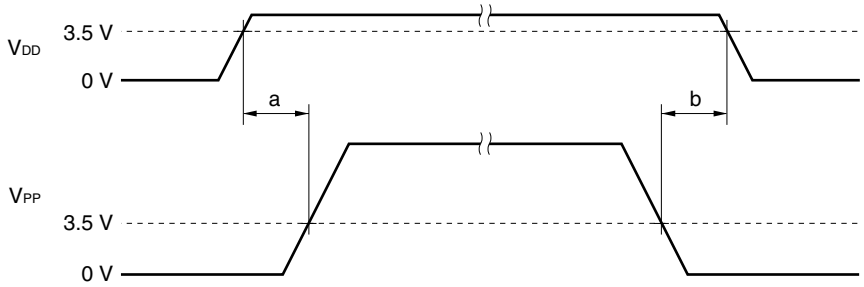
2. Make sure that the following conditions of the V_{PP} voltage application timing are satisfied when the flash memory is written.

- When supply voltage rises

V_{PP} must exceed V_{DD} 1 ms or more after V_{DD} has reached the lower-limit value (3.5 V) of the operating voltage range (see a in the figure below).

- When supply voltage drops

V_{DD} must be lowered 10 μs or more after V_{PP} falls below the lower-limit value (3.5 V) of the operating voltage range of V_{DD} (see b in the figure below).



3. The rms value should be calculated as follows: [rms value] = [Peak value] $\times \sqrt{\text{Duty}}$

Recommended Supply Voltage Ranges (T_A = –40 to +85°C)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Supply voltage	V_{DD1}	When CPU and PLL are operating	4.5	5.0	5.5	V
	V_{DD2}	When CPU is operating and PLL is stopped	3.5	5.0	5.5	V
Data retention voltage	V_{DDR}	When crystal oscillation stops	2.3		5.5	V
Output breakdown voltage	V_{BDS}	P130 to P137 (N-ch open drain)			15	V

DC Characteristics (T_A = –40 to +85°C, V_{DD} = 3.5 to 5.5 V) (1/3)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
High-level input voltage	V_{IH1}	P10 to P17, P21, P23, P30, P31, P36, P37, P40 to P47, P50 to P57, P60 to P67, P71, P73, P75 to P77, P100 to P102, P120, P122 to P124		0.7 V_{DD}		V_{DD}	V
	V_{IH2}	P00 to P07, P20, P22, P24 to P27, P32 to P35, P70, P72, P74, P121, $\overline{\text{RESET}}$		0.8 V_{DD}		V_{DD}	V
Low-level input voltage	V_{IL1}	P10 to P17, P21, P23, P30, P31, P36, P37, P40 to P47, P50 to P57, P60 to P67, P71, P73, P75 to P77, P100 to P102, P120, P122 to P124		0		0.3 V_{DD}	V
	V_{IL2}	P00 to P07, P20, P22, P24 to P27, P32 to P35, P70, P72, P74, P121, $\overline{\text{RESET}}$		0		0.2 V_{DD}	V
High-level output voltage	V_{OH1}	P00 to P07, P20 to P24, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P100 to P102, P120 to P124	4.5 V ≤ V_{DD} ≤ 5.5 V, I_{OH} = –1 mA	V_{DD} – 1.0			V
			3.5 V ≤ V_{DD} < 4.5 V, I_{OH} = –100 μA	V_{DD} – 0.5			V
	V_{OH2}	EO0, EO1	V_{DD} = 4.5 to 5.5 V, I_{OH} = –3 mA	V_{DD} – 1.0			V
Low-level output voltage	V_{OH1}	P00 to P07, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P100 to P102, P120 to P124, P130 to P137	4.5 V ≤ V_{DD} ≤ 5.5 V, I_{OL} = 1 mA			1.0	V
			3.5 V ≤ V_{DD} < 4.5 V, I_{OL} = 100 μA			0.5	V
	V_{OL2}	EO0, EO1	V_{DD} = 4.5 to 5.5 V, I_{OL} = 3 mA			1.0	V
High-level input leakage current	I_{LIH}	P00 to P07, P10 to P17, P20 to P24, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P100 to P102, P120 to P124, $\overline{\text{RESET}}$	V_I = V_{DD}			3	μA

Remark Unless otherwise specified, the characteristics of alternate-function pins are the same as those of port pins.

DC Characteristics (T_A = –40 to +85°C, V_{DD} = 3.5 to 5.5 V) (2/3)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Low-level input current leakage	I_{LIL}	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P100 to P102, P120 to P124, $\overline{\text{RESET}}$	V_I = 0 V			–3	μA
Output off current leakage	I_{LOH1}	P130 to P137	V_O = 15 V			–3	μA
	I_{LOL1}	P130 to P137	V_O = 0 V			3	μA
	I_{LOH2}	P25 to P27 (at N-ch open-drain I/O)	V_O = V_{DD}			–3	μA
	I_{LOL2}	P25 to P27 (at N-ch open-drain I/O)	V_O = 0 V			3	μA
	I_{LOH3}	EO0, EO1	V_O = V_{DD}			–3	μA
	I_{LOL3}	EO0, EO1	V_O = 0 V			3	μA

DC Characteristics (T_A = –40 to +85°C, V_{DD} = 3.5 to 5.5 V) (3/3)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Supply current[Note] (Mask ROM versions)	I_{DD1}	When CPU is operating and PLL is stopped. Sine wave input to X1 pin $V_I = V_{DD}$	fx = 4.5 MHz (μPD178076, 178078)		2.5	15	mA
	I_{DD2}		fx = 6.3 MHz (μPD178076, 178078, 178096A, 178098A)		4.0	20	mA
	I_{DD3}	In HALT mode with PLL stopped. Sine wave input to X1 pin $V_I = V_{DD}$	fx = 4.5 MHz (μPD178076, 178078)		0.2	0.8	mA
	I_{DD4}		fx = 6.3 MHz (μPD178076, 178078, 178096A, 178098A)		0.3	1.0	mA
Supply current[Note] (μPD178F098)	I_{DD1}	When CPU is operating and PLL is stopped. Sine wave input to X1 pin $V_I = V_{DD}$	fx = 4.5 MHz		5.0	18	mA
	I_{DD2}		fx = 6.3 MHz		7.0	20	mA
	I_{DD3}	In HALT mode with PLL stopped. Sine wave input to X1 pin $V_I = V_{DD}$	fx = 4.5 MHz		0.3	0.8	mA
	I_{DD4}		fx = 6.3 MHz		0.4	1.0	mA
Data retention voltage	V_{DDR1}	When crystal resonator is oscillating		3.5		5.5	V
	V_{DDR2}	When crystal oscillation is stopped	Power-failure detection function	2.2			V
	V_{DDR3}		Data memory held	2.0			V
Data retention current	I_{DDR1}	When crystal oscillation is stopped	T_A = 25°C, V_{DD} = 5 V		2.0	4.0	μA
	I_{DDR2}		T_A = –40 to +85°C, V_{DD} = 3.5 to 5.5 V		2.0	20	μA

Note Excluding AV_{DD} current and V_{DD}PLL current.

Remarks 1. fx: System clock oscillation frequency

2. Unless otherwise specified, the characteristics of alternate-function pins are the same as those of port pins.

Reference Characteristics (T_A = –40 to +85°C, V_{DD} = 4.5 to 5.5 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Supply current	I_{DD5}	When CPU and PLL are operating. Sine wave input to VCOH pin f_{IN} = 160 MHz, V_{IN} = 0.15 V_{P-P}	Mask ROM versions		5		mA
			μPD178F098		8		mA

AC Characteristics

(1) Basic operation (T_A = –40 to +85°C, V_{DD} = 3.5 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Cycle time (minimum instruction execution time)	T_{CY}	f_X = 6.3 MHz	0.32		5.08	μs
		f_X = 4.5 MHz[Note 1]	0.44		7.11	μs
TI00, TI01 input high-/low-level width	t_{TIH0}, t_{TIL0}		4/fsam[Note 2]			s
TI50, TI51 input frequency	f_{TI5}				2	MHz
TI50, TI51 input high-/low-level width	t_{TIH5}, t_{TIL5}		200			ns
Interrupt input high-/low-level width	t_{INTH}, t_{INTL}	INTP0 to INTP7	1			μs
$\overline{\text{RESET}}$ pin low-level width	t_{RSL}		10			μs

Notes 1. When the IEBus controller of the μPD178096A, 178098A, and 178F098 is used, the 4.5 MHz crystal resonator cannot be used. Use the 6.3 MHz crystal resonator.

2. f_{sam} = f_X/2, f_X/4, f_X/64 selectable by bits 0 and 1 (PRM00 and PRM01) of prescaler mode register 0 (PRM0). However, f_{sam} = f_X/8 when the valid edge of TI00 is selected as the count clock.

(2) Serial interface (T_A = –40 to +85°C, V_{DD} = 3.5 to 5.5 V)

(a) Serial interface SIO0

(i) 3-wire serial I/O mode ($\overline{SCK0}$ ... internal clock output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK0}$ cycle time	t_{KCY1}	V_{DD} = 4.5 to 5.5 V	800			ns
		V_{DD} = 3.5 to 5.5 V	1600			ns
$\overline{SCK0}$ high-/low-level width	t_{KH1}, t_{KL1}	V_{DD} = 4.5 to 5.5 V	$t_{KCY1}/2 - 50$			ns
		V_{DD} = 3.5 to 5.5 V	$t_{KCY1}/2 - 100$			ns
SI0 setup time (to $\overline{SCK0}\uparrow$)	t_{SIK1}	V_{DD} = 4.5 to 5.5 V	100			ns
		V_{DD} = 3.5 to 5.5 V	150			ns
SI0 hold time (from $\overline{SCK0}\uparrow$)	t_{KSI1}		400			ns
SO0 output delay time from $\overline{SCK0}\downarrow$	t_{KSO1}	C = 100 pF **Note**			300	ns

Note C is the load capacitance of the $\overline{SCK0}$ and SO0 output lines.

(ii) 3-wire serial I/O mode ($\overline{SCK0}$ ... external clock input)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK0}$ cycle time	t_{KCY2}	V_{DD} = 4.5 to 5.5 V	800			ns
		V_{DD} = 3.5 to 5.5 V	1600			ns
$\overline{SCK0}$ high-/low-level width	t_{KH2}, t_{KL2}	V_{DD} = 4.5 to 5.5 V	400			ns
		V_{DD} = 3.5 to 5.5 V	800			ns
SI0 setup time (to $\overline{SCK0}\uparrow$)	t_{SIK2}		100			ns
SI0 hold time (from $\overline{SCK0}\uparrow$)	t_{KSI2}		400			ns
SO0 output delay time from $\overline{SCK0}\downarrow$	t_{KSO2}	C = 100 pF **Note**			300	ns
$\overline{SCK0}$ rise, fall time	t_{R2}, t_{F2}				1000	ns

Note C is the load capacitance of the SO0 output line.

(iii) SBI mode ($\overline{SCK0}$ ... internal clock output)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
$\overline{SCK0}$ cycle time	t_{KCY3}	V_{DD} = 4.5 to 5.5 V		800			ns
		V_{DD} = 3.5 to 5.5 V		3200			ns
$\overline{SCK0}$ high-/low-level width	t_{KH3}, t_{KL3}	V_{DD} = 4.5 to 5.5 V		$t_{KCY3}/2 - 50$			ns
		V_{DD} = 3.5 to 5.5 V		$t_{KCY3}/2 - 150$			ns
SB0, SB1 setup time (to $\overline{SCK0}\uparrow$)	t_{SIK3}	V_{DD} = 4.5 to 5.5 V		100			ns
		V_{DD} = 3.5 to 5.5 V		300			ns
SB0, SB1 hold time (from $\overline{SCK0}\uparrow$)	t_{KSI3}			$t_{KCY3}/2$			ns
SB0, SB1 output delay time from $\overline{SCK0}\downarrow$	t_{KSO3}	R = 1 kΩ C = 100 pF [Note]	V_{DD} = 4.5 to 5.5 V	0		250	ns
			V_{DD} = 3.5 to 5.5 V	0		1000	ns
SB0, SB1$\downarrow$ from $\overline{SCK0}\uparrow$	t_{KSB}			t_{KCY3}			ns
$\overline{SCK0}\downarrow$ from SB0, SB1$\downarrow$	t_{SBK}			t_{KCY3}			ns
SB0, SB1 high-level width	t_{SBH}			t_{KCY3}			ns
SB0, SB1 low-level width	t_{SBL}			t_{KCY3}			ns

Note R and C are the load resistance and load capacitance of the $\overline{SCK0}$, SB0, and SB1 output lines.

(iv) SBI mode ($\overline{SCK0}$ ... external clock input)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
$\overline{SCK0}$ cycle time	t_{KCY4}	V_{DD} = 4.5 to 5.5 V		800			ns
		V_{DD} = 3.5 to 5.5 V		3200			ns
$\overline{SCK0}$ high-/low-level width	t_{KH4}, t_{KL4}	V_{DD} = 4.5 to 5.5 V		400			ns
		V_{DD} = 3.5 to 5.5 V		1600			ns
SB0, SB1 setup time (to $\overline{SCK0}\uparrow$)	t_{SIK4}	V_{DD} = 4.5 to 5.5 V		100			ns
		V_{DD} = 3.5 to 5.5 V		300			ns
SB0, SB1 hold time (from $\overline{SCK0}\uparrow$)	t_{KSI4}			$t_{KCY4}/2$			ns
SB0, SB1 output delay time from $\overline{SCK0}\downarrow$	t_{KSO4}	R = 1 kΩ C = 100 pF [Note]	V_{DD} = 4.5 to 5.5 V	0		250	ns
			V_{DD} = 3.5 to 5.5 V	0		1000	ns
SB0, SB1$\downarrow$ from $\overline{SCK0}\uparrow$	t_{KSB}			t_{KCY4}			ns
$\overline{SCK0}\downarrow$ from SB0, SB1$\downarrow$	t_{SBK}			t_{KCY4}			ns
SB0, SB1 high-level width	t_{SBH}			t_{KCY4}			ns
SB0, SB1 low-level width	t_{SBL}			t_{KCY4}			ns
$\overline{SCK0}$ rise, fall time	t_{R4}, t_{F4}					1000	ns

Note R and C are the load resistance and load capacitance of the SB0 and SB1 output lines.

(v) 2-wire serial I/O mode ($\overline{SCK0}$ ... internal clock output)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
$\overline{SCK0}$ cycle time	t_{KCY5}	R = 1 kΩ		1600			ns
$\overline{SCK0}$ high-level width	t_{KH5}	C = 100 pF Note		$t_{KCY5}/2 - 160$			ns
$\overline{SCK0}$ low-level width	t_{KL5}		V_{DD} = 4.5 to 5.5 V	$t_{KCY5}/2 - 50$			ns
			V_{DD} = 3.5 to 5.5 V	$t_{KCY5}/2 - 100$			ns
SB0, SB1 setup time (to $\overline{SCK0}\uparrow$)	t_{SIK5}		V_{DD} = 4.5 to 5.5 V	300			ns
			V_{DD} = 3.5 to 5.5 V	350			ns
SB0, SB1 hold time (from $\overline{SCK0}\uparrow$)	t_{KSI5}			600			ns
SB0, SB1 output delay time from $\overline{SCK0}\downarrow$	t_{KSO5}			0		300	ns

Note R and C are the load resistance and load capacitance of the $\overline{SCK0}$, SB0, and SB1 output lines.

(vi) 2-wire serial I/O mode ($\overline{SCK0}$ ... external clock input)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
$\overline{SCK0}$ cycle time	t_{KCY6}			1600			ns
$\overline{SCK0}$ high-level width	t_{KH6}			650			ns
$\overline{SCK0}$ low-level width	t_{KL6}			800			ns
SB0, SB1 setup time (to $\overline{SCK0}\uparrow$)	t_{SIK6}			100			ns
SB0, SB1 hold time (from $\overline{SCK0}\uparrow$)	t_{KSI6}			$t_{KCY6}/2$			ns
SB0, SB1 output delay time from $\overline{SCK0}\downarrow$	t_{KSO6}	R = 1 kΩ	V_{DD} = 4.5 to 5.5 V	0		300	ns
		C = 100 pF Note	V_{DD} = 3.5 to 5.5 V	0		500	ns
$\overline{SCK0}$ rise, fall time	t_{R6}, t_{F6}					1000	ns

Note R and C are the load resistance and load capacitance of the SB0 and SB1 output lines.

(vii) I²C bus mode (SCL ... internal clock output)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
SCL cycle time	t_{KCY7}	R = 1 kΩ		10			μs
SCL high-level width	t_{KH7}	C = 100 pF Note		t_{KCY7} – 160			ns
SCL low-level width	t_{KL7}			t_{KCY7} – 50			ns
SDA0, SDA1 setup time (to SCL↑)	t_{SIK7}			200			ns
SDA0, SDA1 hold time (from SCL↓)	t_{KSI7}			0			ns
SDA0, SDA1 output delay time (from SCL↓)	t_{KSO7}		V_{DD} = 4.5 to 5.5 V	0		300	ns
			V_{DD} = 3.5 to 5.5 V	0		500	ns
SDA0, SDA1↓ from SCL↑ or SDA0, SDA1↑ from SCL↑	t_{KSB}			200			ns
SCL↓ from SDA0, SDA1↓	t_{SBK}			400			ns
SDA0, SDA1 high-level width	t_{SBH}			500			ns

Note R and C are the load resistance and load capacitance of SCL, SDA0 and SDA1 output line.

(viii) I²C bus mode (SCL ... external clock input)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
SCL cycle time	t_{KCY8}			1000			ns
SCL high-/low-level width	t_{KH8}, t_{KL8}			400			ns
SDA0, SDA1 setup time (to SCL↑)	t_{SIK8}			200			ns
SDA0, SDA1 hold time (from SCL↓)	t_{KSI8}			0			ns
SDA0, SDA1 output delay time from SCL↓	t_{KSO8}	R = 1 kΩ	V_{DD} = 4.5 to 5.5 V	0		300	ns
		C = 100 pF Note	V_{DD} = 3.5 to 5.5 V	0		500	ns
SDA0, SDA1↓ from SCL↑ or SDA0, SDA1↑ from SCL↑	t_{KSB}			200			ns
SCL↓ from SDA0, SDA1↓	t_{SBK}			400			ns
SDA0, SDA1 high-level width	t_{SBH}			500			ns
SCL rise, fall time	t_{R8}, t_{F8}					1000	ns

Note R and C are the load resistance and load capacitance of the SDA0 and SDA1 output lines.

(b) Serial interface SIO1

(i) 3-wire serial I/O mode ($\overline{SCK1}$ ... internal clock output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK1}$ cycle time	t_{KCY9}		800			ns
$\overline{SCK1}$ high/low-level width	t_{KH9}, t_{KL9}		$t_{KCY9}/2 - 50$			ns
SI1 setup time (to $\overline{SCK1}\uparrow$)	t_{SIK9}		100			ns
SI1 hold time (from $\overline{SCK1}\uparrow$)	t_{KSI9}		400			ns
SO1 output delay time (from $\overline{SCK1}\downarrow$)	t_{KSO9}	C = 100 pF **Note**			300	ns

Note C is the load capacitance of the $\overline{SCK1}$ and SO1 output lines.

(ii) 3-wire serial I/O mode ($\overline{SCK1}$ ... external clock input)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK1}$ cycle time	t_{KCY10}		800			ns
$\overline{SCK1}$ high/low-level width	t_{KH10}, t_{KL10}		400			ns
SI1 setup time (to $\overline{SCK1}\uparrow$)	t_{SIK10}		100			ns
SI1 hold time (from $\overline{SCK1}\uparrow$)	t_{KSI10}		400			ns
SO1 output delay time (from $\overline{SCK1}\downarrow$)	t_{KSO10}	C = 100 pF **Note**			300	ns
$\overline{SCK1}$ rise, fall time	t_{R10}, t_{F10}				1000	ns

Note C is the load capacitance of the SO1 output line.

(iii) 3-wire serial I/O mode with automatic transmit/receive function ($\overline{SCK1}$ ... internal clock output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK1}$ cycle time	t_{KCY11}		800			ns
$\overline{SCK1}$ high/low-level width	t_{KH11}, t_{KL11}		$t_{KCY11}/2 - 50$			ns
SI1 setup time (to $\overline{SCK1}\uparrow$)	t_{SIK11}		100			ns
SI1 hold time (from $\overline{SCK1}\uparrow$)	t_{KSI11}		400			ns
SO1 output delay time (from $\overline{SCK1}\downarrow$)	t_{KSO11}	C = 100 pF Note			300	ns
STB$\uparrow$ from $\overline{SCK1}\uparrow$	t_{SBD}		$t_{KCY11}/2 - 100$		$t_{KCY11}/2 + 100$	ns
Strobe signal high-level width	t_{SBW}		$t_{KCY11}/2 - 30$		$t_{KCY11}/2 + 30$	ns
Busy signal setup time (to busy signal detection timing)	t_{BYS}		100			ns
Busy signal hold time (from busy signal detection timing)	t_{BYH}		100			ns
$\overline{SCK1}\downarrow$ from busy inactive	t_{SPS}		200			ns

Note C is the load capacitance of the SO1 output line.

(iv) 3-wire serial I/O mode with automatic transmit/receive function ($\overline{SCK1}$ ... external clock input)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK1}$ cycle time	t_{KCY12}		800			ns
$\overline{SCK1}$ high/low-level width	t_{KH12}, t_{KL12}		400			ns
SI1 setup time (to $\overline{SCK1}\uparrow$)	t_{SIK12}		100			ns
SI1 hold time (from $\overline{SCK1}\uparrow$)	t_{KSI12}		400			ns
SO1 output delay time (from $\overline{SCK1}\downarrow$)	t_{KSO12}	C = 100 pF Note			300	ns
$\overline{SCK1}$ rise, fall time	t_{R12}, t_{F12}				1000	ns

Note C is the load capacitance of the SO1 output line.

(c) Serial interface SIO3

(i) 3-wire serial I/O mode ($\overline{SCK3}$ ... internal clock output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK3}$ cycle time	t_{KCY13}		800			ns
$\overline{SCK3}$ high/low-level width	t_{KH13}, t_{KL13}		$t_{KCY13}/2 - 50$			ns
SI3 setup time (to $\overline{SCK3}\uparrow$)	t_{SIK13}		100			ns
SI3 hold time (from $\overline{SCK3}\uparrow$)	t_{KSI13}		400			ns
SO3 output delay time (from $\overline{SCK3}\downarrow$)	t_{KSO13}	C = 100 pF Note			300	ns

Note C is the load capacitance of the $\overline{SCK3}$ and SO3 output lines.

(ii) 3-wire serial I/O mode ($\overline{SCK3}$ ... external clock input)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
$\overline{SCK3}$ cycle time	t_{KCY14}		800			ns
$\overline{SCK3}$ high/low-level width	t_{KH14}, t_{KL14}		400			ns
SI3 setup time (to $\overline{SCK3}\uparrow$)	t_{SIK14}		100			ns
SI3 hold time (from $\overline{SCK3}\uparrow$)	t_{KSI14}		400			ns
SO3 output delay time (from $\overline{SCK3}\downarrow$)	t_{KSO14}	C = 100 pF Note			300	ns
$\overline{SCK3}$ rise, fall time	t_{R14}, t_{F14}				1000	ns

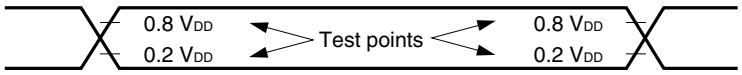
Note C is the load capacitance of the SO3 output line.

(d) Serial interface UART0 (dedicated baud rate generator output)Note

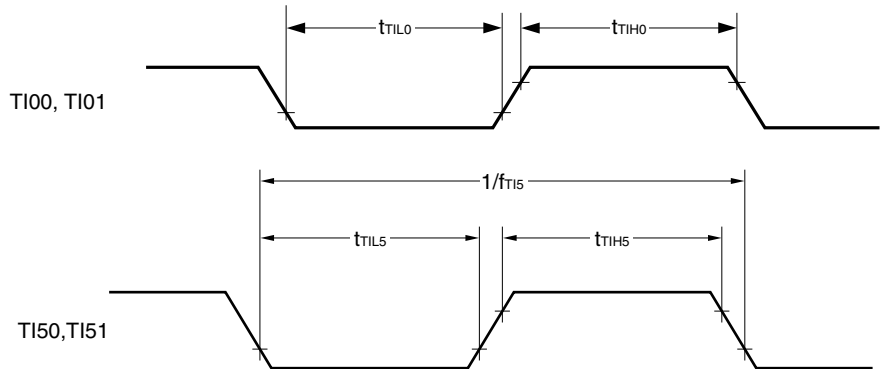
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate					38400	bps

Note μPD178076, 178078, and 178F098 only.

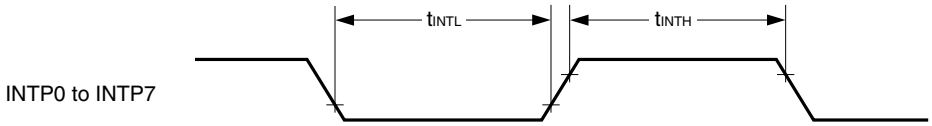
AC Timing Test Points (Excluding X1 Input)



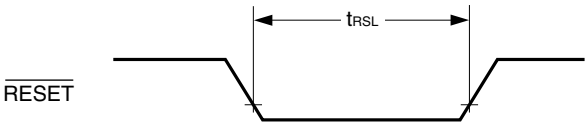
TI Timing



Interrupt Input Timing

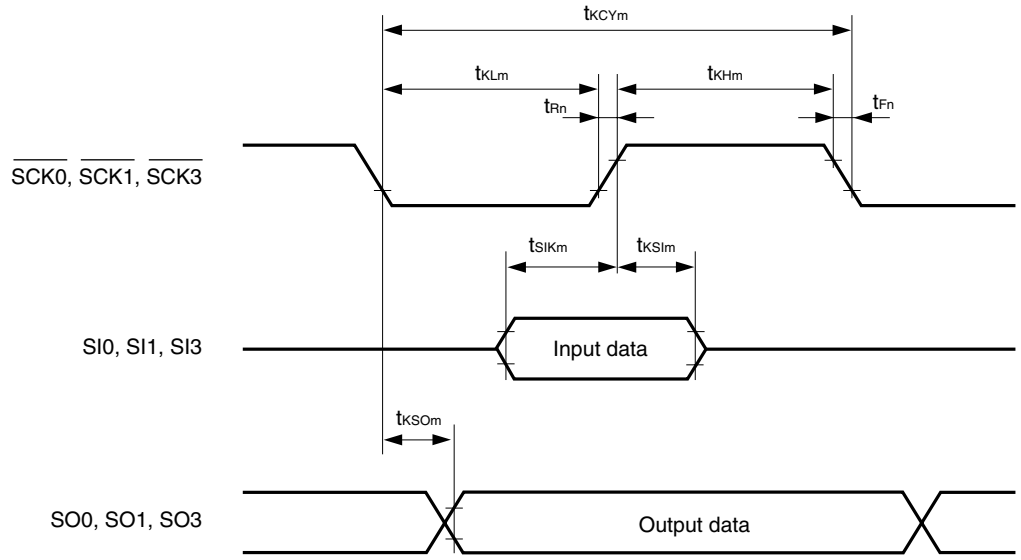


$\overline{\text{RESET}}$ Input Timing



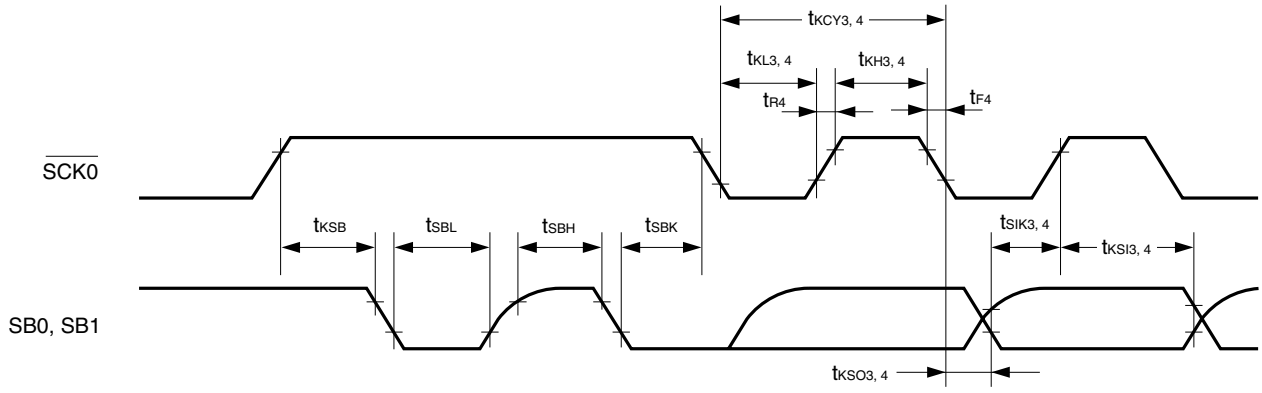
Serial Transfer Timing

3-wire serial I/O mode:

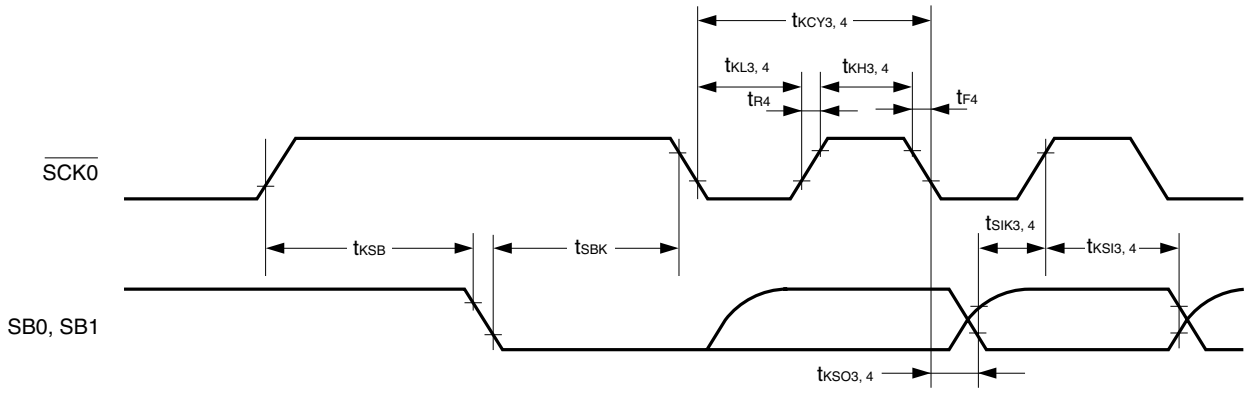


Remark m = 1, 2, 9, 10, 13, 14
n = 2, 10, 14

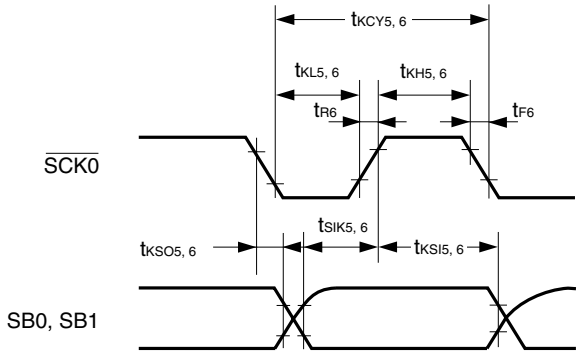
SBI mode (bus release signal transfer):



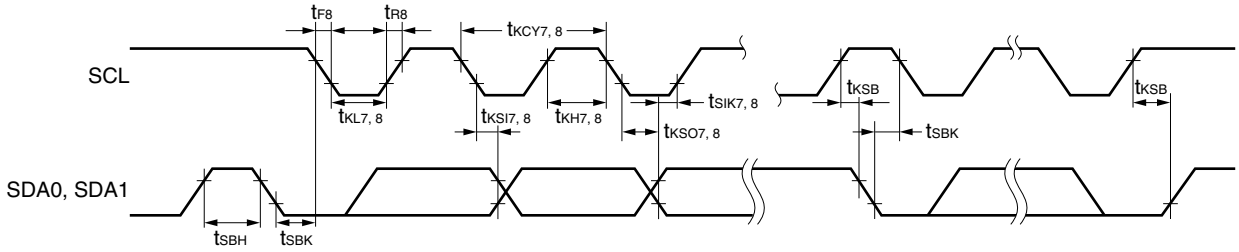
SBI mode (command signal transfer):



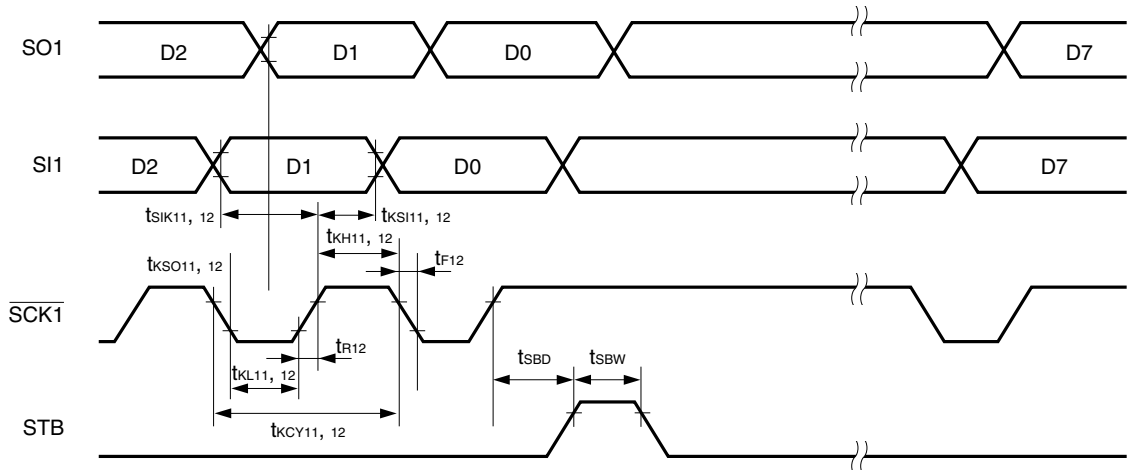
2-wire serial I/O mode:



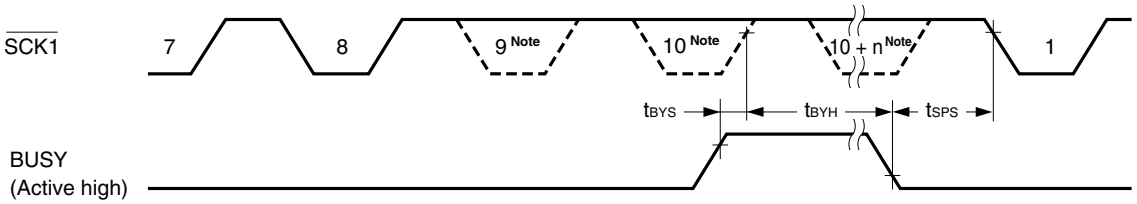
I²C bus mode:



3-wire serial I/O mode with automatic transmit/receive function:



3-wire serial I/O mode with automatic transmit/receive function (busy processing):



Note The signal is not actually driven low here; it is shown as such to indicate the timing.

IEBus Controller Characteristics[Note 1] (T_A = –40 to +85°C, V_{DD} = 3.5 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
IEBus system clock frequency	f_s	Fixed to mode 1		6.3[Note 2]		MHz

Notes 1. μPD178096A, 178098A, only 178F098 only.

2. Although the system clock frequency is 6.0 MHz in the IEBus standard, in these products, normal operation is guaranteed at 6.3 MHz.

Remark 6.0 MHz and 6.3 MHz cannot both be used as the IEBus system clock frequency.

A/D Converter Characteristics (T_A = –40 to +85°C, V_{DD} = AV_{DD} = 3.5 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution			8	8	8	bit
Total conversion error[Notes 1, 2]		V_{DD} = 4.5 to 5.5 V			±1.0	%FSR
		V_{DD} = 3.5 to 5.5 V			±1.4	%FSR
Conversion time	t_{CONV}		15.2		45.7	μs
Analog input voltage	V_{IAN}		0		V_{DD}	V

Notes 1. Excluding quantization error (±0.2%FSR)

2. This value is indicated as a ratio to the full-scale value.

PLL Characteristics (T_A = –40 to +85°C, V_{DD} = 4.5 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Operating frequency	f_{IN1}	VCOL pin, MF mode, sine wave input, V_{IN} = 0.15 V_{P-P}	0.5		3.0	MHz
	f_{IN2}	VCOL pin, HF mode, sine wave input, V_{IN} = 0.15 V_{P-P}	10		40	MHz
	f_{IN3}	VCOH pin, VHF mode, sine wave input, V_{IN} = 0.15 V_{P-P}	60		130	MHz
	f_{IN4}	VCOH pin, VHF mode, sine wave input, V_{IN} = 0.3 V_{P-P}	40		160	MHz

Remark The above values are the result of evaluation of the device by NEC Electronics. If the device is likely to be affected by noise in your application, it is recommended to use the device at an amplitude voltage higher than the above values.

IFC Characteristics (T_A = –40 to +85°C, V_{DD} = 4.5 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Operating frequency	f_{IN5}	AMIFC pin, AMIF count mode, sine wave input, V_{IN} = 0.15 V_{P-P}	0.4		0.5	MHz
	f_{IN6}	FMIFC pin, FMIF count mode, sine wave input, V_{IN} = 0.15 V_{P-P}	10		11	MHz
	f_{IN7}	FMIFC pin, AMIF count mode, sine wave input, V_{IN} = 0.15 V_{P-P}	0.4		0.5	MHz

Remark The above values are the result of evaluation of the device by NEC Electronics. If the device is likely to be affected by noise in your application, it is recommended to use the device at an amplitude voltage higher than the above values.

Flash Memory Programming Characteristics (T_A = 10 to 40°C, V_{DD} = 3.5 to 5.5 V)

(1) Write/erase characteristics

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Write current (V_{DD} pin)Note	I_{DDW}	When $V_{PP} = V_{PP1}$, f_X = 6.3 MHz			23	mA
Write current (V_{PP} pin)Note	I_{PPW}	When $V_{PP} = V_{PP1}$, f_X = 6.3 MHz			20	mA
Erase current (V_{DD} pin)Note	I_{DDE}	When $V_{PP} = V_{PP1}$, f_X = 6.3 MHz			23	mA
Erase current (V_{PP} pin)Note	I_{PPE}	When $V_{PP} = V_{PP1}$			100	mA
Unit erase time	t_{ER}		0.5	1	1	s
Total erase time	t_{ERA}				20	s
Number of rewrites	C_{WRT}	Erase and write are counted as one cycle			20	Times
V_{PP} power supply voltage	V_{PP0}	In normal mode	0		0.2 V_{DD}	V
	V_{PP1}	During flash memory programming	9.7	10.0	10.3	V

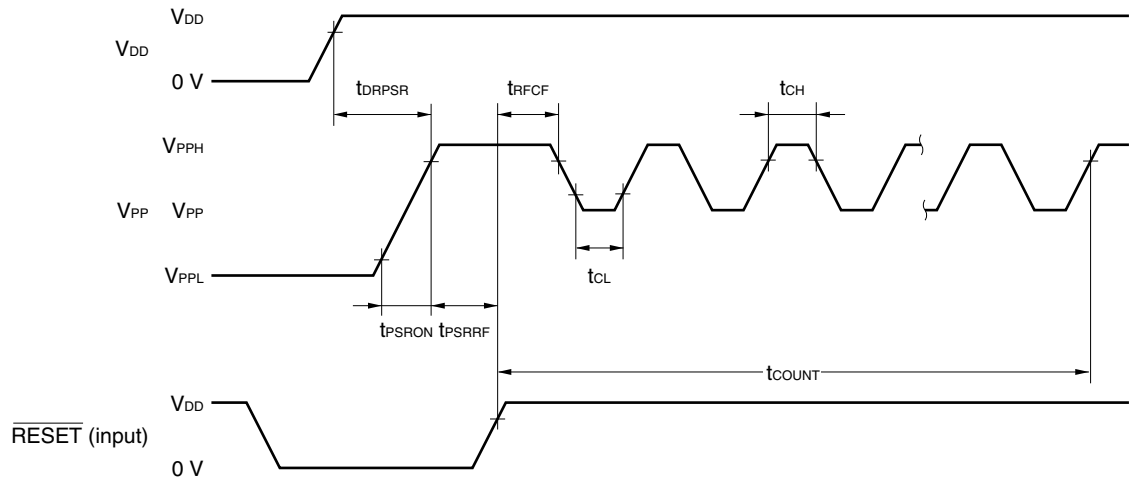
Note AV_{DD} current and port current (current flowing to internal pull-up resistors) are not included.

Remark f_X: System clock oscillation frequency

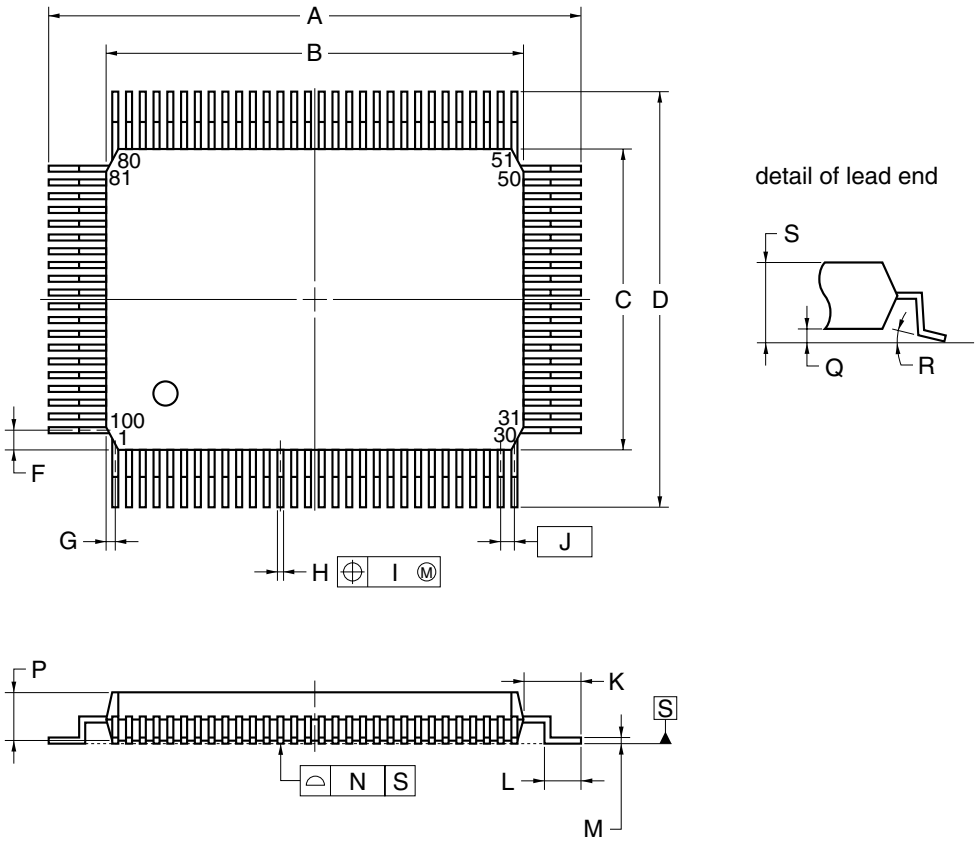
(2) Serial write operation characteristics

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
V_{PP} setup time	t_{PSRON}	V_{PP} high voltage	1.0			μs
V_{PP}↑ setup time from V_{DD}↑	t_{DRPSR}	V_{PP} high voltage	10			μs
$\overline{RESET}$↑ setup time from V_{PP}↑	t_{PSRRF}	V_{PP} high voltage	1.0			μs
V_{PP} count start time from $\overline{RESET}$↑	t_{RFCF}		1.0			μs
Count execution time	t_{COUNT}				2.0	ms
V_{PP} counter high-level width	t_{CH}		8.0			μs
V_{PP} counter low-level width	t_{CL}		8.0			μs
V_{PP} counter noise elimination width	t_{NFW}			40		ns

Flash Write Mode Setting Timing



100-PIN PLASTIC QFP (14x20)



NOTE

Each lead centerline is located within 0.15 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	23.6±0.4
B	20.0±0.2
C	14.0±0.2
D	17.6±0.4
F	0.8
G	0.6
H	0.30±0.10
I	0.15
J	0.65 (T.P.)
K	1.8±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.

P100GF-65-3BA1-4

★ CHAPTER 27 RECOMMENDED SOLDERING CONDITIONS

The μPD178078, 178098A Subseries should be soldered and mounted under the following recommended conditions. For soldering methods and conditions other than those recommended below, contact an NEC Electronics sales representative.
For technical information, see the following website.

Semiconductor Device Mount Manual (http://www.necel.com/pkg/en/mount/index.html)

Table 27-1. Soldering Conditions for Surface-Mount Type

μPD178076GF-XXX-3BA: 100-pin plastic QFP (14 × 20)
μPD178078GF-XXX-3BA: 100-pin plastic QFP (14 × 20)
μPD178096AGF-XXX-3BA: 100-pin plastic QFP (14 × 20)
μPD178098AGF-XXX-3BA: 100-pin plastic QFP (14 × 20)
μPD178F098GF-3BA: 100-pin plastic QFP (14 × 20)

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C, Time: 30 seconds max. (210°C min.), Number of times: 3 max.	IR35-00-3
VPS	Package peak temperature: 215°C, Time: 40 seconds max. (200°C min.), Number of times: 3 max.	VP15-00-3
Wave soldering	Solder bath temperature: 260°C max., Time: 10 seconds max., Number of times: 1, Preheating temperature: 120°C max., (Package surface temperature)	WS60-00-1
Partial heating	Pin temperature: 300°C max., Time: 3 seconds max. (per device side)	–

Caution Do not use two or more soldering methods together (except partial heating).

★

APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for the development of systems which employ the μPD178078, 178098A Subseries.

Figure A-1 shows a configuration example of the tools.

- **Support for PC98-NX series**

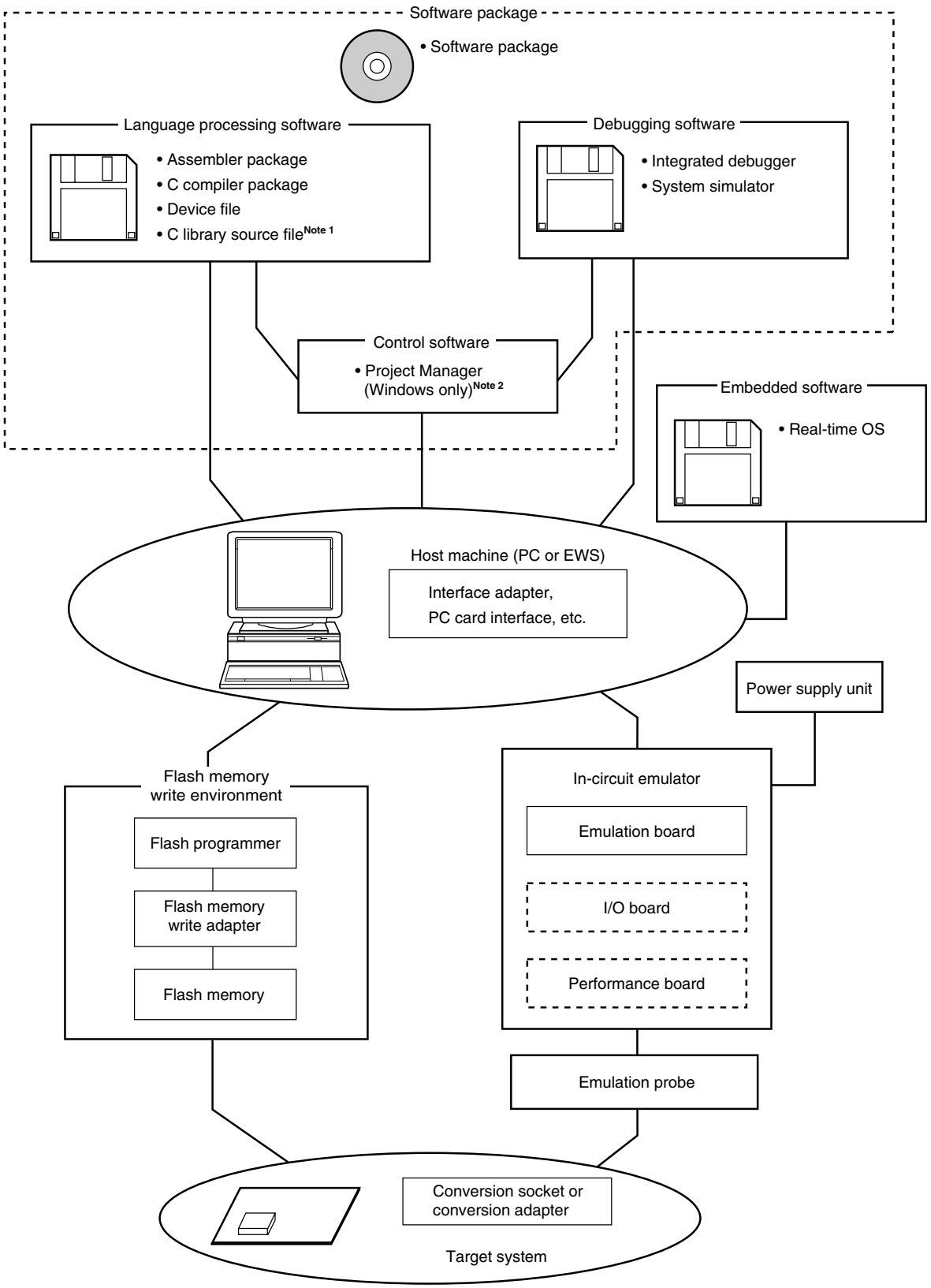
  Unless otherwise specified, products supported by IBM PC/AT™ compatible machines can be used for PC98-NX series computers. When using PC98-NX series computers, refer to the description for IBM PC/AT compatible machines.

- **Windows**

  Unless otherwise specified, "Windows" means the following OSs.

  - Windows 3.1
  - Windows 95
  - Windows 98
  - Windows 2000
  - Windows NT™ Ver. 4.0

Figure A-1. Configuration of Development Tools



Notes 1. The C library source file is not included in the software package.

2. The Project Manager is included in the assembler package.
The Project Manager is only used for Windows.

A.1 Software Package

SP78K0 Software package	This package contains various software tools for 78K/0 Series development. The following tools are included. RA78K0, CC78K0, ID78K0-NS, SM78K0, and various device files
	Part Number: μS××××SP78K0

Remark ×××× in the part number differs depending on the OS used.

μS××××SP78K0

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT and compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	

A.2 Language Processing Software

RA78K0 Assembler package	This assembler converts programs written in mnemonics into object codes executable with a microcontroller. Further, this assembler is provided with functions capable of automatically creating symbol tables and branch instruction optimization. This assembler should be used in combination with a device file (DF178098) (sold separately). **<Precaution when using RA78K0 in PC environment>** This assembler package is a DOS-based application. It can also be used in Windows, however, by using the Project Manager (included in assembler package) in Windows.
	Part Number: μS××××RA78K0
CC78K0 C compiler package	This compiler converts programs written in C language into object codes executable with a microcontroller. This compiler should be used in combination with an assembler package and device file (both sold separately). **<Precaution when using CC78K0 in PC environment>** This C compiler package is a DOS-based application. It can also be used in Windows, however, by using the Project Manager (included in assembler package) in Windows.
	Part Number: μS××××CC78K0
DF178098**Note 1** Device file	This file contains information peculiar to the device. This device file should be used in combination with tools (RA78K0, CC78K0, SM78K0, ID78K0-NS, ID78K0, and RX78K0) (sold separately). The corresponding OS and host machine differ depending on the tool used.
	Part Number: μS××××DF178098
CC78K0-L**Note 2** C library source file	This is a source file of functions configuring the object library included in the C compiler package. This file is required to match the object library included in C compiler package to the user's specifications. It does not depend on the operating environment because it is a source file.
	Part Number: μS××××CC78K0-L

Notes 1. The DF178098 can be used in common with the RA78K0, CC78K0, SM78K0, ID78K0-NS, ID78K0, and RX78K0.

2. CC78K0-L is not included in the software package (SP78K0).

Remark ××××in the part number differs depending on the host machine and OS used.

μS××××RA78K0
μS××××CC78K0

××××	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT and compatibles	Windows (Japanese version)	3.5-inch 2HD FD
BB13		Windows (English version)	
AB17		Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	
3P17	HP9000 series 700™	HP-UX™ (Rel. 10.10)	
3K17	SPARCstation™	SunOS™ (Rel. 4.1.4), Solaris™ (Rel. 2.5.1)	

μS××××DF178098
μS××××CC78K0-L

××××	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT and compatibles	Windows (Japanese version)	3.5-inch 2HD FD
BB13		Windows (English version)	
3P16	HP9000 series 700	HP-UX (Rel. 10.10)	DAT
3K13	SPARCstation	SunOS (Rel. 4.1.4), Solaris (Rel. 2.5.1)	3.5-inch 2HD FD
3K15			1/4-inch CGMT

A.3 Control Software

Project Manager	This is control software designed to enable efficient user program development in the Windows environment. All operations used in development of a user program, such as starting the editor, building, and starting the debugger, can be performed from the Project Manager. **<Caution>** The Project Manager is included in the assembler package (RA78K0). It can only be used in Windows.

A.4 Flash Memory Writing Tools

Flashpro III (Part number: FL-PR3, PG-FP3) Flashpro IV (Part number: FL-PR4, PG-FP4) Flash programmer	Flash programmer dedicated to microcontrollers with on-chip flash memory.
FA-100GF-3BA Flash memory writing adapter	Flash memory writing adapter used connected to Flashpro III/Flashpro IV. • FA-100GF-3BA: 100-pin plastic QFP (GF-3BA type)

Remark FL-PR3, FL-PR4, and FA-100GF-3BA are products of Naito Densei Machida Mfg. Co., Ltd.
Contact: +81-45-475-4191 Naito Densei Machida Mfg. Co., Ltd.

A.5 Debugging Tools (Hardware)

A.5.1 When using in-circuit emulator IE-78K0-NS, IE-78K0-NS-A

IE-78K0-NS In-circuit emulator		The in-circuit emulator serves to debug hardware and software when developing application systems using a 78K/0 Series product. It is used with an integrated debugger (ID78K0-NS). This emulator should be used in combination with a power supply unit, emulation probe, and interface adapter, which is required to connect this emulator to the host machine.
IE-78K0-NS-PA Performance board		This board is used for extending the IE-78K0-NS functions, and is used connected to the IE-78K0-NS. With the addition of this board, the addition of a coverage function, enhancement of tracer and timer functions, and other such debugging function enhancements are possible.
IE-78K0-NS-A In-circuit emulator		In-circuit emulator that combines the IE-78K0-NS and IE-78K0-NS-PA
IE-70000-MC-PS-B Power supply unit		This adapter is used for supplying power from a 100 to 240 V AC output.
IE-70000-98-IF-C Interface adapter		This adapter is required when using a PC-9800 series computer (except notebook type) as the IE-78K0-NS host machine (C bus compatible).
IE-70000-CD-IF-A PC card interface		This is PC card and interface cable required when using a notebook-type computer as the IE-78K0-NS host machine (PCMCIA socket compatible).
IE-70000-PC-IF-C Interface adapter		This adapter is required when using an IBM PC/AT compatible computer as the IE-78K0-NS host machine (ISA bus compatible).
IE-70000-PCI-IF-A Interface adapter		This adapter is required when using a PC with a PCI bus as the IE-78K0-NS host machine.
IE-178098-NS-EM1 Emulation board		This board emulates the operations of the peripheral hardware peculiar to a device. It should be used in combination with an in-circuit emulator.
NP-100GF-TQ NP-H100GF-TQ Emulation probe		This probe is used to connect the in-circuit emulator to the target system and is designed for a 100-pin plastic QFP (GF-3BA type). It should be used in combination with the TGF-100RBP.
	TGF-100RBP Conversion adapter	This conversion socket connects the NP-100GF-TQ or NP-H100GF-TQ to the target system board designed to mount a 100-pin plastic QFP (GF-3BA type).
NP-100GF Emulation Probe		This probe is for a 100-pin plastic QFP (GF-3BA type) and connects an in-circuit emulator and the target system.
	EV-9200GF-100 Conversion Socket (See Figures A-2 and A-3)	This conversion socket connects the board of a target system created to mount a 100-pin plastic QFP (GF-3BA type) and NP-100GF.

Remarks 1. NP-100GF, NP-100GF-TQ, and NP-H100GF-TQ are products of Naito Densei Machida Mfg. Co., Ltd.
Contact: +81-45-475-4191 Naito Densei Machida Mfg. Co., Ltd.

2. TGF-100RBP is a product of TOKYO ELETECH CORPORATION.
Inquiry: Daimaru Kogyo, Ltd. Phone: Tokyo +81-3-3820-7112 Electronics Dept.
Osaka +81-6-6244-6672 Electronics 2nd Dept.

3. The EV-9200GF-100 is sold in sets of 5 units.

A.5.2 When using in-circuit emulator IE-78001-R-A

IE-78001-R-A In-circuit emulator		This is an in-circuit emulator for debugging the hardware and software when an application system using the 78K/0 Series is developed. It is used with an integrated debugger (ID78K0). This emulator is used with an emulation probe and interface adapter for connecting a host machine.
IE-70000-98-IF-C Interface adapter		This adapter is necessary when a PC-9800 series PC (except notebook type) is used as the host machine for the IE-78001-R-A (C bus compatible).
IE-70000-PC-IF-C Interface adapter		This adapter is necessary when an IBM PC/AT or compatible machine is used as the host machine for the IE-78001-R-A (ISA bus compatible).
IE-178098-NS-EM1 Emulation board		This board is used with an in-circuit emulator and emulation probe conversion board to emulate device-specific peripheral hardware.
IE-78K0-R-EX1 Emulation probe conversion board		This board is necessary for using the IE-178098-NS-EM1 on the IE-78001-R-A.
EP-78064GF-R Emulation probe		This probe is for an 100-pin plastic QFP (GF-3BA type) and connects an in-circuit emulator and the target system.
	EV-9200GF-100 Conversion socket (See Figures A-2 and A-3)	This conversion socket connects the board of a target system created to mount an 100-pin plastic QFP (GF-3BA type) and EP-78064GF-R.

Remark The EV-9200GF-100 is sold in sets of five units.

A.6 Debugging Tools (Software)

SM78K0 System simulator	This is a system simulator for the 78K/0 Series. The SM78K0 is Windows-based software. It is used to perform debugging at the C source level or assembler level while simulating the operation of the target system on a host machine. Use of the SM78K0 allows the execution of application logical testing and performance testing on an independent basis from hardware development, thereby providing higher development efficiency and software quality. The SM78K0 should be used in combination with a device file (DF178098) (sold separately).
	Part Number: μS××××SM78K0
ID78K0-NS Integrated debugger (supporting in-circuit emulators IE-78K0-NS and IE-78K0-NS-A)	This debugger supports the in-circuit emulators for the 78K/0 Series. The ID78K0-NS is Windows-based software. It has improved C-compatible debugging functions and can display the results of tracing with the source program using an integrating window function that associates the source program, disassemble display, and memory display with the trace result. It should be used in combination with a device file (sold separately).
ID78K0 Integrated debugger (supporting in-circuit emulator IE-78001-R-A)	Part Number: μS××××ID78K0-NS μS××××ID78K0

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××SM78K0
μS××××ID78K0-NS
μS××××ID78K0

××××	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT and compatibles	Windows (Japanese version)	3.5-inch 2HD FD
BB13		Windows (English version)	
AB17		Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	

A.7 Embedded Software

RX78K0 Real-time OS	The RX78K0 is a real-time OS conforming to the μITRON specifications. A tool (configurator) for generating the nucleus of the RX78K0 and multiple information tables is supplied. Used in combination with an assembler package (RA78K0) and device file (DF178098) (both sold separately). **<Precaution when using RX78K0 in PC environment>** The real-time OS is a DOS-based application. It should be used at the DOS prompt when using in Windows.
	Part number: μS××××RX78013-ΔΔΔΔ

Caution When purchasing the RX78K0, fill in the purchase application form in advance and sign the user agreement.

Remark ×××× and ΔΔΔΔ in the part number differ depending on the host machine and OS used.

μS××××RX78013-ΔΔΔΔ

ΔΔΔΔ	Product Outline	Maximum Number for Use in Mass Production
001	Evaluation object	Do not use for mass-produced product.
100K	Mass-producted object	0.1 million units
001M		1 million units
010M		10 million units
S01	Source program	Source program for mass-produced object

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT and compatibles	Windows (Japanese version)	
BB13		Windows (English version)	

A.8 Upgrading Old Version of In-Circuit Emulator for 78K/0 Series to IE-78001-R-A

If you already have an old type of in-circuit emulator for the 78K/0 series (IE-78000-R or IE-78000-R-A), it can be upgraded to have equivalent functions to the IE-78001-R-A by exchanging the break board with the IE-78001-R-BK.

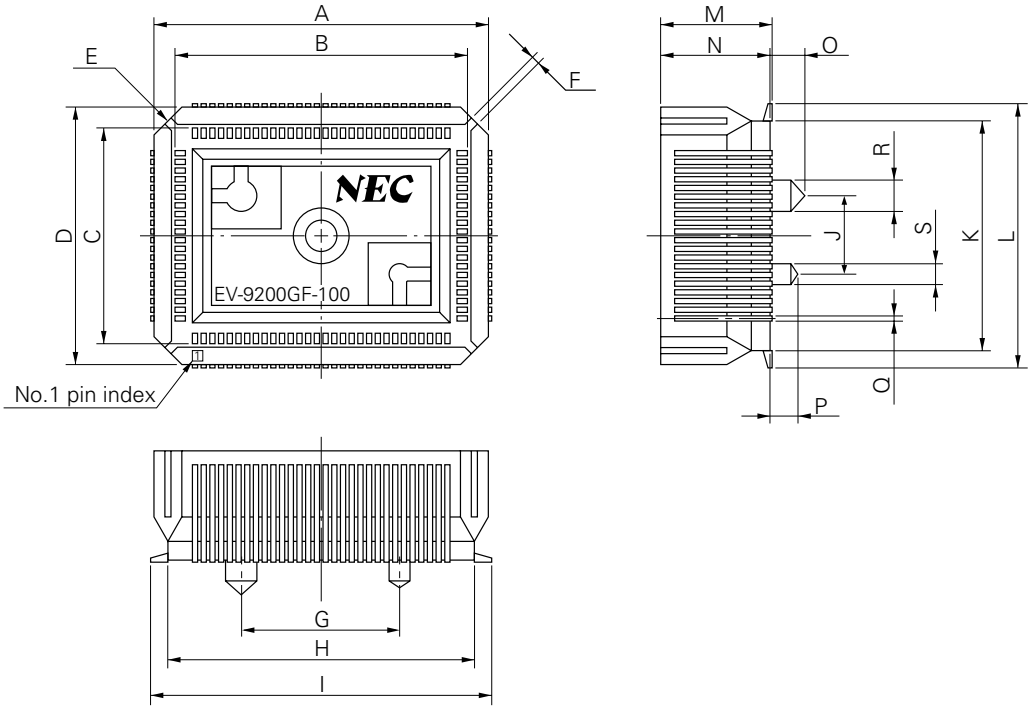
Table A-1. Upgrading Old Version of In-Circuit Emulator for 78K/0 Series to IE-78001-R-A

Your In-Circuit Emulator	Upgrading of Housing[Note]	Necessary Board
IE-78000-R	Necessary	IE-78001-R-BK
IE-78000-R-A	Not necessary	

Note To upgrade the housing, your in-circuit emulator must be brought to NEC Electronics.

A.9 Drawing for Conversion Socket (EV-9200GF-100) Package and Recommended Board Mounting Pattern

Figure A-2. EV-9200GF-100 Package Drawing (for Reference Only)

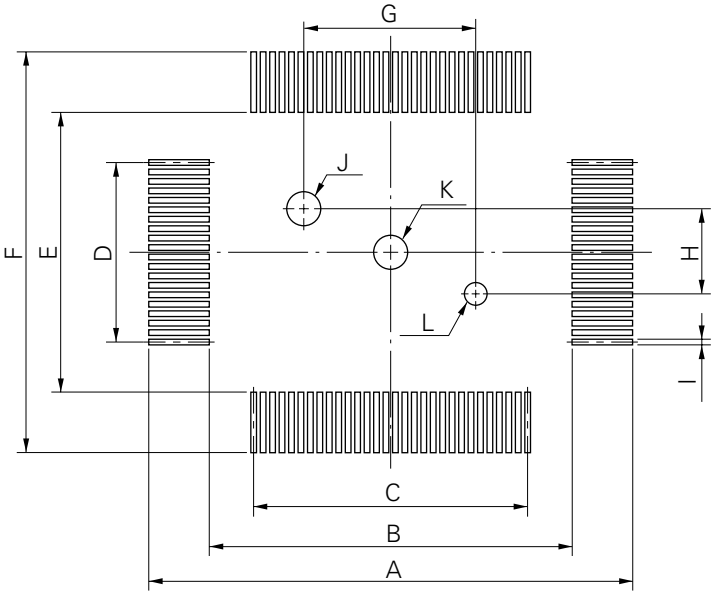


EV-9200GF-100-G0E

ITEM	MILLIMETERS	INCHES
A	24.6	0.969
B	21	0.827
C	15	0.591
D	18.6	0.732
E	4-C 2	4-C 0.079
F	0.8	0.031
G	12.0	0.472
H	22.6	0.89
I	25.3	0.996
J	6.0	0.236
K	16.6	0.654
L	19.3	076
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 A-3. EV-9200GF-100 Recommended Board Mounting Pattern (for Reference Only)

Based on EV-9200GF-100
(2) Pad drawing (in mm)



EV-9200GF-100-P1E

ITEM	MILLIMETERS	INCHES
A	26.3	1.035
B	21.6	0.85
C	0.65±0.02 × 29=18.85±0.05	$0.026^{+0.001}_{-0.002} \times 1.142 = 0.742^{+0.002}_{-0.002}$
D	0.65±0.02 × 19=12.35±0.05	$0.026^{+0.001}_{-0.002} \times 0.748 = 0.486^{+0.003}_{-0.002}$
E	15.6	0.614
F	20.3	0.799
G	12±0.05	$0.472^{+0.003}_{-0.002}$
H	6±0.05	$0.236^{+0.003}_{-0.002}$
I	0.35±0.02	$0.014^{+0.001}_{-0.001}$
J	φ2.36±0.03	$\phi 0.093^{+0.001}_{-0.002}$
K	φ2.3	φ0.091
L	φ1.57±0.03	$\phi 0.062^{+0.001}_{-0.002}$

Caution The dimensions of the mount pad for EV-9200 and that for target device (QFP) may differ in some parts. For the recommended mount pad dimensions for QFP, refer to the "Semiconductor Device Mount Manual" website (http://www.necel.com/pkg/en/mount/index.html).

★ APPENDIX B REGISTER INDEX

B.1 Register Index (Register Name)

B.2 Register Index (Symbol)

★ APPENDIX C REVISION HISTORY

The revision history of this edition is listed in the table below. “Chapter” indicates the chapter of the previous edition where the revision was made.

(1/2)

Edition	Revision	Chapter
2nd	Addition of μPD178096A and 178098A	Throughout
	Modification of μPD178F098 from under development to developed	
	Modification of **1.5 Development of 8-Bit DTS Series**	**CHAPTER 1 OUTLINE**
	Modification of pin handing in **2.2.26 V_{PP} (μPD178F098 only)**	**CHAPTER 2 PIN FUNCTIONS**
	Modification of **Table 2-1 Pin I/O Circuit Types** and **Figure 2-1 Pin I/O Circuits**	
	Addition of description of programming area in **3.1.2 Internal data memory space**	**CHAPTER 3 CPU ARCHITECTURE**
	Modification of **Figure 3-10 Data to Be Saved to Stack Memory** and **Figure 3-11 Data to Be Restored from Stack Memory**	
	Modification of **[Example]** in **3.4.4 Short direct addressing**	
	Addition of **[Illustration]** to **3.4.7 Based addressing**, **3.4.8 Based indexed addressing**, and **3.4.9 Stack addressing**	
	Addition of description of output latches after reset to **4.4 Port Function Operations**	**CHAPTER 4 PORT FUNCTIONS**
	6.2 Configuration of 16-Bit Timer/Event Counter 0 • Addition of **Cautions** to **(2) 16-bit capture/compare register 00 (CR00)** • Addition of **Table 6-3 CR01 Capture Trigger and Valid Edge of TI00 Pin (CRC02 = 1)** • Addition of **Caution** to **(3) 16-bit capture/compare register 01 (CR01)**	**CHAPTER 6 16-BIT TIMER/EVENT COUNTER 0**
	Addition of **Caution** to **Figure 6-5 Format of Prescaler Mode Register 0 (PRM0)**	
	6.4.5 One-shot pulse output operation • Modification of **Figure 6-26 Timing of One-Shot Pulse Output Operation with Software Trigger** • Addition of **Note** to **(2) One-shot pulse output with external trigger**	
	Addition of **6.5 Program List**	
	Addition of **(6) (c) One-shot pulse output function** to **6.6 Notes on 16-Bit Timer/Event Counter 0**	
	7.2 Configuration of 8-Bit Timer/Event Counters 50, 51 • Addition of **Note** to **(1) 8-bit timer counters 50 and 51 (TM50 and TM51)** • Addition of description of PWM mode to **(2) 8-bit compare registers 50 and 51 (CR50 and CR51)**	**CHAPTER 7 8-BIT TIMER/EVENT COUNTERS**
	Addition of **7.5 Program List**	
	Addition of **(4) Noise countermeasures and (6) Input impedance of ANI0 to ANI7 pins** to **11.5 A/D Converter Cautions**	**CHAPTER 11 A/D CONVERTER**
	Addition of **Figure 16-2 Block Diagram of Baud Rate Generator**	**CHAPTER 16 SERIAL INTERFACE UART0 (μPD178076, 178078, 178F098 ONLY)**
	Addition of **Caution** to **Figure 16-6 Permissible Error of Baud Rate Allowing for Sampling Error (k = 0)**	

(2/2)

Edition	Revision	Chapter
2nd	Addition of **Caution** about inversion of IEBus protocol and signal inside the microcontroller to **17.1.6 Transfer format of IEBus** and **17.1.8 Bit format**	**CHAPTER 17 IEBus CONTROLLER (μPD178096A, 178098A, 178F098 ONLY)**
	Modification of **Note** and **Caution** in **17.1.6 (9) Acknowledge bit**	
	Addition of description of lock setting conditions and lock release conditions to **17.1.7 (4) Locking and unlocking**	
	Addition of description of timing error detection for each period to **17.1.8 Bit format**	
	Addition of **Notes** about automatic master reprocessing to **Table 17-7 Comparison of Existing and Simple IEBus Interface Functions**	
	17.4.2 Description of internal registers • Explanation of each register thoroughly modified and Note added • Addition of figures of interrupt timing to **Figures 17-16** to **17-19** • Addition of **Figure 17-23 Example of Broadcast Communication Flag Operation** • Addition of **Table 17-9 Reset Conditions of Flags in ISR Register**	
	17.5 Interrupt Operations of IEBus Controller • Thorough modification of contents • Addition of **17.5.3 Communication error source processing list**	
	Addition of description of wait of slave unit to **17.6.2 Master reception**	
	Correction of description of drive type of EO1 pin in **19.4.1 Operation of each block of PLL frequency synthesizer**	**CHAPTER 19 PLL FREQUENCY SYNTHESIZER**
	Correction of **Note** in **Table 22-1 Hardware Status After Reset**	**CHAPTER 22 RESET FUNCTION**
	Thorough modification of descriptions of flash memory programming as **23.3 Flash Memory Features**	**CHAPTER 23 μPD178F098**
	Addition of chapters	**CHAPTER 25 ELECTRICAL SPECIFICATIONS**
		CHAPTER 26 PACKAGE DRAWING
		CHAPTER 27 RECOMMENDED SOLDERING CONDITIONS
	Thorough modification of descriptions of development tools Deletion of **EMBEDDED SOFTWARE**	**APPENDIX A DEVELOPMENT TOOLS**
	Addition of chapters	**APPENDIX B REGISTER INDEX**
		APPENDIX C REVISION HISTORY