Preliminary User's Manual

μPD178024, 178124 Subseries

8-bit Single-chip Microcontroller

μPD178022
μPD178023
μPD178024
μPD178122
μPD178123
μPD178124
μPD178F124

Document No. U13915EJ1V0UM00 (1st edition)
Date Published February 1999 N CP(K)

[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.

EEPROM is a trademark of NEC Corporation.

Windows and WindowsNT 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.

SunOS is a trademark of Sun Microsystems, Inc.

Ethernet is a trademark of Xerox Corporation.

NEWS and NEWS-OS are trademarks of Sony Corporation.

OSF/Motif is a trademark of Open Software Foundation, Inc.

TRON is an abbreviation of The Realtime Operating system Nucleus.

ITRON is an abbreviation of Industrial TRON.

The export of these products from Japan is regulated by the Japanese government. The export of some or all of these products may be prohibited without governmental license. To export or re-export some or all of these products from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

License not needed	: µPD178F124GF-3B9, 178F124GC-8BT
The customer must judge the need for license	: µPD178022GF-xxx-3B9, 178022GC-xxx-8BT, 178023GF-xxx-3B9, µPD178023GC-xxx-8BT, 178024GF-xxx-3B9, 178024GC-xxx-8BT, µPD178122GF-xxx-3B9, 178122GC-xxx-8BT, 178123GF-xxx-3B9, µPD178123GC-xxx-8BT, 178124GF-xxx-3B9, 178124GC-xxx-8BT

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

This product is manufactured and sold under license from CP8 Transac company concerning the patent for microcontrollers with on-chip EEPROM. This product cannot be used for an IC card (SMART CARD).

Purchase of NEC I^2C components conveys a license under the Philips I^2C Patent Rights to use these components in an I^2C system, provided that the system conforms to the I^2C Standard Specification as defined by Philips.

The information contained in this document is being issued in advance of the production cycle for the device. The parameters for the device may change before final production or NEC Corporation, at its own discretion, may withdraw the device prior to its production.

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.

NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.

While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.

NEC devices are classified into the following three quality grades:
"Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated "quality assurance program" for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device 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: Aircrafts, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.

The quality grade of NEC devices is "Standard" unless otherwise specified in NEC's Data Sheets or Data Books. If customers intend to use NEC devices for applications other than those specified for Standard quality grade, they should contact an NEC sales representative in advance.

Anti-radioactive design is not implemented in this product.

M5 96.5

Regional Information

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

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

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

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

NEC Electronics (Germany) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.l.
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH
Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.
Velizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 01-504-2787
Fax: 01-504-2860

NEC Electronics (Germany) GmbH
Scandinavia Office
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

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

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

NEC Electronics Singapore Pte. Ltd.
United Square, Singapore 1130
Tel: 65-253-8311
Fax: 65-250-3583

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

NEC do Brasil S.A.
Electron Devices Division
Rodovia Presidente Dutra, Km 214
07210-902-Guarulhos-SP Brasil
Tel: 55-11-6465-6810
Fax: 55-11-6465-6829

J98. 11

PREFACE

Readers

This manual has been prepared for user engineers who want to understand the functions of the μPD178024 and 178124 subseries and design and develop its application systems and programs.

Purpose

This manual is intended for users to understand the functions described in the Organization below.

Organization

The μPD178024 and 178124 subseries manual is separated into two parts: this manual and the instruction edition (common to the 78K/0 series).

μPD178024, 178124 subseries User's Manual	78K/0 series User's Manual Instruction
• Pin functions • Internal block functions • Interrupt • Other on-chip peripheral functions	• CPU functions • Instruction set • Explanation of each instruction

How to Read This Manual

Before reading this manual, you should have general knowledge of electric and logic circuits and microcomputers.

○ When you want to understand the functions in general:
→ Read this manual in the order of the contents.

○ To know the μPD178024, 178124 subseries instruction function in detail:
→ Refer to the **78K/0 series User's Manual -Instructions (U12326E)**

○ How to interpret the register format:
→ For the circled bit number, the bit name is defined as a reserved word in DF178124 and RA78K/0, and in CC78K/0, already defined in the header file named sfrbit.h.

○ To know the electrical specifications of the μPD178024, 178124 subseries:
→ Refer to separately available Data Sheet.

Conventions

Data representation weight	:	High digits on the left and low digits on the right
Active low representations	:	$\overline{\text{xxx}}$ (line over the pin and signal names)
Note	:	Description of **Note** in the text.
Caution	:	Information requiring particular attention
Remark	:	Additional explanatory material
Numeral representations	:	Binary ... ×××× or ××××B Decimal ... ×××× Hexadecimal ... ××××H

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

- **Related documents for μPD178024, 178124 subseries**

Document Name		Document No.	
		Japanese	English
μPD178022, 178023, 178024, 178122, 178123, 178124 Preliminary Product Information		Planned	Planned
μPD178F124 Preliminary Product Information		Planned	Planned
μPD178024, 178124 Subseries Preliminary User's Manual		U13915J	This manual
78K/0 Series User's Manual —Instruction		U12326J	U12326E
78K/0 Series Instruction Table		U10903J	—
78K/0 Series Instruction Set		U10904J	—
μPD178024, 178124 Subseries Special Function Register Table		Planned	—
78K/0 Series Application Note	Basics (I)	U12704J	U12704E

- **Development Tool Documents (User's Manuals)**

Document Name		Document Number	
		Japanese	English
RA78K0 Assembler Package	Operation	U11802J	U11802E
	Assembly Language	U11801J	U11801E
	Structured Assembly Language	U11789J	U11789E
RA78K Series Structured Assembler Preprocessor		U12323J	EEU-1402
CC78K0 C Compiler	Operation	U11517J	U11517E
	Language	U11518J	U11518E
CC78K0 C Compiler Application Note	Programming Know-How	U13034J	U13034E
CC78K Series Library Source File		U12322J	–
IE-78001-R-A		Planned	Planned
IE-78K0-NS		Planned	Planned
IE-178134-NS-EM1		Planned	Planned
EP-78130		Planned	Planned
ID78K0-NS Integrated Debugger		U12900J	U12900E
SM78K0 System Simulator Windows™ Base	Reference	U10181J	U10181E
SM78K Series System Simulator	External Part User Open Interface Specifications	U10092J	U10092E
ID78K/0 Integrated Debugger EWS Base	Reference	U11151J	–
ID78K0 Integrated Debugger Windows Base	Reference	U11539J	U11539E
ID78K0 Integrated Debugger Windows Base	Guide	U11649J	U11649E

Caution The above documents are subject to change without prior notice. Be sure to use the latest version document when starting design.

- **Documents for Embedded Software (User's Manual)**

Document Name		Document No.	
		Japanese	English
78K/0 Series Real-Time OS	Fundamentals	U11537J	U11537E
	Installation	U11536J	U11536E
78K/0 Series OS MX78K0	Fundamental	U12257J	U12257E

- **Other Documents**

Document Name	Document No.	
	Japanese	English
NEC IC Package Manual (CD-ROM)	–	C13388E
Semiconductor Device Mounting Technology Manual	C10535J	C10535E
Quality Grades on NEC Semiconductor Devices	C11531J	C11531E
NEC Semiconductor Reliability/Quality Control System	C10983J	C10983E
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892J	C11892E
Semiconductor Devices Quality/Reliability Handbook	C11893J	—
Microcomputer Related Product Guide—Third Party Manufacturers	U11416J	—

Caution The above documents are subject to change without prior notice. Be sure to use the latest version document when starting design.

TABLE OF CONTENTS

LIST OF FIGURES (1/5)

LIST OF FIGURES (2/5)

LIST OF FIGURES (3/5)

LIST OF FIGURES (4/5)

LIST OF FIGURES (5/5)

LIST OF TABLES (1/2)

LIST OF TABLES (2/2)

[MEMO]

CHAPTER 1 OUTLINE

1.1 Features

- On-chip ROM and RAM

Item Part Number	Program Memory		Data Memory	
			Internal high-speed RAM	EEPROM
μPD178022	ROM	16 Kbytes	512 bytes	–
μPD178023		24 Kbytes	1024 bytes	
μPD178024		32 Kbytes		
μPD178122		16 Kbytes	512 bytes	128 bytes
μPD178123		24 Kbytes	1024 bytes	
μPD178124		32 Kbytes		
μPD178F124	Flash memory	32 Kbytes		

- Instruction set suitable for system control
  - Bit processing across entire address space
  - Multiplication/division instructions
- General-purpose I/O ports: 62 pins
- 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: 6 channels
- Serial interface: clocked 2 channels (except μPD178F124), 3 channels (μPD178F124)
  - I^{2}C bus mode[Note] : 1 channel
  - 3-wire serial I/O mode : 1 channel
  - UART mode (μPD178F124 only) : 1 channel

Note When using the I^{2}C bus mode (including when this mode is implemented by software without using the internal hardware), consult NEC when you place an order for mask.

- Timer: 4 channels
  - Basic timer (timer carry FF): 1 channel
  - 8-bit timer/event counter : 2 channels
  - Watchdog timer : 1 channel
- Buzzer output
- Vectored interrupt

Item / Part Number	Non-Maskable Interrupt[Note]	Maskable Interrupt[Note]		Software Interrupt
		External	Internal	
μPD178022, 178023, 178024	1 source	5 sources	8 sources	1 source
μPD178122, 178123, 178124			9 sources	
μPD178F124			12 sources	

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 : 0.45/0.89/1.78/3.56/7.11 μs (with 4.5-MHz crystal resonator)
- Supply voltage : V_{DD} = 4.5 to 5.5 V (with CPU, PLL operating)
  V_{DD} = 3.5 to 5.5 V (with CPU operating)
- Power-ON clear circuit

1.2 Applications

Car stereo

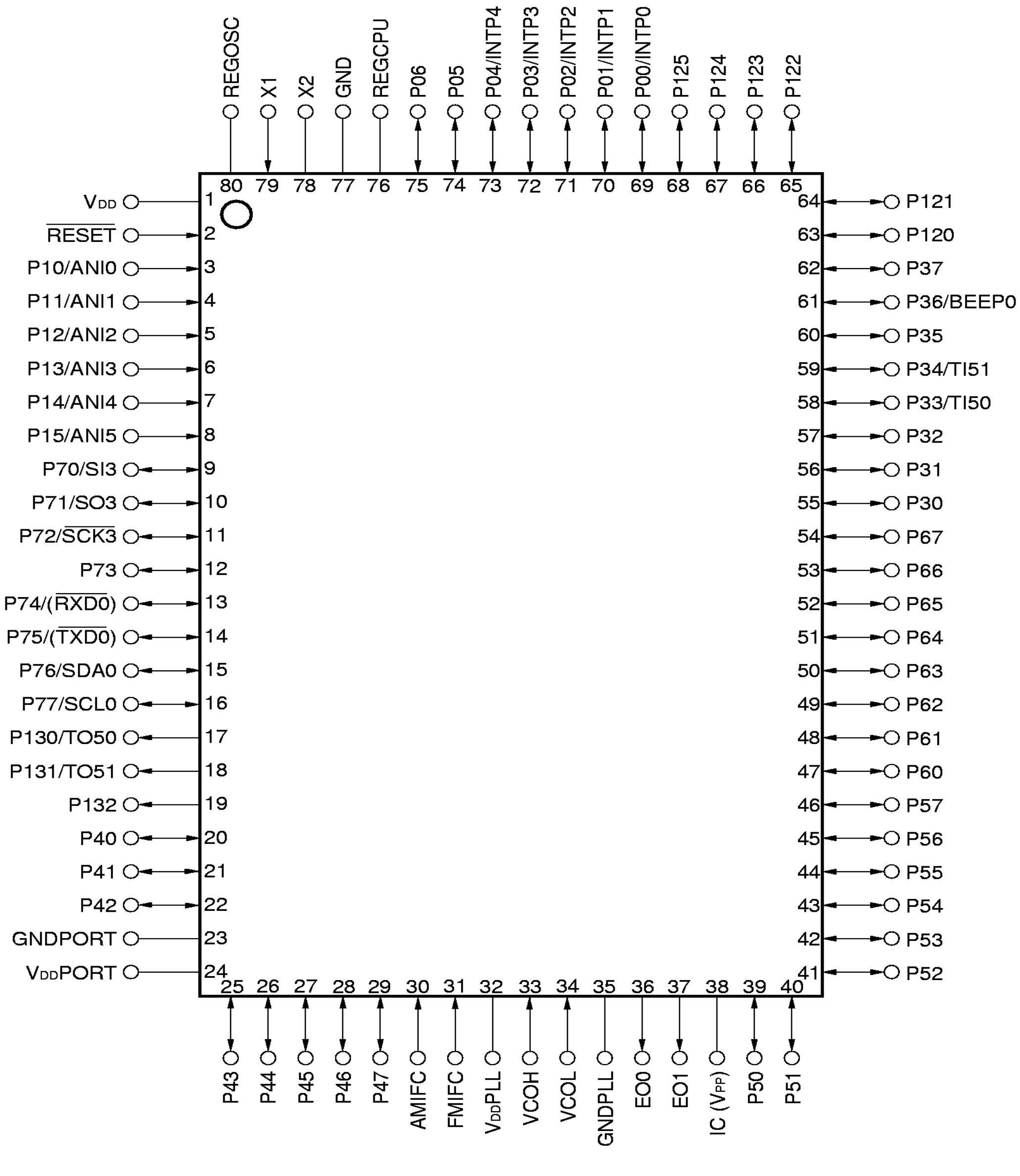
1.3 Ordering Information

Part Number	Package
μPD178022GF-×××-3B9	80-pin plastic QFP (14 × 20 mm, 0.8-mm pitch)
μPD178022GC-×××-8BT	80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)
μPD178023GF-×××-3B9	80-pin plastic QFP (14 × 20 mm, 0.8-mm pitch)
μPD178023GC-×××-8BT	80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)
μPD178024GF-×××-3B9	80-pin plastic QFP (14 × 20 mm, 0.8-mm pitch)
μPD178024GC-×××-8BT	80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)
μPD178122GF-×××-3B9	80-pin plastic QFP (14 × 20 mm, 0.8-mm pitch)
μPD178122GC-×××-8BT	80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)
μPD178123GF-×××-3B9	80-pin plastic QFP (14 × 20 mm, 0.8-mm pitch)
μPD178123GC-×××-8BT	80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)
μPD178124GF-×××-3B9	80-pin plastic QFP (14 × 20 mm, 0.8-mm pitch)
μPD178124GC-×××-8BT	80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)
μPD178F124GF-3B9	80-pin plastic QFP (14 × 20 mm, 0.8-mm pitch)
μPD178F124GC-8BT	80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)

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

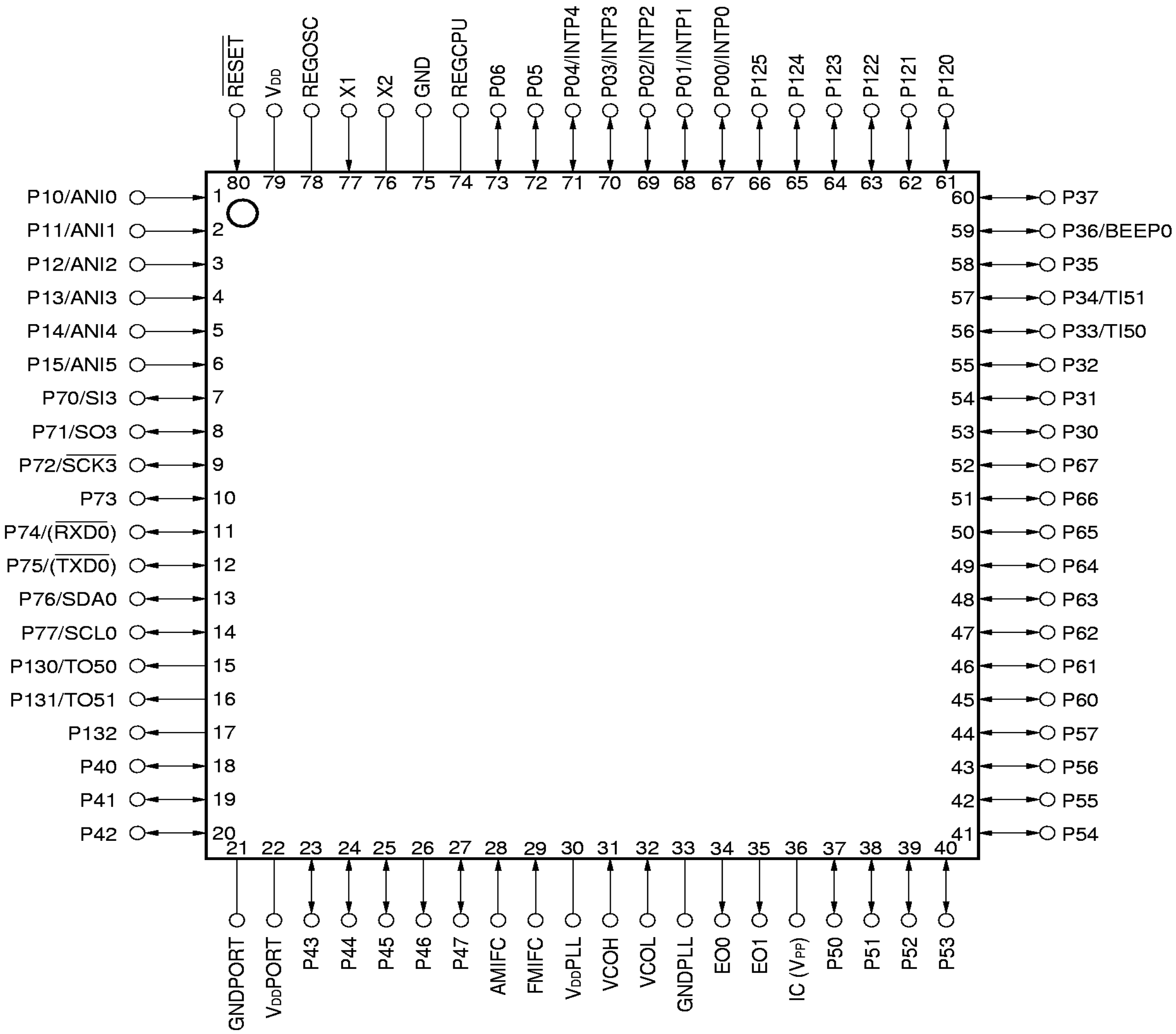
1.4 Pin Configuration (Top View)

- **80-pin plastic QFP (14 × 20 mm, 0.8-mm pitch)**
  μPD178022GF-×××-3B9, 178023GF-×××-3B9, 178024GF-×××-3B9
  μPD178122GF-×××-3B9, 178123GF-×××-3B9, 178124GF-×××-3B9
  μPD178F124GF-3B9



- **80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)**

  μPD178022GC-xxx-8BT, 178023GC-xxx-8BT, 178024GC-xxx-8BT
  μPD178122GC-xxx-8BT, 178123GC-xxx-8BT, 178124GC-xxx-8BT
  μPD178F124GC-8BT



Cautions 1. Directly connect the IC (Internally Connected) pin and V_{PP} pin to GND.

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

3. Keep the voltage at GNDPORT and GNDPLL same as that at GND.

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

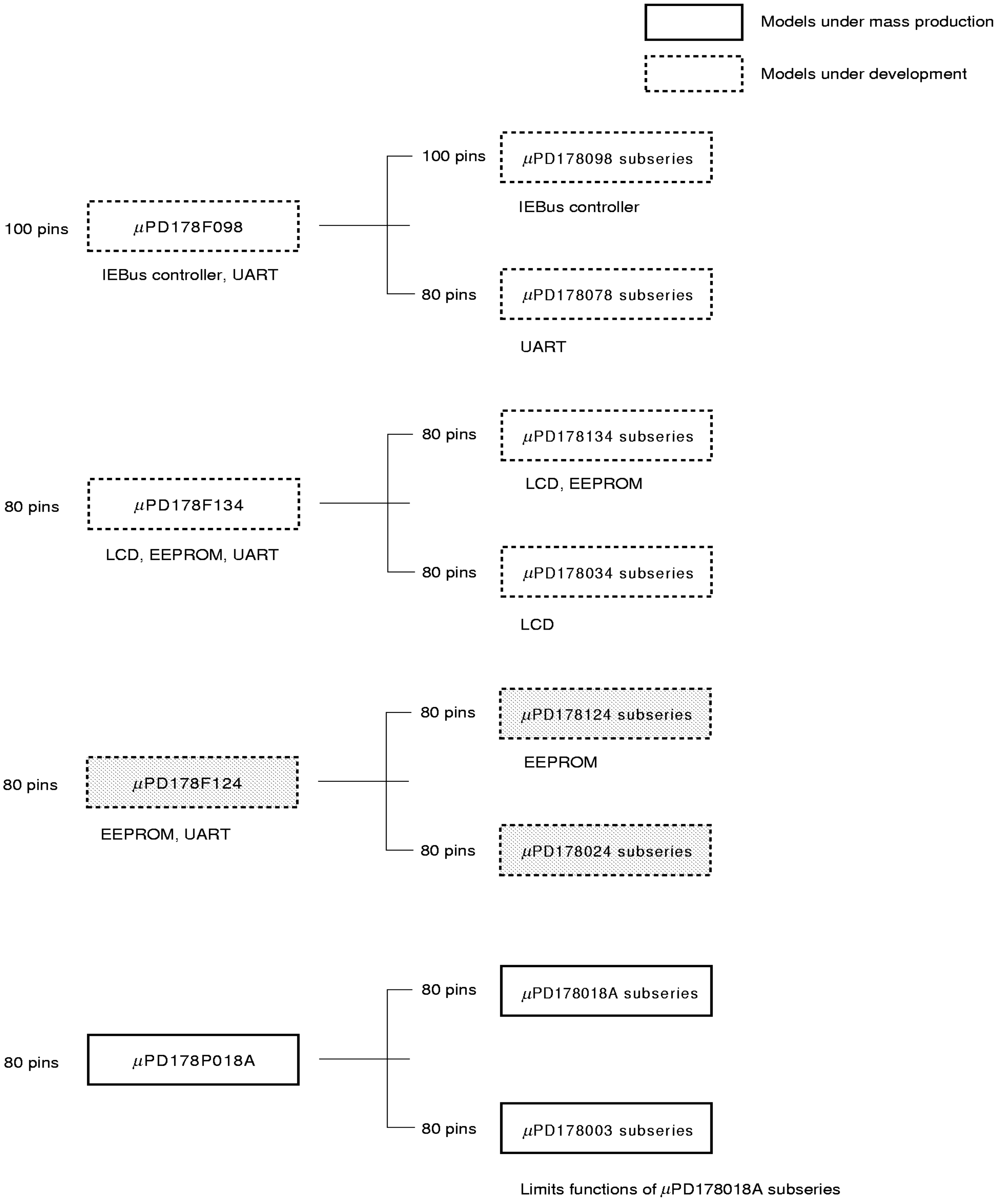
Remark ( ): μPD178F124 only

Pin Name

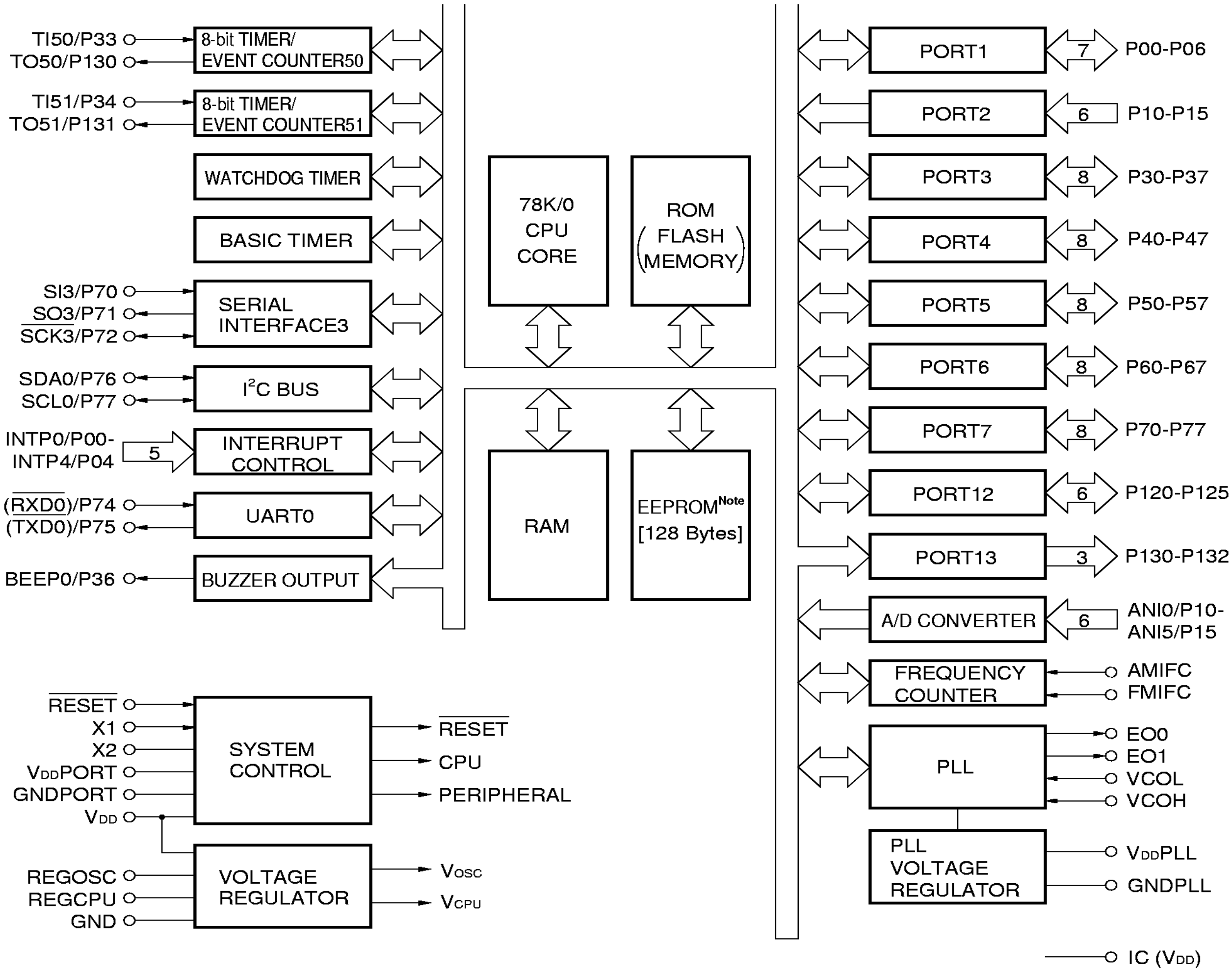
AMIFC	: AM intermediate frequency counter input
ANI0-ANI5	: A/D converter input
BEEP0	: Buzzer output
EO0, EO1	: Error out output
FMIFC	: FM intermediate frequency counter input
GND	: Ground
GNDPLL	: PLL ground
GNDPORT	: Port ground
IC	: Internally connected
INTP0-INTP4	: Interrupt input
P00-P06	: Port 0
P10-P15	: Port 1
P30-P37	: Port 3
P40-P47	: Port 4
P50-P57	: Port 5
P60-P67	: Port 6
P70-P77	: Port 7
P120-P125	: Port 12
P130-P132	: Port 13
REGCPU	: Regulator for CPU power supply
REGOSC	: Regulator for oscillator
$\overline{\text{RESET}}$	: Reset input
RXD0$^{\text{Note}}$	: Serial (UART0) data input
$\overline{\text{SCK3}}$	: Serial (SIO3) clock input/output
SCL0	: Serial (IIC0) clock input/output
SDA0	: Serial (IIC0) data input/output
SI3	: Serial (SIO3) data input
SO3	: Serial (SIO3) data output
TI50, TI51	: 8-bit timer clock input
TO50, TO51	: 8-bit timer output
TXD0$^{\text{Note}}$	: Serial (UART0) data output
VCOL, VCOH	: Local oscillation input
V_{DD}	: Power supply
V_{DD}PLL	: PLL power supply
V_{DD}PORT	: Port power supply
$V_{PP}$$^{\text{Note}}$	: Programming power supply
X1, X2	: Crystal resonator

Note μPD178F124 only

1.5 Development of 8-bit DTS Series



1.6 Block Diagram



Note μPD178022, 178023, and 178024 do not contain the EEPROM.

Remarks 1. The internal ROM and RAM capacities differ depending on the product.

2. ( ): μPD178F124

1.7 Functional Outline

Item		μPD178022	μPD178023	μPD178024	μPD178122	μPD178123	μPD178124	μPD178F124
Internal memory	ROM	16 Kbytes (Mask ROM)	24 Kbytes (Mask ROM)	32 Kbytes (Mask ROM)	16 Kbytes (Mask ROM)	24 Kbytes (Mask ROM)	32 Kbytes (Mask ROM)	32 Kbytes (Flash memory)
	High-speed RAM	512 bytes	1024 bytes		512 bytes	1024 bytes		
	EEPROM	—			128 bytes			
General-purpose register		8 bits × 32 registers (8 bits × 8 registers × 4 banks)						
Minimum instruction execution time		0.45 μs/0.89 μs/1.78 μs/3.56 μs/7.11 μs (with crystal resonator of f_X = 4.5 MHz)						
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 port		Total : 62 pins • CMOS I/O : 53 pins • CMOS input : 6 pins • N-ch open-drain output : 3 pins						
A/D converter		8-bit resolution × 6 channels						
Serial interface		• I^{2}C bus mode[Note]: 1 channel • 3-wire mode: 1 channel						UART mode: 1 channel
Timer		• Basic timer (timer carry FF (10 Hz)) : 1 channel • 8-bit timer/event counter : 2 channels • Watchdog timer : 1 channel						
Buzzer output		1 kHz, 1.5 kHz, 3 kHz, 4 kHz						
Vectored interrupt source	Maskable	Internal : 8 External: 5			Internal : 9 External: 5			Internal : 12 External: 5
	Non-maskable	Internal: 1						
	Software	Internal: 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 in software (1, 3, 9, 10, 12.5, 25, 50 kHz)						
	Charge pump	Error out output: 2 pins						
	Phase comparator	Unlock detectable in software						

Note When the I^{2}C bus mode is used (including when the mode is implemented in software without using the peripheral hardware), consult NEC when ordering a mask.

Item	μPD178022	μPD178023	μPD178024	μPD178122	μPD178123	μPD178124	μPD178F124
Frequency counter	Frequency measurement • AMIFC pin: For 450-kHz counting • FMIFC pin: For 450-kHz/10.7-MHz counting						
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, however.) • 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	• 80-pin plastic QFP (14 × 20 mm, 0.8-mm pitch) • 80-pin plastic QFP (14 × 14 mm, 0.65-mm pitch)						

Note For details, refer to **CHAPTER 19 RESET FUNCTION**.

CHAPTER 2 PIN FUNCTION

2.1 Pin Function List

(1) Port pins

Pin Name	I/O	Function	At Reset	Shared by:
P00-P04	I/O	Port 0. 7-bit I/O port. Can be set in input or output mode in 1-bit units.	Input	INTP0-INTP4
P05, P06				—
P10-P15	Input	Port 1. 6-bit input port.	Input	ANI0-ANI5
P30-P32	I/O	Port 3. 8-bit I/O port. Can be set in input or output mode in 1-bit units.	Input	—
P33				TI50
P34				TI51
P35				—
P36				BEEP0
P37				—
P40-47	I/O	Port 4. 8-bit I/O port. Can be set in input or output mode in 1-bit units. When using as input port, internal pull-up resistors can be specified in software. Interrupt function by key input is provided.	Input	—
P50-P57	I/O	Port 5. 8-bit I/O port. Can be set in input or output mode in 1-bit units.	Input	—
P60-P67	I/O	Port 6. 8-bit I/O port. Can be set in input or output mode in 1-bit units.	Input	—
P70	I/O	Port 7. 8-bit I/O port. Can be set in input or output mode in 1-bit units.	Input	SI3
P71				SO3
P72				$\overline{\text{SCK3}}$
P73				—
P74				$\overline{\text{RXD0}}$[Note]
P75				$\overline{\text{TXD0}}$[Note]
P76				SDA0
P77				SCL0
P120-P125	I/O	Port 12. 6-bit I/O port. Can be set in input or output mode in 1-bit units.	Input	—
P130	Output	Port 13. 3-bit output port. N-ch open-drain output port (12 V withstand)	Low-level output	TO50
P131				TO51
P132				—

Note μPD178F124 only

(2) Pins other than port pins

Pin Name	I/O	Function		At Reset	Shared by:
INTP0-INTP4	Input	External maskable interrupt input whose valid edge (rising edge, falling edge, or both rising and falling edges) can be specified.		Input	P00-P04
SI3	Input	Serial data input to serial interface.		Input	P70
SO3	Output	Serial data output from serial interface.		Input	P71
SDA0	I/O	Serial data input/output to/from serial interface.	N-ch open drain I/O	Input	P76
$\overline{\text{SCK3}}$	I/O	Serial clock input/output to/from serial interface.		Input	P72
SCL0			N-ch open drain I/O		P77
$\overline{\text{RXD0}}$ Note 1	Input	Serial data input to asynchronous serial interface (UART0).		Input	P74
$\overline{\text{TXD0}}$ Note 1	Output	Serial data output from asynchronous serial interface (UART0).			P75
TI50	Input	External count clock input to 8-bit timer (TM50).		Input	P34
TI51		External count clock input to 8-bit timer (TM51).			P35
TO50	Output	8-bit timer (TM50) output.		Low-level output	P130
TO51		8-bit timer (TM51) output.			P131
BEEP0	Output	Buzzer output.		Input	P36
ANI0-ANI5	Input	Analog input to A/D converter.		Input	P10-P15
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		Inputs local oscillation frequency of PLL (in VHF mode).			
AMIFC	Input	Input to AM intermediate frequency counter.		Input	–
FMIFC		Input to FM or AM intermediate frequency counter.			
$\overline{\text{RESET}}$	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 0.1-μF capacitor.		–	–
REGCPU	–	Regulator for CPU power supply. Connect this pin to GND via 0.1-μF capacitor.		–	–
V_{DD}	–	Positive power supply.		–	–
GND	–	Ground.		–	–
V_{DD}PORT	–	Port power supply.		–	–
GNDPORT	–	Port ground.		–	–
V_{DD}PLL Note 2	–	PLL positive power supply.		–	–
GNDPLL Note 2	–	PLL ground.		–	–
IC	–	Internally connected. Directly connect this pin to GND.			
V_{PP}	–	Pin to apply high voltage at program writing/verifying. Directly connect this pin to GND in normal operating mode.		–	–

Notes 1. μPD178F124 only.

2. 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 P06 (Port 0)

These are 7-bit input/output ports. Besides serving as input/output ports, they function as an external interrupt input. The following operating modes can be specified bit-wise.

(1) Port mode

Function as 7-bit input/output ports. Can be specified for input or output ports bit-wise with a port mode register 0 (PM0).

(2) Control mode

These ports function as an external interrupt input pin (INTP0-INTP4).

These are external interrupt input pins which can specify valid edges (rising edge, falling edge, and both rising and falling edges).

2.2.2 P10 to P15 (Port 1)

These are 6-bit input ports. Besides serving as input ports, they function as an A/D converter analog input.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 6-bit input ports.

(2) Control mode

These ports function as A/D converter analog input pins (ANI0-ANI5).

2.2.3 P30 to P37 (Port 3)

These are 8-bit input/output ports. Beside serving as input/output ports, they function as timer input and buzzer output. Which function is selected can be specified by the peripheral function sharing register 3 (PF3).

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 8-bit input/output ports. They can be specified bit-wise as input or output ports with port mode register 3 (PM3).

(2) Control mode

These ports function as timer input (TI50, T51) and buzzer output (BEEP0).

(a) TI50, TI51

Pins for external clock input to the 8-bit timer/event counter.

(b) BEEP0

Buzzer output pin.

2.2.4 P40 to P47 (Port 4)

These are 8-bit input/output ports.

They can be specified in 8-bit units for input or output ports by using the port mode register 4 (PM4).

When using as input port, internal pull-up resistors can be specified by pull-up resistor option register 4 (PU4). Interrupt function by key input is provided.

2.2.5 P50 to P57 (Port 5)

These are 8-bit input/output ports.

They can be specified bit-wise as input/output ports with port mode register 5 (PM5).

2.2.6 P60 to P67 (Port 6)

These are 8-bit input/output ports.

They can be specified bit wise as input or output ports with port mode register 6 (PM6).

2.2.7 P70 to P77 (Port 7)

These pins constitute an 8-bit input/output port. In addition to port pins, these pins are also used to input/output serial interface data, clock, and asynchronous serial interface data.

These pins can be set in the following operation modes in 1-bit units.

(1) Port mode

These pins function as an 8-bit I/O port. They can be specified in 1-bit units as input or output ports with port mode register 7 (PM7).

(2) Control mode

These pins function as serial interface data input/output, clock input/output, and asynchronous serial interface data input/output pins.

(a) SI3, SO3, SDA0[Note]

Serial data input/output pins of the serial interface.

(b) $\overline{\text{SCK3}}$, SCL0[Note]

Serial clock input/output pins of the serial interface.

(c) $\overline{\text{RXD0}}$, $\overline{\text{TXD0}}$ (μPD178F124 only)

Serial data input/output pins of asynchronous serial interface.

Note For the I^2C bus mode.

2.2.8 P120 to P125 (Port 12)

These pins constitute a 6-bit I/O port.

This port can be specified in 1-bit units as the input or output mode with port mode register 12 (PM12).

2.2.9 P130 to P132 (Port 13)

These are 3-bit output ports. They are N-ch open drain ports with a 12-V withstand voltage. Besides serving as output port, they are used for timer output.

The following operating modes can be specified bit-wise.

(1) Port mode

These ports function as 3-bit output port.

(2) Control mode

These ports function as output for 8-bit timer/event counter.

TO50, TO51

These pins are output for 8-bit timer/event counter.

2.2.10 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.

2.2.11 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 in 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 in 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 VCOL pin does not become active even if the FM mode is selected. In this case, set VCOHDMD to 1 (high-impedance state).

2.2.12 AMIFC

Input pin of the AM intermediate frequency counter.

2.2.13 FMIFC

Input pin of the FM intermediate frequency counter or AM intermediate frequency counter.

2.2.14 $\overline{\text{RESET}}$

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

2.2.15 X1, X2

Crystal resonator connect pins for system clock oscillation.

2.2.16 REGOSC

Regulator pin for oscillator. Connect to GND via a 0.1-μF capacitor.

2.2.17 REGCPU

Regulator pin for CPU power supply. Connect to GND via a 0.1-μF capacitor.

2.2.18 V_{DD}

Positive power supply pin.

2.2.19 GND

Ground potential pin.

2.2.20 V_{DD}PORT

Positive power supply pin for port.

2.2.21 GNDPORT

Ground potential pin for port.

2.2.22 V_{DD}PLL

Positive power supply pin for PLL.

2.2.23 GNDPLL

Ground potential pin for PLL.

2.2.24 V_{PP} (μPD178F124 only)

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

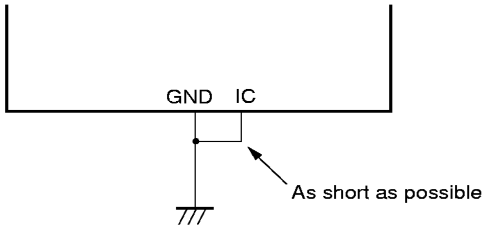
In the normal operation mode, directly connect this pin to GND.

2.2.25 IC (Mask ROM version only)

The IC (Internally Connected) pin is provided to set the test mode to check the μPD178024 and 178124 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 those two pins is too long or an 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 I/O Circuits of Pins and Recommended Connections of Unused Pins

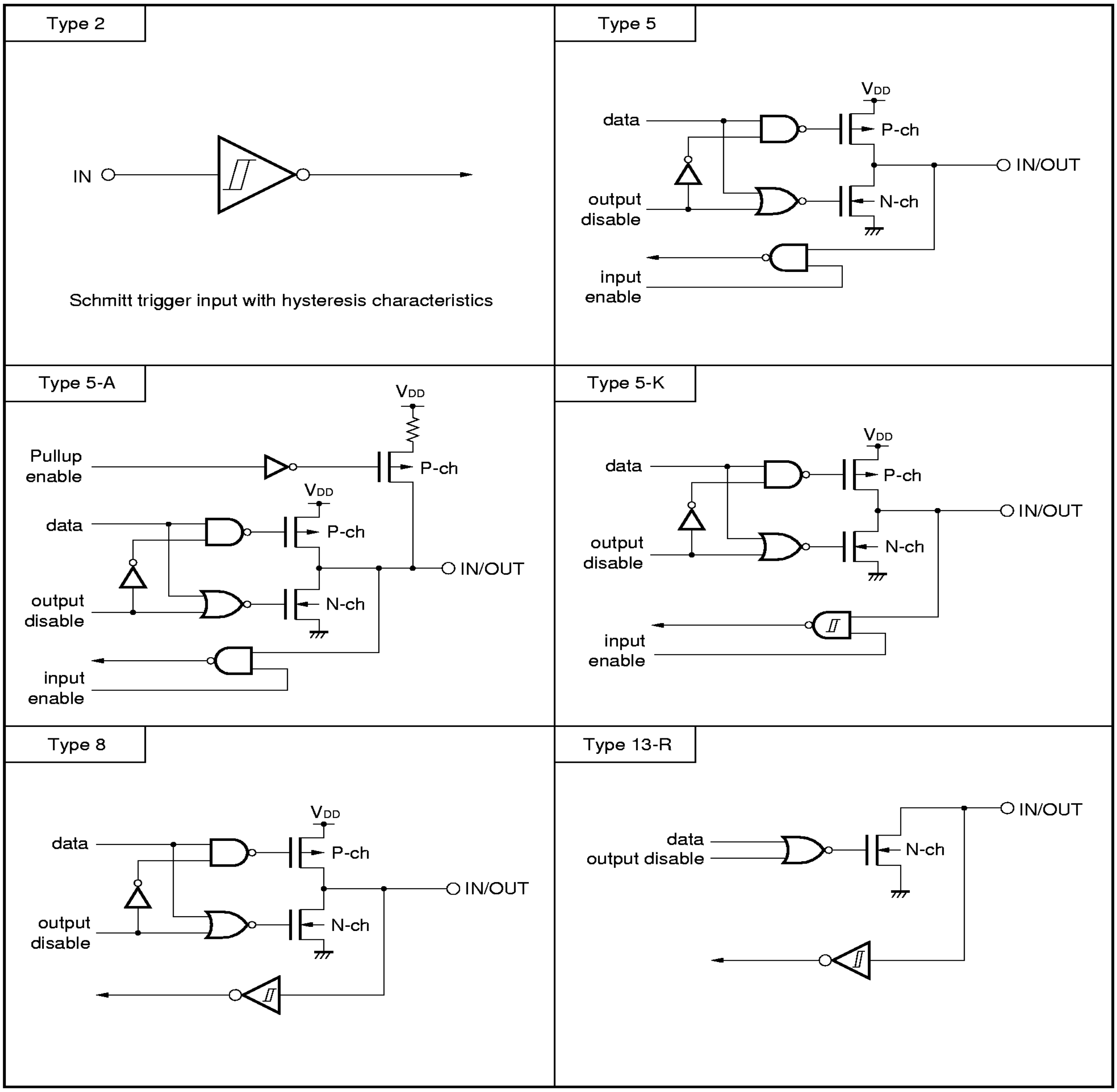
Table 2-1 shows the types of the I/O circuits of the respective pins and the recommended connections of the pins when they are not used. For the configuration of the I/O circuit of each pin, refer to Figure 2-1.

Table 2-1. I/O Circuit Type of Each Pin

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pin
P00/INTP0-P04/INTP4	8	I/O	Set these pins in general-purpose input mode in software, and connect each of them to V_{DD}, V_{DD}PORT, GND, or GNDPORT via resistor.
P05, P06			
P10/ANI0-P15/ANI5	25	Input	Connect each of them to V_{DD}, V_{DD}PORT, GND, or GNDPORT via resistor.
P30-P32	5	I/O	Set these pins in general-purpose input mode in software, and output low-level signal. Leave unconnected.
P33/TI50	5-K		
P34/TI51			
P35	5		
P36/BEEP0			
P37			
P40-P47	5-A		Set these pins in general-purpose input mode in software, and connect each of them to GND or GNDPORT via resistor.
P50-P57	5		Set these pins in general-purpose input mode in software, and connect each of them to V_{DD}, V_{DD}PORT, GND, or GNDPORT via resistor.
P60-P67	5		Set these pins in general-purpose input mode in software, and output low-level signal. Leave unconnected.
P70/SI3	5-K		Set these pins in general-purpose input mode in software, and connect each of them to V_{DD}, V_{DD}PORT, GND, or GNDPORT via resistor.
P71/SO3	5		
P72/$\overline{\text{SCK3}}$	5-K		
P73	5		
P74/$\overline{\text{RXD0}}$$^{\text{Note}}$	5-K		
P75/$\overline{\text{TXD0}}$$^{\text{Note}}$	5		
P76/SDA0	13-R		
P77/SCL0			
P120-P125	5		
P130/TO50	19	Output	Set these pins to low-level output in software and leave unconnected.
P131/TO51			
P132			
EO0, EO1	DTS-EO1	Output	Leave unconnected.
VCOL, VCOH	DTS-AMP	Input	Disable PLL in software and select pull-down.
AMIFC, FMIFC			Set these pins in general-purpose input port mode in software and connect each of them to V_{DD}, V_{DD}PORT, GND, or GNDPORT via resistor.
REGOSC, REGCPU	–	–	Connect these pins to GND via 0.1-μF capacitor.
$\overline{\text{RESET}}$	2	Input	–
V_{DD}PLL	–	–	Connect this pin to V_{DD}.
GNDPLL			Directly connect these pins to GND or GNDPORT.
IC (Mask ROM model)			
V_{PP} (μPD178F124 only)			

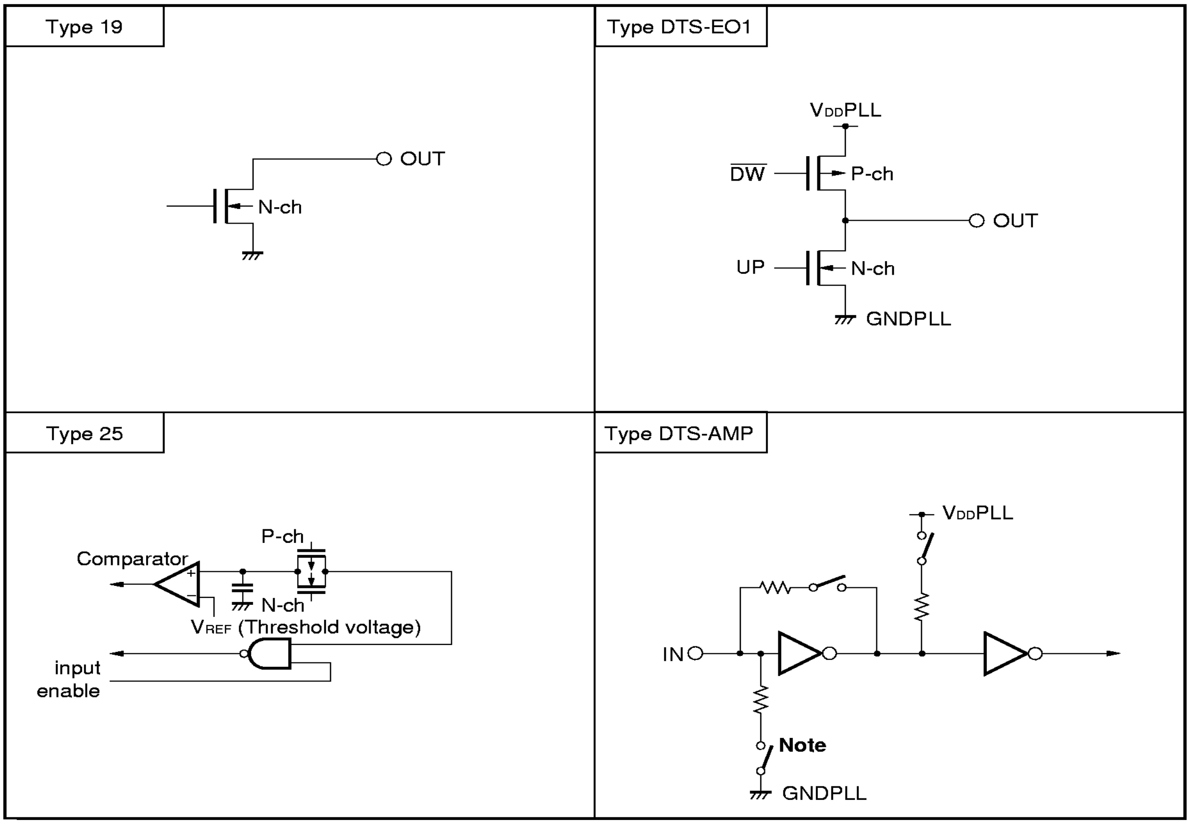
Note $\overline{\text{RXD0}}$ and $\overline{\text{TXD0}}$ are provided for the μPD178F124 only.

Figure 2-1. I/O Circuits of Respective Pins (1/2)



Remark V$_{DD}$ and GND are the positive power supply and ground pins for all port pins. Take V$_{DD}$ and GND as V$_{DD}$PORT and GNDPORT.

Figure 2-1. I/O Circuits of Respective Pins (2/2)



Note This switch is selectable in 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. Take V_{DD} and GND as V_{DD}PORT and GNDPORT.

[MEMO]

CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Spaces

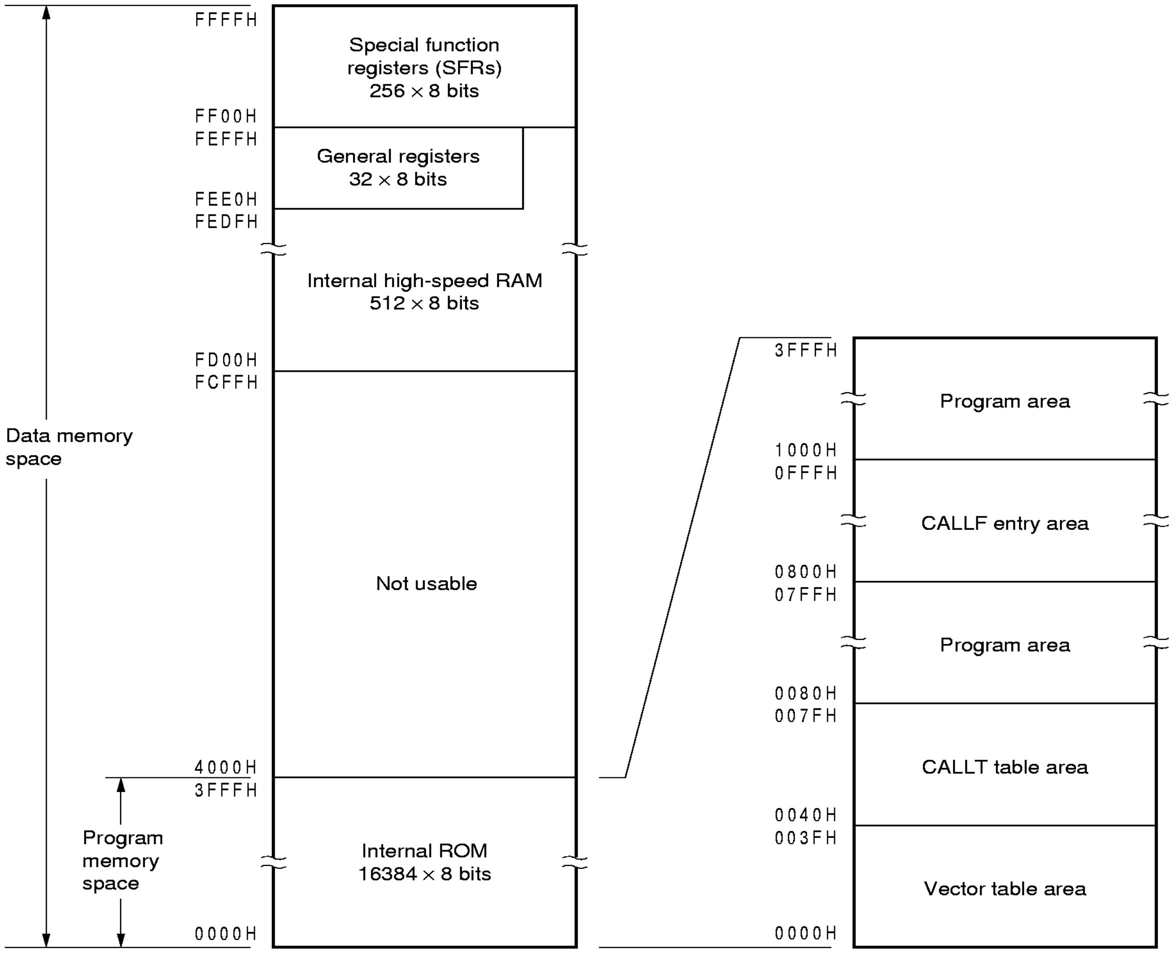
The initial value of the memory size select register (IMS) is CFH . The following values must be set to the registers of each model.

Part Number	IMS
μPD178022, 178122	44H
μPD178023, 178123	C6H
μPD178024, 178124	C8H
μPD178F124	Value equivalent to mask ROM model

(1) μPD178022

Set the value of the memory size select register (IMS) to 44H (the initial value is CFH).

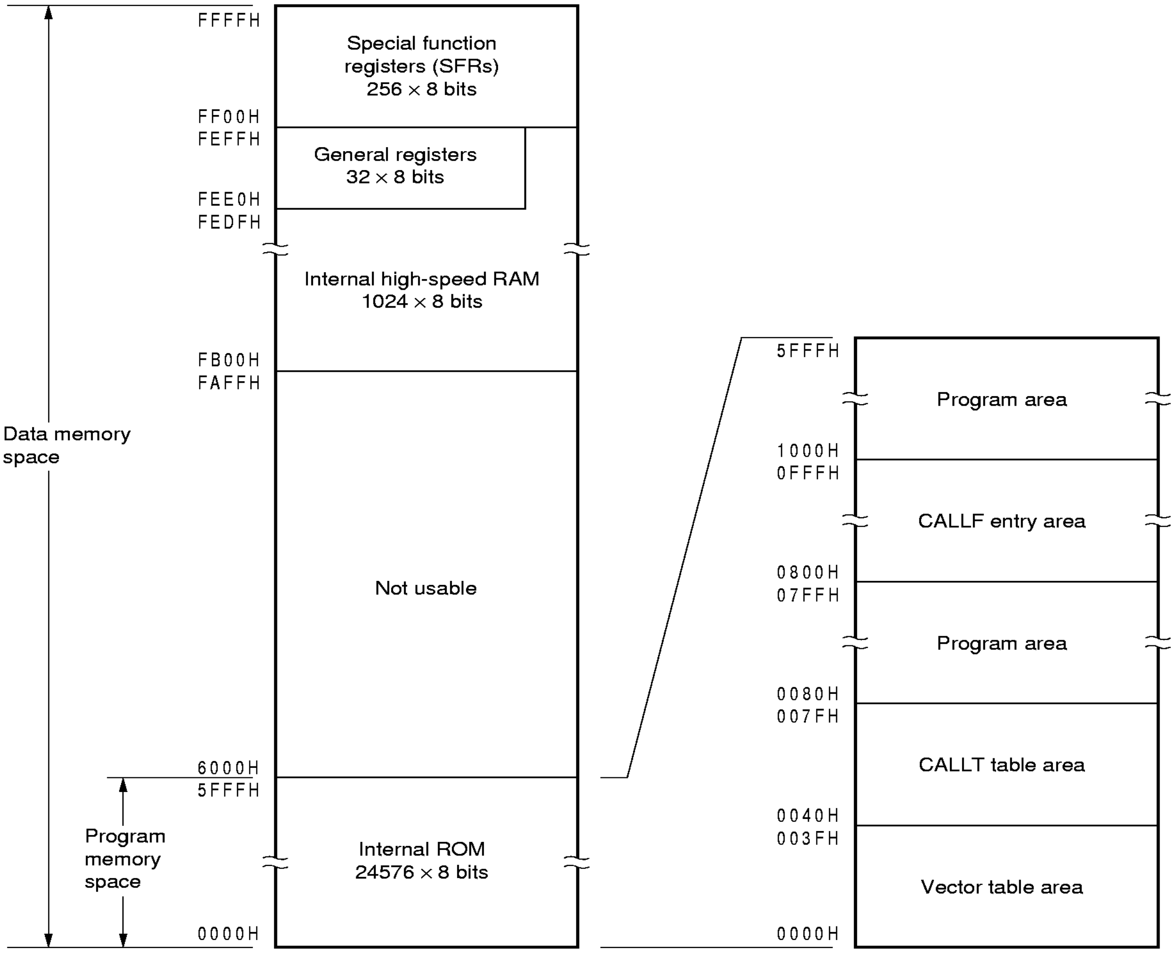
Figure 3-1. Memory Map (μPD178022)



(2) μPD178023

Set the value of the memory size select register (IMS) to C6H (the initial value is CFH).

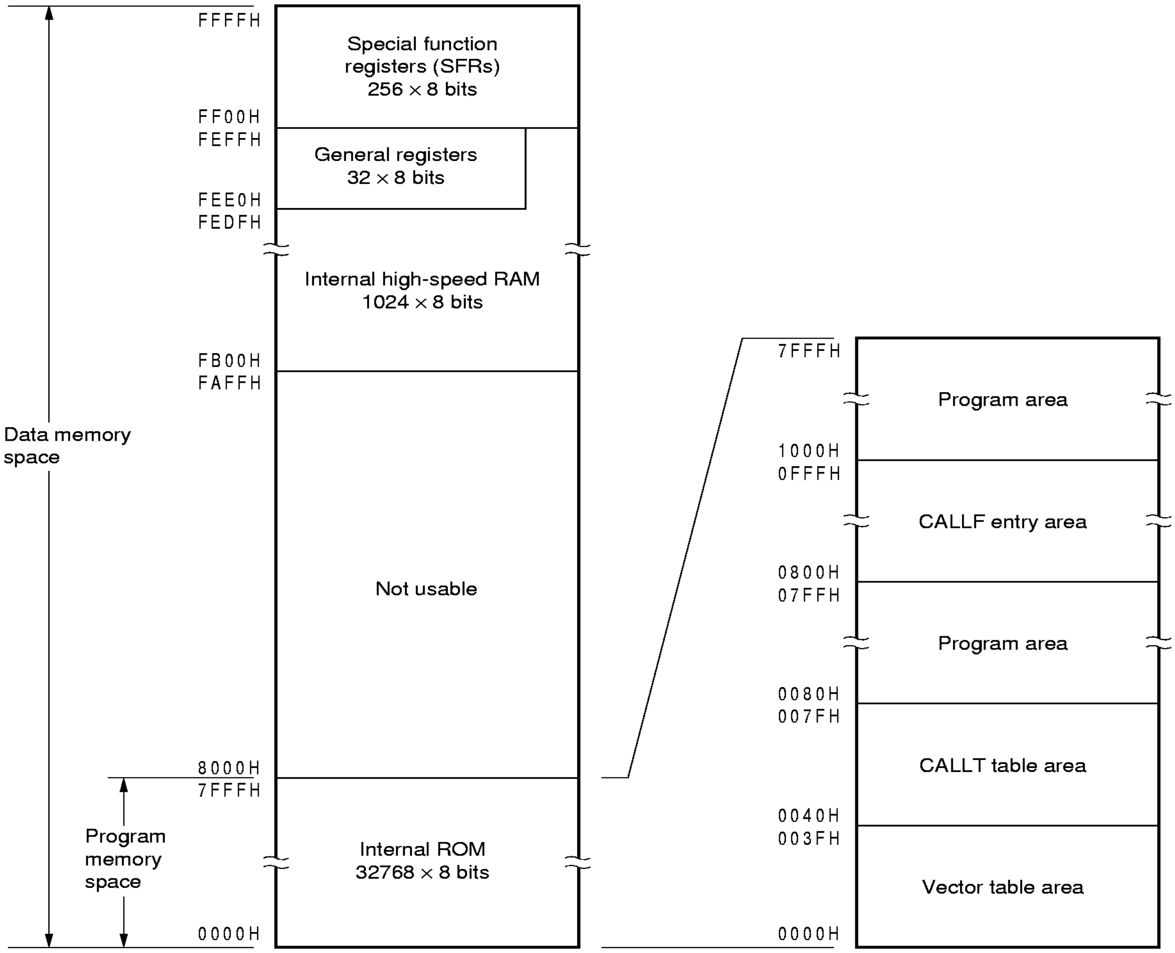
Figure 3-2. Memory Map (μPD178023)



(3) μPD178024

Set the value of the memory size select register (IMS) to C8H (the initial value is CFH).

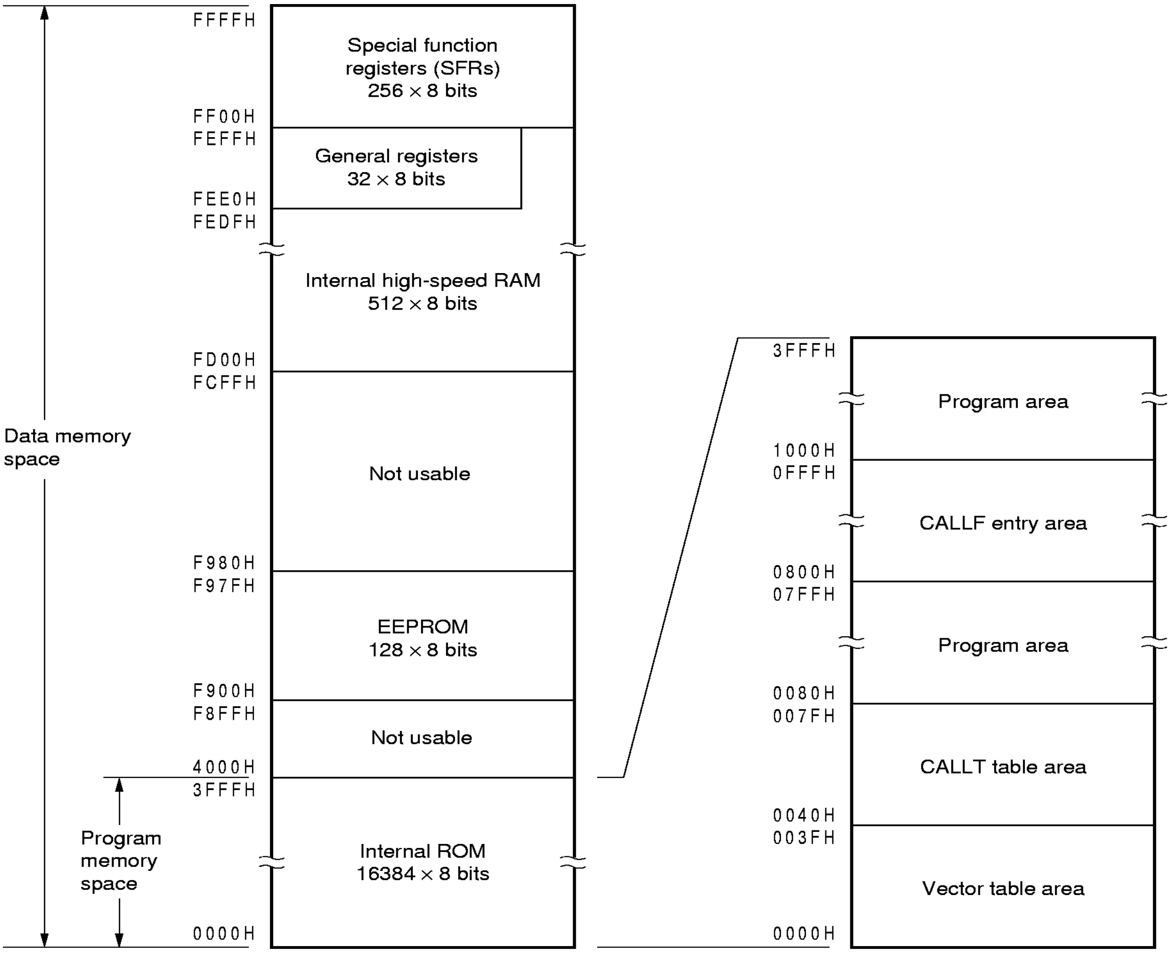
Figure 3-3. Memory Map (μPD178024)



(4) μPD178122

Set the value of the memory size select register (IMS) to 44H (the initial value is CFH).

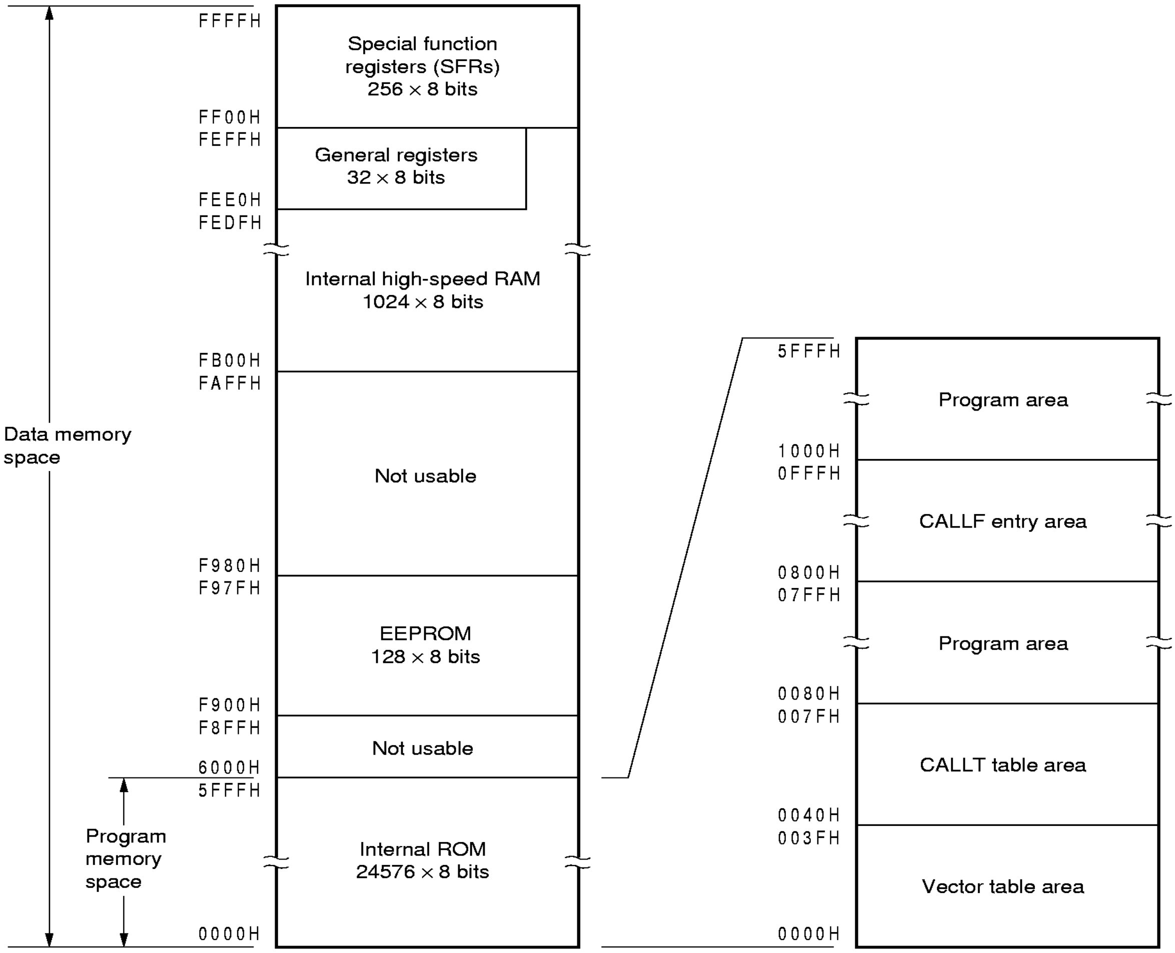
Figure 3-4. Memory Map (μPD178122)



(5) μPD178123

Set the value of the memory size select register (IMS) to C6H (the initial value is CFH).

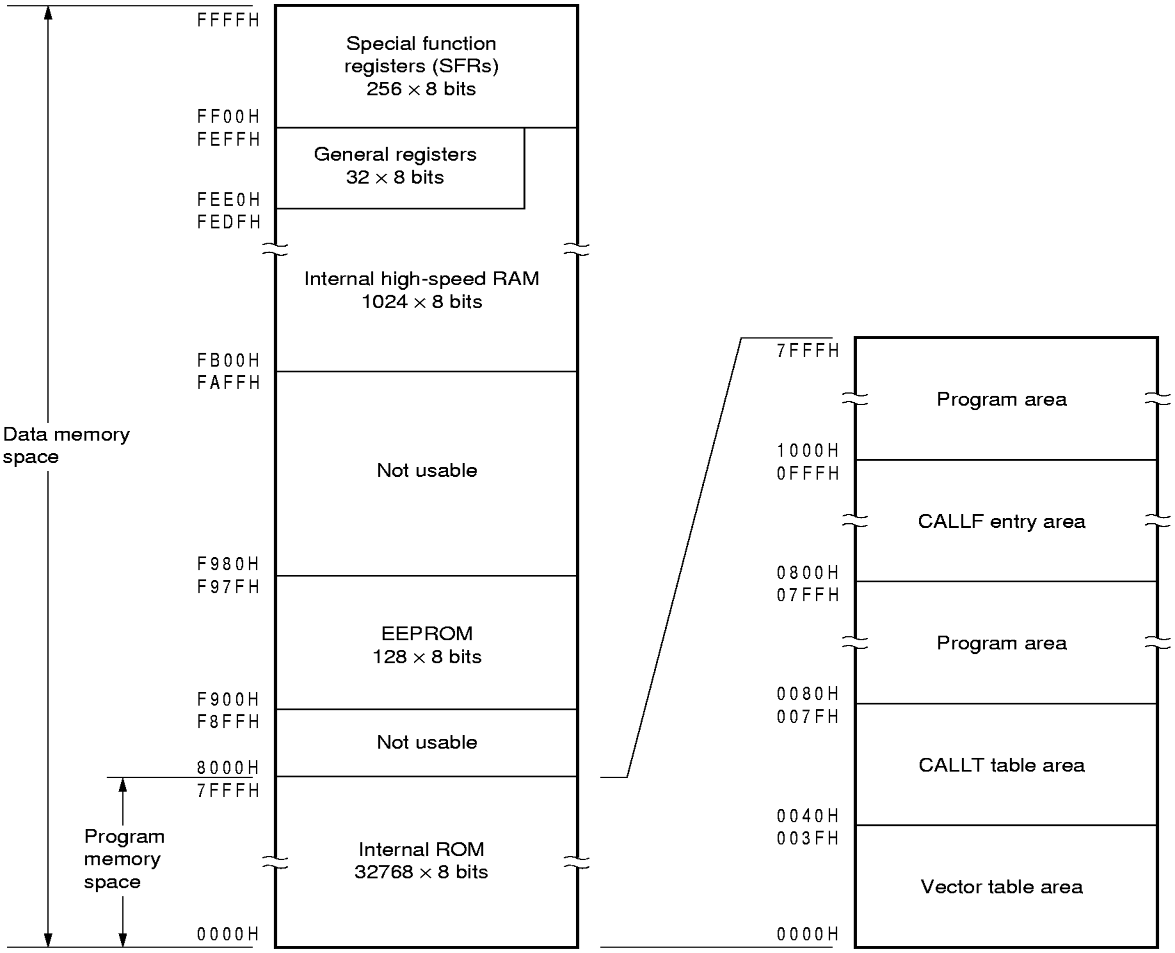
Figure 3-5. Memory Map (μPD178123)



(6) μPD178124

Set the value of the memory size select register (IMS) to C8H (the initial value is CFH).

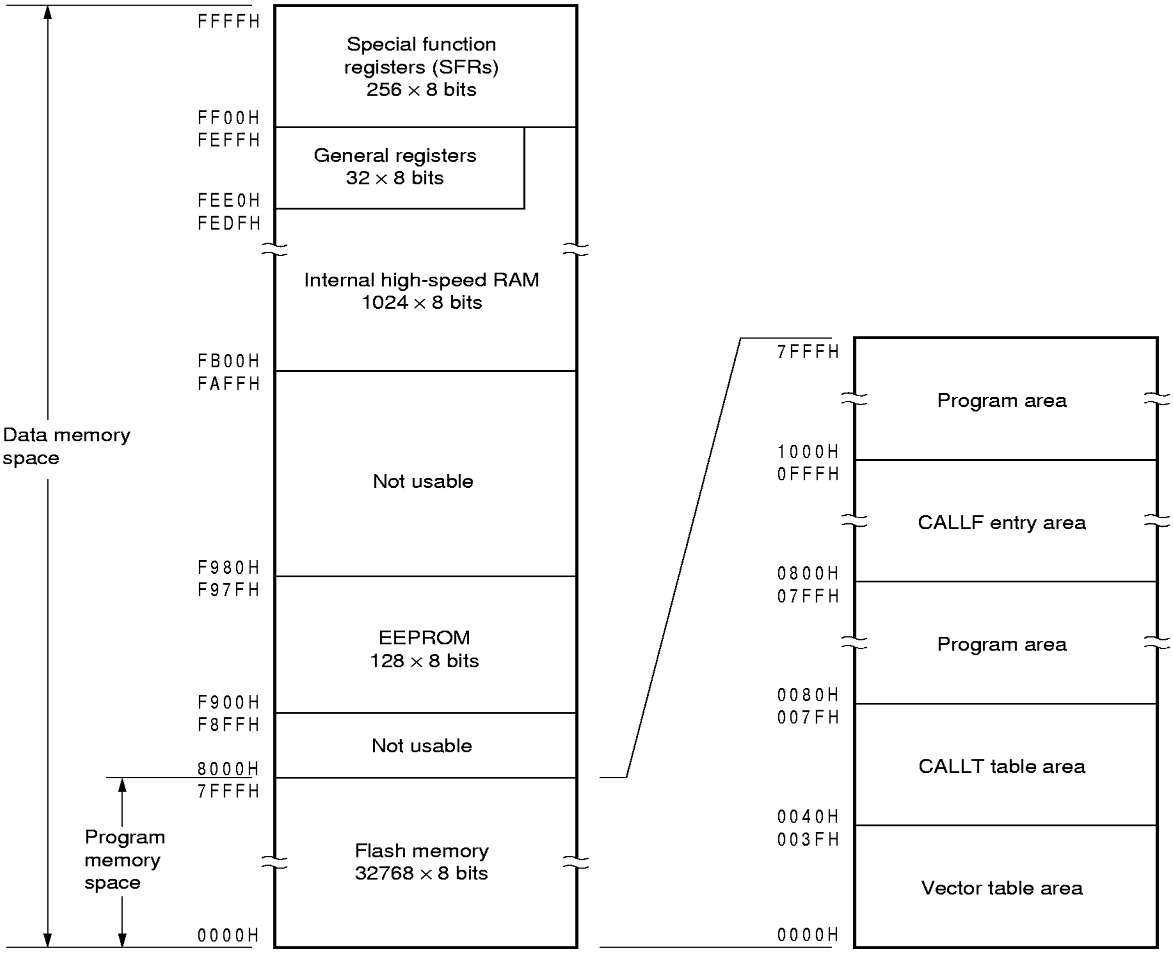
Figure 3-6. Memory Map (μPD178124)



(7) μPD178F124

Set the value of the memory size select register (IMS) to the value corresponding to that of the mask ROM models (the initial value is CFH).

Figure 3-7. Memory Map (μPD178F124)



3.1.1 Internal program memory space

Programs and table data are stored in internal program space, and are usually addressed by the program counter (PC).

The μPD178024 and 178124 subseries have an internal ROM (or flash memory) as shown in the following table.

Table 3-1. Internal Memory Capacities

Model	Structure	Capacity
μPD178022, 178122	Mask ROM	16384 × 8 bits (0000H-3FFFH)
μPD178023, 178123		24576 × 8 bits (0000F-5FFFH)
μPD178024, 178124		32768 × 8 bits (0000H-7FFFH)
μPD178F124	Flash memory	

The following areas are assigned to the internal program memory space.

(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 each interrupt request are stored in the vector table area. Of the 16-bit address, low-order 8 bits are stored at even addresses and high-order 8 bits are stored at odd addresses.

Table 3-2. Vector Table

Vector Table Address	Interrupt Request
0004H	INTWDT
0006H	INTP0
0008H	INTP1
000AH	INTP2
000CH	INTP3
000EH	INTP4
0010H	INTKY
0012H	INTIIC0
0014H	INTBTM0
0016H	INTAD3
0018H	INTEE[Note 1]
001AH	INTCSI3
001CH	INTTM50
001EH	INTTM51
0020H	INTSER0[Note 2]
0022H	INTSR0[Note 2]
0024H	INTST0[Note 2]
003EH	BRK

Notes 1. μPD178122, 178123, 178124, and 178F124 only.

2. μPD178F124 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 μPD178024 and 178124 subseries units incorporate the following RAMs.

(1) Internal high-speed RAM

The μPD178022 and 178122 have an internal high-speed RAM structure of 512 × 8 bits, and the μPD178023, 178024, 178123, 178124, and 178F124 have a RAM structure of 1024 × 8 bits.

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

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

3.1.3 Special Function Register (SFR) area

An on-chip peripheral hardware special function register (SFR) is allocated in the area FF00H to FFFFH. (Refer to **Table 3-4 Special Funtion Registers**).

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

3.1.4 Data memory addressing

Addressing refers to the method of specifying the address of the instruction to be executed next or the address of the register or memory relevant to the execution of instructions.

The address of an instruction to be executed next is addressed by the program counter (PC) (for details, refer to **3.3 Instruction Address Addressing**).

Several addressing modes are provided for addressing the memory relevant to the execution of instructions for the μPD178024, 178124 Subseries, based on operability and other considerations. For areas containing data memory in particular, special addressing methods designed for the functions of special function registers (SFR) and general-purpose registers are available for use. Data memory addressing is illustrated in Figures 3-8 to 3-14. For the details of each addressing mode, refer to **3.4 Operand Address Addressing**.

Figure 3-8. Data Memory Addressing (μPD178022)

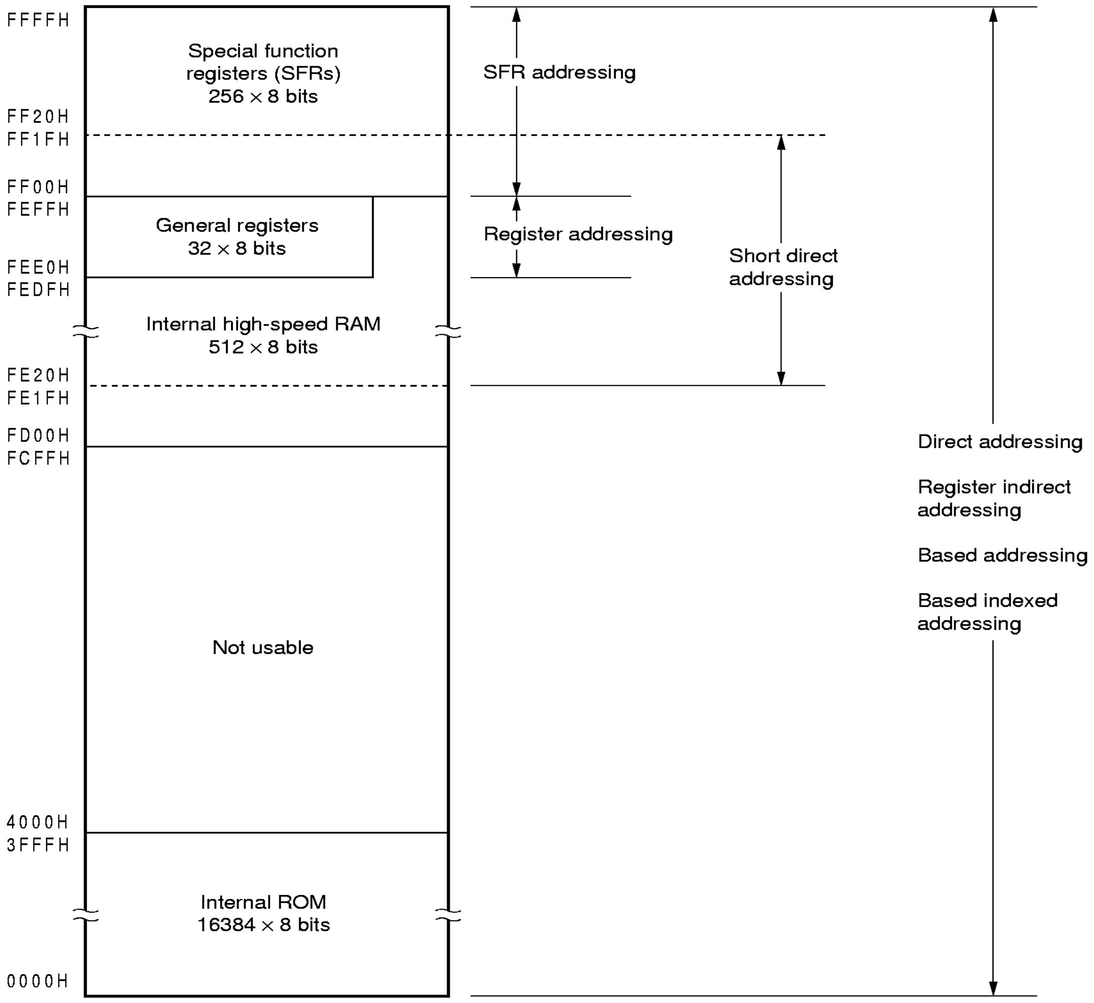


Figure 3-9. Data Memory Addressing (μPD178023)

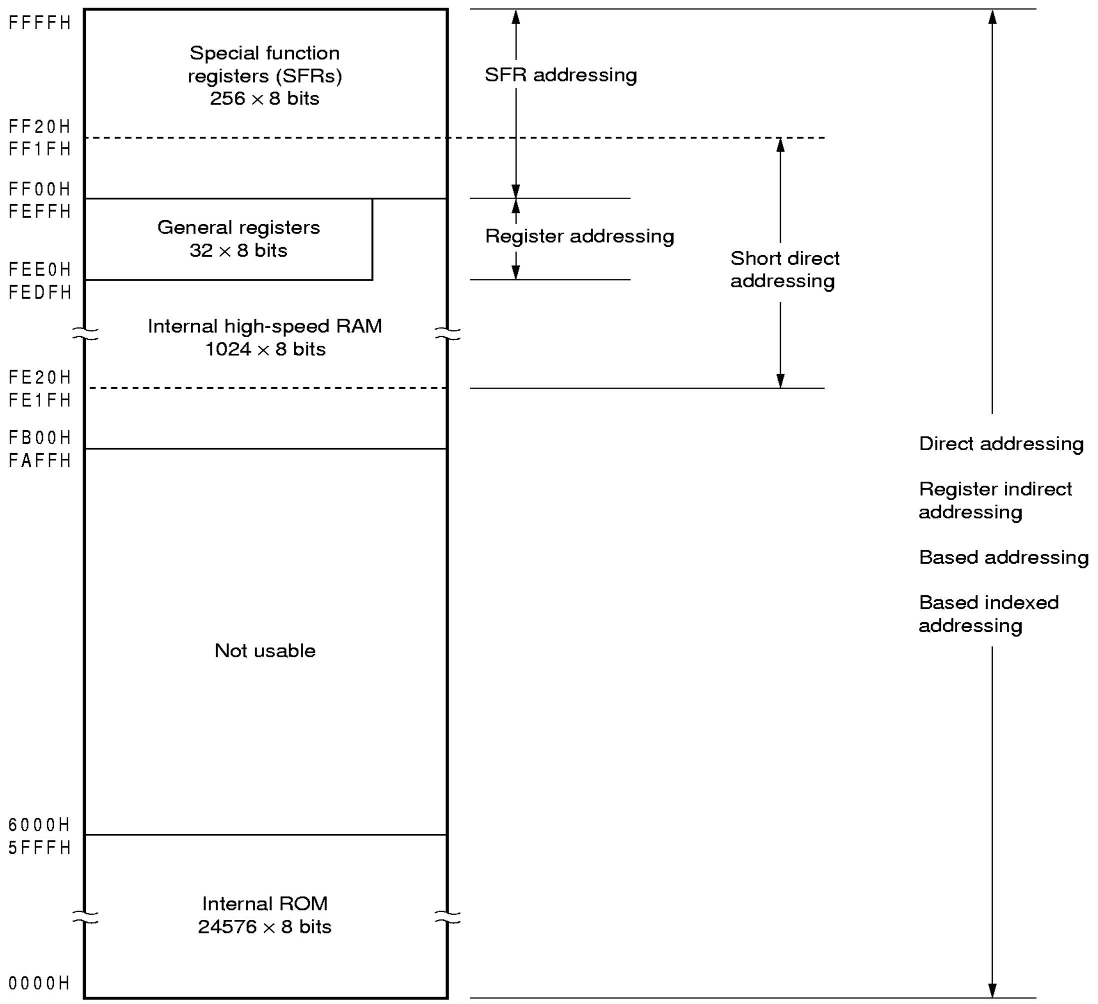


Figure 3-10. Data Memory Addressing (μPD178024)

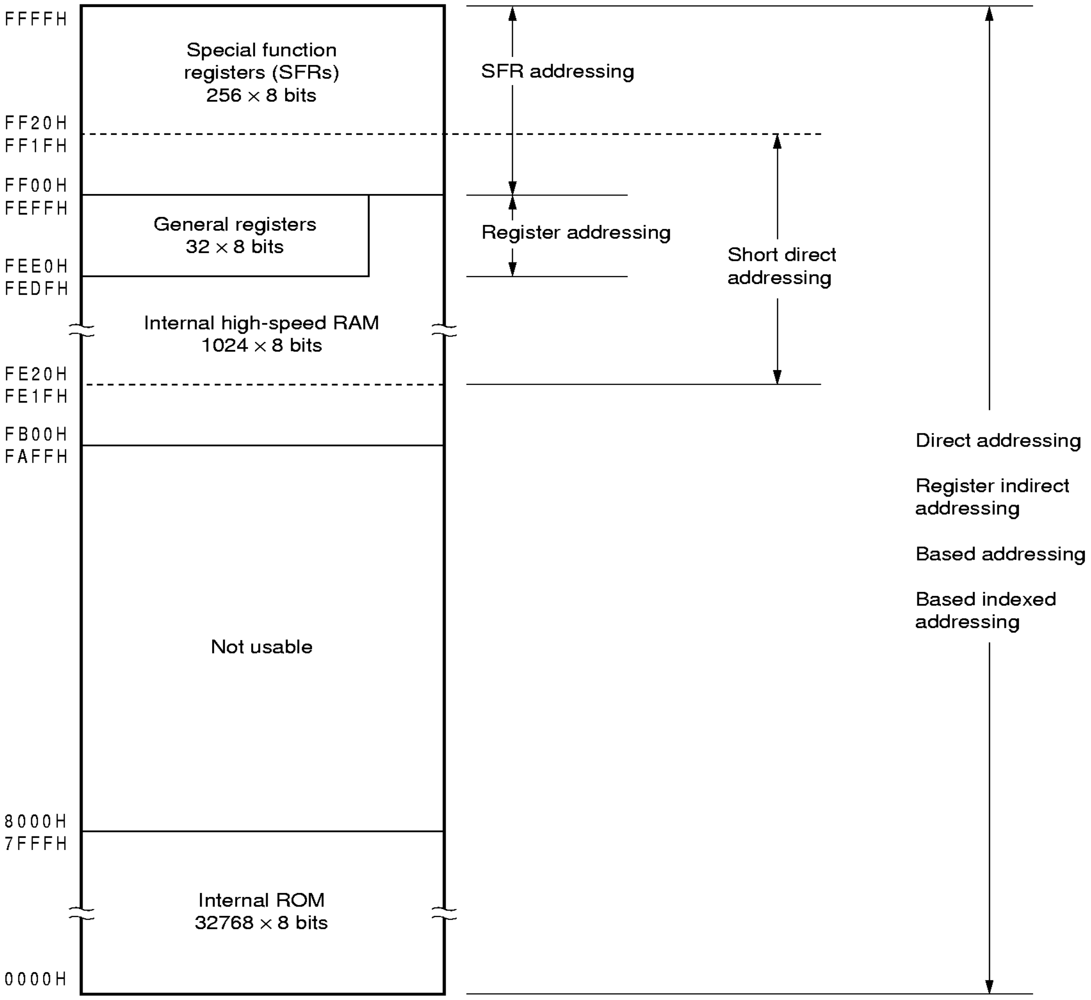


Figure 3-11. Data Memory Addressing (μPD178122)

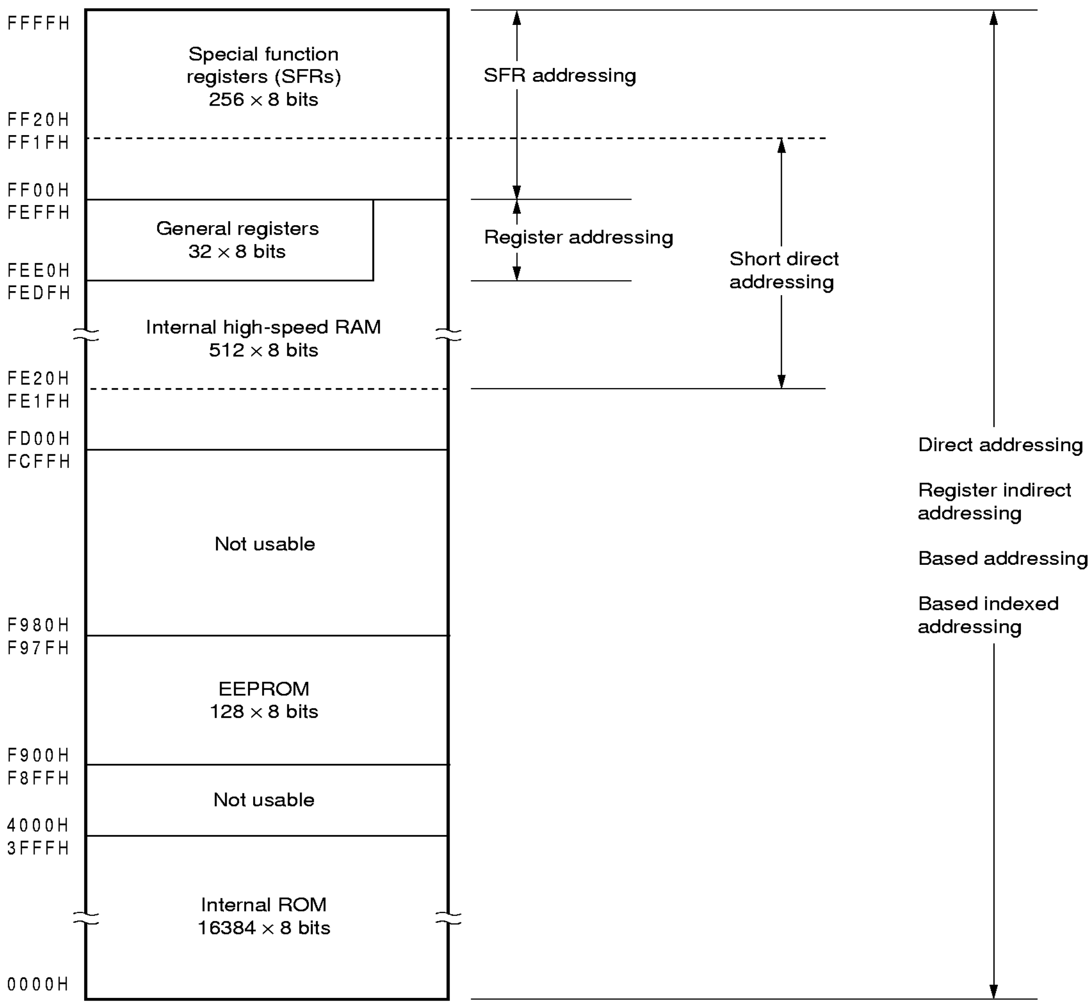


Figure 3-12. Data Memory Addressing (μPD178123)

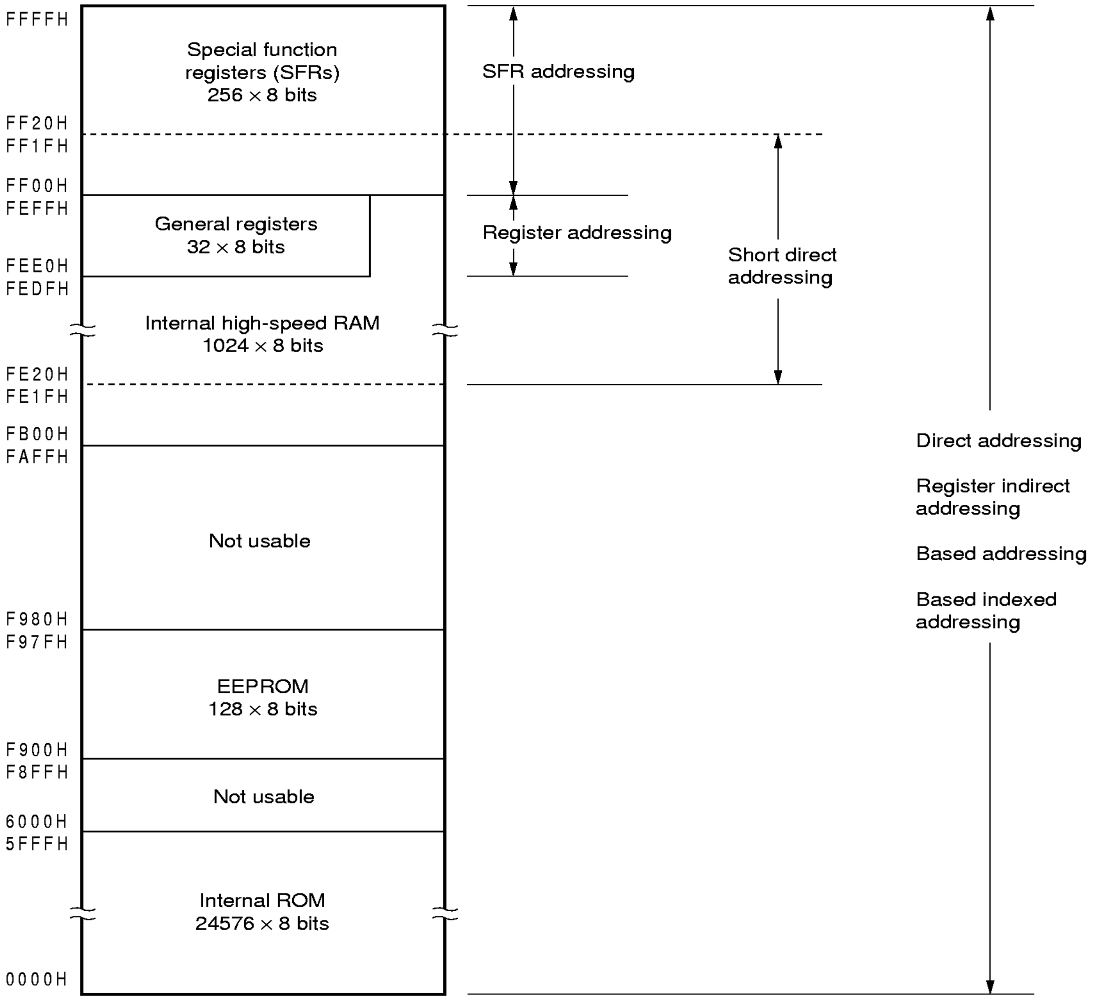


Figure 3-13. Data Memory Addressing (μPD178124)

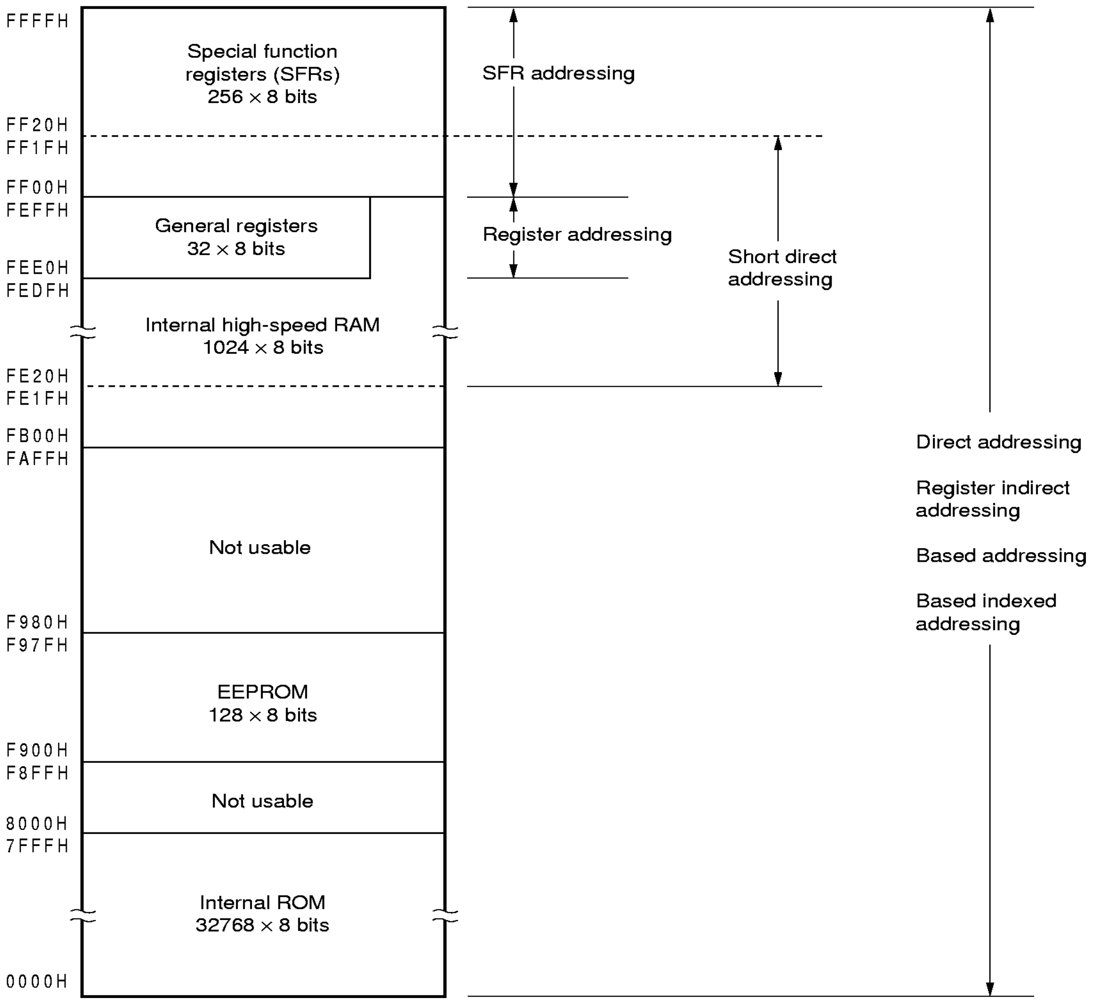
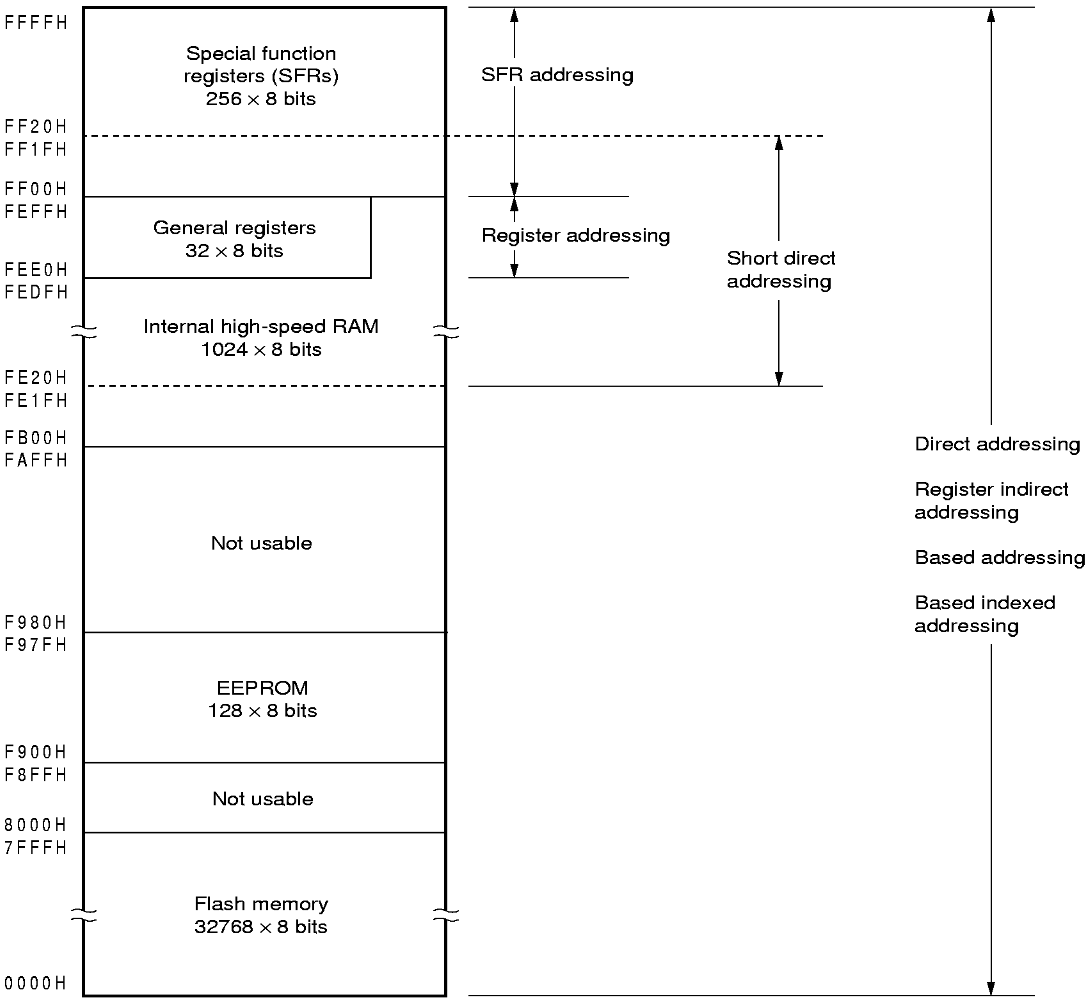


Figure 3-14. Data Memory Addressing (μPD178F124)



3.2 Processor Registers

The μPD178024 and 178124 subseries units 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 which 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 3-15. Configuration of Program Counter

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

(2) Program status word (PSW)

The program status word is an 8-bit register consisting of various flags to be set/reset by instruction execution.

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 3-16. 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 acknowledge 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 acknowledging 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 is reset to (0) upon DI instruction execution or interrupt acknowledgement 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.

In these flags, the 2-bit information which indicates the register bank selected by SEL RBn instruction execution is stored.

(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 **15.3 (3) Priority specification flag registers (PR0L, PR0H, PR1L)**) is disabled. Whether an interrupt request is actually accepted depends on the status of the interrupt enable flag (IE).

(f) Carry flag (CY)

This flag stores 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 to hold the start address of the memory stack area. Only the internal high-speed RAM area (FD00H-FEFFH for μPD178022 and 178122, FB00H-FEFFH for μPD178023, 178024, 178123, 178124, and 178F124) can be set as the stack area.

Figure 3-17. 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 (restored) from the stack memory.

Each stack operation saves/restores data as shown in Figures 3-18 and 3-19.

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

Figure 3-18. Data to be Saved to Stack Memory

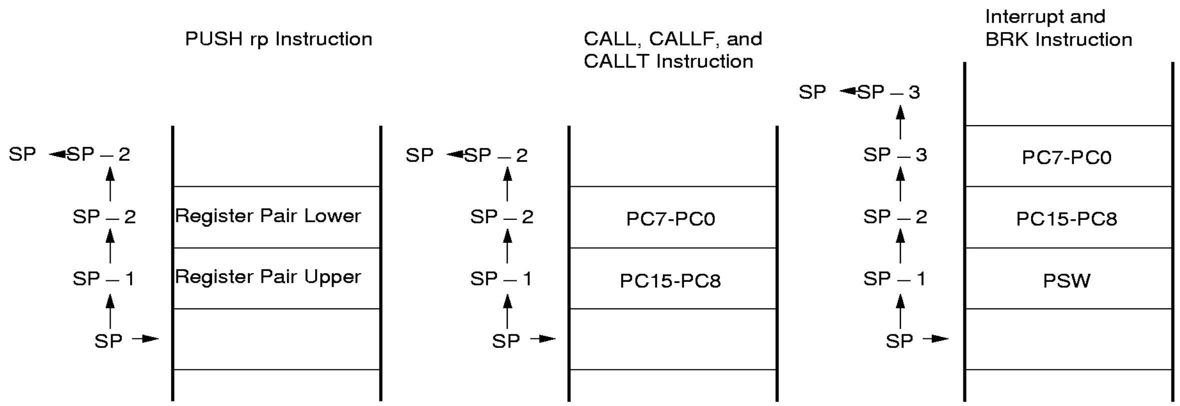
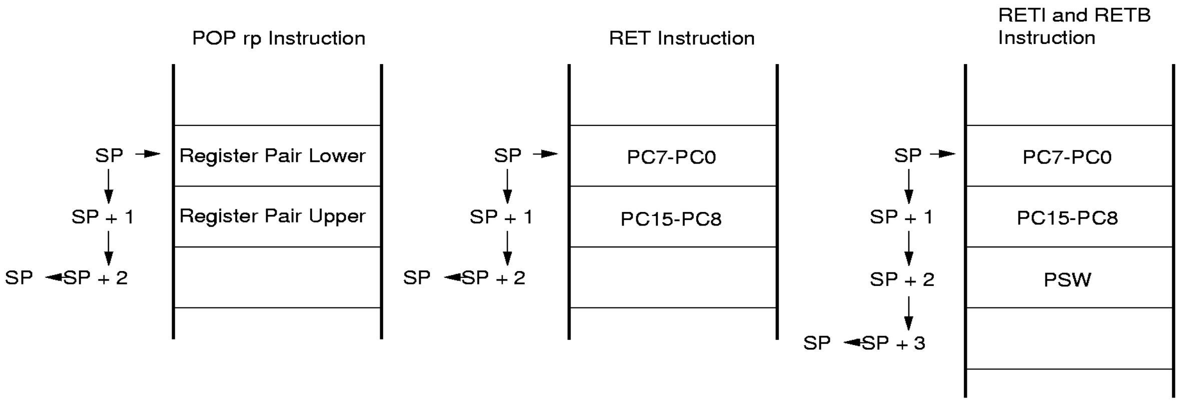


Figure 3-19. Data to be Restored from Stack Memory



3.2.2 General registers

A general register is mapped at particular addresses (FEE0H to FEFFH) of the data memory. It consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L and H).

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

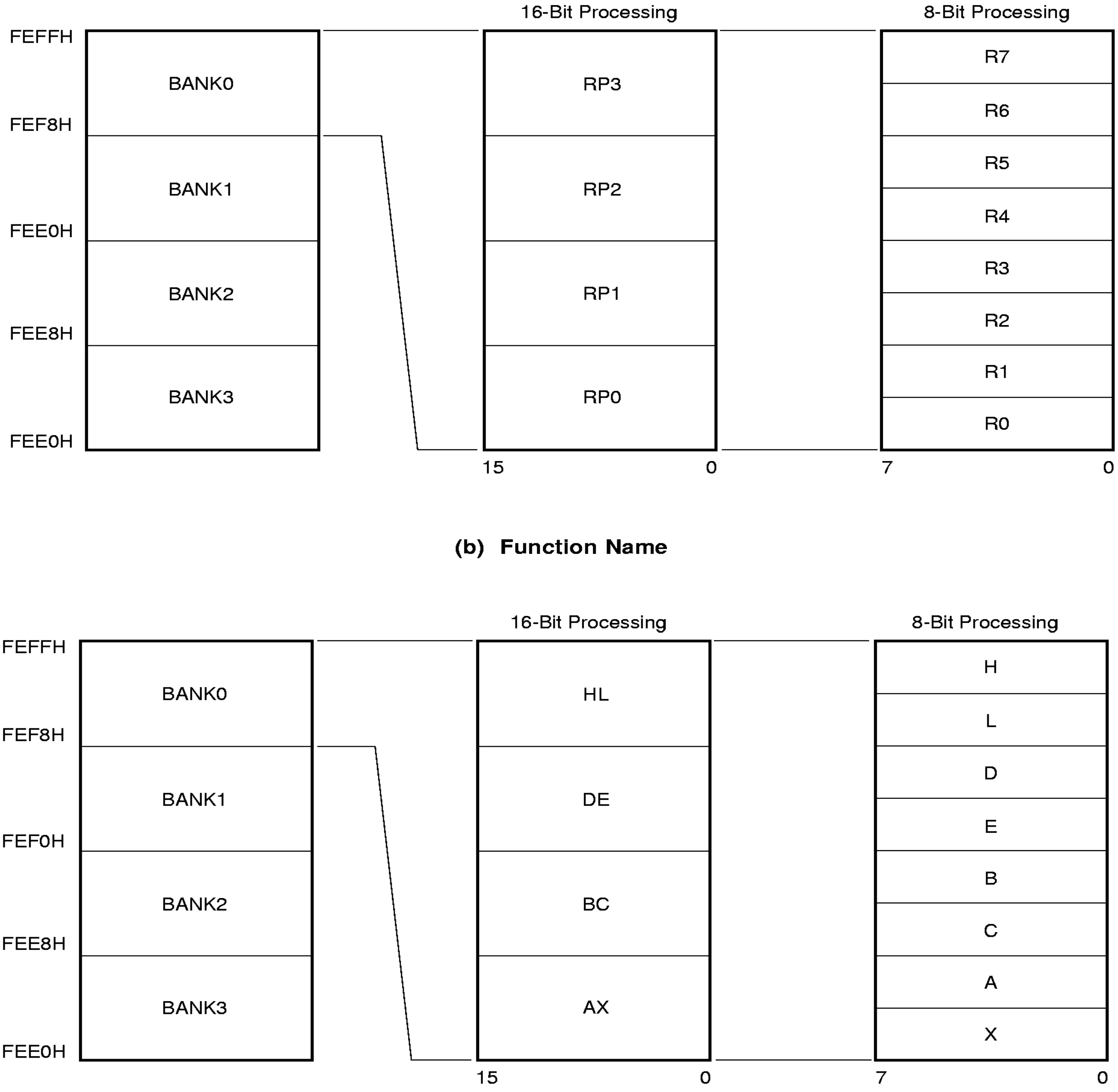
They can be written in terms of function names (X, A, C, B, E, D, L, H, AX, BC, DE and HL) and absolute names (R0 to R7 and RP0 to RP3).

Register banks to be used for instruction execution are set with 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 Address 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 3-20. Configuration of General Register



(a) Absolute Name

(b) Function Name

3.2.3 Special Function Registers (SFR)

Unlike a general register, each special function register has special functions.

It is allocated in the FF00H to FFFFH area.

The special function register can be manipulated like the general register, with the operation, transfer and bit manipulation instructions. Manipulatable bit units, 1, 8 and 16, depend on the special function register type.

Each manipulation bit 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 meaning of items in the table is as follows.

- **Symbol**
  This is a symbol to indicate an address of the special function register.
  These symbols are reserved for the DF178124 and RA78K/0, and defined by header file sfrbit.h for the CC78K/0.
  They can be written as instruction operands when the RA78K/0, ID78K0, or SD78K/0 is used.
- **R/W**
  Indicates whether the corresponding special function register can be read or written.

R/W	:	Read/write enable
R	:	Read only
R&Reset	:	Read only (reset to 0 when read)
W	:	Write only

- **Bit units for manipulation**
  ○ indicates manipulatable bit units 1, 8, and 16. – indicates the bit units that cannot be manipulated.
- **At reset**
  Indicates each register status upon reset. The values of special function registers whose addresses are not shown in the table are undefined at reset.

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

Address	Special Function Register (SFR) Name	Symbol	R/W	Bit Units for Manipulation			At Reset
				1 bit	8 bits	16 bits	
FF00H	Port 0	P0	R/W	○	○	—	00H
FF01H	Port 1	P1	R	○	○	—	
FF03H	Port 3	P3	R/W	○	○	—	
FF04H	Port 4	P4		○	○	—	
FF05H	Port 5	P5		○	○	—	
FF06H	Port 6	P6		○	○	—	
FF07H	Port 7	P7		○	○	—	
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		○	○	—	
FF1AH	IIC shift register 0	IIC0		—	○	—	Undefined
FF1BH	POC status register	POCS	R&Reset	—	○	—	Retained[Note 2]
FF20H	Port mode register 0	PM0	R/W	○	○	—	FFH
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		○	○	—	
FF2CH	Port mode register 12	PM12		○	○	—	
FF34H	Pull-up resistor option register 4	PU4		○	○	—	00H
FF41H	BEEP0 clock select register 0	BEEPCL0		○	○	—	
FF42H	Watchdog timer clock select register	WDCS		○	○	—	
FF43H	IIC transfer select register 0	IICCL0		○	○	—	
FF48H	External interrupt rising edge enable flag	EGP		○	○	—	
FF49H	External interrupt falling edge enable flag	EGN		○	○	—	
FF5AH	Asynchronous serial interface mode register 0[Note 3]	ASIM0		○	○	—	
FF5BH	Asynchronous serial interface status register 0[Note 3]	ASIS0	R	—	○	—	
FF5CH	Baud rate generator control register 0[Note 3]	BRGC0	R/W	—	○	—	

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 at reset by power-ON clear. This register is not reset by the $\overline{\text{RESET}}$ pin or watchdog timer.

3. μPD178F124 only.

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

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

Address	Special Function Register (SFR) Name		Symbol		R/W	Bit Units for Manipulation			At Reset
						1 bit	8 bits	16 bits	
FF5DH	Transmit shift register 0[Note 1]		TXS0		W	—	○	—	FFH
	Receive buffer register 0[Note 1]		RXB0		R	—	○	—	
FF60H	IIC status register 0		IICS0		R/W	○	○	—	00H
FF61H	IIC control register 0		IICC0			○	○	—	
FF62H	Slave address register 0		SVA0			—	○	—	Undefined
FF6EH	Serial I/O shift register 3		SIO3			—	○	—	
FF6FH	Serial operating mode register 3		CSIM3			○	○	—	00H
FF80H	8-bit timer compare register 50		CR50			—	○	—	Undefined
FF81H	8-bit timer 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
FFA2H	PLL unlock F/F judge register		PLLUL		R&Reset	○	○	—	Retained[Note 2]
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			○	○	—	00H[Note 3]
FFABH	IF counter gate judge register		IFCJG		R	○	○	—	00H
FFACH	IF counter control register		IFCCR		W	○	○	—	
FFAEH	IF counter register		IFCR	IFCRL	R	—	—	○	
FFAFH				IFCRH		—	—		
FFB1H	EEPROM write control register[Note 4]		EEWC		R/W	○	○	—	

Notes 1. μPD178F124 only.

2. Undefined by power-on clear reset only.

3. Though the initial value of the DTS system clock select register (DTSCK) is 00H, be sure to set this register to 01H before using it.

4. μPD178122, 178123, 178124, 128F124 only.

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

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

Address	Special Function Register (SFR) Name	Symbol		R/W	Bit Units for Manipulation			At Reset
					1 bit	8 bits	16 bits	
FFD0H | FFDFH	External access area[Note 1]			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[Note 2]	IF1L			○	○	—	
FFE4H	Interrupt mask flag register 0L	MK0	MK0L		○	○	○	FFH
FFE5H	Interrupt mask flag register 0H		MK0H		○	○		
FFE6H	Interrupt mask flag register 1L[Note 2]	MK1L			○	○	—	
FFE8H	Priority specification flag register 0L	PR0	PR0L		○	○	○	
FFE9H	Priority specification flag register 0H		PR0H		○	○		
FFEAH	Priority specification flag register 1L[Note 2]	PR1L			○	○	—	
FFF0H	Memory size select register	IMS			○	○	—	CFH[Note 3]
FFF4H	Internal extension RAM size select register	IXS			○	○	—	0CH[Note 4]
FFF9H	Watchdog timer mode register	WDTM			○	○	—	00H
FFFAH	Oscillation stabilization time select register	OSTS			○	○	—	04H
FFFBH	Processor clock control register	PCC			○	○	—	

Notes 1. The external access area cannot be accessed by means of SFR addressing. Use direct addressing to access this area.

2. μPD178F124 only

3. The initial value of the memory size select register (IMS) is CFH. Set the values of these registers of each model as follows:

Part Number	IMS
μPD178022, 178122	44H
μPD178023, 178123	C6H
μPD178024, 178124	C8H
μPD178F124	Value equivalent to mask ROM model

4. Do not assign a value other than the value at reset to the internal extension RAM size select register (IXS).

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

3.3 Instruction Address Addressing

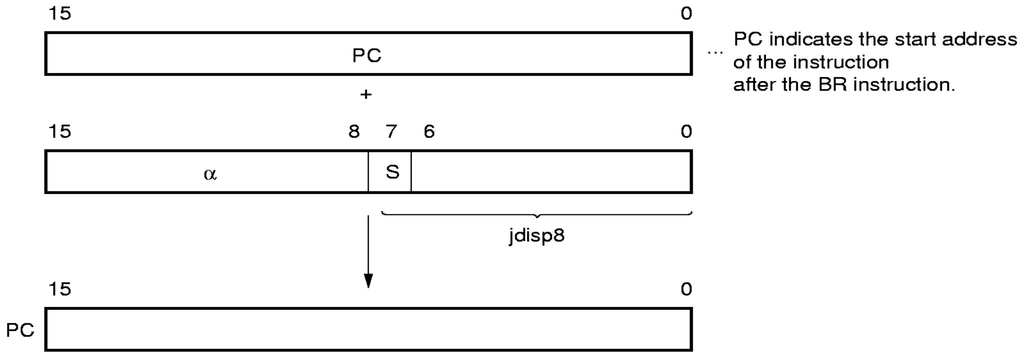
An instruction address is determined by program counter (PC) contents, and the contents is normally incremented (+1 for each byte) automatically according to the number of bytes of an instruction to be fetched each time another instruction is executed. When a branch instruction is executed, the branch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

3.3.1 Relative addressing

[Function]

The value obtained by adding 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]



... PC indicates the start address of the instruction after the BR instruction.

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

3.3.2 Immediate addressing

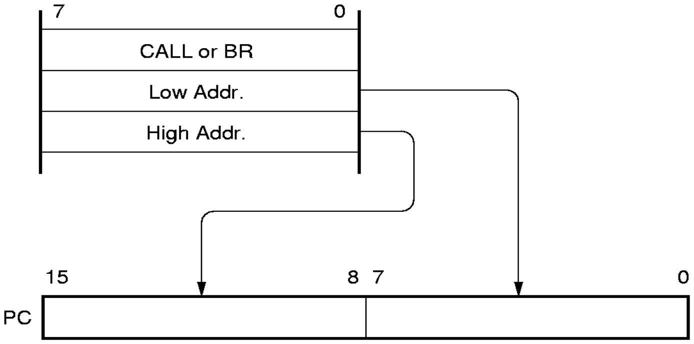
[Function]

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

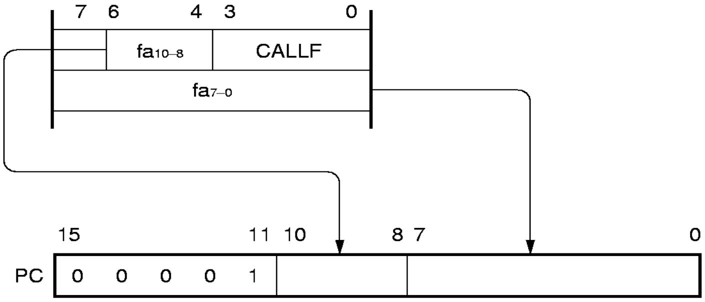
This function is carried out when the CALL !addr16 or BR !addr16 or CALLF !addr11 instruction is executed. The CALL !addr16 and BR !add16 instructions can be used to branch to any location in the memory. The CALLF !addr11 instruction is used to branch to the area between 0800H through 0FFFH.

[Illustration]

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



In the case of CALLF !addr11 instruction



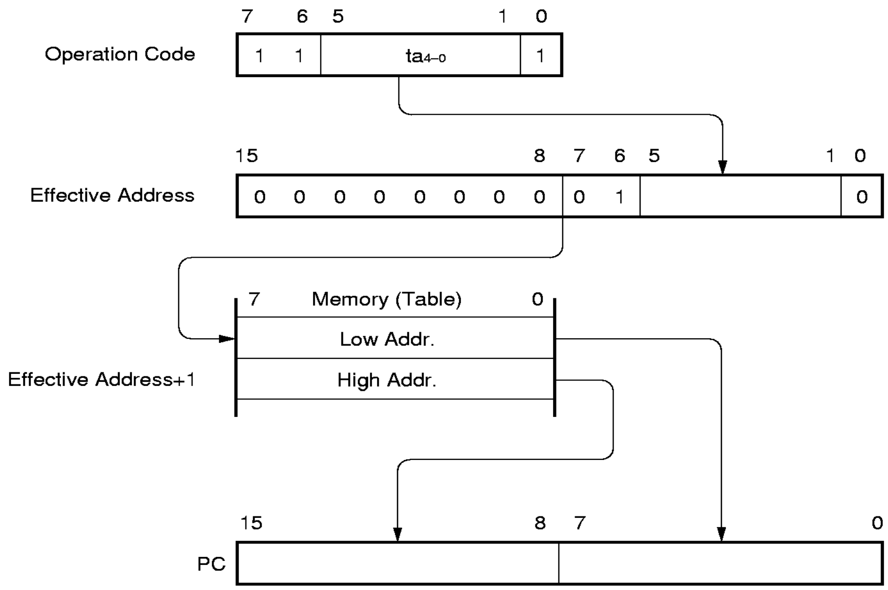
3.3.3 Table indirect addressing

[Function]

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 through 7FH, and can be used to branch to any location in the memory.

[Illustration]



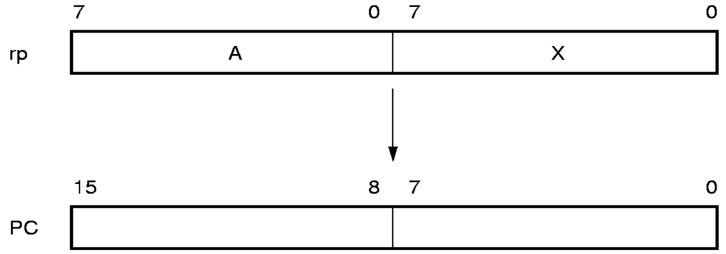
3.3.4 Register addressing

[Function]

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 various methods are available to specify the register and memory (addressing) which undergo manipulation during instruction execution.

3.4.1 Implied addressing

[Function]

The register which functions as an accumulator (A and AX) in the general register is automatically addressed (implied). Of the μPD178024 and 178124 subseries instruction words, the following instructions employ implied addressing.

Instruction	Register To Be 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 which 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 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 code (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]

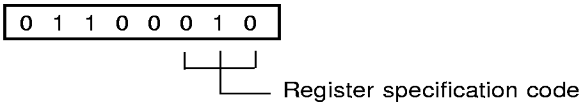
Symbol	Description
r	X, A, C, B, E, D, L, H
rp	AX, BC, DE, HL

'r' and 'rp' can be written with 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]

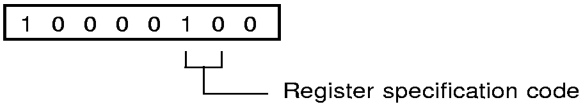
MOV A, C; when selecting C register as r

Operation code



INCW DE; when selecting DE register pair as rp

Operation code



3.4.3 Direct addressing

[Function]

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

[Operand format]

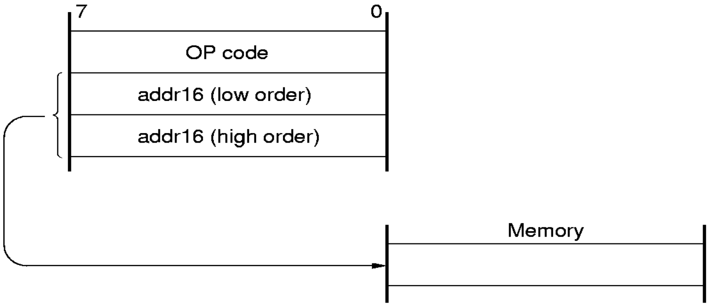
Symbol	Description
addr16	Label or 16-bit immediate data

[Example]

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

Operation code	1 0 0 0 1 1 1 0	Op code
	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. This addressing is applied to the fixed 256-byte space FE20H to FF1FH. An internal RAM and a special-function register (SFR) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.

The SFR area (FF00H to FF1FH) where short direct addressing is applied is one part of all the SFR areas. 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]

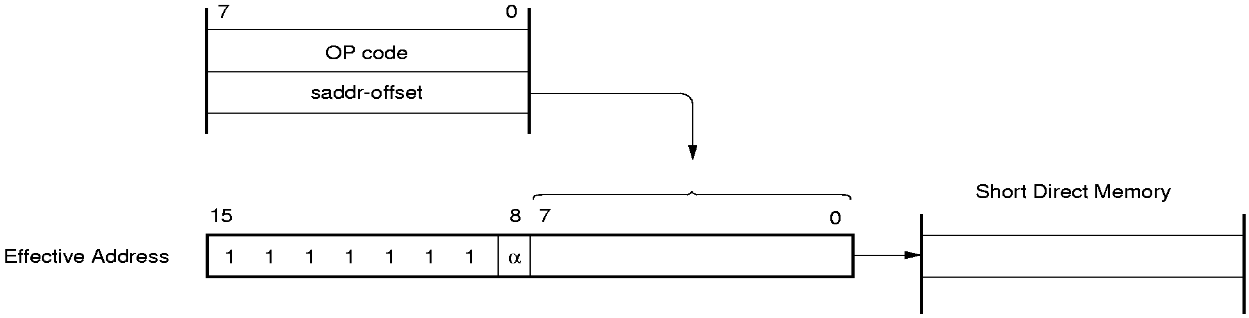
Symbol	Description
saddr	Label of FE20H to FF1FH immediate data
saddrp	Label of FE20H to FF1FH immediate data (even address only)

[Example]

MOV 0FE30H, #50H; when setting saddr to FE30H and immediate data to 50H

Operation code	0 0 0 1 0 0 0 1	Op code
	0 0 1 1 0 0 0 0	30H (saddr-offset)
	0 1 0 1 0 0 0 0	50H (immediate data)

[Illustration]



When 8-bit immediate data is 20H to FFH, α = 0
When 8-bit immediate data is 00H to 1FH, α = 1

3.4.5 Special Function Register (SFR) addressing

[Function]

The 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 SFR 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 address only)

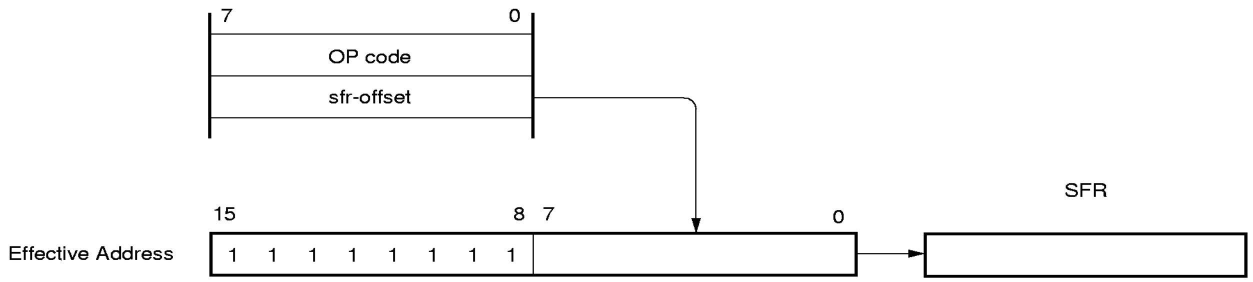
[Example]

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]

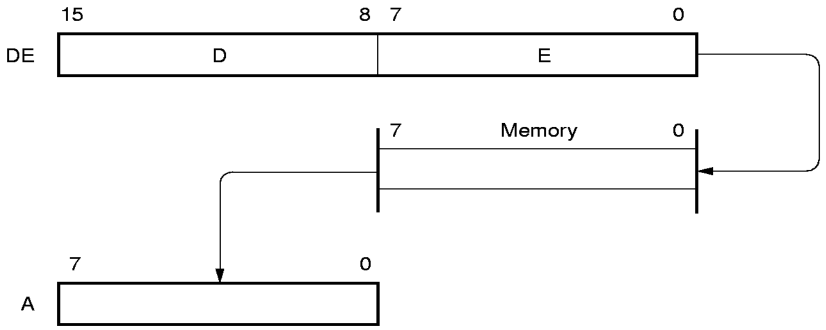
[Example]

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 carried out for all the memory spaces.

[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

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 carried out for all the memory spaces.

[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`

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 enables 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`

CHAPTER 4 EEPROM (μPD178122, 178123, 178124, 178F124 only)

4.1 Functions of EEPROM

The μPD178124 subseries (μPD178122, 178123, 178124, and 178F124) products contain on-chip 128 × 8-bit EEPROM (Electrically Erasable PROM) in addition to internal high-speed RAM, as data memory.

EEPROM differs from ordinary RAM in that its contents are saved even after power is cut off. Moreover, unlike EPROM, its contents can be erased electrically without using UV light. For this reason, it is suitable for applications where the setting values of the odometer and trip meter on the dash board are saved, etc.

EEPROM can be manipulated with 8-bit memory manipulation instructions.

The EEPROM contained on-chip in the μPD178124 subseries has the following features.

(1) Written contents are saved even when the power is cut off.

(2) Can be manipulated with 8-bit memory manipulation instructions in the same way as ordinary RAM.

(3) Erasure and writing is performed in the time set with EWCS0 and EWCS1 (EEPROM write control register (EEWC) bits 4 and 5) (refer to **Figure 4-2**). Therefore, the write time control software load is reduced. Moreover, during writing, instructions other than instructions related to EEPROM writing and reading can also be executed. For details about EEPROM characteristics, refer to Data Sheet (to be prepared).

(4) When write is completed, interrupt request signal (INTEE) is issued.

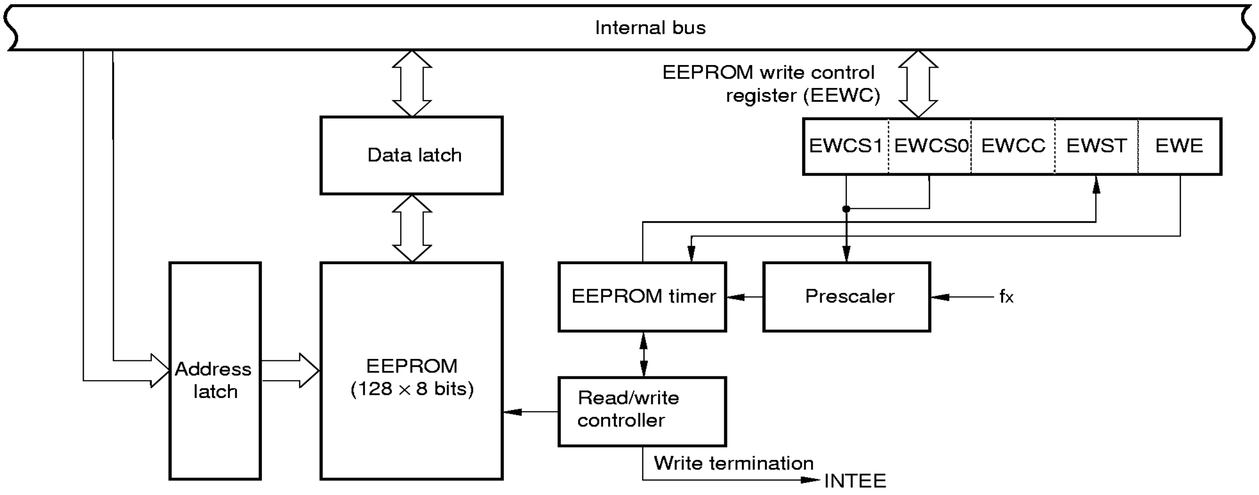
(5) The write enabled/disabled status can be checked with EWST (EEPROM write control register (EEWC) bit 1).

4.2 Configuration of EEPROM

EEPROM is composed of EEPROM itself and a control area.

The control area consists of the EEPROM write control register (EEWC) that controls EEPROM writing, and an area that generates an interrupt request signal (INTEE) upon detecting write termination.

Figure 4-1. Block Diagram of EEPROM



4.3 EEPROM Control Register

EEPROM is controlled with the EEPROM write control register (EEWC).
EEWC is set with either a 1-bit or 8-bit memory manipulation instruction.
Reset input sets this register to 00H.

Figure 4-2. Format of EEPROM Write Control Register (EEWC)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
EEWC	0	0	EWCS1	EWCS0	0	EWCC	EWST	EWE	FFB1H	00H	R/W

EWCS1	EWCS0	EEPROM Write Time (fx = 4.5 MHz)
0	1	3.87 ms
Others		Setting prohibited

EWCC	EEPROM Operation Control
0	Operating mode
1	EEPROM stop

EWST	EEPROM Write Status
0	Not currently writing to EEPROM (EEPROM write/read is enabled. However, when EWE = 0, write is disabled)
1	Currently writing to EEPROM (EEPROM write/read is disabled)

EWE	EEPROM Write Operation Control
0	EEPROM write disabled
1	EEPROM write enabled

Cautions

1. **When EWE is cleared (to 0) during EEPROM writing, writing is immediately interrupted. Data that was being written becomes undefined. Be sure to clear EWE before stopping the system clock during the write period. This means that the contents of EEPROM cannot be read for a period of 46 instructions after EWCC has been cleared.**
   0.44 μs (minimum instruction execution time) × 46 instructions ≒ 20.2 μs
2. **After EWCC is cleared (to 0), set a wait time of 20 μs or more by software to read EEPROM contents.**
3. **Be sure to check that EWST is 0 before performing EEPROM access.**
4. **Bits 3, 6, and 7 must be set to 0.**
5. **In order to set the low current consumption mode, set EWCC to 1 in the standby state.**

4.4 EEPROM Reading

Reading of EEPROM data is performed with the following procedure.

<1> Check that EWST (EEPROM write control register (EEWC) bit 1) is 0 (EEPROM writing is not in progress).

<2> Execute read instruction.

Cautions 1. Before reading, be sure to check that EWST is 0. If an EEPROM read instruction is executed during EEPROM write, read values are undefined.

2. If reading EEPROM contents immediately after changing EWCC (EEPROM write control register (EEWC) bit 2) from 1 to 0, set a wait time of at least 20 µs by software.
If no wait time is set, the correct values cannot be read.

Example Insertion of NOP instructions to set wait time of 20 µs or more.

```
CLR1 EWCC
NOP           }
NOP           } Insert NOP instructions to secure a wait time of 20 µs or more.
MOV A,!0F800H
```

4.5 EEPROM Writing

Data writing to EEPROM is performed with the following procedure.

<1> Check that EWST (EEPROM write control register (EEWC) bit 1) is 0 (EEPROM writing is not in progress).
<2> Set the write time with EWCS0 and EWCS1 (EEWC bits 4 and 5).
<3> Set EWE (EEWC bit 0) to 1 (EEPROM writing enabled).
<4> Execute write instruction.

If performing several write operations in succession, perform the next write operation after the current write operation has been completed. The following methods can be used for write termination and time control.

(1) Method using write termination interrupt request (INTEE)

After writing 1 data, wait for generation of write termination interrupt request while processing other than write is performed. When write termination interrupt request is generated, start next write operation.

(2) Method using write status flag (EWST)

Poll EWST (EEPROM write control register (EEWC) bit 1), and wait for EWST to become 0. When EWST becomes 0, start the next write operation.

4.6 EEPROM Control-Related Interrupt

EEPROM write termination interrupt request (INTEE) is generated from EEPROM.

INTEE is an interrupt request generated upon termination of EEPROM writing.

This interrupt request is generated when the time set with EWCS0 and EWCS1 (EEPROM write control register (EEWC) bits 4 and 5) has elapsed.

When this interrupt request is generated, data writing to EEPROM is terminated, indicating that writing of the next data is enabled.

4.7 Notes on EEPROM Writing

The following shows cautions of EEPROM write.
Before performing EEPROM write, be sure to read the following cautions.

(1) Before writing, be sure to check that EWST (EEPROM write control register (EEWC) bit 1) is 0. If executing another write instructions during EEPROM writing, the instruction executed last will be ignored.

(2) For write time, refer to **Figure 4-2**.

(3) If performing several write operations in succession, be sure to wait until the current write operation is completed before starting the next one.

(4) Even if the mode changes to HALT mode during EEPROM writing, writing is continued.

(5) If the mode changes to STOP mode during EEPROM writing, the data being written becomes undefined. If this STOP mode is cancelled by interrupt request, a write termination interrupt request (INTEE) is generated after the STOP mode has been cancelled. If you want to set the STOP mode after normally terminating write processing, check that write processing has ended using any of the available methods (refer to **4.5 EEPROM Writing**), then set the STOP mode.

(6) If you want to read the EEPROM contents immediately after changing EWCC (EEPROM write control register (EEWC) bit 2) from 1 to 0, set a wait time of 20 μs or more by software. If no wait time is set, the data read will not be correct.

CHAPTER 5 PORT FUNCTIONS

5.1 Functions of Port

The μPD178024 and 178124 subseries units incorporate six input ports, three output ports and 53 input/output ports. Figure 5-1 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 input/output pins.

Figure 5-1. Port Types

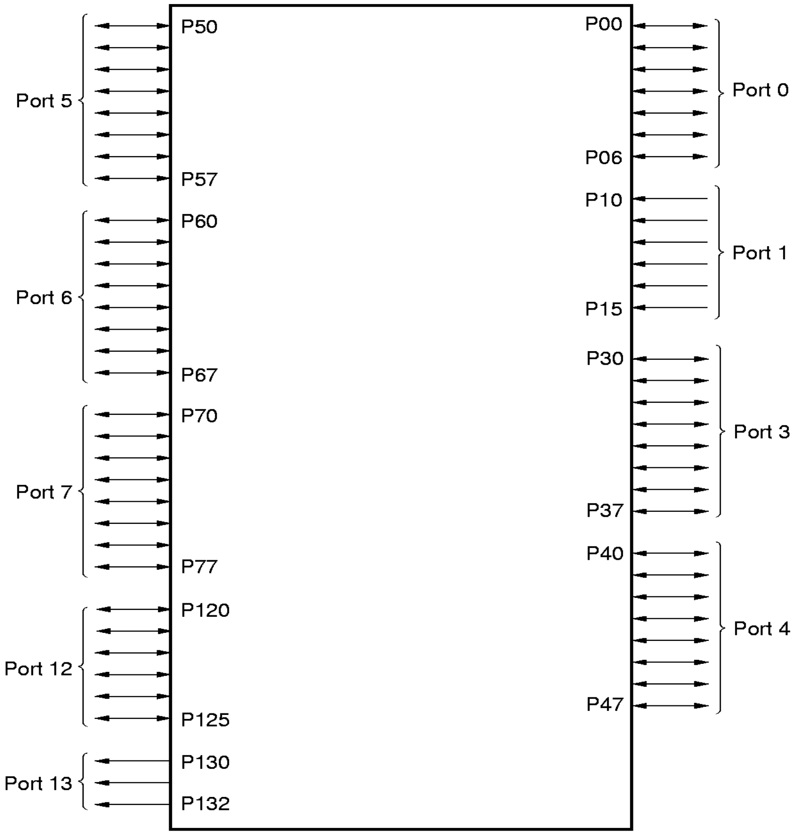


Table 5-1. Port Function

Pin Name	I/O	Function	Shared by:
P00-P04	I/O	Port 0. 7-bit I/O port. Can be set in input or output mode in 1-bit units.	INTP0-INTP4
P05, P06			–
P10-P15	Input	Port 1. 6-bit input port.	ANI0-ANI5
P30-P32	I/O	Port 3.	–
P33		8-bit I/O port.	TI50
P34		Can be set in input or output mode in 1-bit units.	TI51
P35			–
P36			BEEP0
P37			–
P40-47	I/O	Port 4. 8-bit I/O port. Can be set in input or output mode in 1-bit units. When using as input port, internal pull-up resistors can be specified in software. Interrupt function by key input is provided.	–
P50-P57	I/O	Port 5. 8-bit I/O port. Can be set in input or output mode in 1-bit units.	–
P60-P67	I/O	Port 6. 8-bit I/O port. Can be set in input or output mode in 1-bit units.	–
P70	I/O	Port 7.	SI3
P71		8-bit I/O port.	SO3
P72		Can be set in input or output mode in 1-bit units.	$\overline{\text{SCK3}}$
P73			–
P74			$\overline{\text{RXD0}}$[Note]
P75			$\overline{\text{TXD0}}$[Note]
P76			SDA0
P77			SCL0
P120-P125	I/O	Port 12. 6-bit I/O port. Can be set in input or output mode in 1-bit units.	–
P130	Output	Port 13.	TO50
P131		3-bit output port.	TO51
P132		N-ch open-drain output port (12 V withstand)	–

Note μPD178F124 only.

5.2 Configuration of Port

A port consists of the following hardware:

Table 5-2. Configuration of Port

Item	Configuration
Control register	Port mode register (PMm: m = 0, 3-7, 12)
Port	Total: 62 ports (6 inputs, 3 outputs, 53 I/Os)

5.2.1 Port 0

Port 0 is a 7-bit input/output port with output latch. P00 to P06 pins can specify the input mode/output mode in 1-bit units with the port mode register 0 (PM0).

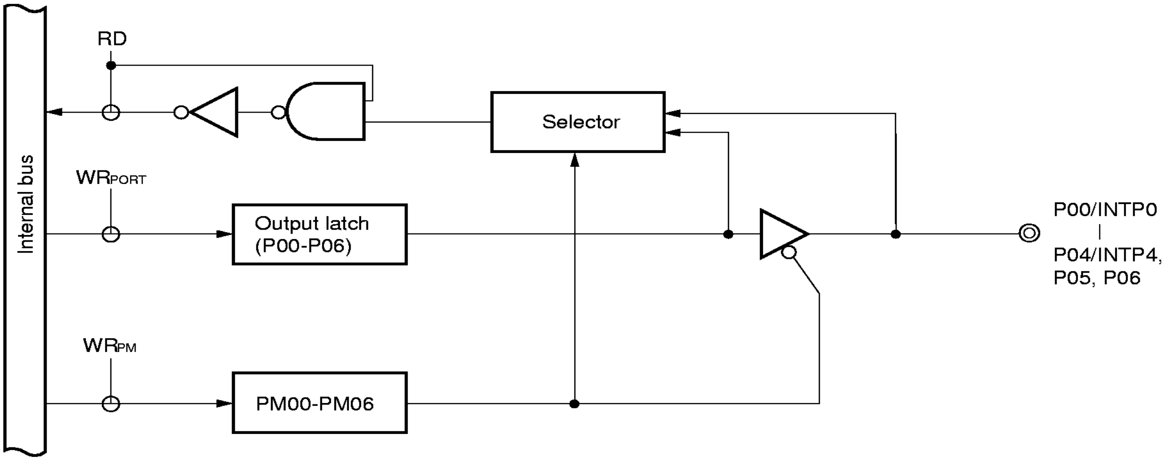
Alternate functions include external interrupt request input.

Reset input sets port 0 to the input mode.

Figure 5-2 shows the block diagram of port 0.

Caution Because port 0 also serves for external interrupt request input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. Thus, when the output mode is used, set the interrupt mask flag to 1.

Figure 5-2. Block Diagram of P00 to P06



PM : Port mode register
RD : Port 0 read signal
WR : Port 0 write signal

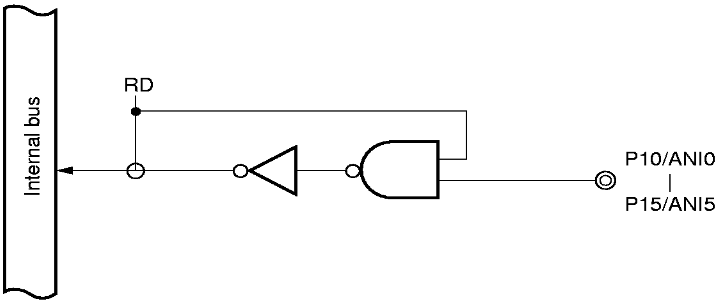
5.2.2 Port 1

Port 1 is a 6-bit input port.

Alternate functions include an A/D converter analog input.

Figure 5-3 shows the block diagram of port 1.

Figure 5-3. Block Diagram of P10 to P15



RD : Port 1 read signal

5.2.3 Port 3

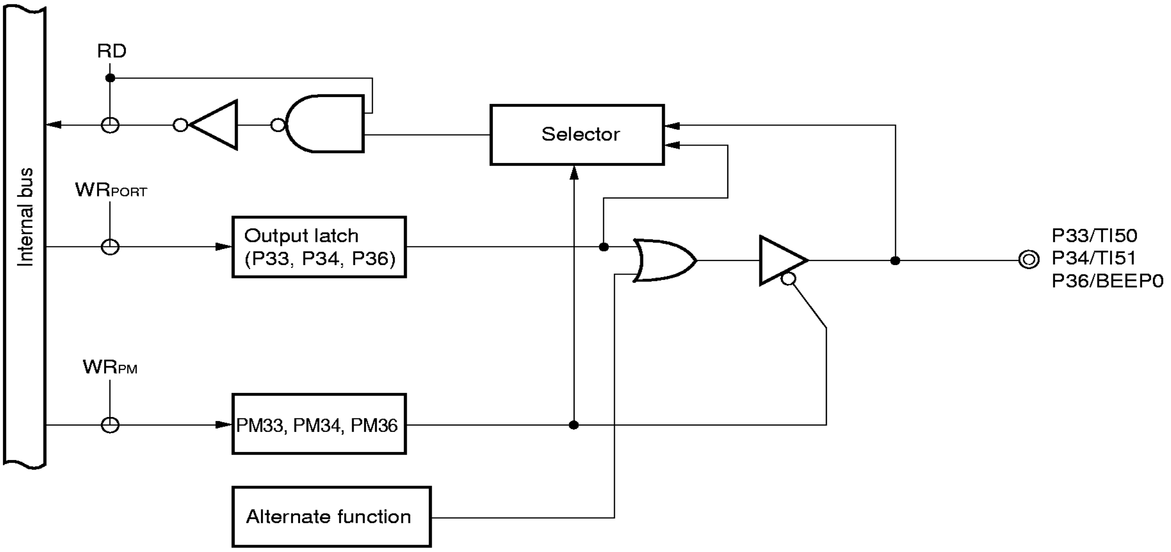
Port 3 is an 8-bit input/output port with output latch. P30 to P37 pins can specify the input mode/output mode in 1-bit units with the port mode register 3 (PM3).

Alternate functions include timer input and buzzer output.

Reset input sets port 3 to the input mode.

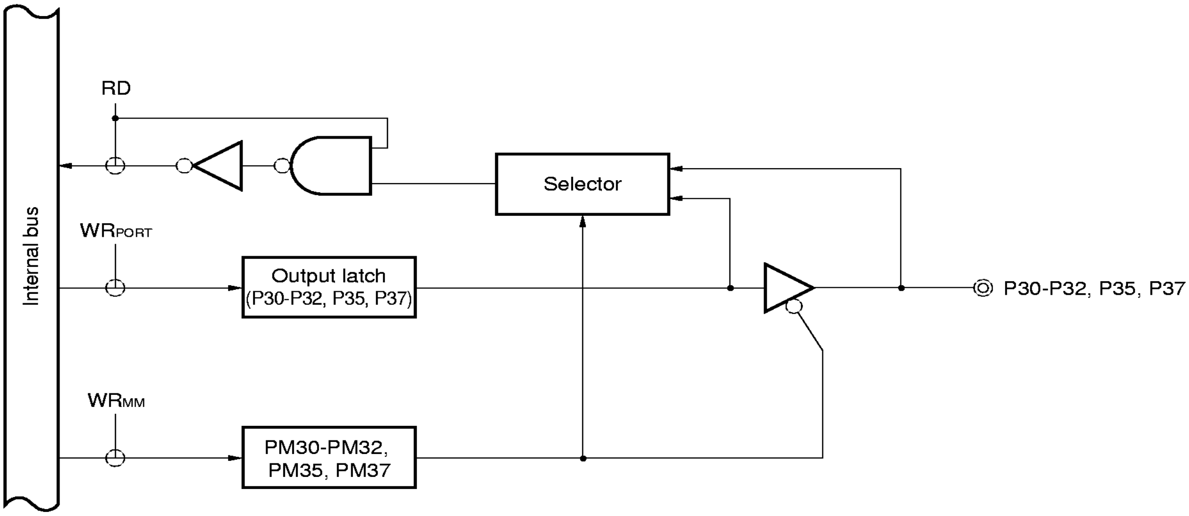
Figures 5-4 and 5-5 show the block diagrams of port 3.

Figure 5-4. Block Diagram of P33, P34, and P36



PM : Port mode register
RD : Port 3 read signal
WR : Port 3 write signal

Figure 5-5. Block Diagram of P30-P32, P35, and P37



PM : Port mode register
RD : Port 3 read signal
WR : Port 3 write signal

5.2.4 Port 4

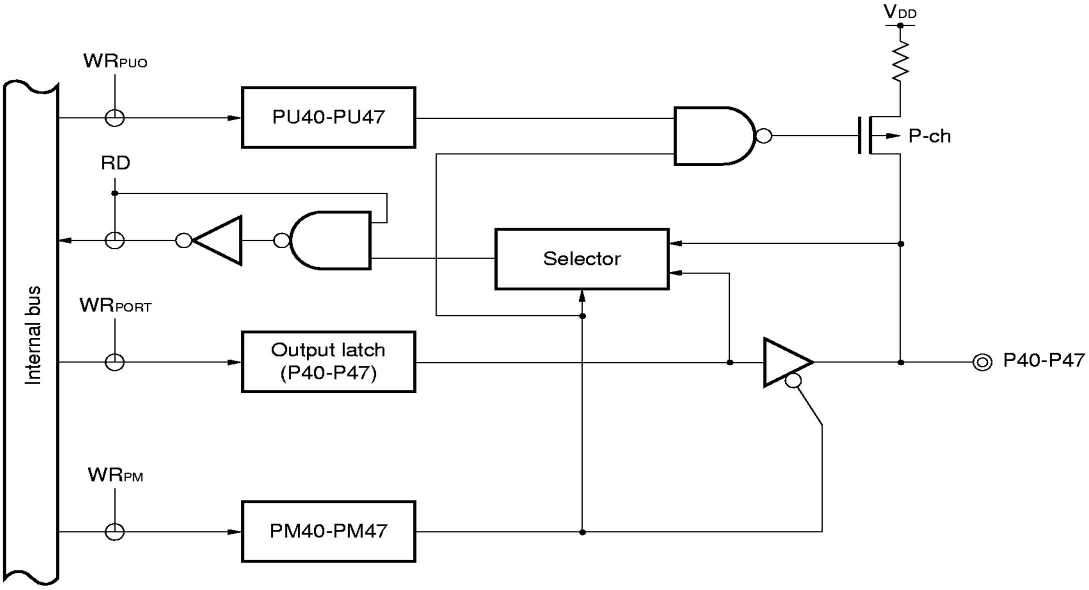
Port 4 is an 8-bit input/output port with output latch. The P40 to P47 pins can specify the input mode/output mode in 1-bit units with port mode register 4 (PM4). When the P40 to P47 pins are used as input ports, a pull-up resistor can be connected to them in 1-bit units with pull-up resistor option register 4 (PU4).

The interrupt request flag (KYIF) can be set to 1 by detecting key inputs. When using this function, be sure to set MEM register to 01H.

$\overline{\text{RESET}}$ input sets port 4 to input mode.

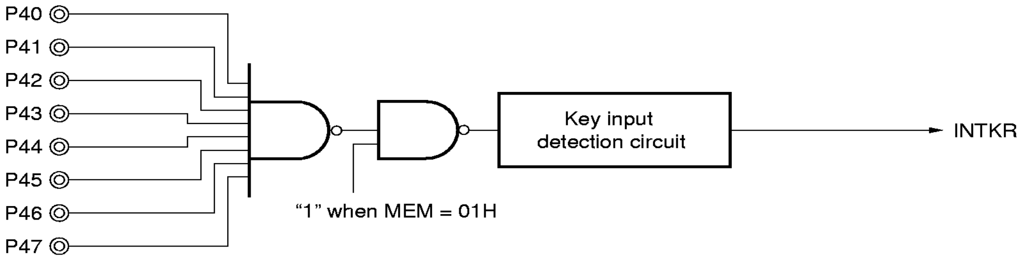
Figures 5-6 and 5-7 show a block diagram of port 4 and block diagram of the key input detection circuit, respectively.

Figure 5-6. Block Diagram of P40 to P47



PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 4 read signal
WR : Port 4 write signal

Figure 5-7. Block Diagram of Key Input Detection Circuit



Caution This register is valid only when MEM register is set to 01H.

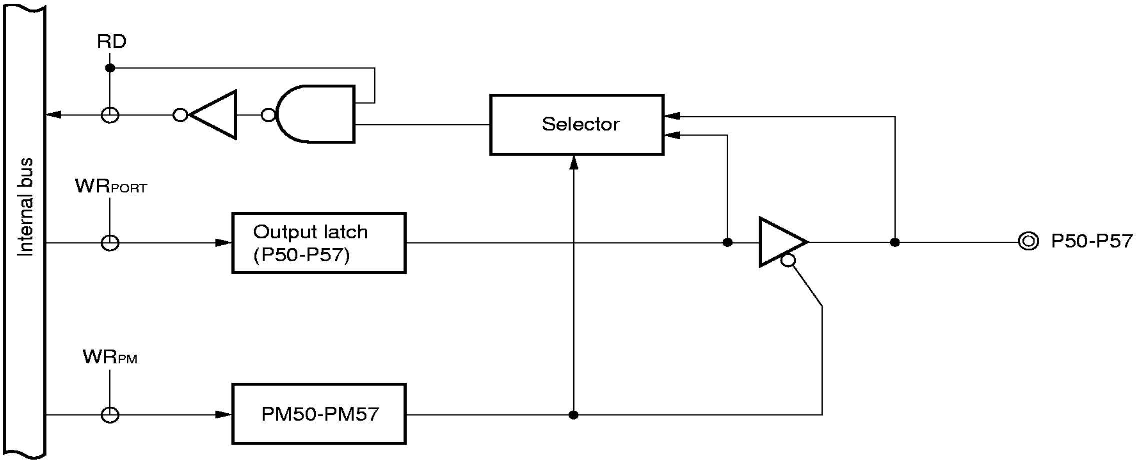
5.2.5 Port 5

Port 5 is an 8-bit input/output port with output latch. P50 to P57 pins can specify the input mode/output mode in 1-bit units with the port mode register 5 (PM5).

Reset input sets port 5 to the input mode.

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

Figure 5-8. Block Diagram of P50 to P57



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

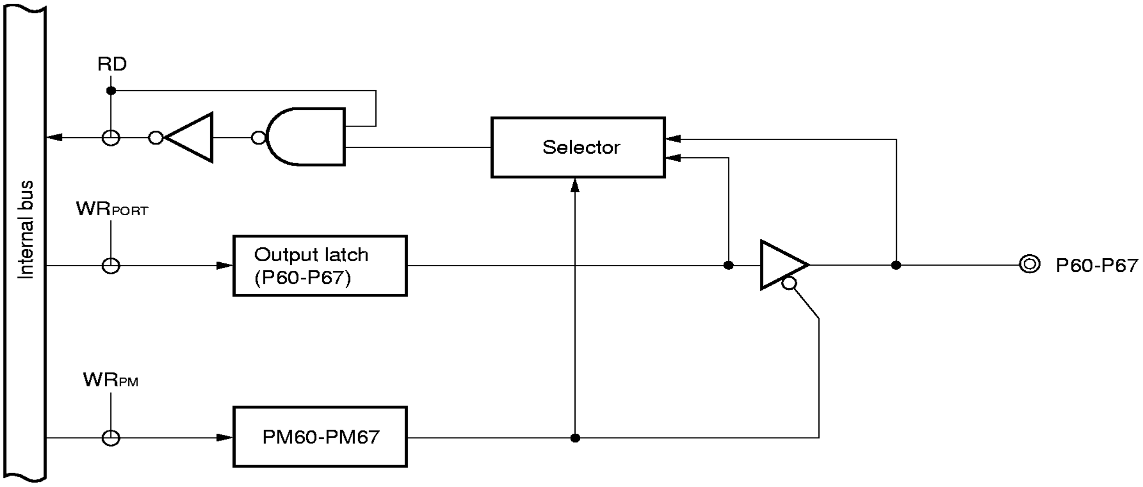
5.2.6 Port 6

Port 6 is an 8-bit input/output port with output latch. P60 to P67 pins can specify the input mode/output mode in 1-bit units with the port mode register 6 (PM6).

Reset input sets port 6 to the input mode.

Figure 5-9 shows the block diagram of port 6.

Figure 5-9. Block Diagram of P60 to P67



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

5.2.7 Port 7

This is an 8-bit input/output port with an output latch. Pins P70 through P77 can be set in the input or output mode in 1-bit units by using port mode register 7 (PM7).

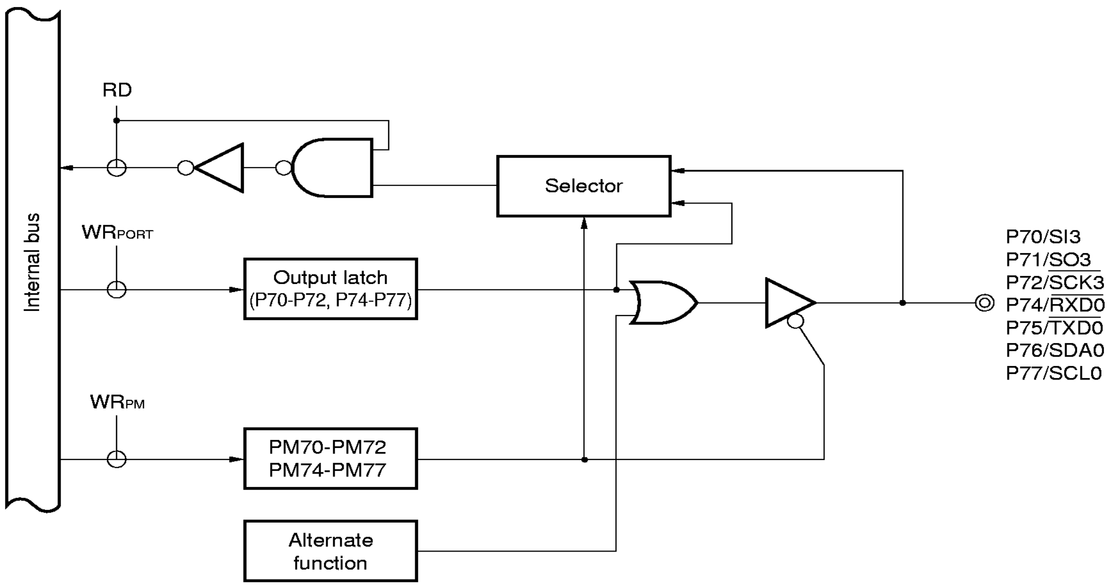
This port is also multiplexed with serial interface data input/output, clock input/output, and asynchronous interface data input/output pins[Note].

Reset input sets port 7 to the input mode.

Figures 5-10 and 5-11 show the block diagrams of port 7.

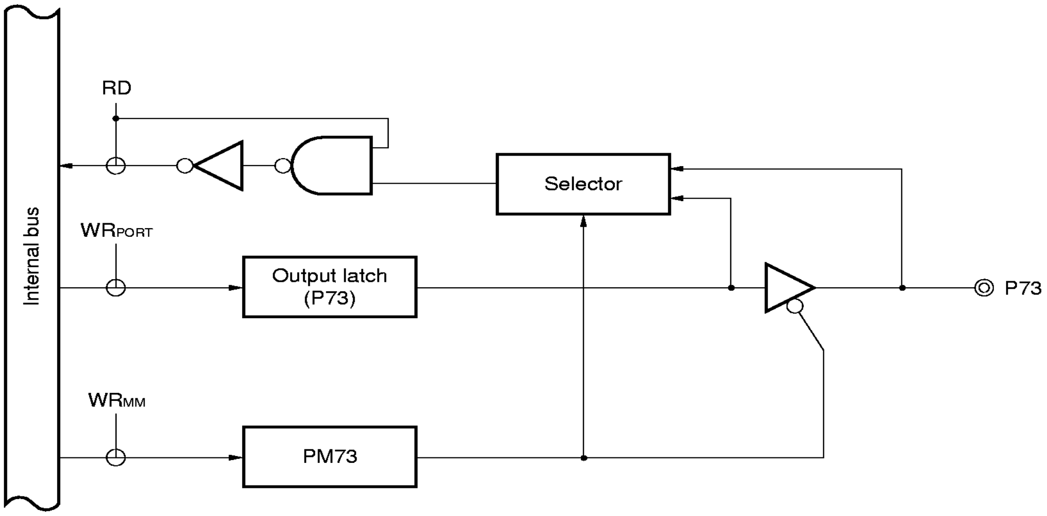
Note μPD178F124 only

Figure 5-10. Block Diagram of P70 to P72, P74 to P77



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

Figure 5-11. Block Diagram of P73



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

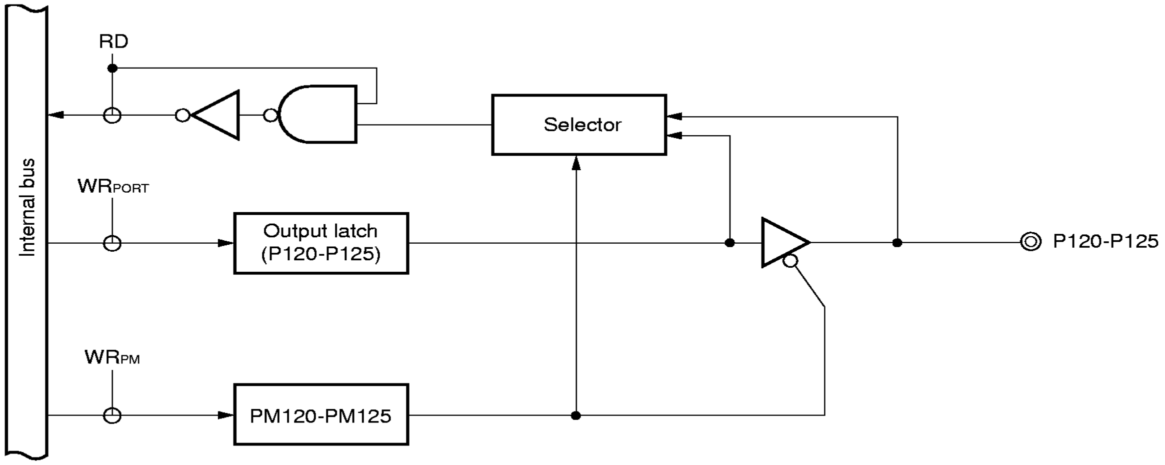
5.2.8 Port 12

Port 12 is a 6-bit input/output port with output latch. P120 to P125 pins can specify the input mode/output mode in 1-bit units with the port mode register 12 (PM12).

Reset input sets port 12 to the input mode.

Figure 5-12 shows the block diagram of port 12.

Figure 5-12. Block Diagram of P120 to P125



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

5.2.9 Port 13

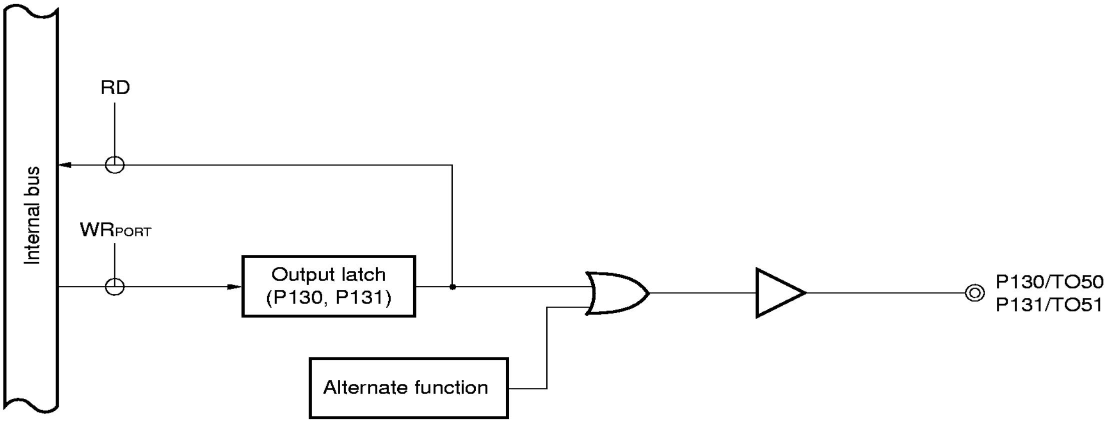
This is a 3-bit N-ch open-drain output port with an output latch.

The pins of this port are also used as timer output pins.

These pins are set in the general-purpose output port mode at reset.

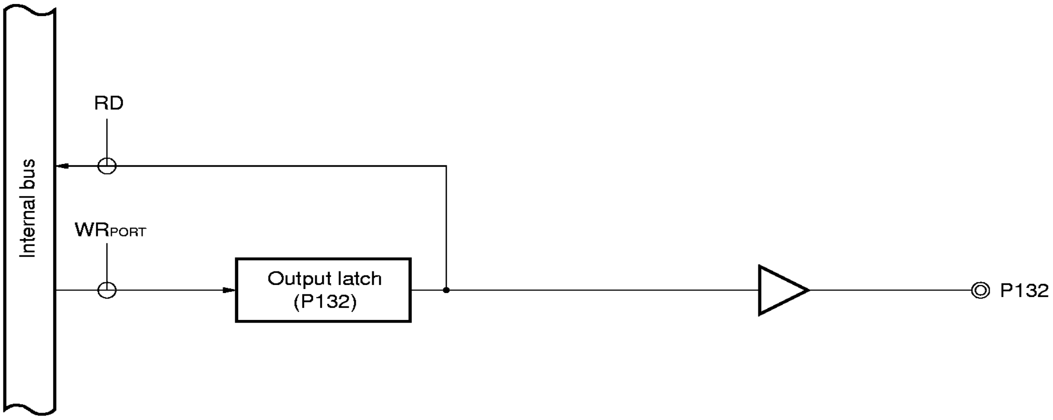
The port 13 block diagram is shown in Figures 5-13 and 5-14.

Figure 5-13. Block Diagram of P130 and P131



RD : Port 13 read signal
WR : Port 13 write signal

Figure 5-14. Block Diagram of P132



RD : Port 13 read signal
WR : Port 13 write signal

5.3 Port Function Control Registers

The following two types of registers control the ports.

- Port mode registers (PM0, PM3 to PM7, PM12)
- Pull-up resistor option register (PU4)

(1) Port mode registers (PM0, PM3 to PM7, PM12)

These registers are used to set port input/output in 1-bit units.

PM0 and PM3 to PM7, and PM12 are independently set with a 1-bit or 8-bit memory manipulation instruction.

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

When using a port pin as a shared function pin, set the values of port mode registers and output latch as shown in Table 5-3.

Cautions 1. P10 to P17 are input-only pins, and P100 to P132 are output-only pins.

2. As port 0 has an alternate function 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 the output mode is used, therefore, the interrupt mask flag should be set to 1 beforehand.

Table 5-3. Port Mode Register and Output Latch Settings When Using Alternate Functions

Pin Name	Alternate Functions		PM××	P××
	Name	Input/Output		
P00-P04	INTP0-INTP4	Input	1	×
P10-P15	ANI0-ANI5	Input	1	×
P33	TI50	Input	1	×
P34	TI51	Input	1	×
P36	BEEP0	Output	0	0
P70	SI3	Input	1	×
P71	SO3	Output	0	0
P72	$\overline{\text{SCK3}}$	Input	1	×
		Output	0	0
P74	$\overline{\text{RXD0}}$	Input	1	×
P75	$\overline{\text{TXD0}}$	Output	0	0
P76	SDA0	Input	1	×
		Output	0	0
P77	SCL0	Input	1	×
		Output	0	0
P130	TO50	Output	—	0
P131	TO51	Output	—	0

Caution When using the above alternate function pins as an output port, be sure to set the output latch (P××) to 0.

Remark × : don't care
PM×× : port mode register
P×× : output latch of port

Figure 5-15. Format of Port Mode Registers

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
PM0	1	PM06	PM05	PM04	PM03	PM02	PM01	1	FF20H	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
PM12	1	1	PM125	PM124	PM123	PM122	PM121	PM120	FF2CH	FFH	R/W

PMmn	Pmn Pin Input/Output Mode Selection (m=0, 3-7, 12 : n=0-7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

(2) Pull-up resistor option register 4 (PU4)

This register is used to specify the use of the internal pull-up resistors of port 4. Even if the use of the pull-up resistor is specified for a particular bit in PU4, the pull-up resistor can only be used if that bit is specified as being in the input mode. Bits specified as output mode cannot use the pull-up resistor regardless of the setting of the PU4.

PU4 can be set with a 1-bit or 8-bit memory manipulation instruction.

Reset sets PU4 to 00H.

Figure 5-16. Format of Pull-up Resistor Option Register 4 (PU4)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
PU4	PU47	PU46	PU45	PU44	PU43	PU42	PU41	PU40	FF34H	00H	R/W

PU4n	Selection of Internal Pull-up Resistor for P4n (n = 0-7)
0	Internal pull-up resistor not used
1	Internal pull-up resistor used

5.4 Port Function Operations

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

5.4.1 Writing to input/output 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 retained until data is written to the output latch again.

(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 retained until data is written to the output latch again.

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, on a port with a mixture of input and output pins, the output latch contents for pins specified as input are undefined except for the manipulated bit.

5.4.2 Reading from input/output 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.

5.4.3 Operations on input/output 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 retained until data is written to the output latch again.

(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, on 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 6 CLOCK GENERATOR

6.1 Functions of Clock Generator

The clock generator generates the clock to be supplied to the CPU and peripheral hardware. This system clock oscillator is connected to 4.5-MHz cystal resonator. At this time, 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 control circuit, PLL frequency synthesizer, and frequency counter.

Oscillation can be stopped by executing the STOP instruction.

Figure 6-1. Format of DTS System Clock Select Register (DTSCK)

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

DTSCK0	Selects System Clock
1	4.5 MHz
0	Setting prohibited

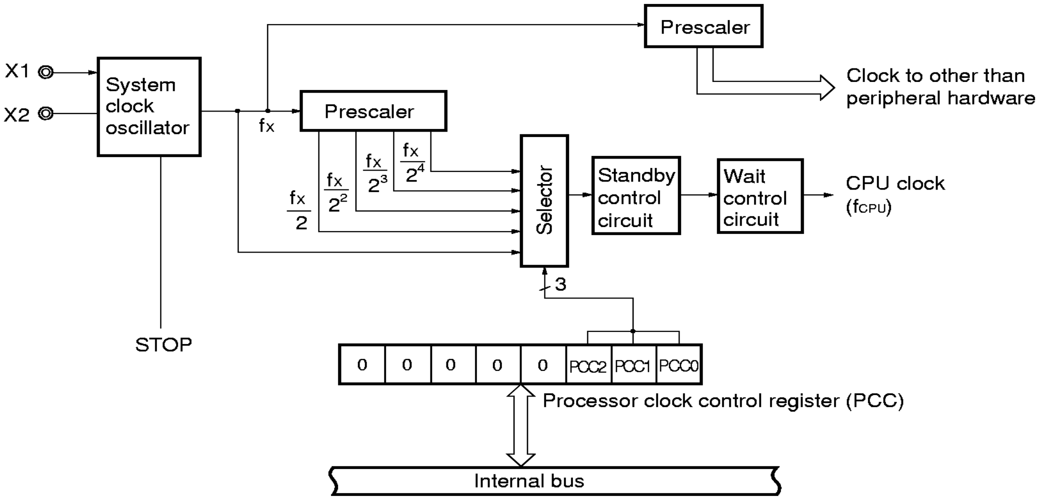
6.2 Configuration of Clock Generator

The clock generator consists of the following hardware.

Table 6-1. Configuration of Clock Generator

Item	Configuration
Control register	Processor clock control register (PCC)
Oscillator	System clock oscillator

Figure 6-2. Block Diagram of Clock Generator



6.3 Clock Generator Control Register

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

The PCC sets CPU clock.

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

Reset input sets the PCC to 04H.

Figure 6-3. Format of Processor Clock Control Register (PCC)

Symbol	7	6	5	4	3	2	1	0	Address	At 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
0	0	0	f_X (0.45 μs)
0	0	1	$f_X/2$ (0.89 μs)
0	1	0	$f_X/2^2$ (1.78 μs)
0	1	1	$f_X/2^3$ (3.56 μs)
1	0	0	$f_X/2^4$ (7.11 μs)
Other than above			Setting prohibited

Note Bits 3 to 7 are Read Only.

Remarks 1. f_X : System clock oscillation frequency

2. ( ) : Minimum instruction execution time: $2/f_{CPU}$ at f_X = 4.5 MHz operation

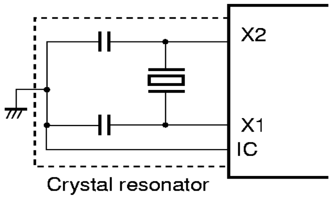
6.4 System Clock Oscillator

6.4.1 System clock oscillator

The system clock oscillator oscillates with a crystal resonator (4.5 MHz TYP.) connected to the X1 and X2 pins. Figure 6-4 shows an external circuit of the system clock oscillator.

Figure 6-4. External Circuit of System Clock Oscillator

Crystal oscillation

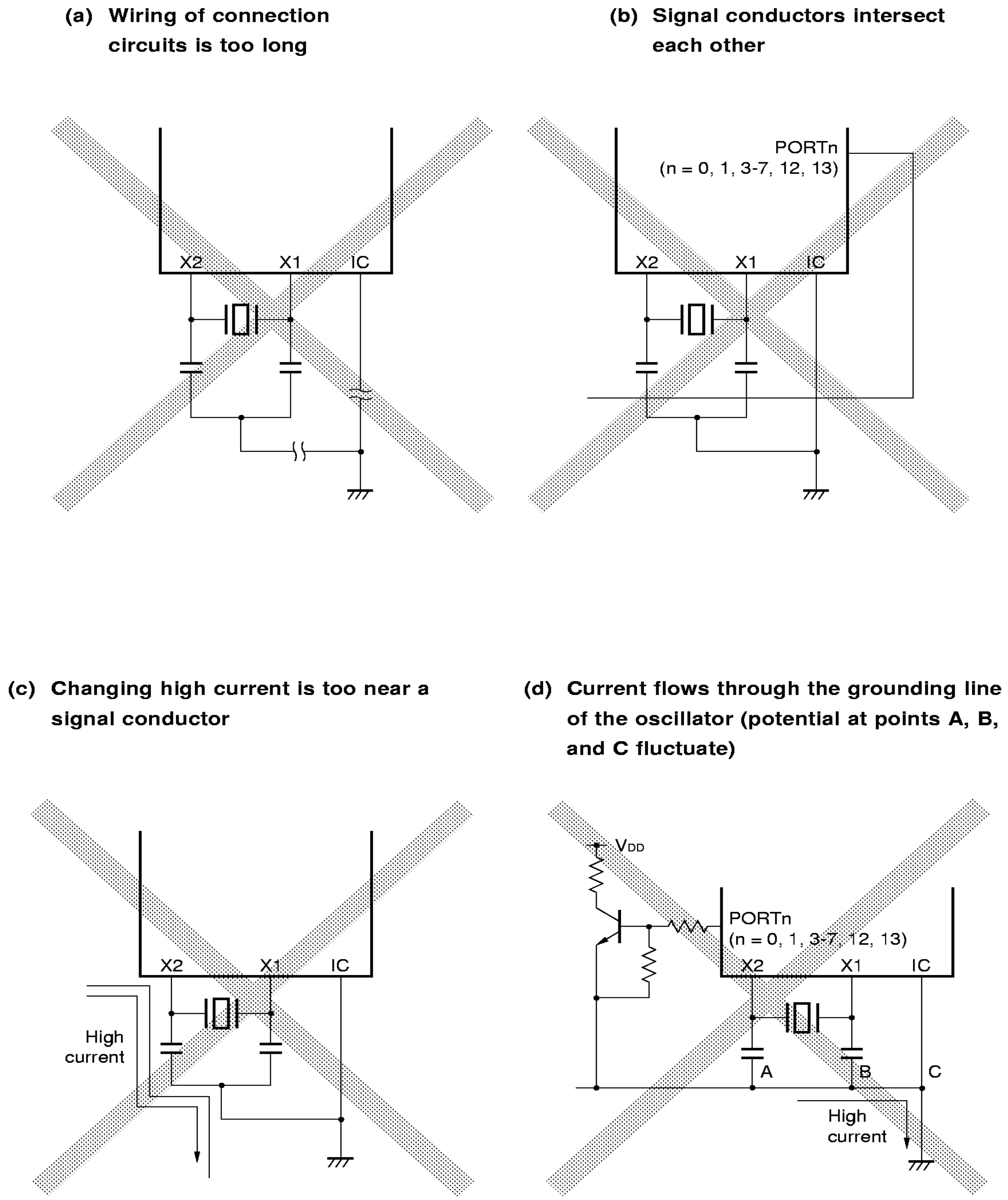


Caution When using a system clock oscillator, carry out wiring in the broken line area in Figure 6-4 to prevent any effects from wiring capacities.

- **Minimize the wiring length.**
- **Do not allow wiring to intersect with other signal conductors. Do not allow wiring to come near changing high current.**
- **Set the potential of the grounding position of the oscillator capacitor to that of GND. Do not ground to any ground pattern where high current is present.**
- **Do not fetch signals from the oscillator.**

Figure 6-5 shows examples of resonator having bad connection.

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



(a) Wiring of connection circuits is too long

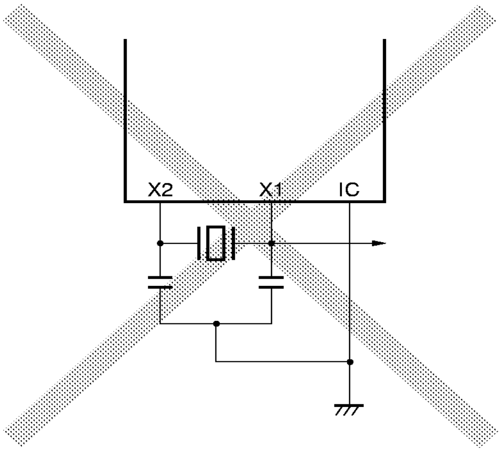
(b) Signal conductors intersect each other

(c) Changing high current is too near a signal conductor

(d) Current flows through the grounding line of the oscillator (potential at points A, B, and C fluctuate)

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

(e) Signals are fetched



6.4.2 Scaler

The scaler divides the system clock oscillator output (fx) and generates various clocks.

6.5 Clock Generator Operations

The clock generator generates the following various types of 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 with the processor clock control register (PCC).

(a) Upon generation of $\overline{\text{RESET}}$ signal, the lowest speed mode of the system clock (7.11 μs when operated at 4.5 MHz) is selected (PCC = 04H). System clock oscillation stops while low level is applied to $\overline{\text{RESET}}$ pin.

(b) One of the five CPU clock types (0.45, 0.89, 1.78, 3.56, 7.11 μs at 4.5 MHz) can be selected by setting the PCC and OSMS.

(c) Two standby modes, the STOP and HALT modes, 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.

6.6 Changing System Clock and CPU Clock Settings

6.6.1 Time required for switchover between system clock and CPU clock

The system clock and CPU clock can be switched over by means of 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, but operation continues on the pre-switchover clock for several instructions (refer to **Table 6-2**).

Table 6-2. 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 7 8-BIT TIMERS/EVENT COUNTERS

7.1 Functions of 8-Bit Timers/Event Counters

The 8-bit timers/event counters (TM50 and TM51) have the following two modes:

- Mode in which an 8-bit timer/event counter is used alone (single mode)
- Mode in which the two timers/event counters are cascaded (cascade 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)

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

- Interval timer
- External event counter
- Square wave output
- PWM output

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

By connecting the two 8-bit timers/event counters in cascade, they 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 the 8-bit timers/event counters.

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

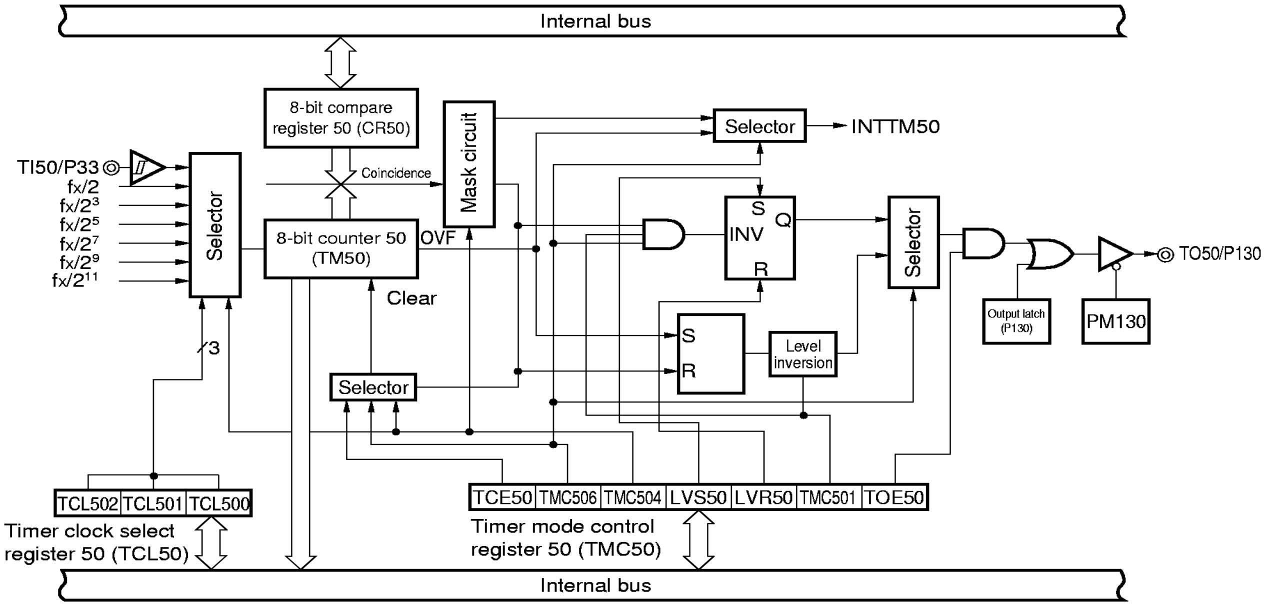
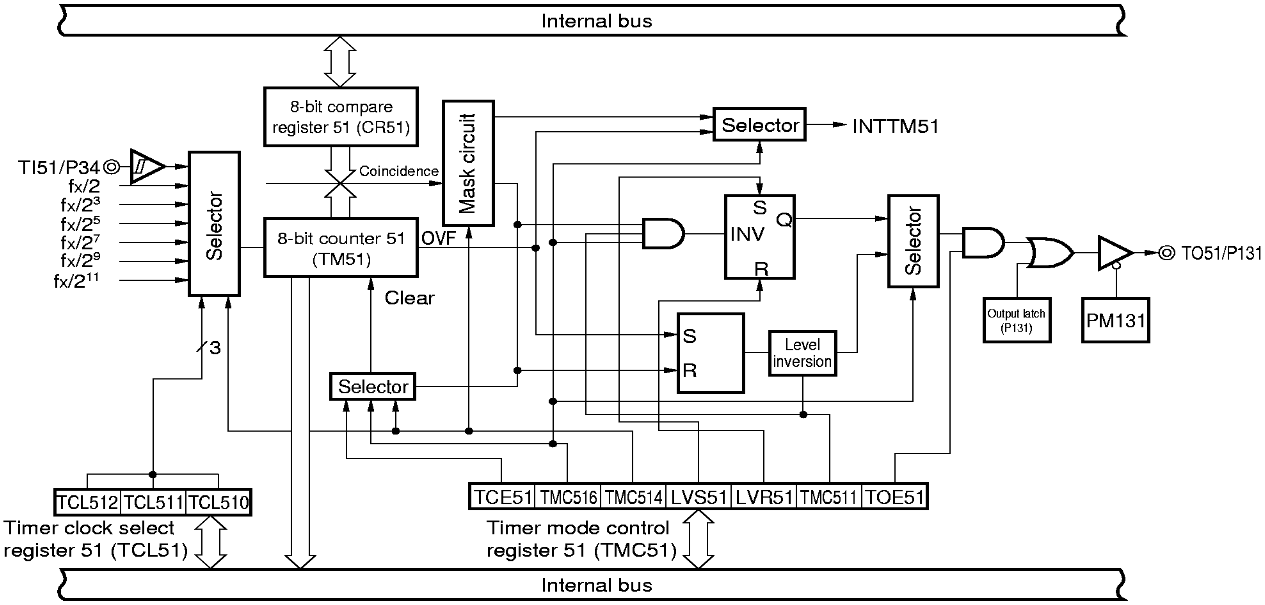


Figure 7-2. Block Diagram of 8-Bit Timer/Event Counter 51



7.2 Configuration of 8-Bit Timers/Event Counters

The 8-bit timers/event counters consist of the following hardware:

Table 7-1. Configuration of 8-Bit Timers/Event Counters

Item	Configuration
Timer register	8-bit counters 50 and 51 (TM50 and TM51)
Register	8-bit compare registers 50 and 51 (CR50 and CR51)
Timer output	2 lines (TO50 and TO51)
Control register	• Timer clock select registers 50 and 51 (TCL50 and TCL51) • 8-bit timer mode control registers 50 and 51 (TMC50 and TMC51)

(1) 8-bit 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.

TM50 and TM51 can be cascaded and used as a 16-bit timer.

When TM50 and TM51 are cascaded and used as a 16-bit timer, its value can be used by using a 16-bit memory manipulation instruction. However, because TM50 and TM51 are connected with the internal 8-bit bus, each of them is read at a time. Therefore, read the value of TM50 and TM51 when used as a 16-bit timer two times for comparison, taking a change in the value during counting into consideration.

If the count value is read while the timer is operating, stop input of the count clock, and read the count value at that point. The count value is cleared to 00H in the following cases:

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

<2> Clearing TCE5n

<3> Coincidence between TM5n and CR5n in mode in which the timer is cleared and started on coincidence between TM5n and CR5n

Caution When TM50 and TM51 are cascaded, the value of the timer is cleared to 00H even if the least significant bit (TCE51) of the timer mode control register 51 (TMC51) is cleared.

Remark n = 0, 1

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

The values set to CR50 and CR51 are always compared with the values of 8-bit counters 50 and 51 (TM50 and TM51). When the value of a compare register coincides with the count value of the corresponding counter, an interrupt request (INTTM50 or INTTM51) is generated (in a mode other than PWM mode).

If TM50 and TM51 are cascaded and used as a 16-bit timer, CR50 and CR51 operate together as a 16-bit compare register. Therefore, a 16-bit counter value and a 16-bit compare register value are compared, and when the two values coincide, an interrupt request (INTTM50) is generated. At this time, interrupt request INTTM51 is also generated. Therefore, mask INTTM51 when using TM50 and TM51 in the cascade mode.

Caution When TM0 and TM51 are cascaded, be sure to write data to the timers after stopping the timer operation.

7.3 Registers Controlling 8-Bit Timers/Event Counters

The following two types of registers control the 8-bit timers/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)

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

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

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

At reset, the values of these registers are initialized to 00H.

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

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

TCL502	TCL501	TCL500	Selects Count Clock
0	0	0	Falling edge of TI50
0	0	1	Rising edge of TI50
0	1	0	fx/2 (2.25 MHz)
0	1	1	fx/2^3 (563 kHz)
1	0	0	fx/2^5 (141 kHz)
1	0	1	fx/2^7 (35.2 kHz)
1	1	0	fx/2^9 (8.79 kHz)
1	1	1	fx/2^{11} (2.20 kHz)

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

2. Be sure to reset bits 3 through 7 to “0”.

Remarks 1. In the cascade mode, the setting of bits TCL500 through TCL502 of the low-order timer (TM50) is valid.

2. fx: System clock oscillation frequency

3. ( ): fx = 4.5 MHz

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

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

TCL512	TCL511	TCL510	Selects Count Clock
0	0	0	Falling edge of TI51
0	0	1	Rising edge of TI51
0	1	0	fx/2 (2.25 MHz)
0	1	1	fx/2^3 (563 kHz)
1	0	0	fx/2^5 (141 kHz)
1	0	1	fx/2^7 (35.2 kHz)
1	1	0	fx/2^9 (8.79 kHz)
1	1	1	fx/2^{11} (2.20 kHz)

Cautions 1. Before changing the data of TCL51, be sure to stop the timer operation.

2. Be sure to reset bits 3 through 7 to “0”.

Remarks 1. In the cascade mode, the setting of bits TCL510 through TCL512 is invalid.

2. fx: System clock oscillation frequency

3. ( ): fx = 4.5 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 counters 50 and 51 (TM50 and TM51)
<2> Selecting operation mode of 8-bit counters 50 and 51 (TM50 and TM51)
<3> Selecting single mode or cascade mode
<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 at 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	At reset	R/W
TMC50	TCE50	TMC506	0	TMC504	LVS50	LVR50	TMC501	TOE50	FF85H	00H	R/W

TCE50	Controls Count Operation of TM50
0	Clears counter to 0 and disables count operation (disables prescaler)
1	Starts count operation

TMC506	Selects Operating Mode of TM50
0	Mode of clearing and starting TM50 on coincidence between TM50 and CR50
1	PWM (free-running) mode

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

LVS50	LVR50	Setting Status of Timer Output F/F
0	0	Not affected
0	1	Resets timer output F/F to 0
1	0	Sets timer output F/F to 1
1	1	Setting prohibited

TMC501	Other Than PWM Mode (TMC506 = 0)	PWM Mode (TMC506 = 1)
	Controls timer F/F	Selects active level
0	Disables inversion operation	High active
1	Enables inversion operation	Low active

TOE50	Controls Timer Output
0	Disables output (port mode)
1	Enables output

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	At reset	R/W
TMC51	TCE51	TMC516	0	TMC514	LVS51	LVR51	TMC511	TOE51	FF88H	00H	R/W

TCE51	Controls Count Operation of TM51
0	Clears counter to 0 and disables count operation (disables prescaler)
1	Starts count operation

TMC516	Selects Operating Mode of TM51
0	Mode of clearing and starting TM51 on coincidence between TM51 and CR51
1	PWM (free-running) mode

TMC514	Selects Single Mode or Cascade Mode
0	Single mode
1	Cascade mode (connected to low-order timer (TM50))

LVS51	LVR51	Setting Status of Timer Output F/F
0	0	Not affected
0	1	Resets timer output F/F to 0
1	0	Sets timer output F/F to 1
1	1	Setting prohibited

TMC511	Other Than PWM Mode (TMC516 = 0)	PWM Mode (TMC516 = 1)
	Controls timer F/F	Selects active level
0	Disables inversion operation	High active
1	Enables inversion operation	Low active

TOE51	Controls Timer Output
0	Disables output (port mode)
1	Enables output

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.

7.4 Operations of 8-Bit Timers/Event Counters

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 set in advance in 8-bit compare register 5n (CRn).

When the count value of 8-bit counter 5n (TM5n) coincides with the value set in 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 through 2 (TCL5n0 through TCL5n2) of the timer clock select register 5n (TCL5n).

For the operation if the value of the compare register is changed while the timer count operation, refer to **(2)** in **7.5 Notes on 8-Bit Timers/Event Counters**.

[Setting]

<1> Set each register.
- TCL5n : Select a count clock.
- CR5n : Compare value
- TMC5n : Select a mode in which TM5n is cleared and started on coincidence between TM5n and CR5n (TMC5n = 0000×××0B: × = don't care).

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

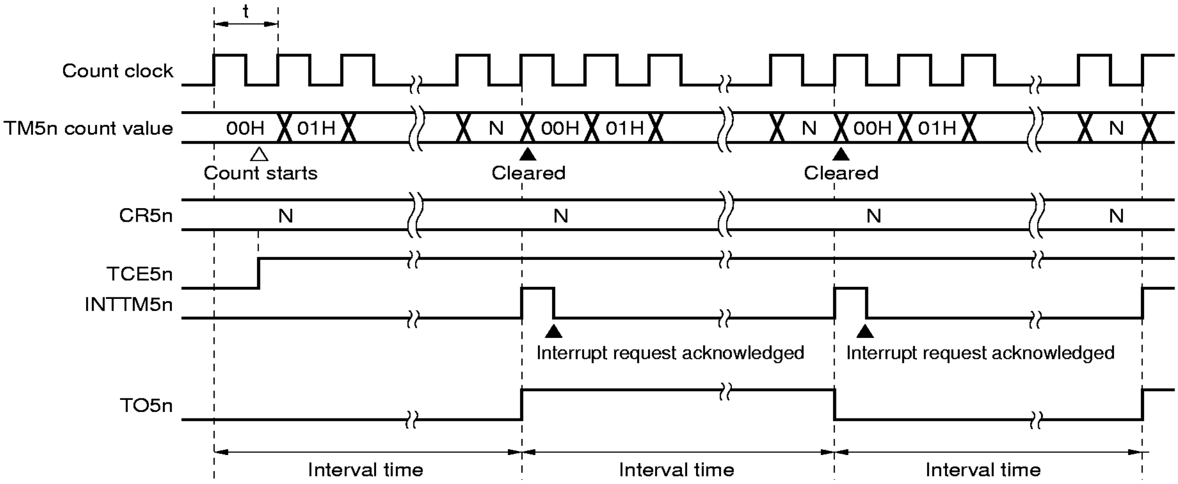
<3> INTTM5n is generated if the values of TM5n and CR5n coincide (TM5n is cleared to 00H).

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

(a) Basic operation

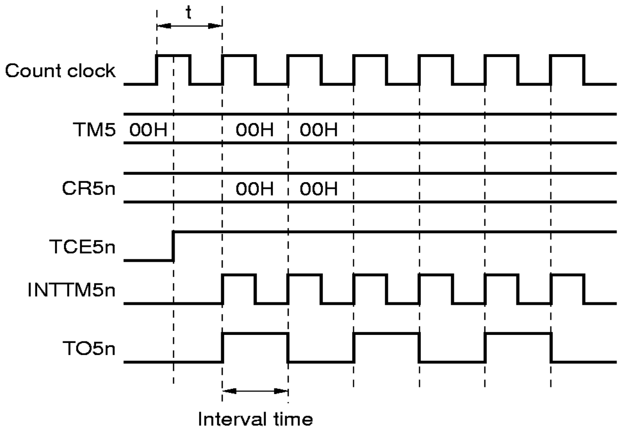


Remarks 1. Interval time = (N + 1) × t: N = 00H to FFH

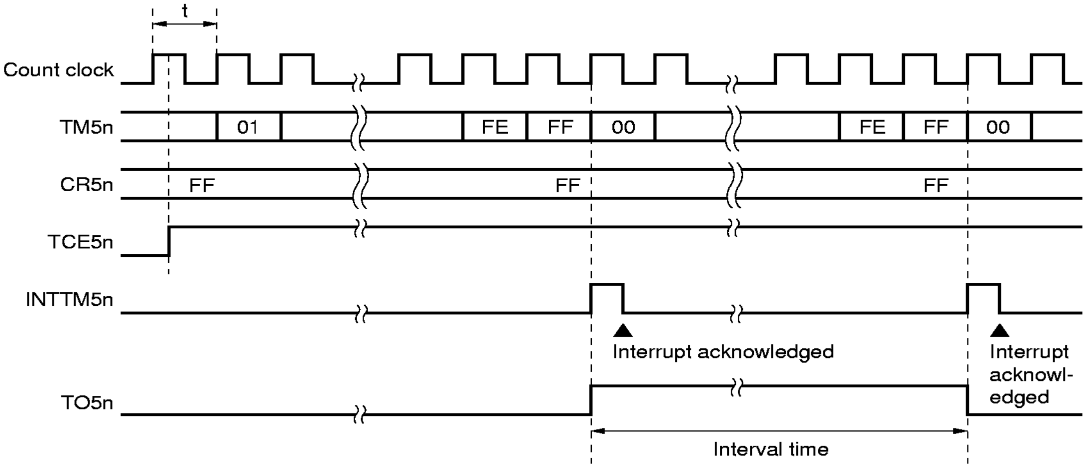
2. n = 0 or 1

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

(b) When CR5n = 00H



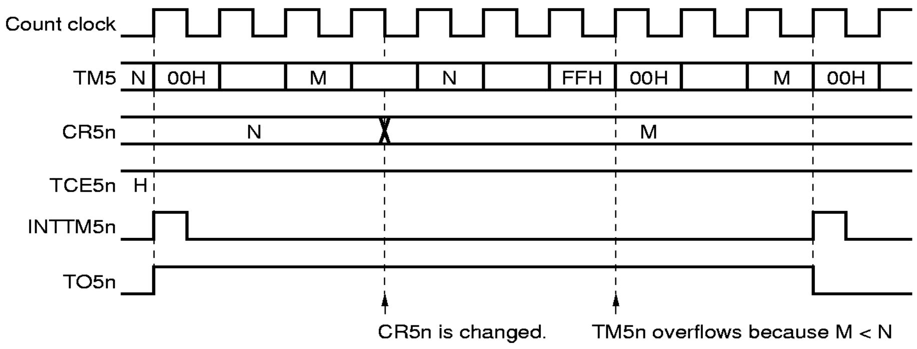
(c) When CR5n = FFH



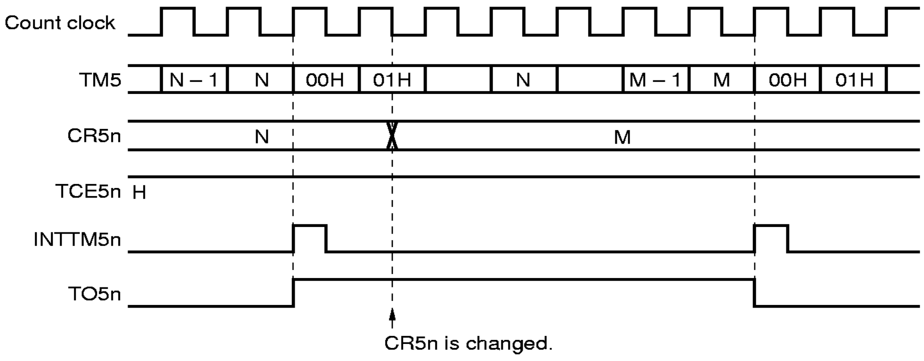
n = 0 or 1

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

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



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



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 TI5n pin with 8-bit counter 5n (TM5n).

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

When the count value of TM5n coincides with 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 coincides with the value of CR5n, INTTM5n is generated.

[Setting]

<1> Set each register.

- TCL5n : Select the valid edge of TI5n input.
- CR5n : Compare value
- TMC5n : Select a mode in which TM5n is cleared and started on coincidence between TM5n and CR5n.

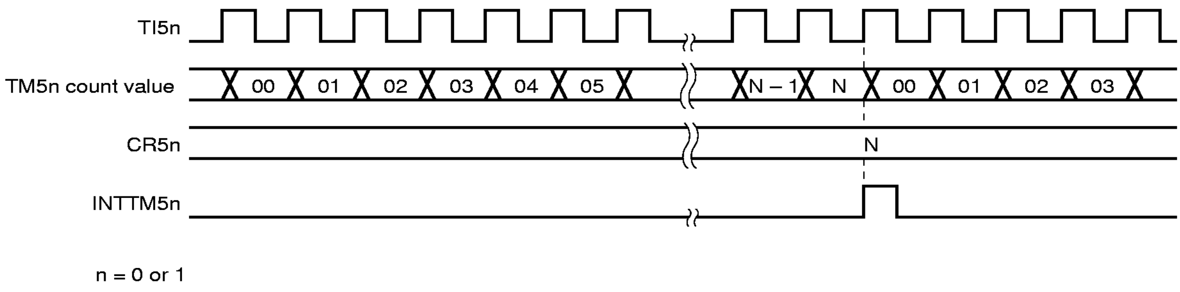
<2> The count operation is started when TEC5n is set to 1.

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

<4> After that, INTTM5n is generated each time the value of TM5n coincides with the value of CR5n. To stop the count operation, clear TCE5n to 0.

Remark n = 0 or 1

Figure 7-8. Operation Timing of External Event Counter (with rising edge specified)



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 interval specified by the value set in advance in 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 set in advance 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 latch and port mode register to "0".
- TCL5n : Select a count clock.
- CR5n : Compare value
- TMC5n : Mode in which TM5n is cleared and started on coincidence between TM5n and CR5n

LVS5n	LVR5n	Sets 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.

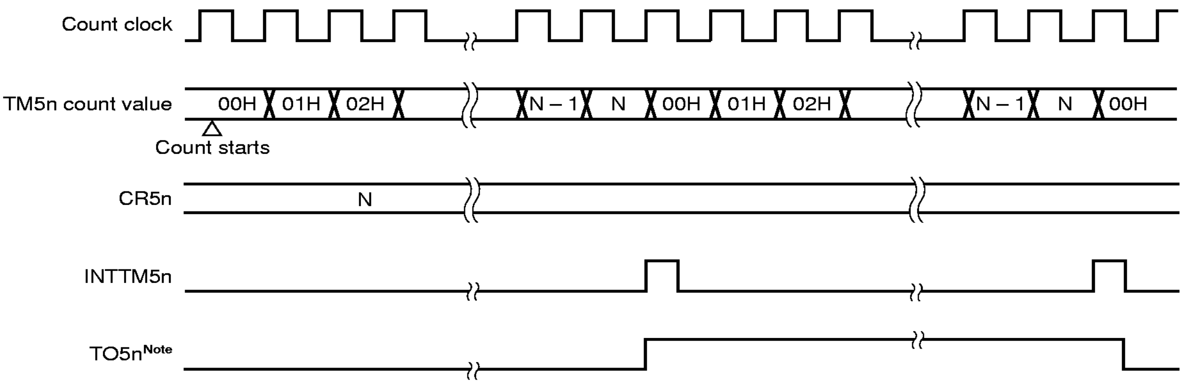
<2> When TCE5n is set to 1, the count operation is started.

<3> When the value of TM5n coincides with 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 7-9. Timing of Square 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".

The pulse with a duty factor determined by the value set in 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 through 2 (TCL5n0 through 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 one in one cycle in the PWM mode.

Remark n = 0 or 1

(1) Basic operation of PWM output

[Setting]

<1> Set port latches (P130 and P131) to "0".

<2> Select an active level width by using the 8-bit compare register (CR5n).

<3> Select a count clock by using the timer clock select register 5n (TCL5n).

<4> Select an active level by using bit 1 (TMC5n1) of TMC5n.

<5> When bit 7 (TCE5n) of TMC5n is set to "1", the count operation is started.
To stop the count operation, reset TCE5n to "0".

[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 set in step <1> above is output. This active level is output until the value of CR5n coincides with the count value of 8-bit counter 5n (TM5n).

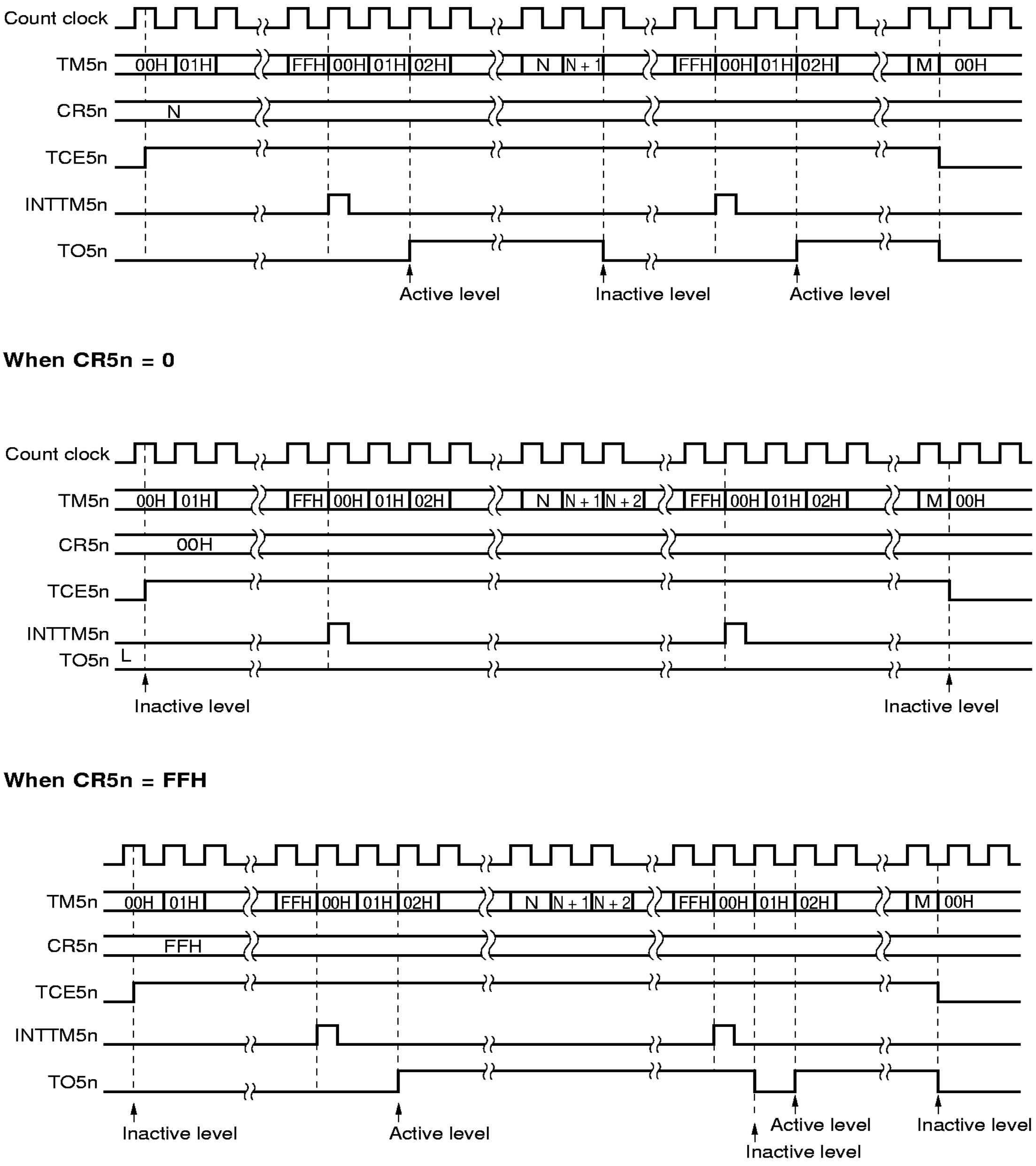
<3> The PWM output remains inactive after CR5n and the count value of TM5n have coincided, until an overflow occurs again.

<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 7-10. Operation Timing of PWM Output



(a) Basic operation (when active level = H)

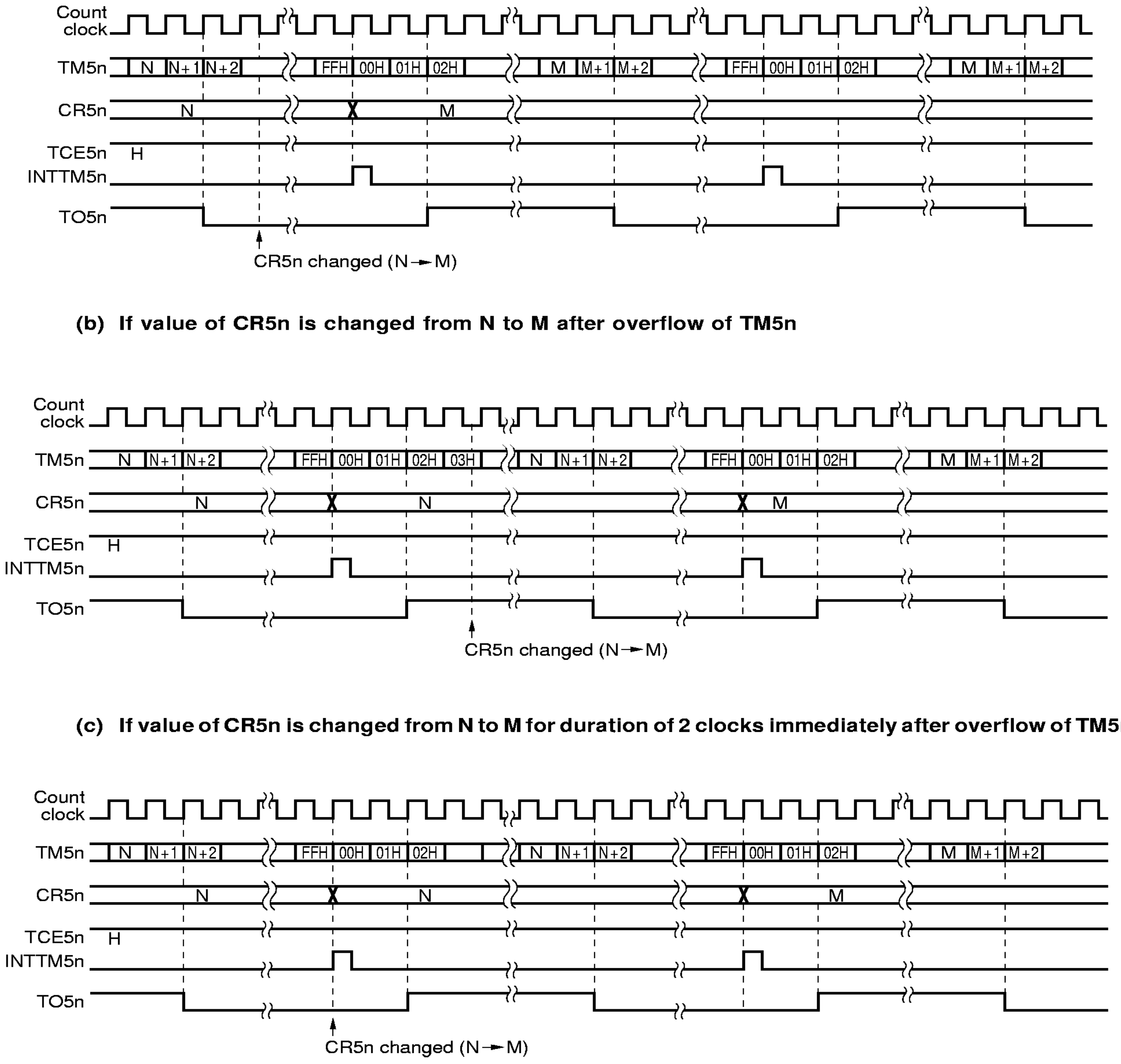
(b) When CR5n = 0

(c) When CR5n = FFH

n = 0 or 1

(2) Operation when CR5n is changed

Figure 7-11. Timing of Operation When CR5n Is Changed



(a) If value of CR5n is changed from N to M before overflow of TM5n

(b) If value of CR5n is changed from N to M after overflow of TM5n

(c) If value of CR5n is changed from N to M for duration of 2 clocks immediately after overflow of TM5n

n = 0 or 1

Caution The value of CR5n can be changed only once in one cycle in the PWM mode.

7.4.5 As interval timer operation (16-bit)

The 8-bit times/event counters are used together in a 16-bit timer/counter mode when bit 4 (TMC514) of the 8-bit timer mode control register 51 (TM51) is set to “1”.

In this mode, the 8-bit timers/event counters are used as a 16-bit interval timer that repeatedly generates an interrupt request at intervals specified by the count value set in advance in the 8-bit compare registers (CR50 and CR51).

At this time, CR50 serves as the low-order 8 bits of the 16-bit compare register, and CR51 serves as the high-order 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 : Compare values (Each compare value can be set in a range of 00H to FFH.)
- TMC50 and TMC51 : Select a mode in which the interval timer is cleared and started on coincidence between TM50 and CR50 (or between TM51 and CR51).
  $\left(\begin{array}{l}\text{TM50} \rightarrow \text{TMC50} = 0000\times\times\times0\text{B} \ \times\text{: don't care} \\ \text{TM51} \rightarrow \text{TMC51} = 0001\times\times\times0\text{B} \ \times\text{: don't care}\end{array}\right)$

<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 coincides with 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 timer operation.

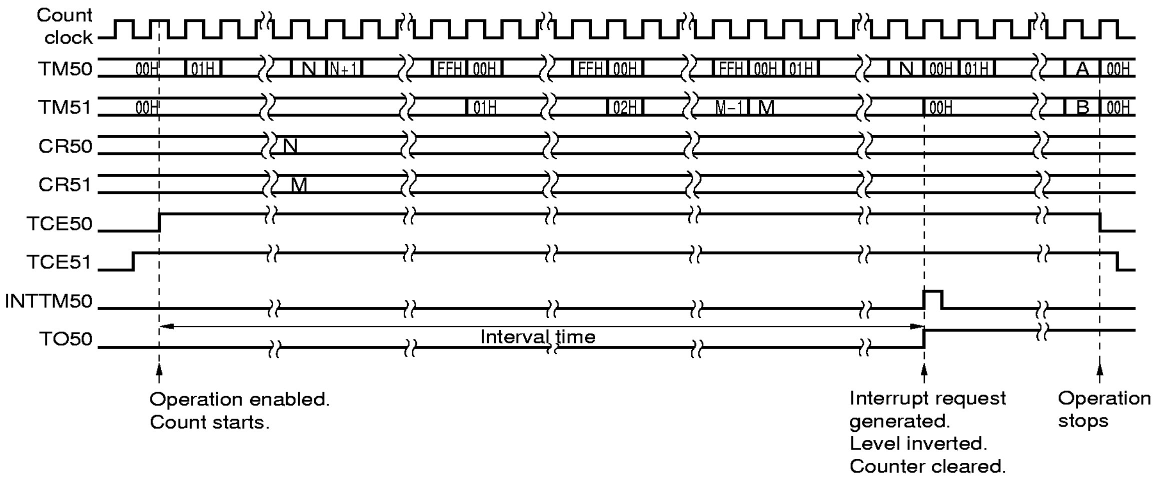
2. Even if the 8-bit timers/counters are cascaded, INTTM51 of TM51 is generated when the count value of TM51 coincides with CR51. Be sure to mask TM51 to disable this interrupt.

3. Set TCE50 and TCE51 in the order of TM51 and TM50.

4. Counting can be restarted or stopped by setting or resetting only TCE50 of TM50 to 1 or 0.

Figure 7-12 shows a timing example in the 16-bit cascade mode.

Figure 7-12. 16-Bit Cascade Mode

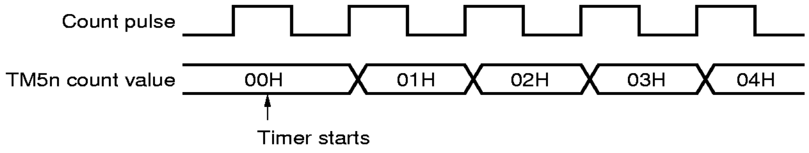


7.5 Notes on 8-Bit Timers/Event Counters

(1) Error on starting timer

An error of up to 1 clock occurs after the timer has been started until a coincidence signal is generated. This is because 8-bit counter 5n (TM5n) is started asynchronously with the count pulse.

Figure 7-13. Start Timing of 8-Bit Counter

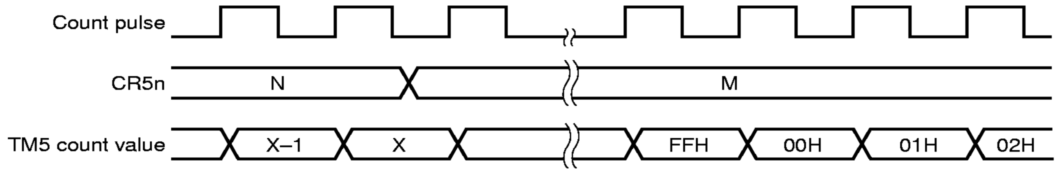


n = 0 or 1

(2) Operation after changing compare register during timer count operation

If a new value of 8-bit compare register 5n (CR5n) is less than the value of 8-bit counter 5n (TM5n), counting continues, TM5n overflows and starts counting from 0. If the new value of CR5n (M) is less than the old value (N), therefore, it is necessary to restart the timer after changing CR5n.

Figure 7-14. Timing after Changing Compare Register Value during Timer Count Operation



Caution Be sure to clear TCE5n to 0 to set the STOP status, except when TI5n input is selected.

Remarks 1. N > X > M

2. n = 0 or 1

(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 fx, the count clock to be selected should be fx/4 or less in order that TM5n can be read.

Remark n = 0 or 1

[MEMO]

CHAPTER 8 BASIC TIMER

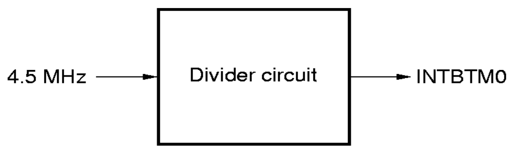
The basic timer is used for time management during program execution.

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 Use the basic timer after setting bit 0 of the DTS system clock select register (DETSCK) to 1 after power application, and after reset by the $\overline{\text{RESET}}$ pin (refer to 6.1 Functions of Clock Generator). 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.

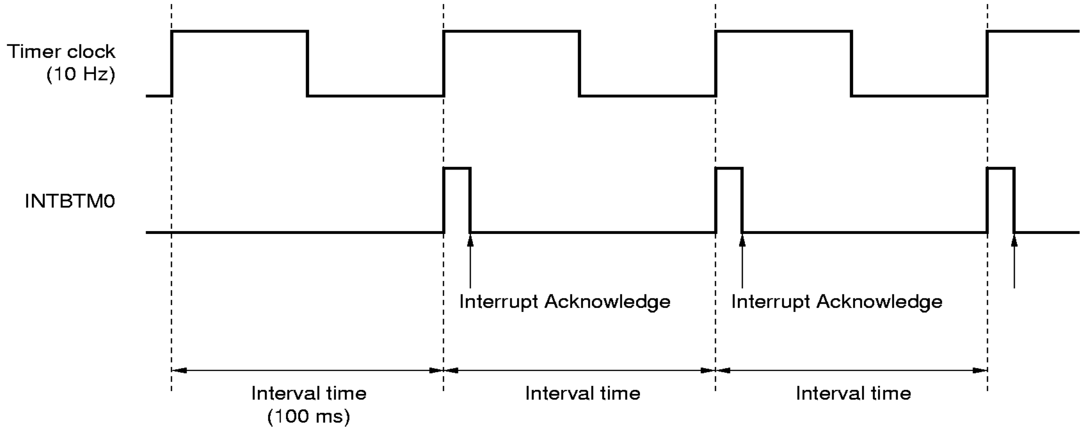
8.3 Operation of Basic Timer

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. Interrupt request signal (INTBTM0) is generated every 100 ms.

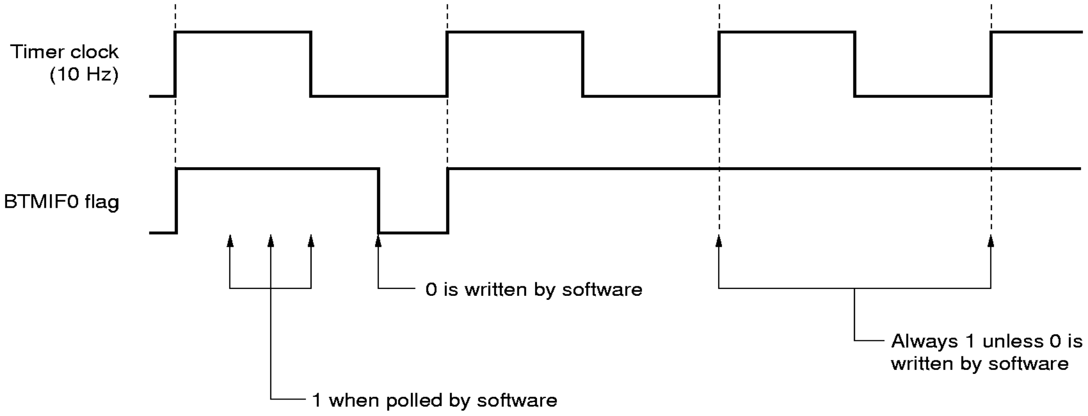
The timer clock frequency is 10 Hz.

Figure 8-2. Operation Timing of Basic Timer



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. Operating Timing to Poll BTMIF0 Flag



For the registers controlling the basic timer, refer to **CHAPTER 15 INTERRUPT FUNCTIONS**.

CHAPTER 9 WATCHDOG TIMER

9.1 Functions of Watchdog Timer

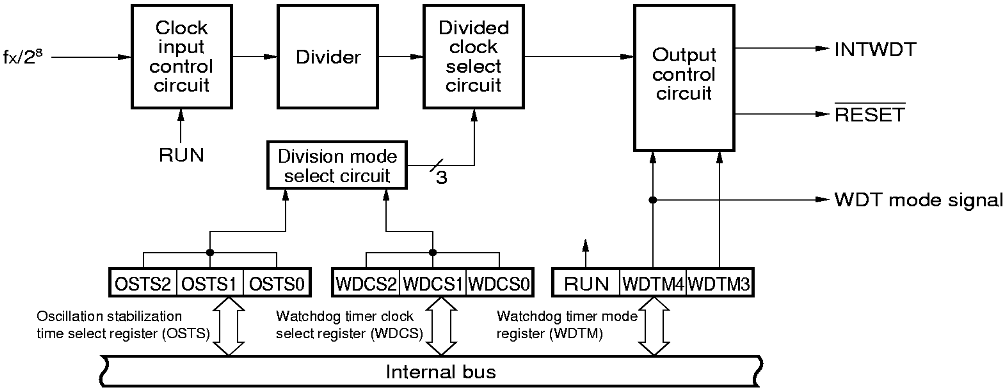
The watchdog timer has the following functions.

- Watchdog timer
- Interval timer
- Selecting oscillation stabilization time

Caution Select the watchdog timer mode or the interval timer mode with the watchdog timer mode register (WDTM). (The watchdog timer and interval timer cannot be used simultaneously.)

Figure 9-1 shows a block diagram.

Figure 9-1. Block Diagram of Watchdog Timer



fx/2⁸
Clock input control circuit
Divider
Divided clock select circuit
Output control circuit
INTWDT
RESET
RUN
Division mode select circuit
3
WDT mode signal
OSTS2
OSTS1
OSTS0
WDCS2
WDCS1
WDCS0
RUN
WDTM4
WDTM3
Oscillation stabilization time select register (OSTS)
Watchdog timer clock select register (WDCS)
Watchdog timer mode register (WDTM)
Internal bus

(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 Times

Inadvertent Program Loop Detection Time
2^{12}/fx (910 μs)
2^{13}/fx (1.82 ms)
2^{14}/fx (3.64 ms)
2^{15}/fx (7.28 ms)
2^{16}/fx (14.6 ms)
2^{17}/fx (29.1 ms)
2^{18}/fx (58.3 ms)
2^{20}/fx (233 ms)

Remark fx: System clock oscillation frequency
( ): fx = 4.5 MHz

(2) Interval timer mode

Interrupt requests are generated at the preset time intervals.

Table 9-2. Interval Times

Interval Time
2^{12} /fx (910 μs)
2^{13}/fx (1.82 ms)
2^{14}/fx (3.64 ms)
2^{15}/fx (7.28 ms)
2^{16}/fx (14.6 ms)
2^{17}/fx (29.1 ms)
2^{18}/fx (58.3 ms)
2^{20}/fx (233 ms)

Remark fx: System clock oscillation frequency
( ): fx = 4.5 MHz

9.2 Configuration of Watchdog Timer

The watchdog timer consists of the following hardware.

Table 9-3. Configuration of Watchdog Timer

Item	Configuration
Control register	Watchdog timer clock select register (WDCS) Watchdog timer mode register (WDTM) Oscillation stabilization time select register (OSTS)

9.3 Watchdog Timer Control Registers

The following three types of registers are used to control the watchdog timer.

- Watchdog timer clock select register (WDCS)
- Watchdog timer mode register (WDTM)
- Oscillation stabilization time select register (OSTS)

(1) Watchdog timer clock select register (WDCS)

This register sets the watchdog timer and overflow time of the interval timer.

WDCS is set with 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	At 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 (910 μs)
0	0	1	2^{13}/fx (1.82 ms)
0	1	0	2^{14}/fx (3.64 ms)
0	1	1	2^{15}/fx (7.28 ms)
1	0	0	2^{16}/fx (14.6 ms)
1	0	1	2^{17}/fx (29.1 ms)
1	1	0	2^{18}/fx (58.3 ms)
1	1	1	2^{20}/fx (233 ms)

Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 4.5 MHz

(2) Watchdog timer mode register (WDTM)

This register sets the watchdog timer operating mode and enables/disables counting.

WDTM is set with 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	At reset	R/W
WDTM	RUN	0	0	WDTM4	WDTM3	0	0	0	FFF9H	00H	R/W

RUN	Watchdog Timer Operating Mode Selection[Note 1]
0	Count stop
1	Counter is cleared and counting starts.

WDTM4	WDTM3	Watchdog Timer Operating Mode Selection[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. WDTM starts interval timer operation at a time RUN is set to 1.

3. Once set to 1, WDTM3 and WDTM4 cannot be cleared to 0 by software.

Caution **When 1 is set in RUN so that the watchdog timer is cleared, the actual overflow time is up to 0.5 % shorter than the time set by timer clock select register (WDCS).**

Remark ×: don't care

(3) 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 using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is set to 04H at reset. To release the STOP mode by $\overline{\text{RESET}}$ input, therefore, it takes 2^{17}/fx to release the STOP mode.

Figure 9-4. Format of Oscillation Stabilization Time Select Register (OSTS)

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

OSTS2	OSTS1	OSTS0	Selects Oscillation Stabilization Time
0	0	0	2^{12}/fx (910 μs)
0	0	1	2^{14}/fx (3.64 ms)
0	1	0	2^{15}/fx (7.28 ms)
0	1	1	2^{16}/fx (14.6 ms)
1	0	0	2^{17}/fx (29.1 ms)
Others			Setting prohibited

Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 4.5 MHz

9.4 Operations of Watchdog Timer

9.4.1 Watchdog timer operation

When bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 1, the watchdog timer is operated to detect any inadvertent program loop.

The watchdog timer count clock (inadvertent program loop detection time interval) can be selected with bits 0 to 2 (WDCS0 to WDCS2) of the timer clock select register 2 (WDCS). Watchdog timer starts count operation by setting bit 7 (RUN) of WDTM to 1. After the watchdog timer starts count operation, set RUN to 1 within the set inadvertent program loop time interval.

The watchdog timer can be cleared and counting is started by setting RUN to 1. If RUN is not set to 1 and the inadvertent program loop detection time is past, system reset or a non-maskable interrupt request is generated according to the WDTM bit 3 (WDTM3) value.

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 a maximum of 0.5 %.

Table 9-4. Watchdog Timer Inadvertent Program Loop Detection Time

Inadvertent Program Loop Detection Time
2^{12}/fx (910 μs)
2^{13}/fx (1.82 ms)
2^{14}/fx (3.64 ms)
2^{15}/fx (7.28 ms)
2^{16}/fx (14.6 ms)
2^{17}/fx (29.1 ms)
2^{18}/fx (58.3 ms)
2^{20}/fx (233 ms)

Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 4.5 MHz

9.4.2 Interval timer operation

The watchdog timer operates as an interval timer which 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) is set to 1 and 0, respectively.

A count clock (interval time) can be selected by using bits 0 through 2 (WDCS0 through 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 operated 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 maskable interrupt requests, the INTWDT default has the highest 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 setting with WDTM may be shorter than the set time by a maximum of 0.5 %.

Table 9-5. Interval Timer Interval Time

Interval Time
2^{12}/fx (910 μs)
2^{13}/fx (1.82 ms)
2^{14}/fx (3.64 ms)
2^{15}/fx (7.28 ms)
2^{16}/fx (14.6 ms)
2^{17}/fx (29.1 ms)
2^{18}/fx (58.3 ms)
2^{20}/fx (233 ms)

Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 4.5 MHz

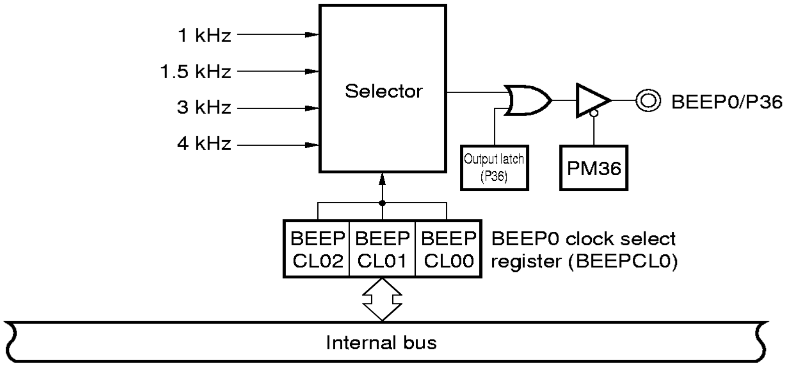
CHAPTER 10 BUZZER OUTPUT CONTROL CIRCUIT

10.1 Functions of Buzzer Output Control Circuit

The buzzer output control circuit outputs a square wave of the buzzer frequency selected by the BEEP0 clock select register (BEEPCL0) from the BEEP0/P36 pin.

Figure 10-1 shows the block diagram of BEEP0.

Figure 10-1. Block Diagram of Buzzer Output Control Circuit



10.2 Configuration of Buzzer Output Control Circuit

The buzzer output control circuit consists of the following hardware:

Table 10-1. Configuration of Buzzer Output Control Circuit

Item	Configuration
Control register	BEEP0 clock select register (BEEPCL0)

10.3 Registers Controlling Buzzer Output Circuit

The buzzer output control circuit is controlled by the following register:

- BEEP0 clock select register (BEEPCL0)

(1) BEEP0 clock select register (BEEPCL0)

This register selects the frequency of buzzer output.
BEEPCL0 is set by using a 1-bit or 8-bit memory manipulation instruction.
This register is initialized to 00H at reset.

Figure 10-2. Format of BEEP0 Clock Select Register (BEEPCL0)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
BEEP CL0	0	0	0	0	0	BEEP CL02	BEEP CL01	BEEP CL00	FF41H	00H	R/W

BEEP CL02	BEEP CL01	BEEP CL00	Selects Frequency of BEEP0 Output
0	×	×	Disables buzzer output (port function)
1	0	0	1 kHz
0	0	1	3 kHz
1	1	0	4 kHz
1	1	1	1.5 kHz

Caution The selected clock may not be correctly output during the period of 1 cycle immediately after the output clock has been changed.

10.4 Operation of Buzzer Output Control Circuit

The buzzer frequency is output in the following procedure:

<1> Select a buzzer output clock by using bits 0 through 2 (BEEPCL00 through BEEPCL02) of the BEEP0 clock select register (BEEPCL0).
<2> Reset the output latch of P36 to 0.
<3> Reset bit 6 (PM36) of the port mode register 3 to 0 (set the output mode).

CHAPTER 11 A/D CONVERTER

11.1 Functions of A/D Converter

The A/D converter converts an analog input into a digital value. It consists of 6 channels (ANI0 to ANI5) with an 8-bit resolution.

The conversion method is based on successive approximation and the conversion result is held in the 8-bit A/D conversion result register 3 (ADCR3).

Conversion is started by setting the A/D converter mode register 3.

Select one channel of analog input from ANI0 to ANI5 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.

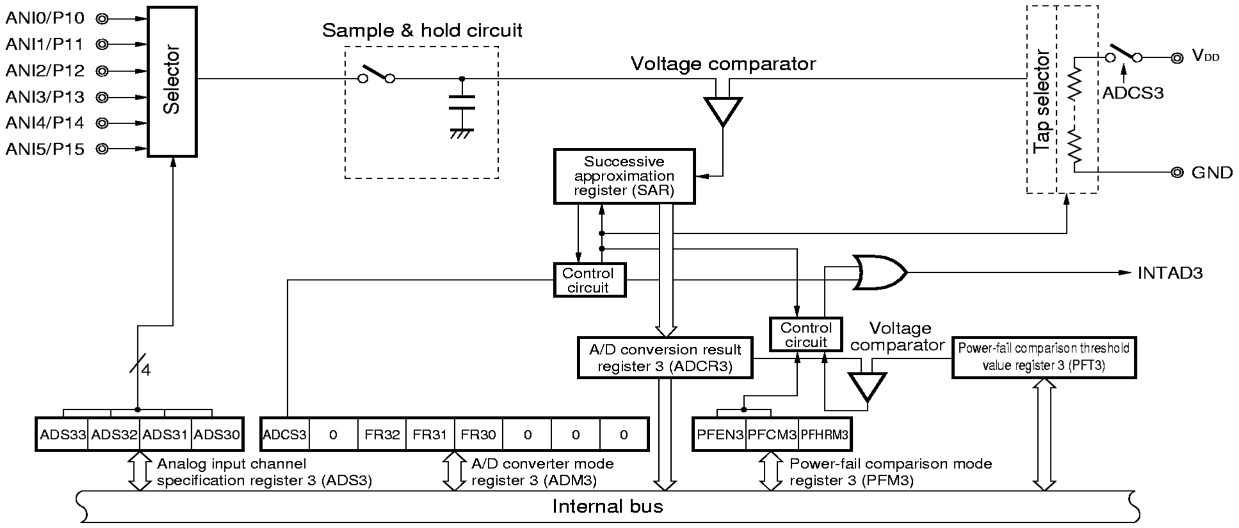
11.2 Configuration of A/D Converter

The A/D converter consists of the following hardware.

Table 11-1. Configuration of A/D Converter

Item	Configuration
Analog input	6 channels (ANI0 to ANI5)
Control register	A/D converter mode register 3 (ADM3) Analog input channel specification register 3 (ADS3) Power-fail comparison mode register 3 (PFM3)
Register	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


ANI0/P10
ANI1/P11
ANI2/P12
ANI3/P13
ANI4/P14
ANI5/P15
Selector
Sample & hold circuit
Voltage comparator
Tap selector
VDD
ADCS3
GND
Successive approximation register (SAR)
Control circuit
INTAD3
Control circuit
Voltage comparator
A/D conversion result register 3 (ADCR3)
Power-fail comparison threshold value register 3 (PFT3)
4
ADS33
ADS32
ADS31
ADS30
ADCS3
0
FR32
FR31
FR30
0
0
0
PFEN3
PFCM3
PFHRM3
Analog input channel specification register 3 (ADS3)
A/D converter mode register 3 (ADM3)
Power-fail comparison mode register 3 (PFM3)
Internal bus

(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 (termination 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 is an 8-bit register to store the A/D conversion result. Each time A/D conversion terminates, the conversion result is loaded from the successive approximation register (SAR).

ADCR is read with an 8-bit memory manipulation instruction.

Reset input makes ADCR undefined.

Caution When data is written to the A/D converter mode register (ADM3) and analog input channel specification register 3 (ADS3), the contents of ADCR3 can be 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 a threshold value to be compared with the value of the 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 each analog input signal sequentially applied from the input circuit and sends it to the voltage comparator. This circuit holds the sampled analog input voltage value during A/D conversion.

(5) Voltage comparator

The voltage comparator compares the analog input to the series resistor string output voltage.

(6) Resistor string

The resistor string is connected between V_{DD} and GND, and generates a voltage to be compared to the analog input.

(7) ANI0 to ANI5 pins

These are 6-channel analog input pins to input analog signals to undergo A/D conversion to the A/D converter.

Cautions 1. Use ANI0 to ANI5 input voltages within the specified range. If a voltage higher than V_{DD} or lower than GND is applied (even if within the absolute maximum ratings), the converted value of the corresponding channel becomes indeterminate and may adversely affect the converted values of other channels.

2. The analog input pins (ANI0 through ANI5) are multiplexed with input port pins (P10 through P15). When one of ANI0 through ANI5 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 the pin adjacent to the pin executing A/D conversion, the A/D conversion value may not be obtained as expected due to coupling noise. Do not apply a pulse to the pin adjacent to the pin executing A/D conversion.

11.3 Registers Controlling A/D Converter

The following three types of 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.

ADM3 is set by using a 1-bit or 8-bit memory manipulation instruction.

This register is reset to 00H at reset.

Figure 11-2. Format of A/D Converter Mode Register 3 (ADM3)

Symbol	<7>	6	5	4	3	2	1	0	Address	At reset	R/W
ADM3	ADCS3	0	FR32	FR31	FR30	0	0	0	FF12H	00H	R/W

ADCS3	Controls A/D Conversion Operation
0	Stops conversion operation
1	Enables conversion operation

FR32	FR31	FR30	Selects Conversion Time
0	0	0	288/fx (64.0 μs)
0	0	1	240/fx (53.3 μs)
0	1	0	192/fx (42.7 μs)
1	0	0	144/fx (32.0 μs)
1	0	1	120/fx (26.7 μs)
1	1	0	96/fx (21.3 μs)
Others			Setting prohibited

Cautions 1. The conversion result is undefined immediately after bit 7 (ADCS3) has been set to “1”.

2. To change the data of bits 3 through 5 (FR30 through FR32), stop the A/D conversion operation.

Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 4.5 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 using an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H at reset.

Figure 11-3. Format of Analog Input Channel Specification Register 3 (ADS3)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADS3	0	0	0	0	ADS33	ADS32	ADS31	ADS30	FF13H	00H	R/W

ADS33	ADS32	ADS31	ADS30	Specifies 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
Others				Setting prohibited

(3) Power-fail comparison mode register 3 (PFM3)

PFM3 is set by using an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H at reset.

Figure 11-4. Format of Power-Fail Comparison Mode Register 3 (PFM3)

Symbol	<7>	<6>	<5>	4	3	2	1	0	Address	At reset	R/W
PFM3	PFEN3	PFCM3	PFHRM3	0	0	0	0	0	FF16H	00H	R/W

PFEN3	Enables or Disables Power-fail Comparison
0	Disables power-fail comparison
1	Enables power-fail comparison

PFCM3	Selects Power-fail Comparison Mode
0	Generates interrupt request (INTAD) when ADCR3 ≥ PFT
1	Generates interrupt request (INTAD) when ADCR3 < PFT

PFHRM3 Note	Selects Power-fail HALT Repeat Mode
0	Disables power-fail HALT repeat mode
1	Enables power-fail HALT repeat mode

Note When bit 5 (PFHRM3) is set to 1, power-fail comparison manipulation is enabled in the HALT mode in which A/D conversion is repeated until an interrupt request (INTAD) is generated (this bit is reset to 0 when INTAD is generated).

11.4 Operations of A/D Converter

11.4.1 Basic operations of A/D converter

(1) Select one channel for A/D conversion with A/D converter analog input channel specification register 3 (ADS3).

(2) Sample the voltage input to the selected analog input channel with 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 termination 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) V_{DD}.

(5) The voltage difference between the series resistor string voltage tap and analog input is compared with a voltage comparator. If the analog input is greater than (1/2) V_{DD}, the MSB of SAR remains set. If the input is smaller than (1/2) V_{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) V_{DD}
- Bit 7 = 0 : (1/4) V_{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

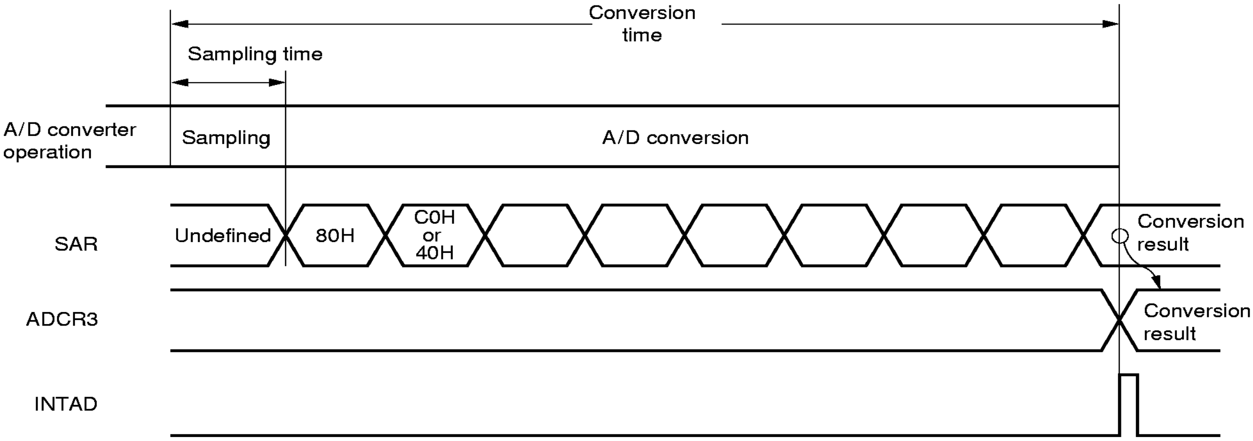
(7) Comparison of this sort continues up to bit 0 of SAR.

(8) Upon completion of the comparison of 8 bits, any effective digital resultant value remains in SAR and the resultant value is transferred to and latched in the A/D conversion result register 3 (ADCR3).

At the same time, the A/D conversion termination interrupt request (INTAD) can also be generated.

Caution The value immediately after A/D conversion has been started may not satisfy the ratings.

Figure 11-5. A/D Converter Basic Operation



A/D conversion operations are performed continuously until the bit 7 (ADCS3) of the ADM is reset (0) by software.

If a write to the 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.

After reset input, the value of ADCR3 is undefined.

11.4.2 Input voltage and conversion results

The relation between the analog input voltage input to the analog input pins (ANI0 to ANI5) and the A/D conversion result (the value stored in A/D conversion result register 3 (ADCR3) is shown by the following expression.

$$\text{ADCR3} = \text{INT}\left(\frac{V_{IN}}{V_{DD}} \times 256 + 0.5\right)$$

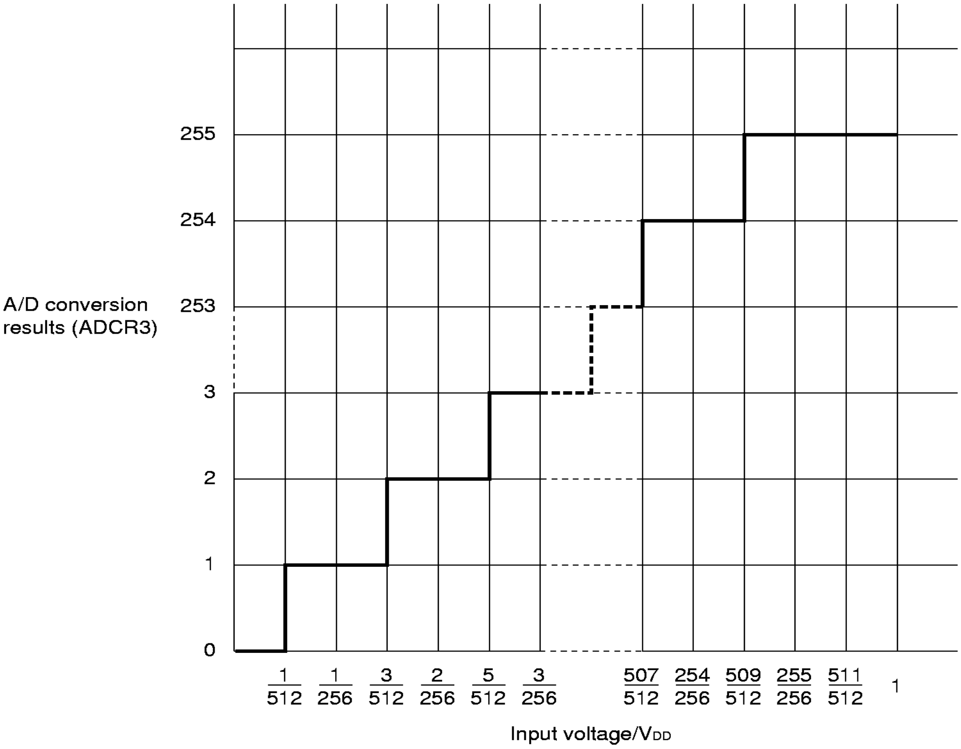
or

$$(\text{ADCR3} - 0.5) \times \frac{V_{DD}}{256} \leq V_{IN} < (\text{ADCR3} + 0.5) \times \frac{V_{DD}}{256}$$

Where, INT( ) : Function which returns integer parts of value in parentheses.
V_{IN} : Analog input voltage
V_{DD} : V_{DD} pin voltage
ADCR3 : A/D conversion result register 3 (ADCR3) value

Figure 11-6 shows the relation between the analog input voltage and the A/D conversion result.

Figure 11-6. Relations between Analog Input Voltage and A/D Conversion Result



11.4.3 A/D converter operating mode

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 through ANI5 is converted into a digital signal. The result of the A/D conversion is stored in the 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 the power-fail comparison threshold value register 3 (PFT3) is compared. If the result of the comparison coincides with the condition set by bit 6 (PFCM3) of the power-fail comparison mode register 3 (PFM3), an interrupt request signal (INTAD) is generated.

(1) A/D conversion operation mode

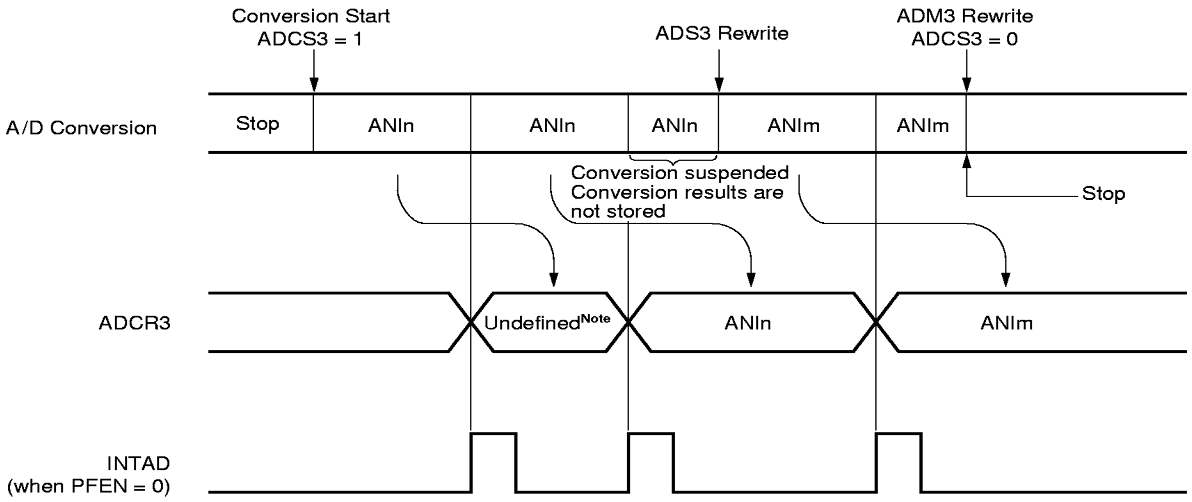
When bit 7 (ADCS3) of A/D converter mode register 3 (ADM3) is set to 1, the A/D conversion starts on the voltage applied to the analog input pins specified with bits 0 to 3 (ADS30 to ADS33) of ADS3.

Upon termination of the A/D conversion, the conversion result is stored in the A/D conversion result register 3 (ADCR3) and the interrupt request signal (INTAD) is generated. After one A/D conversion operation is started and terminated, the next A/D conversion operation starts immediately. The A/D conversion operation continues 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 im-mediately.

Figure 11-7. A/D Conversion Operation



Remarks **1.** n = 0, 1, ... , 5

2. m = 0, 1, ... , 5

Note The conversion result is illegal immediately after bit 7 (ADCS3) of the A/D converter mode register 3 (ADM3) has been set to 1 (to enable conversion).

Caution Reset bit 5 (PFHRM3) of the 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 analog input is compared in units of 8 bits.

If the result of the comparison coincides with the condition set by bit 6 (PFCM3) of the power-fail comparison mode register 3 (PFM3), an interrupt request (INTAD) is generated.

Moreover, 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 (PEEN3) 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 the A/D conversion result register 3 (ADCR3) to the power-fail comparison threshold value register 3 (PFT3).

By setting bit 7 (ADCS3) of the 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 the 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 coincides with 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	At 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**.

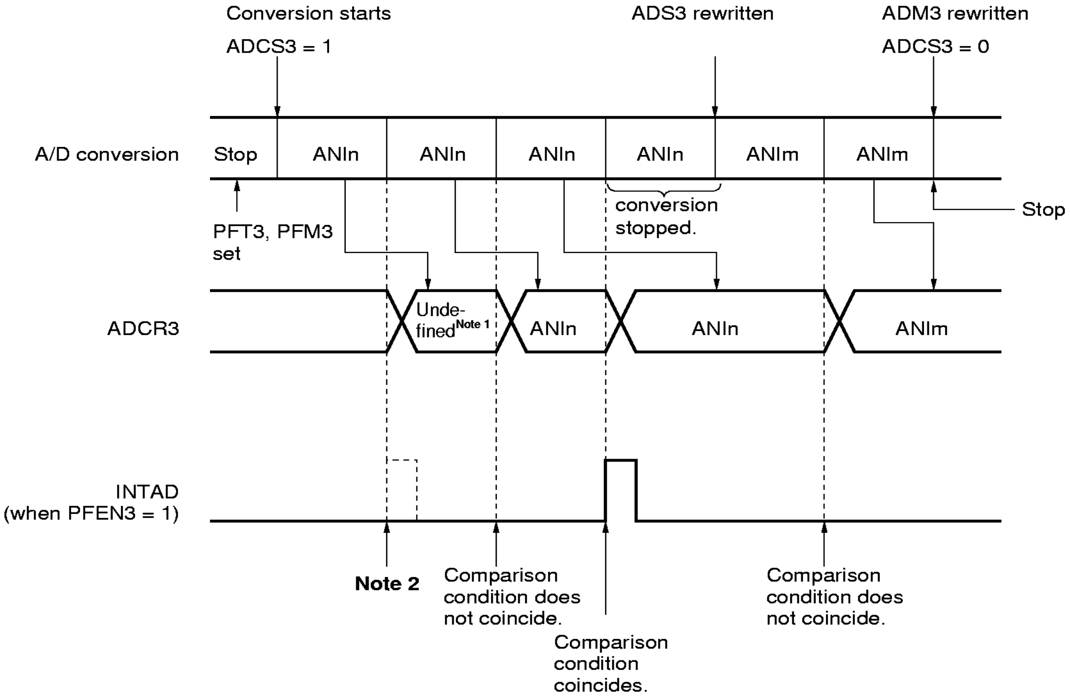
Cautions 1. In the power-fail comparison mode, the first result (A/D conversion result and interrupt request (INTAD)) of the A/D conversion (started by setting bit 7 (ADCS3) of the A/D converter mode register 3 (ADM3) to 1) is not correct.

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 the 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 the A/D conversion result register 3 (ADCR3). If bit 5 (PFHRM3) of PFM3 is set in the normal operating mode (other than HALT Mode), 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 the A/D converter mode register 3 (ADM3) is set to 1 (to start conversion).

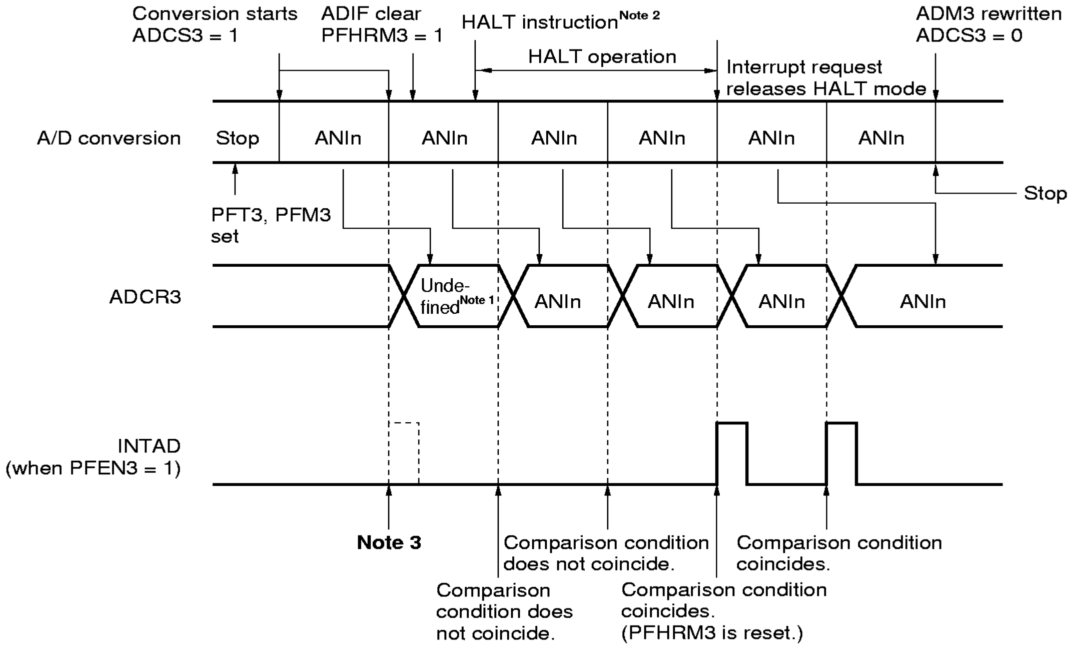
2. The first result of the A/D conversion (A/D conversion result and interrupt request) is not correct. Do not use this result because there is a possibility that it will be determined that the comparison condition has coincided even if it has not.

Caution Set the 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, ... 5
m = 0, 1, ... 5

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 the 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 the power-fail comparison mode register 3 (PFM3) has been set to 1, before executing the HALT instruction.

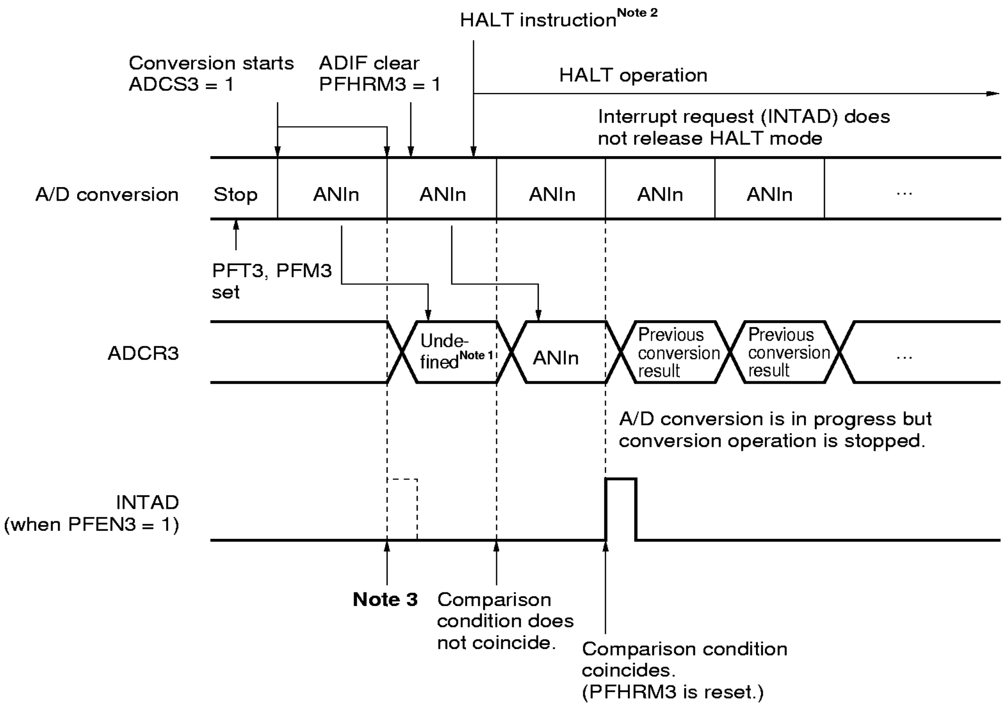
3. The first result of the A/D conversion (A/D conversion result and interrupt request) is not correct. Do not use this result because there is a possibility that it will be determined that the comparison condition has coincided even if it has not.

Caution Be sure to set bit 5 (PFHRM3) of PFM3 to 1 (to enable the HALT repeat mode setting).

Remark n = 0, 1, ... 5

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)


HALT instruction[Note 2]
Conversion starts
ADCS3 = 1
ADIF clear
PFHRM3 = 1
HALT operation
Interrupt request (INTAD) does not release HALT mode
A/D conversion
Stop
ANIn
ANIn
ANIn
ANIn
ANIn
...
PFT3, PFM3 set
ADCR3
Undefined[Note 1]
ANIn
Previous conversion result
Previous conversion result
...
A/D conversion is in progress but conversion operation is stopped.
INTAD
(when PFEN3 = 1)
Note 3
Comparison condition does not coincide.
Comparison condition coincides.
(PFHRM3 is reset.)

Notes 1. The conversion data is undefined immediately after bit 7 (ADCS3) of the 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 the power-fail comparison mode register 3 (PFM3) has been set to 1, before executing the HALT instruction.

3. The first result of the 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 coincided 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, ... 5

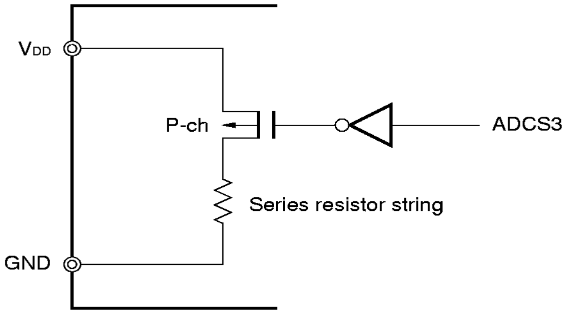
11.5 Notes on A/D Converter

(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 stopping the conversion operation (by resetting bit 7 (ADCS3) of the A/D converter mode register 3 (ADM3) to 0).

Figure 11-10 shows how to reduce the current consumption in the standby mode.

Figure 11-10. Example of Reducing Current Consumption in Standby Mode



(2) Input range of ANI0 to ANI5

The input voltages of ANI0 to ANI5 should be within the specification range. In particular, if a voltage above V_{DD} or below GND is input (even if within the absolute maximum rating range), the conversion value for that channel will be indeterminate. 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 conversion completion interrupt request signal (INTAD) generated.

(4) ANI0 to ANI5

The analog input pins ANI0 to ANI5 also function as input port (P10 to P15) pins.

When A/D conversion is performed with any of pins ANI0 to ANI5 selected, be sure not to execute a PORT1 input instruction 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.

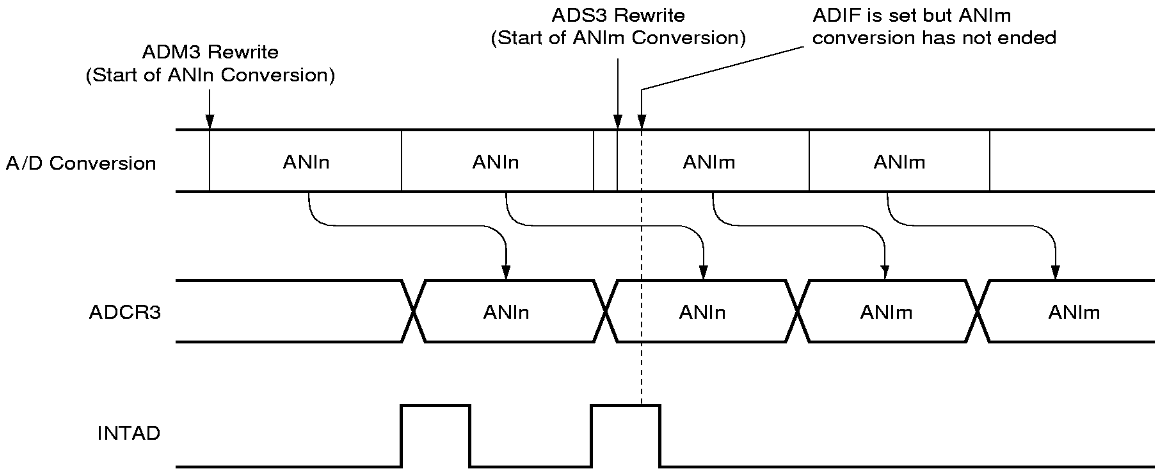
(5) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even if the analog input channel specification register 3 (ADS3) is changed.

Caution is therefore required since, if a change of analog input pin is performed 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 the A/D conversion is stopped and then resumed, clear the ADIF before it is resumed.

Figure 11-11. A/D Conversion End Interrupt Request Generation Timing


ADS3 Rewrite
(Start of ANIm Conversion)
ADIF is set but ANIm conversion has not ended
ADM3 Rewrite
(Start of ANIn Conversion)
A/D Conversion
ANIn
ANIn
ANIm
ANIm
ADCR3
ANIn
ANIn
ANIm
ANIm
INTAD

Remarks 1. n = 0, 1, ..., 5

2. m = 0, 1, ..., 5

(6) Conversion result immediately after starting A/D conversion

The first A/D conversion result value is undefined immediately after the A/D conversion operation has been started. Poll the A/D conversion completion interrupt request (INTAD) and discard the first conversion result.

(7) Reading A/D conversion result register 3 (ADCR3)

If data is written to the A/D converter mode register 3 (ADM3) and analog input channel specification register 3 (ADS3), the contents of ADCR3 can be 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.

[MEMO]

CHAPTER 12 SERIAL INTERFACE (IIC0)

12.1 Function of Serial Interface (IIC0)

The serial interface (IIC0) has the following two modes:

(1) Operation stop mode

This mode is used when serial transfer is not performed. It can therefore be used to reduce power consumption.

(2) I^2C bus mode (multimaster supported)

This mode is used for 8-bit data transfers with several devices via two lines: a serial clock line (SCL0) and a serial data bus line (SDA0).

This mode complies with the I^2C bus format and can output “start condition”, “data”, and “stop condition” data segments when transmitting via the serial data bus. These data segments are automatically detected by hardware during reception.

Since SCL0 and SDA0 are open-drain outputs, the IIC0 requires pull-up resistors for the serial clock line (SCL0) and the serial data bus line (SDA0).

Figure 12-1 shows the block diagram of the serial interface (IIC0).

Figure 12-1. Block Diagram of Serial Interface (IIC0)

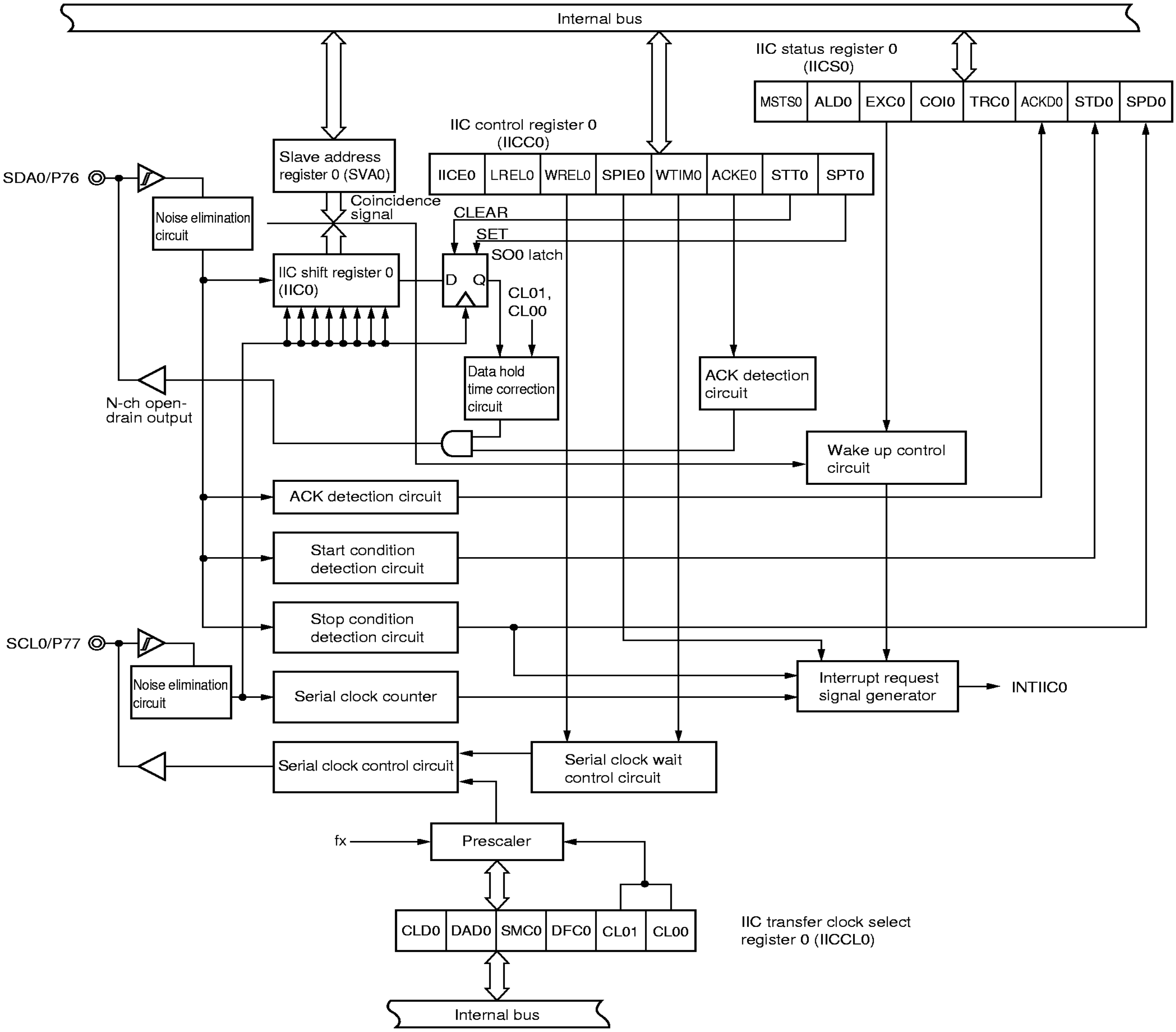
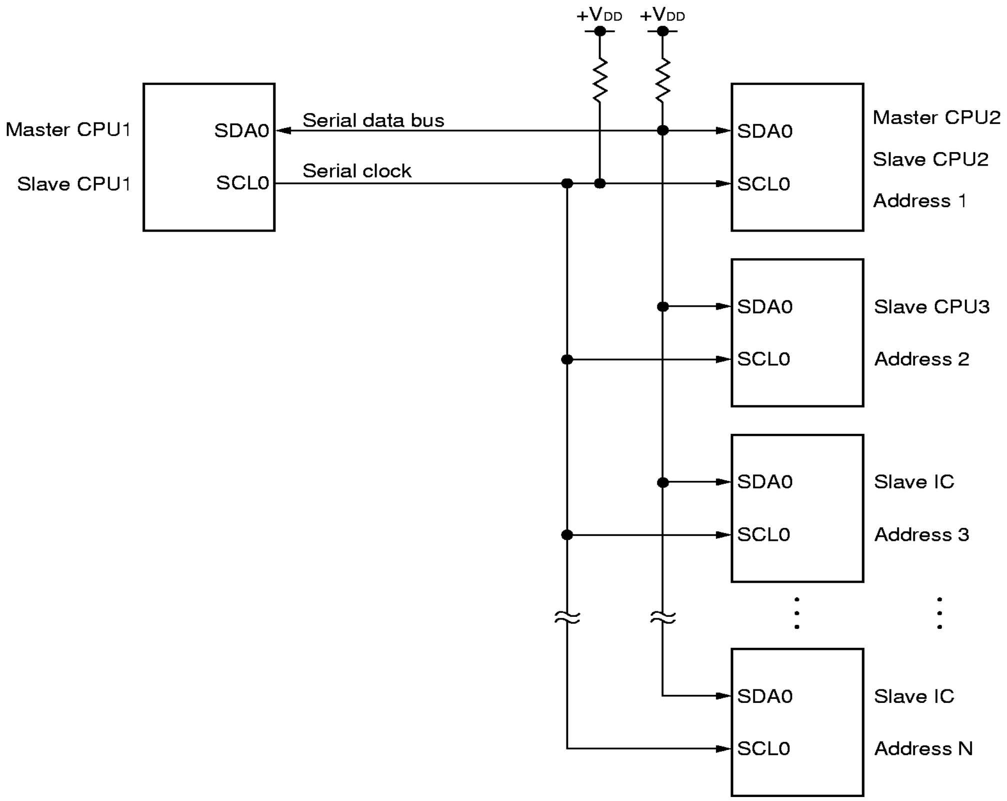

Internal bus
IIC status register 0 (IICS0)
MSTS0
ALD0
EXC0
COI0
TRC0
ACKD0
STD0
SPD0
IIC control register 0 (IICC0)
IICE0
LREL0
WREL0
SPIE0
WTIM0
ACKE0
STT0
SPT0
SDA0/P76
Slave address register 0 (SVA0)
Noise elimination circuit
Coincidence signal
CLEAR
SET
SO0 latch
IIC shift register 0 (IIC0)
D
Q
CL01, CL00
Data hold time correction circuit
ACK detection circuit
Wake up control circuit
N-ch open-drain output
ACK detection circuit
Start condition detection circuit
Stop condition detection circuit
SCL0/P77
Noise elimination circuit
Serial clock counter
Interrupt request signal generator
INTIIC0
Serial clock control circuit
Serial clock wait control circuit
fx
Prescaler
CLD0
DAD0
SMC0
DFC0
CL01
CL00
IIC transfer clock select register 0 (IICCL0)
Internal bus

Figure 12-2 shows a serial bus configuration example.

Figure 12-2. Serial Bus Configuration Example Using I²C Bus



12.2 Configuration of Serial Interface (IIC0)

The serial interface (IIC0) consists of the following hardware.

Table 12-1. Configuration of Serial Interface (IIC0)

Item	Configuration
Registers	IIC shift register 0 (IIC0) Slave address register 0 (SVA0)
Control registers	IIC control register 0 (IICC0) IIC status register 0 (IICS0) IIC transfer clock select register 0 (IICCL0)

(1) IIC shift register 0 (IIC0)

IIC0 is used to convert 8-bit serial data to 8-bit parallel data and to convert 8-bit parallel data to 8-bit serial data.
IIC0 can be used for both transmission and reception.
Write and read operations to IIC0 are used to control the actual transmit and receive operations.
IIC0 is set by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets the IIC0 00H.

(2) Slave address register 0 (SVA0)

This register sets local addresses when in slave mode.
SVA0 is set by an 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input mades SVA0 undefined.

(3) SO0 latch

The SO0 latch is used to retain the SDA0 pin’s output level.
This can be controlled directly in software.

(4) Wake-up control circuit

This circuit generates an interrupt request when the address received by this register matches the address value set to the slave address register 0 (SVA0) or when an extension code is received.

(5) Clock selector

This selects the sampling clock to be used.

(6) Serial clock counter

This counter counts the serial clocks that are output or input during transmit/receive operations and is used to verify that 8-bit data was sent or received.

(7) Interrupt request signal generator

This circuit controls the generation of interrupt request signals (INTIIC0).
An I^{2}C interrupt request is generated following either of two triggers.

- Eighth or ninth clock of the serial clock (set by WTIM0 bit) **Note**
- Interrupt request generated when a stop condition is detected (set by SPIE0 bit) **Note**

Note WTIM0 bit : bit 3 of the IIC control register 0 (IICC0)
SPIE0 bit : bit 4 of the IIC control register 0 (IICC0)

(8) Serial clock control circuit

During master mode, this circuit generates the clock output via the SCL0 pin from a sampling clock.

(9) Serial clock wait control circuit

This circuit controls the wait timing.

(10) ACK output circuit, stop condition detection circuit, start condition detection circuit, and ACK detection circuit

These circuits are used to output and detect various control signals.

(11) Data hold time correction circuit

This circuit generates the hold time for data corresponding to the falling edge of the serial clock.

12.3 Registers Controlling Serial Interface (IIC0)

The following registers control the serial interface (IIC0):

- IIC control register 0 (IICC0)
- IIC status register 0 (IICS0)
- IIC transfer clock select register 0 (IICCL0)

The following registers are also used.

- IIC shift register 0 (IIC0)
- Slave address register 0 (SVA0)

(1) IIC control register 0 (IICC0)

This register is used to enable/disable I^2C operations, set wait timing, and set other I^2C operations.
IICC0 can be set by a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets IICC0 to 00H.

Caution In I^2C bus mode, set the port mode register (PM××) as follows. Set the output latch to 0.
- **Set P76 (SDA0) to output mode (PM76 = 0)**
- **Set P77 (SCL0) to output mode (PM77 = 0)**

Figure 12-3. Format of IIC Control Register 0 (IICC0) (1/3)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	At reset	R/W
IICC0	IICE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0	FF61H	00H	R/W

<table>
<tr><th>IICE0</th><th colspan="2">I^{2}C Operation Enable</th></tr>
<tr><td>0</td><td colspan="2">Stops operation. Presets IIC status register 0 (IICS0). Stops internal operation.</td></tr>
<tr><td>1</td><td colspan="2">Enables operation.</td></tr>
<tr><td colspan="2">Condition for clearing (IICE0 = 0)</td><td>Condition for setting (IICE0 = 1)</td></tr>
<tr><td colspan="2">• Cleared by instruction
• When $\overline{\text{RESET}}$ is input</td><td>• Set by instruction</td></tr>
</table>

<table>
<tr><th>LREL0</th><th colspan="2">Exit from Communications</th></tr>
<tr><td>0</td><td colspan="2">Normal operation</td></tr>
<tr><td>1</td><td colspan="2">This exits from the current communications operation and sets standby mode. This setting is automatically cleared after being executed. Its uses include cases in which a locally irrelevant extension code has been received.
The SCL0 and SDA0 lines go into the high impedance state.
The following flags are cleared.
• STD0 • ACKD0 • TRC0 • COI0 • EXC0 • MSTS0 • STT0 • SPT0</td></tr>
<tr><td colspan="3">The standby mode following exit from communications remains in effect until the following communications entry conditions are met.
• After a stop condition is detected, restart is in master mode.
• An address match or extension code reception occurs after the start condition.</td></tr>
<tr><td colspan="2">Condition for clearing (LREL0 = 0) Note</td><td>Condition for setting (LREL0 = 1)</td></tr>
<tr><td colspan="2">• Automatically cleared after execution
• When $\overline{\text{RESET}}$ is input</td><td>• Set by instruction</td></tr>
</table>

<table>
<tr><th>WREL0</th><th colspan="2">Cancel Wait</th></tr>
<tr><td>0</td><td colspan="2">Does not cancel wait</td></tr>
<tr><td>1</td><td colspan="2">Cancels wait. This setting is automatically cleared after wait is canceled.</td></tr>
<tr><td colspan="3">When WREL0 is set (wait released) at the 9th clock pulse during the wait period in transmission status (TRC0 = 1), the SDA0 line goes into a high impedance state (TRC0 = 0).</td></tr>
<tr><td colspan="2">Condition for clearing (WREL0 = 0) Note</td><td>Condition for setting (WREL0 = 1)</td></tr>
<tr><td colspan="2">• Automatically cleared after execution
• When $\overline{\text{RESET}}$ is input</td><td>• Set by instruction</td></tr>
</table>

<table>
<tr><th>SPIE0</th><th colspan="2">Enable/Disable Generation of Interrupt Request When Stop Condition Is Detected</th></tr>
<tr><td>0</td><td colspan="2">Disable</td></tr>
<tr><td>1</td><td colspan="2">Enable</td></tr>
<tr><td colspan="2">Condition for clearing (SPIE0 = 0) Note</td><td>Condition for setting (SPIE0 = 1)</td></tr>
<tr><td colspan="2">• Cleared by instruction
• When $\overline{\text{RESET}}$ is input</td><td>• Set by instruction</td></tr>
</table>

Note This flag's signal is invalid when IICE0 = 0.

Figure 12-3. Format of IIC Control Register 0 (IICC0) (2/3)

<table>
<tr><th>WTIM0</th><th colspan="2">Control of Wait and Interrupt Request Generation</th></tr>
<tr><td>0</td><td colspan="2">Interrupt request is generated at the eighth clock's falling edge.
Master mode : After output of eight clocks, clock output is set to low level and wait is set.
Slave mode : After input of eight clocks, the clock is set to low level and wait is set for master device.</td></tr>
<tr><td>1</td><td colspan="2">Interrupt request is generated at the ninth clock's falling edge.
Master mode : After output of nine clocks, clock output is set to low level and wait is set.
Slave mode : After input of nine clocks, the clock is set to low level and wait is set for master device.</td></tr>
<tr><td colspan="3">This bit's setting is invalid during an address transfer and is valid after the transfer is completed. When in master mode, a wait is inserted at the falling edge of the ninth clock during address transfers. For a slave device that has received a local address, a wait is inserted at the falling edge of the ninth clock after an ACK signal is issued. When the slave device has received an extension code, a wait is inserted at the falling edge of the eighth clock.</td></tr>
<tr><td colspan="2">Condition for clearing (WTIM0 = 0) Note</td><td>Condition for setting (WTIM0 = 1)</td></tr>
<tr><td colspan="2">• Cleared by instruction
• When $\overline{\text{RESET}}$ is input</td><td>• Set by instruction</td></tr>
</table>

<table>
<tr><th>ACKE0</th><th colspan="2">Acknowledge Control</th></tr>
<tr><td>0</td><td colspan="2">Disable acknowledge.</td></tr>
<tr><td>1</td><td colspan="2">Enable acknowledge. During the ninth clock period, the SDA0 line is set to low level. However, the ACK is invalid during address transfers and is valid when EXC0 = 1.</td></tr>
<tr><td colspan="2">Condition for clearing (ACKE0 = 0) Note</td><td>Condition for setting (ACKE0 = 1)</td></tr>
<tr><td colspan="2">• Cleared by instruction
• When $\overline{\text{RESET}}$ is input</td><td>• Set by instruction</td></tr>
</table>

<table>
<tr><th>STT0</th><th colspan="2">Start Condition Trigger</th></tr>
<tr><td>0</td><td colspan="2">Does not generate a start condition.</td></tr>
<tr><td>1</td><td colspan="2">When bus is released (during STOP mode):
Generates a start condition (for starting as master). The SDA0 line is changed from high level to low level and then the start condition is generated. Next, after the rated amount of time has elapsed, SCL0 is changed to low level.
When bus is not used:
This trigger functions as a start condition reserve flag. When set, it releases the bus and then automatically generates a start condition.
Wait status (during master mode):
Generates a restart condition after wait is released.</td></tr>
<tr><td colspan="3">Cautions concerning set timing
• For master reception : Cannot be set during transfer. Can be set only at the waiting period when ACKE0 has been set to 0 and slave has been notified of final reception.
• For master transmission : A start condition may not be generated normally during the ACK period. Therefore, set it during the waiting period.
• Cannot be set at the same time as SPT0</td></tr>
<tr><td colspan="2">Condition for clearing (STT0 = 0) Note</td><td>Condition for setting (STT0 = 1)</td></tr>
<tr><td colspan="2">• Cleared by instruction
• Cleared by loss in arbitration
• Cleared after start condition is generated by master device
• When LREL0 = 1
• When $\overline{\text{RESET}}$ is input</td><td>• Set by instruction</td></tr>
</table>

Note This flag's signal is invalid when IICE0 = 0.

Figure 12-3. Format of IIC Control Register 0 (IICC0) (3/3)

<table>
<tr><th>SPT0</th><th colspan="2">Stop Condition Trigger</th></tr>
<tr><td>0</td><td colspan="2">Stop condition is not generated.</td></tr>
<tr><td>1</td><td colspan="2">Stop condition is generated (termination of master device's transfer).
After the SDA0 line goes to low level, either set the SCL0 line to high level or wait until it goes to high level. Next, after the rated amount of time has elapsed, the SDA0 line changes from low level to high level and a stop condition is generated.</td></tr>
<tr><td colspan="3">Cautions concerning set timing
• For master reception : Cannot be set during transfer. Can be set only at the waiting period when ACKE0 has been set to 0 and slave has been notified of final reception.
• For master transmission: A stop condition cannot be generated normally during the ACK0 period. Therefore, set it during the waiting period.
• Cannot be set at the same time as STT0.
• SPT0 can be set only when in master mode. [Note 1]
• When WTIM0 has been set to 0, if SPT0 is set during the wait period that follows output of eight clocks, note that a stop condition will be generated during the high level period of the ninth clock.
When a ninth clock must be output, WTIM0 should be changed from 0 to 1 during the wait period following output of eight clocks, and SPT0 should be set during the wait period that follows output of the ninth clock.</td></tr>
<tr><td colspan="2">Condition for clearing (SPT0 = 0) [Note 2]</td><td>Condition for setting (SPT0 = 1)</td></tr>
<tr><td colspan="2">• Cleared by instruction
• Cleared by loss in arbitration
• Automatically cleared after stop condition is detected
• When LREL0 = 1
• When $\overline{\text{RESET}}$ is input</td><td>• Set by instruction</td></tr>
</table>

Notes 1. Set SPT0 only during master mode. However, you must set SPT0 and generate a stop condition before the first stop condition is detected following the switch to operation enable status. For details, refer to **12.5.15 Other cautions**.

2. This flag's signal is invalid when IICE0 = 0.

Caution **When bit 3 (TRC0) of the IIC status register 0 (IICS0) is set to "1", WREL0 is set during the ninth clock and wait is canceled, after which TRC0 is cleared and the SDA0 line is set for high impedance.**

Remarks 1. STD0 : Bit 1 of IIC status register 0 (IICS0)
ACKD0 : Bit 2 of IIC status register 0 (IICS0)
TRC0 : Bit 3 of IIC status register 0 (IICS0)
COI0 : Bit 4 of IIC status register 0 (IICS0)
EXC0 : Bit 5 of IIC status register 0 (IICS0)
MSTS0 : Bit 7 of IIC status register 0 (IICS0)

2. Bits 0 and 1 (SPT0, STT0) become 0 when they are read after data setting.

(2) IIC status register 0 (IICS0)

This register indicates the status of the I^2C.

IICS0 can be set by a 1-bit or 8-bit memory manipulation instruction. IICS0n is a read-only register.

$\overline{\text{RESET}}$ input sets the value to 00H.

Figure 12-4. Format of IIC Status Register 0 (IICS0) (1/3)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	At reset	R/W
IICS0	MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0	FF60H	00H	R/W

MSTS0	Master Device Status
0	Slave device status or communication standby status
1	Master device communication status

Condition for clearing (MSTS0 = 0)	Condition for setting (MSTS0 = 1)
• When a stop condition is detected • When ALD0 = 1 • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When a start condition is generated

ALD0	Detection of Arbitration Loss
0	This status means either that there was no arbitration or that the arbitration result was a "win".
1	This status indicates the arbitration result was a "loss". MSTS0 is cleared.

Condition for clearing (ALD0 = 0)	Condition for setting (ALD0 = 1)
• Automatically cleared after IICS0 is read Note • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When the arbitration result is a "loss".

EXC0	Detection of Extension Code Reception
0	Extension code was not received.
1	Extension code was received.

Condition for clearing (EXC0 = 0)	Condition for setting (EXC0 = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When the high-order four bits of the received address data is either "0000" or "1111" (set at the rising edge of the eighth clock).

Note This register is also cleared when a bit manipulation instruction is executed for bits other than IICS0.

Figure 12-4. Format of IIC Status Register 0 (IICS0) (2/3)

COI0	Detection of Matching Addresses
0	Addresses do not match.
1	Addresses match.
Condition for clearing (COI0 = 0)	Condition for setting (COI0 = 1)
• When a start condition is detected • When a stop condition is detected • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When the received address matches the local address (SVA0) (set at the rising edge of the eighth clock).

TRC0	Detection of Transmit/Receive Status
0	Receive status (other than transmit status). The SDA0 line is set for high impedance.
1	Transmit status. The value in the SO0 latch is enabled for output to the SDA0 line (valid starting at the rising edge of the first byte's ninth clock).
Condition for clearing (TRC0 = 0)	Condition for setting (TRC0 = 1)
• When a stop condition is detected • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • Cleared by WREL0 = 1[Note] • When ALD0 changes from 0 to 1 • When $\overline{\text{RESET}}$ is input Master • When "1" is output to the first byte's LSB (transfer direction specification bit) Slave • When a start condition is detected • When "0" is output to the first byte's LSB (transfer direction specification bit) When not used for communication	Master • When a start condition is generated Slave • When "1" is input by the first byte's LSB (transfer direction specification bit)

ACKD0	Detection of ACK
0	ACK was not detected.
1	ACK was detected.
Condition for clearing (ACKD0 = 0)	Condition for setting (ACKD0 = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• After the SDA0 line is set to low level at the rising edge of the SCL0's ninth clock

Note Cleared by setting WREL to 0 during ninth clock wait period.

Figure 12-4. Format of IIC Status Register 0 (IICS0) (3/3)

STD0	Detection of Start Condition
0	Start condition was not detected.
1	Start condition was detected. This indicates that the address transfer period is in effect

Condition for clearing (STD0 = 0)	Condition for setting (STD0 = 1)
• When a stop condition is detected • At the rising edge of the next byte's first clock following address transfer • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When a start condition is detected

SPD0	Detection of Stop Condition
0	Stop condition was not detected.
1	Stop condition was detected. The master device's communication is terminated and the bus is released.

Condition for clearing (SPD0 = 0)	Condition for setting (SPD0 = 1)
• At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input	• When a stop condition is detected

Remark LREL0 : Bit 6 of IIC control register 0 (IICC0)
IICE0 : Bit 7 of IIC control register 0 (IICC0)

(3) IIC transfer clock select register 0 (IICCL0)

This register is used to set the transfer clock for the I^{2}C bus.

IICCL0 can be set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets IICCL0 to 00H.

Figure 12-5. Format of IIC Transfer Clock Select Register 0 (IICCL0) (1/2)

Symbol	7	6	<5>	<4>	3	2	1	0	Address	At reset	R/W
IICCL0	0	0	CLD0	DAD0	SMC0	DFC0	CL01	CL00	FF43H	00H	R/W$^{\text{Note}}$

<table>
<tr><th>CLD0</th><th colspan="2">Detection of SCL0 Line Level (valid only when IICE0 = 1)</th></tr>
<tr><td>0</td><td colspan="2">SCL0 line was detected at low level.</td></tr>
<tr><td>1</td><td colspan="2">SCL0 line was detected at high level.</td></tr>
<tr><td colspan="2">Condition for clearing (CLD0 = 0)</td><td>Condition for setting (CLD0 = 1)</td></tr>
<tr><td colspan="2">• When the SCL0 line is at low level
• When IICE0 = 0
• When $\overline{\text{RESET}}$ is input</td><td>• When the SCL0 line is at high level</td></tr>
</table>

<table>
<tr><th>DAD0</th><th colspan="2">Detection of SDA0 Line Level (valid only when IICE0 = 1)</th></tr>
<tr><td>0</td><td colspan="2">SDA0 line was detected at low level.</td></tr>
<tr><td>1</td><td colspan="2">SDA0 line was detected at high level.</td></tr>
<tr><td colspan="2">Condition for clearing (DAD0 = 0)</td><td>Condition for setting (DAD0 = 1)</td></tr>
<tr><td colspan="2">• When the SDA0 line is at low level
• When IICE0 = 0
• When $\overline{\text{RESET}}$ is input</td><td>• When the SDA0 line is at high level</td></tr>
</table>

<table>
<tr><th>SMC0</th><th colspan="2">Operation Mode Switching</th></tr>
<tr><td>0</td><td colspan="2">Operation in standard mode</td></tr>
<tr><td>1</td><td colspan="2">Operation in high-speed mode</td></tr>
<tr><td colspan="2">Condition for clearing (SMC0 = 0)</td><td>Condition for setting (SMC0 = 1)</td></tr>
<tr><td colspan="2">• Cleared by instruction
• When $\overline{\text{RESET}}$ is input</td><td>• Set by instruction</td></tr>
</table>

Note Bits 4 and 5 are read-only bits.

Figure 12-5. Format of IIC Transfer Clock Select Register 0 (IICCL0) (2/2)

DFC0	Control of Digital Filter Operation Note
0	Digital filter OFF
1	Digital filter ON

CL01	CL00	Selection of Transfer Rate	
		Standard mode	High-speed mode
0	0	fx/44 (51.1 kHz)	fx/24 (93.8 kHz)
0	1	fx/86 (26.2 kHz)	
1	0	fx/86 (52.3 kHz)	fx/24 (187.5 kHz)
1	1	Setting prohibited	

Note The digital filter can be used when in high-speed mode. Response time is slower when the digital filter is used.

Caution Stop serial transfer once before rewriting CL01 and CL00 to other than the same value.

Remarks 1. IICE0: Bit 7 of IIC control register 0 (IICC0)

2. fx: System clock oscillation frequency

3. Figures in parentheses are for operation with fx = 4.5 MHz.

(4) I²C shift register 0 (IIC0)

This register is used for serial transmission/reception (shift operations) that are synchronized with the serial clock. It can be read from or written to in 8-bit units, but data should not be written to IIC0 during a data transfer.

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
IIC0									FF1AH	Undefined	R/W

(5) Slave address register 0 (SVA0)

This register holds the I²C's slave addresses.

It can be read from or written to in 8-bit units, but bit 0 is fixed to "0".

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
SVA0								0	FF62H	Undefined	R/W

12.4 Functions of I²C Bus Mode

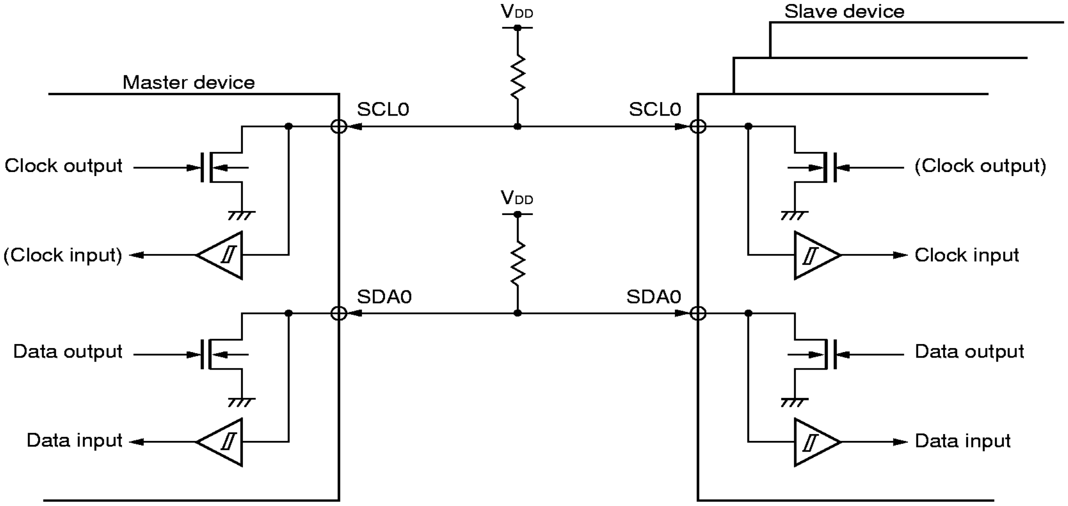
12.4.1 Pin configuration

The serial clock pin (SCL0) and serial data bus pin (SDA0) are configured as follows.

(1) SCL0 ········· This pin is used for serial clock input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.
(2) SDA0 ········· This pin is used for serial data input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

Since outputs from the serial clock line and the serial data bus line are N-ch open drain outputs, an external pull-up resistor is required.

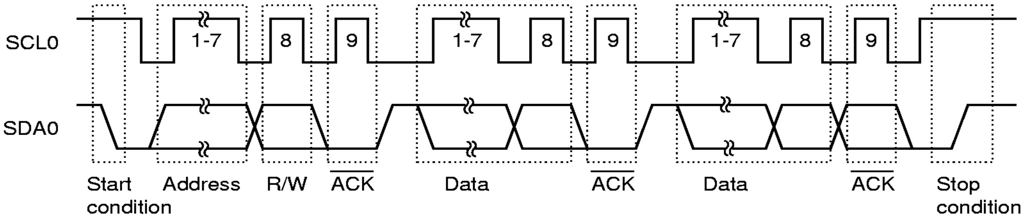
Figure 12-6. Pin Configuration



12.5 I^{2}C Bus Definitions and Control Methods

The following section describes the I^{2}C bus's serial data communication format and the signals used by the I^{2}C bus. Figure 12-7 shows the transfer timing for the "start condition", "data", and "stop condition" output via the I^{2}C bus's serial data bus.

Figure 12-7. I^{2}C Bus's Serial Data Transfer Timing



The master device outputs the start condition, slave address, and stop condition.

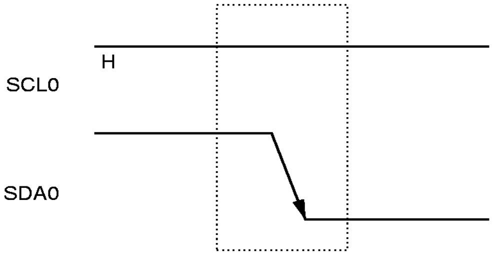
The acknowledge signal ($\overline{\text{ACK}}$) can be output by either the master or slave device (normally, it is output by the device that receives 8-bit data).

The serial clock (SCL0) is continuously output by the master device. However, in the slave device, the SCL0's low level period can be extended and a wait can be inserted.

12.5.1 Start conditions

A start condition is met when the SCL0 pin is at high level and the SDA0 pin changes from high level to low level. The start conditions for the SCL0 pin and SDA0 pin are signals that the master device outputs to the slave device when starting a serial transfer. The slave device includes hardware for detecting start conditions.

Figure 12-8. Start Conditions



A start condition is output when the IIC control register 0 (IICC0)'s bit 1 (STT0) is set (to "1") after a stop condition has been detected (SPD0: Bit 0 = 1 in the IIC status register 0 (IICS0)). When a start condition is detected, IICS0's bit 1 (STD0) is set (to "1").

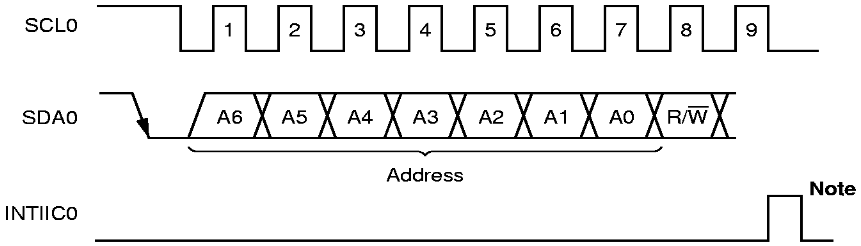
12.5.2 Addresses

The address is defined by the 7 bits of data that follow the start condition.

An address is a 7-bit data segment that is output in order to select one of the slave devices that are connected to the master device via bus lines. Therefore, each slave device connected via the bus lines must have a unique address.

The slave devices include hardware that detects the start condition and checks whether or not the 7-bit address data matches the data values stored in the slave address register 0 (SVA0). If the address data matches the SVA0 values, the slave device is selected and communicates with the master device until the master device transmits a start condition or stop condition.

Figure 12-9. Address



Note INTIIC0 is not issued if data other than a local address or extension code is received during slave device operation.

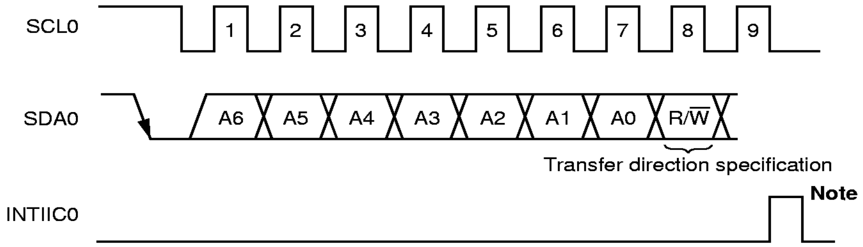
The slave address and the eighth bit, which specifies the transfer direction as described in **12.5.3 Transfer direction specification** below, are together written to the IIC shift register 0 (IIC0) and are then output. Received addresses are written to IIC0.

The slave address is assigned to the high-order 7 bits of IIC0.

12.5.3 Transfer direction specification

In addition to the 7-bit address data, the master device sends 1 bit that specifies the transfer direction. When this transfer direction specification bit has a value of "0", it indicates that the master device is transmitting data to a slave device. When the transfer direction specification bit has a value of "1", it indicates that the master device is receiving data from a slave device.

Figure 12-10. Transfer Direction Specification



Note INTIIC0 is not issued if data other than a local address or extension code is received during slave device operation.

12.5.4 Acknowledge ($\overline{\text{ACK}}$) signal

The acknowledge ($\overline{\text{ACK}}$) signal is used by the transmitting and receiving devices to confirm serial data reception.

The receiving device returns one $\overline{\text{ACK}}$ signal for each 8 bits of data it receives. The transmitting device normally receives an $\overline{\text{ACK}}$ signal after transmitting 8 bits of data. However, when the master device is the receiving device, it does not output an $\overline{\text{ACK}}$ signal after receiving the final data to be transmitted. The transmitting device detects whether or not an $\overline{\text{ACK}}$ signal is returned after it transmits 8 bits of data. When an $\overline{\text{ACK}}$ signal is returned, the reception is judged as normal and processing continues. If the slave device does not return an $\overline{\text{ACK}}$ signal, the master device outputs either a stop condition or a restart condition and then stops the current transmission. Failure to return an $\overline{\text{ACK}}$ signal may be caused by the following two factors.

<1> Reception was not performed normally.
<2> The final data was received.

When the receiving device sets the SDA0 line to low level during the ninth clock, the $\overline{\text{ACK}}$ signal becomes active (normal receive response).

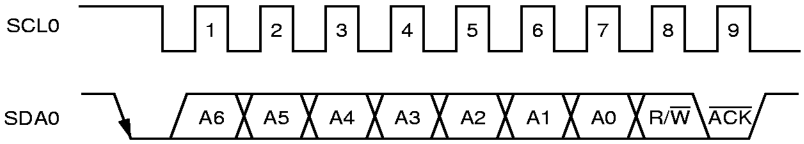
When bit 2 (ACKE0) of the IIC control register 0 (IICC0) is set to 1, automatic $\overline{\text{ACK}}$ signal generation is enabled.

Transmission of the eighth bit following the 7 address data bits causes bit 3 (TRC0) of the IIC status register 0 (IICS0) to be set. When this TRC0 bit's value is "0", it indicates receive mode. Therefore, ACKE0 should be set to "1".

When the slave device is receiving (when TRC0 = 0), if the slave devices does not need to receive any more data after receiving several bytes, setting ACKE0 to "0" will prevent the master device from starting transmission of the subsequent data.

Similarly, when the master device is receiving (when TRC0 = 0) and the subsequent data is not needed and when either a restart condition or a stop condition should therefore be output, setting ACKE0 to "0" will prevent the $\overline{\text{ACK}}$ signal from being returned. This prevents the MSB data from being output via the SDA line (i.e., stops transmission) during transmission from the slave device.

Figure 12-11. $\overline{\text{ACK}}$ Signal



When the local address is received, an $\overline{\text{ACK}}$ signal is automatically output in sync with the falling edge of the SCL's eighth clock regardless of the ACKE0 value. No $\overline{\text{ACK}}$ signal is output if the received address is not a local address.

The $\overline{\text{ACK}}$ signal output method during data reception is based on the wait timing setting, as described below.

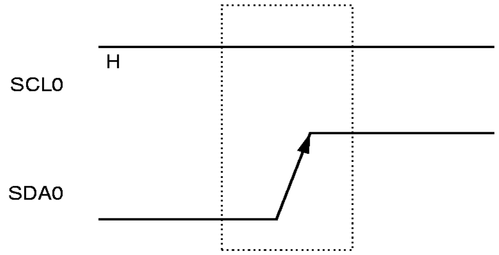
- When 8-clock wait is selected : $\overline{\text{ACK}}$ signal is output when ACKE0 is set to "1" before wait cancellation.
- When 9-clock wait is selected : $\overline{\text{ACK}}$ signal is automatically output at the falling edge of the SCL0's eighth clock if ACKE0 has already been set to "1".

12.5.5 Stop condition

When the SCL0 pin is at high level, changing the SDA0 pin from low level to high level generates a stop condition.

A stop condition is a signal that the master device outputs to the slave device when serial transfer has been completed. The slave device includes hardware that detects stop conditions.

Figure 12-12. Stop Condition



A stop condition is generated when bit 0 (SPT0) of the IIC control register 0 (IICC0) is set (to “1”). When the stop condition is detected, bit 0 (SPD0) of the IIC status register 0 (IICS0) is set (to “1”) and INTIIC0 is generated when bit 4 (SPIE0) of IICC0 is set (to “1”).

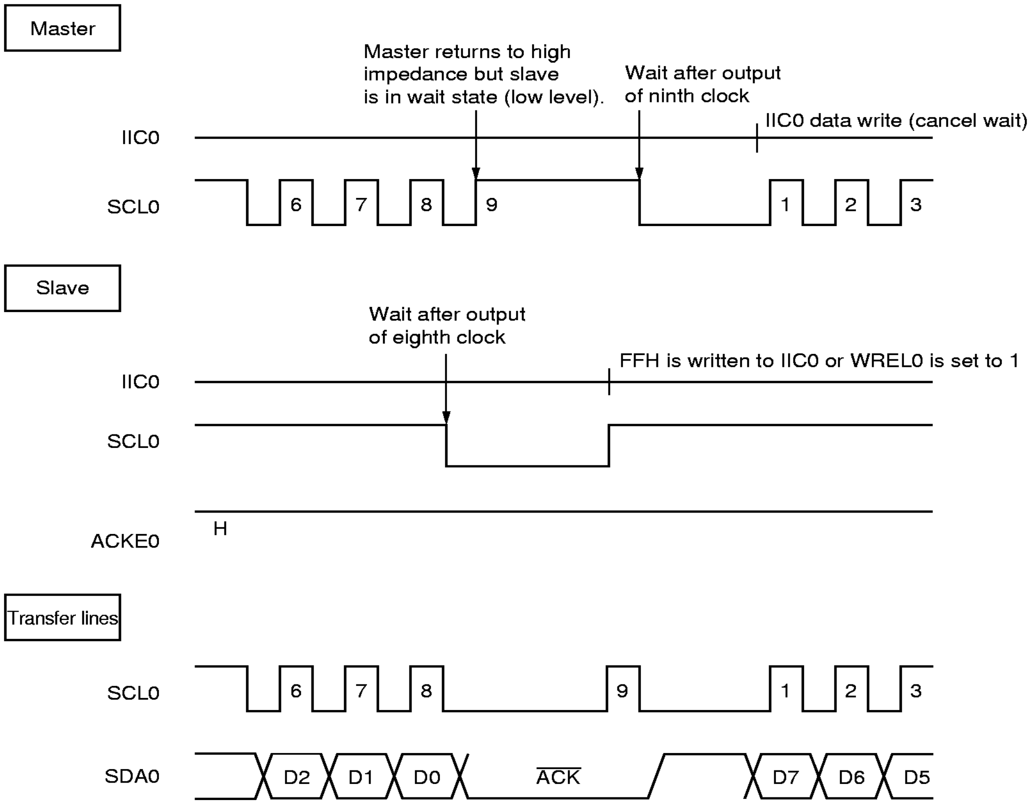
12.5.6 Wait signal ($\overline{\text{WAIT}}$)

The wait signal ($\overline{\text{WAIT}}$) is used to notify the communication partner that a device (master or slave) is preparing to transmit or receive data (i.e., is in a wait state).

Setting the SCL0 pin to low level notifies the communication partner of the wait status. When wait status has been canceled for both the master and slave devices, the next data transfer can begin.

Figure 12-13. Wait Signal (1/2)

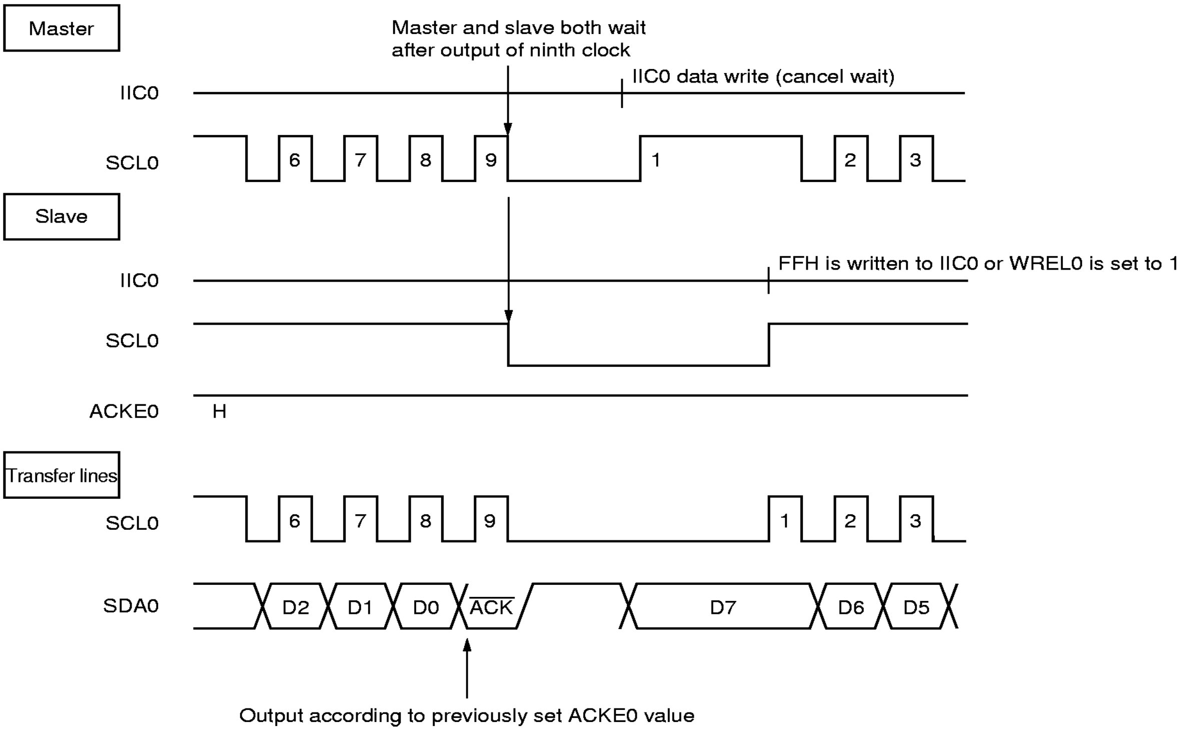
(1) When master device has a nine-clock wait and slave device has an eight-clock wait (master transmits, slave receives, and ACKE0 = 1)



Master
Master returns to high impedance but slave is in wait state (low level).
Wait after output of ninth clock
IIC0 data write (cancel wait)
IIC0
SCL0
6
7
8
9
1
2
3
Slave
Wait after output of eighth clock
FFH is written to IIC0 or WREL0 is set to 1
IIC0
SCL0
ACKE0
H
Transfer lines
SCL0
6
7
8
9
1
2
3
SDA0
D2
D1
D0
ACK
D7
D6
D5

Figure 12-13. Wait Signal (2/2)

(2) When master and slave devices both have a nine-clock wait
(master device transmits, slave receives, and ACKE0 = 1)


Master
Master and slave both wait after output of ninth clock
IIC0
IIC0 data write (cancel wait)
SCL0
6
7
8
9
1
2
3
Slave
IIC0
FFH is written to IIC0 or WREL0 is set to 1
SCL0
ACKE0
H
Transfer lines
SCL0
6
7
8
9
1
2
3
SDA0
D2
D1
D0
$\overline{\text{ACK}}$
D7
D6
D5
Output according to previously set ACKE0 value

Remark ACKE0 : Bit 2 of IIC control register (IICC0)
WREL0 : Bit 5 of IIC control register (IICC0)

A wait may be automatically generated depending on the setting for bit 3 (WTIM0) of the IIC control register 0 (IICC0).

Normally, when bit 5 (WREL0) of IICC0 is set to "1" or when FFH is written to the IIC shift register 0 (IIC0), the wait status is canceled and the transmitting side write data to IIC0 to cancel the wait status.

The master device can also cancel the wait status via either of the following methods.

- By setting bit 1 (STT0) of IICC0 to "1"
- By setting bit 0 (SPT0) of IICC0 to "1"

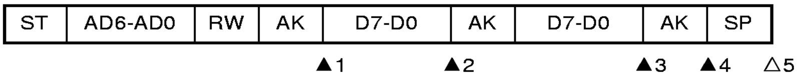
12.5.7 I^{2}C interrupt requests (INTIIC0)

The INTIIC0 interrupt request timing and the IIC status register 0 (IICS0) settings corresponding to that timing are described below.

(1) Master device operation

(a) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)

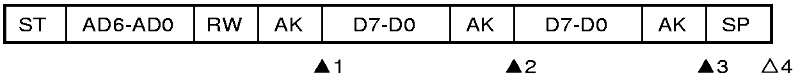
(i) When WTIM0 = 0



▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000×000B
▲3 : IICS0 = 1000×000B (Sets WTIM0)
▲4 : IICS0 = 1000××00B (Sets SPT0)
△5 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1



▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000×100B
▲3 : IICS0 = 1000××00B (Sets SPT0)
△4 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

(i) When WTIM0 = 0

ST	AD6-AD0	RW	AK	D7-D0	AK	ST	AD6-AD0	RW	AK	D7-D0	AK	SP
			▲1	▲2	▲3				▲4	▲5	▲6	△7

▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000×000B (Sets WTIM0)
▲3 : IICS0 = 1000××00B (Clears WTIM0, sets STT0)
▲4 : IICS0 = 1000×110B
▲5 : IICS0 = 1000×000B (Sets WTIM0)
▲6 : IICS0 = 1000××00B (Sets SPT0)
△7 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1

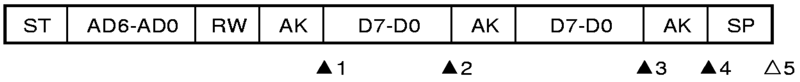
ST	AD6-AD0	RW	AK	D7-D0	AK	ST	AD6-AD0	RW	AK	D7-D0	AK	SP
			▲1		▲2				▲3		▲4	△5

▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000××00B
▲3 : IICS0 = 1000×110B
▲4 : IICS0 = 1000××00B (Sets SPT0)
△5 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(c) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

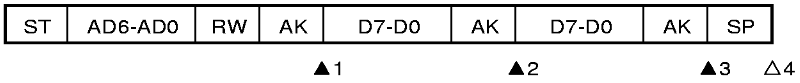
(i) When WTIM0 = 0



▲1 : IICS0 = 1010×110B
▲2 : IICS0 = 1010×000B
▲3 : IICS0 = 1010×000B (Sets WTIM0)
▲4 : IICS0 = 1010××00B (Sets SPT0)
△5 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1



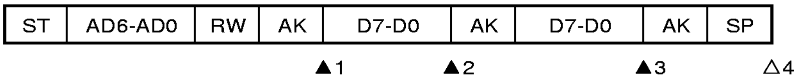
▲1 : IICS0 = 1010×110B
▲2 : IICS0 = 1010×100B
▲3 : IICS0 = 1010××00B (Sets SPT0)
△4 : IICS0 = 00001001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(2) Slave device operation (Slave address data reception time (matches with SVA0))

(a) Start ~ Address ~ Data ~ Data ~ Stop

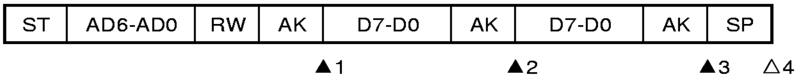
(i) When WTIM0 = 0



▲1 : IICS0 = 0001×110B
▲2 : IICS0 = 0001×000B
▲3 : IICS0 = 0001×000B
△4 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1

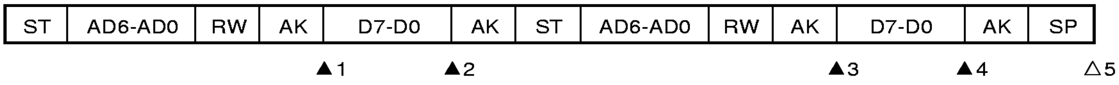


▲1 : IICS0 = 0001×110B
▲2 : IICS0 = 0001×100B
▲3 : IICS0 = 0001××00B
△4 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

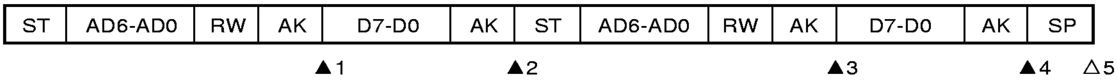
(i) When WTIM0 = 0 (after restart, matches with SVA0)



▲1 : IICS0 = 0001×110B
▲2 : IICS0 = 0001×000B
▲3 : IICS0 = 0001×110B
▲4 : IICS0 = 0001×000B
△5 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don’t care

(ii) When WTIM0 = 1 (after restart, matches with SVA0)

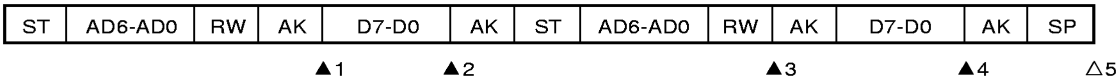


▲1 : IICS0 = 0001×110B
▲2 : IICS0 = 0001××00B
▲3 : IICS0 = 0001×110B
▲4 : IICS0 = 0001××00B
△5 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don’t care

(c) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

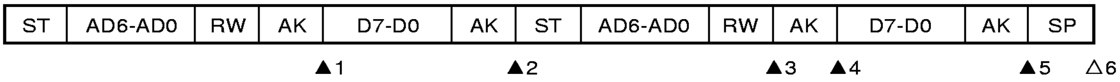
(i) When WTIM0 = 0 (after restart, extension code reception)



▲1 : IICS0 = 0001×110B
▲2 : IICS0 = 0001×000B
▲3 : IICS0 = 0010×010B
▲4 : IICS0 = 0010×000B
△5 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1 (after restart, extension code reception)

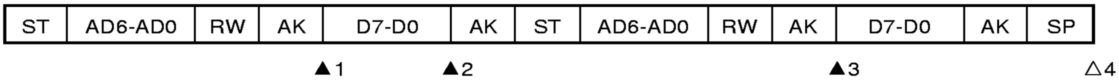


▲1 : IICS0 = 0001×110B
▲2 : IICS0 = 0001××00B
▲3 : IICS0 = 0010×010B
▲4 : IICS0 = 0010×110B
▲5 : IICS0 = 0010××00B
△6 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(d) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

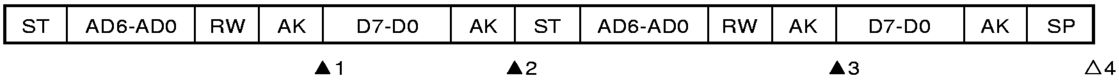
(i) When WTIM0 = 0 (after restart, does not match with address (= not extension code))



▲1 : IICS0 = 0001×110B
▲2 : IICS0 = 0001×000B
▲3 : IICS0 = 00000×10B
△4 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1 (after restart, does not match with address (= not extension code))



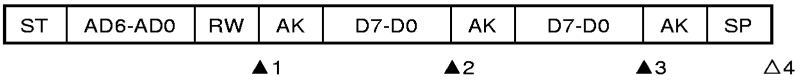
▲1 : IICS0 = 0001×110B
▲2 : IICS0 = 0001××00B
▲3 : IICS0 = 00000×10B
△4 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(3) Slave device operation (when receiving extension code)

(a) Start ~ Code ~ Data ~ Data ~ Stop

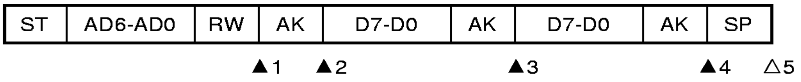
(i) When WTIM0 = 0



▲1 : IICS0 = 0010×010B
▲2 : IICS0 = 0010×000B
▲3 : IICS0 = 0010×000B
△4 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1

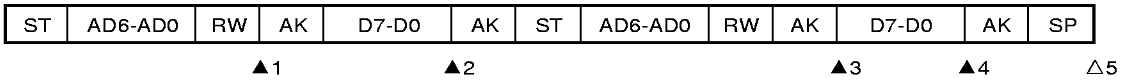


▲1 : IICS0 = 0010×010B
▲2 : IICS0 = 0010×110B
▲3 : IICS0 = 0010×100B
▲4 : IICS0 = 0010××00B
△5 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(b) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

(i) When WTIM0 = 0 (after restart, matches with SVA0n)



▲1 : IICS0 = 0010×010B

▲2 : IICS0 = 0010×000B

▲3 : IICS0 = 0001×110B

▲4 : IICS0 = 0001×000B

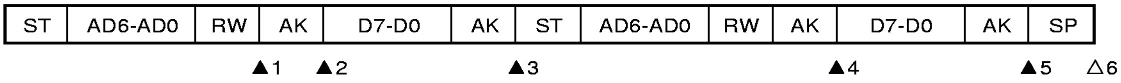
△5 : IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

× : don't care

(ii) When WTIM0 = 1 (after restart, matches with SVA0)



▲1 : IICS0 = 0010×010B

▲2 : IICS0 = 0010×110B

▲3 : IICS0 = 0010××00B

▲4 : IICS0 = 0001×110B

▲5 : IICS0 = 0001××00B

△6 : IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

× : don't care

(c) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

(i) When WTIM0 = 0 (after restart, extension code reception)

ST	AD6-AD0	RW	AK	D7-D0	AK	ST	AD6-AD0	RW	AK	D7-D0	AK	SP
			▲1		▲2				▲3		▲4	△5

▲1 : IICS0 = 0010×010B
▲2 : IICS0 = 0010×000B
▲3 : IICS0 = 0010×010B
▲4 : IICS0 = 0010×000B
△5 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1 (after restart, extension code reception)

ST	AD6-AD0	RW	AK	D7-D0	AK	ST	AD6-AD0	RW	AK	D7-D0	AK	SP
			▲1	▲2		▲3			▲4	▲5		▲6 △7

▲1 : IICS0 = 0010×010B
▲2 : IICS0 = 0010×110B
▲3 : IICS0 = 0010××00B
▲4 : IICS0 = 0010×010B
▲5 : IICS0 = 0010×110B
▲6 : IICS0 = 0010××00B
△7 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(d) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

(i) When WTIM0 = 0 (after restart, does not match with address (= not extension code))

ST	AD6-AD0	RW	AK	D7-D0	AK	ST	AD6-AD0	RW	AK	D7-D0	AK	SP
			▲1		▲2					▲3		△4

▲1 : IICS0 = 0010×010B
▲2 : IICS0 = 0010×000B
▲3 : IICS0 = 00000×10B
△4 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1 (after restart, does not match with address (= not extension code))

ST	AD6-AD0	RW	AK	D7-D0	AK	ST	AD6-AD0	RW	AK	D7-D0	AK	SP
			▲1	▲2		▲3				▲4		△5

▲1 : IICS0 = 0010×010B
▲2 : IICS0 = 0010×110B
▲3 : IICS0 = 0010××00B
▲4 : IICS0 = 00000×10B
△5 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(4) Operation without communication

(a) Start ~ Code ~ Data ~ Data ~ Stop

ST	AD6-AD0	RW	AK	D7-D0	AK	D7-D0	AK	SP
								△1

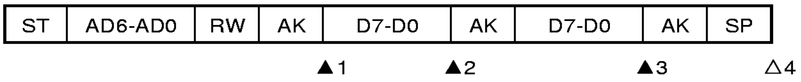
△1 : IICS0 = 00000001B

Remark △ : Generated only when SPIE0 = 1

(5) Arbitration loss operation (operation as slave after arbitration loss)

(a) When arbitration loss occurs during transmission of slave address data

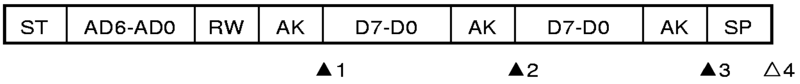
(i) When WTIM0 = 0



▲1 : IICS0 = 0101×110B (**Example** When ALD0 is read during interrupt servicing)
▲2 : IICS0 = 0001×000B
▲3 : IICS0 = 0001×000B
△4 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1



▲1 : IICS0 = 0101×110B (**Example** When ALD0 is read during interrupt servicing)
▲2 : IICS0 = 0001×100B
▲3 : IICS0 = 0001××00B
△4 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don't care

(b) When arbitration loss occurs during transmission of extension code

(i) When WTIM0 = 0

ST	AD6-AD0	RW	AK	D7-D0	AK	D7-D0	AK	SP	
			▲1		▲2		▲3		△4

▲1 : IICS0 = 0110×010B (**Example** When ALD0 is read during interrupt servicing)
▲2 : IICS0 = 0010×000B
▲3 : IICS0 = 0010×000B
△4 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1

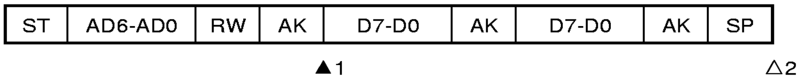
ST	AD6-AD0	RW	AK	D7-D0	AK	D7-D0	AK	SP	
			▲1	▲2		▲3		▲4	△5

▲1 : IICS0 = 0110×010B (**Example** When ALD0 is read during interrupt servicing)
▲2 : IICS0 = 0010×110B
▲3 : IICS0 = 0010×100B
▲4 : IICS0 = 0010××00B
△5 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(6) Operation when arbitration loss occurs (no communication after arbitration loss)

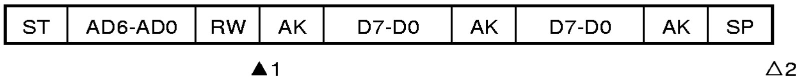
(a) When arbitration loss occurs during transmission of slave address data



▲1 : IICS0 = 01000110B (**Example** When ALD0 is read during interrupt servicing)
△2 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1

(b) When arbitration loss occurs during transmission of extension data

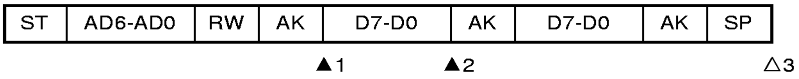


▲1 : IICS0 = 0110×010B (**Example** When ALD0 is read during interrupt servicing)
LREL0 is set to “1” by software
△2 : IICS0 = 00000001B

Remark ▲ : Always generated
△ : Generated only when SPIE0 = 1
× : don’t care

(c) When arbitration loss occurs during transmission of data

(i) When WTIM0 = 0



▲1 : IICS0 = 10001110B

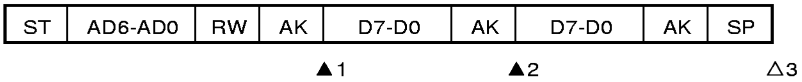
▲2 : IICS0 = 01000000B (**Example** When ALD0 is read during interrupt servicing)

△3 : IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

(ii) When WTIM0 = 1



▲1 : IICS0 = 10001110B

▲2 : IICS0 = 01000100B (**Example** When ALD0 is read during interrupt servicing)

△3 : IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

(d) When loss occurs due to restart condition during data transfer

(i) Not extension code (Example: mismatches with SVA0, WTIM0 = 1)

ST	AD6-AD0	RW	AK	D7-Dn	ST	AD6-AD0	RW	AK	D7-D0	AK	SP
			▲1					▲2			△3

▲1 : IICS0 = 1000×110B

▲2 : IICS0 = 01000110B (**Example** When ALD0 is read during interrupt servicing)

△3 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care
n = 6-0

(ii) Extension code

ST	AD6-AD0	RW	AK	D7-Dn	ST	AD6-AD0	RW	AK	D7-D0	AK	SP
			▲1				▲2				△3

▲1 : IICS0 = 1000×110B

▲2 : IICS0 = 0110×010B (**Example** When ALD0 is read during interrupt servicing)

Sets LREL0 = 1 by software

△3 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care
n = 6-0

(e) When loss occurs due to stop condition during data transfer

ST	AD6-AD0	RW	AK	D7-Dn	SP	
			▲1			△2

▲1 : IICS0 = 1000×110B
△2 : IICS0 = 01000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care
n = 6-0

(f) When arbitration loss occurs due to low-level data when attempting to generate a restart condition

(i) When WTIM0 = 0

STT0 = 1
↓

ST	AD6-AD0	RW	AK	D7-D0	AK	D7-D0	AK	D7-D0	AK	SP	
			▲1	▲2	▲3	▲4					△5

▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000×000B (Sets WTIM0)
▲3 : IICS0 = 1000××00B (Clears WTIM0, sets STT0 = 1)
▲4 : IICS0 = 01000000B (**Example** When ALD0 is read during interrupt servicing)
△5 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1

STT0 = 1
↓

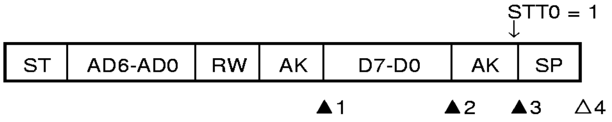
ST	AD6-AD0	RW	AK	D7-D0	AK	D7-D0	AK	D7-D0	AK	SP	
			▲1		▲2		▲3				△4

▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000×100B (Sets STT0)
▲3 : IICS0 = 01000100B (**Example** When ALD0 is read during interrupt servicing)
△4 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(g) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

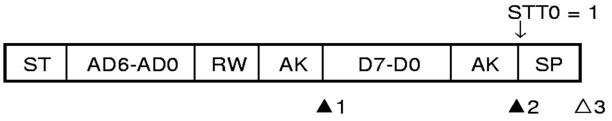
(i) When WTIM0 = 0



▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000×000B (Sets WTIM0)
▲3 : IICS0 = 1000××00B (Sets STT0)
△4 : IICS0 = 01000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(ii) When WTIM0 = 1

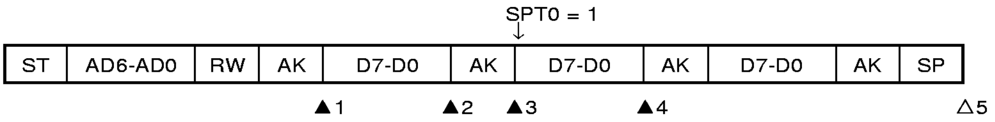


▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000××00B (Sets STT0)
△3 : IICS0 = 01000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
× : don't care

(h) When arbitration loss occurs due to low-level data when attempting to generate a stop condition

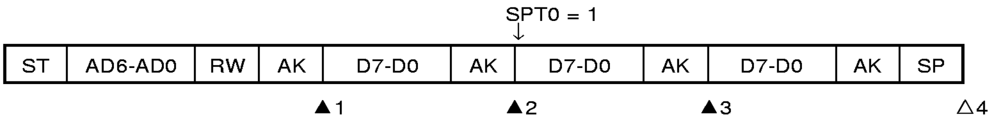
(i) When WTIM0 = 0



▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000×000B (Sets WTIM0)
▲3 : IICS0 = 1000××00B (Clears WTIM0, sets SPT0)
▲4 : IICS0 = 01000000B (**Example** When ALD0 is read during interrupt servicing)
△5 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

(ii) When WTIM0 = 1



▲1 : IICS0 = 1000×110B
▲2 : IICS0 = 1000××00B (Sets SPT0)
▲3 : IICS0 = 01000000B (**Example** When ALD0 is read during interrupt servicing)
△4 : IICS0 = 00000001B

Remark ▲: Always generated
△: Generated only when SPIE0 = 1
×: don't care

12.5.8 Interrupt request (INTIIC0) generation timing and wait control

The setting of bit 3 (WTIM0) in the IIC control register 0 (IICC0) determines the timing by which INTIIC0 is generated and the corresponding wait control, as shown in Table 12-2.

Table 12-2. INTIIC0 Timing and Wait Control

WTIM	During Slave Device Operation			During Master Device Operation		
	Address	Data reception	Data transmission	Address	Data reception	Data transmission
0	9[Notes 1, 2]	8[Note 2]	8[Note 2]	9	8	8
1	9[Notes 1, 2]	9[Note 2]	9[Note 2]	9	9	9

Notes 1. The slave device's INTIIC0 signal and wait period occurs at the falling edge of the ninth clock only when there is a match with the address set to the slave address register 0 (SVA0).
At this point, $\overline{\text{ACK}}$ is output regardless of the value set to IICC0's bit 2 (ACKE0). For a slave device that has received an extension code, INTIIC0 occurs at the falling edge of the eighth clock.

2. If the received address does not match the contents of the slave address register 0 (SVA0), neither INTIIC0 nor a wait occurs.

Remark The numbers in the table indicate the number of the serial clock's clock signals. Interrupt requests and wait control are both synchronized with the falling edge of these clock signals.

(1) During address transmission/reception

- Slave device operation : Interrupt and wait timing are determined regardless of the WTIM0 bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIM0 bit.

(2) During data reception

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIM0 bit.

(3) During data transmission

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIM0 bit.

(4) Wait cancellation method

The four wait cancellation methods are as follows.

- By setting bit 5 (WREL0) of IIC control register 0 (IICC0) to "1"
- By writing to the IIC shift register 0 (IIC0)
- By setting a start condition (setting bit 1 (STT0) of IIC control register 0 (IICC0) to "1")
- By setting a stop condition (setting IICC0's bit 0 (SPT0) to "1")

When 8-clock wait has been selected (WTIM0 = 0), the output level of $\overline{\text{ACK}}$ must be determined prior to wait cancellation.

(5) Stop condition detection

INTIIC0 is generated when a stop condition is detected.

12.5.9 Address match detection method

When in I^2C bus mode, the master device can select a particular slave device by transmitting the corresponding slave address.

Address match can be detected automatically by hardware. An interrupt frequency (INTIIC0) occurs when a local address has been set to the slave address register 0 (SVA0) and when the address set to SVA0 matches the slave address sent by the master device, or when an extension code has been received.

12.5.10 Error detection

During I^2C bus mode, the status of the serial data bus (SDA0) during data transmission is captured by the IIC shift register 0 (IIC0) of the transmitting device, so the IIC0 data prior to transmission can be compared with the transmitted IIC0 data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match.

12.5.11 Extension code

(1) When the high-order 4 bits of the receive address are either "0000" or "1111", the extension code flag (EXC0) is set for extension code reception and an interrupt request (INTIIC0) is issued at the falling edge of the eighth clock. The local address stored in the slave address register 0 (SVA0) is not affected.

(2) If "111110××" is set to SVA0 by a 10-bit address transfer and "111110××" is transferred from the master device, the results are as follows. Note that INTIIC0 occurs at the falling edge of the eighth clock.

- High-order four bits of data match: EXC0 = 1 **Note**
- Seven bits of data match: COI0 = 1 **Note**

Note EXC0 : Bit 5 of IIC status register 0 (IICS0)
COI0 : Bit 4 of IIC status register 0 (IICS0)

(3) Since the processing after the interrupt request occurs differs according to the data that follows the extension code, such processing is performed by software.
For example, after the extension code is received, if you do not wish to operate the target device as a slave device, you can set bit 6 (LREL0) of the IIC control register 0 (IIC0) to "1" to set the standby mode for the next communication operation.

Table 12-3. Extension Code Bit Definitions

Slave Address	R/W Bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	×	CBUS address
0000 010	×	Address that is reserved for different bus format
1111 0××	×	10-bit slave address specification

12.5.12 Arbitration

When several master devices simultaneously output a start condition (when STT0 is set to 1 before STD0 is set to 1 Note), communication among the master devices is performed as the number of clocks are adjusted until the data differs. This kind of operation is called arbitration.

When one of the master devices loses in arbitration, an arbitration loss flag (ALD0) in the IIC status register 0 (IICS0) is set via the timing by which the arbitration loss occurred, and the SCL0 and SDA0 lines are both set for high impedance, which releases the bus.

The arbitration loss is detected based on the timing of the next interrupt request (the eighth or ninth clock, when a stop condition is detected, etc.) and the ALD0 = 1 setting that has been made by software.

For details of interrupt request timing, refer to **12.5.7 I^{2}C interrupt requests (INTIIC0)**.

Note STD0 : Bit 1 of IIC status register (IICS0)
STT0 : Bit 1 of IIC control register (IICC0)

Figure 12-14. Arbitration Timing Example

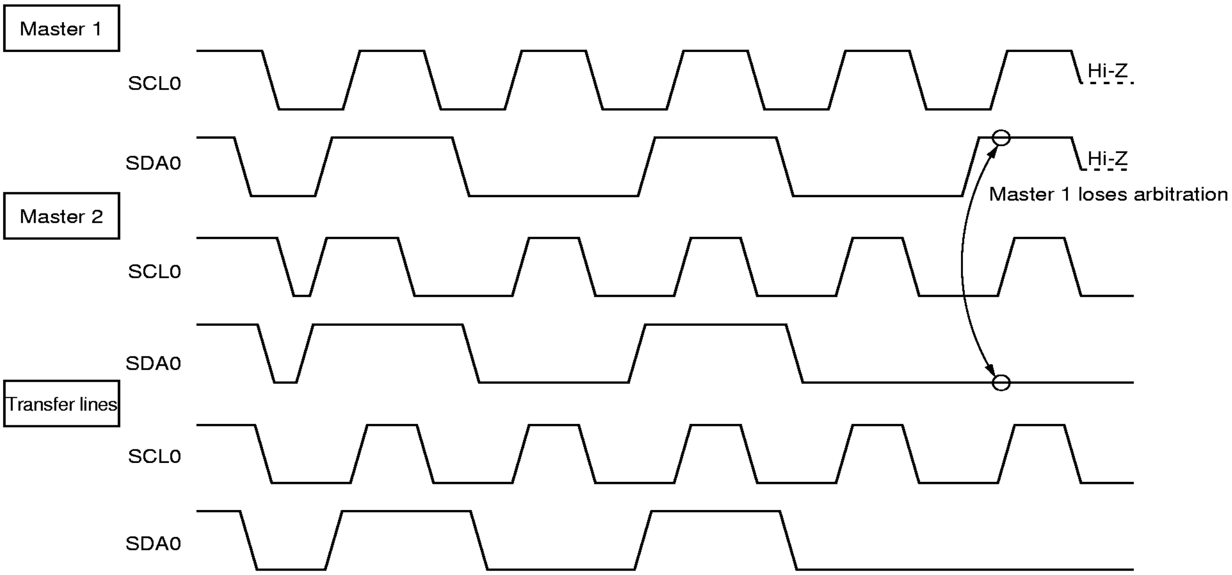


Table 12-4. Status during Arbitration and Interrupt Request Generation Timing

Status during Arbitration	Interrupt Request Generation Timing
During address transmission	At falling edge of eighth or ninth clock following byte transfer Note 1
Read/write data after address transmission	
During extension code transmission	
Read/write data after extension code transmission	
During data transmission	
During $\overline{\text{ACK}}$ signal transfer period after data transmission	
When restart condition is detected during data transfer	
When stop condition is detected during data transfer	When stop condition is output (when SPIE0 = 1) Note 2
When data is at low level while attempting to output a restart condition	At falling edge of eighth or ninth clock following byte transfer Note 1
When stop condition is detected while attempting to output a restart condition	When stop condition is output (when SPIE0 = 1) Note 2
When data is at low level while attempting to output a stop condition	At falling edge of eighth or ninth clock following byte transfer Note 1
When SCL0 is at low level while attempting to output a restart condition	

Notes 1. When WTIM0 (bit 3 of the IIC control register 0 (IICC0)) = 1, an interrupt request occurs at the falling edge of the ninth clock. When WTIM0 = 0 and the extension code's slave address is received, an interrupt request occurs at the falling edge of the eighth clock.

2. When there is a chance that arbitration will occur, set SPIE0 = 1 for master device operation.

Remark SPIE0 : Bit 5 of the IIC control register 0 (IICC0)

12.5.13 Wake up function

The I^2C bus slave function is a function that generates an interrupt request (INTIIC0) when a local address and extension code have been received.

This function makes processing more efficient by preventing unnecessary interrupt requests from occurring when addresses do not match.

When a start condition is detected, wake-up standby mode is set. This wake-up standby mode is in effect while addresses are transmitted due to the possibility that an arbitration loss may change the master device (which has output a start condition) to a slave device.

However, when a stop condition is detected, bit 5 (SPIE0) of the IIC control register 0 (IICC0) is set regardless of the wake up function, and this determines whether interrupt requests are enabled or prohibited.

12.5.14 Communication reservation

To start master device communications when not currently using a bus, a communication reservation can be made to enable transmission of a start condition when the bus is released. There are two modes under which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{\text{ACK}}$ is not returned and the bus was released when bit 6 (LREL0) of the IIC control register 0 (IICC0) was set to "1").

If bit 1 (STT0) of IICC0 is set while the bus is not used (after a stop condition is detected), a start condition is automatically generated and wait status is set.

When the bus release is detected (when a stop condition is detected), writing to the IIC shift register 0 (IIC0) causes the master's address transfer to start. At this point, IICC0's bit 4 (SPIE0) should be set.

When STT0 has been set, the operation mode (as start condition or as communication reservation) is determined according to the bus status.

- If the bus has been released .... a start condition is generated
- If the bus has not been released (standby mode) .... communication reservation

Check whether the communication reservation operates or not with MSTS0 (bit 7 of the IIC status register 0 (IICS0)) after SST0 is set and a wait time elapses.

Wait periods, which should be set via software, are listed in Table 12-5. These wait periods can be set via the settings for bits 3, 1, and 0 (SMC0, CL01, and CL00) in the IIC clock select register 0 (IICCL0).

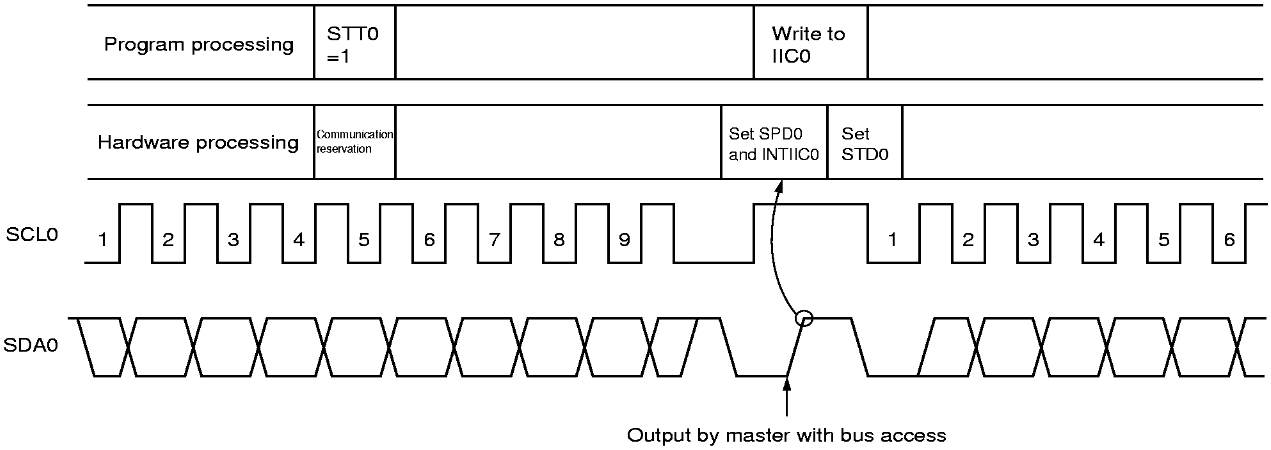
Table 12-5. Wait Periods

SMC0	CL01	CL00	Wait Time
0	0	0	26 clocks × 1/f_X
0	0	1	46 clocks × 1/f_X
0	1	0	92 clocks × 1/f_X
0	1	1	Setting prohibited
1	0	0	16 clocks × 1/f_X
1	0	1	
1	1	0	32 clocks × 1/f_X
1	1	1	Setting prohibited

Remark f_X: System oscillation frequency

Figure 12-5 shows communication reservation timing.

Figure 12-15. Communication Reservation Timing



Remark IIC0 : IIC shift register 0
STT0 : Bit 1 of IIC control register 0 (IICC0)
STD0 : Bit 1 of IIC status register 0 (IICS0)
SPD0 : Bit 0 of IIC status register 0 (IICS0)

Communication reservations are accepted via the following timing. After bit 1 (STD0) of the IIC status register 0 (IICS0) is set to “1”, a communication reservation can be made by setting bit 1 (STT0) of the IIC control register 0 (IICC0) to “1” before a stop condition is detected.

Figure 12-16. Timing for Accepting Communication Reservations

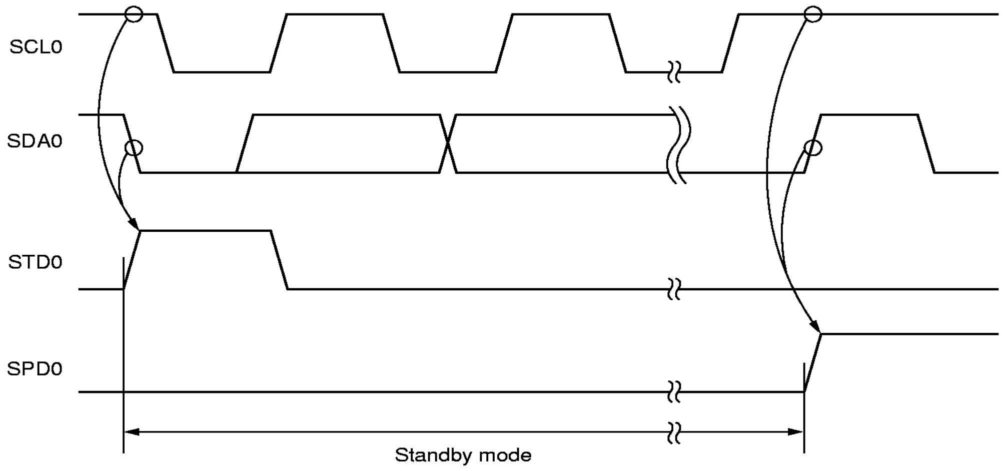
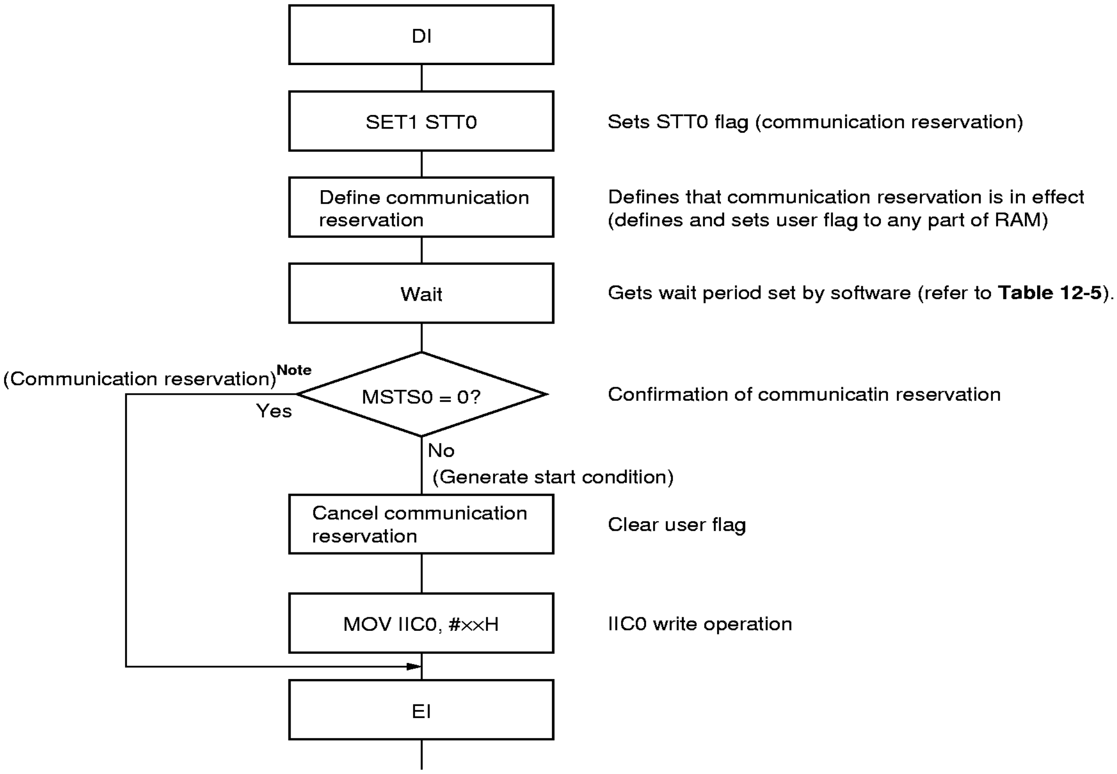


Figure 12-17 shows the communication reservation protocol.

Figure 12-17. Communication Reservation Protocol


DI
SET1 STT0
Sets STT0 flag (communication reservation)
Define communication reservation
Defines that communication reservation is in effect (defines and sets user flag to any part of RAM)
Wait
Gets wait period set by software (refer to **Table 12-5**).
(Communication reservation)Note
MSTS0 = 0?
Yes
Confirmation of communicatin reservation
No
(Generate start condition)
Cancel communication reservation
Clear user flag
MOV IIC0, #xxH
IIC0 write operation
EI

Note The communication reservation operation executes a write to the IIC shift register 0 (IIC0) when a stop condition interrupt request occurs.

Remark STT0 : Bit 1 of IIC control register 0 (IICC0)
MSTS0 : Bit 7 of IIC status register 0 (IICS0)
IIC0 : IIC shift register 0

12.5.15 Other cautions

After a reset, when changing from a mode in which no stop condition has been detected (the bus has not been released) to a master device communication mode, first generate a stop condition to release the bus, then perform master device communication.

When using multiple masters, it is not possible to perform master device communication when the bus has not been released (when a stop condition has not been detected).

Use the following sequence for generating a stop condition.

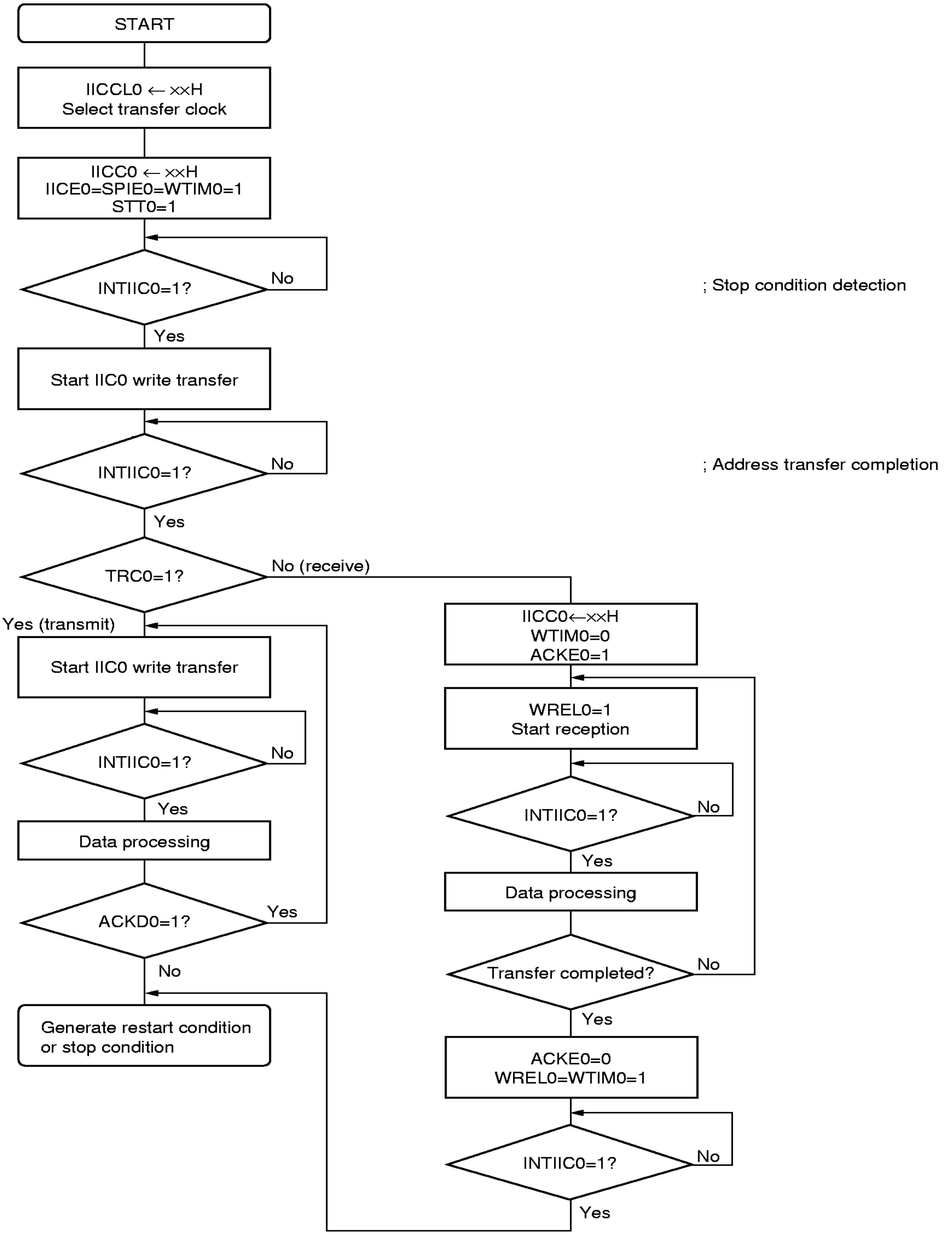
(a) Set IIC transfer clock select register (IICCL0).
(b) Set bit 7 (IICE0) of the IIC control register 0 (IICC0).
(c) Set bit 0 of IICC0.

12.5.16 Communication operations

(1) Master operations

The following is a flow chart of the master operations.

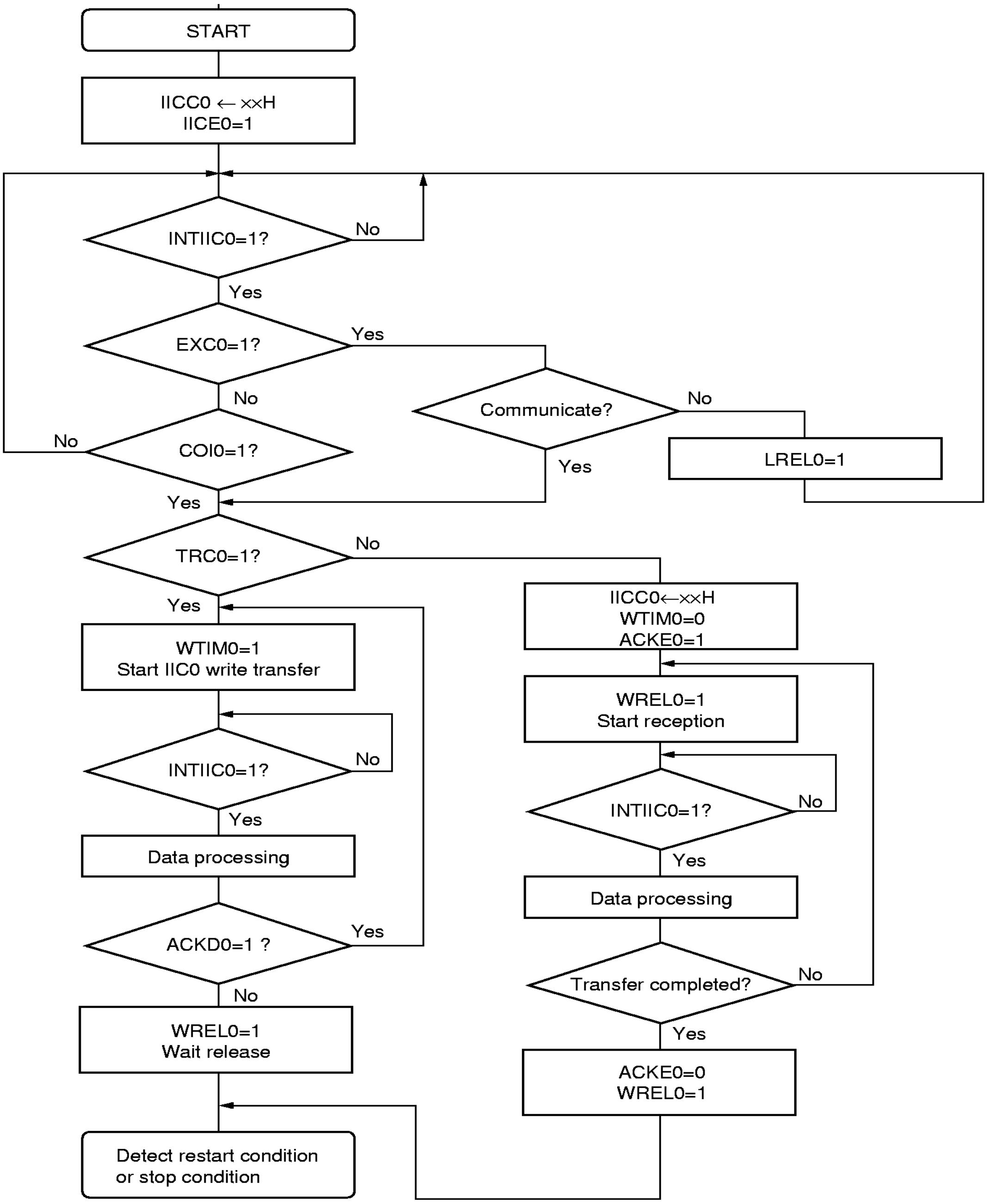
Figure 12-18. Flow Chart of Master Operation



(2) Slave operation

An example of slave operation is shown below.

Figure 12-19. Slave Operation Flow Chart



12.6 Timing Charts

When using the I^{2}C bus mode, the master device outputs an address via the serial bus to select one of several slave devices as its communication partner.

After outputting the slave address, the master device transmits the TRC0 bit (bit 3 of the IIC status register 0 (IICS0)) that specifies the data transfer direction and then starts serial communication with the slave device.

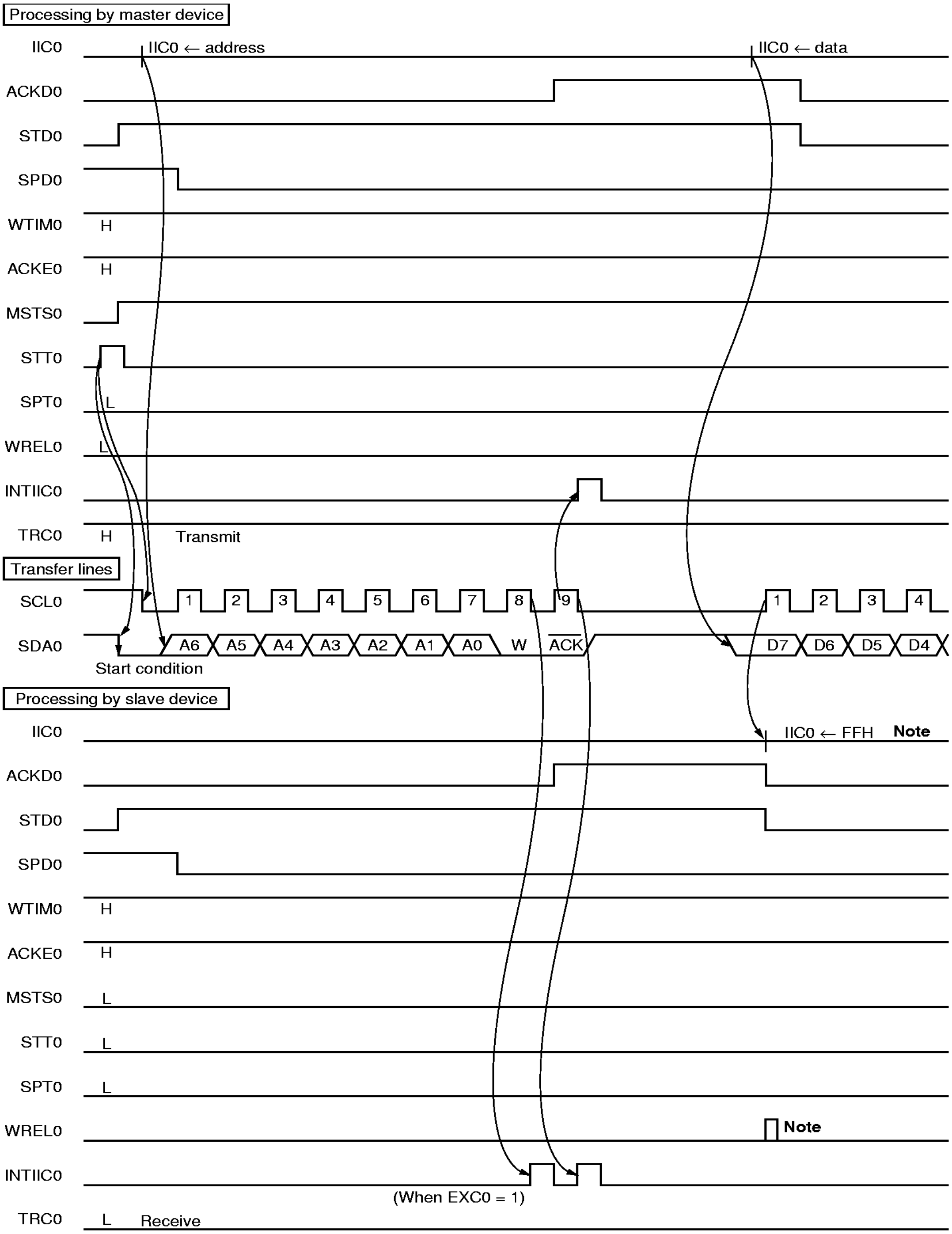
Figures 12-20 and 12-21 show timing charts of the data communication.

The IIC bus shift register 0 (IIC0)'s shift operation is synchronized with the falling edge of the serial clock (SCL0). The transmit data is transferred to the SO0 latch and is output (MSB first) via the SDA0 pin.

Data input via the SDA0 pin is captured into IIC0 at the rising edge of SCL0.

Figure 12-20. Example of Master to Slave Communication
(when 9-clock Wait Is Selected for Both Master and Slave) (1/3)

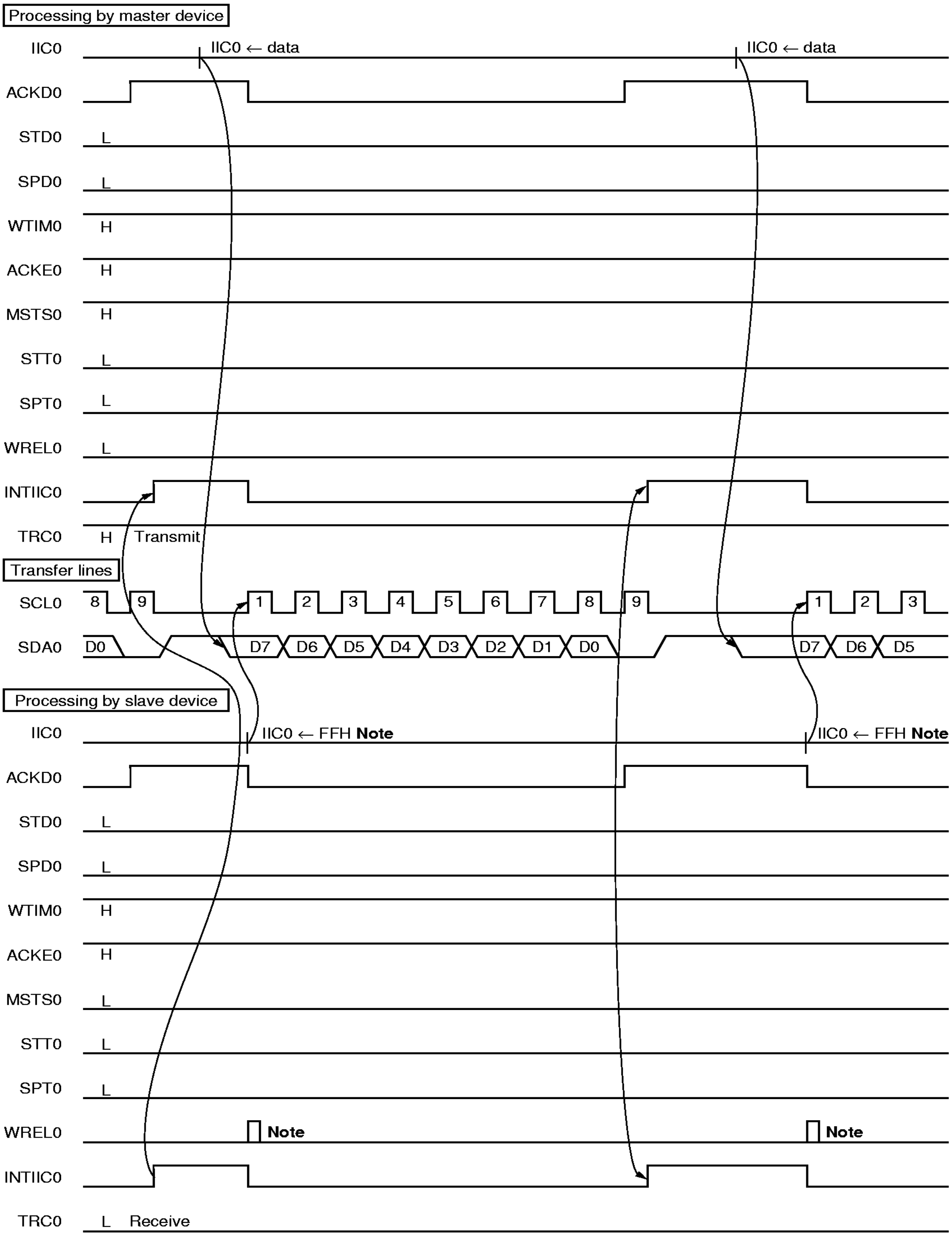
(1) Start condition ~ address



Note To cancel slave wait, write "FFH" to IIC0 or set WREL0.

Figure 12-20. Example of Master to Slave Communication (when 9-clock Wait Is Selected for Both Master and Slave) (2/3)

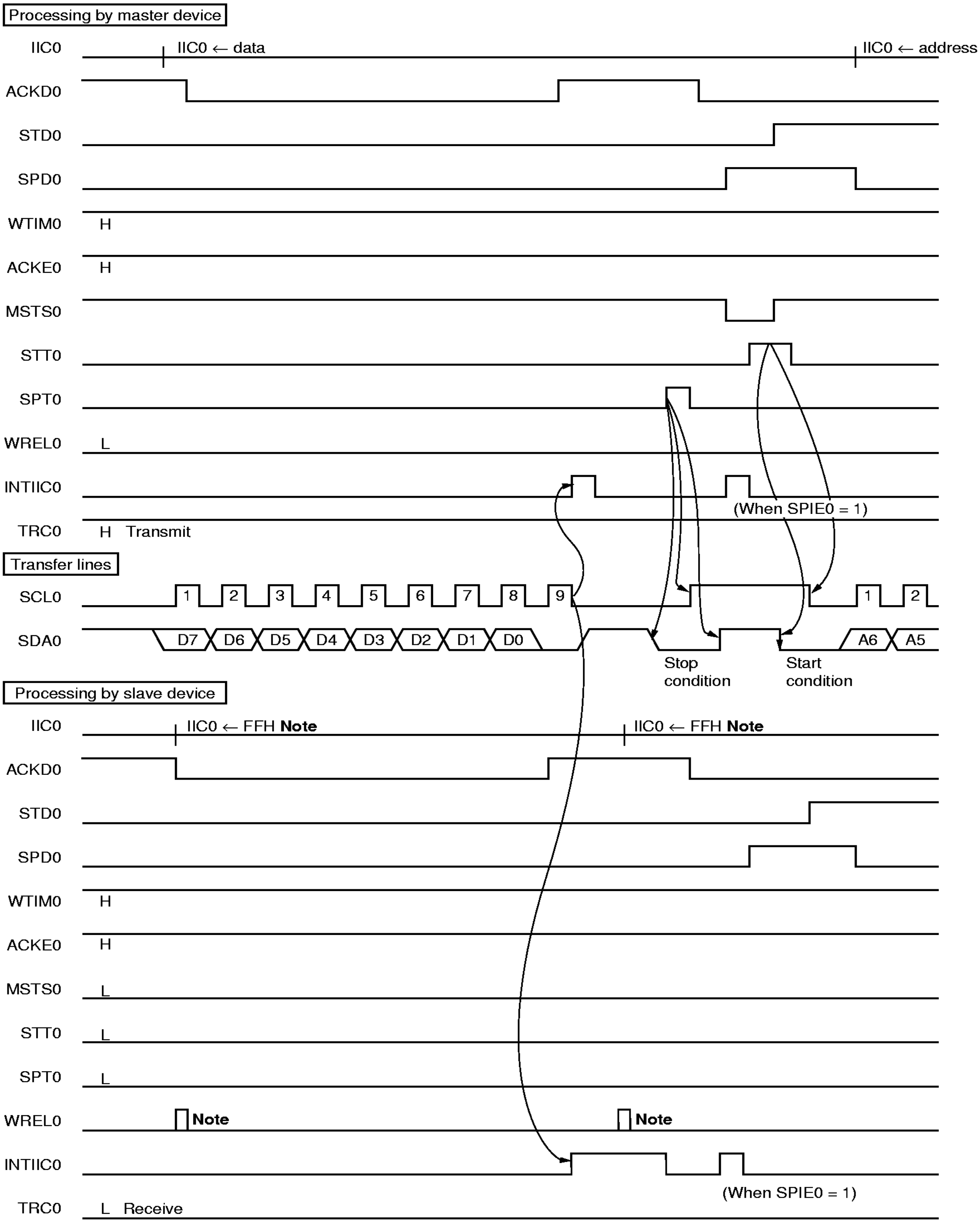
(2) Data



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

Figure 12-20. Example of Master to Slave Communication (when 9-clock Wait Is Selected for Both Master and Slave) (3/3)

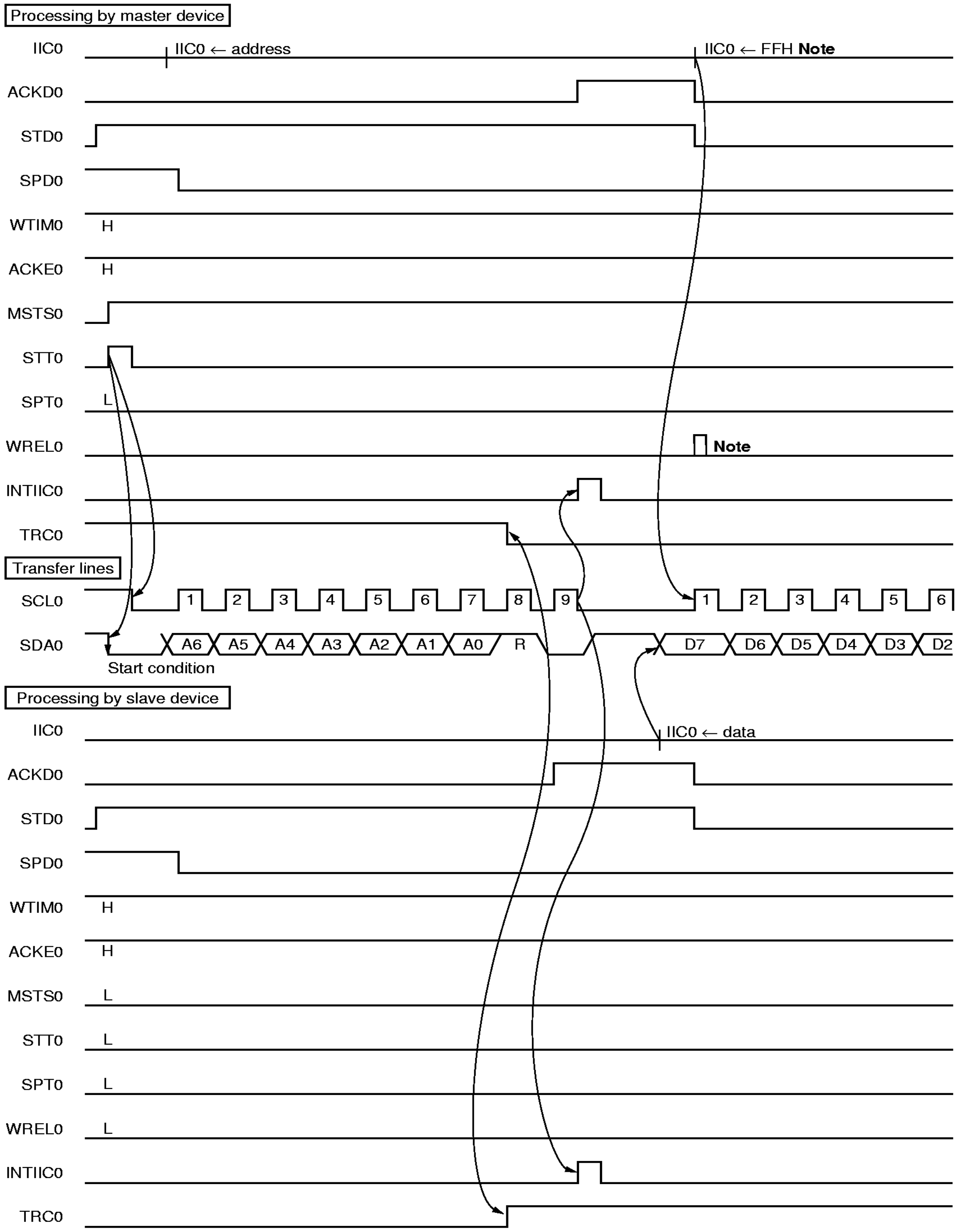
(3) Stop condition



Note To cancel slave wait, write "FFH" to IIC0 or set WREL0.

Figure 12-21. Example of Slave to Master Communication (when 9-clock Wait Is Selected for Both Master and Slave) (1/3)

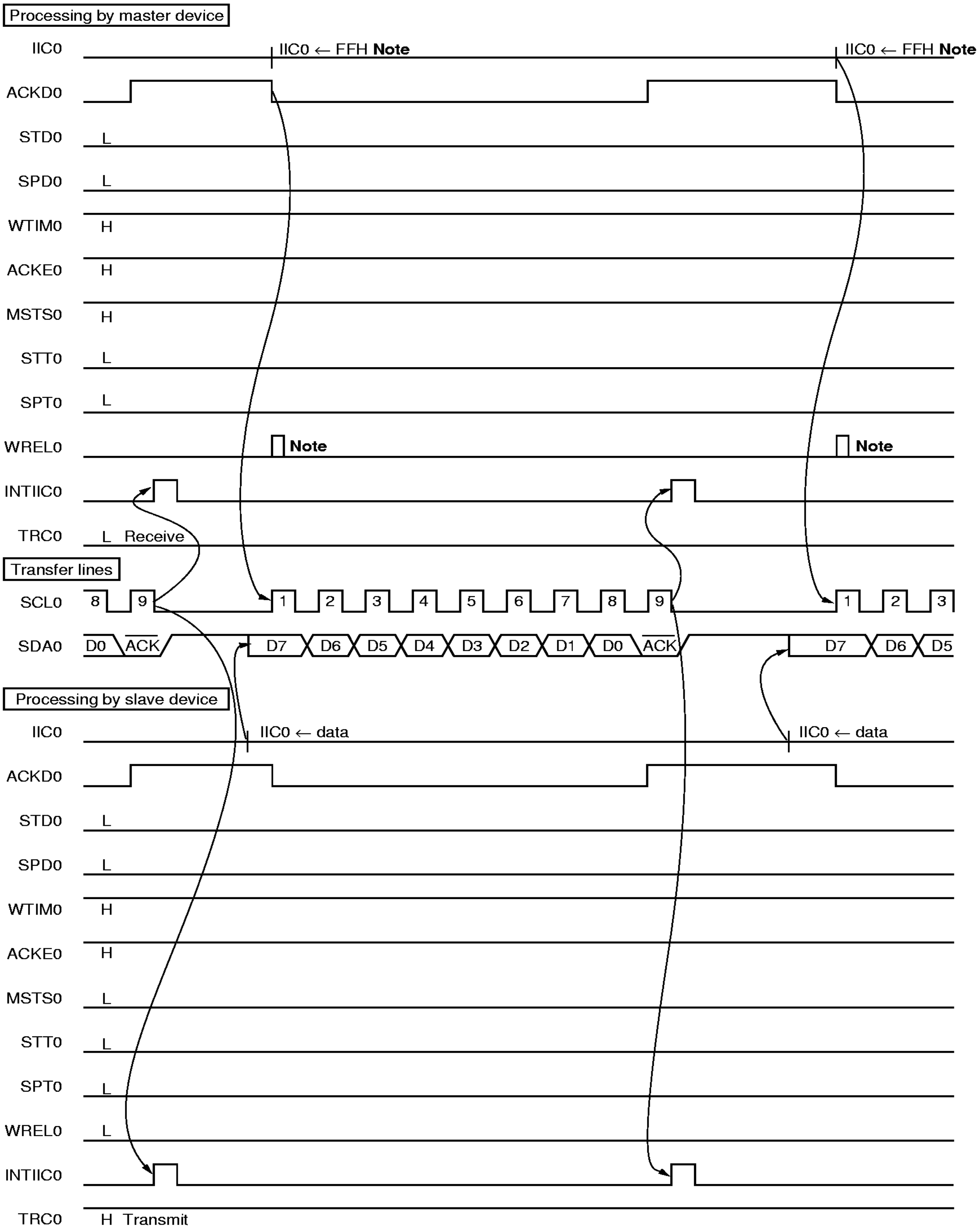
(1) Start condition ~ address



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

Figure 12-21. Example of Slave to Master Communication (when 9-clock Wait Is Selected for Both Master and Slave) (2/3)

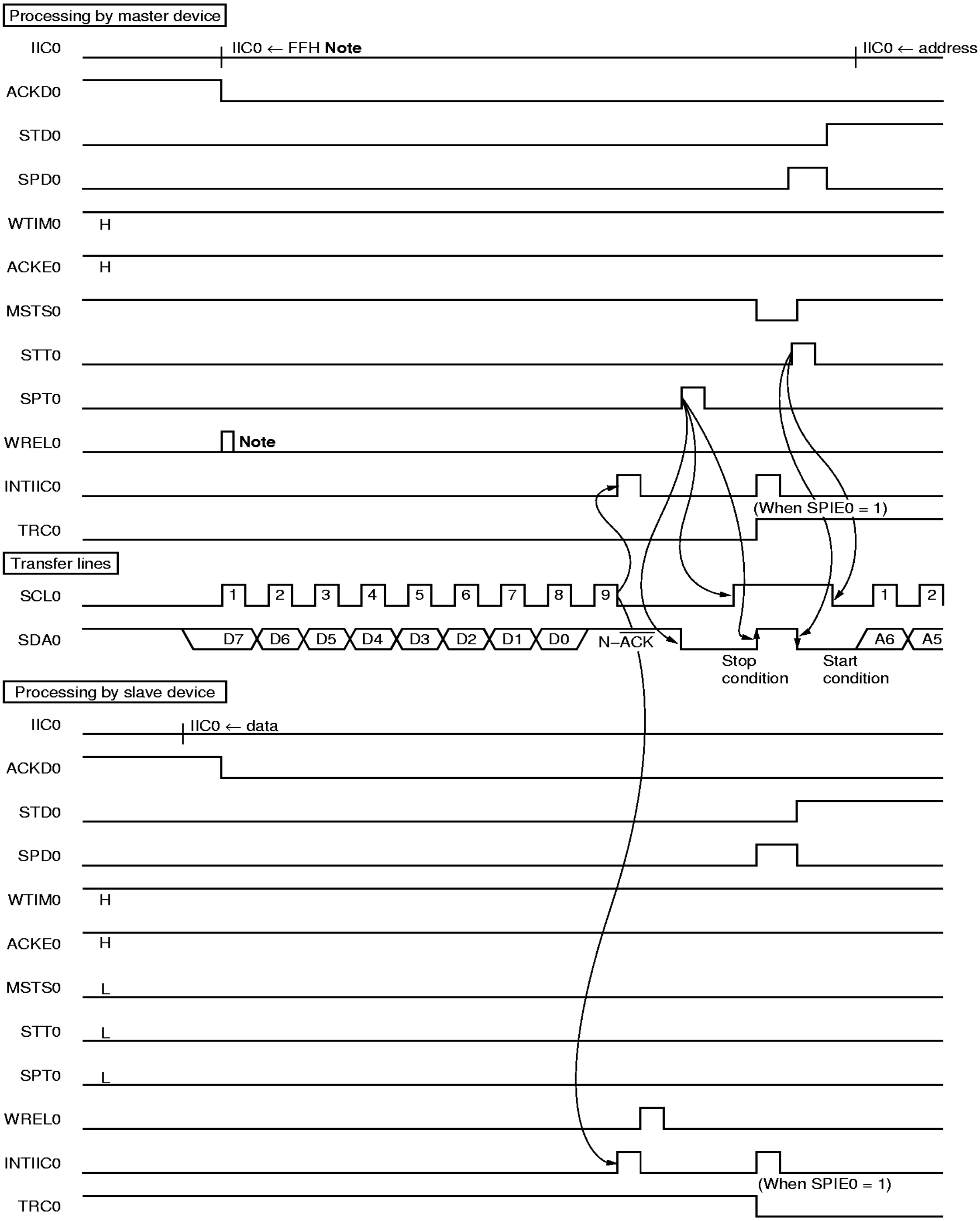
(2) Data



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

Figure 12-21. Example of Slave to Master Communication (when 9-clock Wait Is Selected for Both Master and Slave) (3/3)

(3) Stop condition



Note To cancel slave wait, write “FFH” to IIC0 or set WREL0.

[MEMO]

CHAPTER 13 SERIAL INTERFACE (SIO3)

13.1 Function of Serial Interface (SIO3)

The 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 **13.4.1**.

(2) 3-wire serial I/O mode (MSB first)

In this mode, 8-bit data is transferred by using three lines: serial clock ($\overline{\text{SCK3}}$), serial output (SO3), and serial input (SI3).

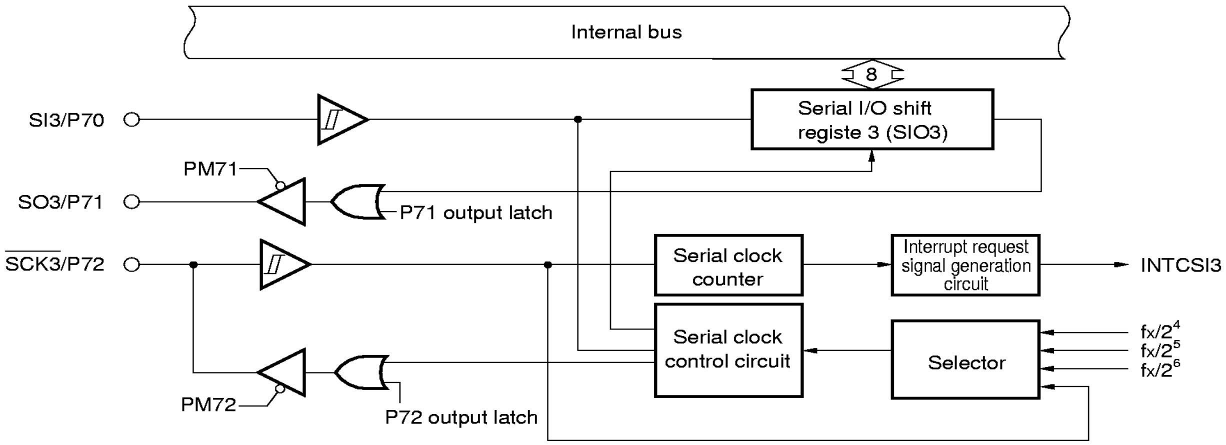
Because transmission and reception can be executed simultaneously in this mode, the processing time of data transfer can be shortened.

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 **13.4.2**.

Figure 13-1 shows the block diagram of the serial interface (SIO3).

Figure 13-1. Block Diagram of Serial Interface (SIO3)


Internal bus
8
Serial I/O shift registe 3 (SIO3)
SI3/P70
PM71
SO3/P71
P71 output latch
$\overline{\text{SCK3}}$/P72
Serial clock counter
Interrupt request signal generation circuit
INTCSI3
Serial clock control circuit
Selector
fx/2^4
fx/2^5
fx/2^6
PM72
P72 output latch

13.2 Configuration of Serial Interface (SIO3)

The serial interface (SIO3) consists of the following hardware:

Table 13-1. Configuration of Serial Interface (SIO3)

Item	Configuration
Register	Serial I/O shift register 3 (SIO3)
Control register	Serial operating mode register 3 (CSIM3)

(1) Serial I/O shift register 3 (SIO3)

This 8-bit register converts parallel data into serial data and transmits or receives the serial data (shift operation) in synchronization with a serial clock.

SIO3 is set by using an 8-bit memory manipulation instruction.

Serial operation is started by writing or reading data to or from SIO3 when bit 7 (CSIE3) of the 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 at reset.

Caution Do not execute access other than that for transfer starting trigger to SIO3 during transfer operation (the read operation disabled when MODE0 = 0, and the write operation is disabled when MODE0 = 1).

13.3 Register Controlling Serial Interface (SIO3)

The following register controls the serial interface (SIO3):

- **Serial operating mode register 3 (CSIM3)**

(1) Serial operating mode register 3 (CSIM3)

This register selects the serial clock of SIO3 and an operating mode, and enables or disables the operation. CSIM3 is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H at reset.

Figure 13-2. Format of Serial Operating Mode Register 3 (CSIM3)

Symbol	<7>	6	5	4	3	2	1	0	Address	At reset	R/W
CSIM3	CSIE3	0	0	0	0	MODE	SCL31	SCL30	FF6FH	00H	R/W

CSIE3	Enables/Disables SIO3 Operation		
	Shift register operation	Serial counter	Port
0	Disables operation	Cleared	Port function[Note 1]
1	Enables operation	Enables counter operation	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	Selects Clock
0	0	External clock input to $\overline{\text{SCK3}}$
0	1	fx/2^4 (281 kHz)
1	0	fx/2^5 (141 kHz)
1	1	fx/2^6 (70.3 kHz)

Notes
1. The SI3, SO3, and $\overline{\text{SCK3}}$ 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).

Caution Set the port mode register (PM××) as follows in the 3-wire serial I/O mode. Reset the output latch to 0.

- **When serial clock is output (master transmission or master reception)**
  Set P72 ($\overline{\text{SCK3}}$) in the output mode (PM72 = 0).
- **When serial clock is input (slave transmission or slave reception)**
  Set P72 in the input mode (PM72 = 1).
- **In transmit or transmit/receive mode**
  Set P71 (SO3) in the output mode (PM71 = 0).
- **In receive mode**
  Set P70 (SI3) in the input mode (PM70 = 1).

Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 4.5 MHz

13.4 Operation of Serial Interface (SIO3)

This section explains the two modes of the serial interface (SIO3).

13.4.1 Operation stop mode

In this mode, serial transfer is not performed.

The P70/SI3, P71/SO3, and P72/$\overline{\text{SCK3}}$ pins can be used as ordinary I/O port pins.

(1) Register setting

The operation stop mode is set by using the serial operating mode register 3 (CSIM3).

CSIM3 is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H at reset.

Symbol	<7>	6	5	4	3	2	1	0	Address	At reset	R/W
CSIM3	CSIE3	0	0	0	0	MODE	SCL31	SCL30	FF6FH	00H	R/W

CSIE3	Enables/Disables SIO3 Operation		
	Shift register operation	Serial counter	Port
0	Disables operation	Cleared	Port function[Note 1]
1	Enables operation	Enables count operation	Serial function + port function[Note 2]

Notes 1. The SI3, SO3, and $\overline{\text{SCK3}}$ 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 as a port pin in the receive mode.

13.4.2 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: serial clock ($\overline{\text{SCK3}}$), serial output (SO3), and serial input (SI3).

(1) Register setting

The 3-wire serial I/O mode is set by using the serial operating mode register 3 (CSIM3).

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

The value of this register is initialized to 00H at reset.

Symbol	<7>	6	5	4	3	2	1	0	Address	At reset	R/W
CSIM3	CSIE3	0	0	0	0	MODE	SCL31	SCL30	FF6FH	00H	R/W

CSIE3	Enables/Disables SIO3 Operation		
	Shift register operation	Serial counter	Port
0	Disables operation	Cleared	Port function[Note 1]
1	Enables operation	Enables counter operation	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	Selects Clock
0	0	External clock input to $\overline{\text{SCK3}}$
0	1	$fx/2^4$ (281 kHz)
1	0	$fx/2^5$ (141 kHz)
1	1	$fx/2^6$ (70.3 kHz)

Notes 1. The SI3, SO3, and $\overline{\text{SCK3}}$ 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).

Caution Set the port mode register (PM××) as follows in the 3-wire serial I/O mode. Reset the output latch to 0.

- **When serial clock is output (master transmission or master reception)**
  Set P72 ($\overline{\text{SCK3}}$) in the output mode (PM72 = 0).
- **When serial clock is input (slave transmission or slave reception)**
  Set P72 in the input mode (PM72 = 1).
- **In transmit or transmit/receive mode**
  Set P71 (SO3) in the output mode (PM71 = 0).
- **In receive mode**
  Set P70 (SI3) in the input mode (PM70 = 1).

Remarks 1. fx: System clock oscillation frequency
2. ( ): fx = 4.5 MHz

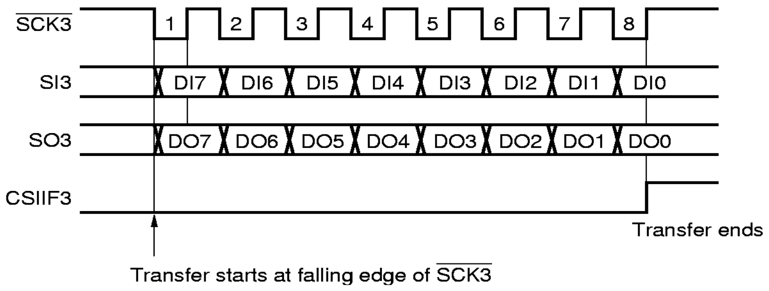
(2) Communication operation

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 the serial I/O shift register 3 (SIO3) is performed at the falling edge of the serial clock ($\overline{\text{SCK3}}$). The transmit data is retained to 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 an interrupt request flag (CSIIF3) is set.

Figure 13-3. Timing in 3-Wire Serial I/O Mode



(3) Starting transfer

Serial transfer is started by writing (or reading) the transfer data to the 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 $\overline{\text{SCK3}}$ 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 an interrupt request flag (CSIIF3) is set.

[MEMO]

14.1 Function of Serial Interface (UART0)

The serial interface (UART0) has the following two modes:

(1) Operation stop mode

In this mode, serial transfer is not performed.

For details, refer to **14.4.1**.

(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.

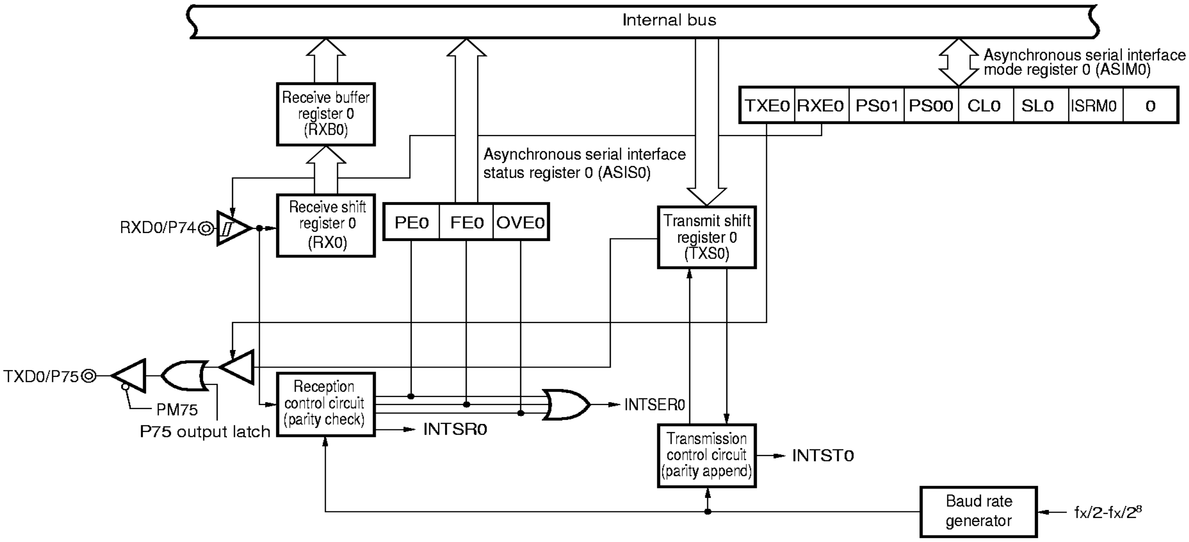
Because a UART-dedicated baud rate generator is provided, communication can be executed at a wide range of baud rates.

Moreover, MIDI standard baud rate (31.25 kbps) can be also generated by using the UART-dedicated baud rate generator.

For details, refer to **14.4.2**.

Figure 14-1 shows the block diagram of the serial interface (UART0).

Figure 14-1. Block Diagram of Serial Interface (UART0)


Internal bus
Asynchronous serial interface mode register 0 (ASIM0)
TXE0
RXE0
PS01
PS00
CL0
SL0
ISRM0
0
Receive buffer register 0 (RXB0)
Asynchronous serial interface status register 0 (ASIS0)
Receive shift register 0 (RX0)
PE0
FE0
OVE0
Transmit shift register 0 (TXS0)
RXD0/P74
TXD0/P75
PM75
P75 output latch
Reception control circuit (parity check)
INTSR0
INTSER0
Transmission control circuit (parity append)
INTST0
Baud rate generator
fx/2-fx/2⁸

14.2 Configuration of Serial Interface (UART0)

The serial interface (UART0) consists of the following hardware:

Table 14-1. Configuration of Serial Interface (UART0)

Item	Configuration
Register	Transmit shift register 0 (TXS0) Receive shift register 0 (RX0) Receive buffer register 0 (RXB0)
Control register	Asynchronous serial interface mode register 0 (ASIM0) Asynchronous serial interface status register 0 (ASIS0) Baud rate generator control register 0 (BRGC0)

(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 through 6 of the data written to TXS0 are transferred as transmit data. By writing data to TXS0, the transmission operation is started.
TXS0 is written by using an 8-bit memory manipulation instruction, but it cannot be read.
At reset, the value of this register is set to FFH.

Caution Do not write data to TXS0 during transmission operation.
Because TXS0 and the 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 the receive buffer register 0 (RXB0).
RX0 cannot be directly manipulated in 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 the receive shift register 0 (RX0).
If the data length is specified to be 7 bits, the receive data is transferred to bits 0 through of RXB0, and the MSB of RXB0 is always 0.
RXB0 can be read by using an 8-bit memory manipulation instruction, but it cannot be written.
The value of this register is set to FFH at reset.

Caution Because RXB0 and the 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 control circuit

This circuit controls the transmission operation by appending a start bit, parity bit, and stop bit to the data written to the transmit shift register 0 (TXS0) according to the contents of the asynchronous serial interface mode register 0 (ASIM0).

(5) Reception control circuit

This circuit controls the reception operation according to the contents of the 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 the asynchronous serial interface status register 0 (ASIS0).

14.3 Registers Controlling Serial Interface (UART0)

The following three registers control the serial interface (UART0):

- Asynchronous serial interface mode register 0 (ASIM0)
- Asynchronous serial interface status register 0 (ASIS0)
- Baud rate generator control register 0 (BRGC0)

(1) Asynchronous serial interface mode register 0 (ASIM0)

This 8-bit register controls the serial transfer operation of the serial interface (UART0).
ASIM0 is set by using a 1-bit or 8-bit memory manipulation instruction.
The value of this register is initialized to 00H at reset.
Figure 14-2 shows the format of ASIM0.

Figure 14-2. Format of Asynchronous Serial Interface Mode Register 0 (ASIM0)

Symbol	<7>	<6>	5	4	3	2	1	0	Address	At 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	Stops operation	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	Specifies Parity Bit
0	0	No parity
0	1	Always appends 0 parity during transmission. Does not check parity during reception (parity error does not occur).
1	0	Odd parity
1	1	Even parity

CL0	Specifies Character Length
0	7 bits
1	8 bits

SL0	Specifies Stop Bit Length of Transmit Data
0	1 bit
1	2 bits

ISRM0	Controls Reception Completion Interrupt in Case of Error
0	Generates reception completion interrupt request in case of error
1	Does not generate reception completion interrupt request in case of error

Note Be sure to reset bit 0 of ASIM0 to “0”.

Cautions 1. Before changing the operating mode, stop the serial transmission/reception operation.

2. Set the port mode registers (PM××) as follows in the UART mode.
Reset the output latch to 0.

- **During reception**
  Set P74 (RXD0) in the input mode (PM74 = 1).
- **During transmission**
  Set P75 (TXD0) in the output mode (PM75 = 0).
- **During transmission/reception**
  Set P74 in the input mode and P75 in 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 using an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H at reset.

Figure 14-3. Format of Asynchronous Serial Interface Register 0 (ASIS0)

Symbol	7	6	5	4	3	2	1	0	Address	At 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 coincide).

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).

Notes **1.** Even when the stop bit length is set to 2 bits by bit 2 (SL0) of the asynchronous serial interface mode register 0 (ASIM0), only 1 stop bit is detected during reception.

2. Be sure to read the receive buffer register 0 (RXB0) if an overrun error has occurred.
The overrun error persistently occurs each time data has been received, until RXB0 is read.

(3) **Baud rate generator control register 0 (BRGC0)**

This register sets the serial clock of the serial interface.

BRGC0 is set by using an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H at reset.

Figure 14-4 shows the format of BRGC0.

Figure 14-4. Format of Baud Rate Generator Control Register 0 (BRGC0)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00	FF5CH	00H	R/W

TPS02	TPS01	TSP00	Selects Source Cock of 5-bit Counter	n
0	0	0	$f_X/2$ (2.25 MHz)	1
0	0	1	$f_X/2^2$ (1.13 MHz)	2
0	1	0	$f_X/2^3$ (563 kHz)	3
0	1	1	$f_X/2^4$ (281 kHz)	4
1	0	0	$f_X/2^5$ (141 kHz)	5
1	0	1	$f_X/2^6$ (70.3 kHz)	6
1	1	0	$f_X/2^7$ (35.2 kHz)	7
1	1	1	$f_X/2^8$ (17.6 kHz)	8

MDL03	MDL02	MDL01	MDL00	Selects 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 through TPS02 ($1 \leq n \leq 8$)
k : Value set by MDL00 through MDL03 ($0 \leq k \leq 14$)
( ) : f_X = 4.5 MHz

14.4 Operation of Serial Interface (UART0)

This section explains the two modes of the serial interface (UART0).

14.4.1 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 by the 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 at reset.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	At 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	Stops operation	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.

14.4.2 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 be also generated by using the UART-dedicated baud rate generator.

(1) Register setting

The UART mode is set by the asynchronous serial interface mode register 0 (ASIM0), asynchronous serial interface status register 0 (ASIS0), and baud rate generator control register 0 (BRGC0).

(a) Asynchronous serial interface mode register 0 (ASIM0)

ASIM0 is set by using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is initialized to 00H at reset.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	At 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	Stops operation	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	Specifies Parity Bit
0	0	No parity
0	1	Always appends 0 parity during transmission. Does not check parity during reception (parity error does not occur).
1	0	Odd parity
1	1	Even parity

CL0	Specifies Character Length
0	7 bits
1	8 bits

SL0	Specifies Stop Bit Length of Transmit Data
0	1 bit
1	2 bits

ISRM0	Controls Reception Completion Interrupt in Case of Error
0	Generates reception completion interrupt request in case of error
1	Does not generate reception completion interrupt request in case of error

Note Be sure to reset bit 0 of ASIM0 to “0”.

Cautions 1. Before changing the operating mode, stop the serial transmission/reception operation.

2. Set the port mode registers (PM××) as follows in the UART mode.
Reset the output latch to 0.

- **During reception**
  Set P74 (RXD0) in the input mode (PM74 = 1).
- **During transmission**
  Set P75 (TXD0) in the output mode (PM75 = 0).
- **During transmission/reception**
  Set P74 in the input mode and P75 in the output mode.

(b) Asynchronous serial interface status register 0 (ASIS0)

ASIS0 is set by using an 8-bit memory manipulation instruction.

The value of this register is initialized to 00H at reset.

Symbol	7	6	5	4	3	2	1	0	Address	At 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 coincide).

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 (RXB0)).

Notes 1. Even when the stop bit length is set to 2 bits by bit 2 (SL0) of the asynchronous serial interface mode register 0 (ASIM0), only 1 stop bit is detected during reception.

2. Be sure to read the receive buffer register 0 (RXB0) if an overrun error has occurred.
The overrun error persistently occurs each time data has been received, until RXB0 is read.

(c) Baud rate generator control register 0 (BRGC0)

BRGC0 is set by using an 8-bit memory manipulation instruction.
The value of this register is initialized to 00H at reset.

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
BRGC0	0	TPS02	TPS01	TPS00	MDL03	MDL02	MDL01	MDL00	FF5CH	00H	R/W

TPS02	TPS01	TPS00	Selects Source Clock of 5-bit Counter	n
0	0	0	$f_X/2$ (2.25 MHz)	1
0	0	1	$f_X/2^2$ (1.13 MHz)	2
0	1	0	$f_X/2^3$ (563 kHz)	3
0	1	1	$f_X/2^4$ (281 kHz)	4
1	0	0	$f_X/2^5$ (141 kHz)	5
1	0	1	$f_X/2^6$ (70.3 kHz)	6
1	1	0	$f_X/2^7$ (35.2 kHz)	7
1	1	1	$f_X/2^8$ (17.6 kHz)	8

MDL03	MDL02	MDL01	MDL00	Selects 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 through TPS02 ($1 \leq n \leq 8$)
k : Value set by MDL00 through MDL03 ($0 \leq k \leq 14$)
( ) : f_X = 4.5 MHz

The transmit/receive clock for the generated baud rate is the system clock divided.

- Generation of transmit/receive clock for baud rate with system clock
  The transmit/receive clock is generated by dividing the system clock. The baud rate generated from the system clock can be calculated by the following expression:

$$[\text{Baud rate}] = \frac{fx}{2^{n+1}(k + 16)} \ [\text{bps}]$$

fx : System clock oscillation frequency
n : Value set by TPS00 through TPS02 ($1 \leq n \leq 8$)
For details, refer to **Table 14-2**.
k : Value set by MDL00 through MDL03 ($0 \leq k \leq 14$)

Table 14-2 shows the relation between the source clock of the 5-bit counter allocated to bits 4 through 6 (TPS00 through TPS02) of BRGC0 and the value of n.

Table 14-2. Relation between Source Clock of 5-Bit Counter and Value n

TPS02	TPS01	TPS00	Selects Source Clock of 5-bit Counter	n
0	0	0	fx/2	1
0	0	1	$fx/2^2$	2
0	1	0	$fx/2^3$	3
0	1	1	$fx/2^4$	4
1	0	0	$fx/2^5$	5
1	0	1	$fx/2^6$	6
1	1	0	$fx/2^7$	7
1	1	1	$fx/2^8$	8

Remark fx: System clock oscillation frequency

- Permissible error range of baud rate

  The permissible range of the baud rate depends on the number of bits in one frame and the division ratio of the counter [1/(16 + k)].

  Table 14-3 shows the relation between the system clock and baud rate, and Figure 14-5 shows an example of permissible error.

Table 14-3. Relation between System Clock and Baud Rate

Baud Rate	fx = 4.5 MHz	
[bps]	BRGC0	Error (%)
300	7DH	1.02
600	6DH	1.02
1200	5DH	1.02
2400	4DH	1.02
4800	3DH	1.02
9600	2DH	1.02
19200	1DH	1.02
31250	12H	0.0
38400	0DH	1.02

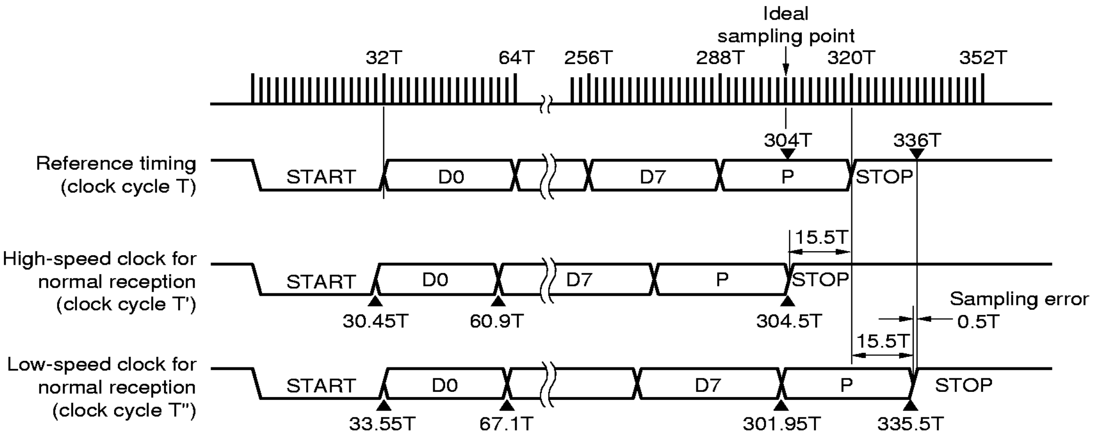
Remark Baud rate = fx/{2^{n+1} × (k + 16)}

fx : System clock oscillation frequency

n : Value set by TPS00 through TPS02 ($1 \leq n \leq 8$)

k : Value set by MDL00 through MDL03 ($0 \leq k \leq 14$)

Figure 14-5. Permissible Error of Baud Rate with Sampling Error Considered (k = 0)



Remark T: Source clock cycle of 5-bit counter

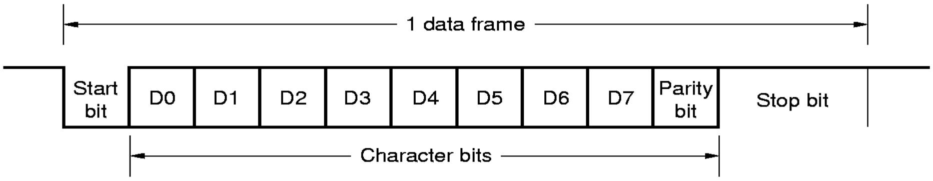
$$\text{Permissible baud rate error (where k = 0)} = \frac{\pm 15.5}{320} \times 100 = 4.8438\ (\%)$$

(2) Basic operation

(a) Data format

Figure 14-6 shows the format of transmit/receive data.

Figure 14-6. 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
- 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 the asynchronous serial interface mode register 0 (ASIM0).

If the character bit length is specified to be 7 bits, only the low-order 7 bits (bits 0 through 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 ASIM0 and 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 the 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 both at the transmitter and receiver sides. With even parity or odd parity, an error of 1 bit (or odd numbers of errors) 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

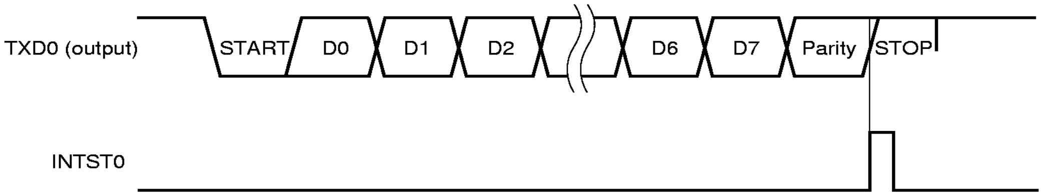
The transmission operation is started when transmit data is written to the transmit shift register 0 (TXS0). The start bit, parity bit, and stop bit are automatically appended.

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.

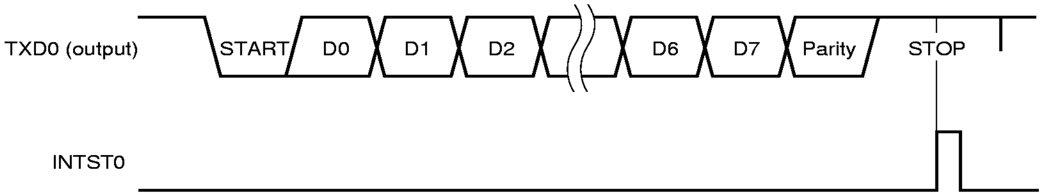
Figure 14-7 shows the timing of the transmission completion interrupt.

Figure 14-7. 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 the asynchronous serial interface mode register 0 (ASIM0) during transmission. If the contents of the ASIM0 register are changed during transmission, the transmission may not be performed any more (it can be returned to the normal status 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 the 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 the 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 the receive buffer register (RXB0), and a reception completion interrupt request (INTSR0) is generated.

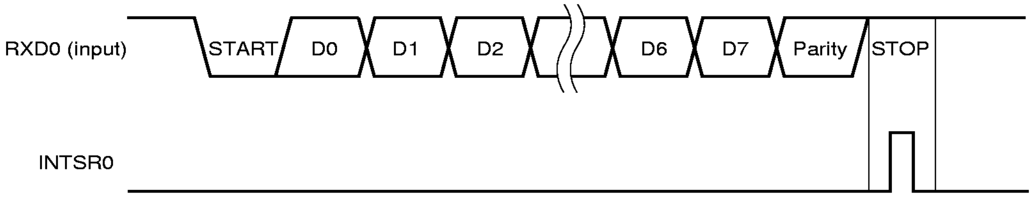
Even if an error occurs, the receive data is transferred if an error occurs in RXB0. If bit 1 (ISRM0) of ASIM0 is cleared to 0 when the error occurs, INTSR0 is generated (refer to **Figure 14-9**).

INTSR0 is not generated if the ISRM0 bit is set to 1.

If the RXE0 bit is reset to 0 during reception, the reception is immediately stopped. At this time, the contents of RXB0 and ASIS0 are not affected, nor are INTSR0 and INTSER0 generated.

Figure 14-8 shows the timing of generating the reception completion interrupt request of the asynchronous serial interface.

Figure 14-8. Timing of Generation of Reception Completion Interrupt of Asynchronous Serial Interface



Caution Be sure to read the 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 as a result.

(e) Reception error

Three types of error, parity error, framing error, and overrun error, may occur during reception. If the error flag in the 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 14-4 shows the causes of the reception error.

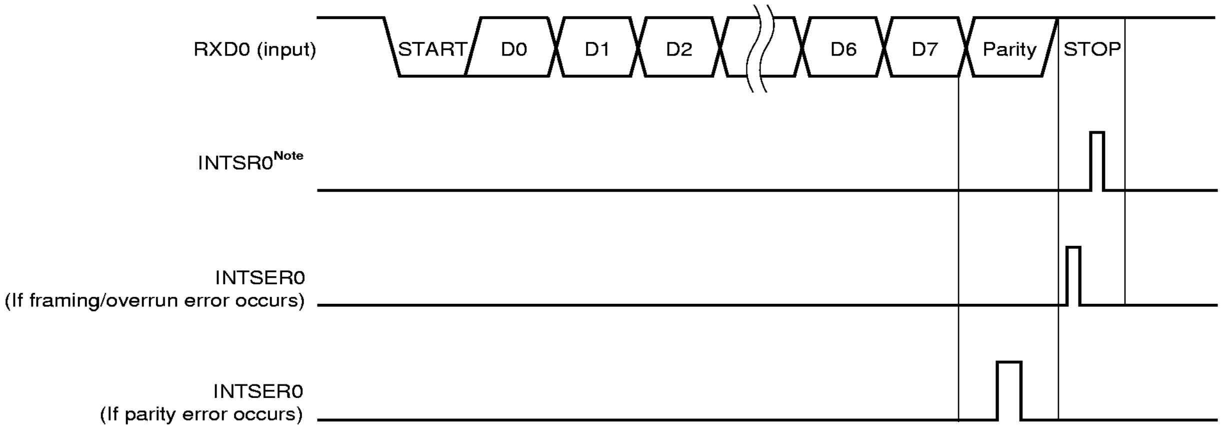
Which error has occurred during reception can be identified by reading the contents of ASIS0 in reception error processing (INTSER0) (refer to **Table 14-4** and **Figure 14-9**).

The contents of ASIS0 are reset when the 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 14-4. Causes of Reception Errors

Reception Errors	Causes	Value of ASIS0
Parity error	Parity specified for transmission does not coincide with parity of receive data.	04H
Framing error	Stop bit is not detected.	02H
Overrun error	Reception is completed at next data before data is read from receive buffer register 0.	01H

Figure 14-9. 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 the asynchronous serial interface status register 0 (ASIS0) are reset to 0 when the 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 the RXB0 even if a reception error occurs. Otherwise, an overrun error occurs when the next data is received, and the reception error status persists as a result.

[MEMO]

CHAPTER 15 INTERRUPT FUNCTIONS

15.1 Interrupt Function Types

The following three types of interrupt functions are used.

(1) Non-maskable interrupt

This interrupt is acknowledged unconditionally even if interrupt is 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 interrupt priority group and a low interrupt priority group by setting the priority specification flag register (PR).

Multiple high priority interrupts can be applied to low priority interrupts. If two or more interrupts with the same priority are simultaneously generated, each interrupts has a predetermined priority (refer to **Table 15-1**).

A standby release signal is generated.

Maskable interrupts are provided for each product as follows:

• μPD178022, 178023, 178024	Internal: 8,	external: 5
• μPD178122, 178123, 178124	Internal: 9,	external: 5
• μPD178F124	Internal: 12,	external: 5

(3) Software interrupt

This is a vectored interrupt to be generated by executing the BRK instruction. It is acknowledged even in a disabled state. The software interrupt does not undergo interrupt priority control.

15.2 Interrupt Sources and Configuration

The μPD178022, 178023, and 178024 have a total of 14 interrupt sources, the μPD178122, 178123, and 178124 have a total of 15 interrupt sources, including non-maskable, maskable, and software interrupts. The μPD178F124 have a total of 18 sources (refer to **Table 15-1**).

Remark As the interrupt source of the watchdog timer (INTWDT), either a non-maskable interrupt or a maskable interrupt (internal) can be selected.

Table 15-1. Interrupt Sources (1/3)

(1) μPD178022, 178023, 178024

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	INTKY	Detection of key input of port 4	Internal	0010H	(B)
	7	INTIIC0	End of transfer by serial interface IIC0		0012H	
	8	INTBTM0	Generation of basic timer match signal		0014H	
	9	INTAD3	End of conversion by A/D converter		0016H	
	10	–	–	–	0018H[Note 3]	–
	11	INTCSI3	End of transfer by serial interface SIO3	Internal	001AH	(B)
	12	INTTM50	Generation of coincidence signal of 8-bit timer/event counter 50		001CH	
	13	INTTM51	Generation of coincidence signal of 8-bit timer/event counter 51		001EH	
Software	–	BRK	Execution of BRK instruction	–	003EH	(D)

Notes 1. If two or more maskable interrupts occur at the same time, they are acknowledged or kept pending according to their default priorities. The default priority 0 is the highest, while 13 is the lowest.

2. (A) to (D) under the heading Basic Configuration Type corresponds to (A) to (D) in Figure 15-1.

3. There are no interrupt sources corresponding to vector addresses 0018H.

Table 15-1. Interrupt Sources (2/3)

(2) μPD178122, 178123, 178124

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	INTKY	Detection of key input of port 4	Internal	0010H	(B)
	7	INTIIC0	End of transfer by serial interface IIC0		0012H	
	8	INTBTM0	Generation of basic timer match signal		0014H	
	9	INTAD3	End of conversion by A/D converter		0016H	
	10	INTEE	End of EEPROM write		0018H	
	11	INTCSI3	End of transfer by serial interface SIO3		001AH	
	12	INTTM50	Generation of coincidence signal of 8-bit timer/event counter 50		001CH	
	13	INTTM51	Generation of coincidence signal of 8-bit timer/event counter 51		001EH	
Software	–	BRK	Execution of BRK instruction	–	003EH	(D)

Notes 1. If two or more maskable interrupts occur at the same time, they are acknowledged or kept pending according to their default priorities. The default priority 0 is the highest, while 13 is the lowest.

2. (A) to (D) under the heading Basic Configuration Type corresponds to (A) to (D) in Figure 15-1.

Table 15-1. Interrupt Sources (3/3)

(3) μPD178F124

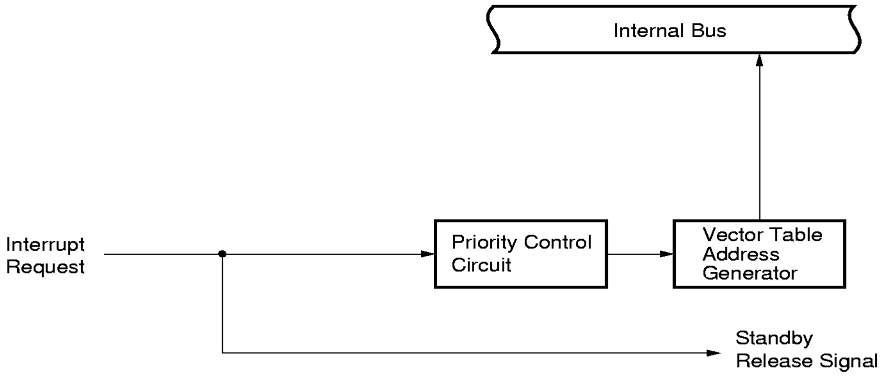
Interrupt Type	Default Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	–	INTWDT	Overflow of watchdog timer (when watchdog timer mode 1 is selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Overflow of watchdog timer (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	INTKY	Detection of key input of port 4	Internal	0010H	(B)
	7	INTIIC0	End of transfer by serial interface IIC0		0012H	
	8	INTBTM0	Generation of basic timer match signal		0014H	
	9	INTAD3	End of conversion by A/D converter		0016H	
	10	INTEE	End of EEPROM write		0018H	
	11	INTCSI3	End of transfer by serial interface SIO3		001AH	
	12	INTTM50	Generation of coincidence signal of 8-bit timer/event counter 50		001CH	
	13	INTTM51	Generation of coincidence 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	
Software	–	BRK	Execution of BRK instruction	–	003EH	(D)

Notes 1. If two or more maskable interrupts occur at the same time, they are acknowledged or kept pending according to their default priorities. The default priority 0 is the highest, while 16 is the lowest.

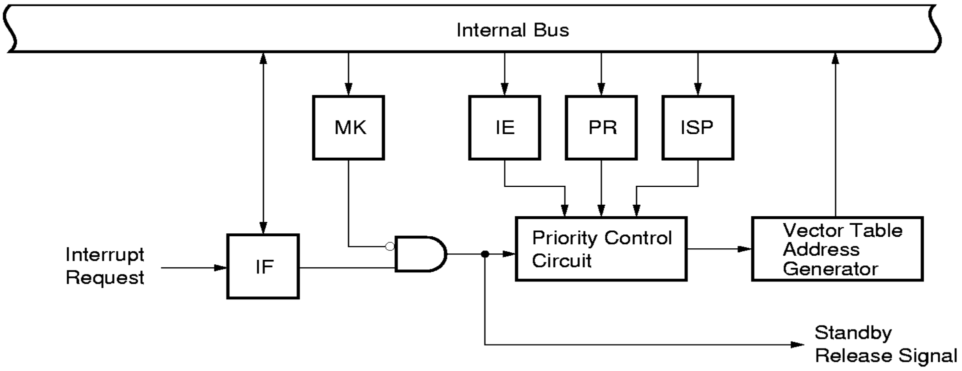
2. (A) to (D) under the heading Basic Configuration Type corresponds to (A) to (D) in Figure 15-1.

Figure 15-1. Basic Configuration of Interrupt Function (1/2)

(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt

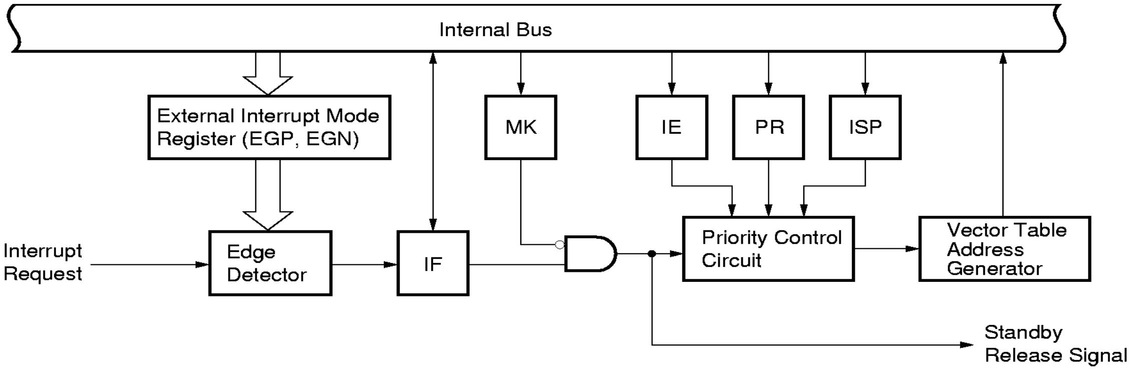
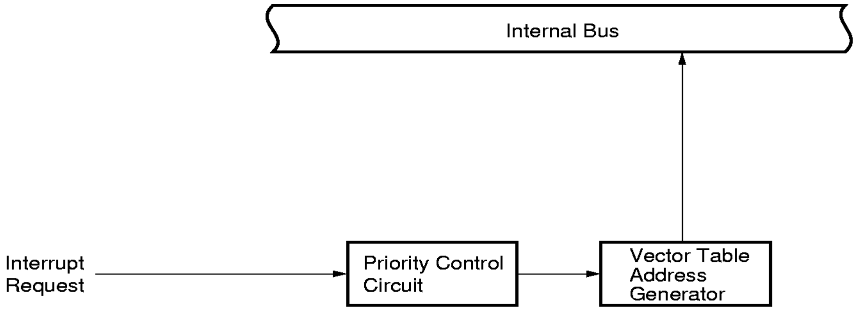


Figure 15-1. Basic Configuration of Interrupt Function (2/2)

(D) Software interrupt



Remark	IF	:	Interrupt request flag
	IE	:	Interrupt enable flag
	ISP	:	Inservice priority flag
	MK	:	Interrupt mask flag
	PR	:	Priority specification flag

15.3 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 15-2 gives a listing of interrupt request flags, interrupt mask flags, and priority specification flags corresponding to interrupt request sources.

Table 15-2. Various 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	
INTKY	KYIF		KYMK		KYPR	
INTIIC0	IICIF0		IICMK0		IICPR0	
INTBTM0	BTMIF0	IF0H	BTMMK0	MK0H	BTMPR0	PR0H
INTAD3	ADIF		ADMK		ADPR	
INTEE[Note 1]	EEIF[Note 1]		EEMK[Note 1]		EEPR[Note 1]	
INTCSI3	CSIIF3		CSIMK3		CSIPR3	
INTTM50	TMIF50		TMMK50		TMPR50	
INTTM51	TMIF51		TMMK51		TMPR51	
INTSER0[Note 2]	SERIF0[Note 2]		SERMK0[Note 2]		SERPR0[Note 2]	
INTSR0[Note 2]	SRIF0[Note 2]		SRMK0[Note 2]		SRPR0[Note 2]	
INTST0[Note 2]	STIF0[Note 2]	IF1L[Note 2]	STMK0[Note 2]	MK1L[Note 2]	STPR0[Note 2]	PR1L[Note 2]

Notes 1. μPD178122, 178123, 178124, and 178F124 only

2. μPD178F124 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 with a 1-bit or 8-bit memory manipulation instruction. If IF0L and IF0H are used as a 16-bit register IF0, use a 16-bit memory manipulation instruction for the setting.

Reset input sets these registers to 00H.

Figure 15-2. Format of Interrupt Request Flag Register (IF0L, IF0H, IF1L)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	At reset	R/W
IF0L	IICIF0	KYIF	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	EEIF[Note 2]	ADIF	BTMIF0	FFE1H	00H	R/W
	7	6	5	4	3	2	1	<0>			
IF1L	0	0	0	0	0	0	0	STIF0[Note 1]	FFE2H	00H	R/W

××IDx	Interrupt Request Flag
0	No interrupt request signal
1	Interrupt request signal is generated; Interrupt request state

Notes 1. These bits are provided on the μPD178F124 only. Be sure to reset these bits of the μPD178022, 178023, 178024, 178122, 178123, and 178124 to "0".

2. These bits are provided on the μPD178122, 178123, 178124, and 178F124 only. Be sure to reset these bits of the μPD178022, 178023, and 178024 to "0".

Cautions 1. TMIF4 flag is R/W enabled only when a watchdog timer is used as an interval timer. If a watchdog timer is used in watchdog timer mode 1, set TMIF4 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 service and to set standby clear enable/disable.

MK0L, MK0H and MK1L are set with a 1-bit or 8-bit memory manipulation instruction. If MK0L and MK0H are used as a 16-bit register MK0, use a 16-bit memory manipulation instruction for the setting.

Reset input sets these registers to FFH.

Figure 15-3. Format of Interrupt Mask Flag Register (MK0L, MK0H, MK1L)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	At reset	R/W
MK0L	IICMK0	KYMK	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	EEIF[Note 2]	ADMK	BTMMK0	FFE5H	FFH	R/W
	7	6	5	4	3	2	1	<0>			
MK1L	1	1	1	1	1	1	1	STMK0[Note 1]	FFE6H	FFH	R/W

××MK×	Interrupt Service Control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Notes 1. These bits are provided on the μPD178F124 only. Be sure to reset these bits of the μPD178022, 178023, 178024, 178122, 178123, and 178124 to “0”.

2. These bits are provided on the μPD178122, 178123, 178124, and 178F124 only. Be sure to reset these bits of the μPD178022, 178023 and 178024 to “0”.

Cautions 1. If WDTMK flag is read when a watchdog timer is used in watchdog timer mode 1, MK0 value becomes undefined.

2. Because port 0 has a dual function as the 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, 1 should be set in the interrupt mask flag 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 orders.

PR0L, PR0H and PR1L are set with a 1-bit or 8-bit memory manipulation instruction. If PR0L and PR0H are used as a 16-bit register PR0, use a 16-bit memory manipulation instruction for the setting.

Reset input sets these registers to FFH.

Figure 15-4. Format of Priority Specification Flag Register (PR0L, PR0H, PR1L)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	At reset	R/W
PR0L	IICPR0	KYPR	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	EEPR[Note 2]	ADPR	BTMPR0	FFE9H	FFH	R/W
	7	6	5	4	3	2	1	<0>			
PR1L	1	1	1	1	1	1	1	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 on the μPD178F124 only. Be sure to reset these bits of the μPD178022, 178023, 178024, 178123, and 178124 to "0".

2. These bits are provided on the μPD178122, 178123, 178124, and 178F124 only. Be sure to reset these bits of the μPD178022, 178023, and 178024 to "0".

Caution When a watchdog timer is used in watchdog timer mode 1, set 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 INTP4.

EGP and EGN are set with a 1-bit or 8-bit memory manipulation instructions.

Reset input sets these registers to 00H.

Figure 15-5. 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	At reset	R/W
EGP	0	0	0	EGP4	EGP3	EGP2	EGP1	EGP0	FF48H	00H	R/W
	7	6	5	4	3	2	1	0			
EGN	0	0	0	EGN4	EGN3	EGN2	EGN1	EGN0	FF49H	00H	R/W

EGPn	EGNn	INTPn Pin Valid Edge Selection (n = 0-4)
0	0	Interrupt prohibited
0	1	Falling edge
1	0	Rising edge
1	1	Both falling and rising edges

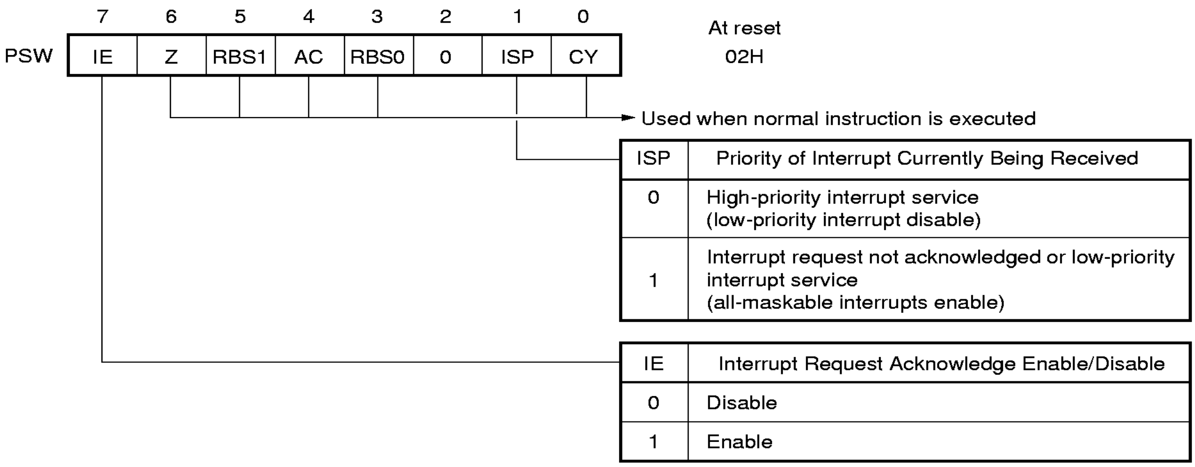
(5) Program status word (PSW)

The program status word is a register to hold the instruction execution result and the current status for interrupt request. The IE flag to set maskable interrupt enable/disable and the ISP flag to control nesting interrupt are mapped.

Besides 8-bit unit read/write, this register can carry out operations with a bit manipulation instruction and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged, if the BRK instruction is executed, the contents of PSW are automatically saved into a stack and the IE flag is reset to 0. If a maskable interrupt request is acknowledged 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 with the PUSH PSW instruction. It is restored from the stack with the RETI, RETB, and POP PSW instructions.

Reset input sets PSW to 02H.

Figure 15-6. Configuration of Program Status Word (PSW)



	7	6	5	4	3	2	1	0	At reset
PSW	IE	Z	RBS1	AC	RBS0	0	ISP	CY	02H

ISP	Priority of Interrupt Currently Being Received
0	High-priority interrupt service (low-priority interrupt disable)
1	Interrupt request not acknowledged or low-priority interrupt service (all-maskable interrupts enable)

IE	Interrupt Request Acknowledge Enable/Disable
0	Disable
1	Enable

15.4 Interrupt Service Operations

15.4.1 Non-maskable interrupt request acknowledge operation

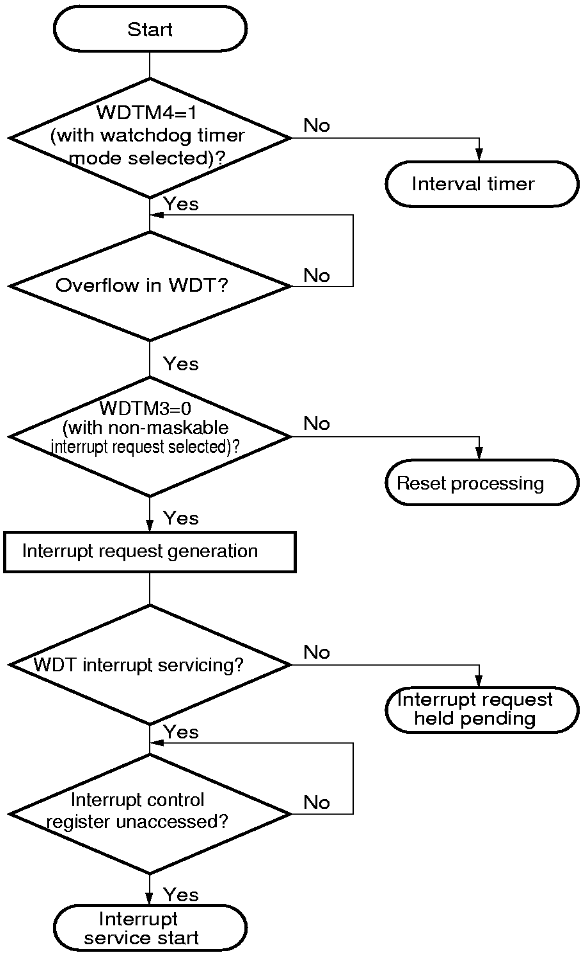
A non-maskable interrupt request is unconditionally acknowledged even if in an interrupt request acknowledge disable state. It does not undergo interrupt priority control and has highest priority over all other interrupts.

If a non-maskable interrupt request is acknowledged, the acknowledged interrupt is saved in the stacks, 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 PC and branched.

A new non-maskable interrupt request generated during execution of a non-maskable interrupt servicing program is acknowledged after the current execution of the non-maskable interrupt servicing program is terminated (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 service program execution, only one non-maskable interrupt request is acknowledged after termination of the non-maskable interrupt service program execution.

Figure 15-7 shows the flowchart from generation of the non-maskable interrupt request to acknowledging it. Figure 15-8 shows the timing of acknowledging the non-maskable interrupt request, and Figure 15-9 shows the operation performed if a more than one non-maskable interrupt request occurs.

Figure 15-7. Flowchart from Generation of Non-Maskable Interrupt Request to Acknowledge



WDTM	:	Watchdog timer mode register
WDT	:	Watchdog timer

Figure 15-8. Non-Maskable Interrupt Request Acknowledge Timing

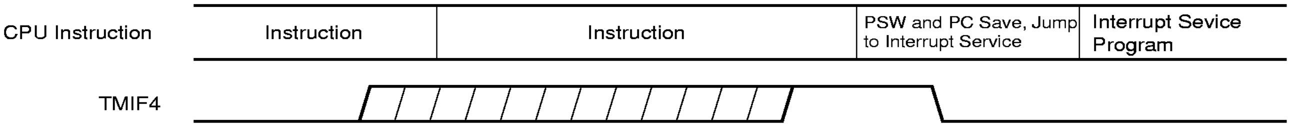
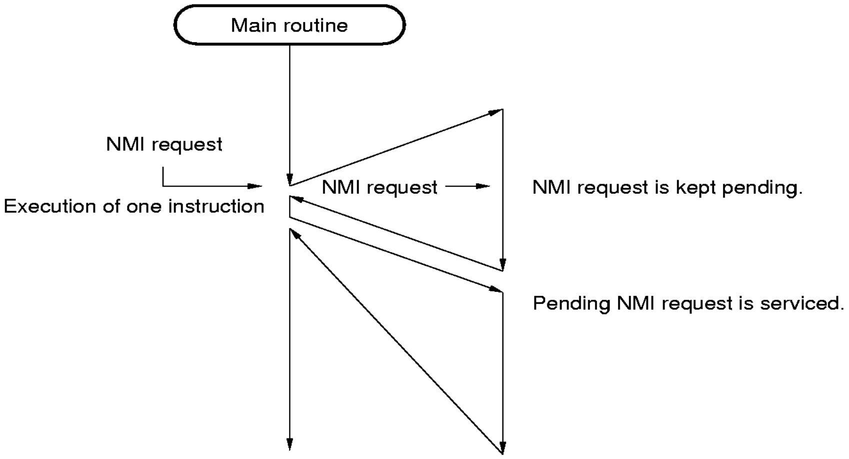
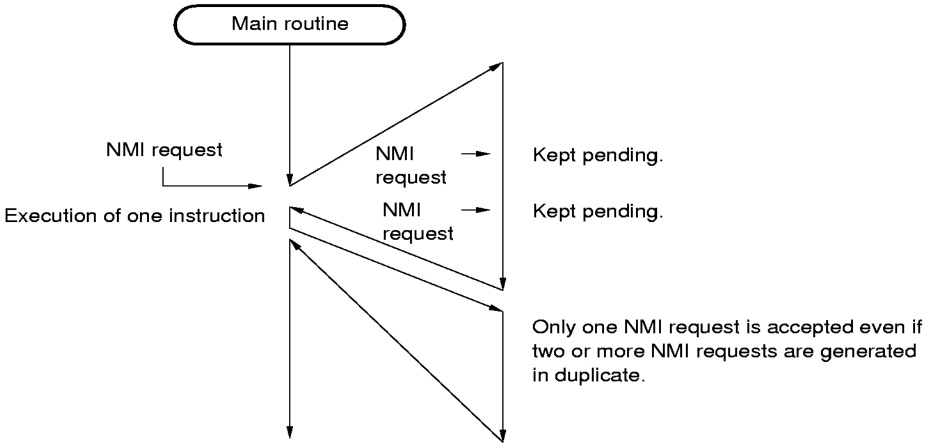


Figure 15-9. Non-Maskable Interrupt Request Acknowledge Operation

(a) If a new non-maskable interrupt request is generated during non-maskable interrupt service program execution


Main routine
NMI request
Execution of one instruction
NMI request
NMI request is kept pending.
Pending NMI request is serviced.

(b) If two non-maskable interrupt requests are generated during non-maskable interrupt service program execution


Main routine
NMI request
Execution of one instruction
NMI request
Kept pending.
NMI request
Kept pending.
Only one NMI request is accepted even if two or more NMI requests are generated in duplicate.

15.4.2 Maskable interrupt request acknowledge 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 enable state (with IE flag set to 1). However, a low-priority interrupt request is not acknowledged during high-priority interrupt service (with ISP flag reset to 0).

Wait times maskable interrupt request generation to interrupt request service are as follows.

For the interrupt acknowledge timing, refer to **Figures 15-11** and **15-12**.

Table 15-3. Times from Maskable Interrupt Request Generation to Interrupt Service

	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 for higher priority with the priority specification flag is acknowledged first. Two or more requests specified for the same priority by the priority specification flag, the default priorities apply.

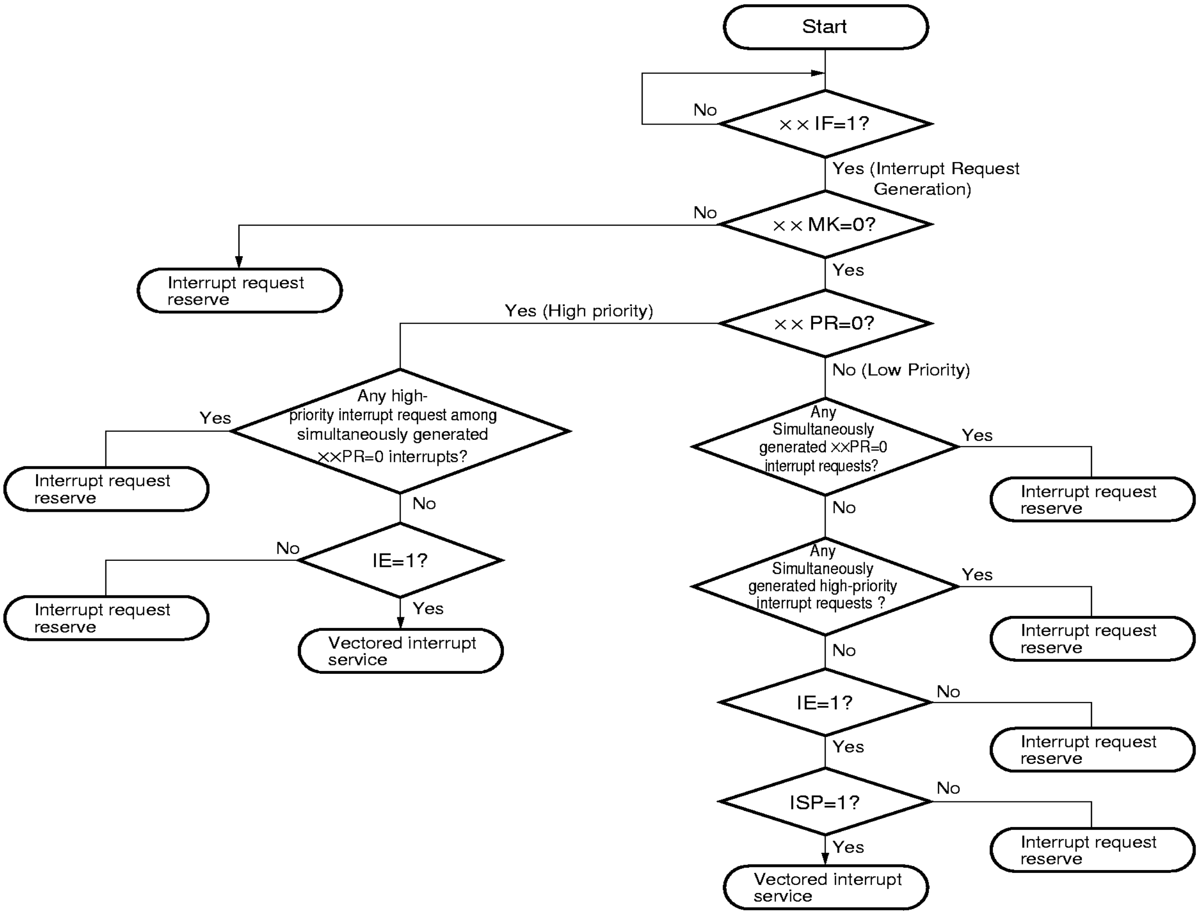
Any reserved interrupt requests are acknowledged when they become acknowledgeable.

Figure 15-10 shows interrupt request acknowledge algorithms.

If a maskable interrupt request is acknowledged, the acknowledged interrupt request is saved in the stacks, 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 PC and branched.

Return from the interrupt is possible with the RETI instruction.

Figure 15-10. Interrupt Request Acknowledge Processing Algorithm



Start
××IF=1?
No
Yes (Interrupt Request Generation)
××MK=0?
No
Interrupt request reserve
Yes
××PR=0?
Yes (High priority)
No (Low Priority)
Any high-priority interrupt request among simultaneously generated ××PR=0 interrupts?
Yes
Interrupt request reserve
No
IE=1?
No
Interrupt request reserve
Yes
Vectored interrupt service
Any Simultaneously generated ××PR=0 interrupt requests?
Yes
Interrupt request reserve
No
Any Simultaneously generated high-priority interrupt requests ?
Yes
Interrupt request reserve
No
IE=1?
No
Interrupt request reserve
Yes
ISP=1?
No
Interrupt request reserve
Yes
Vectored interrupt service

××IF : Interrupt request flag

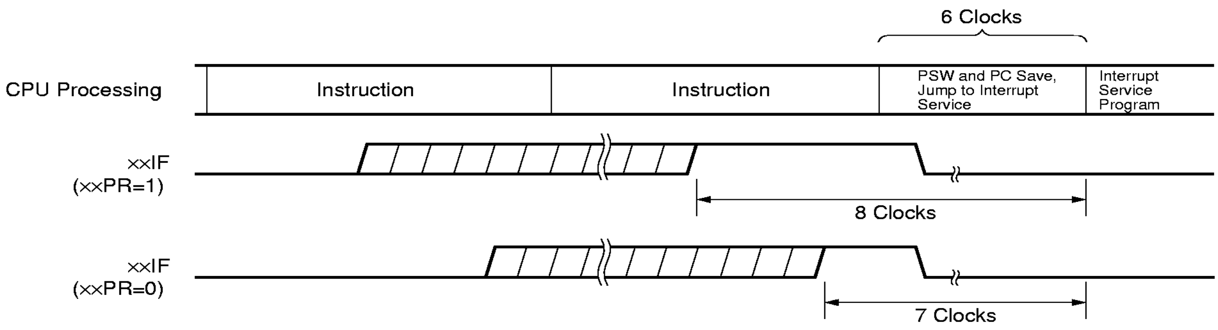
××MK : Interrupt mask flag

××PR : Priority specification flag

IE : Flag controlling acknowledging maskable interrupt request (1 = enable, 0 = disable)

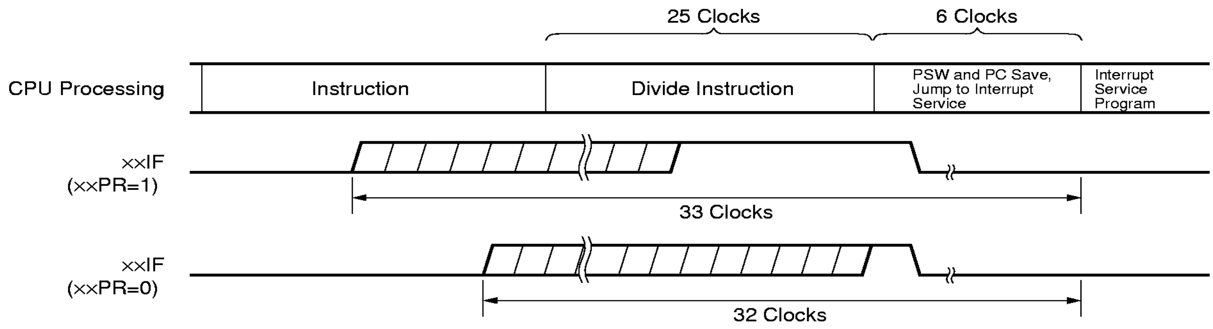
ISP : Flag indicating priority of interrupt currently serviced (0 = interrupt with high priority serviced, 1 = interrupt request is not acknowledged, or interrupt with low priority serviced)

Figure 15-11. Interrupt Request Acknowledge Timing (Minimum Time)



Remark 1 clock: $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

Figure 15-12. Interrupt Request Acknowledge Timing (Maximum Time)



Remark 1 clock: $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

15.4.3 Software interrupt request acknowledge operation

A software interrupt request is acknowledged by BRK instruction execution. Software interrupt cannot be disabled.

If a software interrupt request is acknowledged, it is saved in the stacks, the program status word (PSW) and 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 PC and branched.

Return from the software interrupt is possible with the RETB instruction.

Caution Do not use the RETI instruction for returning from the software interrupt.

15.4.4 Nesting interrupt

Accepting another interrupt request while an interrupt is being serviced is called nesting interrupts.

Nesting does not take place unless the interrupts (except the non-maskable interrupt) are enabled to be acknowledged (IE = 1). Accepting another interrupt request is disabled (IE = 0) when one interrupt has been acknowledged. Therefore, to enable nesting, the EI flag must be set to 1 during interrupt servicing, to enable the another interrupt.

Nesting interrupts may not occur even when the 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, nesting is controlled by using the programmable priority.

If an interrupt with the same level of priority as or the higher priority than the interrupt currently serviced occurs, that interrupt can be acknowledged and nested. If an interrupt with a priority lower than that of the currently serviced interrupt occurs, that interrupt cannot be acknowledged and nested.

An interrupt that is not acknowledged and nested because it is disabled or it has a low priority is kept pending. This interrupt is acknowledged after servicing of the current interrupt has been completed and one instruction of the main routine has been executed.

Nesting is not enabled while the non-maskable interrupt is being serviced.

Table 15-4 shows the interrupts that can be nested, and Figure 15-13 shows an example of nesting.

Table 15-4. Interrupt Request Enabled for Nesting Interrupt during Interrupt Service

Nesting 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 : Nesting interrupt enable

2. D : Nesting interrupt disable

3. ISP and IE are the flags contained in PSW

ISP = 0 : An interrupt with higher priority is being serviced

ISP = 1 : An interrupt request is not accepted or an interrupt with lower priority is being serviced

IE = 0 : Interrupt request acknowledge is disabled

IE = 1 : Interrupt request acknowledge is enabled

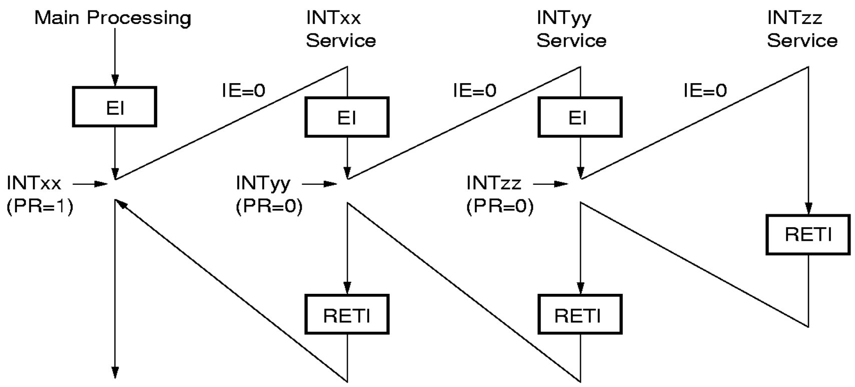
4. PR is a flag contained in PR0L and PR0R.

PR = 0 : Higher priority level

PR = 1 : Lower priority level

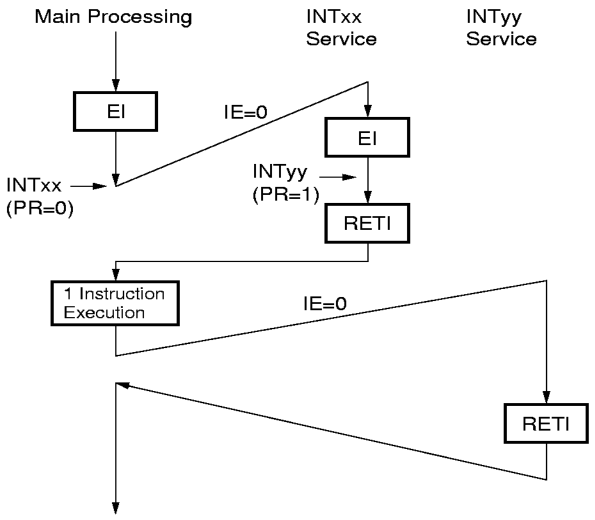
Figure 15-13. Nesting Interrupt Example (1/2)

Example 1. Example where nesting takes place two times



Two interrupt requests, INTyy and INTzz, are acknowledged while interrupt INTxx is serviced, and nesting takes place. Before each interrupt request is acknowledged, the EI instruction is always executed, and the interrupt is enabled.

Example 2. Example where nesting does not take place because of priority control

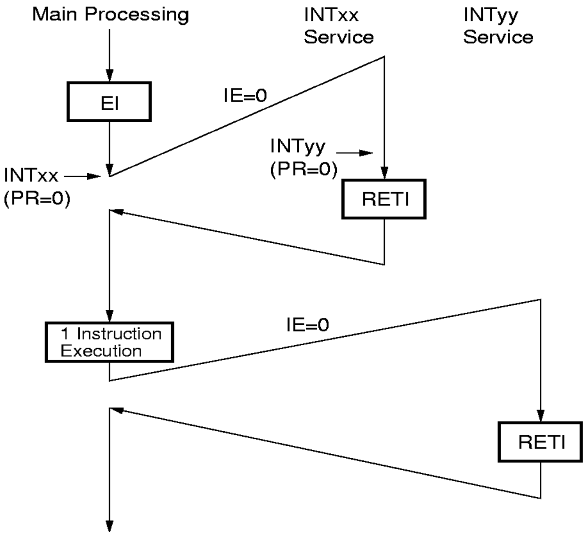


Interrupt request INTyy that is generated while interrupt INTxx is being serviced is not acknowledged because its priority is lower than that of INTxx, and therefore, nesting does not take place. INTyy request is kept pending, and is acknowledged after one instruction of the main routine has been executed.

PR = 0 : High-priority level
PR = 1 : Low-priority level
IE = 0 : Acknowledging interrupt request is disabled.

Figure 15-13. Nesting Interrupt Example (2/2)

Example 3. Example where nesting does not take place because interrupts are not enabled



Because interrupts are not enabled (EI instruction is not issued) in interrupt processing INTxx, interrupt request INTyy is not acknowledged, and nesting does not take place. INTyy request is kept 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.

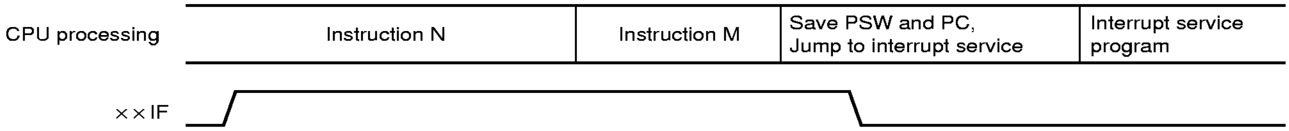
15.4.5 Pending interrupt requests

Even if an interrupt request is generated, the following instructions keep 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 executed. However, the non-maskable interrupt is acknowledged.

Figure 15-14. Pending Interrupt Request



Remarks 1. Instruction N: Instruction that keeps interrupt request pending

2. Instruction M: Instruction that does not keep interrupt request pending

3. Operation of ××IF is not affected by value of ××PR.

[MEMO]

CHAPTER 16 PLL FREQUENCY SYNTHESIZER

16.1 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 reference frequency from the EO0 and EO1 pin.

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 16-1 shows the division modes, input pins used (VCOL pin or VCOH pin), and the value that can be set to the programmable divider.

Table 16-1. 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 be actually input, and input amplitude, refer to Electrical Specifications in Data Sheet.

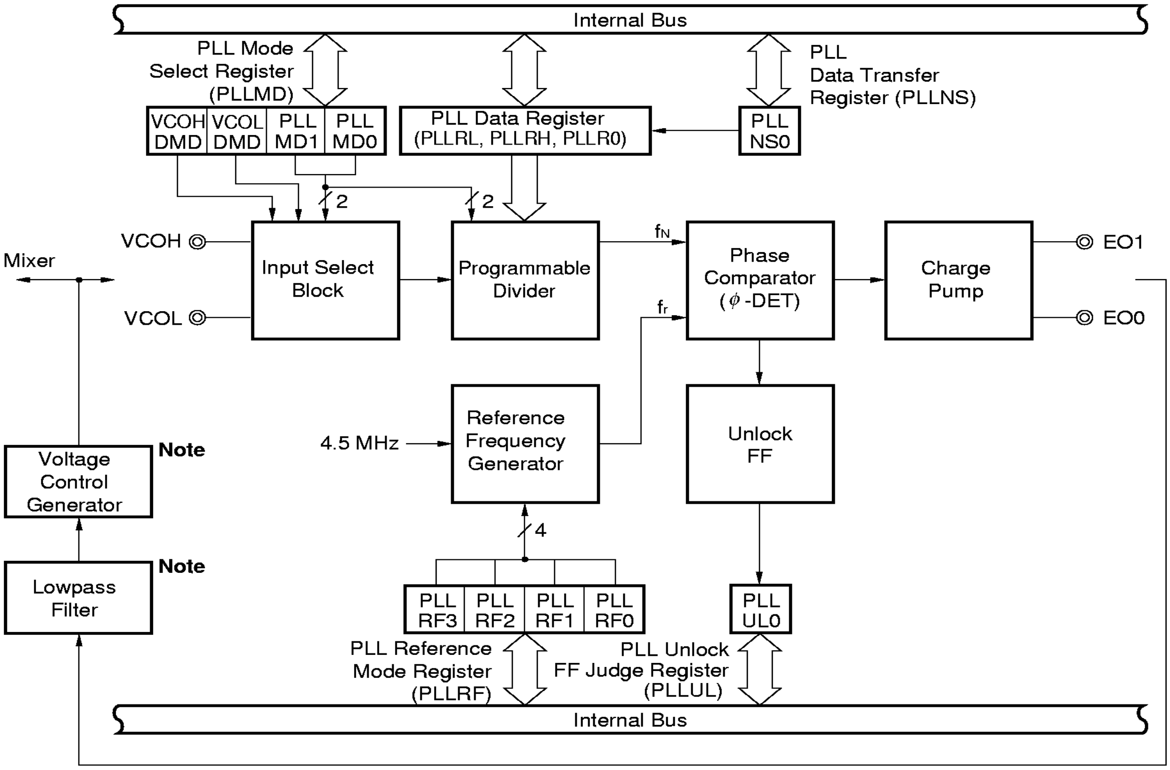
16.2 Configuration of PLL Frequency Synthesizer

The PLL frequency synthesizer consists of the following hardware units.

Table 16-2. Configuration of PLL Frequency Synthesizer

Item	Configuration
Data register	PLL data register L (PLLRL) PLL data register H (PLLRH) PLL data register 0 (PLLR0)
Control register	PLL mode select register (PLLMD) PLL reference mode register (PLLRF) PLL unlock FF judge register (PLLUL) PLL data transfer register (PLLNS)

Figure 16-1. 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 high-order 16 bits of this value are set by the PLL data register L (PLLRL) and PLL data register H (PLLRH). The high-order 16 bits can also be set by the PLL data register (PLLR).

The least significant bit is set by bit 7 (PLLSCN) of the PLL data register 0 (PLLR0).

The contents of these registers are undefined at 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 ($\overline{\text{UP}}$) and down request signal ($\overline{\text{DW}}$).

(6) Unlock FF

The unlock FF detects the unlock status of the PLL frequency synthesizer from the up request signal ($\overline{\text{UP}}$) and down request signal ($\overline{\text{DW}}$) 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).

16.3 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 using a 1-bit or 8-bit memory manipulation instruction.

The value of this register is set to 00H at 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, it holds the value immediately before the HALT mode was set.

Figure 16-2. Format of PLL Mode Select Register (PLLMD)

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	At reset	R/W
PLLMD	0	0	0	0	VCOHDMD	VCOLDMD	PLLMD1	PLLMD0	FFA0H	00H	R/W

VCOH DMD	Selects Disable Status of VCOH Pin
0	Connected to pull-down resistor.
1	High-impedance status

VCOL DMD	Selects Disable Status of VCOL Pin
0	Connected to pull-down resistor.
1	High-impedance status

PLLMD1	PLLMD0	Selects Division Mode of PLL Frequency Synthesizer and VCO Input Pin
0	0	Disables VCOL and VCOH pins[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 pin go low.

Remark Bits 4 through 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 using 1-bit or 8-bit memory manipulation instruction.

The value of this register is set to 0FH at reset and in the STOP mode.

In the HALT mode, it holds the value immediately before the HALT mode was set.

Figure 16-3. Format of PLL Reference Mode Register (PLLRF)

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	At reset	R/W
PLLRF	0	0	0	0	PLLRF3	PLLRF2	PLLRF1	PLLRF0	FFA1H	0FH	R/W

PLLRF3	PLLRF2	PLLRF1	PLLRF0	Sets 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 through 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.

At reset, the value of this register is set to 0×H[Note 1].

In the STOP and HALT modes, this register holds the value immediately before the STOP or HALT mode was set.

Figure 16-4. Format of PLL Unlock FF Judge Register (PLLUL)

Symbol	7	6	5	4	3	2	1	<0>	Address	At reset	R/W
PLLUL	0	0	0	0	0	0	0	PLLUL0	FFA2H	0×H[Note 1]	R[Note 2]

PLLUL0	Detects 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) at 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
At reset	Power-ON clear	0	0	0	0	0	0	0	Undefined
	Watchdog timer	↓	↓	↓	↓	↓	↓	↓	Retained
	$\overline{\text{RESET}}$ input	↓	↓	↓	↓	↓	↓	↓	Retained
STOP mode		↓	↓	↓	↓	↓	↓	↓	Retained
HALT mode		↓	↓	↓	↓	↓	↓	↓	Retained

Remark Bits 1 through 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 at reset and in the STOP mode.

In the HALT mode, this register holds the previous value immediately before the HALT mode is set.

Figure 16-5. Format of PLL Data Transfer Register (PLLNS)

Symbol	7	6	5	4	3	2	1	<0>	Address	At reset	R/W
PLLNS	0	0	0	0	0	0	0	PLLNS0	FFA3H	00H	W

PLLNS0	Transfers Value of PLL Data Register to Programmable Counter and Swallow Counter
0	Does not transfer
1	Transfers

Remark Bits 0 through 7 are fixed to 0 by hardware.

16.4 Operation of PLL Frequency Synthesizer

16.4.1 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).

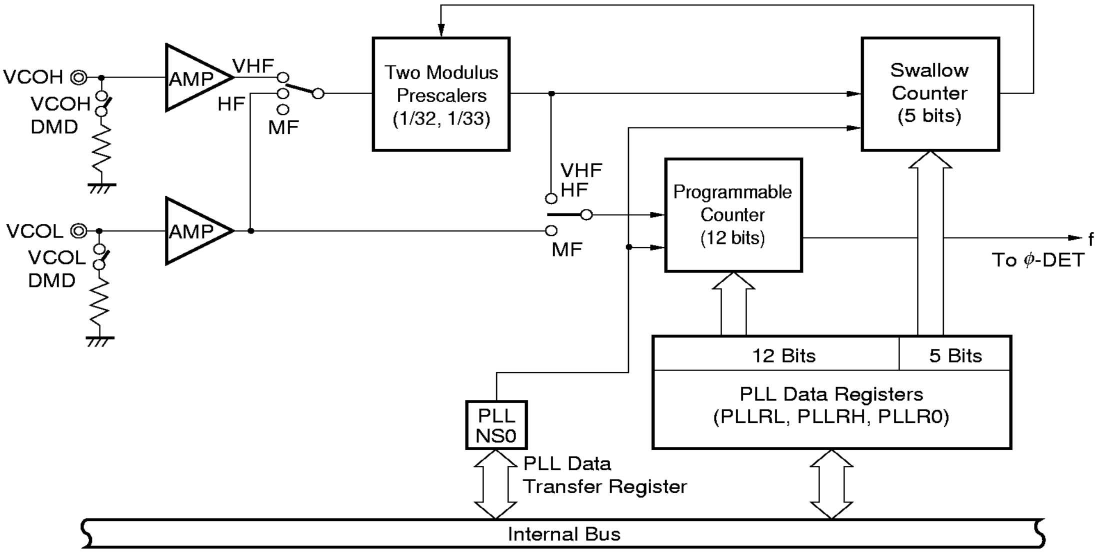
The programmable counter (12 bits) and pulse swallow counter (5 bits) are binary counters.

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 16-6 shows the configuration of the input select block and programmable divider.

Figure 16-6. Configuration of Input Select Block and Programmable Divider


VCOH
VCOH
DMD
VCOL
VCOL
DMD
AMP
AMP
VHF
HF
MF
Two Modulus Prescalers (1/32, 1/33)
Swallow Counter (5 bits)
VHF
HF
MF
Programmable Counter (12 bits)
fN
To φ-DET
12 Bits
5 Bits
PLL Data Registers (PLLRL, PLLRH, PLLR0)
PLL NS0
PLL Data Transfer Register
Internal Bus

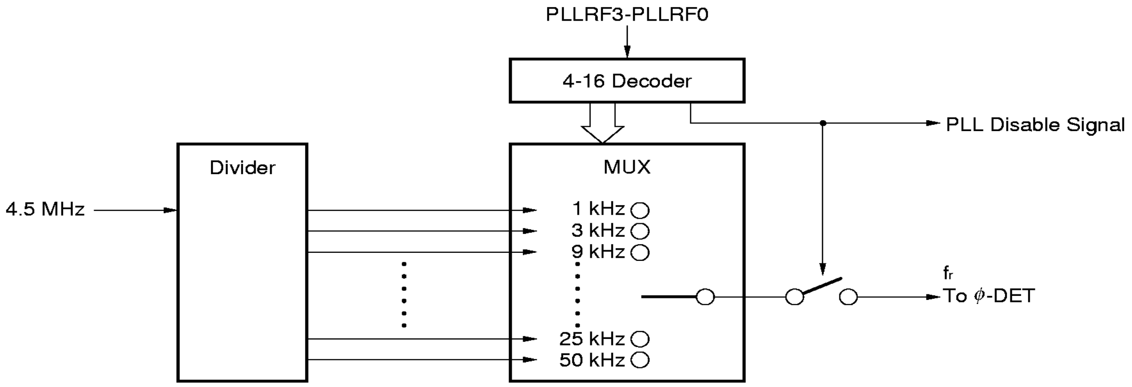
(2) Operation of reference frequency generator

The reference frequency generator divides the 4.5 MHz output of the crystal oscillator 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 16-7 shows the configuration of the reference frequency generator.

Figure 16-7. Configuration of Reference Frequency Generator



(3) Operation of phase comparator (ϕ-DET)

Figure 16-8 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, $\overline{\text{UP}}$, or a down request signal, $\overline{\text{DW}}$.

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 16-9 shows the relation among reference frequency f_r, divided frequency f_N, up request signal $\overline{\text{UP}}$, and down request signal $\overline{\text{DW}}$.

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 16-8. Configuration of Phase Comparator, Charge Pump, and Unlock FF

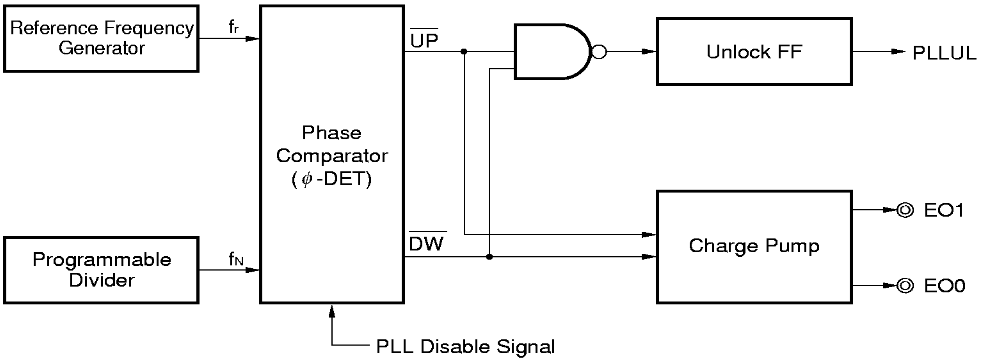
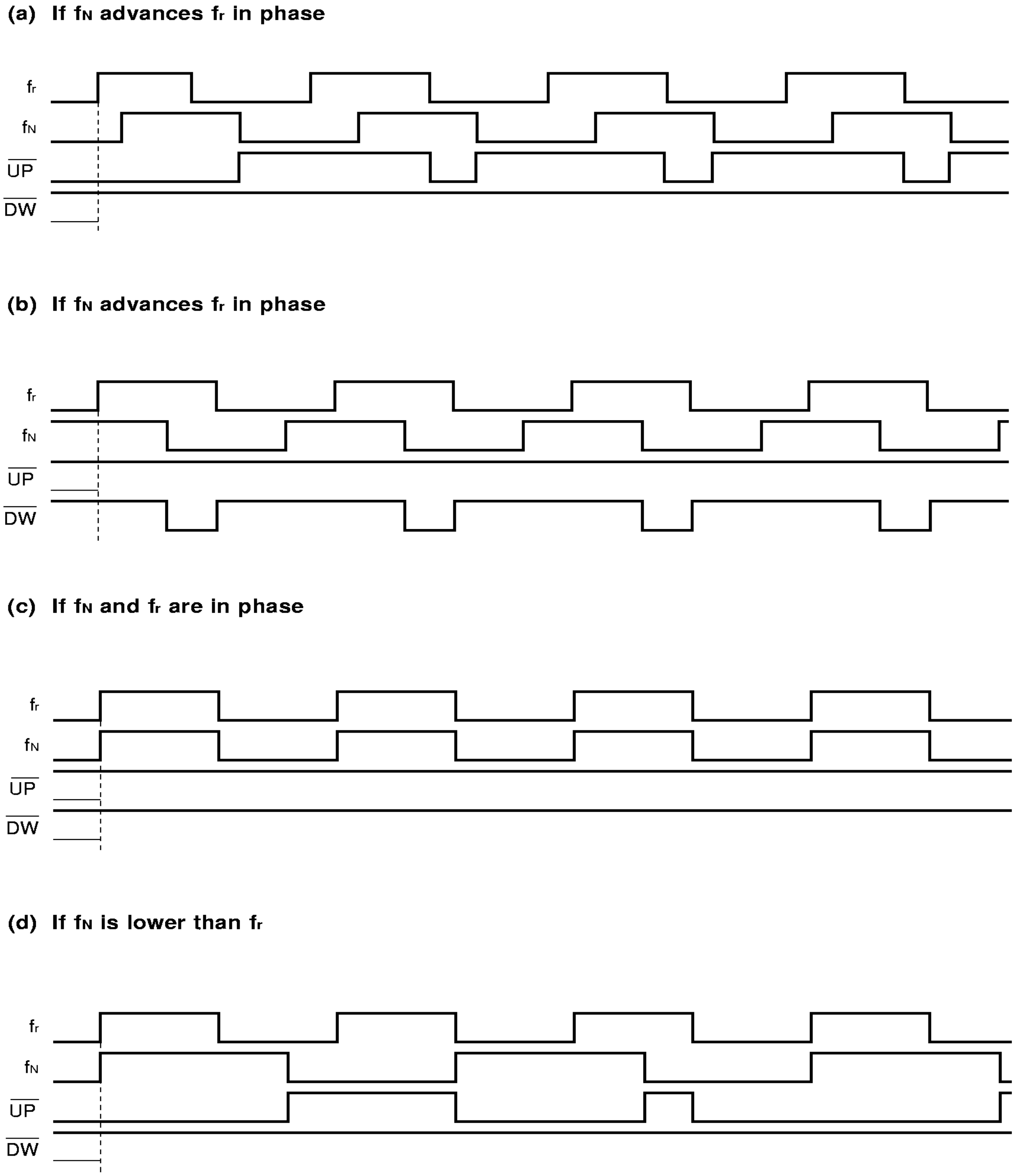


Figure 16-9. Relationship between f_r, f_N, $\overline{\text{UP}}$, and $\overline{\text{DW}}$



(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 ($\overline{\text{UP}}$) or down request ($\overline{\text{DW}}$) signal from the phase comparator (ϕ-DET) from the error out pins (EO0 and EO1 pins). Table 16-3 shows the output signals.

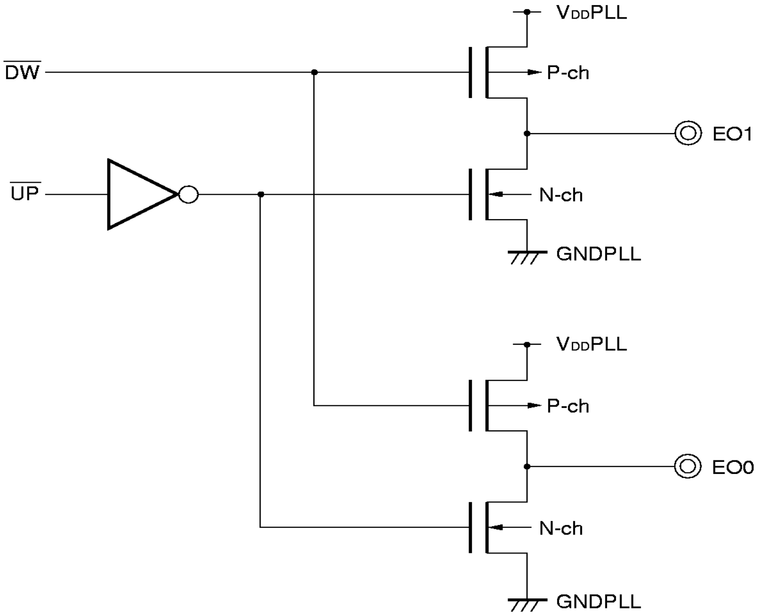
The EO0 pin is of voltage-driven type, and EO1 pin is of current-driven type.

Figure 16-10 shows the configuration of the error out pins.

Table 16-3. 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 16-10. Configuration of Error Out Pins



(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 $\overline{\text{UP}}$ and down request signal $\overline{\text{DW}}$ 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 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 PLLUL, therefore, it must be read at a cycle longer than the cycle ($1/f_r$) of the reference frequency.

16.4.2 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 broadcasting 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 broadcasting stations in the following SW band is shown below.

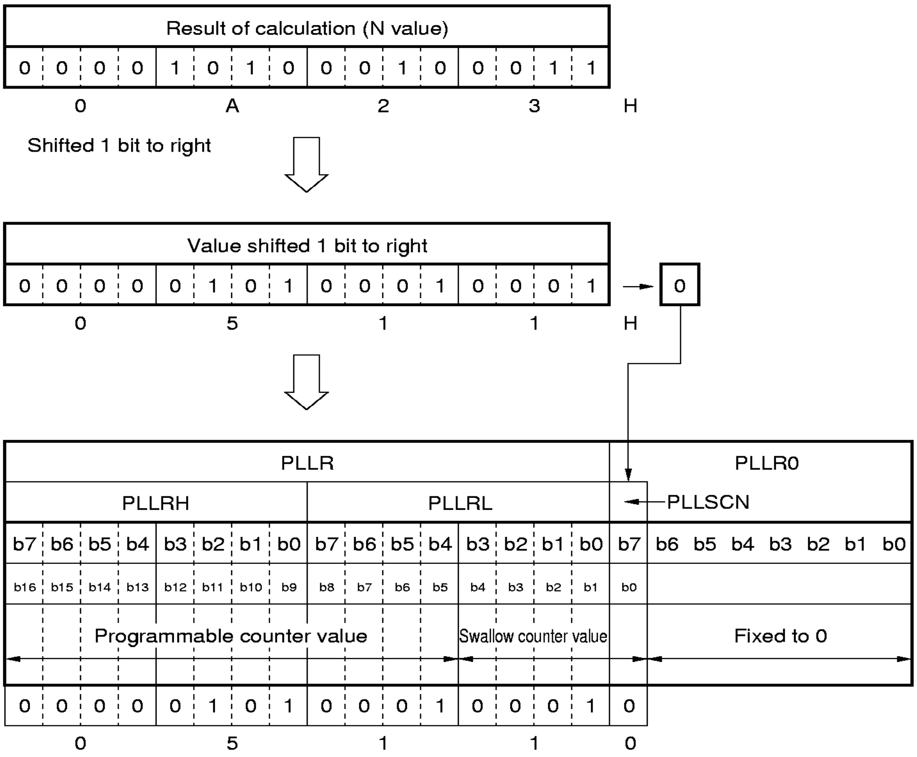
Receive frequency : 25.50 MHz (SW band)
Reference frequency : 10 kHz
Intermediate frequency : 450 kHz

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 the division value N must be set to bit 7 (PLLSCN) of the 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:


Result of calculation (N value)
0 0 0 0 1 0 1 0 0 0 1 0 0 0 1 1
0 A 2 3 H
Shifted 1 bit to right
Value shifted 1 bit to right
0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 1
0
0 5 1 1 H
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
Swallow counter value
Fixed to 0
0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 1 0
0 5 1 1 0

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 high-order 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 broadcasting stations in the following FM band is shown below.

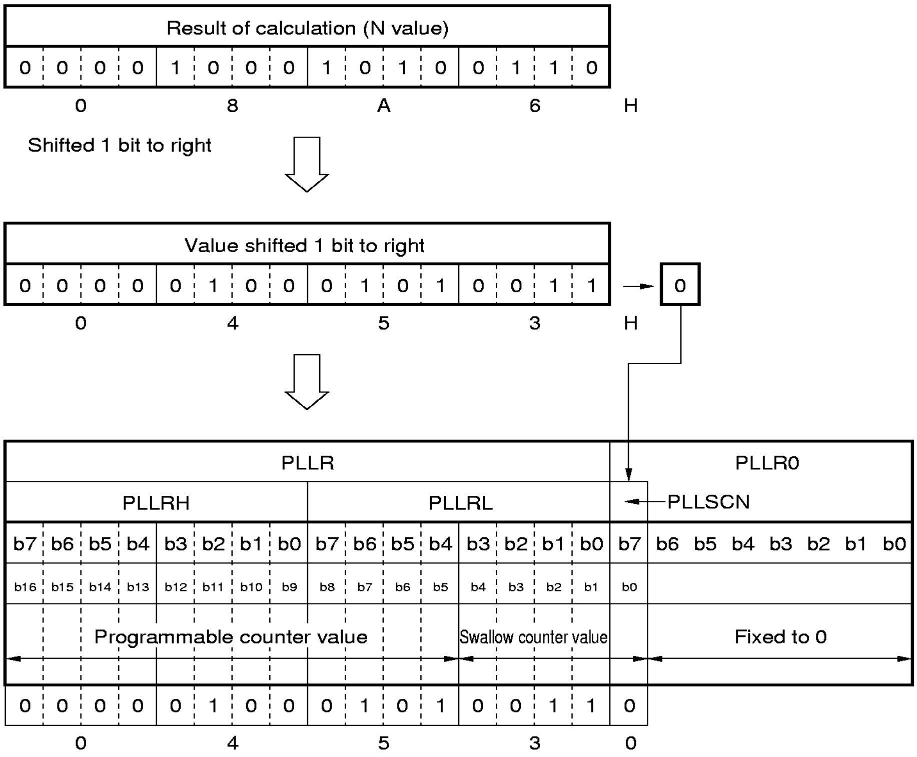
Receive frequency	:	100.0 MHz (FM band)
Reference frequency	:	50 kHz
Intermediate frequency	:	+10.7 MHz

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 the division value N must be set to the 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:


Result of calculation (N value)
0 0 0 0 1 0 0 0 1 0 1 0 0 1 1 0
0 8 A 6 H
Shifted 1 bit to right
Value shifted 1 bit to right
0 0 0 0 0 1 0 0 0 1 0 1 0 0 1 1
0
0 4 5 3 H
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
Swallow counter value
Fixed to 0
0 0 0 0 0 1 0 0 0 1 0 1 0 0 1 1 0
0 4 5 3 0

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 high-order 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}.

$$N_{PLLR} = \frac{f_{VCOH}}{2f_r}$$

16.5 PLL Disable Status

The PLL frequency synthesizer can be stopped (PLL disabled status) by performing any of the following settings while the PLL frequency synthesizer is operating.

- Setting value of bit 3 (PLLRF3) of the PLL reference mode register (PLLRF) to 1 to set PLL disabled status
- Setting STOP mode with the STOP instruction
- Setting reset status with the reset function

The following table shows the operation of each block and the status of each register in the PLL disabled status.

Table 16-4. Operation of Each Block and Register Status in PLL Disabled Status

Block/Register	Status in PLL Disabled 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	

16.6 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}). The input pin not selected becomes the status set in bit 3 (VCOHDMD) and bit 2 (VCOLDMD) of the PLL mode select register (PLLMD).
  For the frequencies that can be actually input and input amplitude, refer to Electrical Characteristics in Data Sheet.

CHAPTER 17 FREQUENCY COUNTER

17.1 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 to the IF counter register.

For the range of the frequency that can be input to the FMIFC and AMIFC pins, refer to Electrical Characteristics in Data Sheet.

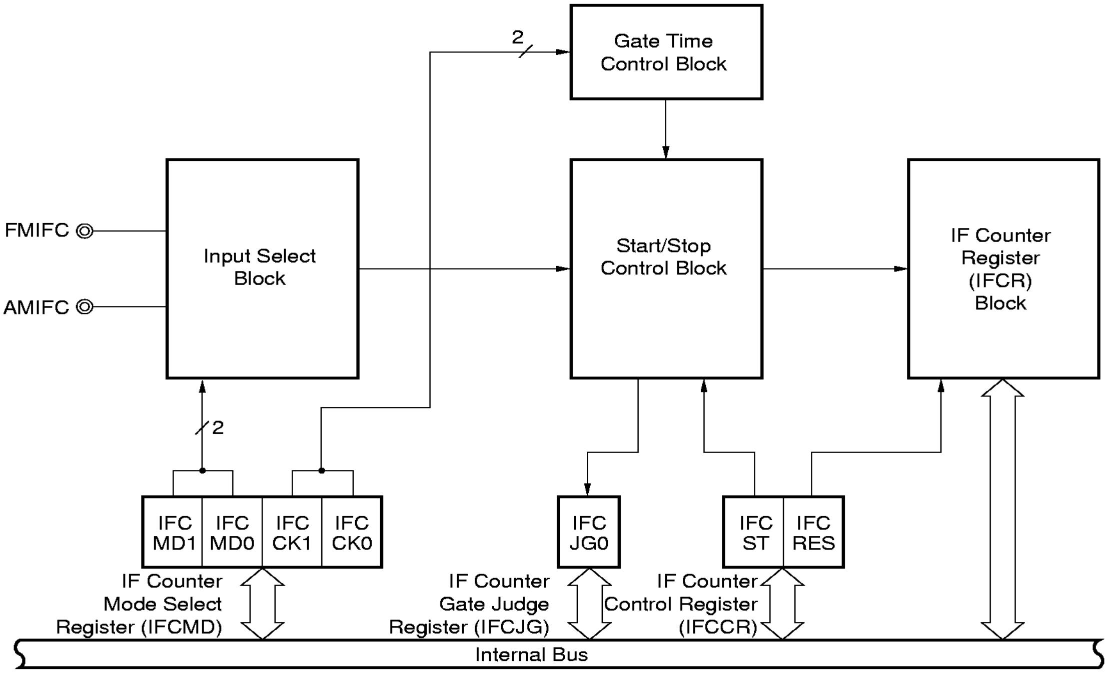
17.2 Configuration of Frequency Counter

The frequency counter consists of the following hardware units.

Table 17-1. Configuration of Frequency Counter

Item	Configuration
Counter register	IF counter register (IFC)
Control register	IF counter mode select register (IFCMD) IF counter control register (IFCR) IF counter gate judge register (IFCJG)

Figure 17-1. Block Diagram of Frequency Counter


Gate Time Control Block
2
FMIFC
AMIFC
Input Select Block
Start/Stop Control Block
IF Counter Register (IFCR) Block
2
IFC MD1
IFC MD0
IFC CK1
IFC CK0
IFC JG0
IFC ST
IFC RES
IF Counter Mode Select Register (IFCMD)
IF Counter Gate Judge Register (IFCJG)
IF Counter Control Register (IFCCR)
Internal Bus

(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 a count mode.

(2) Gate time control block

The gate time control block sets a gate time (count time).

(3) Start/stop control block

The start/stop control block starts counting by 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 to 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 at reset or in the STOP mode. In the HALT mode, it holds the current count value.

17.3 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 a mode and gate time (count time).
This register is set by using a 1-bit or 8-bit memory manipulation instruction.
The value of this register is reset to 00H at reset or in the STOP mode.
In the HALT mode, this register holds the value immediately before the HALT mode is set.

Figure 17-2. Format of IF Counter Mode Select Register (IFCMD)

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	At reset	R/W
IFCMD	0	0	0	0	IFCMD1	IFCMD0	IFCCK1	IFCCK0	FFA9H	00H	R/W

IFCMD1	IFCMD0	Selects Frequency Counter Pin and Mode
0	0	Disables FMIFC AMIFC pins[Note]
0	1	AMIFC pin, AMIF count mode
1	0	FMIFC pin, FMIF count mode
1	1	FMIFC pin, AMIF count mode

IFCCK1	IFCCK0	Selects Gate Time
0	0	1 ms
0	1	4 ms
1	0	8 ms
1	1	Open

Note The FMIFC and AMFIC pins are high-impedance state.

Remark Bits 4 through 7 are fixed to 0 by hardware.

(2) IF counter control register (IFCCR)

This register starts counting by the IF counter register and clears the IF counter register.

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

The value of this register is reset to 00H at reset and in the STOP mode.

In the HALT mode, this register holds the value immediately before the HALT mode is set.

Figure 17-3. Format of IF Counter Control Register (IFCCR)

Symbol	7	6	5	4	3	2	<1>	<0>	Address	At reset	R/W
IFCCR	0	0	0	0	0	0	IFCST	IFCRES	FFACH	00H	W

IFCST	Starts IF Counter Register
0	Nothing is affected
1	Starts counting

IFCRES	Clears Data of IF Counter Register
0	Nothing is affected
1	Clears data of IF counter register

Remark Bits 2 through 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 at reset and in the STOP mode.

In the HALT mode, this register holds the value immediately before the HALT mode is set.

Figure 17-4. Format of IF Counter Gate Judge Register (IFCJG)

Symbol	7	6	5	4	3	2	1	<0>	Address	At reset	R/W
IFCJG	0	0	0	0	0	0	0	IFCJG0	FFABH	00H	R

IFCJG0	Detects 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 through 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.

17.4 Operation of Frequency Counter

(1) Select an input pin, mode, and gate time by using the IF counter mode select register (IFCMD).
Figure 17-5 shows a block that selects an input pin and mode.

(2) Set bit 0 (IFCRES) of the IF counter control register (IFCCR) to 1, and clears 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 since a 1-kHz internal signal has risen after IFCST was set. If the gate time is set to be opened, 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 17-6 shows the gate timing of the frequency counter.

(5) While the gate opens the frequency input to the selected FMIFC or AMIFC pin, the IF counter register counts the frequency.
If the FMIFC pin is used in the FMIF count mode, however, the input frequency is divided by half before it is counted.

The relationship between count value x (decimal), input frequencies (f_{FMIFC} and f_{AMIFC}), and 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 17-5. Block Diagram of Input Pin and Mode Selection

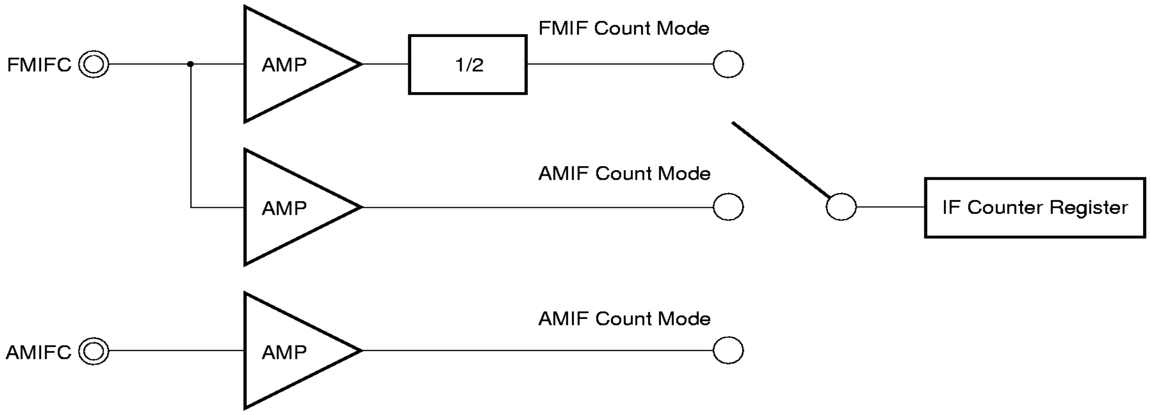
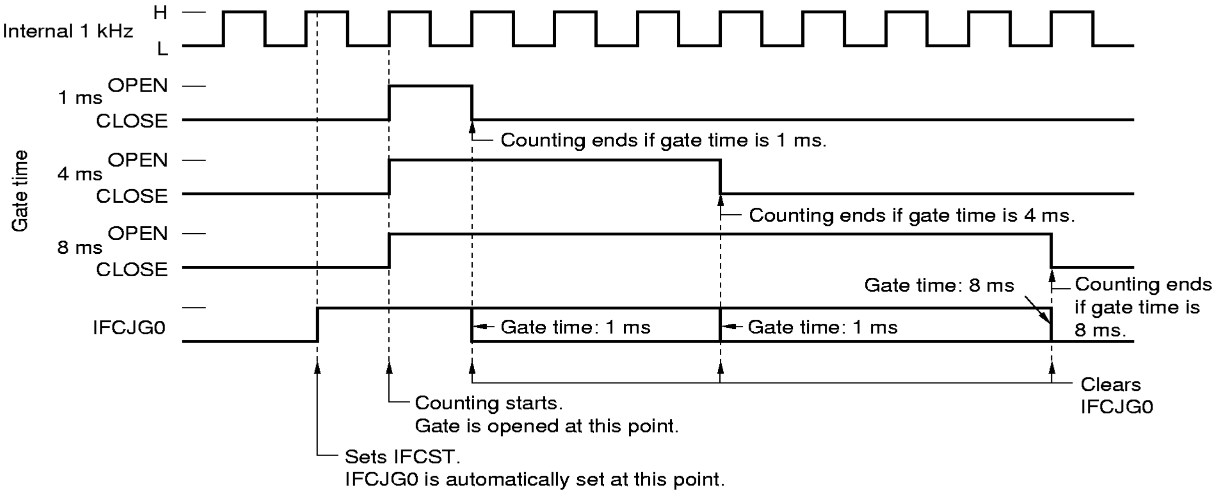
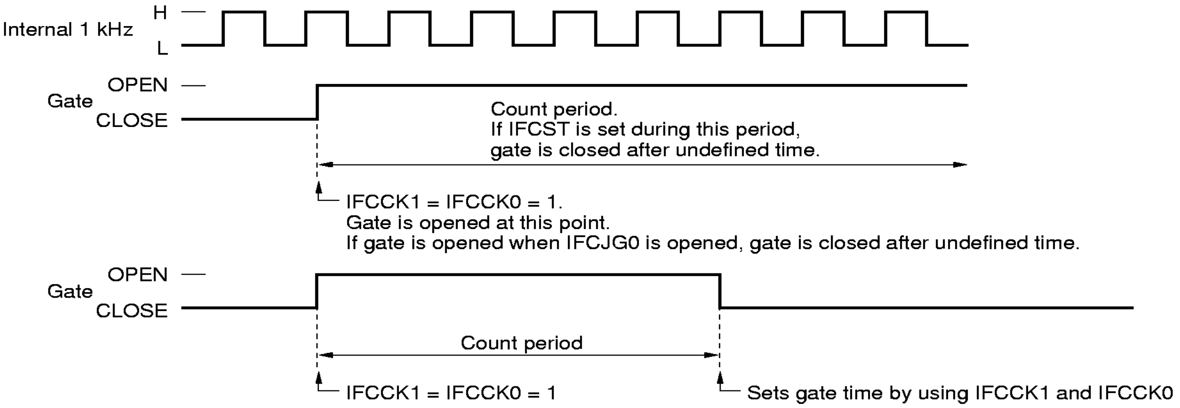


Figure 17-6. Gate Timing of Frequency Counter

(a) If gate time is set to 1, 4, or 8 ms


Internal 1 kHz
H
L
Gate time
1 ms
4 ms
8 ms
OPEN
CLOSE
Counting ends if gate time is 1 ms.
Counting ends if gate time is 4 ms.
Gate time: 8 ms
Counting ends if gate time is 8 ms.
IFCJG0
Gate time: 1 ms
Gate time: 1 ms
Clears IFCJG0
Counting starts.
Gate is opened at this point.
Sets IFCST.
IFCJG0 is automatically set at this point.

(b) If gate is set to be open


Internal 1 kHz
H
L
Gate
OPEN
CLOSE
Count period.
If IFCST is set during this period,
gate is closed after undefined time.
IFCCK1 = IFCCK0 = 1.
Gate is opened at this point.
If gate is opened when IFCJG0 is opened, gate is closed after undefined time.
Gate
OPEN
CLOSE
Count period
IFCCK1 = IFCCK0 = 1
Sets gate time by using IFCCK1 and IFCCK0

Caution If counting is started by using IFCST while this gate is open, the gate is closed after undefined time. To open the gate, therefore, do not set IFCST to 1.

Remark IFCST : Bit 1 of IF counter control register (IFCCR)

IFCJG0 : Bit 0 of IF counter gate judge register (IFJG)

IFCCK1, 0 : Bit 1 and 0 of IF counter mode select register (IFCMD)

17.5 Notes on Frequency Counter

(1) Notes on using frequency counter

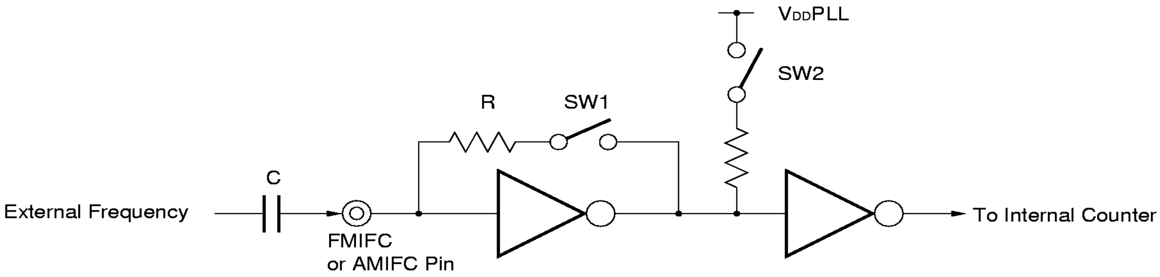
Because signals are input to the frequency counter from an input pin (FMIFC or AMIFC pin) with an AC amplifier as shown in Figure 17-7, cut the DC component of the input signals by using capacitor C.

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 17-7. Frequency Counter Input Pin Circuit



(2) Notes in HALT mode

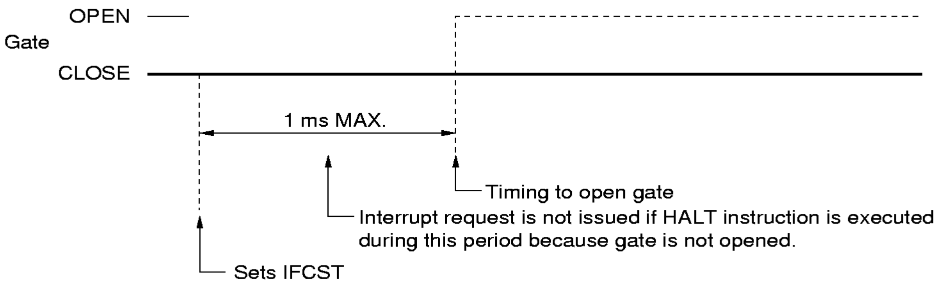
The FMIFC and AMIFC pins hold the status immediately before the HALT status was set.

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 17-8. Gate Status When HALT Instruction Is Executed


OPEN
Gate
CLOSE
1 ms MAX.
Timing to open gate
Interrupt request is not issued if HALT instruction is executed during this period because gate is not opened.
Sets IFCST

(3) Error of frequency counter

The error of the frequency counter includes an error of gate time and a count error.

(1) Error of gate time

The gate time of the frequency counter is created by dividing 4.5 MHz.
Therefore, if 4.5 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 18 STANDBY FUNCTION

18.1 Standby Function and Configuration

18.1.1 Standby function

The standby function is designed to decrease power consumption of the system. The following two modes are available.

(1) HALT mode

HALT instruction execution sets the HALT mode. The HALT mode is intended to stop the CPU operation clock. System clock oscillator continues oscillation. In this mode, current consumption cannot be decreased as in the STOP mode. The HALT mode is valid to restart immediately upon interrupt request 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. CPU current consumption can be considerably decreased.

Data memory low-voltage hold (down to V_{DD} = 2.3 V) is possible. Thus, the STOP mode is effective to hold 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 19 RESET FUNCTION**.

Because this mode can be cleared upon interrupt request, it enables intermittent operations to be carried out. However, because a wait time is necessary to secure an oscillation stabilization time after the STOP mode is cleared, select the HALT mode if it is necessary to start processing immediately upon interrupt request.

All the functions stop operating.

Some registers of the PLL frequency synthesizer and frequency counter are reset, but the other functions are stopped with their current status retained.

Cautions 1. When proceeding to the STOP mode, be sure to stop the peripheral hardware operation and execute 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.

18.1.2 Standby function control register

A wait time after the STOP mode is cleared upon interrupt request till the oscillation stabilizes is controlled with the oscillation stabilization time select register (OSTS).

OSTS is set with an 8-bit memory manipulation instruction.

Reset input sets OSTS to 04H.

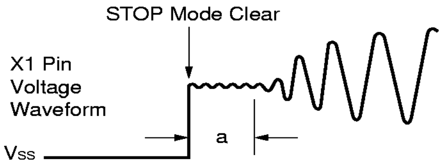
Figure 18-1. Format of Oscillation Stabilization Time Select Register (OSTS)

Symbol	7	6	5	4	3	2	1	0	Address	At 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 (910 μs)
0	0	1	2^{14}/fx (3.64 ms)
0	1	0	2^{15}/fx (7.28 ms)
0	1	1	2^{16}/fx (14.6 ms)
1	0	0	2^{17}/fx (29.1 ms)
Other than above			Setting prohibited

Remark fx : System clock oscillation frequency
( ) : fx = 4.5MHz

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.



18.2 Operations of Standby Function

18.2.1 HALT mode

(1) HALT mode set and operating status

The HALT mode is set by executing the HALT instruction.

The operating status in the HALT mode is described below.

Table 18-1. HALT Mode Operating Status

Item	Status
Clock generation circuit	Can oscillate system clock. Stops clock supply to CPU.
CPU	Stops operating.
Port	Holds status before HALT mode is set.
8-bit timer/event counter	Holds operation before HALT mode is set and can operate.
Basic timer	
Watchdog timer	
Buzzer output control circuit	
A/D converter	Retains operation performed when HALT mode is 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: Power-fail comparison operation can be performed.
Serial interface (IIC0, SIO3, UART0[Note])	Retains operation performed when HALT mode is set and can operate.
EEPROM	Hold operation before HALT mode is set and can operate.
External interrupt	
PLL frequency synthesizer	
Frequency counter	Retains operation performed before HALT mode is set. However, operation is not performed correctly though it is continued.
Power-ON clear circuit	Reset when voltage of less than 3.5 V is detected.

Note μPD178F124 only.

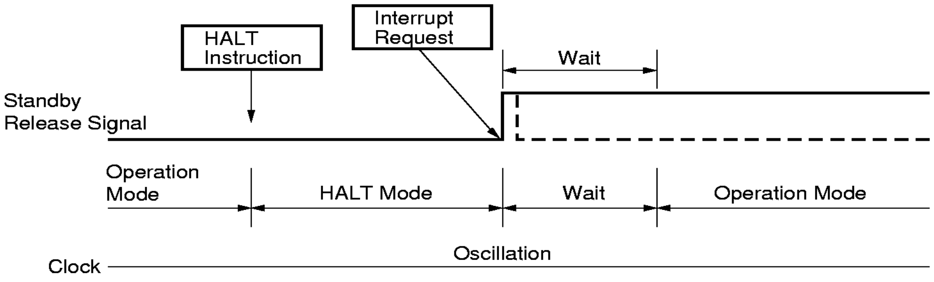
(2) HALT mode release

The HALT mode can be released with the following three types of sources.

(a) Release upon unmasked interrupt request

When an unmasked interrupt request is generated, the HALT mode is released. If interrupt acknowledge is enabled, vectored interrupt service is carried out. If disabled, the next address instruction is executed.

Figure 18-2. HALT Mode Release upon Interrupt Generation



Remarks 1. The broken line indicates the case when the interrupt request which has released the standby status is acknowledged.

2. Wait time will be as follows:

• When vectored interrupt service is carried out:	8 to 9 clocks
• When vectored interrupt service is not carried out:	2 to 3 clocks

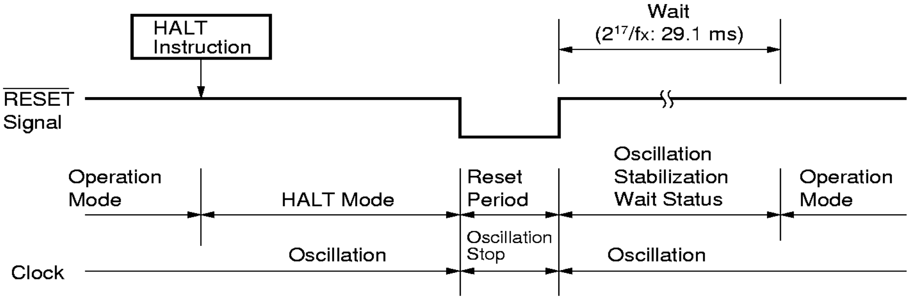
(b) Release upon non-maskable interrupt request

When a non-maskable interrupt is generated, the HALT mode is released and vectored interrupt service is carried out whether interrupt acknowledge 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 normal reset operation, a program is executed after branch to the reset vector address.

Figure 18-3. HALT Mode Release by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 4.5 MHz

Table 18-2. Operation after HALT Mode Release

Release Source	MKxx	PRxx	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt service execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt service execution
	1	×	×	×	HALT mode hold
Non-maskable interrupt request	–	–	×	×	Interrupt service execution
$\overline{\text{RESET}}$ input	–	–	×	×	Reset processing

Remark ×: Don't care

18.2.2 STOP mode

(1) STOP mode set 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 by GND, and the X2 pin is internally pulled up by V_{DD} to minimize the leakage current at the crystal oscillator block.

2. Because the interrupt request signal is used to clear 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. After the wait set using the oscillation stabilization time select register (OSTS), the operation mode is set.

The operating status in the STOP mode is described below.

Table 18-3. STOP Mode Operating Status

Item	Status
Clock generation circuit	Can oscillate system clock. Stops clock supply to CPU.
CPU	Stops operating.
Port	Holds status before HALT mode is set.
8-bit timer/event counter	Operation stops and cannot operate.
Basic timer	
Watchdog timer	
Buzzer output control circuit	
A/D converter	
Serial interface (IIC0, SIO3, UART0[Note])	
EEPROM	
External interrupt	Can operate.
PLL frequency synthesizer	Operation stops and cannot operate.
Frequency counter	
Power-ON clear circuit	$\overline{\text{RESET}}$ generated when detecting 2.3 V or less.

Note μPD178F124 only.

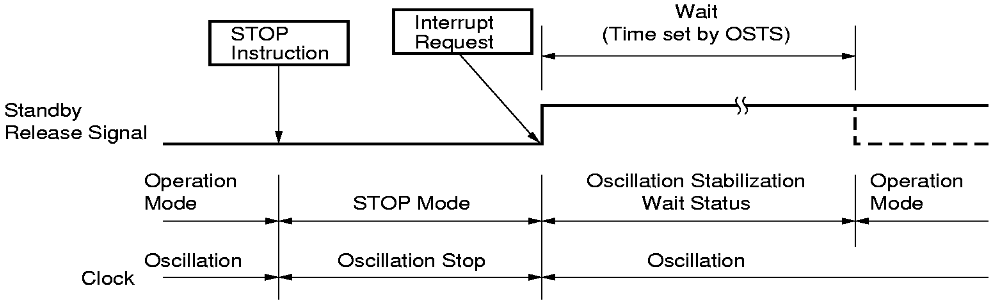
(2) STOP mode release

The STOP mode can be released with the following two types of sources.

(a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the STOP mode is released. If interrupt request acknowledge is enabled after the lapse of oscillation stabilization time, vectored interrupt service is carried out. If interrupt request acknowledge is disabled, the next address instruction is executed.

Figure 18-4. STOP Mode Release by Interrupt Request Generation

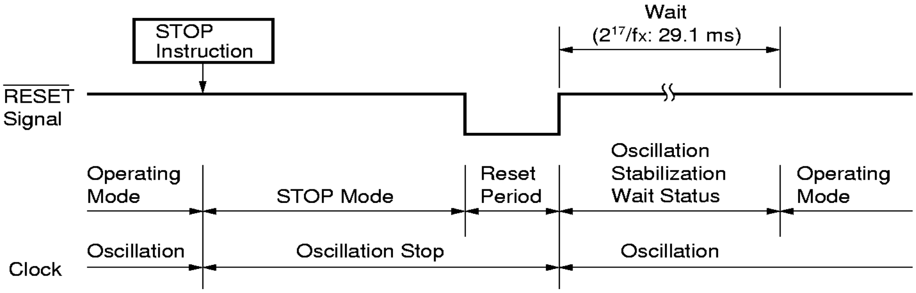


Remark The broken line indicates the case when the interrupt request which has released the standby status is acknowledged.

(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, reset operation is carried out.

Figure 18-5. Release by STOP Mode $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

2. ( ): fx = 4.5 MHz

Table 18-4. Operation after STOP Mode Release

Release Source	MKxx	PRxx	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Next address instruction execution
	0	0	1	×	Interrupt service execution
	0	1	0	1	Next address instruction execution
	0	1	×	0	
	0	1	1	1	Interrupt service execution
	1	×	×	×	STOP mode hold
$\overline{\text{RESET}}$ input	–	–	×	×	Reset processing

Remark ×: Don't care

CHAPTER 19 RESET FUNCTION

19.1 Reset Function

The following three operations are available to generate the reset signal.

(1) External reset input with $\overline{\text{RESET}}$ pin
(2) Internal reset by inadvertent program loop time detection 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 19-1. 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 P132 pins go low, VCOH and VCOL pin are pulled down).

The $\overline{\text{RESET}}$ signal is deasserted when a high level is input to the $\overline{\text{RESET}}$ pin, and the 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 19-1 when the watchdog timer overflow. 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 through P132 pins go low, VCOH and VCOL pin are pulled down).

Reset by the watchdog timer is cleared immediately after reset has been effected, and the 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 HALT mode)

When these reset conditions of power-ON clear are satisfied, reset is effected, and each hardware unit enters the status shown in Table 19-1. 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 through P132 pins go low, however).

Reset by power-ON clear is cleared if the supply voltage rises beyond a specific level, and the 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 for 10 μs or more to the $\overline{\text{RESET}}$ pin.

2. During reset input, system clock oscillation remains stopped.

3. When the STOP mode is cleared by $\overline{\text{RESET}}$ input, the STOP mode register contents are held during reset input. However, the I/O port pin becomes high-impedance. Output dedicated port pin (P130 to P132) becomes low level regardless of the previous status.

Figure 19-1. Reset Function Block Diagram

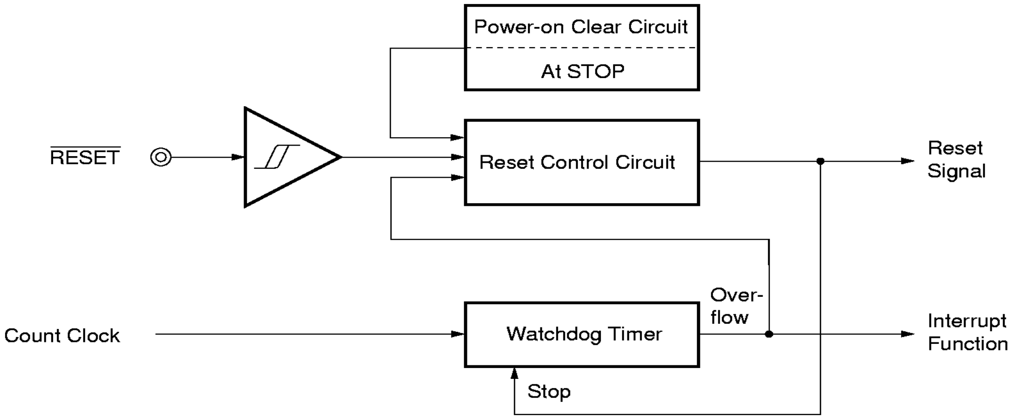
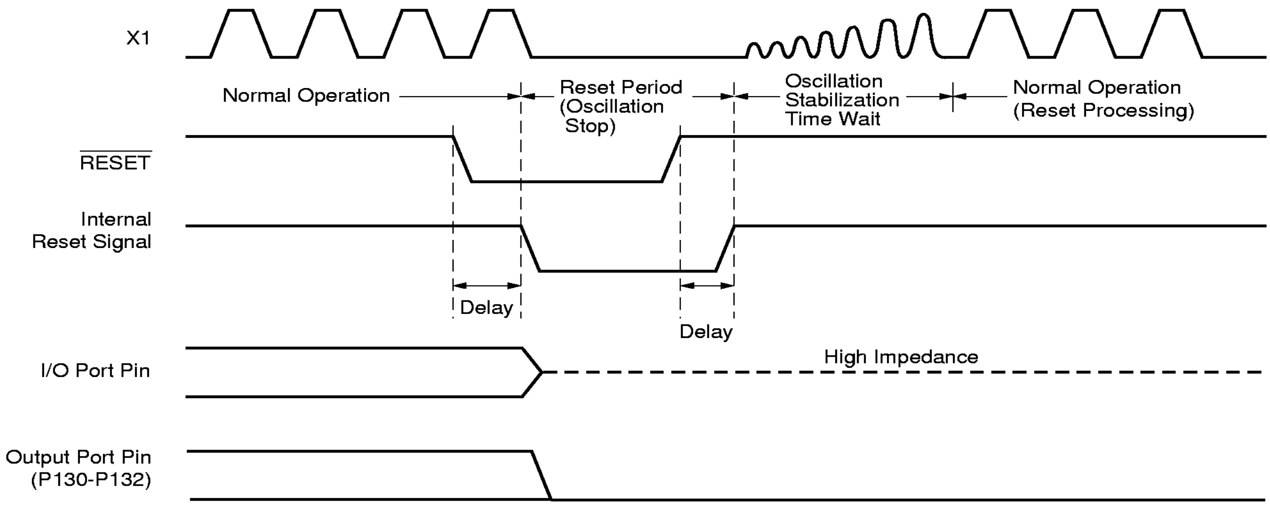


Figure 19-2. Timing of Reset by $\overline{\text{RESET}}$ Input

(a) In normal operating mode


X1
Normal Operation
Reset Period (Oscillation Stop)
Oscillation Stabilization Time Wait
Normal Operation (Reset Processing)
RESET
Internal Reset Signal
Delay
Delay
I/O Port Pin
High Impedance
Output Port Pin (P130-P132)

(b) In STOP mode

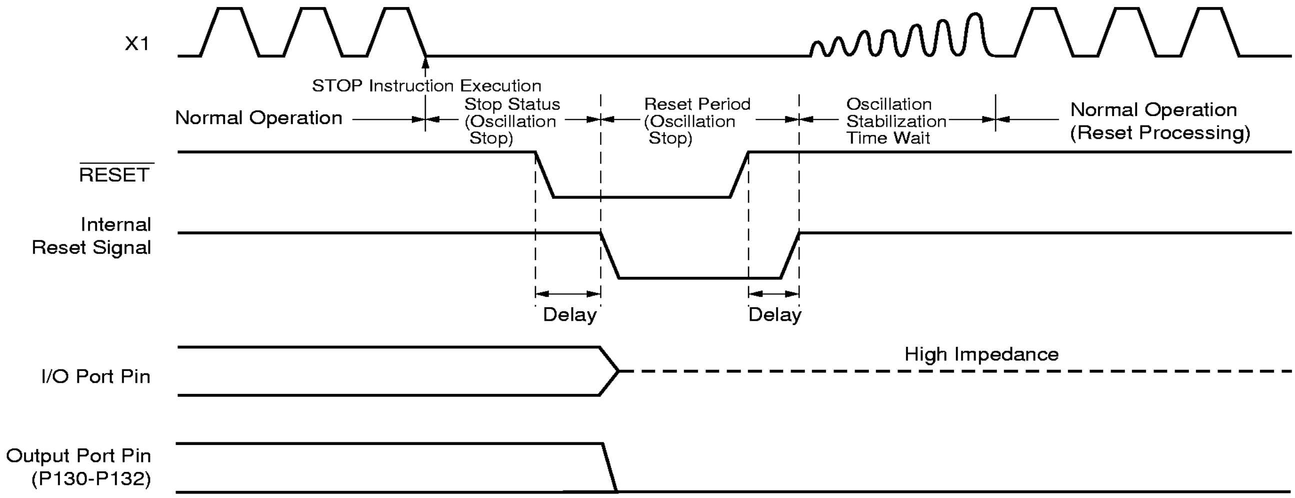

X1
STOP Instruction Execution
Normal Operation
Stop Status (Oscillation Stop)
Reset Period (Oscillation Stop)
Oscillation Stabilization Time Wait
Normal Operation (Reset Processing)
RESET
Internal Reset Signal
Delay
Delay
I/O Port Pin
High Impedance
Output Port Pin (P130-P132)

Figure 19-3. Timing of Reset due to Watchdog Timer Overflow

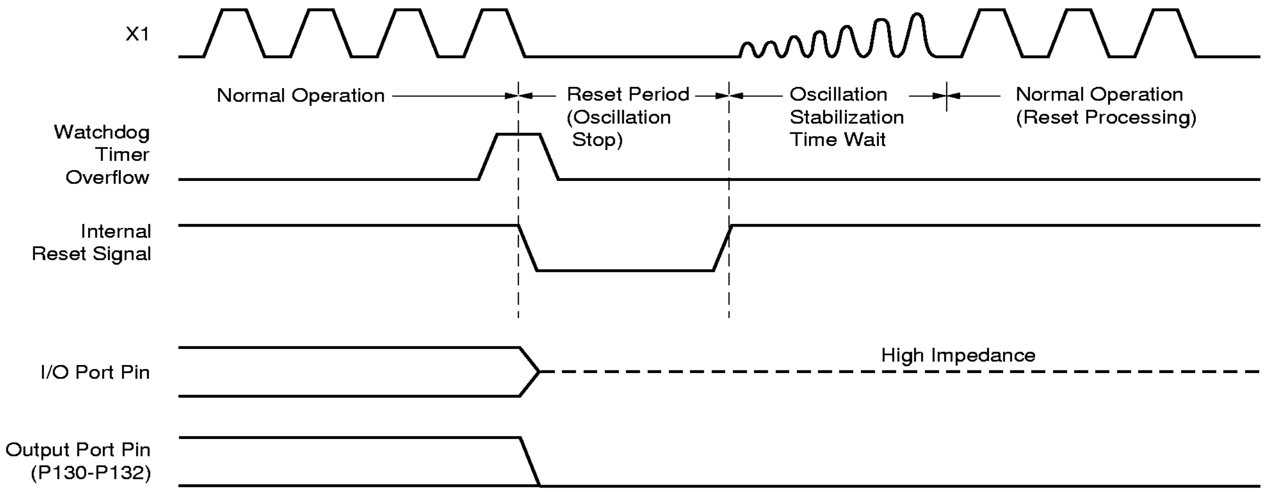
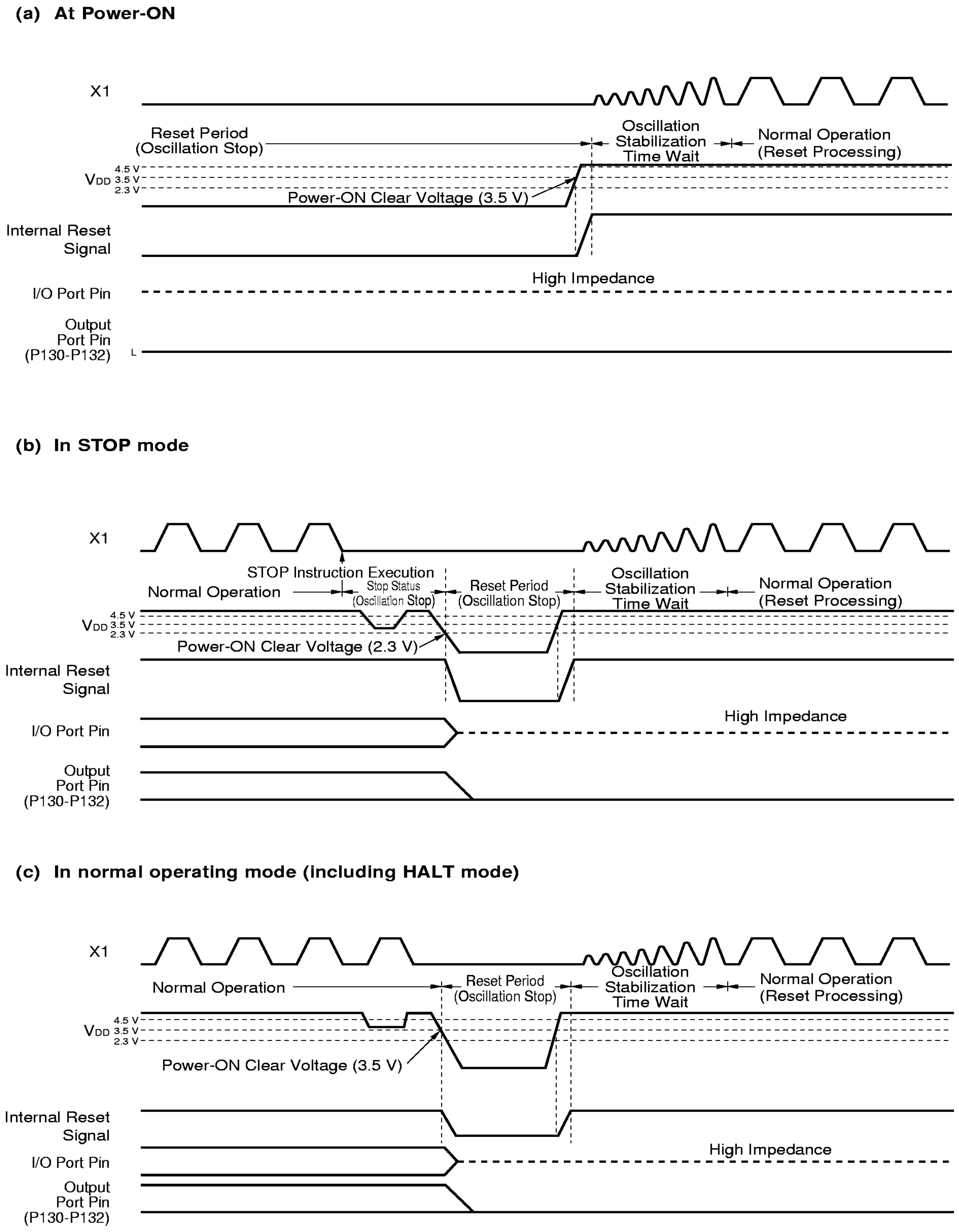

X1
Normal Operation
Reset Period (Oscillation Stop)
Oscillation Stabilization Time Wait
Normal Operation (Reset Processing)
Watchdog Timer Overflow
Internal Reset Signal
I/O Port Pin
High Impedance
Output Port Pin (P130-P132)

Figure 19-4. Timing of Reset by Power-ON Clear



(a) At Power-ON

(b) In STOP mode

(c) In normal operating mode (including HALT mode)

Table 19-1. Hardware Status after Reset (1/2)

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]
Port (output latch)	Ports 0, 1, 3-7, 12, 13 (P0, P1, P3-P7, P12, P13)	00H
Port mode registers (PM0, PM3-PM7, PM12)		FFH
Pull-up resistor option register 4 (PU4)		00H
Processor clock control register (PCC)		04H
Oscillation stabilization time select register (OSTS)		04H
DTS system clock select register (DTSCK)		00H[Note 3]
EEPROM write control register (EEWC)[Note 4]		00H
Memory size select register (IMS)		CFH[Note 5]
Internal extension RAM size select register (IXS)		0CH[Note 6]
8-bit timer/event counter	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
Watchdog timer	Clock select register (WDCS)	00H
	Mode register (WDTM)	00H
Buzzer output control circuit	BEEP0 clock select register 0 (BEEPCL0)	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 remains unchanged after reset.

2. The status before reset is retained even after reset in the standby mode.

3. Though the initial value is 00H, be sure to set it to 01H before using.

4. μPD178122, 178123, 178124, and 178F124 only.

5. The initial value is CFH. Set the following value to this register depending on the model:

μPD178022, 178122	: 44H
μPD178023, 178123	: C6H
μPD178024, 178124	: C8H
μPD178F124	: Value corresponding to mask ROM

6. Do not assign a value other than 0CH.

Table 19-1. Hardware Status after Reset (2/2)

Hardware			Status after Reset
Serial interface	IIC	IIC shift register 0 (IIC0)	Undefined
		Slave address register 0 (SVA0)	Undefined
		IIC transfer clock select register 0 (IICCL0)	00H
		IIC status register 0 (IICS0)	00H
		IIC control register 0 (IICC0)	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
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[Note 1])	00H
		Mask flag registers (MK0L, MK0H, and MK1L[Note 1])	FFH
		Priority specification flag registers (PR0L, PR0H, and PR1L[Note 1])	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)	Retained[Note 2]
		PLL data registers (PLLRH, PLLRL, and PLLR0)	Undefined
		PLL data transfer 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 register (IFCR)	0000H
Power-ON clear		POC status register (POCS)	Retained[Note 3]

Notes 1. μPD178F124 only

2. Undefined only at power-ON clear reset

3. 03H only at power-ON clear reset

19.2 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 this POCM after reset by power-ON clear has been cleared (after program execution has been started from address 0000H).

Figure 19-5. Format of POC Status Register (POCS)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
POCS	0	0	0	0	0	0	VM45	POCM	FF1BH	Retained[Note]	R&Reset

POCM	Detects 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 through power-ON clearing. 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 at reset that is effected by means of power-ON clear.

19.3 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.

Note, however, that this 4.5-V voltage detection function does not cause internal reset.

Figure 19-6. Format of POC Status Register (POCS)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
POCS	0	0	0	0	0	0	VM45	POCM	FF1BH	Retained[Note]	R&Reset

VM45	Detects Voltage Level of V_{DD} Pin
0	Does not detect if V_{DD} pin is less than 4.5 V (4.3±0.3 V)
1	Detects 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 and watchdog timer.

Remark The values of the special function registers other than POCS at reset are the same as the values at power-ON clear.

[MEMO]

CHAPTER 20 µPD178F124

The µPD178F124 is provided with a flash memory to/from which data can be rewritten/erased with the device mounted on a printed circuit board. The differences between the flash memory model (µPD178F124) and mask ROM models (µPD178022, 178023, 178024, 178122, 178123, and 178124) are shown in Table 20-1.

Table 20-1. Differences between µPD178F124 and Mask ROM Models

Item		µPD178F124	µPD178022, 178023, 178024	µPD178122, 178123, 178124
Internal memory	ROM structure	Flash memory	Mask ROM	
	ROM capacity	32K bytes	µPD178022: 16K bytes µPD178023: 24K bytes µPD178024: 32K bytes	µPD178122: 16K bytes µPD178123: 24K bytes µPD178124: 32K bytes
	EEPROM	128 bytes	None	128 bytes
Internal ROM capacity selected by memory size select register (IMS)		Equivalent to mask ROM model	µPD178022: 44H µPD178023: C6H µPD178024: C8H	µPD178122: 44H µPD178123: C6H µPD178124: C8H
Interrupt source		18	14	15
Serial interface		3 channels (IIC0, SIO3, UART0)	2 channels (IIC0, SIO3)	
IC pin		Not provided	Provided	
V_{PP} pin		Provided	Not provided	
Electrical specifications, recommended soldering conditions		These may differ between flash memory model and mask ROM model.		

20.1 Memory Size Select Register

The internal memory capacity of the μPD178F124 can be changed using the memory size select register (IMS). By using this register, the memory of the μPD178F124 can be mapped in the same manner as a mask ROM model with a different internal memory capacity.

Use an 8-bit memory manipulation instruction to set the IMS.

This register is set to CFH at reset.

Be sure to set 44H, C6H, or C8H to the IMS.

Figure 20-1. Format of Memory Size Select Register (IMS)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
IMS	RAM2	RAM1	RAM0	0	ROM3	ROM2	ROM1	ROM0	FFF0H	CFH	R/W

RAM2	RAM1	RAM0	Selects Internal High-speed RAM Capacity
0	1	0	512 bytes
1	1	0	1024 bytes
Others			Setting prohibited

RAM3	RAM2	RAM1	RAM0	Selects Internal ROM Capacity
0	1	0	0	16K bytes
0	1	1	0	24K bytes
1	0	0	0	32K bytes
Others				Setting prohibited

Table 20-2 shows the setting of IMS to perform the same memory mapping as that of a mask ROM model.

Table 20-2. Set Value of Memory Size Select Register

Targeted Model	Set Value of IMS
μPD178022, 178122	44H
μPD178023, 178123	C6H
μPD178024, 178124	C8H

20.2 Internal Extension RAM Size Select Register

The internal extention RAM capacity of the μPD178F124 can be changed using the internal extention RAM size select register (IXS). By using this register, the memory of the μPD178F124 can be mapped in the same manner as a mask ROM model with a different internal extention RAM capacity.

Use an 8-bit memory manipulation instruction to set the IXS.

This register is set to 0CH at reset.

Caution Do not assign a value other than that the value at reset.

Figure 20-2. Format of Internal Extension RAM Size Select Register (IXS)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
IXS	0	0	0	IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	FFF4H	0CH	R/W

IXRAM4	IXRAM3	IXRAM2	IXRAM1	IXRAM0	Selects Internal Extension RAM Capacity
0	1	1	0	0	0 byte
Others					Setting prohibited

Table 20-3 shows the setting of IXS to perform the same memory mapping as that of a mask ROM model.

Table 20-3. Set Value of Internal Extension RAM Size Select Register

Targeted Model	Set Value of IXS
μPD178022, 178023, 178024, 178122, 178123, 178124	0CH

20.3 Flash Memory Programming

The program memory provided in the μPD178F124 is flash memory.

The flash memory can be written on-board, i.e., with the μPD178F124 mounted on the target system. To do so, connect a dedicated flash programmer (Flashpro III (Part number: FL-PR3, PG-FP3)) to the host machine and target system.

Remark FL-PR3 and PG-FP3 are products of Naito Densei Machida Mfg. Co., Ltd.

20.3.1 Selecting communication mode

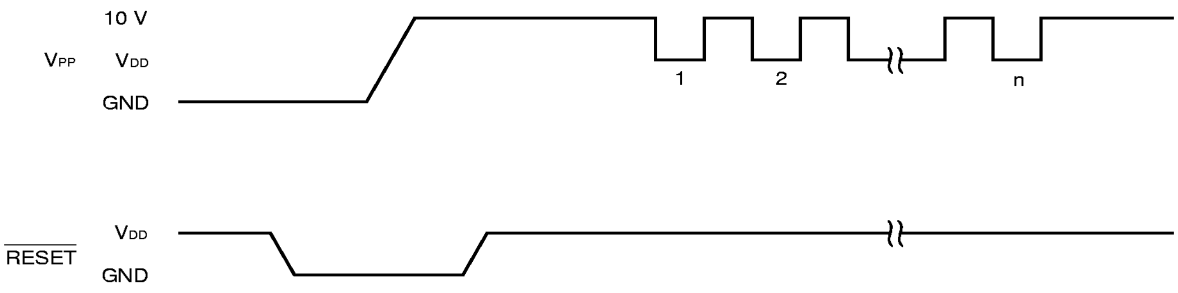
The flash memory is written by using Flashpro III and by means of serial communication. Select a communication mode from those listed in Table 20-4. To select a communication mode, the format shown in Figure 20-3 is used. Each communication mode is selected depending on the number of V_{PP} pulses shown in Table 20-4.

Table 20-4. Communication Modes

Communication Mode	Pins Used	Number of V_{PP} Pulses
3-wire serial I/O (SIO3)	SI3/P70 SO3/P71 $\overline{\text{SCK3}}$/P72	0
IIC0	SDA0/P76 SCL0/P77	4
UART0	$\overline{\text{RXD0}}$/P74 $\overline{\text{TXD0}}$/P75	8

Caution Be sure to select a communication mode by the number of V_{PP} pulses shown in Table 20-4.

Figure 20-3. Format of Communication Mode Selection



20.3.2 Flash memory programming function

An operation such as writing the flash memory is performed when a command or data is transmitted/received in the selected communication mode. The major flash memory programming functions are listed in Table 20-5.

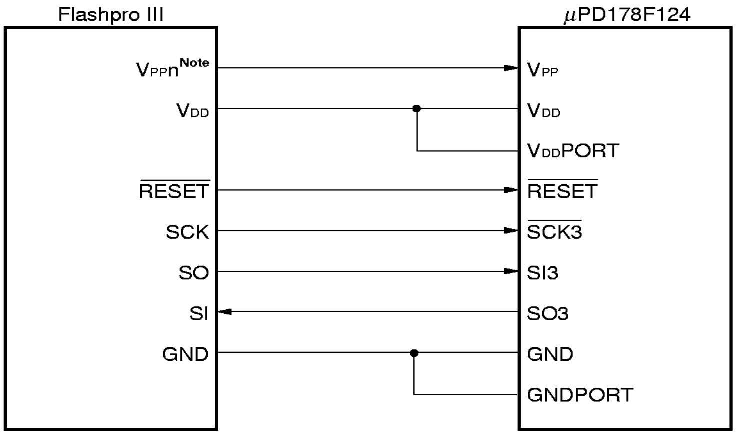
Table 20-5. Functions of Major Flash Memory Programming

Function	Description
Batch erase	Erases all memory contents.
Batch blank check	Checks erased status of entire memory.
Data write	Writes data to flash memory starting from write start address and based on number of data (bytes) to be written).
Batch verify	Compares all contents of memory with input data.

20.3.3 Connecting Flashpro III

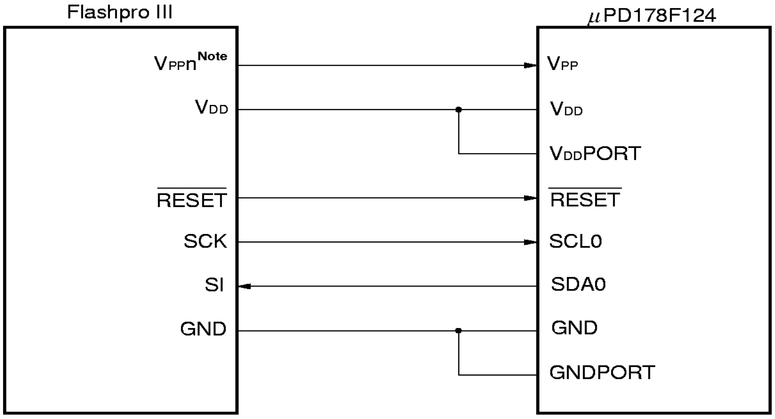
Connection with Flashpro III differs depending on the communication mode (3-wire serial I/O, IIC0, or UART0). Figures 20-4 through 20-6 show the connection in the respective modes.

Figure 20-4. Connection of Flashpro III in 3-Wire Serial I/O Mode



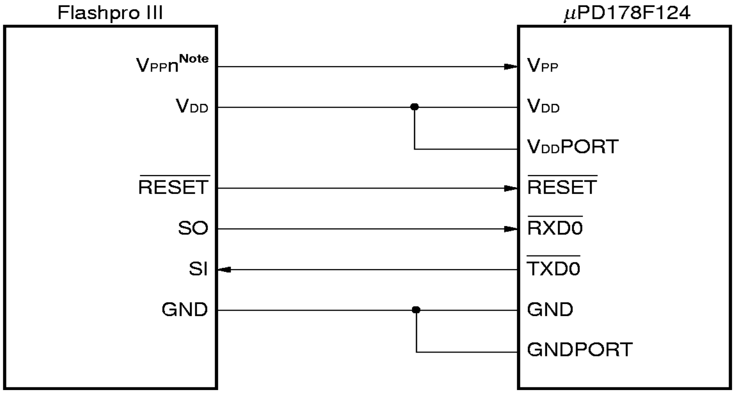
Note n = 1, 2

Figure 20-5. Connection of Flashpro III in IIC0 Mode



Note n = 1, 2

Figure 20-6. Connection of Flashpro III in UART0 Mode



Note n = 1, 2

20.3.4 Example settings for Flashpro III (PG-FP3)

When writing data to flash memory using Flashpro III (PG-FP3), use the following settings:

<1> Load parameter file.
<2> Select the serial mode and serial cock using the type command.
<3> An example of the settings for PG-FP3 is shown in Table 20-6.

Table 20-6. Example settings for Flashpro III (PG-FP3)

Communication Mode	Setting of Flashpro III		Number of V$_{PP}$ pulses[Note 1]
3-wire serial I/O (SIO3)	COMM PORT	SIO ch-0	0
	CPU CLK	On Target Board	
		In Flashpro	
	On Target Board	4.1943 MHz	
	SIO CLK	1.0 MHz	
	In Flashpro	4.0 MHz	
	SIO CLK	1.0 MHz	
IIC0	COMM PORT	IIC ch-0	4
	SLAVE ADDRESS	08H to 77H	
	CPU CLK	On Target Board	
		In Flashpro	
	IIC CLK	1 to 400 kHz	
UART0	COMM PORT	UART ch-0	8
	CPU CLK	On Target Board	
	On Target Board	4.1943 MHz	
	UART BPS	9600 bps[Note 2]	

Notes 1. Number of V$_{PP}$ pulse supplied by Flashpro III (PG-FP3) when serial mode is initialized. This determines the pin used for the communication.

2. Select from 9600 bps, 19200 bps, 38400 bps, and 76800 bps.

Remark		
	COMM PORT	: Selection of serial port
	SIO CLK	: Selection of serial clock frequency
	CPU CLK	: Selection of CPU clock source to be input

[MEMO]

CHAPTER 21 INSTRUCTION SET

This chapter describes each instruction set of the μPD178024 and 178124 subseries as list table. For details of its operation and operation code, refer to the separate document **78K/0 series User's Manual—Instruction (U12326E)**.

21.1 Conventions

21.1.1 Operand symbols and description

Operands are written in "Operand" column of each instruction in accordance with the description of the instruction operand symbols (refer to the assembler specifications for detail). When there are two or more descriptions, select one of them. Alphabetic letters in capitals and symbols, #, !, $ and [ ] are key words 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 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 21-1. 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-FF1FH Immediate data or labels
saddrp	FE20H-FF1FH Immediate data or labels (even address only)
addr16	0000H-FFFFH Immediate data or labels (Only even addresses for 16-bit data transfer instructions)
addr11	0800H-0FFFH Immediate data or labels
addr5	0040H-007FH Immediate data or labels (even address 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**.

21.1.2 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
$\times_H$, $\times_L$	High-order 8 bits and low-order 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)

21.1.3 Description of “flag operation” column

(Blank)	Nt affected
0	Cleared to 0
1	Set to 1
×	Set/cleared according to the result
R	Previously saved value is restored

21.2 Operation List

Instruction Group	Mnemonic	Operands	Byte	Clock		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 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 internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		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 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 internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		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 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 internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		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 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 internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		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 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 internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		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 ← $\overline{\text{CY}}$			×

Notes **1.** When the internal high-speed RAM area is accessed or 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 internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		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 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 internal ROM program.

Instruction Group	Mnemonic	Operands	Byte	Clock		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 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 internal ROM program.

21.3 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

2nd Operand 1st 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

2nd Operand 1st 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

2nd Operand 1st 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

2nd Operand 1st 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

[MEMO]

APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for the development of systems which employ the μPD178024 and 178124 subseries. Figure A-1 shows the configuration example of the tools.

- **Support for PC98-NX series**
  Unless otherwise specified, products compatible with IBM PC/AT^{TM} computers are compatible with PC98-NX series computers. When using PC98-NX series computers, refer to the description for IBM PC/AT computers.

- **Windows**
  Unless otherwise specified, "Windows" means the following OSs.
  - Windows 3.1
  - Windows95
  - WindowsNT^{TM} Ver 4.0

Figure A-1. Configuration of Development Tool (1/2)

(1) When using the in-circuit emulator IE-78K0-NS

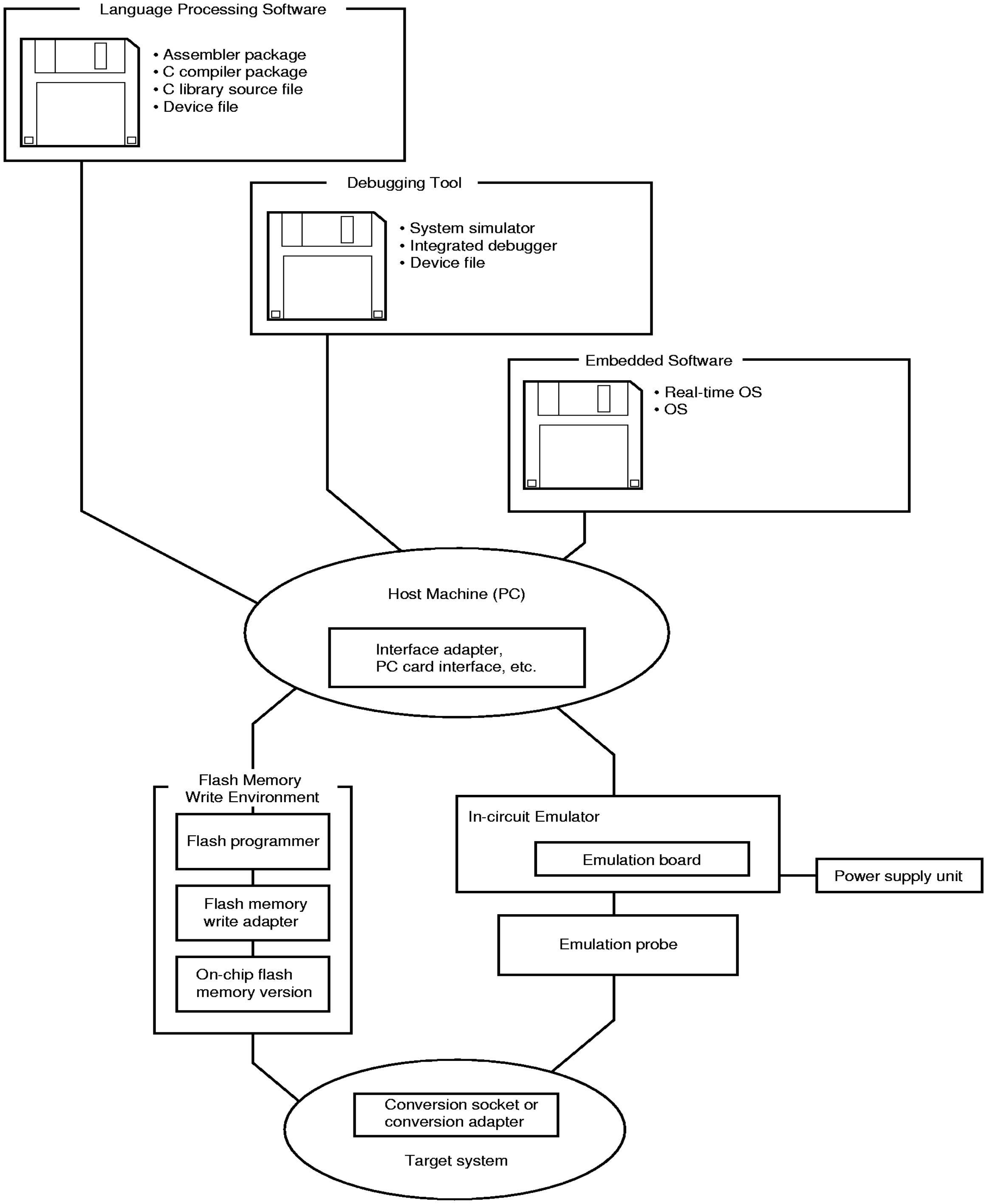
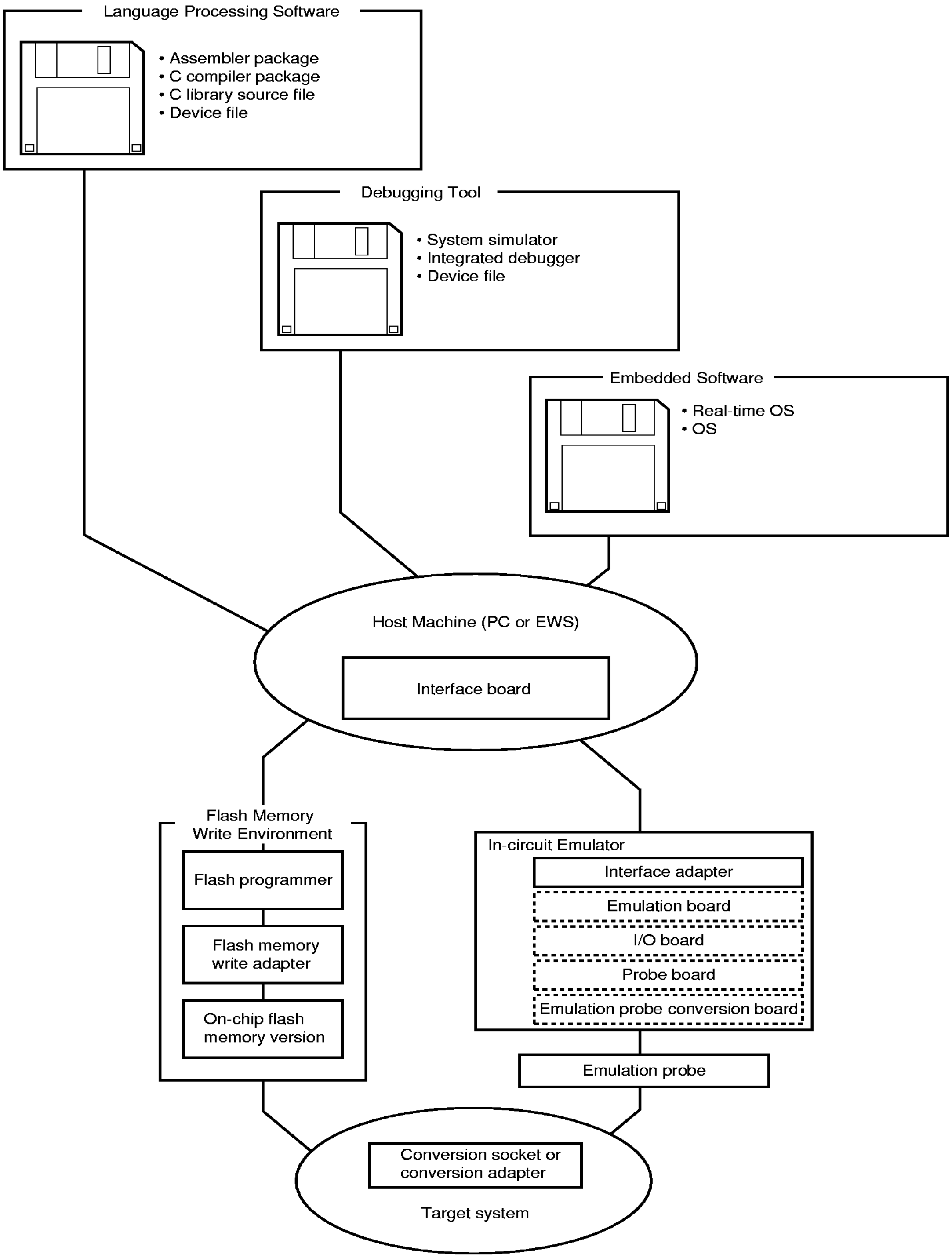


Figure A-1. Configuration of Development Tool (2/2)

(2) When using the in-circuit emulator IE-78001-R-A



Remark Items in broken line boxes differ according to the development environment. Refer to **A.3.1. Hardware**.

A.1 Language Processing Software

RA78K/0 Assembler Package	This assembler converts programs written in mnemonics into an 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 an optical device file (DF178124). **<Precaution when using RA78K/0 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) on Windows.
	Part Number: μSxxxxRA78K0
CC78K/0 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 optical assembler package and device file. **<Precaution when using CC78K/0 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) on Windows.
	Part Number: μSxxxxCC78K0
DF178124[Note] Device File	This file contains information peculiar to the device. This device file should be used in combination with an optical tool (RA78K/0, CC78K/0, SM78K0, ID78K0-NS, and ID78K0). Corresponding OS and host machine differ depending on the tool used.
	Part Number: μSxxxxDF178124
CC78K/0-L 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: μSxxxxCC78K0-L

Note The DF178124 can be used in common with the RA78K/0, CC78K/0, SM78K0, ID78K0-NS, and ID78K0.

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××RA78K0
μS××××CC78K0
μS××××DF178124
μS××××CC78K0-L

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 series	Windows (Japanese version) $^{\text{Note}}$	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version) $^{\text{Note}}$	3.5-inch 2HC FD
BB13		Windows (English version) $^{\text{Note}}$	
3P16	HP9000 series 700$^{\text{TM}}$	HP-UX$^{\text{TM}}$ (Rel. 10.10)	DAT (DDS)
3K13	SPARCstation$^{\text{TM}}$	SunOS$^{\text{TM}}$ (Rel. 4.1.4), Solaris$^{\text{TM}}$ (Rel. 2.5.1)	3.5-inch 2HC FD
3K15			1/4-inch CGMT
3R13	NEWS$^{\text{TM}}$ (RISC)	NEWS-OS$^{\text{TM}}$ (Rel. 6.1)	3.5-inch 2HC FD

Note Can be operated in DOS environment.

A.2 Flash Memory Writing Tools

Flashpro III (Part number: FL-PR3, PG-FP3) Flash Programmer	Flash programmer dedicated to microcontrollers with on-chip flash memory.
FA-80GF FA-80GC Flash Memory Writing Adapter	Flash memory writing adapter used connected to the Flashpro III. • FA-80GF : 80-pin plastic QFP (GF-3B9 type) • FA-80GC : 80-pin plastic QFP (GC-8BT type)
Flashpro III controller	Control program that runs on a PC. This is supplied with Flashpro III.

Remark Flashpro III, FA-80GF, and FA-80GC are products of Naito Densei Machida Mfg. Co., Ltd.
Phone: (044) 822-3813 Naito Densei Machida Mfg. Co., Ltd.

A.3 Debugging Tools

A.3.1 Hardware (1/2)

(1) When using the in-circuit emulator IE-78K0-NS

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 corresponds to integrated debugger (ID78K0-NS). This emulator should be used in combination with power supply unit, emulation probe, and interface adapter which is required to connect this emulator to the host machine.
IE-70000-MC-PS-B Power Supply Unit		This adapter is used for supplying power from a receptacle of 100-V to 240-V AC.
IE-70000-98-IF-C Interface Adapter		This adapter is required when using the 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 the 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 the IBM PC compatible computers as the IE-78K0-NS host machine (ISA bus compatible).
IE-70000-PCI-IF Interface Adapter		This adapter is required when using a PC with a PCI bus as the IE-78K0-NS host machine.
IE-178134-NS-EM4^Note^ 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-80GF Emulation Probe		This probe is used to connect the in-circuit emulator to the target system and is designed for 80-pin plastic QFP (GF-3B9 type).
	EV-9200G-80 Conversion Socket (Refer to **Figures A-2, A-3**)	This conversion socket connects the NP-80GF to the target system board designed to mount an 80-pin plastic QFP (GF-3B9 type).
NP-80GC Emulation Probe		This probe is used to connect the in-circuit emulator to the target system and is designed for 80-pin plastic QFP (GC-8BT type).
	EV-9200GC-80 Conversion Socket (Refer to **Figures A-4, A-5**)	This conversion socket connects the NP-80GC to the target system board designed to mount an 80-pin plastic QFP (GC-8BT type).

Note Under development

Remarks 1. NP-80GF and NP-80GC are products of Naito Densei Machida Mfg. Co., Ltd.
Phone: (044) 822-3813 Naito Densei Machida Mfg. Co., Ltd.

2. EV-9200G-80 and EV-9200GC-80 are sold in five units.

A.3.1 Hardware (2/2)

(2) When using the in-circuit emulator IE-78001-R-A

IE-78001-R-A In-circuit Emulator		The in-circuit emulator serves to debug hardware and software when developing application systems using a 78K/0 Series product. It corresponds to integrated debugger (ID78K0). This emulator should be used in combination with emulation probe and interface adapter, which is required to connect this emulator to the host machine.
IE-70000-98-IF-C Interface Adapter		This adapter is required when using the PC-9800 series computer (except notebook type) as the IE-78001-R-A host machine (C bus compatible).
IE-70000-PC-IF-C Interface Adapter		This adapter is required when using the IBM PC/AT compatible computers as the IE-78001-R-A host machine (ISA bus compatible).
IE-70000-PCI-IF Interface Adapter		This adapter is required when using a PC with a PCI bus as the IE-78001-R-A host machine.
IE-78000-R-SV3 Interface Adapter		This is adapter and cable required when using an EWS computer as the IE-78001-R-A host machine, and is used connected to the board in the IE-78000-R-A. As Ethernet^{TM}, 10Base-5 is supported. With the other method, a commercially available conversion adapter is necessary.
IE-178134-NS-EM4[Note] 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 and emulation conversion board.
	IE-78K0-NS-P01 I/O Board	The board on which FPGA is mounted.
	IE-78K0-R-EX1 Emulation Probe Conversion Board	This board is required when using the IE-178134-NS-EM4 on the IE-78001-R-A.
EP-78130GF-R Emulation Probe		This probe is used to connect the in-circuit emulator to the target system and is designed for 80-pin plastic QFP (GF-3B9 type).
	EV-9200G-80 Conversion Socket (Refer to **Figures A-2, A-3**)	This conversion socket connects the EP-78130GF-R to the target system board designed to mount an 80-pin plastic QFP (GF-3B9 type).
EP-78230GC-R Emulation Probe		This probe is used to connect the in-circuit emulator to the target system and is designed for 80-pin plastic QFP (GC-8BT type).
	EV-9200GC-80 Conversion Socket (Refer to **Figures A-4, A-5**)	This conversion socket connects the EP-78230GC-R to the target system board designed to mount an 80-pin plastic QFP (GC-8BT type).

Note Under development

Remark EV-9200G-80 and EV-9200GC-80 are sold in five units.

A.3.2 Software (1/2)

SM78K0 System Simulator	This system simulator is used to perform debugging at C source level or assembler level while simulating the operation of the target system on a host machine. This simulator runs on Windows. Use of the SM78K0 allows the execution of application logical testing and performance testing on an independent basis from hardware development without having to use an in-circuit emulator, thereby providing higher development efficiency and software quality. The SM78K0 should be used in combination with the optical device file (DF178124).
	Part Number: μS××××SM78K0

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××SM78K0

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version)	3.5-inch 2HC FD
BB13		Windows (English version)	

A.3.2 Software (2/2)

ID78K0-NS Integrated Debugger (supporting in-circuit emulator IE-78K0-NS)	This debugger is a control program to debug 78K/0 Series microcontrollers. It adopts a graphical user interface, which is equivalent visually and operationally to Windows or OSF/Motif™. It also has an enhanced debugging function for C language programs, and thus trace results can be displayed on screen in C-language level by using the windows integration function which links a trace result with its source program, disassembled display, and memory display. In addition, by incorporating function modules such as task debugger and system performance analyzer, the efficiency of debugging programs, which run on real-time OSs can be improved. It should be used in combination with the optional device file.
ID78K0 Integrated Debugger (supporting in-circuit emulator IE-78001-R-A)	
	Part Number: μSxxxxID78K0-NS, μSxxxxID78K0

Remark xxxx in the part number differs depending on the host machine and OS used.

μSxxxxID78K0-NS

xxxx	Host Machine	OS	Supply Medium
AA13	PC-9800 series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version)	3.5-inch 2HC FD
BB13		Windows (English version)	

μSxxxxID78K0

xxxx	Host Machine	OS	Supply Medium
AA13	PC-9800 series	Windows (Japanese version)	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version)	3.5-inch 2HC FD
BB13		Windows (English version)	
3P16	HP9000 series 700	HP-UX (Rel. 10.10)	DAT (DDS)
3K13	SPARCstation	SunOS (Rel. 4.1.4)	3.5-inch 2HC FD
3K15		Solaris (Rel. 2.5.1)	1/4-inch CGMT
3R13	NEWS (RISC)	NEWS-OS (Rel. 6.1)	3.5-inch 2HC FD

A.4 System Upgrade from Former In-circuit Emulator for 78K/0 Series to IE-78001-R-A

If you already have a former in-circuit emulator for 78K/0 Series microcontrollers (IE-78000-R or IE-78000-R-A), that in-circuit emulator can operate as an equivalent to the IE-78001-R-A by replacing its internal break board with the IE-78001-R-BK (under development).

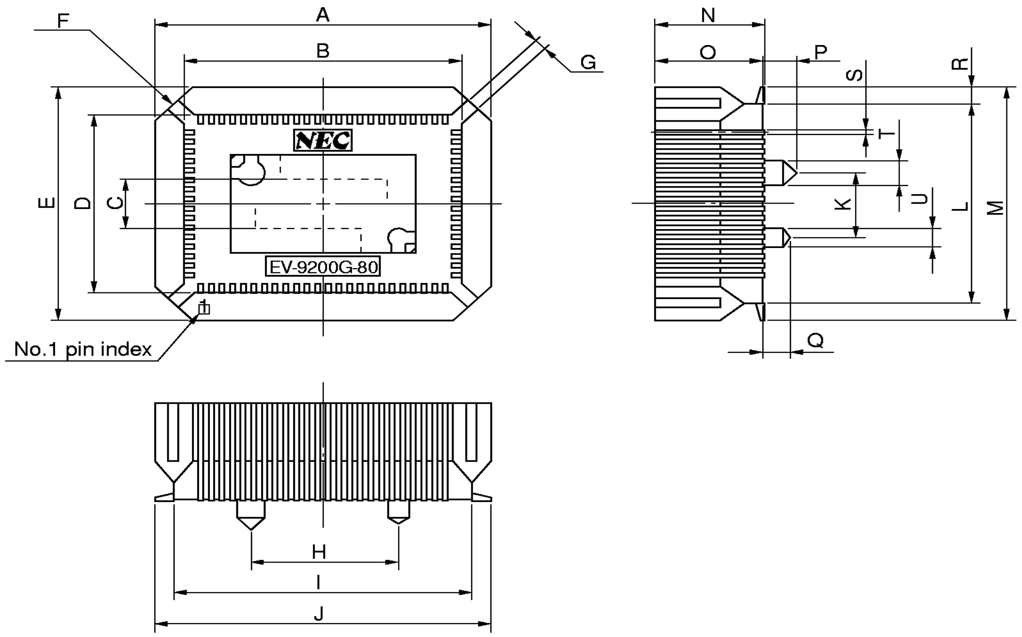
Table A-1. System Upgrade Method from Former In-circuit Emulator for 78K/0 Series to the IE-78001-R-A

In-circuit Emulator Owned	In-circuit Emulator Cabinet System Upgrade[Note]	Board To Be Purchased
IE-78000-R	Required	IE-78001-R-BK
IE-78000-R-A	Not required	

Note For upgrading a cabinet, send your in-circuit emulator to NEC.

Drawing for Conversion Socket (EV-9200G-80) Package and Recommended Board Mounting Pattern

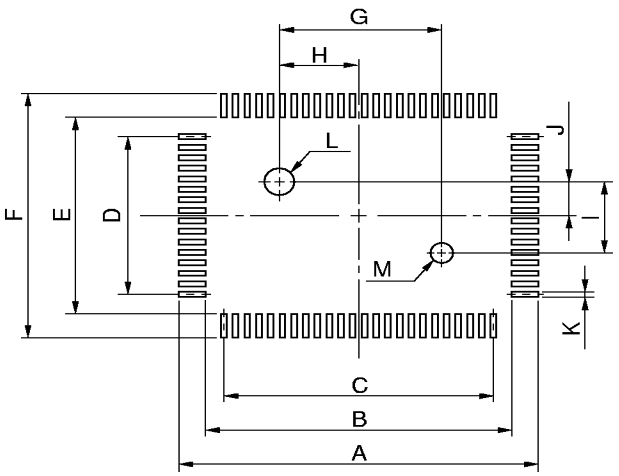
Figure A-2. EV-9200G-80 Package Drawing (for reference only)



EV-9200G-80-G0E

ITEM	MILLIMETERS	INCHES
A	25.0	0.984
B	20.30	0.799
C	4.0	0.157
D	14.45	0.569
E	19.0	0.748
F	4-C 2.8	4-C 0.11
G	0.8	0.031
H	11.0	0.433
I	22.0	0.866
J	24.7	0.972
K	5.0	0.197
L	16.2	0.638
M	18.9	0.744
N	8.0	0.315
O	7.8	0.307
P	2.5	0.098
Q	2.0	0.079
R	1.35	0.053
S	0.35±0.1	$0.014^{+0.004}_{-0.005}$
T	ϕ2.3	ϕ0.091
U	ϕ1.5	ϕ0.059

Figure A-3. EV-9200G-80 Recommended Board Mounting Pattern (for reference only)



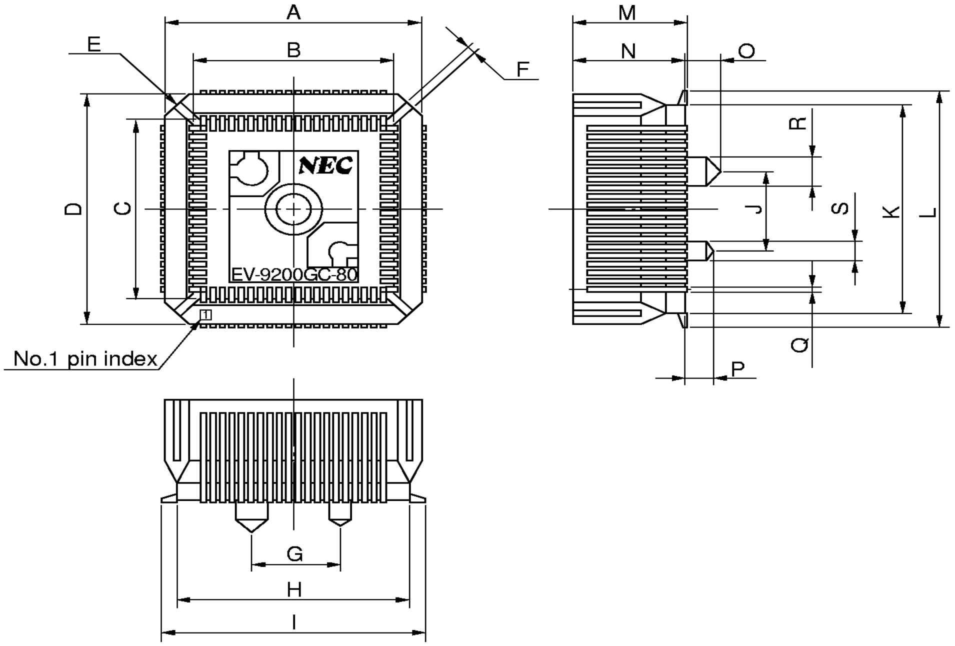
EV-9200G-80-P1E

ITEM	MILLIMETERS	INCHES
A	25.7	1.012
B	21.0	0.827
C	0.8±0.02 × 23=18.4±0.05	$0.031^{+0.002}_{-0.001}$ × 0.906=$0.724^{+0.003}_{-0.002}$
D	0.8±0.02 × 15=12.0±0.05	$0.031^{+0.002}_{-0.001}$ × 0.591=$0.472^{+0.003}_{-0.002}$
E	15.2	0.598
F	19.9	0.783
G	11.00±0.08	$0.433^{+0.004}_{-0.003}$
H	5.50±0.03	$0.217^{+0.001}_{-0.002}$
I	5.00±0.08	$0.197^{+0.003}_{-0.004}$
J	2.50±0.03	$0.098^{+0.002}_{-0.001}$
K	0.5±0.02	$0.02^{+0.001}_{-0.002}$
L	ϕ2.36±0.03	$\phi 0.093^{+0.001}_{-0.002}$
M	ϕ1.57±0.03	$\phi 0.062^{+0.001}_{-0.002}$

Caution Dimensions of mount pad for EV-9200 and that for target device (QFP) may be different in some parts. For the recommended mount pad dimensions for QFP, refer to "SEMICONDUCTOR DEVICE MOUNTING TECHNOLOGY MANUAL" (C10535E).

Drawing for Conversion Socket (EV-9200GC-80) Package and Recommended Board Mounting Pattern

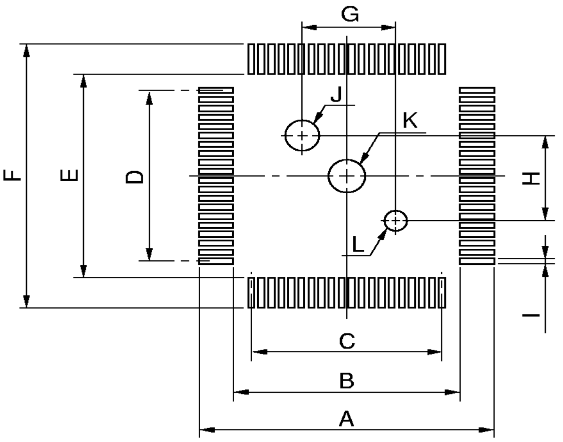
Figure A-4. EV-9200GC-80 Package Drawing (for reference only)



EV-9200GC-80-G1E

ITEM	MILLIMETERS	INCHES
A	18.0	0.709
B	14.4	0.567
C	14.4	0.567
D	18.0	0.709
E	4-C 2.0	4-C 0.079
F	0.8	0.031
G	6.0	0.236
H	16.0	0.63
I	18.7	0.736
J	6.0	0.236
K	16.0	0.63
L	18.7	0.736
M	8.2	0.323
N	8.0	0.315
O	2.5	0.098
P	2.0	0.079
Q	0.35	0.014
R	φ2.3	φ0.091
S	φ1.5	φ0.059

Figure A-5. EV-9200GC-80 Recommended Board Mounting Pattern (for reference only)



EV-9200GC-80-P1E

ITEM	MILLIMETERS	INCHES
A	19.7	0.776
B	15.0	0.591
C	0.65±0.02 × 19=12.35±0.05	$0.026^{+0.001}_{-0.002} \times 0.748 = 0.486^{+0.003}_{-0.002}$
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	15.0	0.591
F	19.7	0.776
G	6.0±0.05	$0.236^{+0.003}_{-0.002}$
H	6.0±0.05	$0.236^{+0.003}_{-0.002}$
I	0.35±0.02	$0.014^{+0.001}_{-0.001}$
J	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 Dimensions of mount pad for EV-9200 and that for target device (QFP) may be different in some parts. For the recommended mount pad dimensions for QFP, refer to "SEMICONDUCTOR DEVICE MOUNTING TECHNOLOGY MANUAL" (C10535E).

APPENDIX B EMBEDDED SOFTWARE

For efficient development and maintenance of the μPD178024 and 178124 subseries, the following embedded products are available.

Real-Time OS (1/2)

RX78K/0 Real-time OS	RX 78K/0 is a real-time OS conforming to the μITRON specifications. Tool (configurator) for generating nucleus of RX78K/0 and plural information tables is supplied. Used in combination with an optional assembler package (RA78K/0) and device file (DF178124). **<Precaution when using RX78K/0 in PC environment>** The real-time OS is a DOS-based application. It should be used in the DOS Prompt when using in Windows.
	Part number: μS××××RX78013-ΔΔΔΔ

Caution When purchasing the RX78K/0, 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-production 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)Note	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version)Note	3.5-inch 2HC FD
BB13		Windows (English version)Note	
3P16	HP9000 series 700	HP-UX (Rel. 10.10)	DAT (DDS)
3K13	SPARCstation	SunOS (Rel. 4.1.4), Solaris (Rel. 2.5.1)	3.5-inch 2HC FD
3K15			1/4-inch CGMT
3R13	NEWS (RISC)	NEWS-OS (Rel. 6.1)	3.5-inch 2HC FD

Note Can also be operated in DOS environment.

Real-Time OS (2/2)

MX78K0 OS	MX78K0 is an OS for μITRON specification subsets. A nucleus for the MX78K0 is also included as a companion product. This manages tasks, events, and time. In the task management, determining the task execution order and switching from task to the next task are performed. **<Precaution when using MX78K0 in PC environment>** The MX78K0 is a DOS-based application. It should be used in the DOS Prompt when using in Windows.
	Part number: μS××××MX78K0-ΔΔΔ

Remark ×××× and ΔΔΔ in the part number differ depending on the host machine and OS used.

μS××××MX78K0-ΔΔΔ

ΔΔΔ	Product Outline	Maximum Number for Use in Mass Production
001	Evaluation object	Use in preproduction stages.
××	Mass-production object	Use in mass production stages.
S01	Source program	Only the users who purchased mass-production objects are allowed to purchase this program.

××××	Host Machine	OS	Supply Medium
AA13	PC-9800 series	Windows (Japanese version)[Note]	3.5-inch 2HD FD
AB13	IBM PC/AT compatibles	Windows (Japanese version)[Note]	3.5-inch 2HC FD
BB13		Windows (English version)[Note]	
3P16	HP9000 series 700	HP-UX (Rel. 10.10)	DAT (DDS)
3K13	SPARCstation	SunOS (Rel. 4.1.4),	3.5-inch 2HC FD
3K15		Solaris (Rel. 2.5.1)	1/4-inch CGMT
3R13	NEWS (RISC)	NEWS-OS (Rel. 6.1)	3.5-inch 2HC FD

Note Can also be operated in DOS environment.

APPENDIX C REGISTER INDEX