To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

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

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

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

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

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

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

RENESAS

User's Manual

μPD789842 Subseries

8-Bit Single-Chip Microcontrollers

μPD789841
μPD789842
μPD78F9842

Document No. U13776EJ3V1UD00 (3rd edition)
Date Published August 2005 N CP(K)

[MEMO]

NOTES FOR CMOS DEVICES

① VOLTAGE APPLICATION WAVEFORM AT INPUT PIN

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (MAX) and V_{IH} (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (MAX) and V_{IH} (MIN).

② HANDLING OF UNUSED INPUT PINS

Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

③ PRECAUTION AGAINST ESD

A 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 when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up 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 benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

④ STATUS BEFORE INITIALIZATION

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

⑤ POWER ON/OFF SEQUENCE

In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current.

The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

⑥ INPUT OF SIGNAL DURING POWER OFF STATE

Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements.

Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

FIP and EEPROM are trademarks of NEC Electronics Corporation.
Windows and Windows NT are trademarks or registered trademarks of Microsoft Corporation in the United States and in other countries.
PC/AT is a trademark of International Business Machines Corporation.
HP9000 series 700 and HP-UX are trademarks of Hewlett-Packard Company.
SPARCstation is a trademark of SPARC International, Inc.
Solaris and SunOS are trademarks of Sun Microsystems, Inc.

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

- **The information in this document is current as of August, 2005. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".
  The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.
  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).
  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

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

(Note)
(1) "NEC Electronics" as used in this statement means NEC Electronics Corporation and also includes its majority-owned subsidiaries.
(2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

M8E 02. 11-1

Regional Information

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

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

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

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

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

NEC Electronics (Europe) GmbH
Duesseldorf, Germany
Tel: 0211-65030

- **Sucursal en España**
  Madrid, Spain
  Tel: 091-504 27 87
- **Succursale Française**
  Vélizy-Villacoublay, France
  Tel: 01-30-67 58 00
- **Filiale Italiana**
  Milano, Italy
  Tel: 02-66 75 41
- **Branch The Netherlands**
  Eindhoven, The Netherlands
  Tel: 040-265 40 10
- **Tyskland Filial**
  Taeby, Sweden
  Tel: 08-63 87 200
- **United Kingdom Branch**
  Milton Keynes, UK
  Tel: 01908-691-133

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

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

NEC Electronics Shanghai Ltd.
Shanghai, P.R. China
Tel: 021-5888-5400

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

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

J05.6

INTRODUCTION

Readers

This manual is intended for user engineers who wish to understand the functions of the μPD789842 Subseries to design and develop its application systems and programs.

- μPD789842 Subseries: μPD789841, μPD789842, and μPD78F9842

Purpose

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

Organization

Two manuals are available for the μPD789842 Subseries: this manual and the Instruction Manual (common to the 78K/0S Series).

μPD789842 Subseries User's Manual (This manual)	78K/0S Series Instruction User's Manual
• Pin functions • Internal block functions • Interrupts • Other internal peripheral functions • Electrical specifications	• CPU function • Instruction set • Instruction description

How to Read This Manual

It is assumed that the readers of this manual have general knowledge of electrical engineering, logic circuits, and microcontrollers.

To understand the overall functions of the μPD789842 Subseries
→ Read this manual in the order of the **CONTENTS**. The mark ★ shows major revised points.
How to read register formats
→ The name of a bit whose number is enclosed with < > is reserved in the assembler and is defined in the C compiler by the header file **sfrbit.h**.
To learn the detailed functions of a register whose register name is known
→ See **APPENDIX C**.
To learn the details of the instruction functions of the 78K/0S Series
→ Refer to **78K/0S Series Instruction User's Manual (U11047E)** available separately
To learn the electrical specifications of the μPD789842 Subseries
→ See **CHAPTER 19 ELECTRICAL SPECIFICATIONS**.

Conventions

Data significance:	Higher digits on the left and lower digits on the right
Active low representation:	$\overline{\times\times\times}$ (Overscore over pin or signal name)
Note:	Footnote for item marked with **Note** in the text
Caution:	Information requiring particular attention
Remark:	Supplementary information
Numerical representation:	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.

Documents Related to Devices

Document Name	Document No.
μPD789842 Subseries User's Manual	This manual
78K/0S Series Instructions User's Manual	U11047E

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

Document Name		Document No.
RA78K0S Assembler Package	Operation	U14876E
	Language	U14877E
	Structured Assembly Language	U11623E
CC78K0S C Compiler	Operation	U14871E
	Language	U14872E
SM78K Series System Simulator Ver. 2.30 or Later	Operation (Windows™ Based)	U15373E
	External Part User Open Interface Specification	U15802E
ID78K Series Integrated Debugger Ver. 2.30 or Later	Operation (Windows Based)	U15185E
Project Manager Ver. 3.12 or Later (Windows Based)		U14610E

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

Document Name	Document No.
IE-78K0S-NS In-Circuit Emulator	U13549E
IE-78K0S-NS-A In-Circuit Emulator	U15207E
IE-789842-NS-EM1 Emulation Board	U14545E

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

Documents Related to Flash Memory Writing

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

Other Related Documents

Document Name	Document No.
SEMICONDUCTOR SELECTION GUIDE -Products and Packages-	X13769X
Semiconductor Device Mount Manual	**Note**
Quality Grades on NEC Semiconductor Devices	C11531E
NEC Semiconductor Device Reliability/Quality Control System	C10983E
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892E

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

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

CONTENTS

CHAPTER 1 GENERAL

1.1 Features

- ROM and RAM capacity

Item / Product Name	Program Memory (ROM/Flash Memory)	Data Memory (High-Speed RAM)
μPD789841	8 KB	256 bytes
μPD789842	16 KB	
μPD78F9842	16 KB	

- Minimum instruction execution time changeable from high-speed (0.24 μs) to low-speed (0.96 μs) (System clock 8.38 MHz operation)
- I/O port: 30 lines
- Serial interface (UART00): 1 channel
- Timer: 6 channels
  - 10-bit inverter control timer: 1 channel
  - 8-bit timer/event counter: 2 channels
  - 8-bit timer counter: 1 channel
  - Watch timer: 1 channel
  - Watchdog timer: 1 channel
- 8-bit resolution A/D converter: 8 channels
- Multiplier: 10 bits × 10 bits = 20 bits
- SWAP: The contents of the higher four bits of an 8-bit register can be exchanged with the lower four bits
- Vectored interrupt sources: 15
- Supply voltage: V_{DD} = 4.0 to 5.5 V

1.2 Applications

Inverter-driven air conditioners etc.

★ 1.3 Ordering Information

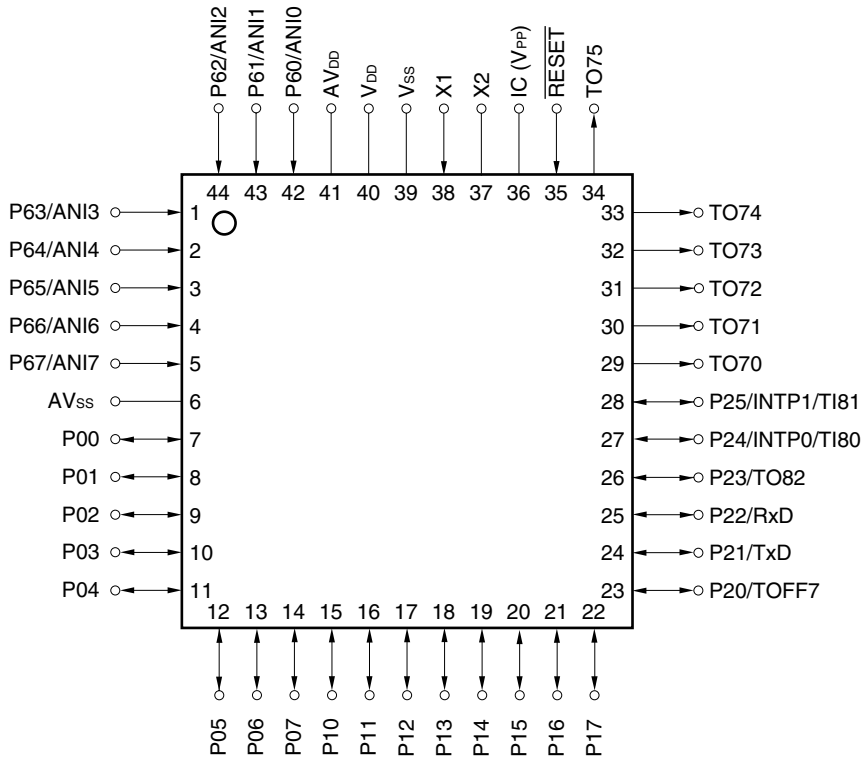
Part Number	Package	Internal ROM
μPD789841GB-×××-3BS-MTX	44-pin plastic QFP (10 × 10)	Mask ROM
μPD789841GB-×××-8ES	44-pin plastic LQFP (10 × 10)	Mask ROM
μPD789841GB-×××-8ES-A	44-pin plastic LQFP (10 × 10)	Mask ROM
μPD789842GB-×××-3BS-MTX	44-pin plastic QFP (10 × 10)	Mask ROM
μPD789842GB-×××-8ES	44-pin plastic LQFP (10 × 10)	Mask ROM
μPD789842GB-×××-8ES-A	44-pin plastic LQFP (10 × 10)	Mask ROM
μPD78F9842GB-3BS-MTX	44-pin plastic QFP (10 × 10)	Flash memory
μPD78F9842GB-8ES	44-pin plastic LQFP (10 × 10)	Flash memory
μPD78F9842GB-8ES-A	44-pin plastic LQFP (10 × 10)	Flash memory

Remarks 1. ××× indicates ROM code suffix.

2. Products that have the part numbers suffixed by "-A" are lead-free products.

1.4 Pin Configuration (Top View)

- **44-pin plastic QFP (10 × 10)**
- **44-pin plastic LQFP (10 × 10)**



Cautions 1. Connect the IC pin directly to the V_{SS} pin.

2. Connect the AV_{DD} pin to the V_{DD} pin.

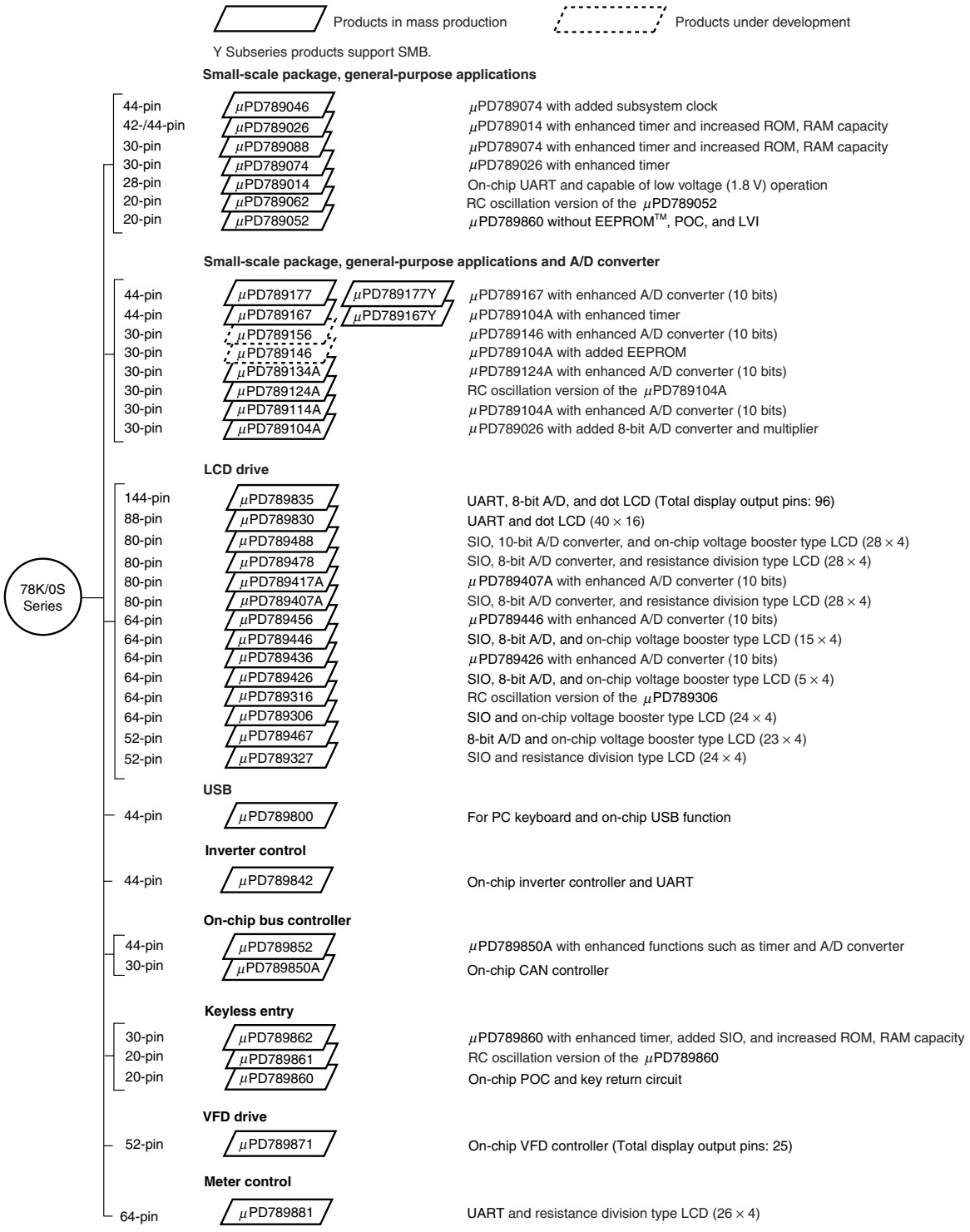
3. Connect the AV_{SS} pin to the V_{SS} pin.

Remark Pin connections in parentheses are intended for the μPD78F9842.

ANI0 to ANI7:	Analog input	RxD:	Receive data
AV_{DD}:	Analog power supply	TI80, TI81:	Timer input
AV_{SS}:	Analog ground	TO70 to TO75, TO82:	Timer output
IC:	Internally connected	TOFF7:	Timer output off
INTP0, INTP1:	External interrupt input	TxD:	Transmit data
P00 to P07:	Port 0	V_{DD}:	Power supply
P10 to P17:	Port 1	V_{PP}:	Programming power supply
P20 to P25:	Port 2	V_{SS}:	Ground
P60 to P67:	Port 6	X1, X2:	Crystal
$\overline{RESET}$:	Reset		

⋆1.5 78K/0S Series Lineup

The products in the 78K/0S Series are listed below. The names enclosed in boxes are subseries names.



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

The major functional differences between the subseries are listed below.

Series for general-purpose applications and LCD drive

Subseries Name \ Function		ROM Capacity	Timer				8-Bit A/D	10-Bit A/D	Serial Interface	I/O	V_{DD} MIN. Value	Remarks
			8-Bit	16-Bit	Watch	WDT						
Small-scale package, general-purpose applications	μPD789046	16 KB	1 ch	1 ch	1 ch	1 ch	–	–	1 ch (UART: 1 ch)	34	1.8 V	–
	μPD789026	4 KB to 16 KB			–							
	μPD789088	16 KB to 32 KB	3 ch							24		
	μPD789074	2 KB to 8 KB	1 ch									
	μPD789014	2 KB to 4 KB	2 ch	–						22		
	μPD789062	4 KB							–	14		RC oscillation version
	μPD789052											–
Small-scale package, general-purpose applications and A/D converter	μPD789177	16 KB to 24 KB	3 ch	1 ch	1 ch	1 ch	–	8 ch	1 ch (UART: 1 ch)	31	1.8 V	–
	μPD789167						8 ch	–				
	μPD789156	8 KB to 16 KB	1 ch		–		–	4 ch		20		On-chip EEPROM
	μPD789146						4 ch	–				
	μPD789134A	2 KB to 8 KB					–	4 ch				RC oscillation version
	μPD789124A						4 ch	–				
	μPD789114A						–	4 ch				–
	μPD789104A						4 ch	–				
LCD drive	μPD789835	24 KB to 60 KB	6 ch	–	1 ch	1 ch	3 ch	–	1 ch (UART: 1 ch)	37	1.8 V[Note]	Dot LCD supported
	μPD789830	24 KB	1 ch	1 ch			–			30	2.7 V	
	μPD789488	32 KB to 48 KB	3 ch					8 ch	2 ch (UART: 1 ch)	45	1.8 V	–
	μPD789478	24 KB to 48 KB					8 ch	–				
	μPD789417A	12 KB to 24 KB					–	7 ch	1 ch (UART: 1 ch)	43		
	μPD789407A						7 ch	–				
	μPD789456	12 KB to 16 KB	2 ch				–	6 ch		30		
	μPD789446						6 ch	–				
	μPD789436						–	6 ch		40		
	μPD789426						6 ch	–				
	μPD789316	8 KB to 16 KB					–		2 ch (UART: 1 ch)	23		RC oscillation version
	μPD789306											–
	μPD789467	4 KB to 24 KB		–			1 ch		–	18		
	μPD789327						–		1 ch	21		

Note Flash memory version: 3.0 V

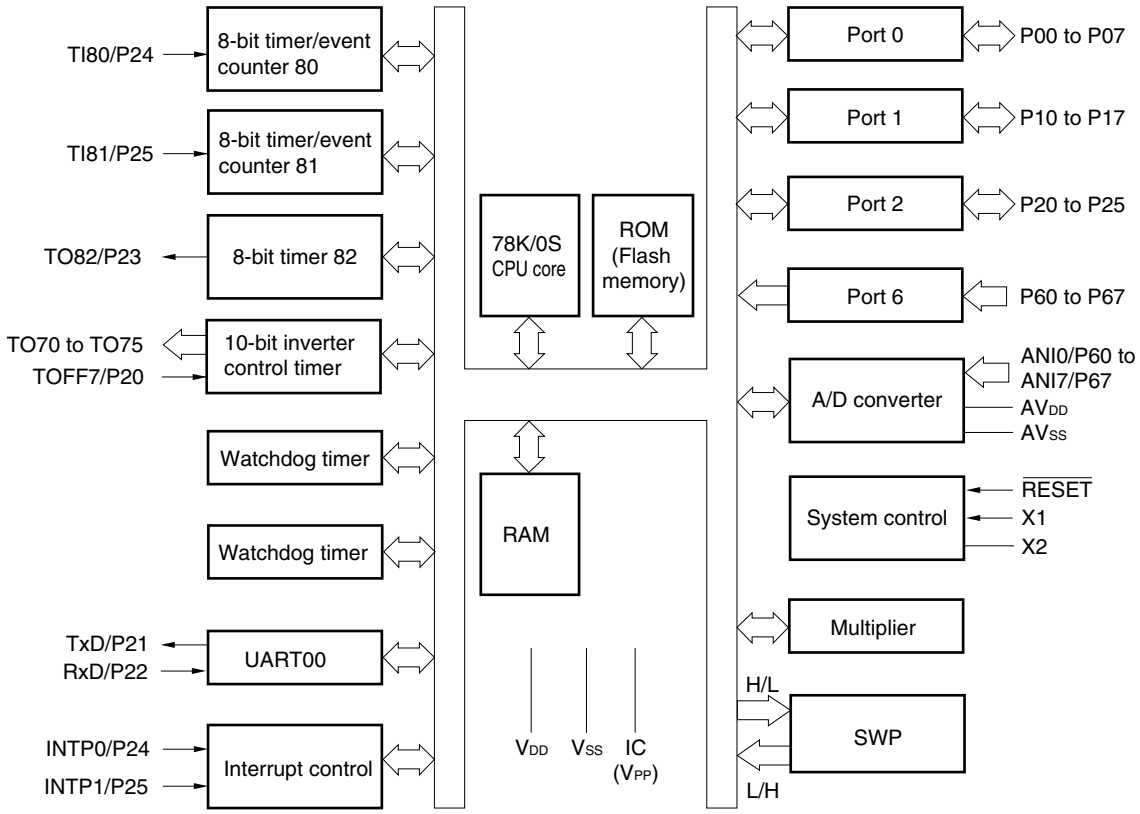
Series for ASSP

Subseries Name \ Function		ROM Capacity	Timer				8-Bit A/D	10-Bit A/D	Serial Interface	I/O	V_{DD} MIN. Value	Remarks
			8-Bit	16-Bit	Watch	WDT						
USB	μPD789800	8 KB	2 ch	–	–	1 ch	–	–	2 ch (USB: 1 ch)	31	4.0 V	–
Inverter control	μPD789842	8 KB to 16 KB	3 ch	**Note 1**	1 ch	1 ch	8 ch	–	1 ch (UART: 1 ch)	30	4.0 V	–
On-chip bus controller	μPD789852	24 KB to 32 KB	3 ch	1 ch	–	1 ch	–	8 ch	3 ch (UART: 2 ch)	31	4.0 V	–
	μPD789850A	16 KB	1 ch				4 ch	–	2 ch (UART: 1 ch)	18		
Keyless entry	μPD789861	4 KB	2 ch	–	–	1 ch	–	–	–	14	1.8 V	RC oscillation version, on-chip EEPROM
	μPD789860											On-chip EEPROM
	μPD789862	16 KB	1 ch	2 ch					1 ch (UART: 1 ch)	22		
VFD drive	μPD789871	4 KB to 8 KB	3 ch	–	1 ch	1 ch	–	–	1 ch	33	2.7 V	–
Meter control	μPD789881	16 KB	2 ch	1 ch	–	1 ch	–	–	1 ch (UART: 1 ch)	28	2.7 V[Note 2]	–

Notes 1. 10-bit timer: 1 channel

2. Flash memory version: 3.0 V

1.6 Block Diagram



Remark Pin connections in parentheses are intended for the μPD78F9842.

1.7 Outline of Functions

Item		μPD789841	μPD789842	μPD78F9842
Internal memory	ROM structure	Mask ROM		Flash memory
	ROM	8 KB	16 KB	16 KB
	RAM	256 bytes		
Minimum instruction execution time		0.24/0.96 μs (operation with system clock running at 8.38 MHz)		
Instruction set		• 16-bit operations • Bit manipulations (such as set, reset, and test)		
I/O ports		Total of 30 port pins 22 CMOS I/O pins 8 CMOS input pins		
Serial interface		UART: 1 channel		
Timers		• 10-bit inverter control timer: 1 channel • 8-bit timer/event counters: 2 channels • 8-bit timer: 1 channel • Watch timer: 1 channel • Watchdog timer: 1 channel		
A/D converters		8-bit resolution × 8 channels		
Multiplier		10 bits × 10 bits = 20 bits		
SWAP		The contents of the higher four bits of an 8-bit register can be exchanged with the lower four bits.		
Vectored interrupt sources	Maskable	12 internal and 2 external interrupts		
	Non-maskable	Internal interrupt		
Power supply voltage		V_{DD} = 4.0 to 5.5 V		
Operating ambient temperature		T_A = −40 to +85°C		
Package		• 44-pin plastic QFP (10 × 10) • 44-pin plastic LQFP (10 × 10)		

The following shows an outline of the timers.

		TM7	TM80	TM81	TM82	WT[Note 1]	WDT[Note 2]
Operating mode	Interval timer	1 channel	1 channel	1 channel	1 channel	1 channel	1 channel
	External event counter	–	1 channel	1 channel	–	–	–
Function	Timer output	1 output	–	–	1 output	–	–
	Square-wave output	–	–	–	1 output	–	–
	Interrupt source	1	1	1	1	2	2

Notes 1. The watch timer can perform both watch timer and interval timer functions at the same time.

2. The watchdog timer provides a watchdog timer function and interval timer function, but only one of the two functions should be selected.

CHAPTER 2 PIN FUNCTIONS

2.1 Pin Function List

(1) Port pins

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P07	I/O	Port 0 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of on-chip pull-up resistors can be specified by pull-up resistor option register 0 (PU0).	Input	–
P10 to P17	I/O	Port 1 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, use of on-chip pull-up resistors can be specified by pull-up resistor option register 0 (PU0).	Input	–
P20	I/O	Port 2 6-bit I/O port Input/output can be specified in 1-bit units. Use of on-chip pull-up resistors can be specified by pull-up resistor option register B2 (PUB2).	Input	TOFF7
P21				TxD
P22				RxD
P23				TO82
P24				INTP0/TI80
P25				INTP1/TI81
P60 to P67	Input	Port 6 8-bit input-only port	Input	ANI0 to ANI7

(2) Non-port pins

Pin Name	I/O	Function	After Reset	Alternate Function
INTP0	Input	External interrupt input for which valid edge (rising and/or falling edge) can be specified	Input	P24/TI80
INTP1				P25/TI81
RxD	Input	Serial data input to asynchronous serial interface	Input	P22
TxD	Output	Serial data output from asynchronous serial interface	Input	P21
TO70 to TO75	Output	10-bit inverter control timer output	Output	–
TOFF7	Input	External input to stop timer output (TO70 to TO75)	Input	P20
TI80	Input	External count clock input to TM80	Input	P24/INTP0
TI81		External count clock input to TM81		P25/INTP1
TO82	Output	TM82 timer output	Input	P23
ANI0 to ANI7	Input	A/D converter analog input	Input	P60 to P67
AV_{SS}	–	A/D converter ground potential	–	–
AV_{DD}		A/D converter analog power supply	–	–
X1	Input	Connection of crystal for system clock oscillation	–	–
X2	–		–	–
$\overline{RESET}$	Input	System reset input	Input	–
V_{DD}	–	Positive supply voltage for ports	–	–
V_{SS}	–	Ground potential for ports	–	–
IC	–	Internally connected. Connect this pin directly to the V_{SS} pin.	–	–
★ V_{PP}	–	This pin is used to set flash memory programming mode and applies a high voltage when a program is written or verified.	–	–

2.2 Description of Pin Functions

2.2.1 P00 to P07 (Port 0)

These pins constitute an 8-bit I/O port and can be set to input or output port mode in 1-bit units by using port mode register 0 (PM0). When these pins are used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 0 (PU0).

2.2.2 P10 to P17 (Port 1)

These pins constitute an 8-bit I/O port and can be set in input or output port mode in 1-bit units by using port mode register 1 (PM1). When these pins are used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 0 (PU0).

2.2.3 P20 to P25 (Port 2)

These pins constitute a 6-bit I/O port. In addition, these pins provide a function to perform external interrupt input, timer I/O, and UART data I/O.

Port 2 can be specified in the following operation modes in 1-bit units.

(1) Port mode

In port mode, P20 to P25 function as a 6-bit I/O port. Port 2 can be set to input or output mode in 1-bit units by using port mode register 2 (PM2). Use of on-chip pull-up resistors for the P20 to P25 pins can be specified by pull-up resistor option register B2 (PUB2).

(2) Control mode

In this mode, P20 to P25 function as external interrupt inputs, timer I/O, and UART data I/O.

(a) INTP0, INTP1

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

(b) TOFF7

This is the external input pin to stop 10-bit inverter control timer output (TO70 to TO75).

(c) TI80, TI81

These are the external count clock input pins of 8-bit timer/event counters 80 and 81.

(d) TO82

This is the timer output pin of 8-bit timer 82.

(e) RxD, TxD

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

Caution When using P20 to P25 as UART data I/O pins, the I/O and output latch must be set according to the function to be used. For details of the setting, see 11.3 (1).

2.2.4 P60 to P67 (Port 6)

These pins constitute an 8-bit input-only port. In addition to general-purpose input port pins, they can also function as A/D converter input pins.

(1) Port mode

In port mode, P60 to P67 function as an 8-bit input-only port.

(2) Control mode

In control mode, P60 to P67 function as A/D converter analog inputs (ANI0 to ANI7).

2.2.5 TO70 to TO75

These are the timer output pins of the 10-bit inverter control timer.

2.2.6 $\overline{\text{RESET}}$

This pin inputs an active-low system reset signal.

2.2.7 X1, X2

These pins are used to connect a crystal for system clock oscillation.

2.2.8 AV_{DD}

This is the analog power supply pin of the A/D converter. Always use the same potential as that of the V_{DD} pin even when A/D converter is not used.

2.2.9 AV_{SS}

This is the ground potential pin of the A/D converter. Always use the same potential as that of the V_{SS} pin even when the A/D converter is not used.

2.2.10 V_{DD}

V_{DD} supplies positive power.

2.2.11 V_{SS}

V_{SS} is the ground potential pin.

2.2.12 V_{PP} (μPD78F9842 only)

A high voltage should be applied to this pin when the flash memory programming mode is set and when the program is written or verified.

★ Connect this pin in either of the following ways.

- Independently connect to a 10 kΩ pull-down resistor.
- By using a jumper on the board, connect directly to the dedicated flash programmer in the programming mode or to V_{SS} in the normal operation mode.

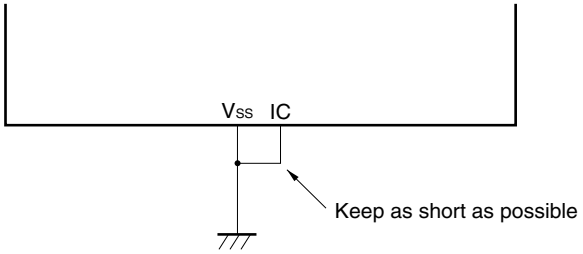
If the wiring between the V_{PP} and V_{SS} pins is long or external noise is superimposed on the V_{PP} pin, the user program may malfunction.

2.2.13 IC

The IC (Internally Connected) pin is used to set the μPD789842 Subseries to test mode before shipment. In normal operation mode, directly connect this pin to the V_{SS} pin with as short a wiring length as possible.

If a potential difference is generated between the IC pin and V_{SS} pin due to a long wiring length between the IC pin and V_{SS} pin or external noise superimposed on the IC pin, the user program may not run correctly.

- Directly connect the IC pin to the V_{SS} pin.



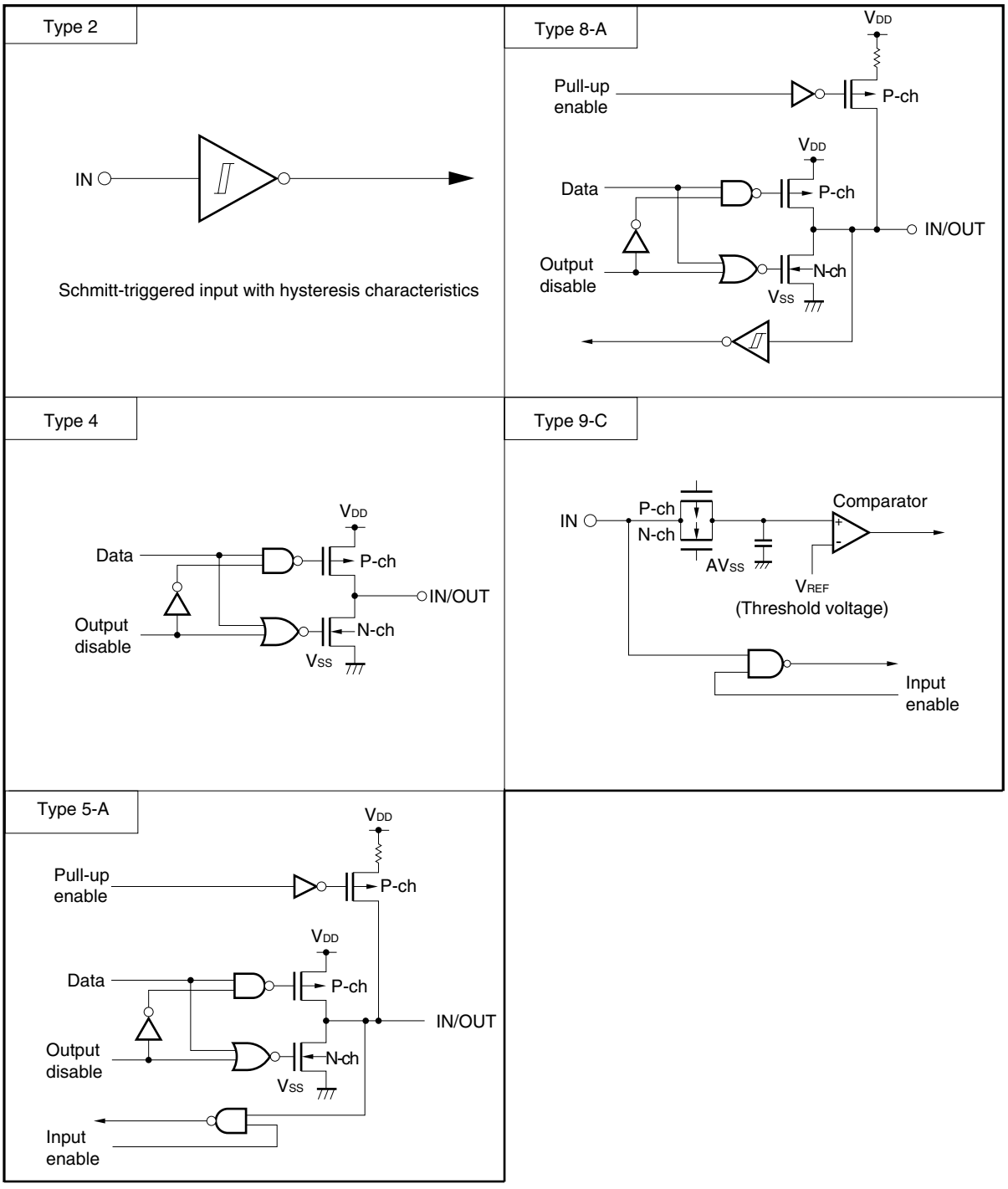
2.3 Pin I/O Circuits and Handling of Unused Pins

Table 2-1 lists the types of I/O circuits for each pin and explains how unused pins are handled.
Figure 2-1 shows the configuration of each type of I/O circuit.

Table 2-1. Type of I/O Circuit for Each Pin and Handling of Unused Pins

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
P00 to P07	5-A	I/O	Input: Connect to the V_{DD} or V_{SS} pin via a resistor. Output: Leave open.
P10 to P17			
P20/TOFF7	8-A		
P21/TxD			
P22/RxD			
P23/TO82			
P24/INTP0/TI80			
P25/INTP1/TI81			
P60/ANI0 to P67/ANI7	9-C	Input	Directly connect to the V_{DD} or V_{SS} pin.
TO70 to TO75	4	Output	Independently connect to the V_{DD} or V_{SS} pin via a resistor.
AV_{DD}	–	–	Directly connect to the V_{DD} pin.
AV_{SS}	–	–	Directly connect to the V_{SS} pin.
$\overline{\text{RESET}}$	2	Input	–
IC (mask ROM version)	–	–	Directly Connect to the V_{SS} pin.
★ V_{PP} (flash memory version)	–	–	Independently connect via a 10 kΩ pull-down resistor or directly connect to V_{SS}.

Figure 2-1. Pin I/O Circuits



CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Space

Products in the μPD789842 Subseries can each access up to 64 KB of memory space.

Figures 3-1 to 3-3 show the memory maps.

Figure 3-1. Memory Map (μPD789841)

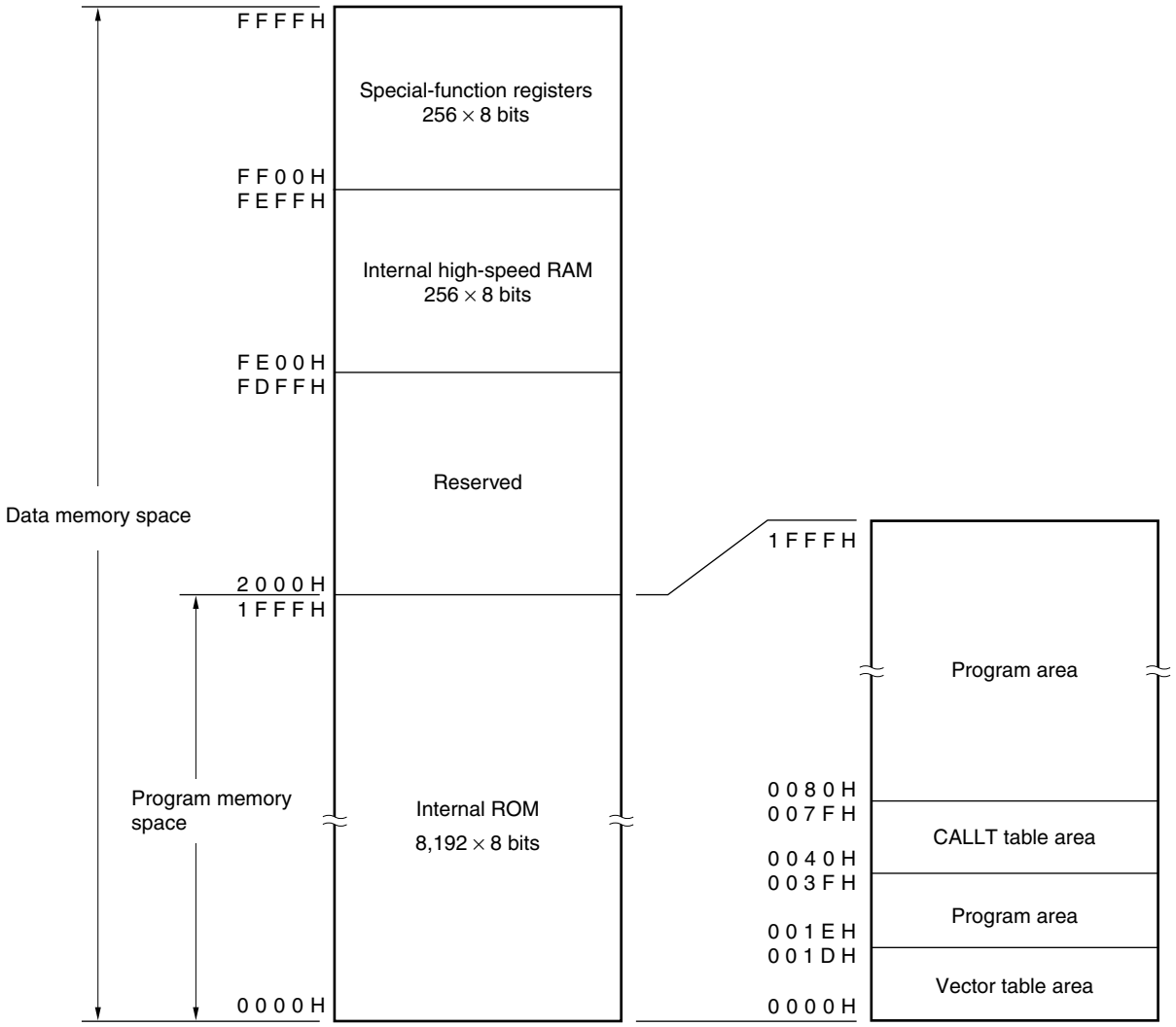


Figure 3-2. Memory Map (μPD789842)

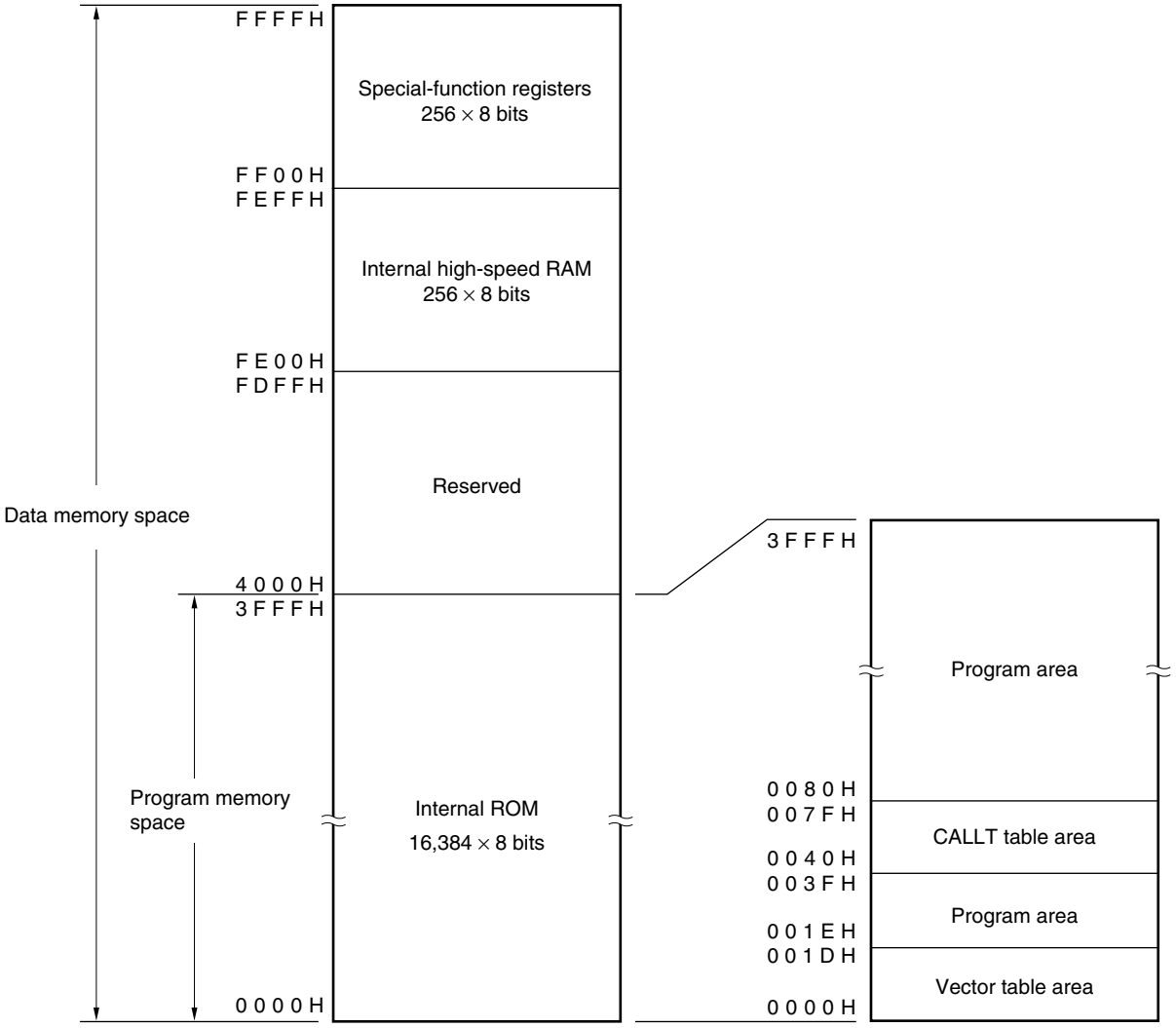
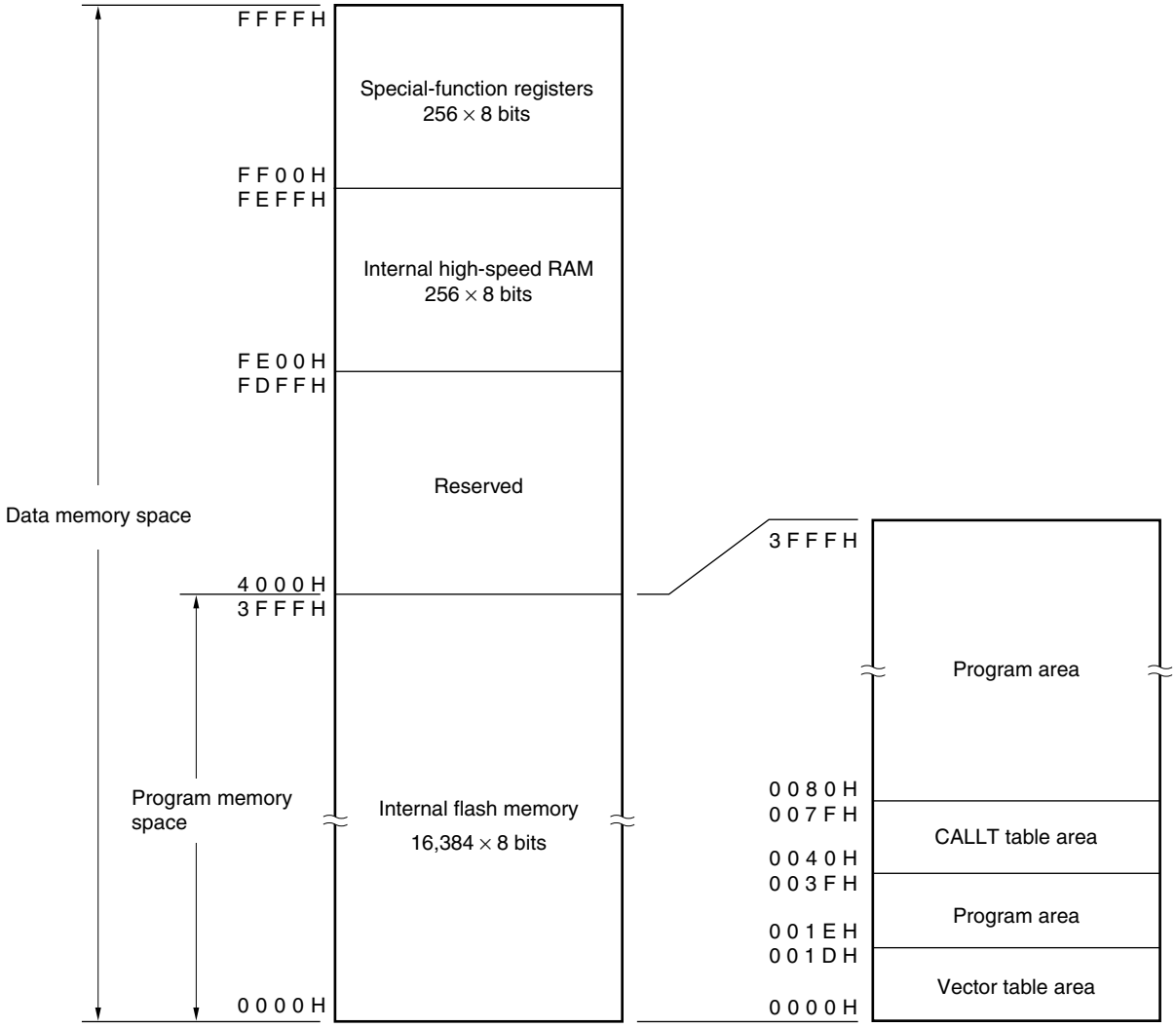


Figure 3-3. Memory Map (μPD78F9842)



3.1.1 Internal program memory space

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

The μPD789842 Subseries provides the following internal ROMs (mask ROM or flash memory) containing the following capacities.

Table 3-1. Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
μPD789841	Mask ROM	8,192 × 8 bits
μPD789842		16,384 × 8 bits
μPD78F9842	Flash memory	16,384 × 8 bits

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

(1) Vector table area

A 30-byte area of addresses 0000H to 001DH is reserved as a vector table area. This area stores program start addresses to be used when branching by $\overline{\text{RESET}}$ input or interrupt request generation. Of a 16-bit program address, the lower 8 bits are stored in an even address, and the higher 8 bits are stored in an odd address.

Table 3-2. Vector Table

Vector Table Address	Interrupt Request	Vector Table Address	Interrupt Request
0000H	$\overline{\text{RESET}}$ input	0010H	INTST00
0004H	INTWDT	0012H	INTWT
0006H	INTP0	0014H	INTWTI
0008H	INTP1	0016H	INTTM80
000AH	INTTM7	0018H	INTTM81
000CH	INTSER00	001AH	INTTM82
000EH	INTSR00	001CH	INTAD

(2) CALLT instruction table area

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

3.1.2 Internal data memory (internal high-speed RAM) space

The μPD789842 Subseries provides a 256-byte internal high-speed RAM.

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

3.1.3 Special function register (SFR) area

Special-function registers (SFRs) of on-chip peripheral hardware are allocated to an area of FF00H to FFFFH (see **Table 3-3**).

3.1.4 Data memory addressing

Each product in the μPD789842 Subseries is provided with a wide range of addressing modes to make memory manipulation as efficient as possible. A data memory area (FE00H to FFFFH) can be accessed using a unique addressing mode according to its use, such as a special-function register (SFR). Figures 3-4 to 3-6 illustrate the data memory addressing modes.

Figure 3-4. Data Memory Addressing Modes (μPD789841)

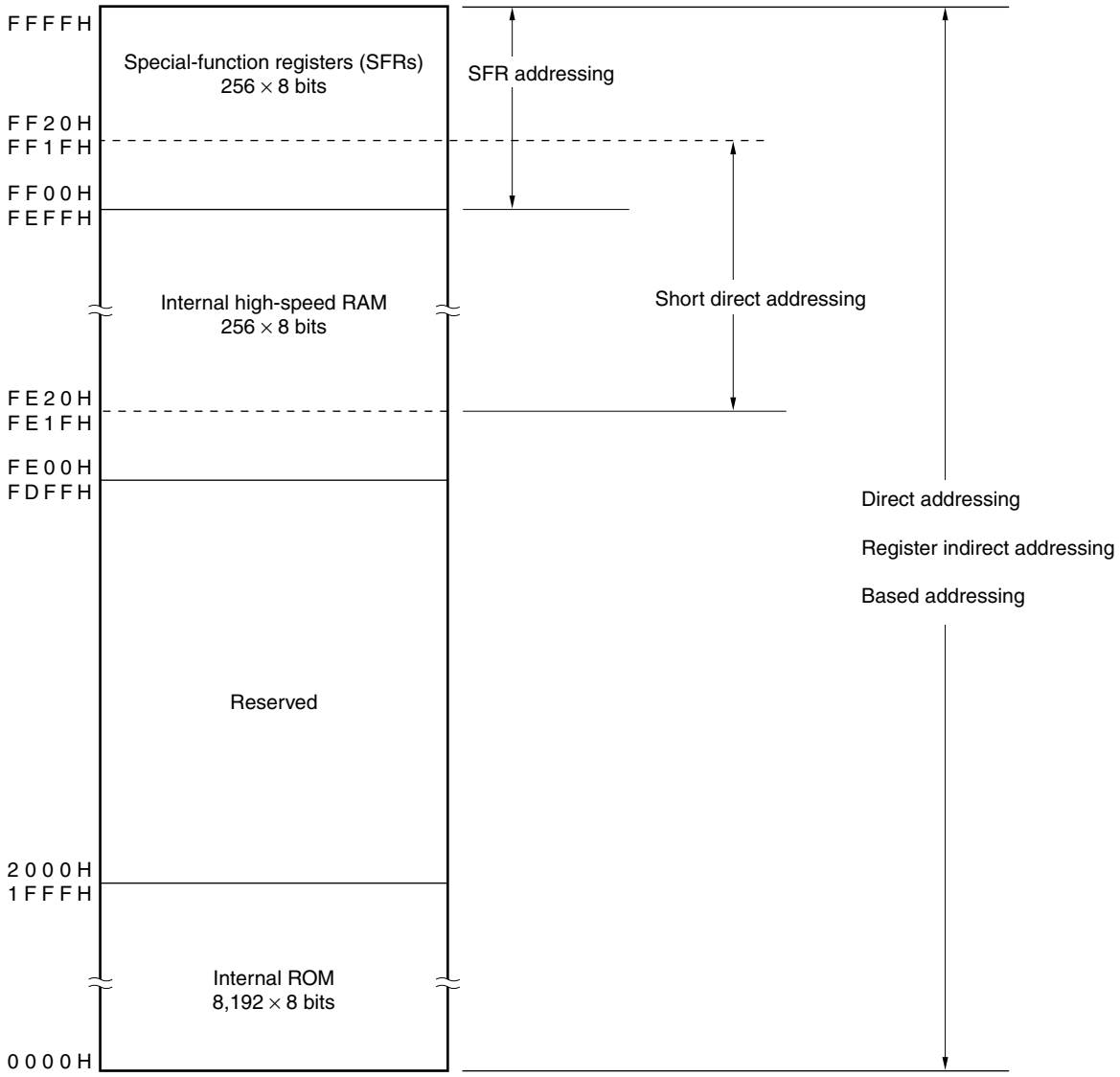


Figure 3-5. Data Memory Addressing Modes (μPD789842)

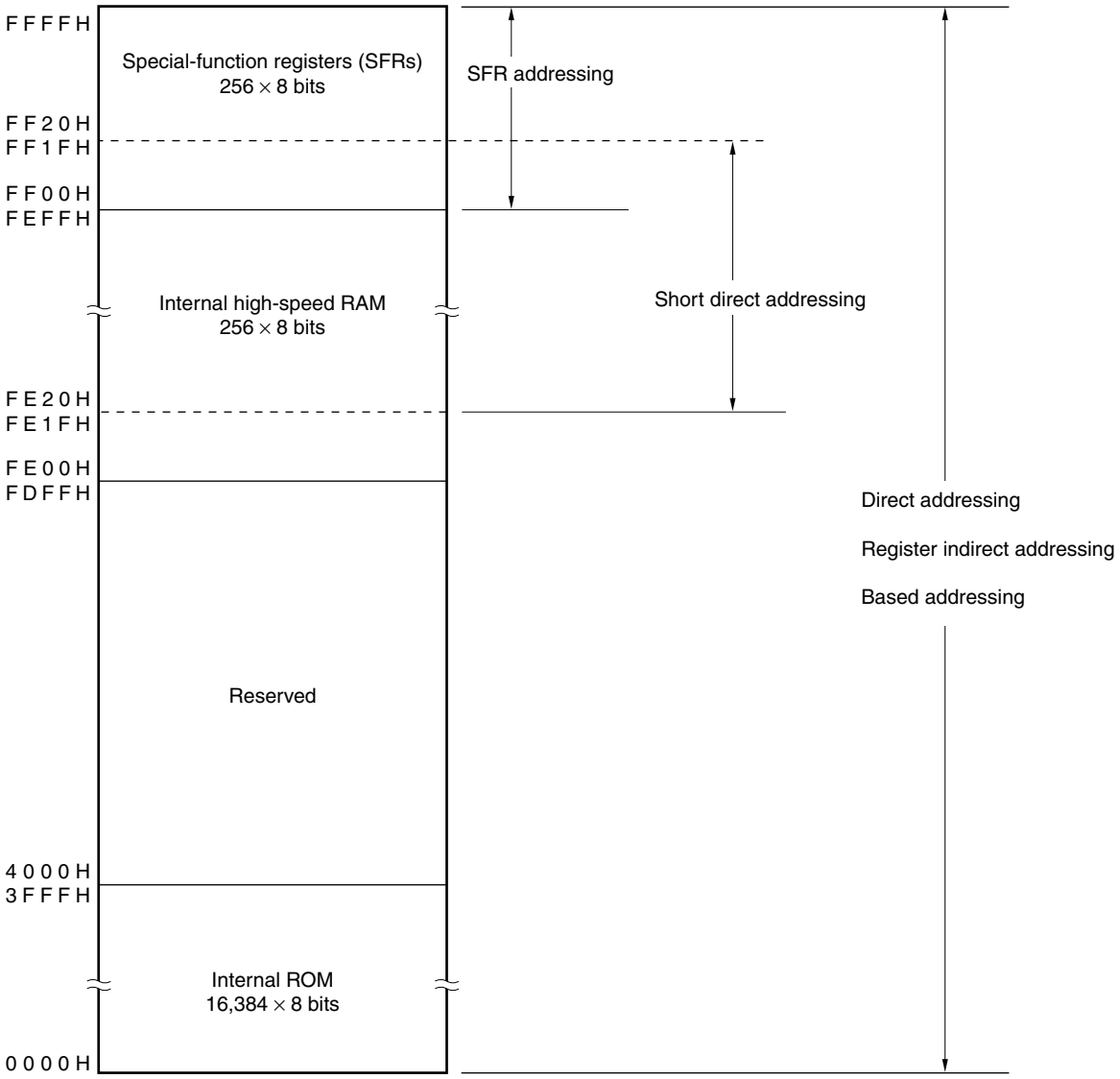
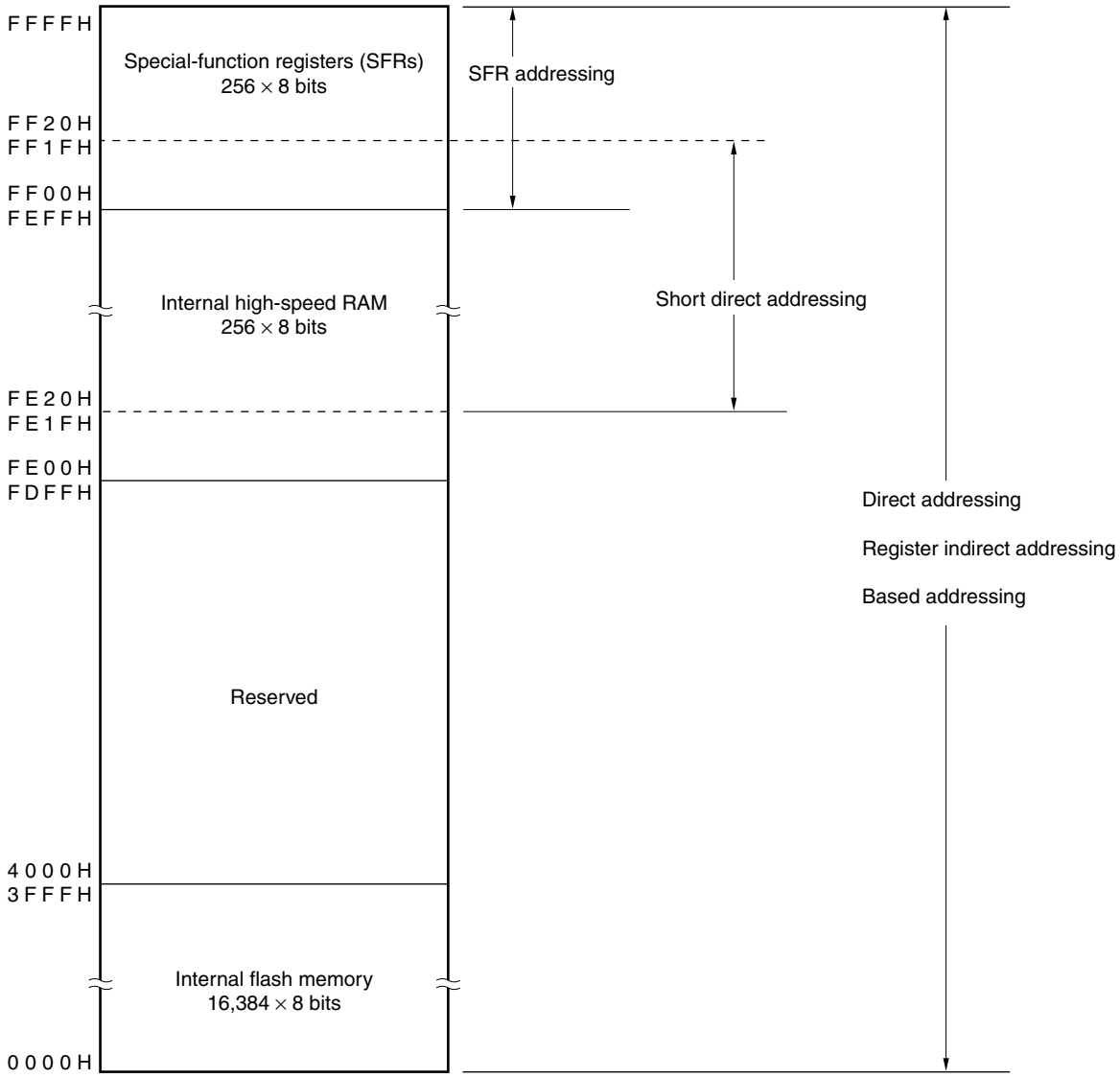


Figure 3-6. Data Memory Addressing Modes (μPD78F9842)



3.2 Processor Registers

The μPD789842 Subseries provides the following on-chip processor registers.

3.2.1 Control registers

The control registers contain special functions to control the program sequence statuses and stack memory. The program counter, program status word, and stack pointer are the control registers.

(1) Program counter (PC)

The program counter is a 16-bit register that holds the address information of the next program to be executed.

In normal operation, the PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data or register contents are set.

$\overline{\text{RESET}}$ input sets the reset vector table values at addresses 0000H and 0001H to the program counter.

Figure 3-7. Program Counter Configuration

	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 or 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 RETI and POP PSW instructions.

$\overline{\text{RESET}}$ input sets the PSW to 02H.

Figure 3-8. Program Status Word Configuration

	7							0
PSW	IE	Z	0	AC	0	0	1	CY

(a) Interrupt enable flag (IE)

This flag controls the interrupt request acknowledgment operations of CPU.

When IE = 0, the IE flag is set to the interrupt disabled (DI) status. All interrupt requests except non-maskable interrupts are disabled.

When IE = 1, the IE flag is set to the interrupt enabled (EI) status and interrupt request acknowledgment is controlled by the interrupt mask flag for each interrupt source.

This flag is reset to 0 upon DI instruction execution or interrupt acknowledgment and is set to 1 upon EI instruction execution.

(b) Zero flag (Z)

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

(c) Auxiliary carry flag (AC)

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

(d) 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 used to hold the start address of the memory stack area. Only the internal high-speed RAM area can be set as the stack area.

Figure 3-9. Stack Pointer Configuration

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

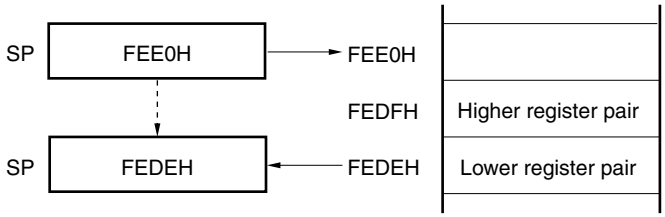
The SP is decremented ahead of write (save) to the stack memory and is incremented after read (restore) from the stack memory.

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

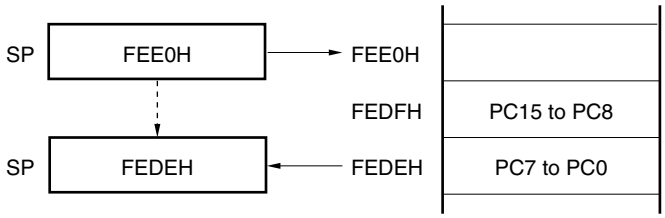
Caution Since $\overline{\text{RESET}}$ input makes the SP contents undefined, be sure to initialize the SP before instruction execution.

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

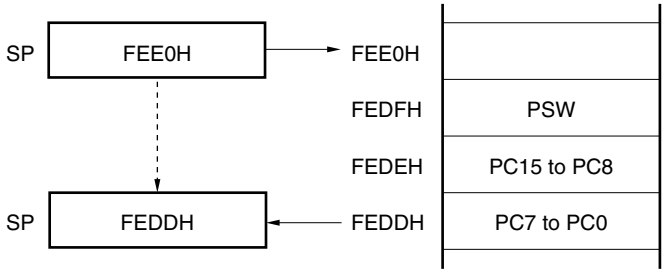
(a) PUSH rp instruction (when SP is FEE0H)



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

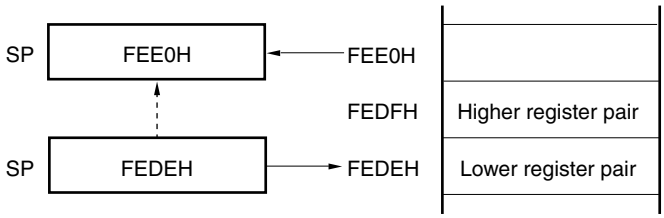


(c) Interrupt instruction (when SP is FEE0H)

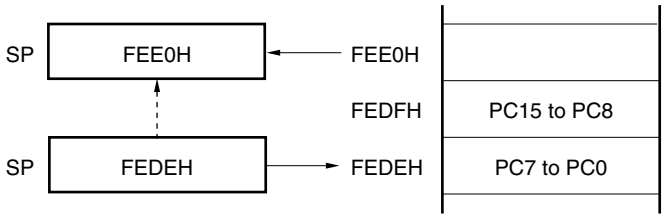


★ **Figure 3-11. Data to Be Restored from Stack Memory**

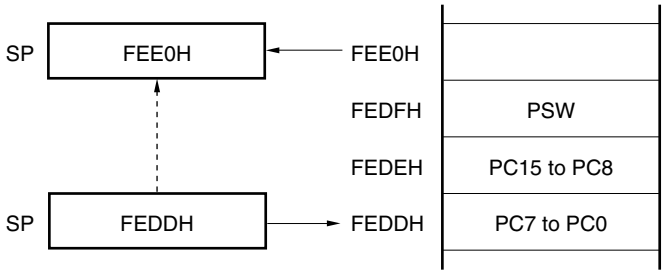
(a) POP rp instruction (when SP is FEDEH)



(b) RET instructions (when SP is FEDEH)



(c) RETI instruction (when SP is FEDDH)



3.2.2 General-purpose registers

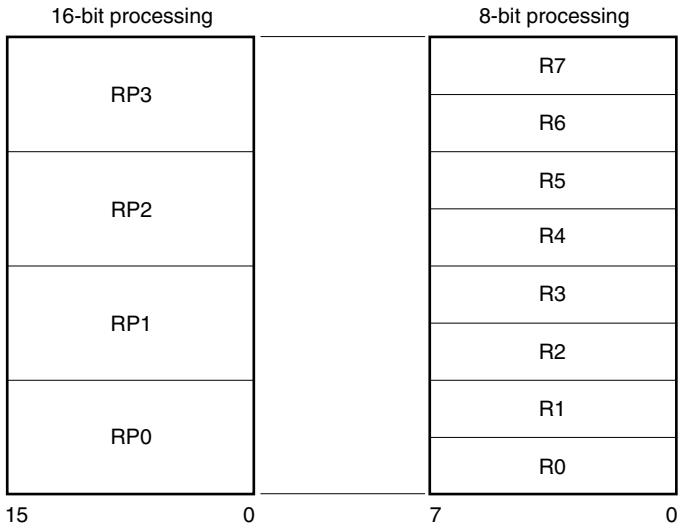
The general-purpose registers consists of eight 8-bit registers (X, A, C, B, E, D, L, and H).

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

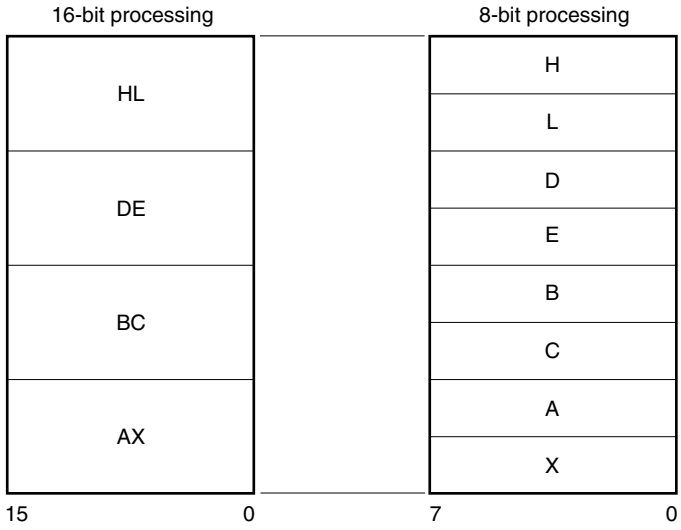
The registers can be described in terms of functional names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

Figure 3-12. General-Purpose Register Configuration

(a) Absolute names



(b) Functional names



3.2.3 Special-function registers (SFRs)

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

The special-function registers are allocated in the 256-byte area FF00H to FFFFH.

The special-function registers can be manipulated, like the general-purpose registers, with operation, transfer, and bit manipulation instructions. The manipulatable bit unit (1, 8, or 16) differs depending on the special-function register type.

Each manipulation bit unit can be specified as follows.

- 1-bit manipulation
  Describe a symbol reserved by the assembler as the operand of a 1-bit manipulation instruction (sfr.bit). An address can also be specified.

- 8-bit manipulation
  Describe a symbol reserved by the assembler as the operand of an 8-bit manipulation instruction (sfr). An address can also be specified.

- 16-bit manipulation
  Describe a symbol reserved by the assembler as the operand of a 16-bit manipulation instruction. When specifying an address, describe an even address.

Table 3-3 lists the special-function registers. The meanings of the symbols in this table are as follows.

- Symbol
  Indicates the addresses of the implemented special-function registers. The symbols shown in this column are reserved words in the assembler, and have already been defined in the header file "sfrbit.h" in the C compiler. Therefore, these symbols can be used as instruction operands if the assembler or integrated debugger is used.

- R/W
  Indicates whether the special-function register in question can be read or written.
  R/W: Read/write
  R: Read only
  W: Write only

- Number of bits manipulated simultaneously
  Indicates the bit units (1, 8, and 16) in which the special-function register in question can be manipulated.

- After reset
  Indicates the status of the special-function register when the $\overline{\text{RESET}}$ signal is input.

Table 3-3. Special-Function Registers (1/2)

Address	Special-Function Register (SFR) Name	Symbol		R/W	Number of Bits Manipulated Simultaneously			After Reset
					1 Bit	8 Bits	16 Bits	
FF00H	Port register 0	P0		R/W	√	√	–	00H
FF01H	Port register 1	P1			√	√	–	
FF02H	Port register 2	P2			√	√	–	
FF06H	Port register 6	P6		R	√	√	–	Undefined
FF08H	10-bit buffer register 0	BFCM0	BFCM0L	R/W	–	√	–	0000H
FF09H			–		–	–	√Note	
FF0AH	10-bit buffer register 1	BFCM1	BFCM1L		–	√	–	
FF0BH			–		–	–	√Note	
FF0CH	10-bit buffer register 2	BFCM2	BFCM2L		–	√	–	
FF0DH			–		–	–	√Note	
FF0EH	10-bit buffer register 3	BFCM3	BFCM3L		–	√	–	00FFH
FF0FH			–		–	–	√Note	
FF11H	A/D conversion result register	ADCRH		R	–	√	–	Undefined
FF14H	10-bit compare register 0	CM0		R/W	–	–	√Note	0000H
FF15H								
FF16H	10-bit compare register 1	CM1			–	–	√Note	
FF17H								
FF18H	10-bit compare register 2	CM2			–	–	√Note	
FF19H								
FF1AH	10-bit compare register 3	CM3			–	–	√Note	00FFH
FF1BH								
FF20H	Port mode register 0	PM0			√	√	–	FFH
FF21H	Port mode register 1	PM1			√	√	–	
FF22H	Port mode register 2	PM2			√	√	–	
FF32H	Pull-up resistor option register B2	PUB2			√	√	–	00H
FF42H	Timer clock selection register 2	TCL2			–	√	–	
FF4AH	Watch timer mode control register	WTM			√	√	–	
FF50H	8-bit compare register 80	CR80		W	–	√	–	Undefined
FF51H	8-bit timer counter 80	TM80		R	–	√	–	00H
FF53H	8-bit timer mode control register 80	TMC80		R/W	√	√	–	
FF54H	8-bit compare register 81	CR81		W	–	√	–	Undefined
FF55H	8-bit timer counter 81	TM81		R	–	√	–	00H
FF57H	8-bit timer mode control register 81	TMC81		R/W	√	√	–	

Note 16-bit access is allowed only with short direct addressing.

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

Address	Special-Function Register (SFR) Name	Symbol	R/W	Number of Bits Manipulated Simultaneously			After Reset
				1 Bit	8 Bits	16 Bits	
FF58H	8-bit compare register 82	CR82	W	–	√	–	Undefined
FF59H	8-bit timer counter 82	TM82	R	–	√	–	00H
FF5BH	8-bit timer mode control register 82	TMC82	R/W	√	√	–	
FF70H	Asynchronous serial interface mode register 00	ASIM00		√	√	–	
FF71H	Asynchronous serial interface status register 00	ASIS00	R	√	√	–	
FF72H	Baud rate generator control register 00	BRGC00	R/W	–	√	–	
FF73H	Transmission shift register 00	TXS00	W	–	√	–	Undefined
	Reception buffer register 00	RXB00	R	–	√	–	FFH
FF80H	A/D converter mode register	ADM	R/W	√	√	–	00H
FF84H	A/D input selection register	ADS		√	√	–	
FFA0H	Multiplier control register 1	MULC1		√	√	–	
FFA1H	10-bit multiplication data register A1	MRA1L	W	–	√	–	Undefined
FFA2H		MRA1H		–	√	–	
FFA3H	10-bit multiplication data register B1	MRB1L		–	√	–	
FFA4H		MRB1H		–	√	–	
FFA5H	20-bit multiplication result register	MUL1LL	R	–	√	–	
FFA6H		MUL1LH		–	√	–	
FFA7H		MUL1HL		–	√	–	
FFA8H	Inverter timer control register 7	TMC7	R/W	√	√	–	00H
FFA9H	Inverter timer mode register 7	TMM7		√	√	–	
FFAAH	Dead time reload register	DTIME		–	√	–	FFH
FFABH	Swapping function register 0	SWP0		–	√	–	**Note**
FFE0H	Interrupt request flag register 0	IF0		√	√	–	00H
FFE1H	Interrupt request flag register 1	IF1		√	√	–	
FFE4H	Interrupt mask flag register 0	MK0		√	√	–	FFH
FFE5H	Interrupt mask flag register 1	MK1		√	√	–	
FFECH	External interrupt mode register 0	INTM0		–	√	–	00H
FFF7H	Pull-up resistor option register 0	PU0		√	√	–	
FFF9H	Watchdog timer mode register	WDTM		√	√	–	
FFFAH	Oscillation stabilization time selection register	OSTS		–	√	–	04H
FFFBH	Processor clock control register	PCC		√	√	–	02H

Note The default differs between read mode and write mode. For details, see **13.2**.

3.3 Instruction Address Addressing

An instruction address is determined by the program counter (PC) contents. The PC contents are normally incremented (+1 for each byte) automatically according to the number of bytes of an instruction to be fetched each time another instruction is executed. When a branch instruction is executed, the branch destination information is set to the PC by the addressing described below to branch control. (For details of each instruction, refer to **78K/0S Series Instructions User's Manual (U11047E)**.)

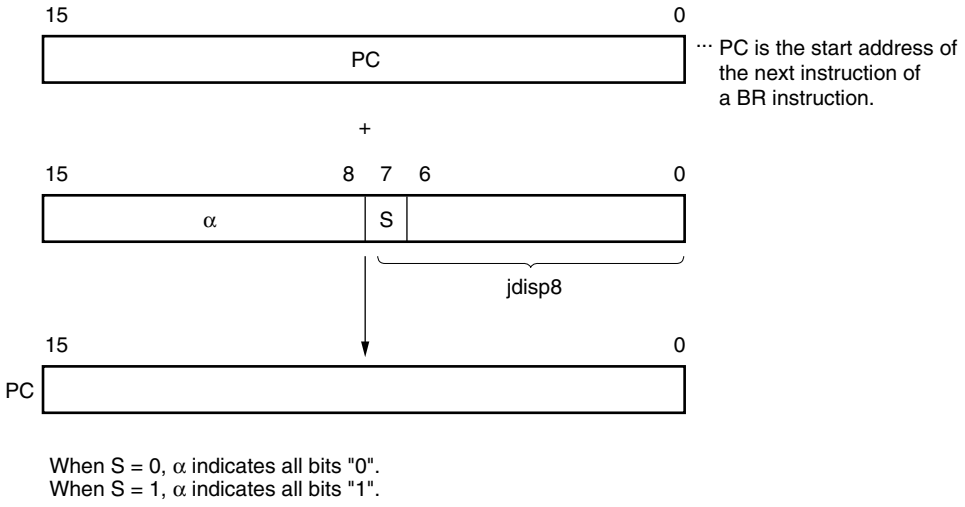
3.3.1 Relative addressing

[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. In other words, the range of branch in relative addressing is between -128 and +127 of the start address of the following instruction.

This function is carried out when the BR $addr16 instruction or a conditional branch instruction is executed.

[Illustration]



When S = 0, α indicates all bits "0".
When S = 1, α indicates all bits "1".

3.3.2 Immediate addressing

[Function]

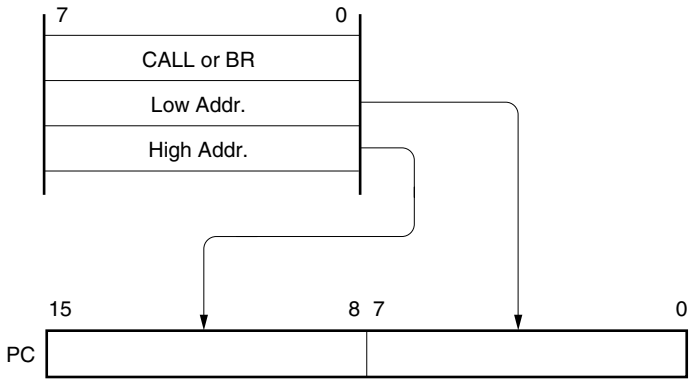
Immediate data in the instruction word is transferred to the program counter (PC) and branched.

This function is carried out when the CALL !addr16 and BR !addr16 instructions are executed.

The CALL !addr16 and BR !addr16 instructions can make a branch to all the memory spaces.

[Illustration]

In case of CALL !addr16 and BR !addr16 instructions



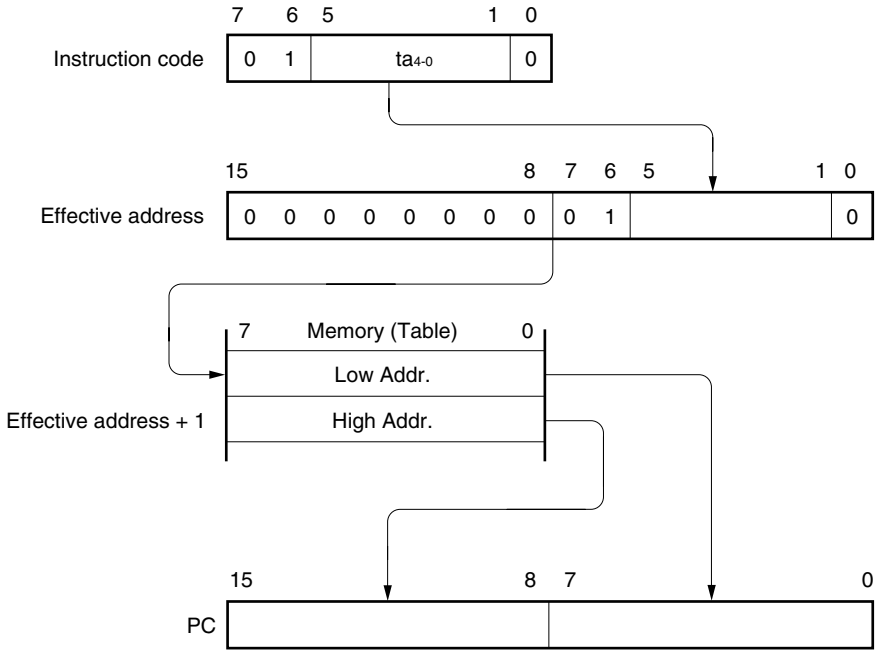
3.3.3 Table indirect addressing

[Function]

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

Table indirect addressing is carried out when the CALLT [addr5] instruction is executed. This instruction can refer to the address stored in the memory table 40H to 7FH and make a branch to all the memory spaces.

[Illustration]



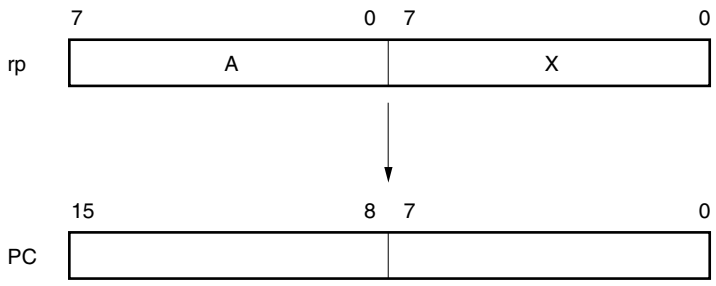
3.3.4 Register addressing

[Function]

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

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

[Illustration]



3.4 Operand Address Addressing

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

3.4.1 Direct addressing

[Function]

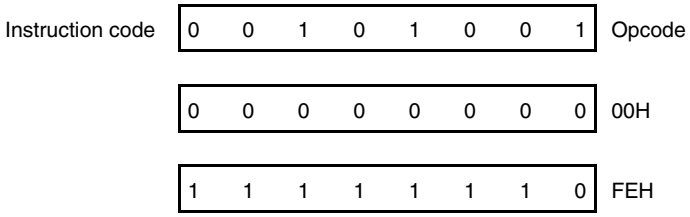
The memory indicated by immediate data in an instruction word is directly addressed.

[Operand format]

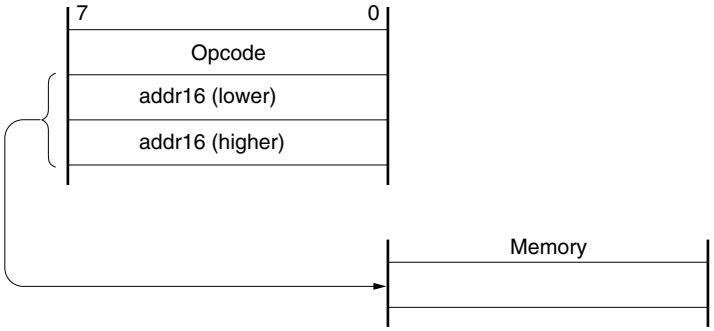
Identifier	Description
addr16	Label or 16-bit immediate data

[Description example]

MOV A, !FE00H; When setting !addr16 to FE00H



[Illustration]



3.4.2 Short direct addressing

[Function]

The memory to be manipulated in the fixed space is directly addressed with 8-bit data in an instruction word.

The fixed space where this addressing is applied to is the 256-byte space FE20H to FF1FH. An internal high-speed RAM and special-function registers (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 a part of the total SFR area. In this area, ports which are frequently accessed in a program and a compare register of the timer are mapped, and these SFRs can be manipulated with a small number of bytes and clocks.

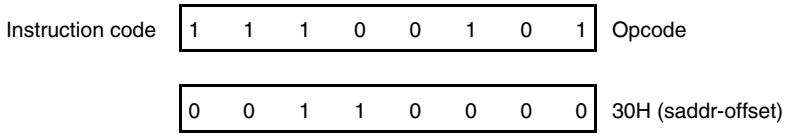
When 8-bit immediate data is at 20H to FFH, bit 8 of an effective address is set to 0. When it is at 00H to 1FH, bit 8 is set to 1. See [Illustration].

[Operand format]

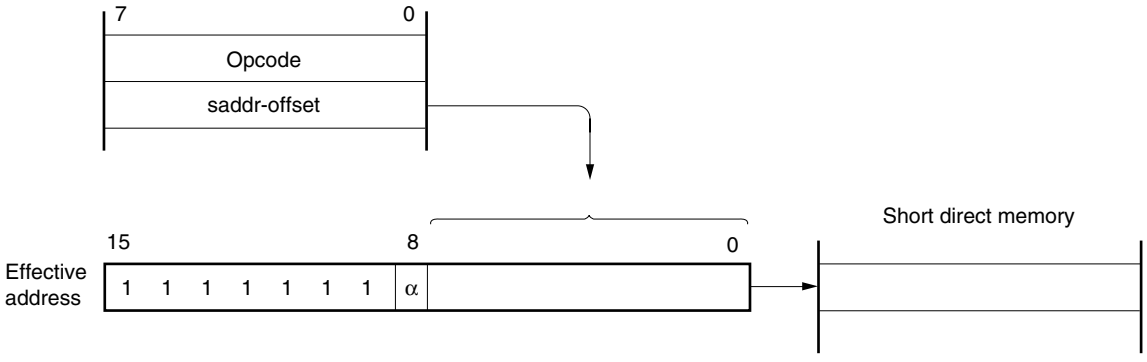
Identifier	Description
saddr	Label or immediate data indicating FE20H to FF1FH
saddrp	Label or immediate data indicating FE20H to FF1FH (even address only)

★ **[Description example]**

MOV FE30H, A; When transferring register A value to saddr (FE30H)



[Illustration]



When 8-bit immediate data is 20H to FFH, $\alpha = 0$.
When 8-bit immediate data is 00H to 1FH, $\alpha = 1$.

3.4.3 Special-function register (SFR) addressing

[Function]

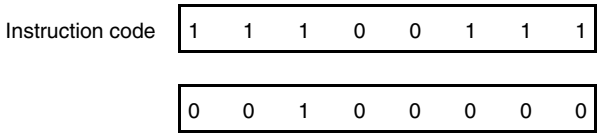
A memory-mapped special-function register (SFR) is addressed with 8-bit immediate data in an instruction word. This addressing is applied to the 240-byte spaces FF00H to FFCFH and FFE0H to FFFFH. However, the SFRs mapped at FF00H to FF1FH can be accessed with short direct addressing.

[Operand format]

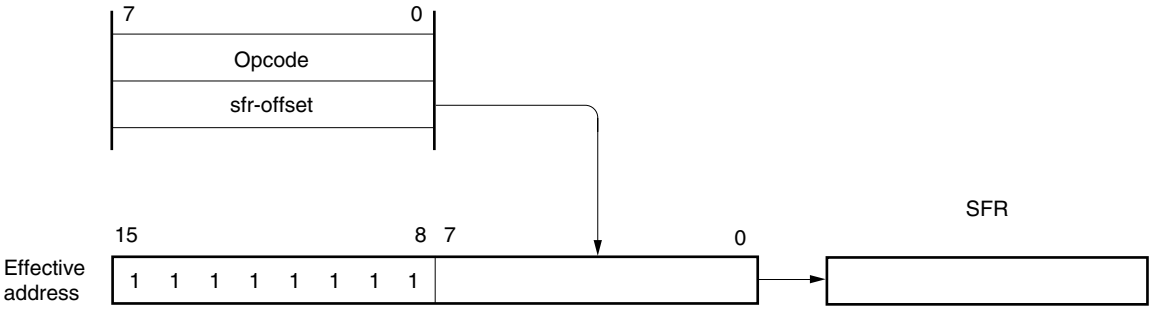
Identifier	Description
sfr	Special-function register name

[Description example]

MOV PM0, A; When selecting PM0 for sfr



[Illustration]



3.4.4 Register addressing

[Function]

A general-purpose register is accessed as an operand.

The general-purpose register to be accessed is specified with the register specify code and functional name in the instruction 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 three bits in the instruction code.

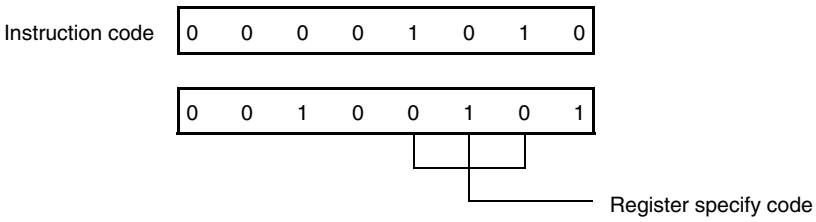
[Operand format]

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

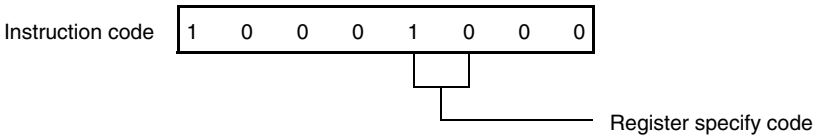
'r' and 'rp' can be described with absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

[Description example]

MOV A, C; When selecting the C register for r



INCW DE; When selecting the DE register pair for rp



3.4.5 Register indirect addressing

[Function]

The memory is addressed with the contents of the register pair specified as an operand. The register pair to be accessed is specified with the register pair specify code in the instruction code. This addressing can be carried out for all the memory spaces.

[Operand format]

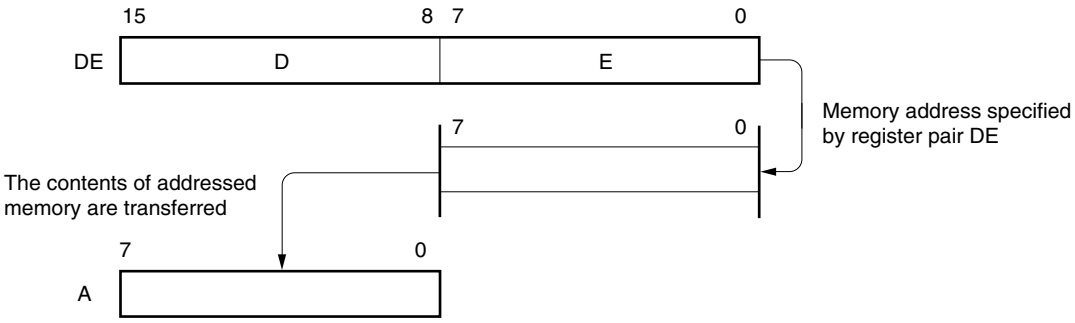
Identifier	Description
–	[DE], [HL]

[Description example]

MOV A, [DE]; When selecting register pair [DE]

Instruction code	0	0	1	0	1	0	1	1

[Illustration]



3.4.6 Based addressing

[Function]

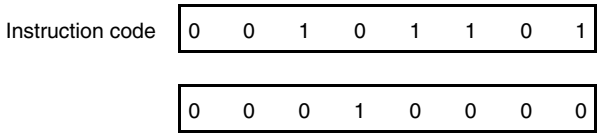
8-bit immediate data is added to the contents of the base register, that is, the HL register pair, and the sum is used to address the memory. Addition is performed by expanding the offset data as a positive number to 16 bits. A carry from the 16th bit is ignored. This addressing can be carried out for all the memory spaces.

[Operand format]

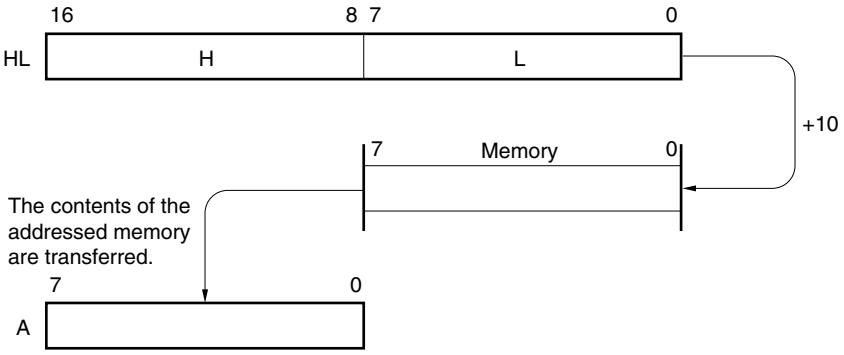
Identifier	Description
–	[HL+byte]

[Description example]

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



★ **[Illustration]**



3.4.7 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 or restored upon generation of an interrupt request.

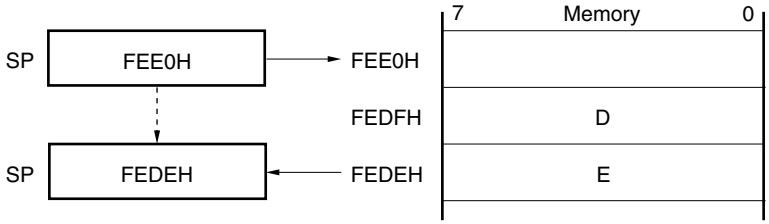
Stack addressing provides access to the internal high-speed RAM area only.

[Description example]

In the case of PUSH DE (saving DE register)

Instruction code	1	0	1	0	1	0	1	0

★ **[Illustration]**



CHAPTER 4 PORT FUNCTIONS

4.1 Port Functions

The μPD789842 Subseries is provided with the ports shown in Figure 4-1. These ports are used to enable several types of control.

Table 4-1 lists the functions of each port.

These ports, while originally designed as digital I/O ports, can also be used for other functions, as summarized in **2.2**.

Figure 4-1. Port Types

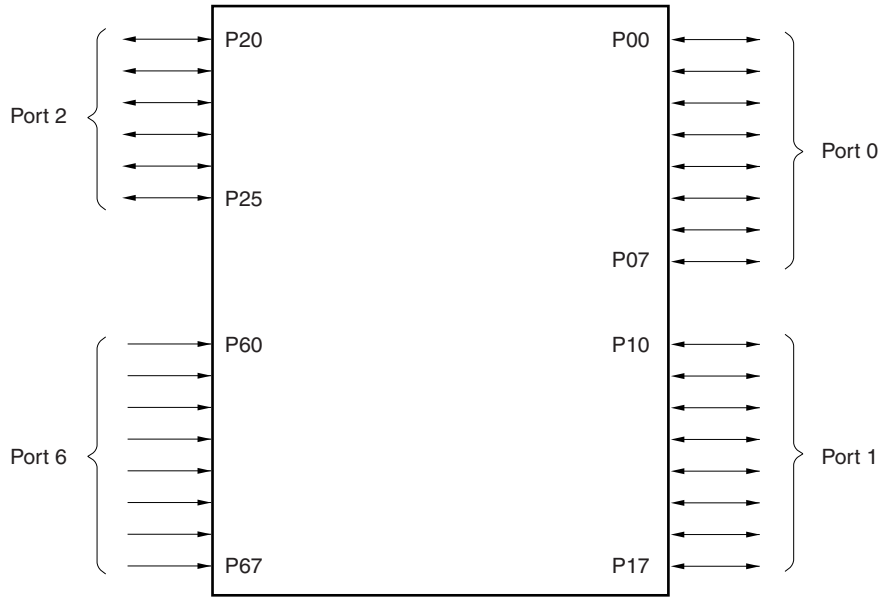


Table 4-1. Port Functions

Name	Pin Name	Function
Port 0	P00 to P07	I/O port. Input/output can be specified in 1-bit units. When used as an input port, use of on-chip pull-up resistors can be specified by pull-up resistor option register 0 (PU0).
Port 1	P10 to P17	I/O port. Input/output can be specified in 1-bit units. When used as an input port, use of on-chip pull-up resistors can be specified by pull-up resistor option register 0 (PU0).
Port 2	P20 to P25	I/O port. Input/output can be specified in 1-bit units. Use of on-chip pull-up resistors can be specified by pull-up resistor option register B2 (PUB2).
Port 6	P60 to P67	Input-only port

4.2 Port Configuration

Ports have the following hardware configuration.

Table 4-2. Configuration of Port

Parameter	Configuration
Control registers	Port mode register (PM0 to PM2)
	Pull-up resistor option register (PU0, PUB2)
Ports	Total: 30 (CMOS I/O: 22, CMOS input: 8)
Pull-up resistors	Total: 22 (software control: 22)

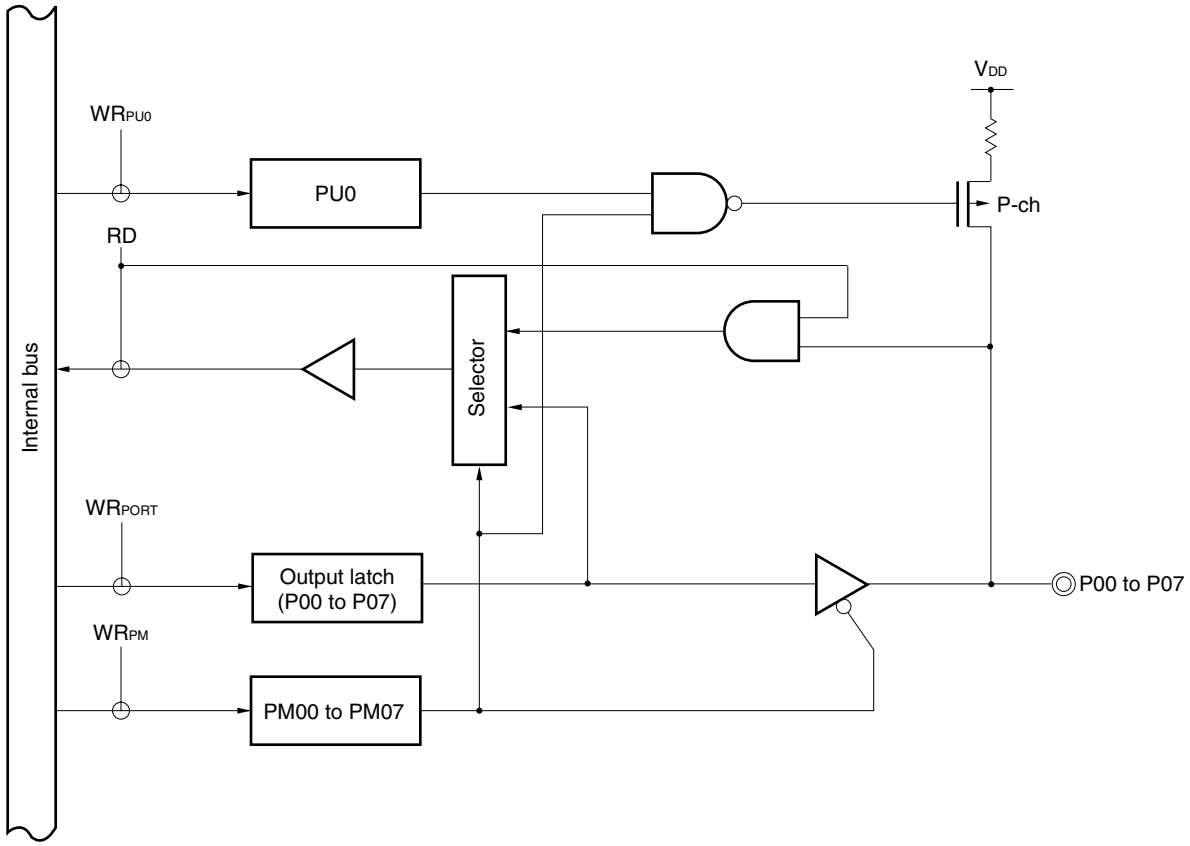
4.2.1 Port 0

This is an 8-bit I/O port with an output latch. Port 0 can be specified in input or output mode in 1-bit units by using port mode register 0 (PM0). When the P00 to P07 pins are used as input port pins, on-chip pull-up resistors can be connected in 8-bit units by using pull-up resistor option register 0 (PU0).

$\overline{\text{RESET}}$ input sets port 0 to input mode.

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

Figure 4-2. Block Diagram of P00 to P07



PU0: Pull-up resistor option register 0
PM: Port mode register
RD: Port 0 read signal
WR: Port 0 write signal

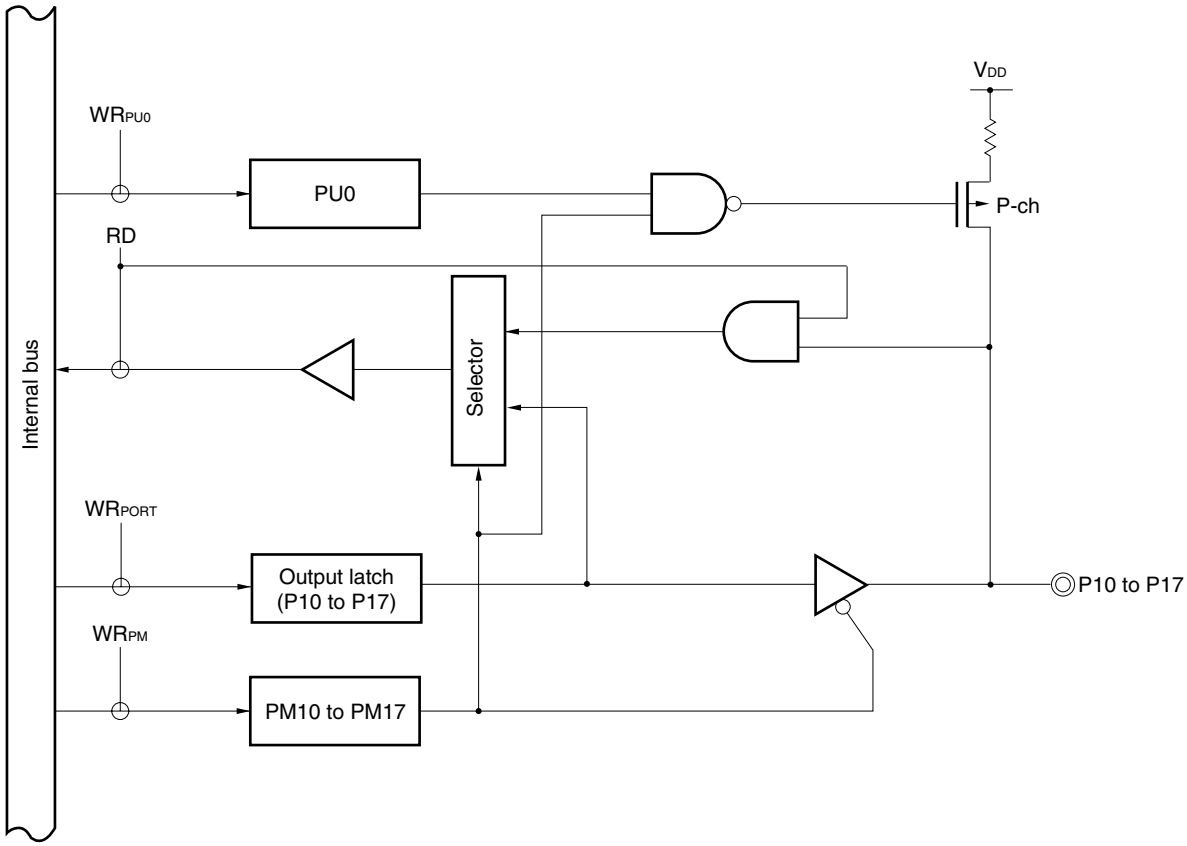
4.2.2 Port 1

This is an 8-bit I/O port with an output latch. Port 1 can be specified in input or output mode in 1-bit units by using port mode register 1 (PM1). When the P10 to P17 pins are used as input port pins, on-chip pull-up resistors can be connected in 8-bit units by using pull-up resistor option register 0 (PU0).

$\overline{\text{RESET}}$ input sets port 1 to input mode.

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

Figure 4-3. Block Diagram of P10 to P17



PU0: Pull-up resistor option register 0
PM: Port mode register
RD: Port 1 read signal
WR: Port 1 write signal

4.2.3 Port 2

This is a 6-bit I/O port with an output latch. Port 2 can be specified in input or output mode in 1-bit units by using port mode register 2 (PM2).

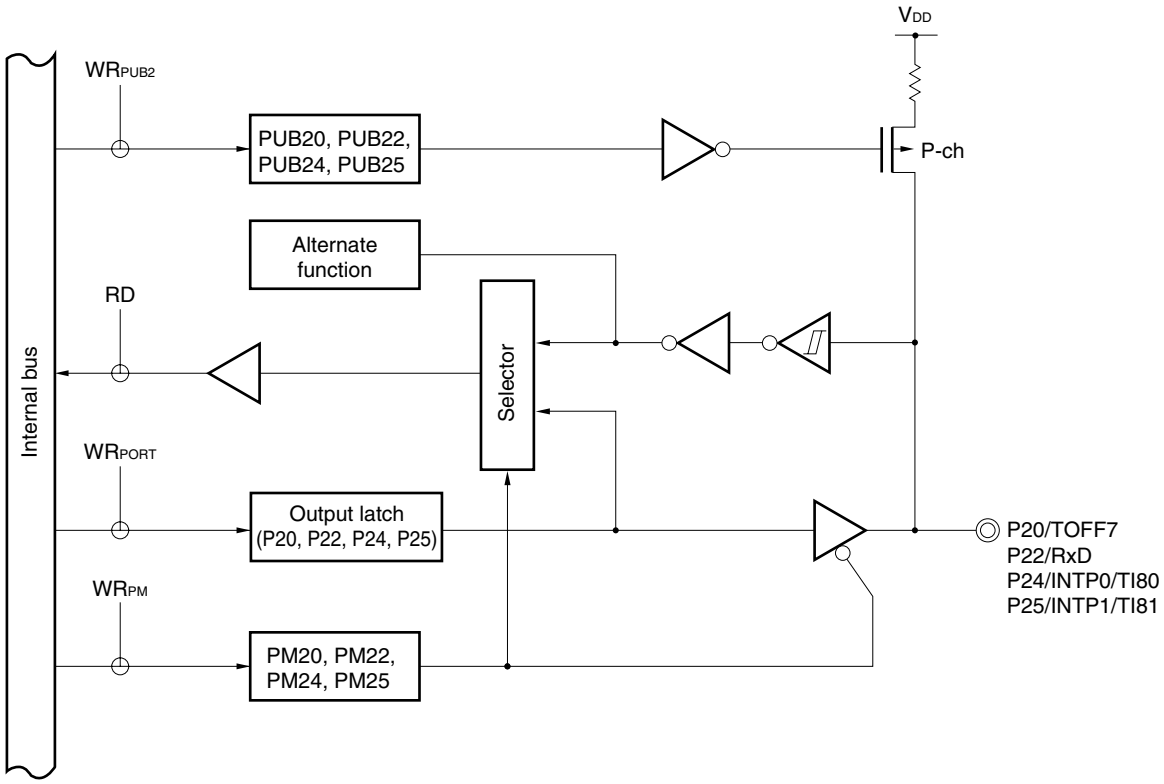
The P20 to P25 pins can be connected to on-chip pull-up resistors in 1-bit units by using pull-up resistor option register B2 (PUB2).

This port is also used as external interrupt inputs, timer I/O, and data I/O to and from the asynchronous serial interface.

$\overline{\text{RESET}}$ input sets port 2 to input mode.

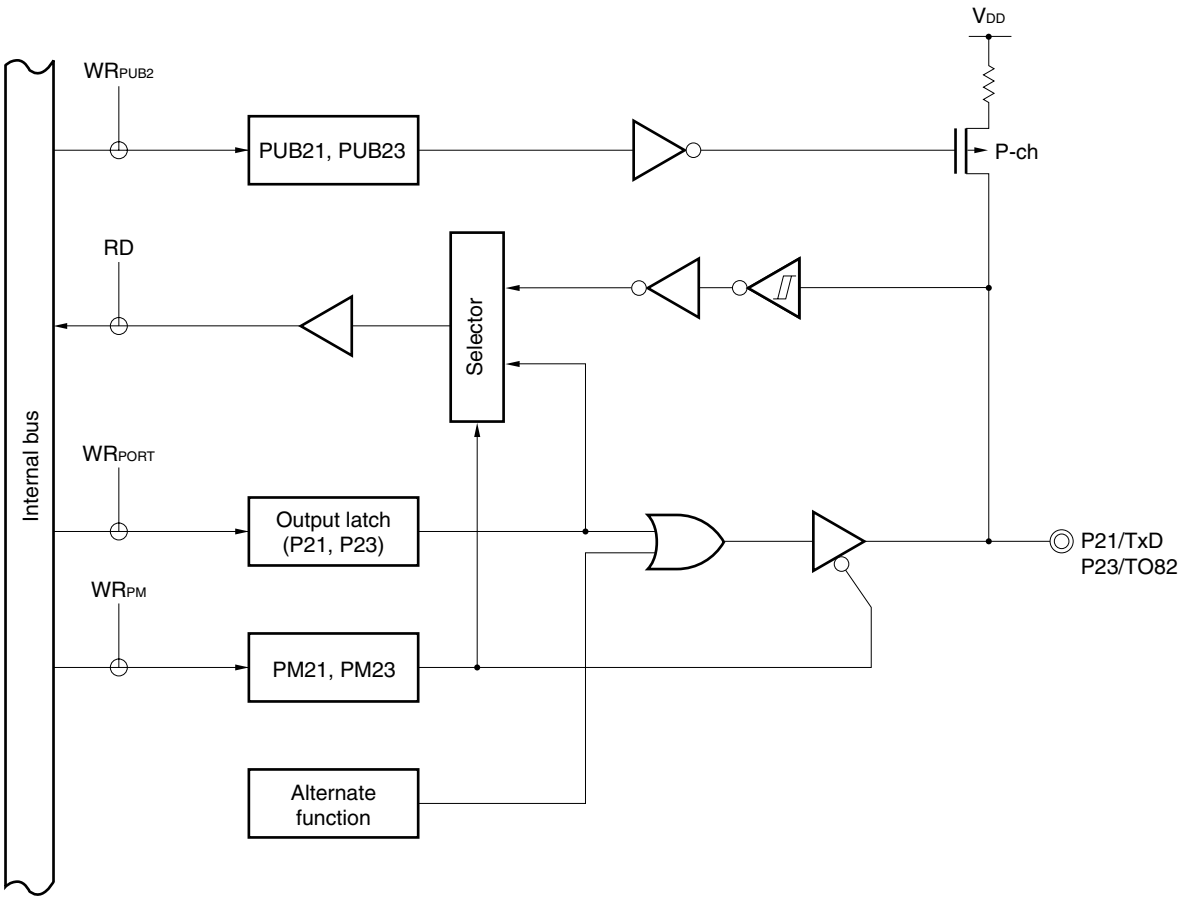
Figures 4-4 and 4-5 show block diagrams of port 2.

Figure 4-4. Block Diagram of P20, P22, P24, and P25



PUB2: Pull-up resistor option register B2
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

Figure 4-5. Block Diagram of P21 and P23



PUB2: Pull-up resistor option register B2
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

4.2.4 Port 6

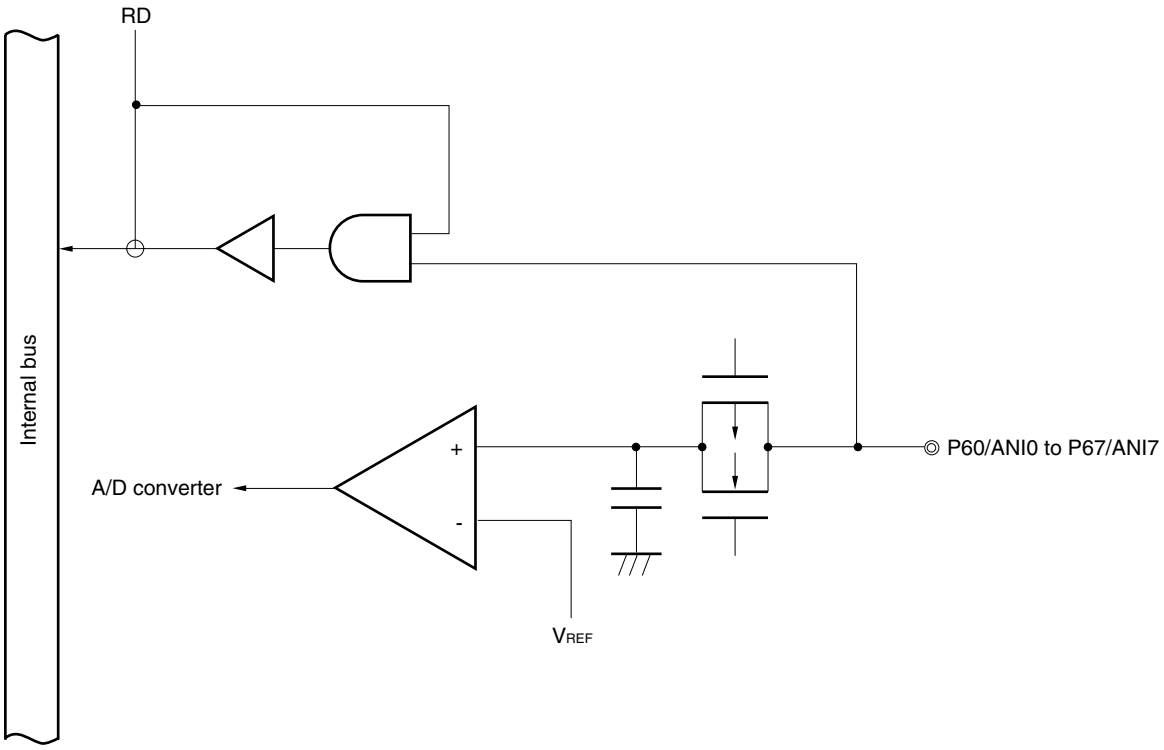
This is an 8-bit input-only port.

This port is also used as analog inputs to the A/D converter.

$\overline{\text{RESET}}$ input sets port 6 to input mode.

Figure 4-6 shows a block diagram of port 6.

Figure 4-6. Block Diagram of P60 to P67



RD: Port 6 read signal

4.3 Port Function Control Registers

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

- Port mode registers (PM0 to PM2)
- Pull-up resistor option register 0 (PU0)
- Pull-up resistor option register B2 (PUB2)

(1) Port mode registers (PM0 to PM2)

The port mode registers separately specify each port bit as input or output.

Each port mode register is set using a 1-bit or 8-bit memory manipulation instruction.

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

When port pins are used for alternate functions, the corresponding port mode register and output latch must be set or reset as described in Table 4-3.

Caution When port 2 is functioning as an output port and its output level is changed, an interrupt request flag is set, because this port is also used as the input for an external interrupt. To use port 2 in output mode, therefore, the interrupt mask flag must be set to 1 in advance.

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

Pin Name	Alternate Function		PM××	P××
	Name	I/O		
P20	TOFF7	Input	1	×
P23	TO82	Output	0	0
P24	INTP0	Input	1	×
	TI80	Input	1	×
P25	INTP1	Input	1	×
	TI81	Input	1	×

Caution When using P21 and P22 as the serial interface, the I/O or output latch must be set according to the function to be used. For details of the setting, see 11.3 (1).

Remark ×: Don't care
PM××: Port mode register
P××: Port output latch

Figure 4-7. Format of Port Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	FF21H	FFH	R/W
PM2	1	1	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W

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

(2) Pull-up resistor option register 0 (PU0)

This register sets whether an on-chip pull-up resistor is used for each of ports 0 and 1. On the port specified to use an on-chip pull-up resistor by PU0, the pull-up resistor can be internally used only for the bits set to input mode. No on-chip pull-up resistors can be used for the bits set to output mode regardless of the setting of PU0. This also applies when the pins are used as alternate-function output pins.

PU0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears PU0 to 00H.

Figure 4-8. Format of Pull-Up Resistor Option Register 0

Symbol	7	6	5	4	3	2	<1>	<0>	Address	After reset	R/W
PU0	0	0	0	0	0	0	PU01	PU00	FFF7H	00H	R/W

PU0m	Pm on-chip pull-up resistor selection (m = 0, 1)
0	On-chip pull-up resistor not connected
1	On-chip pull-up resistor connected

Caution Be sure to clear bits 2 to 7 to 0.

(3) Pull-up resistor option register B2 (PUB2)

This register specifies whether an on-chip pull-up resistor is connected to each pin of port 2. The pin specified by PUB2 to use an on-chip pull-up resistor is connected to the on-chip pull-up resistor regardless of the setting of the port mode register.

PUB2 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears PUB2 to 00H.

Figure 4-9. Format of Pull-Up Resistor Option Register B2

Symbol	7	6	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
PUB2	0	0	PUB25	PUB24	PUB23	PUB22	PUB21	PUB20	FF32H	00H	R/W

PUB2m	P2m on-chip pull-up resistor selection (m = 0 to 5)
0	On-chip pull-up resistor not connected
1	On-chip pull-up resistor connected

Caution Be sure to clear bits 6 and 7 to 0.

4.4 Operation of Port Functions

The operation of a port differs depending on whether the port is set in input or output mode, as described below.

Caution A 1-bit memory manipulation instruction is executed to manipulate 1 bit of a port. However, this instruction accesses the port in 8-bit units. When this instruction is executed to manipulate a bit of an I/O port, therefore, the contents of the output latch of the pin that is set in input mode and not subject to manipulation become undefined.

4.4.1 Writing to I/O port

(1) In output mode

A value can be written to the output latch of a port by using a transfer instruction. The contents of the output latch can be output from the pins of the port.

Once data is written to the output latch, it is retained until new data is written to the output latch.

★ Reset input clears the data of the output latch.

(2) In input mode

A value can be written to the output latch by using a transfer instruction. However, the status of the port pin is not changed because the output buffer is OFF.

Once data is written to the output latch, it is retained until new data is written to the output latch.

4.4.2 Reading from I/O port

(1) In output mode

The contents of the output latch can be read by using a transfer instruction. The contents of the output latch are not changed.

(2) In input mode

The status of a pin can be read by using a transfer instruction. The contents of the output latch are not changed.

4.4.3 Arithmetic operation of I/O port

(1) In output mode

An arithmetic operation can be performed with the contents of the output latch. The result of the operation is written to the output latch. The contents of the output latch are output from the port pins.

Once data is written to the output latch, it is retained until new data is written to the output latch.

★ Reset input clears the data of the output latch.

(2) In input mode

The contents of the output latch become undefined. However, the status of the pin is not changed because the output buffer is off.

CHAPTER 5 CLOCK GENERATOR

5.1 Clock Generator Functions

The clock generator generates the clock to be supplied to the CPU and peripheral hardware. The following system clock oscillator is used.

- **System clock oscillator**

  This circuit oscillates at 8.0 to 8.5 MHz. Oscillation can be stopped by executing the STOP instruction.

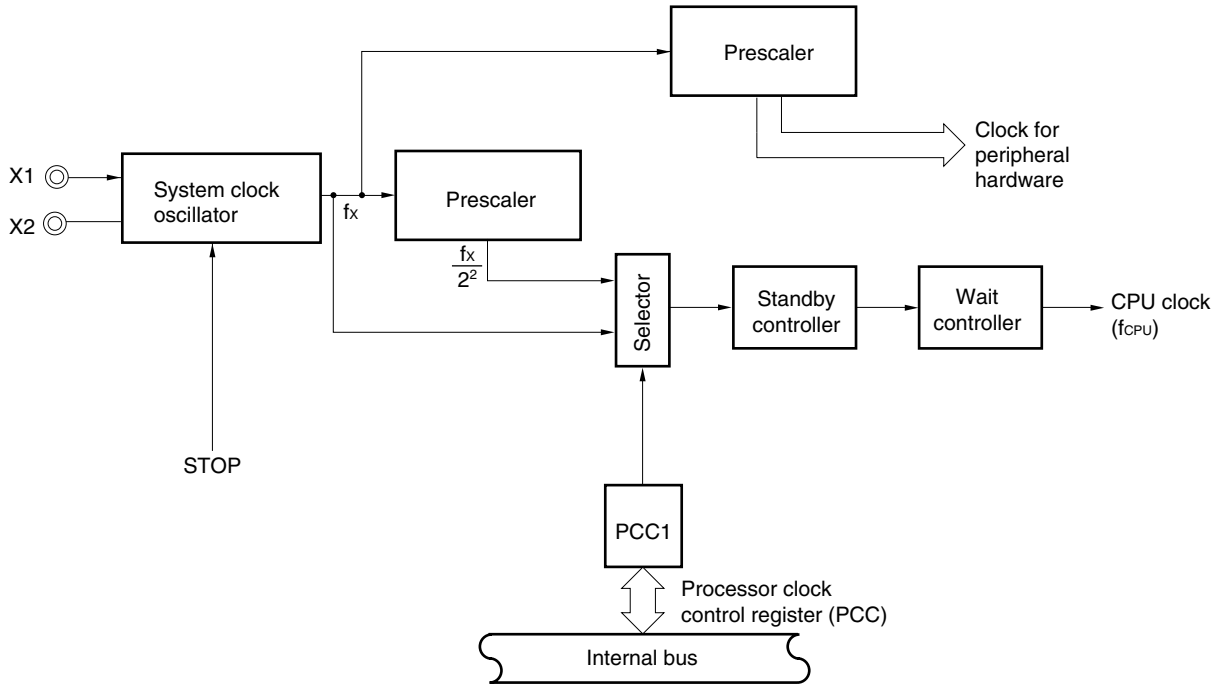
5.2 Clock Generator Configuration

The clock generator consists of the following hardware.

Table 5-1. Configuration of Clock Generator

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

Figure 5-1. Block Diagram of Clock Generator



5.3 Register Controlling Clock Generator

The clock generator is controlled by the following register.

(1) Processor clock control register (PCC)

PCC sets the CPU clock selection and the division ratio.

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

$\overline{\text{RESET}}$ input sets PCC to 02H.

★ **Figure 5-2. Format of Processor Clock Control Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PCC	0	0	0	0	0	0	PCC1	0	FFFBH	02H	R/W

PCC1	CPU clock (f_{CPU}) selection	Minimum instruciton execution time: $2/f_{CPU}$
		Operation at f_X = 8.38 MHz
0	f_X	0.24 μs
1	$f_X/2^2$	0.96 μs

Caution Be sure to clear bits 0 and 2 to 7 to 0.

Remark f_X: System clock oscillation frequency

5.4 System Clock Oscillator

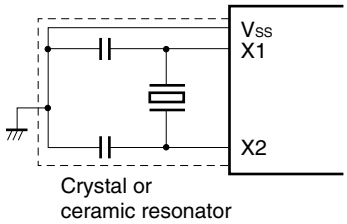
5.4.1 System clock oscillator

The system clock oscillator is oscillated by the crystal or ceramic resonator (8.38 MHz TYP.) connected across the X1 and X2 pins.

Figure 5-3 shows the external circuit of the system clock oscillator.

Figure 5-3. External Circuit of System Clock Oscillator

Crystal or ceramic oscillation



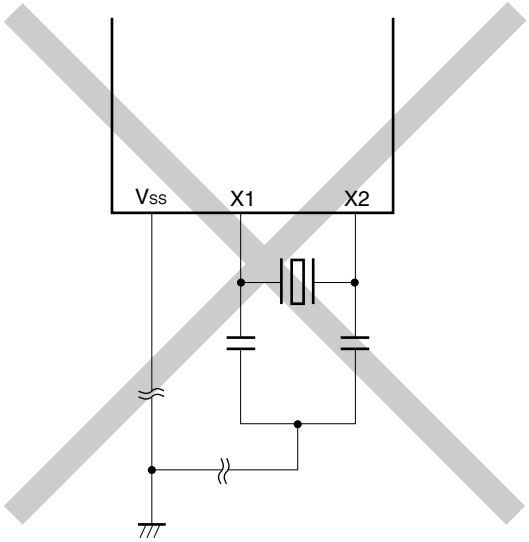
Caution When using the system clock oscillator, wire as follows in the area enclosed by the broken lines in Figure 5-3 to avoid an adverse effect from wiring capacitance.

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with any other signal lines. Do not route the wiring in the vicinity of a line through which a high fluctuating current flows.**
- **Always make the ground of the capacitor of the oscillator the same potential as V_{SS}. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

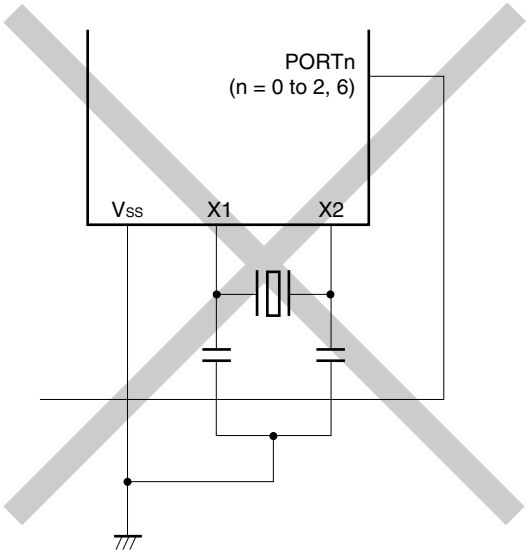
Figure 5-4 shows examples of incorrect resonator connection.

Figure 5-4. Examples of Incorrect Resonator Connection (1/2)

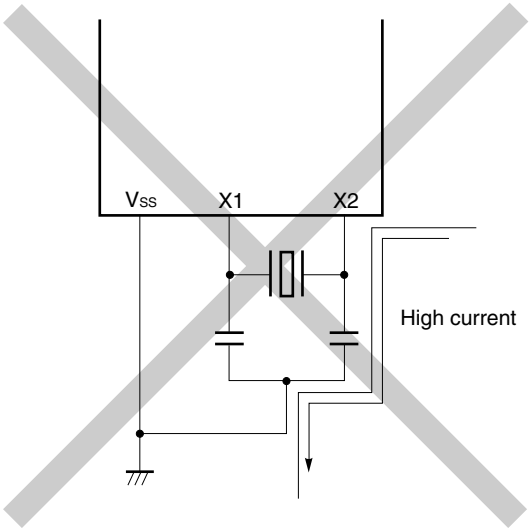
(a) Too long Wiring



(b) Crossed signal line



(c) Wiring near high fluctuating current



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

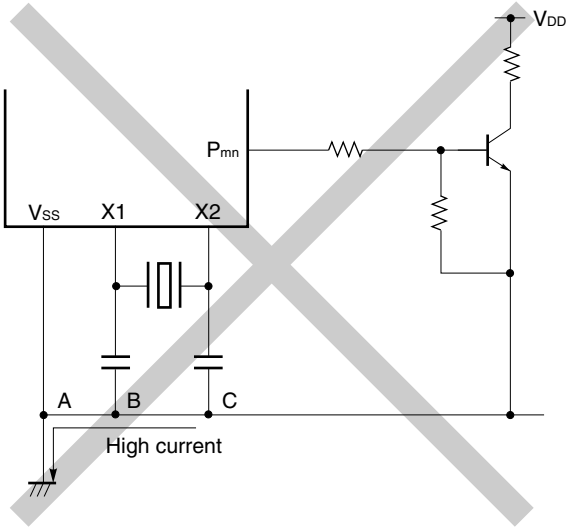
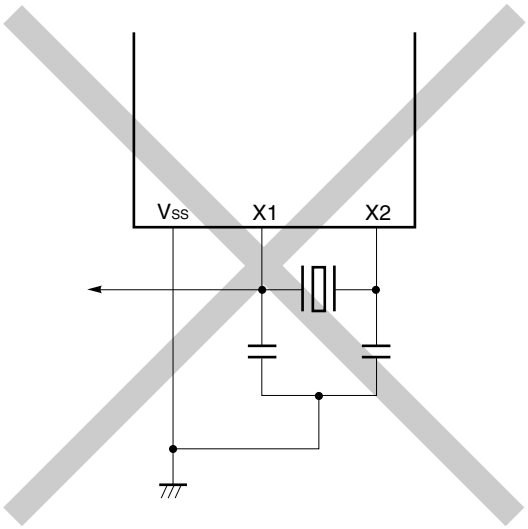


Figure 5-4. Examples of Incorrect Resonator Connection (2/2)

(e) Signals are fetched



5.4.2 Divider

The divider generates clocks by dividing the system clock oscillator output (fx).

5.5 Clock Generator Operation

The clock generator generates the following clocks and controls the operation modes of the CPU, such as standby mode.

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

The operation of the clock generator is determined by the processor clock control register (PCC), as follows.

(a) The slow mode (0.96 μs: at 8.38 MHz operation) of the system clock is selected when the $\overline{\text{RESET}}$ signal is generated (PCC = 02H). While a low level is being input to the $\overline{\text{RESET}}$ pin, oscillation of the system clock is stopped.

★ (b) Two types of minimum instruction execution time (0.24 μs and 0.96 μs: at 8.38 MHz operation) can be selected by setting PCC.

(c) Two standby modes, STOP and HALT, can be used.

(d) The clock pulse for the peripheral hardware is generated by dividing the frequency of the system clock. Therefore, the peripheral hardware stops when the system clock stops.

5.6 Changing Setting of CPU Clock

5.6.1 Time required for switching CPU clock

The CPU clock can be selected by using bit 1 (PCC1) of the processor clock control register (PCC).

Actually, the specified clock is not selected immediately after the setting of PCC has been changed; the old clock is used for the duration of several instructions after that (see **Table 5-2**).

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

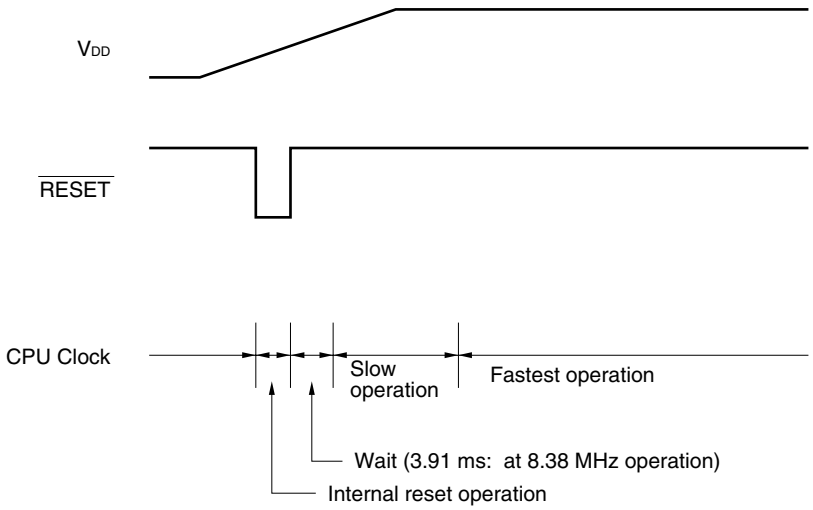
Set Value Before Switching	Set Value After Switching	
PCC1	PCC1	PCC1
	0	1
0		4 clocks
1	2 clocks	

Remark Two clocks is the minimum instruction execution time of the CPU clock before switching.

5.6.2 Switching CPU clock

The following figure illustrates how the CPU clock switches.

Figure 5-5. Switching Between System Clock and CPU Clock



<1> The CPU is reset when the $\overline{\text{RESET}}$ pin is made low on power application. The effect of resetting is released when the $\overline{\text{RESET}}$ pin is later made high, and the system clock starts oscillating. At this time, the time during which oscillation stabilizes (2^{15}/fx) is automatically secured.

After that, the CPU starts instruction execution at the slow speed of the system clock (0.96 μs: at 8.38 MHz operation).

<2> After the time required for the V_{DD} voltage to rise to the level at which the CPU can operate at high speed has elapsed, the processor clock control register (PCC) is rewritten so that high-speed operation can be selected.

CHAPTER 6 10-BIT INVERTER CONTROL TIMER

6.1 10-Bit Inverter Control Timer Functions

The 10-bit inverter control timer is used to control the inverter.

The 10-bit inverter control timer incorporates an 8-bit dead time generation timer and can output a waveform that does not overlap the active level. It can output pulses on six channels, both for the positive phase and negative phase. In addition, it is equipped with an active level change function, and an output off function that is controlled by an external input and the watchdog timer interrupt request input.

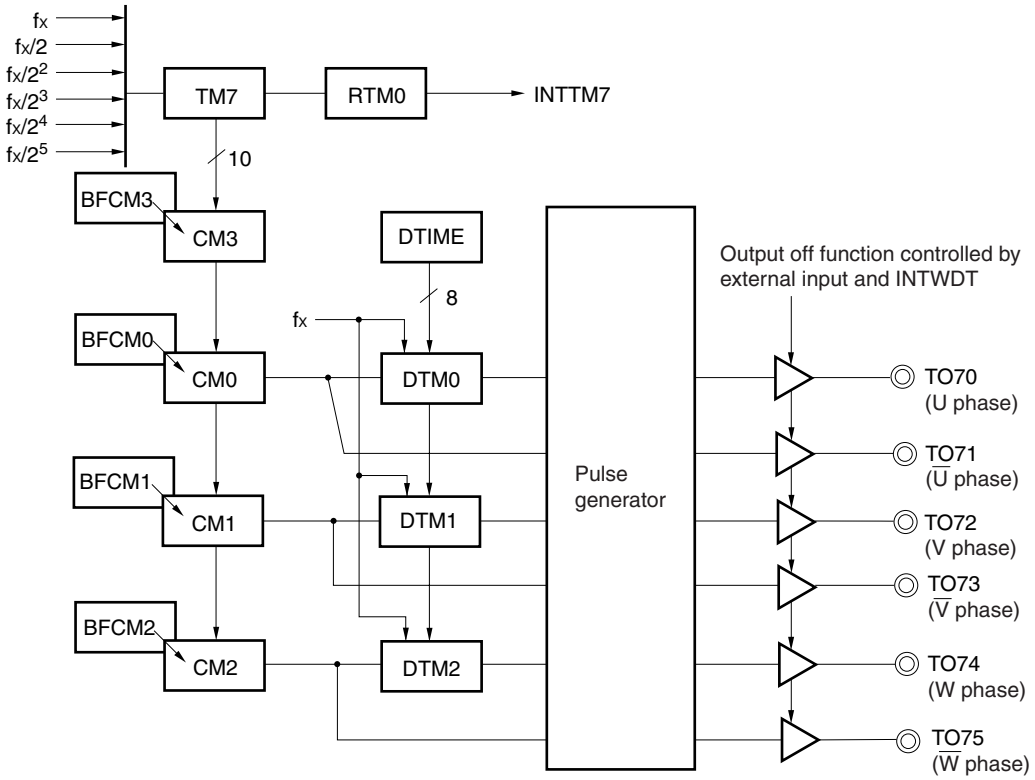
6.2 10-Bit Inverter Control Timer Configuration

The 10-bit inverter control timer consists of the following hardware.

Table 6-1. Configuration of 10-Bit Inverter Control Timer

Item	Configuration
Timer counters	10-bit up/down counter × 1 (TM7) 8-bit down counter × 3 (DTM0 to DTM2) 3-bit up counter × 1 (RTM0)
Registers	10-bit compare register × 4 (CM0 to CM3) 10-bit buffer register × 4 (BFCM0 to BFCM3) 8-bit reload register × 1 (DTIME)
Timer outputs	6 (TO70 to TO75)
Control registers	Inverter timer control register 7 (TMC7) Inverter timer mode register 7 (TMM7)

Figure 6-1. Block Diagram of 10-Bit Inverter Control Timer



(1) 10-bit up/down counter (TM7)

TM7 is a 10-bit up/down counter that counts count pulses.

A count operation is performed in sync with the rising edge of the count clock. Upon being started, TM7 starts counting from 0. When the value of TM7 matches the value preset in compare register 3 (CM3), TM7 stops counting up and starts counting down.

If, while TM7 is counting down, the value reaches 000H, an underflow signal is generated, as is the interrupt request signal INTTM7. In the event of an underflow, TM7 stops counting down and starts counting up.

INTTM7 is normally generated for every underflow. Depending on the settings of the IDEV0 to IDEV2 bits of inverter timer control register 7 (TMC7), however, the generator can be divided.

TM7 can be neither read nor written.

The cycle of TM7 is controlled by CM3.

There are six count clocks: f_X, $f_X/2$, $f_X/2^2$, $f_X/2^3$, $f_X/2^4$, $f_X/2^5$. Any one of these clocks can be selected.

TM7 is cleared to 000H upon the input of $\overline{\text{RESET}}$ or when the CE7 bit of TMC7 is cleared.

(2) 10-bit compare registers 0 to 2 (CM0 to CM2)

CM0 to CM2 are 10-bit compare registers. Normally, their contents are compared with those of TM7 and, when they match, they change the contents of the flip-flop.

Also, each of CM0 to CM2 is provided with a buffer register (BFCM0 to BFCM2). The contents of these buffers are transferred to CM0 to CM2 when the interrupt request signal INTTM7 is generated.

CM0 to CM2 can be written to only while TM7 is stopped.

Set the output timing by writing data to BFCM0 to BFCM2.

CM0 to CM2 are cleared to 000H upon the input of $\overline{\text{RESET}}$ or when the CE7 bit of TMC7 is cleared.

(3) 10-bit compare register 3 (CM3)

CM3 is a 10-bit compare register. It is used to control the maximum value of TM7. When the value of TM7 matches that of CM3 or reaches 0, TM7 stops counting up and starts counting down, or vice versa, at the next count clock.

Also, CM3 is provided with a buffer register (BFCM3). The contents of this buffer are transferred to CM3 when the interrupt request signal INTTM7 is generated.

CM3 can be written to only while TM7 is stopped.

Set the cycle of TM7 by writing data to BFCM3.

The value of CM3 is cleared to 0FFH upon the input of $\overline{\text{RESET}}$.

Do not set CM3 to 000H.

(4) 10-bit buffer registers 0 to 3 (BFCM0 to BFCM3)

BFCM0 to BFCM3 are 10-bit registers. Data is transferred from each buffer register to the corresponding compare register (CM0 to CM3) upon the generation of the interrupt request signal INTTM7.

BFCM0 to BFCM3 can be read/written regardless of whether TM7 is counting or stopped.

Upon the input of $\overline{\text{RESET}}$, BFCM0 to BFCM2 are set to 000H, while BFCM3 is set to 0FFH.

BFCM0 to BFCM3 can be read/written not only in word units but also in byte units. To perform reading/writing in units of less than 8 bits, use BFCM0L to BFCM3L.

(5) Dead time reload register (DTIME)

DTIME is an 8-bit register that is used for setting the dead time. It is used in common by the three dead time timers (DTM0 to DTM2). Note, however, that data is loaded from DTIME into each of DTM0 to DTM2 at a different timing.

DTIME can be written only while TM7 is not counting. Even if an instruction is issued to rewrite the contents of DTIME while counting is being performed, the contents of DTIME will not be changed.

Upon the input of $\overline{\text{RESET}}$, DTIME is set to FFH.

When DTIME is set to 00H, output is performed using the dead time of count clock fx.

(6) Dead time timers 0 to 2 (DTM0 to DTM2)

DTM0 to DTM2 are 8-bit down-counters that are used to generate dead time.

When the values of CM0 to CM2 and TM7 match, the value of the dead time reload register (DTIME) is reloaded into DTM0 to DTM2, which then begin counting down again. When each of DTM0 to DTM2 changes from 00H to FFH, an underflow signal is generated and the counters stop at FFH.

The fx count clock is used.

DTM0 to DTM2 can be neither read nor written.

DTM0 to DTM2 are set to FFH upon the input of $\overline{\text{RESET}}$ or when the CE7 bit of TMC7 is cleared.

(7) Buffer transmission control timer (RTM0)

RTM0 is a 3-bit up-counter. It incorporates a function for dividing the interrupt request signal INTTM7.

RTM0 is incremented when TM7 issues an underflow signal. When the value of RTM0 matches that of the division count set in bits IDEV0 to IDEV2 of TMC7, INTTM7 is generated. RTM0 can be neither read nor written.

RTM0 is set to 7H upon the input of $\overline{\text{RESET}}$. RTM0 is also set to 7H upon the issuance of INTTM7, as well as when the CE7 bit of TMC7 is cleared.

6.3 Registers Controlling 10-Bit Inverter Control Timer

The following two registers control the 10-bit inverter control timer.

- Inverter timer control register 7 (TMC7)
- Inverter timer mode register 7 (TMM7)

(1) Inverter timer control register 7 (TMC7)

Inverter timer control register 7 (TMC7) is used to control the operation of TM7, DTM0 to DTM2, and RTM0. It is also used to select the count clock used by the 10-bit inverter control timer, and to select the compare register transfer cycle.

TMC7 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TMC7 to 00H.

Figure 6-2. Format of Inverter Timer Control Register 7

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
TMC7	CE7	0	TCL72	TCL71	TCL70	IDEV2	IDEV1	IDEV0	FFA8H	00H	R/W

CE7	Control of TM7, DTM0 to DTM2, RTM0
0	Clear and stop (TO70 to TO75 are Hi-Z)
1	Count enabled

TCL72	TCL71	TCL70	Count clock selection
0	0	0	f_X (8.38 MHz)
0	0	1	$f_X/2$ (4.19 MHz)
0	1	0	$f_X/2^2$ (2.1 MHz)
0	1	1	$f_X/2^3$ (1.05 MHz)
1	0	0	$f_X/2^4$ (524 kHz)
1	0	1	$f_X/2^5$ (262 kHz)
Other than above			Setting prohibited

IDEV2	IDEV1	IDEV0	Selection of INTTM7 generation frequency
0	0	0	Generated upon each TM7 underflow (every time)
0	0	1	Generated upon every second TM7 underflow
0	1	0	Generated upon every third TM7 underflow
0	1	1	Generated upon every fourth TM7 underflow
1	0	0	Generated upon every fifth TM7 underflow
1	0	1	Generated upon every sixth TM7 underflow
1	1	0	Generated upon every seventh TM7 underflow
1	1	1	Generated upon every eighth TM7 underflow

Remarks 1. f_X: System clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 8.38 MHz.

(2) Inverter timer mode register 7 (TMM7)

Inverter timer mode register 7 (TMM7) is used to specify the active level of outputs TO70 to TO75, control the operation, and set the valid edge of TOFF7.

TMM7 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TMM7 to 00H.

Figure 6-3. Format of Inverter Timer Mode Register 7

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TMM7	0	0	0	PNOFFB	ALV	TOEDG	TOSPP	TOSPW	FFA9H	00H	R/W

PNOFFB[Note]	Flag indicating status of TM7 output to TO70 to TO75
0	TM7 output disabled status (TO70 to TO75 are Hi-Z)
1	TM7 output enabled status

ALV	Specification of active level of TO70 to TO75 output
0	Low level
1	High level

TOEDG	Specification of valid edge of TOFF7
0	Falling edge
1	Rising edge

TOSPP	Control of TO70 to TO75 output stop at Valid edge
0	Output is not stopped
1	Output is stopped (TO70 to TO75 are Hi-Z)

TOSPW	Control of TO70 to TO75 output stop according to INTWDT
0	Output is not stopped
1	Output is stopped (TO70 to TO75 are Hi-Z)

Note The PNOFFB bit is used as a flag for reading only. It can neither be set nor reset by software.
When TM7 is stopped (CE7 = 0), or operating (CE7 = 1), PNOFFB is reset when output is stopped by either TOFF7 or INTWDT.

Caution Be sure to clear bits 5 to 7 to 0.

Remarks 1. TO70 to TO75 enter the Hi-Z status in the following cases. Note, however, that if CE7 = 1, timers TM7, DTM0 to DTM2, and RTM0 do not stop.

- When TOSPP = 1, and the valid edge is input to the TOFF7 pin
- When TOSPW = 1, and the specified interrupt request is generated

To restore the outputs of TO70 to TO75, apply the following procedure.

<1> Write 0 to CE7 and then stop each timer.
<2> Write 0 to the flags of those functions whose output is stopped
<3> Re-set each of the registers to their default values.

2. PNOFFB, ALV, CE7, and TO70 to TO75 are related as follows.

PNOFFFB	ALV	CE7	TO70, TO72, TO74	TO71, TO73, TO75
Other than below			Hi-Z	Hi-Z
1	0/1	1	PWM waveform cycle (positive phase)	PWM waveform cycle (negative phase)

6.4 10-Bit Inverter Control Timer Operation

(1) Setting procedure

<1> Using bits TCL70 to TCL72 of inverter timer control register 7 (TMC7), set the count clock of TM7. Using bits IDEV0 to IDEV2, set the frequency at which the interrupt request signal INTTM7 is generated.

<2> Using the ALV bit of inverter timer mode register 7 (TMM7), set the active level for pins TO70 to TO75.

<3> Set the half-cycle width of the first PWM cycle in CM3.
- PWM frequency = CM3 value $\times$ 2 $\times$ TM7 clock rate
  (The TM7 clock rate is set using TMC7.)

<4> Set the half-cycle width of the second PWM cycle in 10-bit buffer register 3 (BFCM3).

<5> Set the dead time width in the dead time reload register (DTIME).
- Dead time width = (DTIME+1) $\times$ fx
  fx: internal system clock

<6> Set the flip-flop set/reset timing used for the first cycle in CM0 to CM2.

<7> Set the flip-flop set/reset timing used for the second cycle in BFCM0 to BFCM2.

<8> Set the CE7 bit of TMC7 to 1. This enables the operation of TM7, dead time timers 0 to 2 (DTM0 to DTM2), and the buffer transmission control timer (RTM0).

Caution A bit manipulation instruction must be used to set the CE7 bit.

<9> While TM7 is operating, set the half-cycle width of the next PWM cycle in BFCM3.

<10> While TM7 is operating, set the flip-flop set/reset timing used for the next frequency in BFCM0 to BFCM2.

<11> When the operation of TM7 is stopped, clear the CE7 bit of TMC7 to 0.

Caution Note that it is not possible to change the value of another bit while the CE7 bit is being set.

(2) Correspondence between of output waveform width and set value

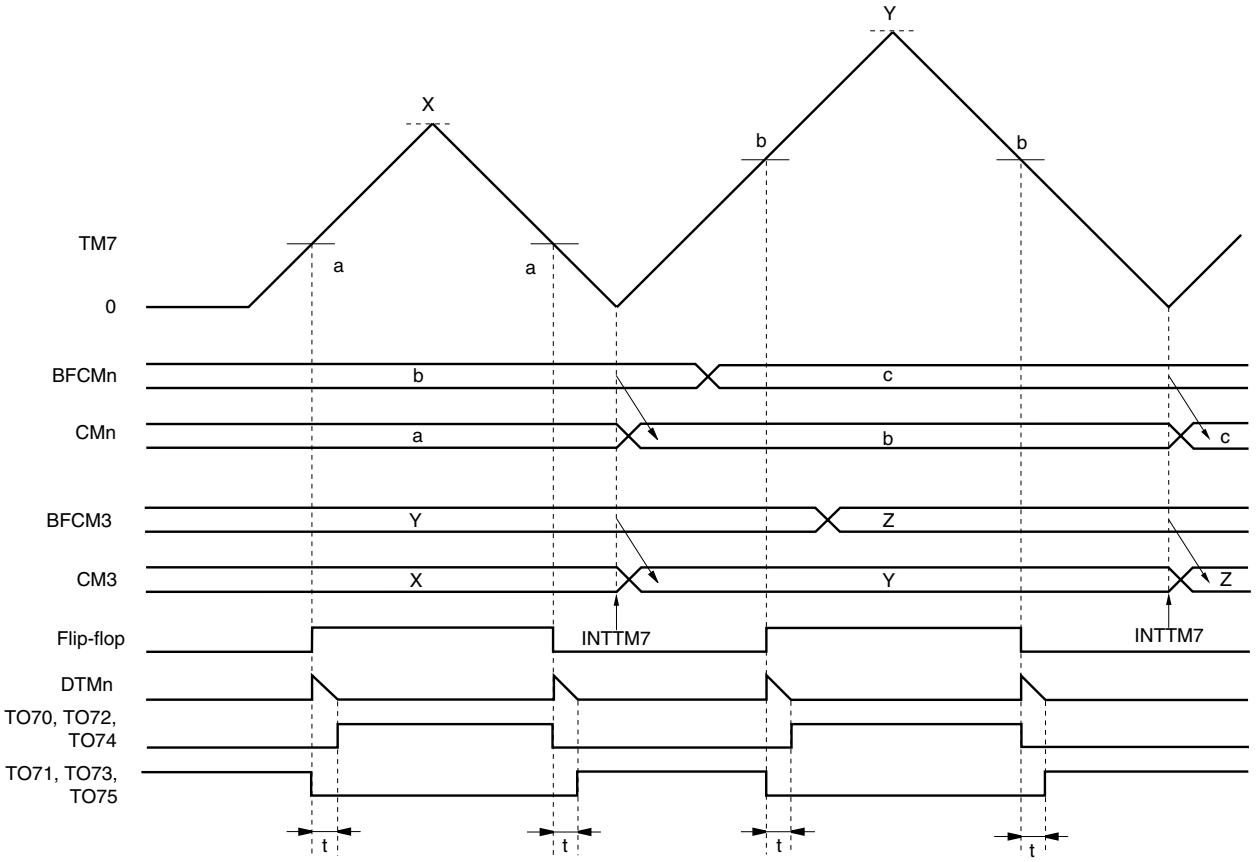
- PWM cycle = CM3 × 2 × T_{TM7}
- Dead time width T_{DTM} = (DTIME + 1) × fx
- Active width of positive phase (TO70, TO72, TO74 pins)
  = { (CM3 - CM_{up}) + (CM3 - CM_{down}) } × T_{TM7} - T_{DTM}
- Active width of negative phase (TO71, TO73, TO75 pins)
  = (CM_{down} + CM_{up}) × T_{TM7} - T_{DTM}

fx: System clock oscillation frequency
T_{TM7}: TM7 count clock
CM_{up}: Value set in CM0 to CM2 when TM7 is counting up.
CM_{down}: Value set in CM0 to CM2 when TM7 is counting down.

Caution When "0" or "minus" is set for the active width of the positive or negative phase, TO70 to TO75 output a fixed inactive level waveform with an active width of "0".

(3) Operation timing

Figure 6-4. TM7 Operation Timing (Basic Operation)



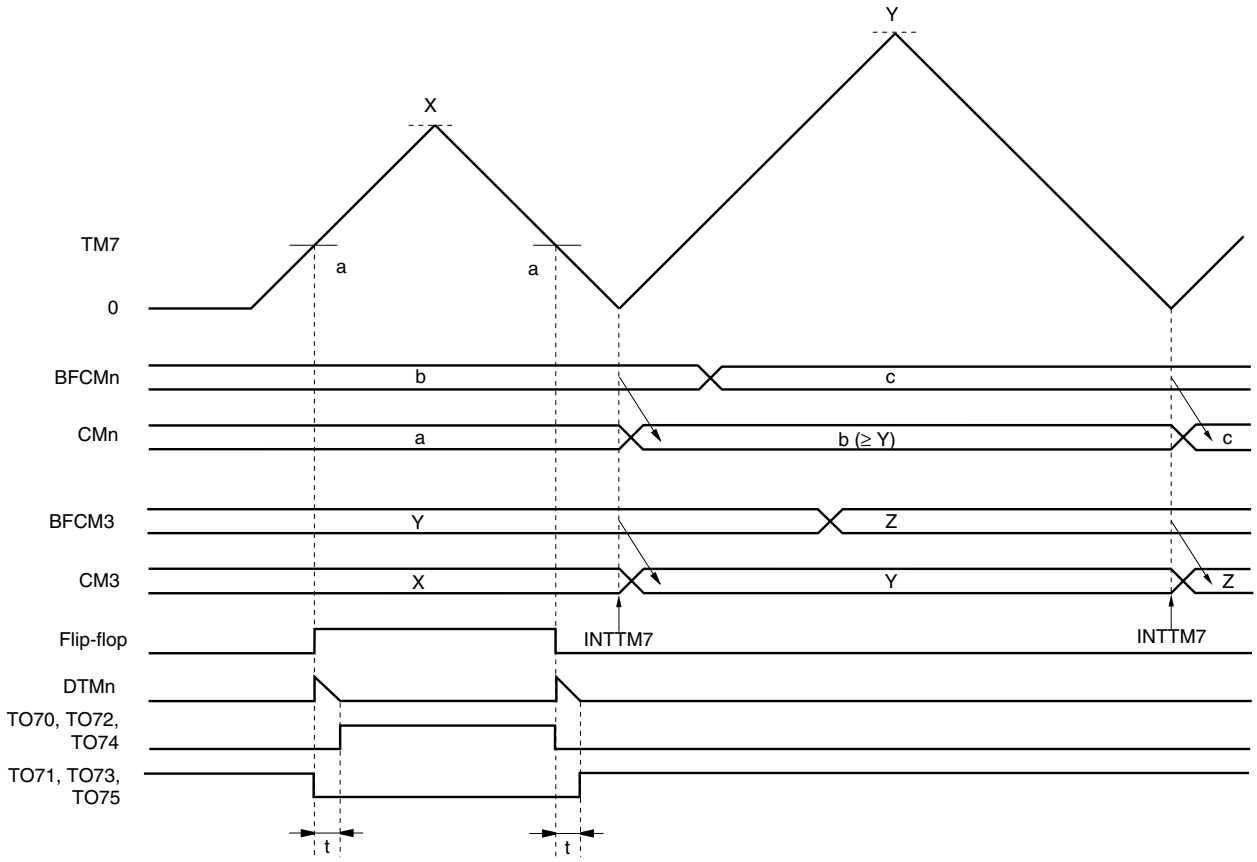
Remarks 1. n = 0 to 2

2. t: Dead time = (DTIME + 1) × fx

(fx: System clock oscillation frequency)

3. The above diagram shows the active-high state when the generation of INTTM7 is not divided.

Figure 6-5. TM7 Operation Timing (CMn (BFCMn) ≥ CM3 (BFCM3))



Remarks 1. n = 0 to 2

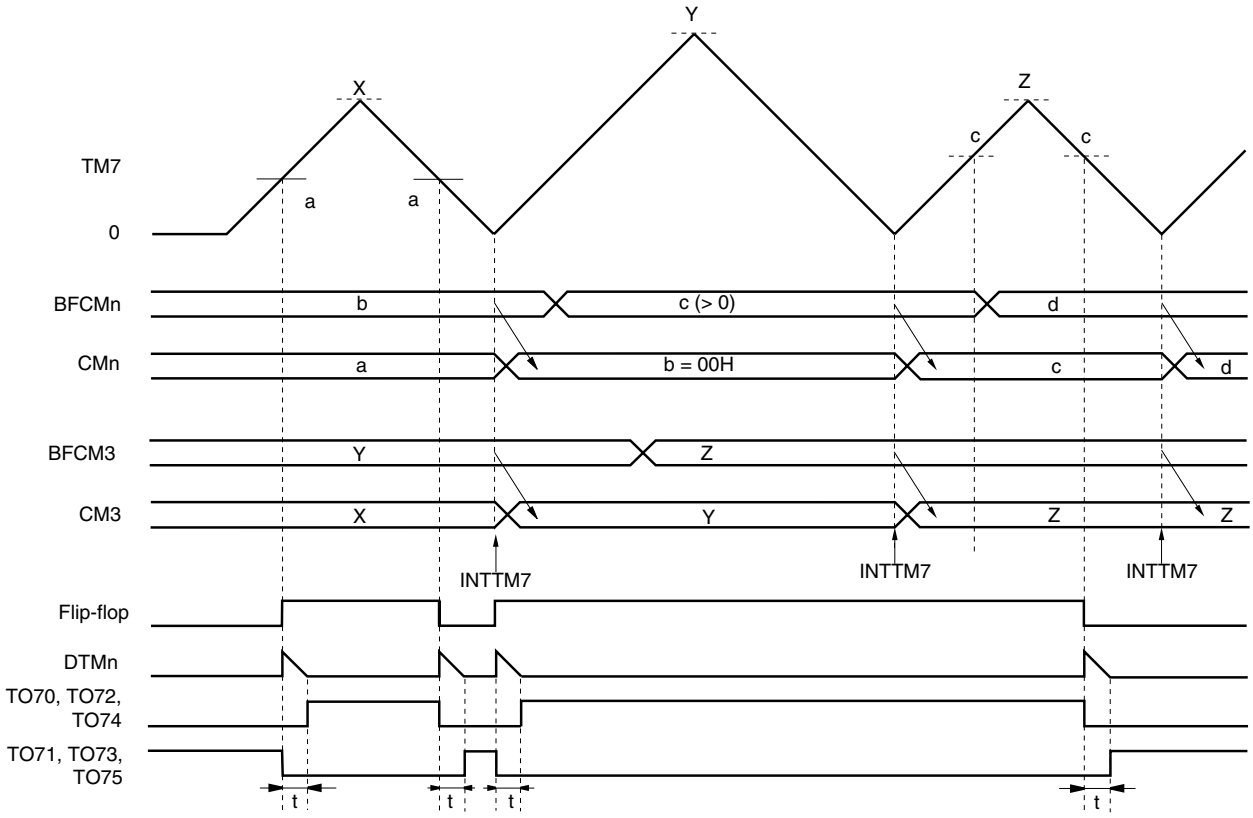
2. t: Dead time = (DTIME + 1) × fx

(fx: System clock oscillation frequency)

3. The above diagram illustrates the active-high state when the generation of INTTM7 is not divided.

When BFCMn is set to a larger value than that set in CM3, a low level is output in the positive phase (TO70, TO72, and TO74 pins), while a high level is output in the negative phase (TO71, TO73, and TO75 pins). This setting is effective for inverter control, when you wish to output a low width or high width that exceeds the PWM period.

Figure 6-6. TM7 Operation Timing (CMn (BFCMn) = 000H)



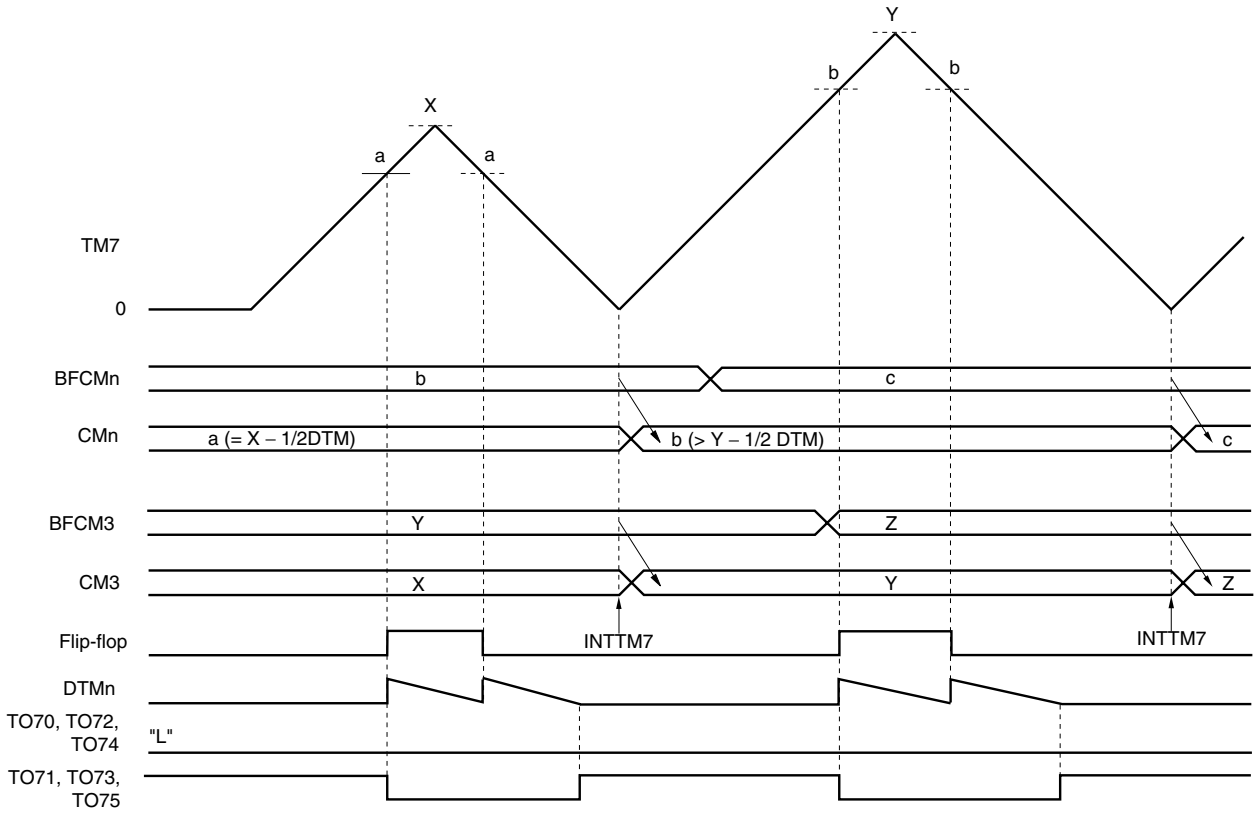
Remarks 1. n = 0 to 2

2. t: Dead time = (DTIME + 1) × fx

(fx: System clock oscillation frequency)

3. The above diagram illustrates the active-high state when the generation of INTTM7 is not divided.

Figure 6-7. TM7 Operation Timing (CMn (BFCMn) = CM3 - 1/2DTM, CMn (BFCMn) > CM3 - 1/2DTM)



Remarks 1. n = 0 to 2

2. The above diagram illustrates the active-high state, when the generation of INTTM7 is not divided.

CHAPTER 7 8-BIT TIMER/EVENT COUNTERS 80, 81, 82

7.1 Functions of 8-Bit Timer/Event Counters 80, 81, 82

The 8-bit timer/event counters (TM80, TM81, TM82) have the following functions.

- Interval timer (TM80, TM81, TM82)
- External event counter (TM80, TM81 only)
- Square-wave output (TM82 only)

The μPD789842 Subseries features a built-in 2-channel 8-bit timer/event counter (TM80, TM81) and a single-channel 8-bit timer (TM82). For TM82, therefore, replace all references to "timer event counter" with "timer."

(1) 8-bit interval timer

When the 8-bit timer/event counter is used as an interval timer, it generates an interrupt at any preset time interval.

Table 7-1. Interval Time of 8-Bit Timer/Event Counter 80

Minimum Interval Time	Maximum Interval Time	Resolution
2^6/fx (7.64 μs)	2^{14}/fx (1.96 ms)	2^6/fx (7.64 μs)
2^9/fx (61.1 μs)	2^{17}/fx (15.6 ms)	2^9/fx (61.1 μs)

Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

Table 7-2. Interval Time of 8-Bit Timer/Event Counter 81

Minimum Interval Time	Maximum Interval Time	Resolution
2^4/fx (1.91 μs)	2^{12}/fx (0.49 ms)	2^4/fx (1.91 μs)
2^8/fx (30.5 μs)	2^{16}/fx (7.82 ms)	2^8/fx (30.5 μs)

Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

Table 7-3. Interval Time of 8-Bit Timer 82

Minimum Interval Time	Maximum Interval Time	Resolution
2^3/fx (0.95 μs)	2^{11}/fx (0.24 ms)	2^3/fx (0.95 μs)
2^7/fx (15.3 μs)	2^{15}/fx (3.91 ms)	2^7/fx (15.3 μs)
2^{10}/fx (0.12 ms)	2^{18}/fx (31.3 ms)	2^{10}/fx (0.12 ms)

Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

(2) External event counter

The number of pulses of an externally input signal can be measured.

(3) Square-wave output

A square wave of arbitrary frequency can be output.

Table 7-4. Square-Wave Output Range of 8-Bit Timer 82

Minimum Interval Time	Maximum Interval Time	Resolution
2^3/fx (0.95 μs)	2^{11}/fx (0.24 ms)	2^3/fx (0.95 μs)
2^7/fx (15.3 μs)	2^{15}/fx (3.91 ms)	2^7/fx (15.3 μs)
2^{10}/fx (0.12 ms)	2^{18}/fx (31.3 ms)	2^{10}/fx (0.12 ms)

Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

7.2 Configuration of 8-Bit Timer/Event Counters 80, 81, 82

8-bit timer/event counters 80, 81, and 82 consist of the following hardware.

Table 7-5. Configuration of 8-Bit Timer/Event Counters 80, 81, 82

Item	Configuration
Timer counter	8 bits × 3 (TM80 to TM82)
Register	Compare register: 8 bits × 3 (CR80 to CR82)
Timer output	1 (TO82)
Control registers	8-bit timer mode control registers 80 to 82 (TMC80 to TMC82) Port mode register 2 (PM2) Port register 2 (P2)

Figure 7-1. Block Diagram of 8-Bit Timer/Event Counter 80

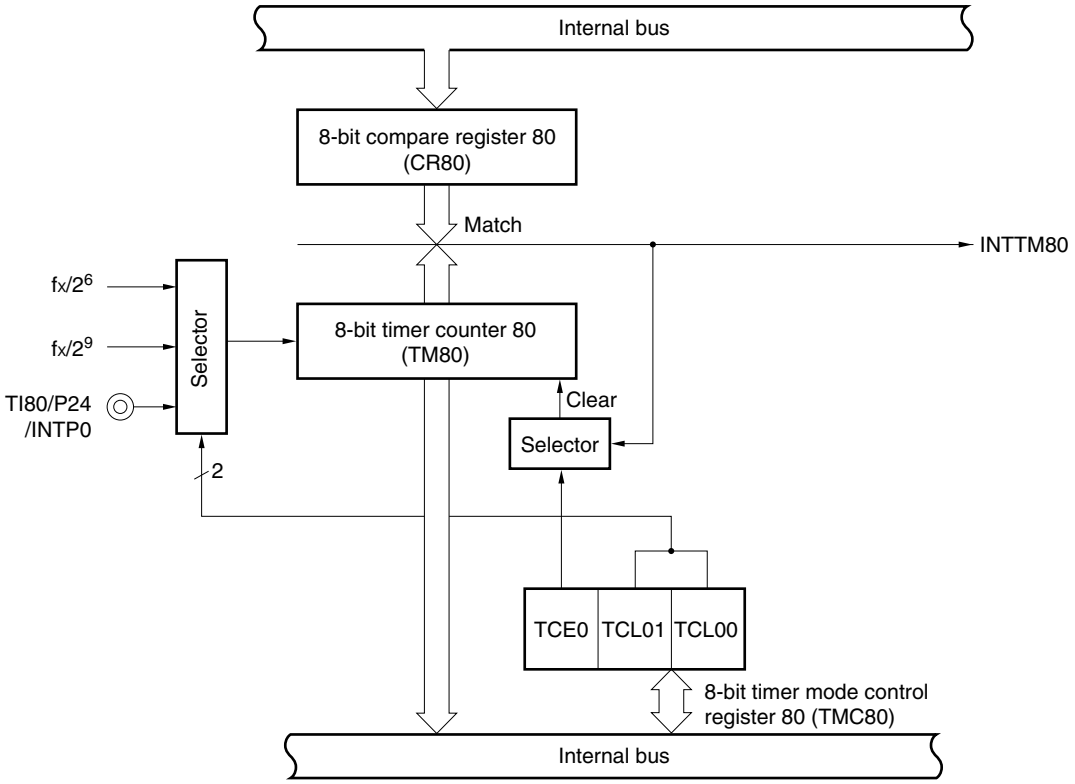


Figure 7-2. Block Diagram of 8-Bit Timer/Event Counter 81

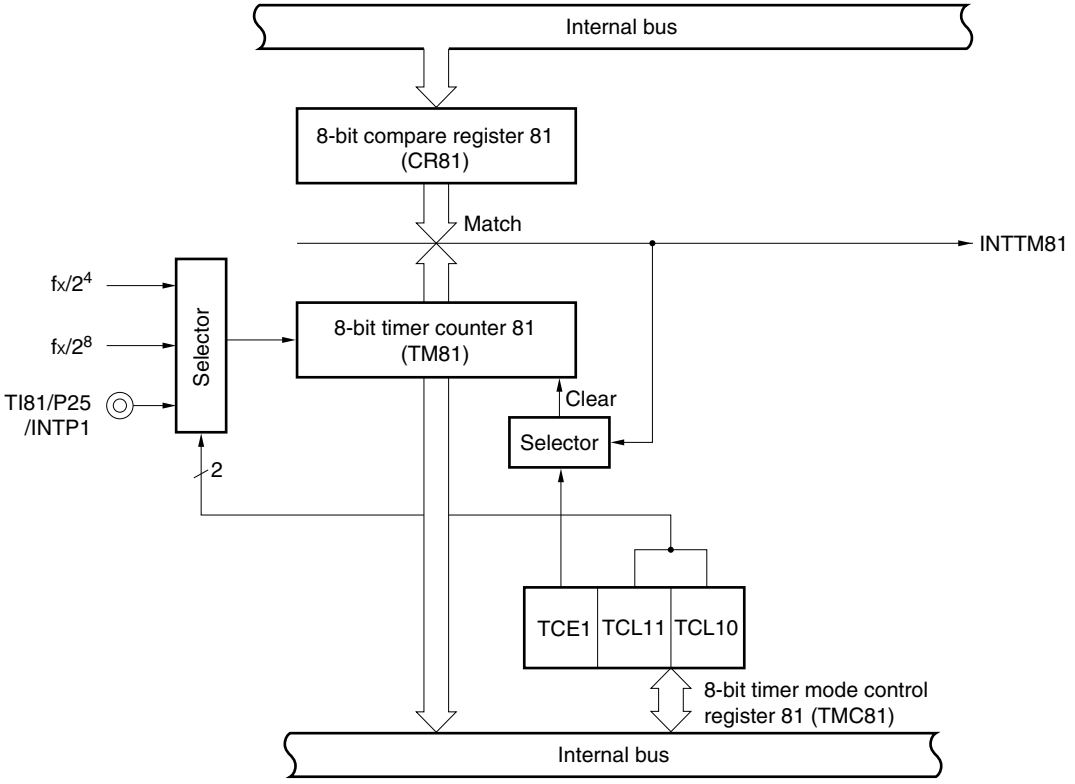
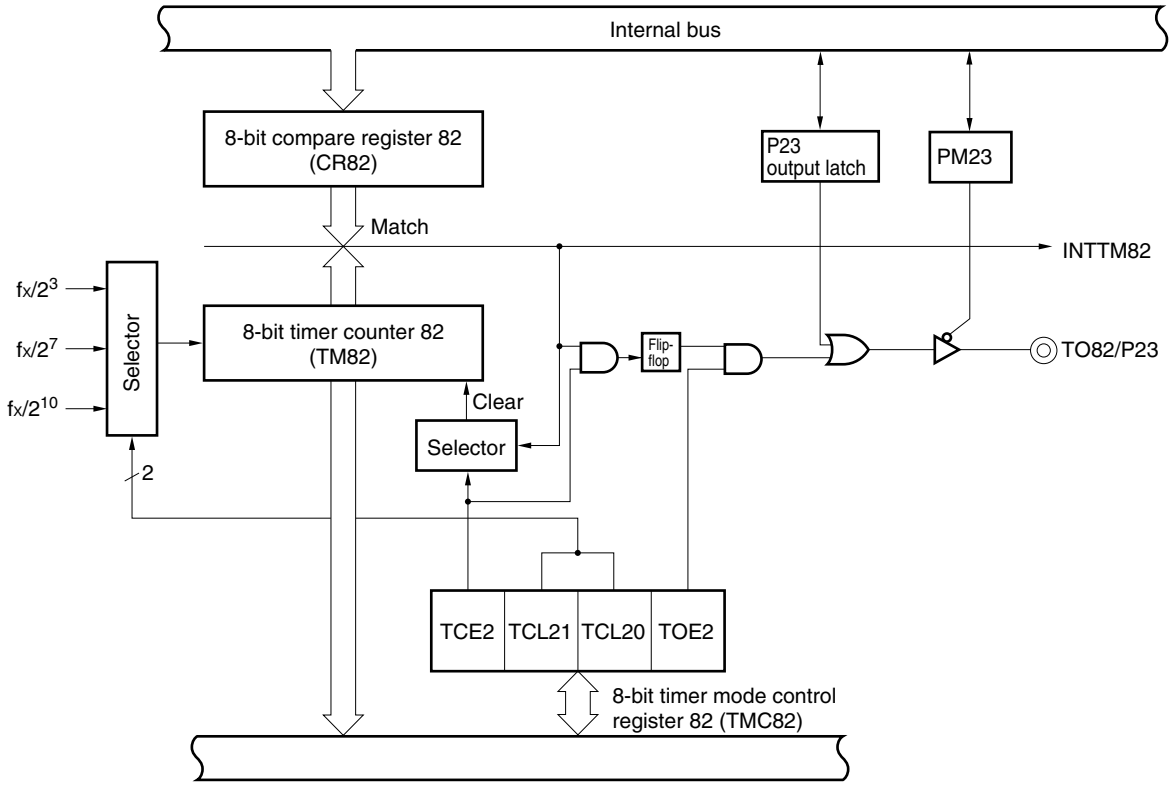


Figure 7-3. Block Diagram of 8-Bit Timer 82



(1) 8-bit compare register 8n (CR8n)

The value specified by CR8n is compared with the count in 8-bit timer counter 8n (TM8n). If they match, an interrupt request (INTTM8n) is issued.

CR8n is manipulated with an 8-bit memory manipulation instruction. Any value from 00H to FFH can be set.

$\overline{\text{RESET}}$ input makes CR8n undefined.

Caution Be sure to stop the operation of the timer before rewriting CR8n. If CR8n is rewritten while the timer is operation-enabled, an interrupt request match signal may be generated at the time of the rewrite.

Remark n = 0 to 2

(2) 8-bit timer counter 8n (TM8n)

TM8n is used to count the number of pulses.

Its contents are read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TM8n to 00H.

Remark n = 0 to 2

7.3 Registers Controlling 8-Bit Timer/Event Counters 80, 81, 82

The following three types of registers are used to control 8-bit timer/event counters 80, 81, and 82.

- 8-bit timer mode control registers 80 to 82 (TMC80 to TMC82)
- Port mode register 2 (PM2)
- Port register 2 (P2)

(1) 8-bit timer mode control register 80 (TMC80)

TMC80 determines whether to enable or disable 8-bit timer counter 80 (TM80) and specifies the count clock for 8-bit timer/event counter 80.

TMC80 is manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TMC80 to 00H.

Figure 7-4. Format of 8-Bit Timer Mode Control Register 80

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
TMC80	TCE0	0	0	0	0	TCL01	TCL00	0	FF53H	00H	R/W

TCE0	8-bit timer counter 80 operation control
0	Operation disabled (TM80 is cleared to 00H)
1	Operation enabled

TCL01	TCL00	8-bit timer/event counter 80 count clock selection
0	0	$f_X/2^6$ (131 kHz)
0	1	$f_X/2^9$ (16.4 kHz)
1	0	Rising edge of TI80
1	1	Falling edge of TI80

Cautions 1. Be sure to clear bits 0 and 3 to 6 to 0.

2. Always stop the timer before setting TMC80.

Remarks 1. f_X: System clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 8.38 MHz.

(2) 8-bit timer mode control register 81 (TMC81)

TMC81 determines whether to enable or disable 8-bit timer counter 81 (TM81) and specifies the count clock for 8-bit timer/event counter 81.

TMC81 is manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TMC81 to 00H.

Figure 7-5. Format of 8-Bit Timer Mode Control Register 81

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
TMC81	TCE1	0	0	0	0	TCL11	TCL10	0	FF57H	00H	R/W

TCE1	8-bit timer counter 81 operation control
0	Operation disabled (TM81 is cleared to 00H)
1	Operation enabled

TCL11	TCL10	8-bit timer/event counter 81 count clock selection
0	0	$f_X/2^4$ (524 kHz)
0	1	$f_X/2^8$ (32.7 kHz)
1	0	Rising edge of TI81
1	1	Falling edge of TI81

Cautions 1. Be sure to clear bits 0 and 3 to 6 to 0.

2. Always stop the timer before setting TMC81.

Remarks 1. f_X: System clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 8.38 MHz.

(3) 8-bit timer mode control register 82 (TMC82)

TMC82 determines whether to enable or disable 8-bit timer counter 82 (TM82) and specifies the count clock for 8-bit timer counter 82. It also controls the operation of the output controller of 8-bit timer counter 82.

TMC82 is manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TMC82 to 00H.

Figure 7-6. Format of 8-Bit Timer Mode Control Register 82

Symbol	<7>	6	5	4	3	2	1	<0>	Address	After reset	R/W
TMC82	TCE2	0	0	0	0	TCL21	TCL20	TOE2	FF5BH	00H	R/W

TCE2	8-bit timer counter 82 operation control
0	Operation disabled (TM82 is cleared to 00H)
1	Operation enabled

TCL21	TCL20	8-bit timer counter 82 count clock selection
0	0	$f_X/2^3$ (1.05 MHz)
0	1	$f_X/2^7$ (65.5 kHz)
1	0	$f_X/2^{10}$ (8.18 kHz)
1	1	Setting prohibited

TOE2	8-bit timer 82 output control
0	Operation disabled (port mode)
1	Output enabled

Cautions 1. Be sure to clear bits 3 to 6 to 0.

2. Always stop the timer before setting TMC82.

Remarks 1. f_X: System clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 8.38 MHz.

(4) Port mode register 2 (PM2)

PM2 specifies whether each bit of port 2 is used for input or output.

★ To use the P23/TO82 pin for timer output, the PM23 and P23 output latches must be reset to 0. When using the P24/TI80/INTP0 and P25/TI81/INTP1 pins as timer inputs, set PM24 and PM25 to 1. At this time, the output latch of P24 and P25 can be either 1 or 0.

PM2 is manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM2 to FFH.

Figure 7-7. Format of Port Mode Register 2

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM2	1	1	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W

PM2n	P2n pin input/output mode selection (n = 0 to 5)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

7.4 Operation of 8-Bit Timer/Event Counters 80, 81, 82

7.4.1 Operation as interval timer

The interval timer repeatedly generates an interrupt at time intervals specified by the count value preset to 8-bit compare registers 80, 81, and 82 (CR80, CR08, and CR82).

To operate the 8-bit timer/event counter as an interval timer, make the settings in the following order.

<1> Set 8-bit timer counter 8n (TM8n) to operation-disabled (TCEn (bit 7 of 8-bit timer mode control register 8n (TMC8n)) = 0)
<2> Select the count clock of the 8-bit timer/event counter (see **Tables 7-6** to **7-8**)
<3> Set the count value to CR8n
<4> Set TM8n to operation-enabled (TCEn = 1)

When the count value of 8-bit timer counter 8n (TM8n) matches the value set to CR8n, the value of TM8n is cleared to 00H and TM8n continues counting. At the same time, an interrupt request signal (INTTM8n) is generated.

Tables 7-6 to 7-8 show the interval time, and Figure 7-8 shows the timing of interval timer operation.

Caution When the setting of the count clock using TMC8n and the setting of TM8n to operation-enabled using an 8-bit memory manipulation instruction are performed at the same time, an error of one clock or more may occur in the first cycle after the timer is started. Because of this, when the 8-bit timer/event counter operates as an interval timer, be sure to make the settings in the order described above.

Remark n = 0 to 2

Table 7-6. Interval Time of 8-Bit Timer/Event Counter 80

TCL01	TCL00	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	2^6/fx (7.64 μs)	2^{14}/fx (1.96 ms)	2^6/fx (7.64 μs)
0	1	2^9/fx (61.1 μs)	2^{17}/fx (15.6 ms)	2^9/fx (61.1 μs)
1	0	TI80 input cycle	$2^8 \times$ TI80 input cycle	TI80 input edge cycle
1	1	TI80 input cycle	$2^8 \times$ TI80 input cycle	TI80 input edge cycle

Remarks 1. fx: System clock oscillation frequency
2. The parenthesized values apply to operation at fx = 8.38 MHz.

Table 7-7. Interval Time of 8-Bit Timer/Event Counter 81

TCL11	TCL10	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	2^4/fx (19.1 μs)	2^{12}/fx (0.49 ms)	2^4/fx (1.91 μs)
0	1	2^8/fx (30.5 μs)	2^{16}/fx (7.82 ms)	2^8/fx (30.5 μs)
1	0	TI81 input cycle	$2^8 \times$ TI81 input cycle	TI81 input edge cycle
1	1	TI81 input cycle	$2^8 \times$ TI81 input cycle	TI81 input edge cycle

Remarks 1. fx: System clock oscillation frequency
2. The parenthesized values apply to operation at fx = 8.38 MHz.

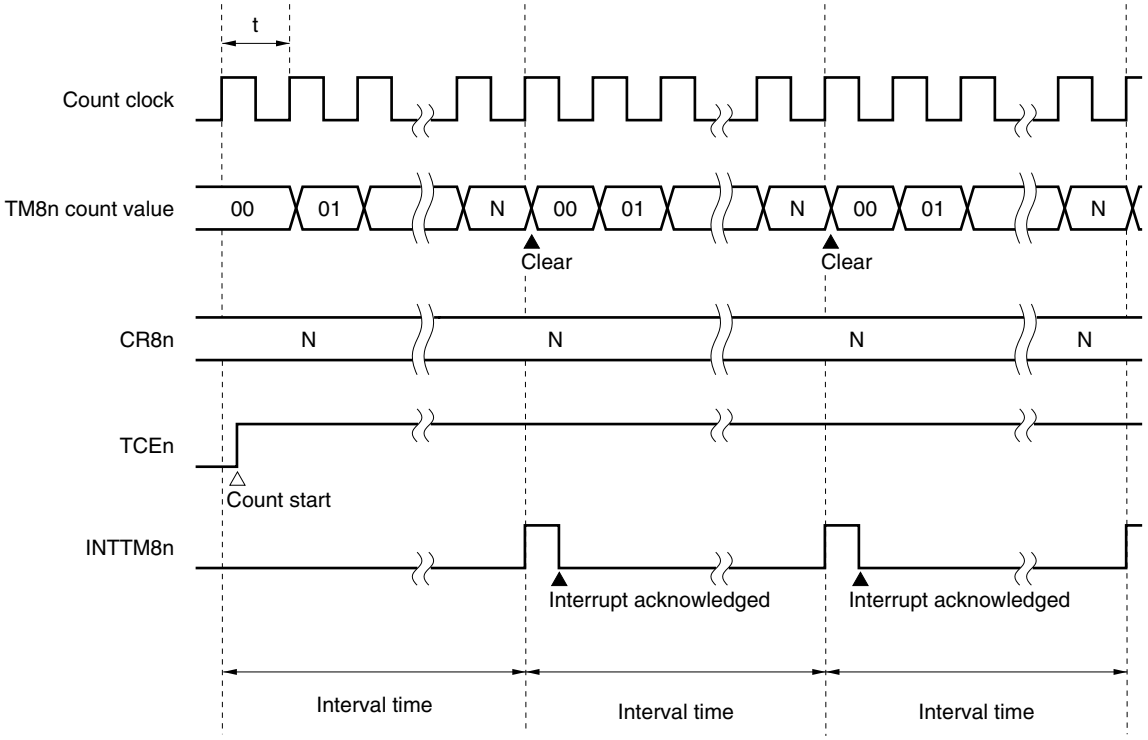
Table 7-8. Interval Time of 8-Bit Timer 82

TCL21	TCL20	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	2^3/fx (0.95 μs)	2^{11}/fx (0.24 ms)	2^3/fx (0.95 μs)
0	1	2^7/fx (15.3 μs)	2^{15}/fx (3.91 ms)	2^7/fx (15.3 μs)
1	0	2^{10}/fx (0.12 ms)	2^{18}/fx (31.3 ms)	2^{10}/fx (0.12 ms)
1	1	Setting prohibited		

Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

Figure 7-8. Interval Timer Operation Timing of TM80, TM81, TM82



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

2. n = 0 to 2

7.4.2 Operation as external event counter[Note]

The external event counter counts the number of external clock pulses input to the TI80/P24/INTP0 and TI81/P25/INTP1 pins by using 8-bit timer counters 80 and 81 (TM80 and TM81).

To operate the 8-bit timer/event counter as an external event counter, make the settings in the following order.

<1> Set P24 and P25 to input mode (PM24 = 1, PM25 = 1)
<2> Set 8-bit timer counter 8n (TM08) to operation-disabled (TCEn (bit 7 of 8-bit timer mode control register 8n (TMC8n)) = 0)
<3> Specify the rising edge/falling edge of TI8n (see **Figures 7-4** and **7-5**)
<4> Set the count value to CR8n
<5> Set TM8n to operation-enabled (TCEn = 1)

Note This function is only for TM80 and TM81.

Each time the valid edge specified by bit 1 (TCLn0) of TMC8n is input, the value of 8-bit timer counter 8n (TM8n) is incremented.

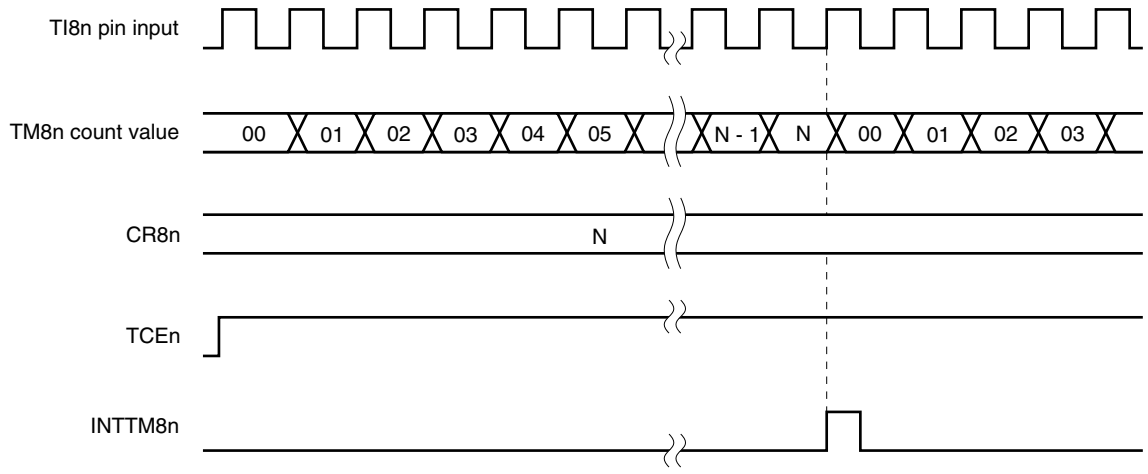
When the count value of TM8n matches the value set to CR8n, the value of TM8n is cleared to 00H and TM8n continues counting. At the same time, an interrupt request signal (INTTM8n) is generated.

Figure 7-9 shows the timing of external event counter operation (with rising edge specified).

Caution When the setting of the count clock using TMC8n and the setting of TM8n to operation-enabled using an 8-bit memory manipulation instruction are performed at the same time, an error of one clock or more may occur in the first cycle after the timer is started. Because of this, when the 8-bit timer/event counter operates as an external event counter, be sure to make the settings in the order described above.

Remark n = 0, 1

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



Remarks 1. N = 00H to FFH
2. n = 0, 1

7.4.3 Operation as square-wave output[Note]

The 8-bit timer/event counter can generate a square-wave output of any frequency at intervals specified by the count value preset to 8-bit compare register 82 (CR82).

To operate 8-bit timer counter 82 as a square-wave output, make the settings in the following order.

<1> Set P23 to output mode (PM23 = 0), and set the output latch of P23 to 0
<2> Set 8-bit timer counter 82 (TM82) to operation-disabled (TCE2 (bit 0 of 8-bit timer mode control register 82 (TMC82)) = 1)
<3> Set the count clock of 8-bit timer 82 (see **Table 7-9**), and set TO82 to output-enabled (TOE2 (bit 0 of TMC82) = 1)
<4> Set the count value to CR82
<5> Set TM82 to operation-enabled (TCE2 = 1)

Note This function is only for TM82.

When the count value of 8-bit timer counter 82 (TM82) matches the value set in CR82, the TO82/P23 pin output will be inverted. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurs, the TM82 value is cleared to 00H, counting continues, and an interrupt request signal (INTTM82) is generated.

Setting bit 7 of TMC82 (TCE2) to 0 clears the square-wave output to 0.

Table 7-9 lists the square-wave output range, and Figure 7-10 shows the timing of square-wave output.

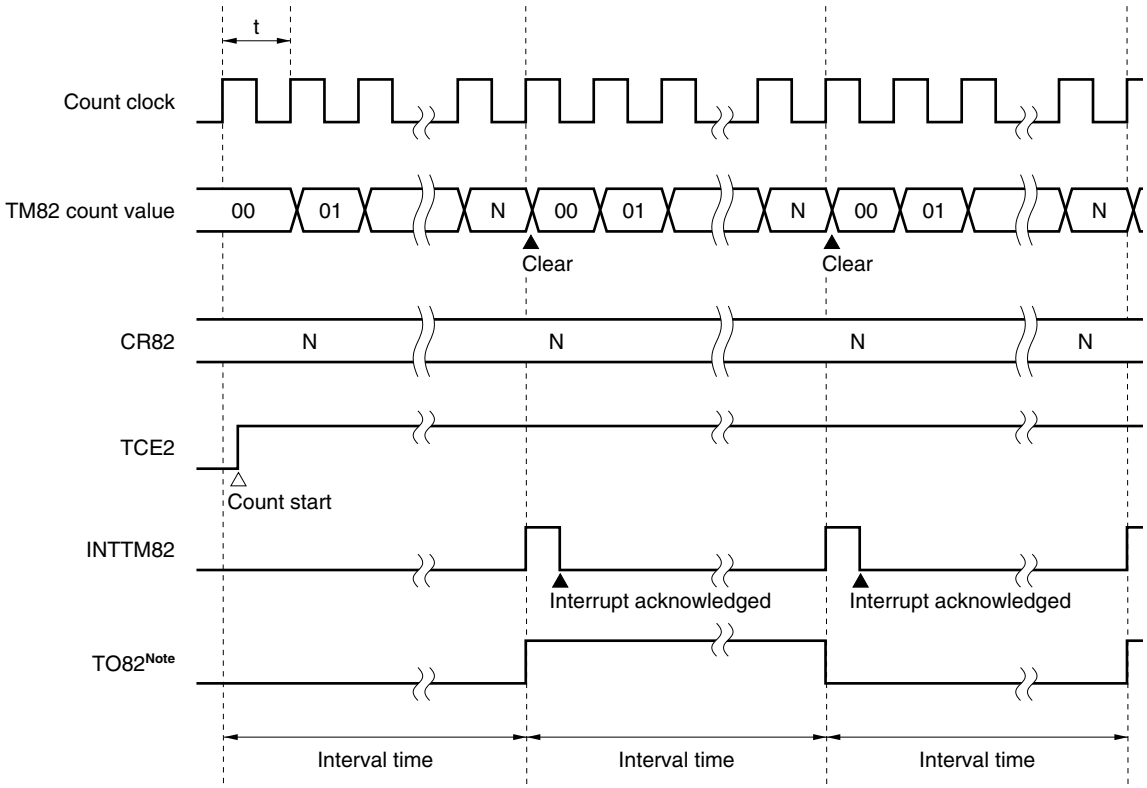
Caution When the setting of the count clock using TMC82 and the setting of TM82 to operation-enabled using an 8-bit memory manipulation instruction are performed at the same time, an error of one clock or more may occur in the first cycle after the timer is started. Because of this, when the 8-bit timer/event counter operates as a square-wave output, be sure to make the settings in the order described above.

Table 7-9. Square-Wave Output Range of 8-Bit Timer 82

TCL21	TCL20	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	$2^3/f_X$ (0.95 μs)	$2^{11}/f_X$ (0.24 ms)	$2^3/f_X$ (0.95 μs)
0	1	$2^7/f_X$ (15.3 μs)	$2^{15}/f_X$ (3.91 ms)	$2^7/f_X$ (15.3 μs)
1	0	$2^{10}/f_X$ (0.12 ms)	$2^{18}/f_X$ (31.3 ms)	$2^{10}/f_X$ (0.12 ms)
1	1	Setting prohibited		

Remarks 1. f_X: System clock oscillation frequency
2. The parenthesized values apply to operation at f_X = 8.38 MHz.

Figure 7-10. Square-Wave Output Timing



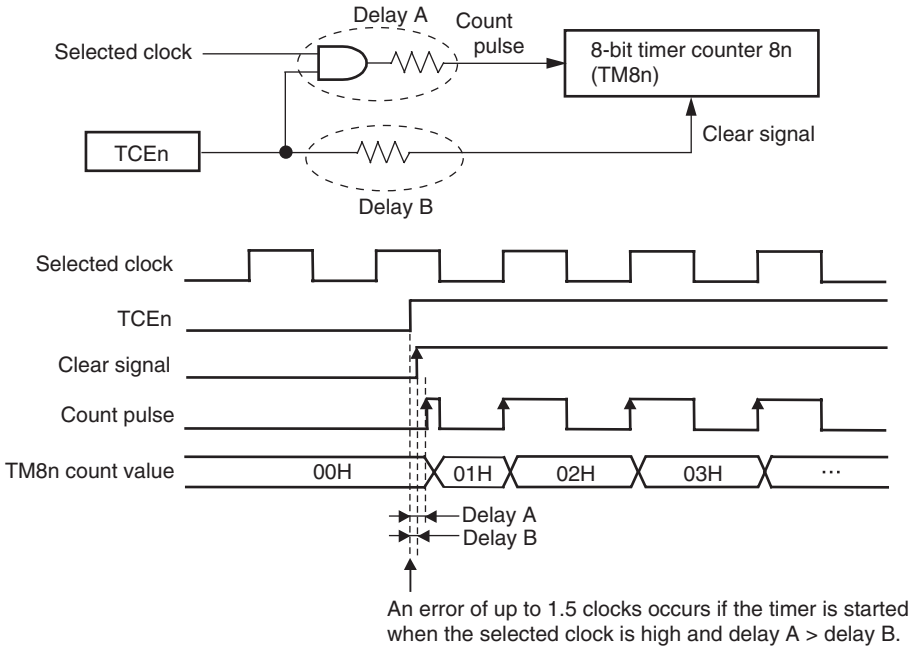
Note The initial value of TO82 when output is enabled (TOE2 = 1) becomes low level.

★ 7.5 Notes on Using 8-Bit Timer/Event Counters 80, 81, 82

(1) Error on starting timer

An error of up to 1.5 clocks is included in the time between the timer being started and a match signal being generated. This is because the rising edge is detected and the counter is incremented if the timer is started while the count clock is high (see **Figure 7-11**).

Figure 7-11. Case of Error Occurrence of up to 1.5 Clocks



Remark n = 0 to 2

(2) Count value if external clock input from TI8n pin is selected

When the rising edge of the external clock signal input from the TI8n pin is selected as the count clock, the count value may start from 01H if the timer is enabled (TCEn = 0 → 1) while the TI8n pin is high. This is because the input signal of the TI8n pin is internally ANDed with the TCEn signal. Consequently, the counter is incremented because the rising edge of the count clock is input to the timer immediately when the TCEn pin is set. Depending on the delay timing, the count value is incremented by one if the rising edge is input after the counter is cleared. Counting is not affected if the rising edge is input before the counter is cleared (the counter operates normally).

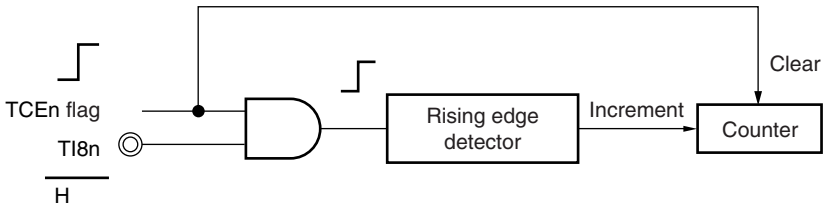
Similarly, when the falling edge of the external signal input from the TI8n pin is selected as the count clock, the count value may start from 01H if the timer is enabled (TCEn = 0 → 1) while the TI8n pin is low.

Use the timer being aware that it has an error of one count, or take either of the following actions A or B.

<Action A> Always start the timer when the TI8n pin is low when the rising edge is selected.
Always start the timer when the TI8n pin is high when the falling edge is selected.

<Action B> Save the count value to a control register when the timer is started, subtract the count value from the count value saved to the control register when reading the count value, and take the result as the true count value.

Figure 7-12. Count Operation if Timer Is Started When TI8n Is High (Rising Edge Selected)



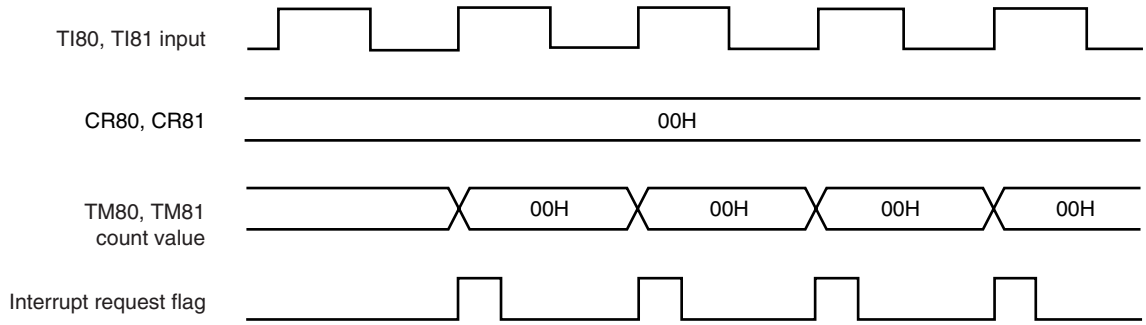
Remark n = 0, 1

(3) Setting of 8-bit compare register 8n

8-bit compare register 8n (CR8n) can be set to 00H.

Therefore, one pulse can be counted when the 8-bit timer operates as an event counter.

Figure 7-13. Timing of Count Operation of 1 Pulse



Caution Stop the timer operation (TCEn (bit 7 of 8-bit timer mode control register 8n (TMC8n)) = 0) before rewriting CR8n. If CR8n is rewritten while the timer operation is enabled, a match interrupt request signal may be generated immediately at the point of rewrite.

Remark n = 0 to 2

(4) Notes on setting STOP mode

Be sure to stop the timer operation (TCEn = 0) before executing the STOP instruction.

Remark n = 0 to 2

CHAPTER 8 WATCH TIMER

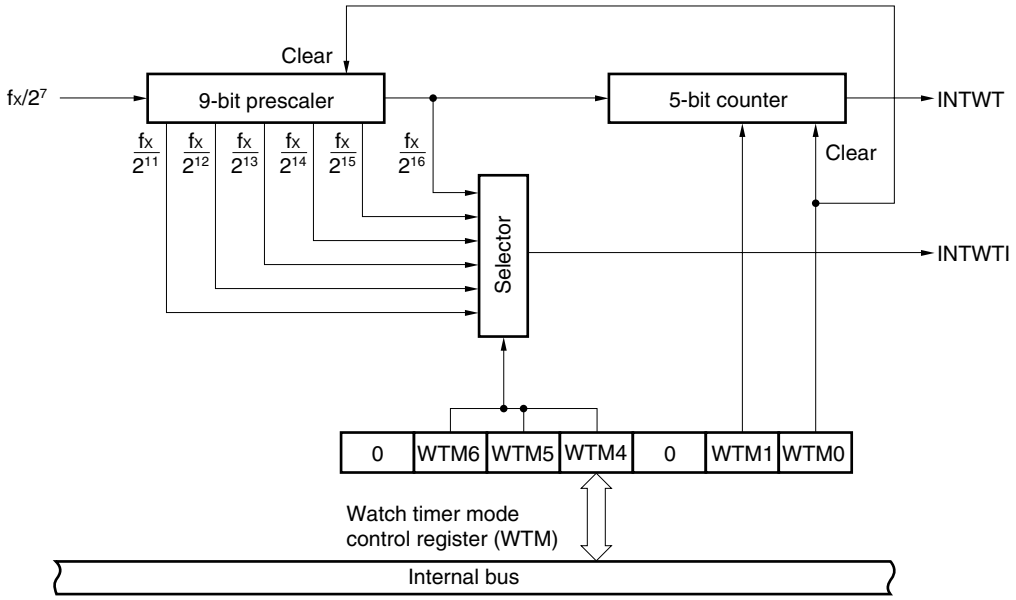
8.1 Watch Timer Functions

The watch timer has the following functions.

- Watch timer
- Interval timer

The watch and interval timers can be used at the same time.
Figure 8-1 shows a block diagram of the watch timer.

Figure 8-1. Block Diagram of Watch Timer



(1) Watch timer

The 8.38 MHz system clock is used to generate an interrupt request (INTWT) at 0.25-second intervals.

(2) Interval timer

The interval timer is used to generate an interrupt request (INTWTI) at specified intervals.

Table 8-1. Interval Generated Using Interval Timer

Interval	At f_X = 8.38 MHz
$2^{11} \times 1/f_X$	244 μs
$2^{12} \times 1/f_X$	489 μs
$2^{13} \times 1/f_X$	978 μs
$2^{14} \times 1/f_X$	1.96 ms
$2^{15} \times 1/f_X$	3.91 ms
$2^{16} \times 1/f_X$	7.82 ms

Remark fx: System clock oscillation frequency

8.2 Watch Timer Configuration

The watch timer consists of the following hardware.

Table 8-2. Watch Timer Configuration

Item	Configuration
Counter	5 bits × 1
Prescaler	9 bits × 1
Control register	Watch timer mode control register (WTM)

8.3 Register Controlling Watch Timer

The following register is used to control the watch timer.

- Watch timer mode control register (WTM)
  WTM selects the count clock for the watch timer and specifies whether to enable clocking of the timer. It also specifies the prescaler interval and how the 5-bit counter is controlled.
  WTM is set with a 1-bit or 8-bit memory manipulation instruction.
  $\overline{\text{RESET}}$ input clears WTM to 00H.

Figure 8-2. Format of Watch Timer Mode Control Register

Symbol	7	6	5	4	3	2	<1>	<0>	Address	After reset	R/W
WTM	0	WTM6	WTM5	WTM4	0	0	WTM1	WTM0	FF4AH	00H	R/W

WTM6	WTM5	WTM4	Prescaler interval selection
0	0	0	$2^{11}/f_X$ (244 μs)
0	0	1	$2^{12}/f_X$ (489 μs)
0	1	0	$2^{13}/f_X$ (978 μs)
0	1	1	$2^{14}/f_X$ (1.96 ms)
1	0	0	$2^{15}/f_X$ (3.91 ms)
1	0	1	$2^{16}/f_X$ (7.82 ms)
Other than above			Setting prohibited

WTM1	Control of 5-bit counter operation
0	Cleared after stop
1	Started

WTM0	Watch timer operation
0	Operation disabled (both prescaler and timer cleared)
1	Operation enabled

Cautions 1. Be sure to clear bits 2, 3 and 7 to 0.

★ **2. Do not change the interval time (using bits 4 to 6 (WTM4 to WTM6) of WTM) while the watch timer is operating.**

Remarks 1. f_X: System clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 8.38 MHz.

8.4 Watch Timer Operation

8.4.1 Operation as watch timer

The 8.38 MHz system clock is used for watch timer operation at 0.25-second intervals.

The watch timer is used to generate an interrupt request at specified intervals.

By setting bits 0 and 1 (WTM0 and WTM1) of the watch timer mode control register (WTM) to 1, the watch timer starts counting. By setting them to 0, the 5-bit counter is cleared and the watch timer stops counting.

Only the watch timer can be started from zero seconds by clearing WTM1 to 0 when the interval timer and watch timer operate at the same time. In this case, however, an error of up to $2^{16} \times 1/f_X$ may occur in the overflow (INTWT) after the zero-second start of the watch timer because the 9-bit prescaler is not cleared to 0.

8.4.2 Operation as interval timer

The interval timer is used to repeatedly generate an interrupt request at the interval specified by a count value set in advance.

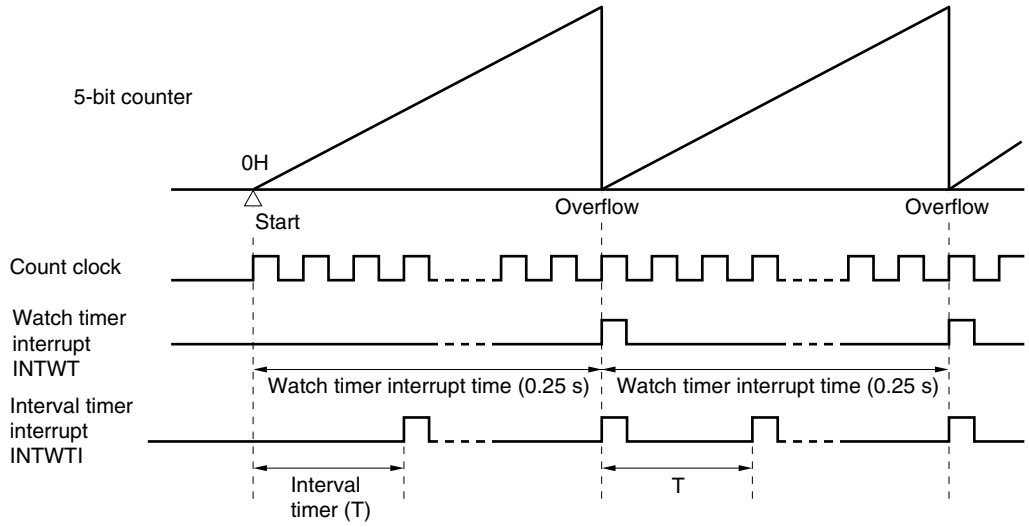
The interval can be selected by bits 4 to 6 (WTM4 to WTM6) of the watch timer mode control register (WTM).

Table 8-3. Interval Generated Using Interval Timer

WTM6	WTM5	WTM4	Interval	At f_X = 8.38 MHz
0	0	0	$2^{11} \times 1/f_X$	244 μs
0	0	1	$2^{12} \times 1/f_X$	489 μs
0	1	0	$2^{13} \times 1/f_X$	978 μs
0	1	1	$2^{14} \times 1/f_X$	1.96 ms
1	0	0	$2^{15} \times 1/f_X$	3.91 ms
1	0	1	$2^{16} \times 1/f_X$	7.82 ms
Other than above			Setting prohibited	

Remark f_X: System clock oscillation frequency

Figure 8-3. Watch Timer/Interval Timer Operation Timing



★ **Caution If the watch timer and 5-bit counter are enabled by the watch timer mode control register (WTM) (by setting bit 0 (WTM0) of WTM to 1), the time from this setting to the occurrence of the first interrupt request (INTWT) is not exactly the value set as the watch timer interrupt time (0.25 s). This is because the 5-bit counter is late by one output cycle of the 9-bit prescaler in starting counting. The second INTWT signal and those that follow are generated exactly at the set time.**

Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

CHAPTER 9 WATCHDOG TIMER

9.1 Watchdog Timer Functions

The watchdog timer has the following functions.

- Watchdog timer
- Interval timer

Caution Select the watchdog timer mode or interval timer mode by using the watchdog timer mode register (WDTM) (the watchdog timer and interval timer cannot be used at the same time).

(1) Watchdog timer

The watchdog timer is used to detect inadvertent program loops. When an inadvertent loop is detected, a non-maskable interrupt or the $\overline{\text{RESET}}$ signal can be generated.

Table 9-1. Inadvertent Loop Detection Time of Watchdog Timer

Inadvertent Loop Detection Time	At f_X = 8.38 MHz
$2^{11} \times 1/f_X$	244 μs
$2^{13} \times 1/f_X$	977 μs
$2^{15} \times 1/f_X$	3.91 ms
$2^{17} \times 1/f_X$	15.6 ms

f_X: System clock oscillation frequency

(2) Interval timer

The interval timer generates an interrupt at a preset interval.

Table 9-2. Interval Time

Interval	At f_X = 8.38 MHz
$2^{11} \times 1/f_X$	244 μs
$2^{13} \times 1/f_X$	977 μs
$2^{15} \times 1/f_X$	3.91 ms
$2^{17} \times 1/f_X$	15.6 ms

f_X: System clock oscillation frequency

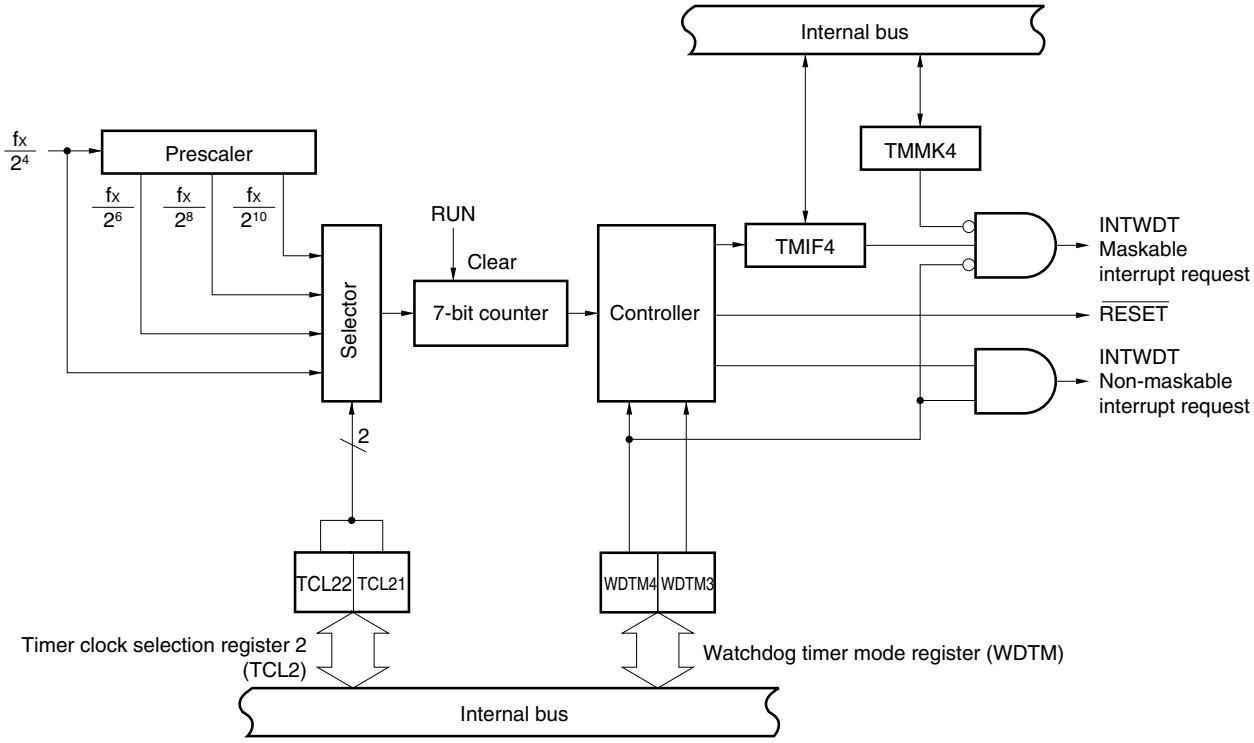
9.2 Watchdog Timer Configuration

The watchdog timer consists of the following hardware.

Table 9-3. Configuration of Watchdog Timer

Item	Configuration
Control registers	Timer clock selection register 2 (TCL2) Watchdog timer mode register (WDTM)

Figure 9-1. Block Diagram of Watchdog Timer



9.3 Watchdog Timer Control Registers

The following two registers are used to control the watchdog timer.

- Timer clock selection register 2 (TCL2)
- Watchdog timer mode register (WDTM)

(1) Timer clock selection register 2 (TCL2)

This register sets the watchdog timer count clock.

TCL2 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TCL2 to 00H.

Figure 9-2. Format of Timer Clock Selection Register 2

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TCL2	0	0	0	0	0	TCL22	TCL21	0	FF42H	00H	R/W

TCL22	TCL21	Watchdog timer count clock selection	Interval
0	0	$f_X/2^4$ (524.3 kHz)	$2^{11}/f_X$ (244.3 μs)
0	1	$f_X/2^6$ (131.1 kHz)	$2^{13}/f_X$ (977.6 μs)
1	0	$f_X/2^8$ (32.7 kHz)	$2^{15}/f_X$ (3.91 ms)
1	1	$f_X/2^{10}$ (8.18 kHz)	$2^{17}/f_X$ (15.6 ms)
Other than above		Setting prohibited	

Remarks 1. f_X: System clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 8.38 MHz.

(2) Watchdog timer mode register (WDTM)

This register sets an operation mode of the watchdog timer, and enables or disables counting of the watchdog timer.

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

$\overline{\text{RESET}}$ input clears WDTM to 00H.

Figure 9-3. Format of Watchdog Timer Mode Register

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
WDTM	RUN	0	0	WDTM4	WDTM3	0	0	0	FFF9H	00H	R/W

RUN	Watchdog timer operation selection[Note 1]
0	Stop counting
1	Clear counter and start counting

WDTM4	WDTM3	Watchdog timer operation mode selection[Note 2]
0	0	Operation stop
0	1	Interval timer mode (overflow and maskable interrupt occur)[Note 3]
1	0	Watchdog timer mode 1 (overflow and non-maskable interrupt occur)
1	1	Watchdog timer mode 2 (overflow occurs and reset operation started)

Notes 1. Once RUN has been set to 1, it cannot be cleared to 0 by software. Therefore, when counting is started, it cannot be stopped by any means other than $\overline{\text{RESET}}$ input.

2. Once WDTM3 and WDTM4 have been set to 1, they cannot be cleared to 0 by software.

3. The watchdog timer starts operations as an interval timer when RUN is set to 1.

Cautions 1. When the watchdog timer is cleared by setting RUN to 1, the actual overflow time is up to 0.8% shorter than the time set by timer clock selection register 2 (TCL2).

2. In watchdog timer mode 1 or 2, set TMMK4 (bit 0 of interrupt mask flag register 0 (MK0)) to 1 after confirming that TMIF4 (bit 0 of interrupt request flag register 0 (IF0)) is set to 0. When watchdog timer mode 1 or 2 is selected under the condition where TMIF4 is 1, a non-maskable interrupt request occurs at the completion of rewriting.

9.4 Watchdog Timer Operation

9.4.1 Operation as watchdog timer

The watchdog timer detects an inadvertent program loop when bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 1.

The count clock (inadvertent loop detection time interval) of the watchdog timer can be selected by bits 1 and 2 (TCL21 and TCL22) of timer clock selection register 2 (TCL2). By setting bit 7 (RUN) of WDTM to 1, the watchdog timer is started. Set RUN to 1 within the set inadvertent loop detection time interval after the watchdog timer has been started. By setting RUN to 1, the watchdog timer can be cleared and start counting. If RUN is not set to 1, and the inadvertent loop detection time is exceeded, the system is reset or a non-maskable interrupt request is generated by the value of bit 3 (WDTM3) of WDTM.

The watchdog timer continues operation in HALT mode, but stops in STOP mode. Therefore, set RUN to 1 before entering STOP mode to clear the watchdog timer, and then execute the STOP instruction.

Caution The actual inadvertent loop detection time may be up to 0.8% shorter than the set time.

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

TCL22	TCL21	Inadvertent Loop Detection Time	At f_X = 8.38 MHz
0	0	$2^{11} \times 1/f_X$	244 μs
0	1	$2^{13} \times 1/f_X$	977 μs
1	0	$2^{15} \times 1/f_X$	3.91 ms
1	1	$2^{17} \times 1/f_X$	15.6 ms

f_X: System clock oscillation frequency

9.4.2 Operation as interval timer

When bits 4 and 3 (WDTM4 and WDTM3) of watching timer mode register (WDTM) are set to 0 and 1, respectively, the watchdog timer also operates as an interval timer that repeatedly generates an interrupt request at intervals specified by a count value set in advance.

Select the count clock (or interval) by setting bits 1 and 2 (TCL21 and TCL22) of timer clock selection register 2 (TCL2). The watchdog timer starts operation as an interval timer when the RUN bit (bit 7 of WDTM) is set to 1.

In interval timer mode, the interrupt mask flag (TMMK4: bit 0 of interrupt mask flag register 0 (MK0)) is valid, and a maskable interrupt request (INTWDT) can be generated. The priority of INTWDT is set as the highest of all the maskable interrupts.

The interval timer continues operation in HALT mode, but stops in STOP mode. Therefore, set RUN to 1 before entering STOP mode to clear the interval timer, and then execute the STOP instruction.

Cautions 1. Once bit 4 (WDTM4) of WDTM is set to 1 (when watchdog timer mode is selected), the interval timer mode is not set unless the $\overline{\text{RESET}}$ signal is input.

2. The interval time immediately after the setting by WDTM may be up to 0.8% shorter than the set time.

Table 9-5. Interval Time of Interval Timer

TCL22	TCL21	Interval	At f_X = 8.38 MHz
0	0	$2^{11} \times 1/f_X$	244 μs
0	1	$2^{13} \times 1/f_X$	977 μs
1	0	$2^{15} \times 1/f_X$	3.91 ms
1	1	$2^{17} \times 1/f_X$	15.6 ms

f_X: System clock oscillation frequency

CHAPTER 10 A/D CONVERTER

10.1 A/D Converter Functions

The A/D converter is an 8-bit resolution converter used to convert an analog input into a digital signal. This converter can control the analog inputs of up to eight channels (ANI0 to ANI7).

A/D conversion can be started only by software.

One of analog inputs ANI0 to ANI7 is selected for A/D conversion. A/D conversion is performed repeatedly, with an interrupt request (INTAD) being issued each time an A/D conversion is completed.

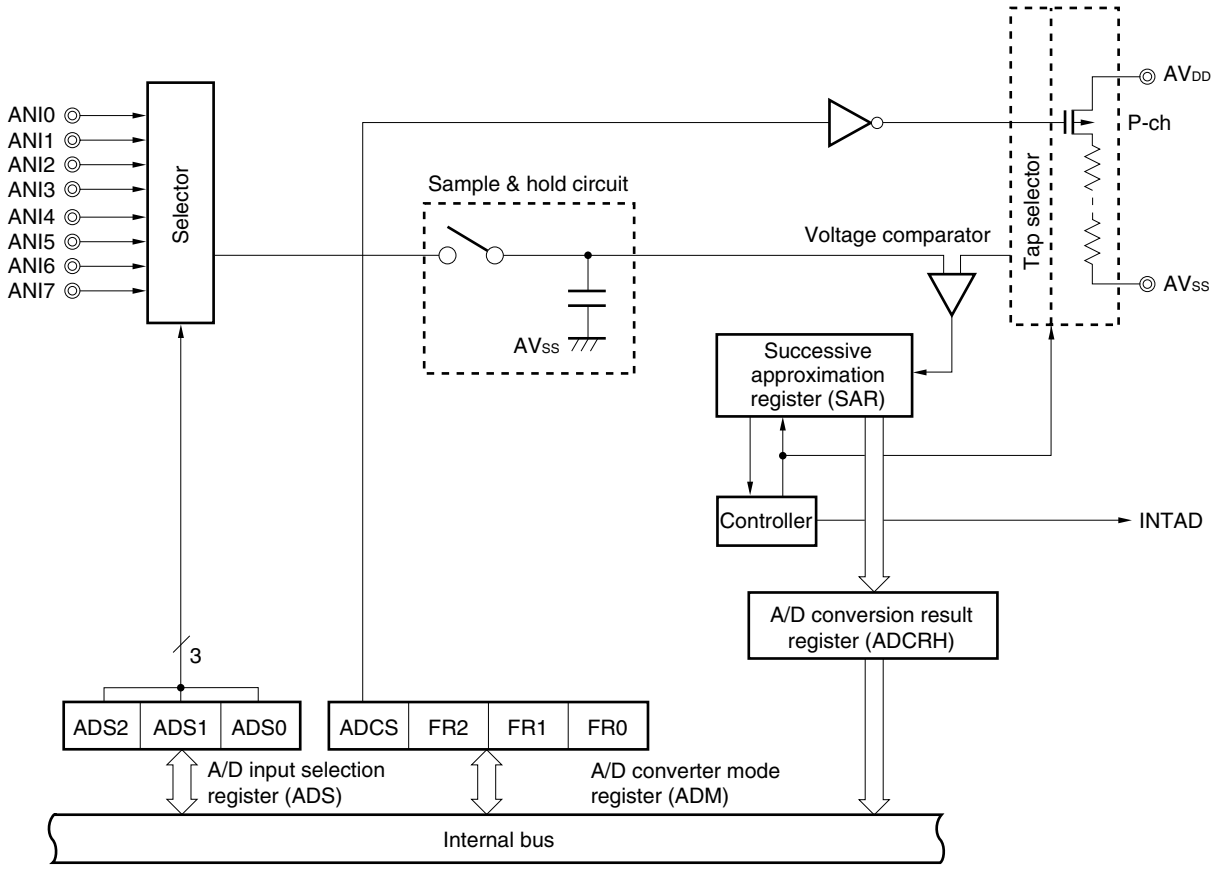
10.2 A/D Converter Configuration

The A/D converter consists of the following hardware.

Table 10-1. Configuration of A/D Converter

Item	Configuration
Analog input	8 channels (ANI0 to ANI7)
Registers	Successive approximation register (SAR) A/D conversion result register (ADCRH)
Control registers	A/D converter mode register (ADM) A/D input selection register (ADS)

Figure 10-1. Block Diagram of A/D Converter



(1) Successive approximation register (SAR)

SAR receives the result of comparing an analog input voltage and a voltage at a voltage tap (comparison voltage), received from the series resistor string, starting from the most significant bit (MSB).

Upon receiving all the bits, down to the least significant bit (LSB), that is, upon the completion of A/D conversion, SAR sends its contents to the A/D conversion result register (ADCRH).

(2) A/D conversion result register (ADCRH)

ADCRH holds the result of A/D conversion. Each time A/D conversion ends, the conversion result received from the successive approximation register is loaded into ADCRH, which is an 8-bit register.

ADCRH can be read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes this register undefined.

(3) Sample & hold circuit

The sample & hold circuit samples the input signal of the analog input pin that is selected by the selector when A/D conversion is started, and holds the sampled analog input voltage value during A/D conversion.

(4) Voltage comparator

The voltage comparator compares an sampled analog input voltage with the voltage output by the series resistor string.

(5) Series resistor string

The series resistor string is configured between AV_{DD} and AV_{SS}. It generates the reference voltages against which analog inputs are compared.

(6) ANI0 to ANI7 pins

The ANI0 to ANI7 pins are the 8-channel analog input pins for the A/D converter. They are used to receive the analog signals for A/D conversion.

Cautions 1. Do not supply the ANI0 to ANI7 pins with voltages that fall outside the rated range. If a voltage of AV_{DD} or higher or AV_{SS} or lower (even if within the absolute maximum ratings) is supplied to any of these pins, the conversion value for the corresponding channel will be undefined. Furthermore, the conversion values for the other channels may also be affected.

2. The analog input pins (ANI0 to ANI7) also function as input port pins (P60 to P67). When A/D conversion is performed with any of the ANI0 to ANI7 pins selected, be sure not to execute an instruction that inputs data to port 6 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 obtained due to coupling noise. Therefore, avoid applying pulses to pins adjacent to the pin undergoing A/D conversion.

(7) AV_{SS} pin

The AV_{SS} pin is the ground potential pin for the A/D converter. This pin must be held at the same potential as the V_{SS} pin, even while the A/D converter is not being used.

(8) AV_{DD} pin

The AV_{DD} pin is the analog power supply pin for the A/D converter. This pin must be held at the same potential as the V_{DD} pin, even while the A/D converter is not being used.

10.3 Registers Controlling A/D Converter

The following two registers are used to control the A/D converter.

- A/D converter mode register (ADM)
- A/D input selection register (ADS)

(1) A/D converter mode register (ADM)

ADM specifies the conversion time for analog inputs. It also specifies whether to enable conversion.

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

$\overline{\text{RESET}}$ input clears ADM to 00H.

Figure 10-2. Format of A/D Converter Mode Register

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
ADM	ADCS	0	FR2	FR1	FR0	0	0	0	FF80H	00H	R/W

ADCS	A/D conversion control
0	Conversion disabled
1	Conversion enabled

FR2	FR1	FR0	A/D conversion time selection[Note 1]
0	0	0	288/fx (34.4 μs)
0	0	1	240/fx (28.6 μs)
0	1	0	192/fx (22.9 μs)
1	0	0	144/fx (17.1 μs)
1	0	1	120/fx (14.3 μs)
1	1	0	96/fx (Setting prohibited[Note 2])
Other than above			Setting prohibited

Notes 1. The specifications of FR2, FR1, and FR0 must be such that the A/D conversion time is at least 14 μs.

2. These bit combinations must not be used, as the A/D conversion time will fall below 14 μs.

Cautions 1. The result of conversion performed immediately after bit 7 (ADCS) is set may be undefined.

2. The conversion result may be undefined after ADCS has been cleared to 0 (For details, see 10.5 (5)).

Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

(2) A/D input selection register (ADS)

ADS specifies the port used to input the analog voltage to be converted to a digital signal.

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

$\overline{\text{RESET}}$ input clears ADS to 00H.

Figure 10-3. Format of A/D Input Selection Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADS	0	0	0	0	0	ADS2	ADS1	ADS0	FF84H	00H	R/W

ADS2	ADS1	ADS0	Analog input channel specification
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
1	1	0	ANI6
1	1	1	ANI7

Caution Be sure to clear bits 3 to 7 to 0.

10.4 A/D Converter Operation

10.4.1 Basic operation of A/D converter

<1> Select a channel for A/D conversion, using the A/D input selection register (ADS).

<2> The voltage supplied to the selected analog input channel is sampled using the sample & hold circuit.

<3> Sampling continues for a certain period of time, after which the sample & hold circuit is put on hold to keep the input analog voltage until A/D conversion is completed.

<4> Bit 7 of the successive approximation register (SAR) is set. The series resistor string tap voltage at the tap selector is set to half of AV_{DD}.

<5> The series resistor string tap voltage is compared with the analog input voltage using the voltage comparator. If the analog input voltage is higher than half of AV_{DD}, the MSB of SAR is left set. If it is lower than half of AV_{DD}, the MSB is reset.

<6> Bit 6 of SAR is set automatically, and comparison shifts to the next stage. The next tap voltage of the series resistor string is selected according to bit 7, which reflects the previous comparison result, as follows.

- Bit 7 = 1: Three quarters of AV_{DD}
- Bit 7 = 0: One quarter of AV_{DD}

The tap voltage is compared with the analog input voltage. Bit 6 is set or reset according to the result of comparison.

- Analog input voltage ≥ tap voltage: Bit 6 = 1
- Analog input voltage < tap voltage: Bit 6 = 0

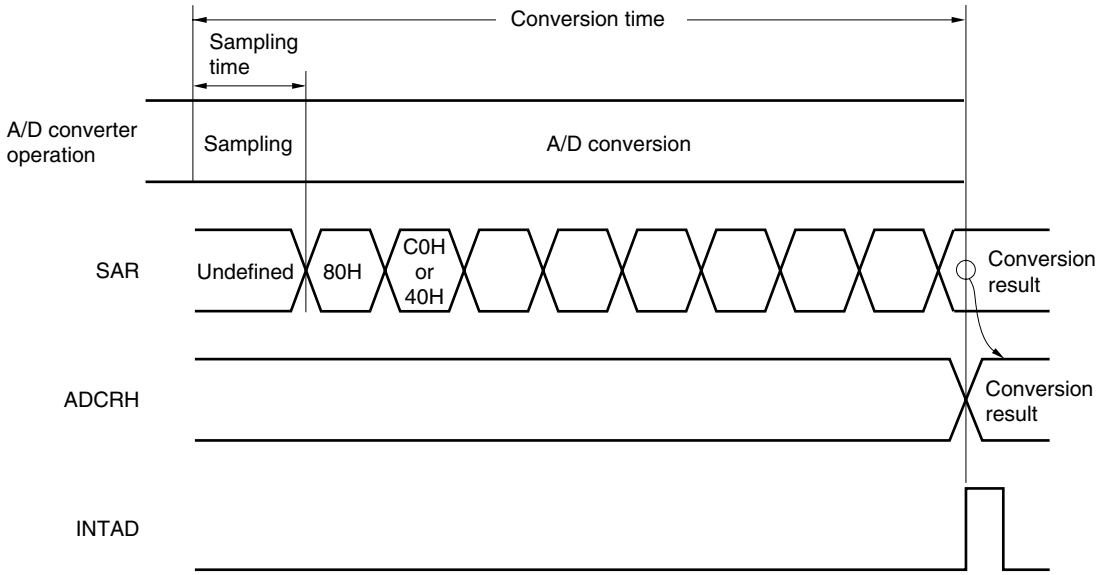
<7> Comparison is repeated until bit 0 of SAR is reached.

<8> When comparison is completed for all of the eight bits, a significant digital result is left in SAR. This value is sent to and latched in the A/D conversion result register (ADCRH). At the same time, it is possible to generate an A/D conversion end interrupt request (INTAD).

Cautions 1. The first A/D conversion value immediately after A/D conversion has been started is undefined.

2. In standby mode, A/D converter operation is stopped.

Figure 10-4. Basic Operation of A/D Converter



A/D conversion continues until bit 7 (ADCS) of the A/D converter mode register (ADM) is reset to 0 by software.

If an attempt is made to write to ADM or the A/D input selection register (ADS) during A/D conversion, the A/D conversion in progress is canceled. In this case, A/D conversion is restarted from the beginning, if ADCS is set to 1.

$\overline{\text{RESET}}$ input makes the A/D conversion result register (ADCRH) undefined.

10.4.2 Input voltage and conversion result

The relationship between the analog input voltage at the analog input pins (ANI0 to ANI7) and the A/D conversion result (A/D conversion result register (ADCRH)) is represented by:

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

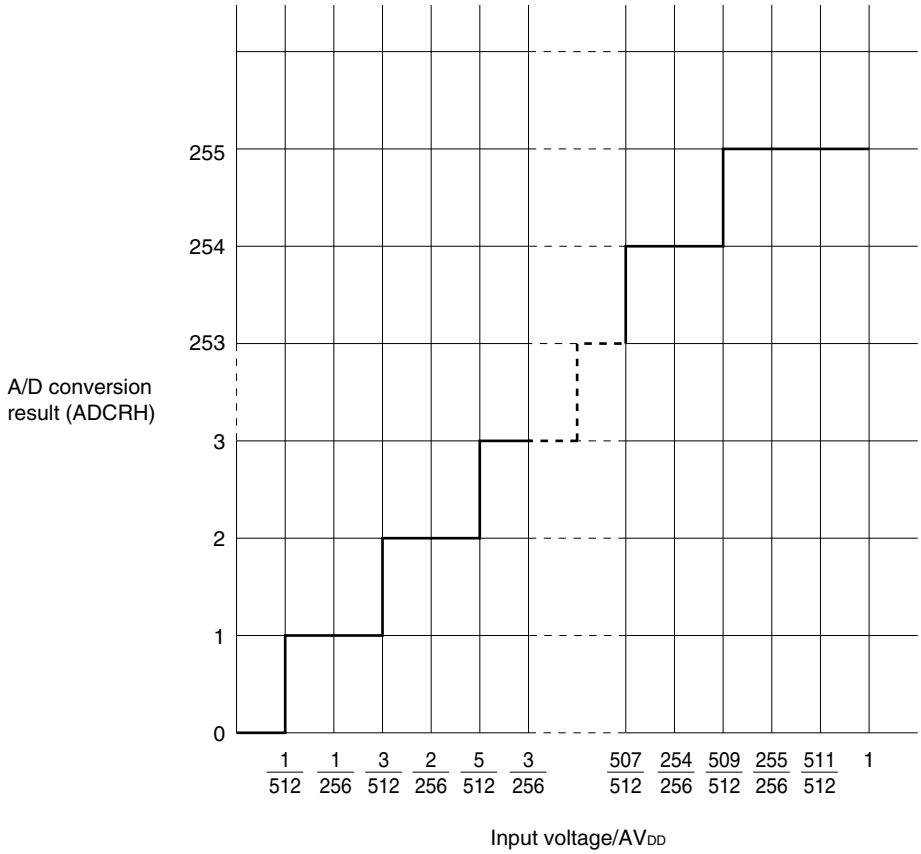
or

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

INT( ) : Function that returns the integer part of the parenthesized value
V_{IN}: Analog input voltage
AV_{DD}: Voltage of AV_{DD} pin
ADCRH: Value in the A/D conversion result register (ADCRH)

Figure 10-5 shows the relationship between the analog input voltage and the A/D conversion result.

Figure 10-5. Relationship Between Analog Input Voltage and A/D Conversion Result



10.4.3 Operation mode of A/D converter

The A/D converter is initially in select mode. In this mode, the A/D input selection register (ADS) is used to select an analog input channel from ANI0 to ANI7 for A/D conversion.

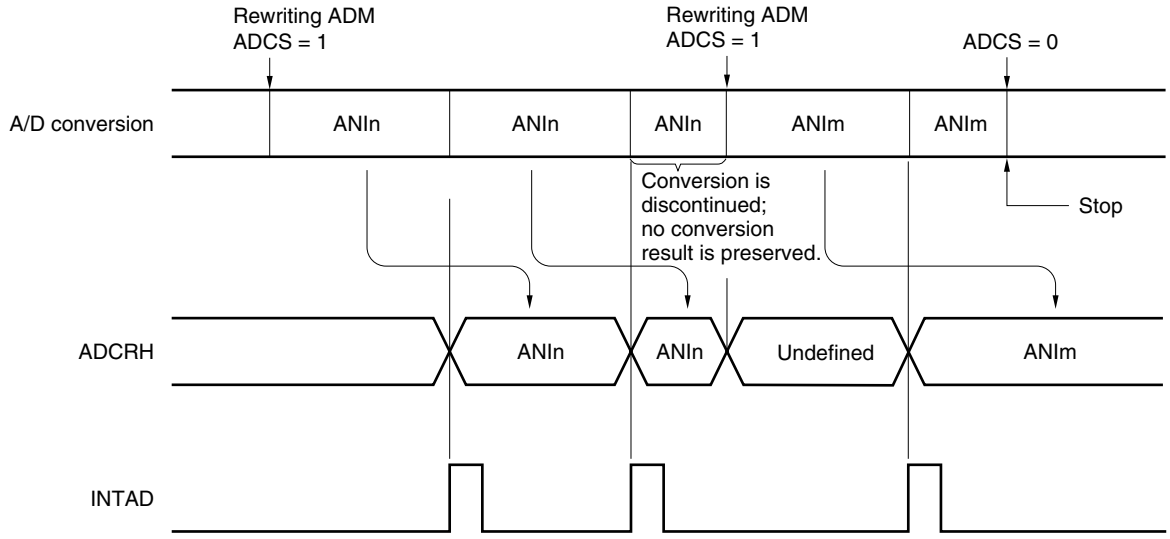
A/D conversion can be started only by software, that is, by setting the A/D converter mode register (ADM).

The A/D conversion result is saved to the A/D conversion result register (ADCRH). At the same time, an interrupt request signal (INTAD) is generated.

- **Software-started A/D conversion**

  Setting bit 7 (ADCS) of the A/D converter mode register (ADM) to 1 triggers A/D conversion for a voltage applied to the analog input pin specified in the A/D input selection register (ADS). Upon completion of A/D conversion, the conversion result is saved to the A/D conversion result register (ADCRH). At the same, an interrupt request signal (INTAD) is generated. Once A/D conversion is activated, and completed, another session of A/D conversion is started. A/D conversion is repeated until new data is written to ADM. If data where ADCS is 1 is written to ADM again during A/D conversion, the session of A/D conversion in progress is discontinued, and a new session of A/D conversion begins for the new data. If data where ADCS is 0 is written to ADM again during A/D conversion, A/D conversion is completely stopped.

★ **Figure 10-6. Software-Started A/D Conversion**



Remarks 1. n = 0 to 7

2. m = 0 to 7

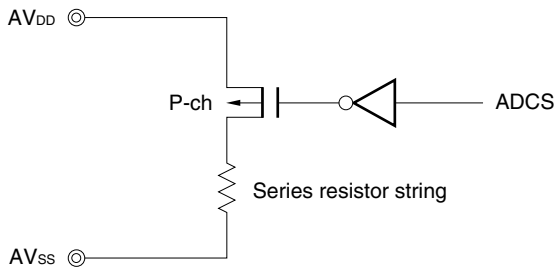
10.5 Notes on Using A/D Converter

(1) Current drain in standby mode

When the A/D converter enters standby mode, it stops operating. Clearing bit 7 (ADCS) of the A/D converter mode register (ADM) to 0 reduces the current drain.

Figure 10-7 shows how to reduce the current drain in standby mode.

Figure 10-7. How to Reduce Current Drain in Standby Mode



(2) Input range for the ANI0 to ANI7 pins

Be sure to keep the input voltage at ANI0 to ANI7 within the rated range. If a voltage AV_{DD} or higher or AV_{SS} or lower (even within the absolute maximum ratings) is input to a conversion channel, the conversion output of the channel becomes undefined. This may also affect the conversion output of the other channels.

(3) Conflict

<1> Conflict between writing to the A/D conversion result register (ADCRH) at the end of conversion and reading from ADCRH with an instruction

Reading from ADCRH takes precedence. After reading, the new conversion result is written to ADCRH.

<2> Conflict between writing to ADCRH at the end of conversion and writing to the A/D converter mode register (ADM) or the A/D input selection register (ADS)

Writing to ADM or ADS takes precedence. A request to write to ADCRH is ignored. No conversion end interrupt request signal (INTAD) is generated.

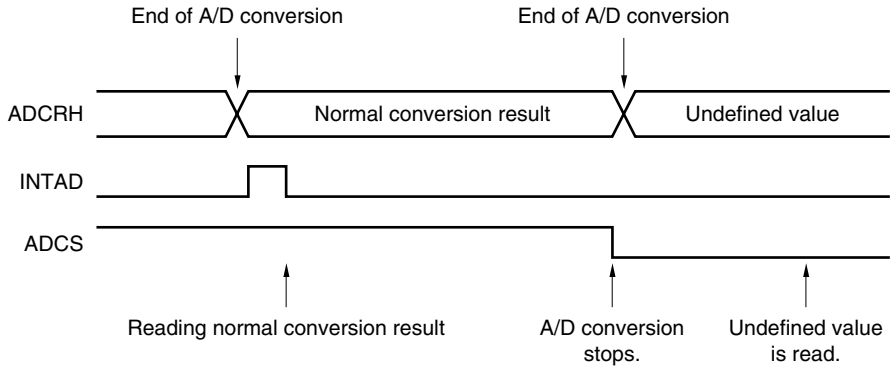
(4) Conversion result immediately after start of A/D conversion

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

(5) Timing of undefined A/D conversion result

The A/D conversion value may become undefined if the timing of the completion of A/D conversion and the timing of stopping the A/D conversion operation conflict. Therefore, read the A/D conversion result while the A/D conversion operation is in progress. Figure 10-8 shows the timing at which the conversion result is read.

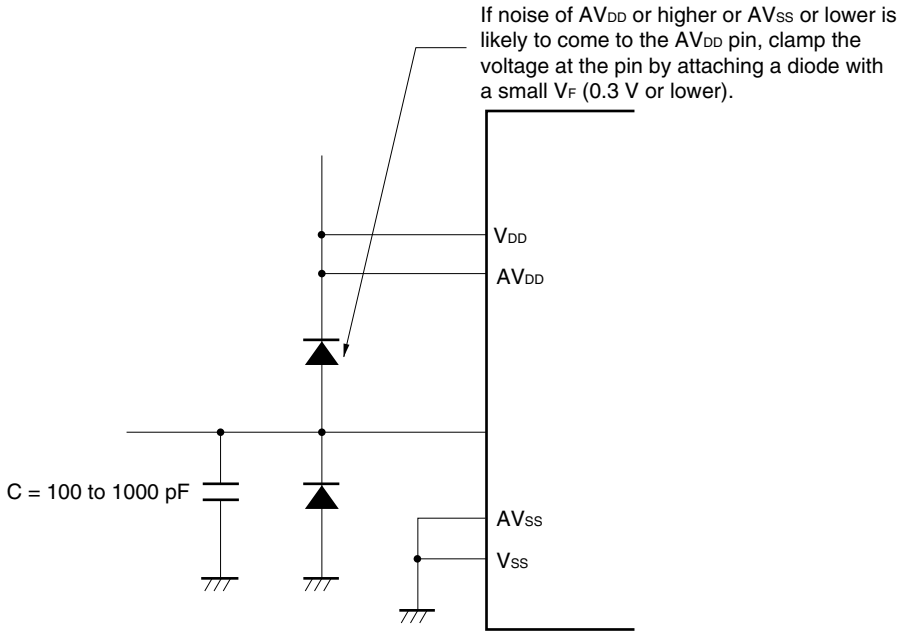
Figure 10-8. Conversion Result Read Timing (if Conversion Result Is Undefined)



(6) Noise prevention

To maintain a resolution of 8 bits, watch for noise at the AV_{DD} and ANI0 to ANI7 pins. The higher the output impedance of the analog input source, the larger the effect by noise. To reduce noise, attach an external capacitor to the relevant pins as shown in Figure 10-9.

Figure 10-9. Analog Input Pin Handling



(7) ANI0 to ANI7

The analog input pins (ANI0 to ANI7) are alternate-function pins. They are also used as port pins (P60 to P67).

If any of ANI0 to ANI7 has been selected for A/D conversion, do not execute input instructions for the ports; otherwise, the conversion resolution may become lower.

If a digital pulse is applied to a pin adjacent to the analog input pin in the process of A/D conversion, coupling noise may occur which prevents an A/D conversion result from being attained as expected. Avoid applying a digital pulse to pins adjacent to the analog input pin undergoing A/D conversion.

★ **(8) Input impedance of ANI0 to ANI7 pins**

This A/D converter executes sampling by charging the internal sampling capacitor for approximately 1/10 of the conversion time.

Therefore, only the leakage current flows during other than sampling, and the current for charging the capacitor flows during sampling. The input impedance therefore varies and has no meaning.

To achieve sufficient sampling, it is recommended that the output impedance of the analog input source be 10 kΩ or less, or attach a capacitor of around 100 pF to the ANI0 to ANI7 pins (see **Figure 10-9**).

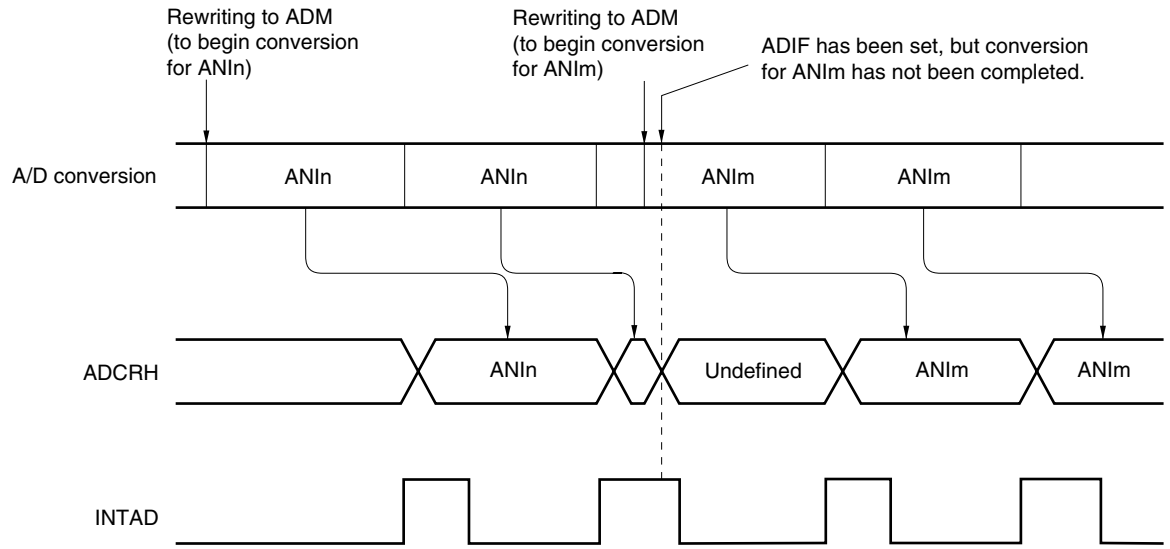
(9) Interrupt request flag (ADIF)

Changing the contents of the A/D converter mode register (ADM) does not clear ADIF (bit 4 of interrupt request flag register 1 (IF1)).

If the analog input pins are changed during A/D conversion, therefore, the conversion result and the conversion end interrupt request flag may reflect the previous analog input immediately before writing to ADM occurs. In this case, ADIF may appear to be set if it is read-accessed immediately after ADM is write-accessed, even when A/D conversion has not been completed for the new analog input.

In addition, when A/D conversion is restarted, ADIF must be cleared beforehand.

Figure 10-10. A/D Conversion End Interrupt Request Generation Timing



Remarks 1. n = 0 to 7

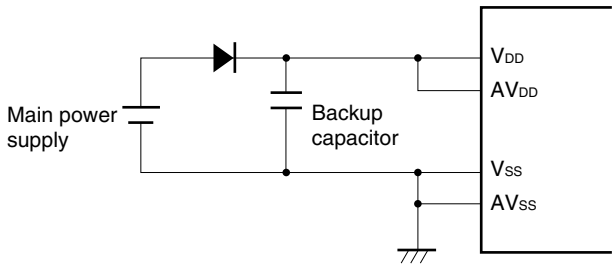
2. m = 0 to 7

(10) AV_{DD} pin

The AV_{DD} pin is used to supply power to the analog circuit. It is also used to supply power to the ANI0 to ANI7 input circuit.

If the application is designed to be changed to backup power, the AV_{DD} pin must be supplied with the same voltage level as the V_{DD} pin, as shown in Figure 10-11.

Figure 10-11. AV_{DD} Pin Handling



(11) Input impedance of the AV_{DD} pin

A series resistor string is connected across the AV_{DD} and AV_{SS} pins.

If the output impedance of the reference voltage source is high, this high impedance is eventually connected in parallel with the series resistor string across the AV_{DD} and AV_{SS} pins, leading to a higher reference voltage error.

11.1 Serial Interface Functions

The serial interface (UART00) has the following two modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode

(1) Operation stop mode

This mode is used when serial transfer is not performed. Power consumption is minimized in this mode.

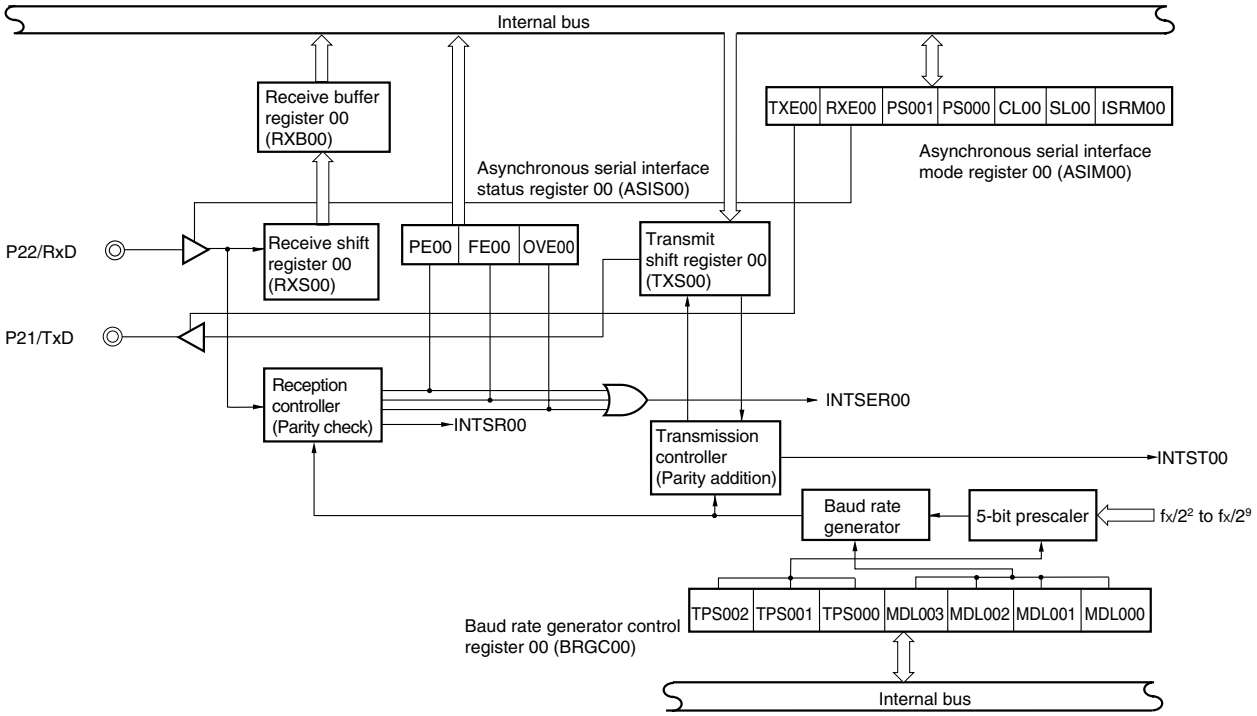
(2) Asynchronous serial interface (UART) mode

This mode is used to transmit and receive the one byte of data that follows a start bit. It supports full-duplex communication.

The serial interface contains a UART-dedicated baud rate generator, enabling communication over a wide range of baud rates. The UART-dedicated baud rate generator also enables the use of a MIDI standard baud rate (31.25 kbps).

Figure 11-1 shows a block diagram of the serial interface (UART00).

Figure 11-1. Block Diagram of Serial Interface



11.2 Serial Interface Configuration

The serial interface (UART00) consists of the following hardware.

Table 11-1. Serial Interface Configuration

Item	Configuration
Registers	Transmit shift register 00 (TXS00) Receive shift register 00 (RXS00) Receive buffer register 00 (RXB00)
Control registers	Asynchronous serial interface mode register 00 (ASIM00) Asynchronous serial interface status register 00 (ASIS00) Baud rate generator control register 00 (BRGC00) Port mode register 2 (PM2) Port register 2 (P2)

(1) Transmit shift register 00 (TXS00)

TXS00 is a register in which transmission data is prepared. The transmit data is output from TXS00 bit-serially.

When the data length is seven bits, bits 0 to 6 of the data in TXS00 will be transmit data. Writing data to TXS00 triggers transmission.

TXS00 can be write-accessed, using an 8-bit memory manipulation instruction, but cannot be read-accessed.

$\overline{\text{RESET}}$ input makes TXS00 undefined.

Caution Do not write to TXS00 during transmission.
TXS00 and receive buffer register 00 (RXB00) are mapped at the same address, so any attempt to read from TXS00 results in a value being read from RXB00.

(2) Receive shift register 00 (RXS00)

RXS00 is a register in which serial data, received at the RxD pin, is converted to parallel data. Once one entire byte has been received, RXS00 feeds the receive data to receive buffer register 00 (RXB00). RXS00 cannot be manipulated directly by a program.

(3) Receive buffer register 00 (RXB00)

RXB00 is used to hold receive data. Once receive shift register 00 (RXS00) has received one entire byte of data, it feeds that data into RXB00.

When the data length is seven bits, the receive data is sent to bits 0 to 6 of RXB00, in which the MSB is fixed to 0.

RXB00 can be read-accessed, using an 8-bit memory manipulation instruction, but cannot be write-accessed.

$\overline{\text{RESET}}$ input sets RXB00 to FFH.

Caution RXB00 and transmit shift register 00 (TXS00) are mapped at the same address, so any attempt to write to RXB00 results in a value being written to TXS00.

(4) Transmission controller

The transmission controller controls transmission. For example, it adds start, parity, and stop bits to the data in transmit shift register 00 (TXS00), according to the setting of asynchronous serial interface mode register 00 (ASIM00).

(5) Reception controller

The reception controller controls reception according to the setting of asynchronous serial interface mode register 00 (ASIM00). It also checks for errors, such as parity errors, during reception. If an error is detected, asynchronous serial interface status register 00 (ASIS00) is set according to the status of the error.

11.3 Registers Controlling Serial Interface

The following three registers are used to control the serial interface (UART00).

- Asynchronous serial interface mode register 00 (ASIM00)
- Asynchronous serial interface status register 00 (ASIS00)
- Baud rate generator control register 00 (BRGC00)

(1) Asynchronous serial interface mode register 00 (ASIM00)

ASIM00 is an 8-bit register that is used to control the serial transfer operation of the serial interface (UART00).

ASIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIM00 to 00H.

Caution When using the serial interface function (UART mode), set the related output latches to 0, and the port mode registers (PM××) as follows.

- **For reception**
  Set P22 (RxD) to input mode (PM22 = 1).
- **For transmission**
  Set P21 (TxD) to output mode (PM21 = 0).
- **For transmission and reception**
  Set P22 and P21 to input and output mode, respectively.

Figure 11-2. Format of Asynchronous Serial Interface Mode Register 00

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM00	TXE00	RXE00	PS001	PS000	CL00	SL00	ISRM00	0	FF70H	00H	R/W

TXE00	RXE00	Operation mode	Function of RxD/P22 pin	Function of TxD/P21 pin
0	0	Operation disabled	Port function (P22)	Port function (P21)
0	1	UART mode (reception only)	Serial function (RxD)	
1	0	UART mode (transmission only)	Port function (P22)	Serial function (TxD)
1	1	UART mode (transmission and reception)	Serial function (RxD)	

PS001	PS000	Parity bit specification
0	0	No parity
0	1	At transmission, the parity bit is fixed to 0. At reception, a parity check is not made; no parity error is reported.
1	0	Odd parity
1	1	Even parity

CL00	Character length specification
0	7 bits
1	8 bits

SL00	Transmission data stop bit length specification
0	1 bit
1	2 bits

ISRM00	Reception completion interrupt control at error occurrence
0	A reception completion interrupt request is generated at error occurrence.
1	A reception completion interrupt request is not generated at error occurrence.

Cautions 1. Be sure to clear bit 0 to 0.

2. When rewriting ASIM00 to the value other than the same data, stop the operation before rewriting.

(2) Asynchronous serial interface status register 00 (ASIS00)

ASIS00 is used to display the type of a receive error, if it occurs while UART mode is set.

ASIS00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIS00 to 00H.

Figure 11-3. Format of Asynchronous Serial Interface Status Register 00

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ASIS00	0	0	0	0	0	PE00	FE00	OVE00	FF71H	00H	R

PE00	Parity error flag
0	Parity error did not occur
1	Parity error occurred (when the transmission parity and reception parity did not match)

FE00	Framing error flag
0	Framing error did not occur
1	Framing error occurred[Note 1] (when stop bit was not detected)

OVE00	Overrun error flag
0	Overrun error did not occur
1	Overrun error occurred[Note 2] (when the next receive operation was completed before the data was read from receive buffer register 00)

Notes 1. Even when the stop bit length is set to 2 bits by setting bit 2 (SL00) of asynchronous serial interface mode register 00 (ASIM00), the stop bit detection in the case of reception is performed with 1 bit.

2. Until receive buffer register 00 (RXB00) is read when an overrun error occurs, an overrun error continues to occur.

Caution Be sure to clear bits 3 to 7 to 0.

(3) Baud rate generator control register 00 (BRGC00)

BRGC00 is used to specify the serial clock for the serial interface.

BRGC00 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears BRGC00 to 00H.

Figure 11-4. Format of Baud Rate Generator Control Register 00

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC00	0	TPS002	TPS001	TPS000	MDL003	MDL002	MDL001	MDL000	FF72H	00H	R/W

TPS002	TPS001	TPS000	Baud rate generator source clock (f_{SCK}) selection
0	0	0	$f_X/2^2$ (2.10 MHz)
0	0	1	$f_X/2^3$ (1.05 MHz)
0	1	0	$f_X/2^4$ (524 kHz)
0	1	1	$f_X/2^5$ (262 kHz)
1	0	0	$f_X/2^6$ (131 kHz)
1	0	1	$f_X/2^7$ (65.5 kHz)
1	1	0	$f_X/2^8$ (32.7 kHz)
1	1	1	$f_X/2^9$ (16.4 kHz)

MDL003	MDL002	MDL001	MDL000	Baud rate generator output clock selection
0	0	0	0	$f_{SCK}/16$
0	0	0	1	$f_{SCK}/17$
0	0	1	0	$f_{SCK}/18$
0	0	1	1	$f_{SCK}/19$
0	1	0	0	$f_{SCK}/20$
0	1	0	1	$f_{SCK}/21$
0	1	1	0	$f_{SCK}/22$
0	1	1	1	$f_{SCK}/23$
1	0	0	0	$f_{SCK}/24$
1	0	0	1	$f_{SCK}/25$
1	0	1	0	$f_{SCK}/26$
1	0	1	1	$f_{SCK}/27$
1	1	0	0	$f_{SCK}/28$
1	1	0	1	$f_{SCK}/29$
1	1	1	0	$f_{SCK}/30$
1	1	1	1	Setting prohibited

Cautions 1. Be sure to clear bit 7 to 0.

2. When writing to BRGC00 is performed during a communication operation, the output of the baud rate generator is disrupted and communications cannot be performed normally. Be sure not to write to BRGC00 during a communication operation.

Remarks 1. f_X: System clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 8.38 MHz.

3. f_{SCK}: Source clock of the baud rate generator

11.4 Serial Interface Operation

The serial interface (UART00) provides the following two modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode

11.4.1 Operation stop mode

In operation stop mode, serial transfer is not executed; therefore, the power consumption can be reduced. In this mode, the pins can be used as normal I/O ports.

(1) Register setting

Operation mode is set by asynchronous serial interface mode register 00 (ASIM00).

ASIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIM00 to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM00	TXE00	RXE00	PS001	PS000	CL00	SL00	ISRM00	0	FF70H	00H	R/W

TXE00	RXE00	Operation mode	Function of RxD/P22 pin	Function of TxD/P21 pin
0	0	Operation disabled	Port function (P22)	Port function (P21)
0	1	UART mode (reception only)	Serial function (RxD)	
1	0	UART mode (transmission only)	Port function (P22)	Serial function (TxD)
1	1	UART mode (transmission and reception)	Serial function (RxD)	

Caution Switch the operation mode after stopping both serial transmission and reception.

11.4.2 Asynchronous serial interface (UART) mode

In this mode, the one-byte data following the start bit is transmitted/received and thus full-duplex communication is possible.

The serial interface contains a UART-dedicated baud rate generator that enables communications at the desired baud rate from many options.

The UART-dedicated baud rate generator also can output the 31.25 kbps baud rate that complies with the MIDI standard.

(1) Register setting

UART mode is set by asynchronous serial interface mode register 00 (ASIM00), asynchronous serial interface status register 00 (ASIS00), baud rate generator control register 00 (BRGC00), port mode register 2 (PM2), and port register 2 (P2).

(a) Asynchronous serial interface mode register 00 (ASIM00)

ASIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIM00 to 00H.

Caution When using the asynchronous serial interface function (UART mode), set the related output latches to 0, and port mode register 2 (PM2) as follows.

- **For reception**
  Set P22 (RxD) to input mode (PM22 = 1).
- **For transmission**
  Set P21 (TxD) to output mode (PM21 = 0).
- **For transmission and reception**
  Set P22 and P21 to input and output mode, respectively.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM00	TXE00	RXE00	PS001	PS000	CL00	SL00	ISRM00	0	FF70H	00H	R/W

TXE00	RXE00	Operation mode	Function of RxD/P22 pin	Function of TxD/P21 pin
0	0	Operation disabled	Port function (P22)	Port function (P21)
0	1	UART mode (reception only)	Serial function (RxD)	
1	0	UART mode (transmission only)	Port function (P22)	Serial function (TxD)
1	1	UART mode (transmission and reception)	Serial function (RxD)	

PS001	PS000	Parity bit specification
0	0	No parity
0	1	At transmission, the parity bit is fixed to 0. At reception, a parity check is not made; no parity error is reported.
1	0	Odd parity
1	1	Even parity

CL00	Character length specification
0	7 bits
1	8 bits

SL00	Transmission data stop bit length specification
0	1 bit
1	2 bits

ISRM00	Reception completion interrupt control at error occurrence
0	A reception completion interrupt request is generated at error occurrence.
1	A reception completion interrupt request is not generated at error occurrence.

Cautions 1. Be sure to clear bit 0 to 0.

2. When rewriting ASIM00 to the value other than the same data, stop the operation before rewriting.

(b) Asynchronous serial interface status register 00 (ASIS00)

ASIS00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIS00 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ASIS00	0	0	0	0	0	PE00	FE00	OVE00	FF71H	00H	R

PE00	Parity error flag
0	Parity error did not occur
1	Parity error occurred (when the transmission parity and reception parity did not match)

FE00	Framing error flag
0	Framing error did not occur
1	Framing error occurred[Note 1] (when stop bit was not detected)

OVE00	Overrun error flag
0	Overrun error did not occur
1	Overrun error occurred[Note 2] (when the next receive operation was completed before the data was read from receive buffer register 00)

Notes 1. Even when the stop bit length is set to 2 bits by setting bit 2 (SL00) of asynchronous serial interface mode register 00 (ASIM00), the stop bit detection in the case of reception is performed with 1 bit.

2. Until receive buffer register 00 (RXB00) is read when an overrun error occurs, an overrun error continues to occur.

Caution Be sure to clear bits 3 to 7 to 0.

(c) Baud rate generator control register 00 (BRGC00)

BRGC00 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears BRGC00 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC00	0	TPS002	TPS001	TPS000	MDL003	MDL002	MDL001	MDL000	FF72H	00H	R/W

TPS002	TPS001	TPS000	Baud rate generator source clock selection	n
0	0	0	$f_X/2^2$ (2.10 MHz)	0
0	0	1	$f_X/2^3$ (1.05 MHz)	1
0	1	0	$f_X/2^4$ (524 kHz)	2
0	1	1	$f_X/2^5$ (262 kHz)	3
1	0	0	$f_X/2^6$ (131 kHz)	4
1	0	1	$f_X/2^7$ (65.5 kHz)	5
1	1	0	$f_X/2^8$ (32.7 kHz)	6
1	1	1	$f_X/2^9$ (16.4 kHz)	7

MDL003	MDL002	MDL001	MDL000	Baud rate generator output clock selection	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	15

Cautions 1. Be sure to clear bit 7 to 0.

2. When writing to BRGC00 is performed during a communication operation, the output of the baud rate generator is disrupted and communications cannot be performed normally. Be sure not to write to BRGC00 during a communication operation.

Remarks 1. f_X: System clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 8.38 MHz.

3. f_{SCK}: Source clock of the baud rate generator

The baud rate transmit/receive clock to be generated is a divided system clock signal.

- **Generation of baud rate transmit/receive clock by means of system clock**

  The transmit/receive clock is generated by dividing the system clock. The baud rate generated from the system clock is estimated by using the following expression.

  $$[\text{Baud rate}] = \frac{f_X}{2^{n+1}(k+16)} \quad [\text{bps}]$$

  fx: System clock oscillation frequency

Table 11-2 shows the relationship between the source clock of the baud rate generator assigned to bits 4 to 6 (TPS000 to TPS002) of BRGC00, and value n.

Table 11-2. Relationship Between Source Clock of Baud Rate Generator and Value n

TPS002	TPS001	TPS000	Baud Rate Generator Source Clock Selection	n
0	0	0	$f_X/2^2$ (2.10 MHz)	0
0	0	1	$f_X/2^3$ (1.05 MHz)	1
0	1	0	$f_X/2^4$ (524 kHz)	2
0	1	1	$f_X/2^5$ (262 kHz)	3
1	0	0	$f_X/2^6$ (131 kHz)	4
1	0	1	$f_X/2^7$ (65.5 kHz)	5
1	1	0	$f_X/2^8$ (32.7 kHz)	6
1	1	1	$f_X/2^9$ (16.4 kHz)	7

Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

- **Permissible error range of baud rate**

  The permissible error range of the baud rate is dependent upon the number of bits of one frame and the division ratio of the counter [1/(16 + k)]. Table 11-3 shows the relationship between the system clock and baud rate.

Table 11-3. Example of Relationship Between System Clock and Baud Rate

Baud Rate [bps]	f_X = 8.38 MHz			
	BRGC00	Error (%)	n	k
300	7BH	1.0	8	11
600	6BH		7	11
1,200	5BH		6	11
2,400	4BH		5	11
4,800	3BH		4	11
9,600	2BH		3	11
19,200	1BH		2	11
31,250	11H	–1.4	2	1
38,400	0BH	1.0	1	11

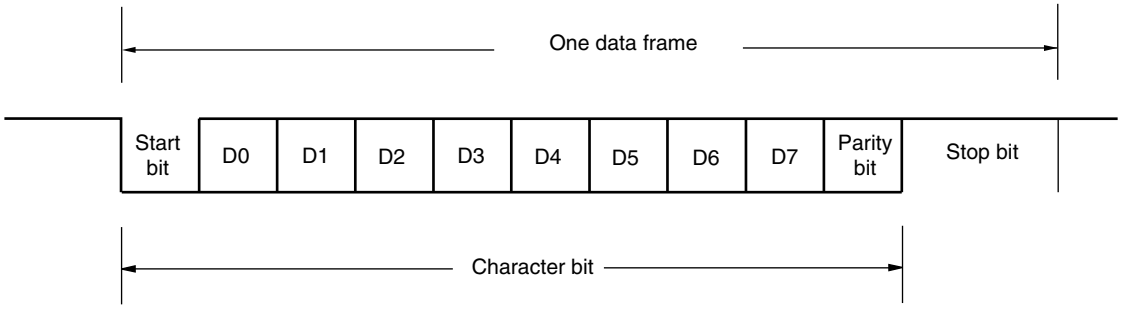
Remark f_X: System clock oscillation frequency

(2) Communication operation

(a) Data format

The transmit/receive data format is as shown in Figure 11-5.

Figure 11-5. Format of Asynchronous Serial Interface Transmit/Receive Data



One data frame consists of the following bits:

- Start bit: 1 bit
- Character bits: 7 bits/8 bits
- Parity bits: Even parity/odd parity/0 parity/no parity
- Stop bit(s) : 1 bit/2 bits

The specification of the character bit length, parity selection, and stop bit length for each data frame is carried out using asynchronous serial interface mode register 00 (ASIM00).

When 7 bits are selected as the number of character bits, only the lower 7 bits (bits 0 to 6) are valid; in transmission the most significant bit (bit 7) is ignored, and in reception the most significant bit (bit 7) is always “0”.

The serial transfer rate is selected by means of baud rate generator control register 00 (BRGC00).

If a serial data receive error is generated, the receive error contents can be determined by reading the status of asynchronous serial interface status register 00 (ASIS00).

(b) Parity types and operation

The parity bit is used to detect a bit error in the communication data. Normally, the same parity bit is used on the transmitting side and the receiving side. With even parity and odd parity, a “1” bit (odd number) error can be detected. With 0 parity and no parity, an error cannot be detected.

(i) Even parity

- **At transmission**

  The transmit operation is controlled so that the number of character bits with a value of “1” in the transmit data including parity bit is even. The parity bit value should be as follows.

  The number of character bits with a value of “1” is an odd number in transmit data: 1
  The number of character bits with a value of “1” is an even number in transmit data: 0

- **At reception**

  The number of character bits with a value of “1” in the reception data including parity bit is counted, and if the number is odd, a parity error occurs.

(ii) Odd parity

- **At transmission**

  Opposite to even parity, the transmit operation is controlled so that the number of character bits with a value of “1” in the transmit data including parity bit is odd. The parity bit value should be as follows.

  The number of character bits with a value of “1” is an odd number in transmit data: 0
  The number of character bits with a value of “1” is an even number in transmit data: 1

- **At reception**

  The number of character bits with a value of “1” in the receive data including parity bit is counted, and if the number is even, a parity error occurs.

(iii) 0 parity

When transmitting, the parity bit is set to “0” irrespective of the transmit data.

When receiving, a parity bit check is not performed. Therefore, a parity error does not occur, irrespective of whether the parity bit is set to “0” or “1”.

(iv) No parity

A parity bit is not added to the transmit data.

At reception, data is received assuming that there is no parity bit. Since there is no parity bit, a parity error does not occur.

(c) Transmission

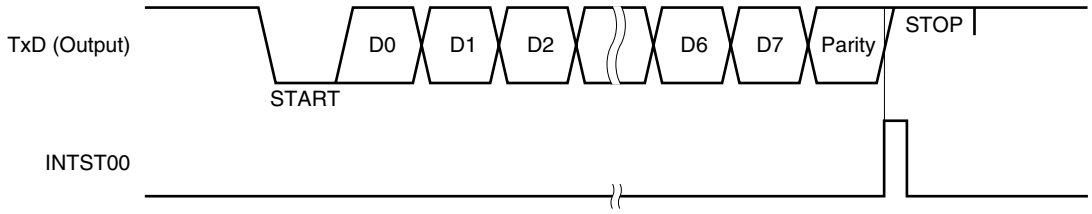
★ A transmit operation is enabled by setting bit 7 (TXE00) of asynchronous serial interface mode register 00 (ASIM00) to 1, and started by writing transmit data to transmit shift register 00 (TXS00). The start bit, parity bit, and stop bit(s) are added automatically.

When the transmit operation starts, the data in TXS00 is shifted out, and when TXS00 is empty, a transmission completion interrupt request (INTST00) is generated.

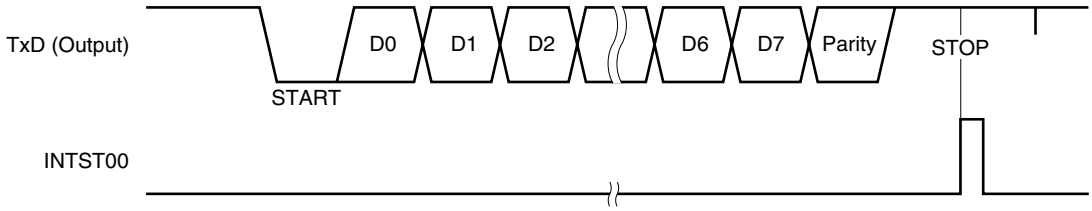
The transmission completion interrupt timing is shown in Figure 11-6.

Figure 11-6. Asynchronous Serial Interface Transmission Completion Interrupt Request Timing

(i) Stop bit length: 1



(ii) Stop bit length: 2



Caution Do not rewrite asynchronous serial interface mode register 00 (ASIM00) during a transmit operation. If the ASIM00 register is written during transmission, subsequent transmission may not be performed (the normal state is restored by $\overline{\text{RESET}}$ input).

(d) Reception

A receive operation is performed via level detection.

When bit 6 (RXE00) of asynchronous serial interface mode register 00 (ASIM00) is set to 1, the receive operation is enabled and sampling of the RxD pin input is performed.

RxD pin input sampling is performed using the serial clock specified by BRGC00.

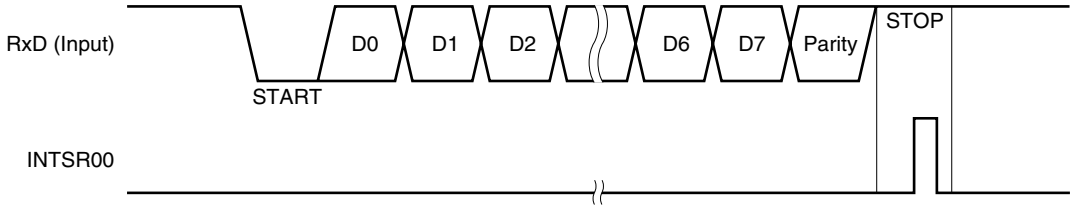
When the RxD pin input becomes low, the 5-bit counter of the baud rate generator starts counting, and at the time when half the time determined by the specified baud rate has passed, the data sampling start timing signal is output. If the RxD pin input sampled again as a result of this start timing signal is low, it is identified as a start bit, the 5-bit counter is initialized and starts counting, and data sampling is performed. When character data, a parity bit, and one stop bit are detected after the start bit, reception of one frame of data ends.

When one frame of data has been received, the receive data in the shift register is transferred to receive buffer register 00 (RXB00), and INTSR00 (reception completion interrupt request) is generated.

If the RXE00 bit is cleared to 0 during the receive operation, the receive operation is stopped immediately. In this case, the contents of RXB00 and ASIS00 are not changed, and INTSR00 and INTSER00 (receive error interrupt requests) are not generated.

Figure 11-7 shows the asynchronous serial interface reception completion interrupt request timing.

Figure 11-7. Asynchronous Serial Interface Reception Completion Interrupt Request Timing



★ **Caution If a receive operation is enabled when the RxD pin input is low level, a receive operation is immediately started. Therefore, be sure to enable a receive operation after making the RxD pin input high level.**

(e) Receive errors

The following three errors may occur during a receive operation: a parity error, framing error, or overrun error. If the error flag in asynchronous serial interface status register 00 (ASIS00) is set to 1 as a result of data reception, a receive error interrupt request (INTSER00) is generated. The receive error interrupt occurs before the reception completion interrupt request (INTSR00). Receive error causes are shown in Table 11-4.

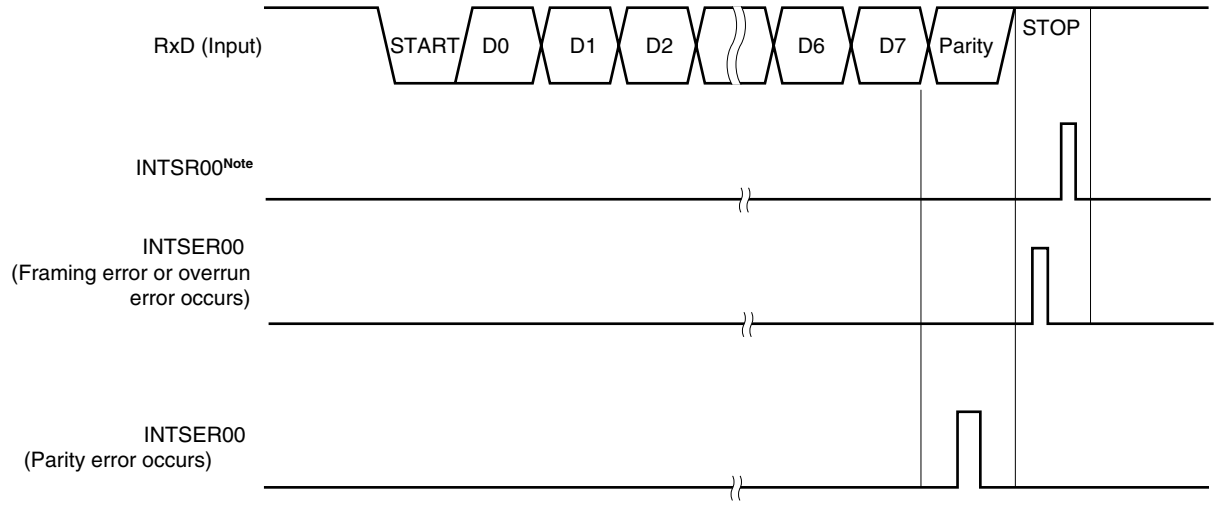
It is possible to determine what kind of error occurred during reception by reading the contents of ASIS00 (see **Figure11-3**).

The contents of ASIS00 are cleared to 0 by reading receive buffer register 00 (RXB00) or receiving the next data (if there is an error in the next data, the corresponding error flag is set).

Table 11-4. Receive Error Causes

Receive Errors	Cause
Parity error	Transmission-time parity specification and receive data parity do not match
Framing error	Stop bit not detected
Overrun error	Reception of next data is completed before data is read from receive buffer register 00

Figure 11-8. Receive Error Timing



Note If a receive error occurs when the ISRM00 bit is set to 1, INTSR00 is not generated.

Cautions 1. The contents of asynchronous serial interface status register 00 (ASIS00) are cleared to 0 by reading receive buffer register 00 (RXB00) or receiving the next data. To ascertain the error contents, read ASIS00 before reading RXB00.

2. Be sure to read receive buffer register 00 (RXB00) even if a receive error occurs. If RXB00 is not read, an overrun error will occur when the next data is received, and the receive error state will continue indefinitely.

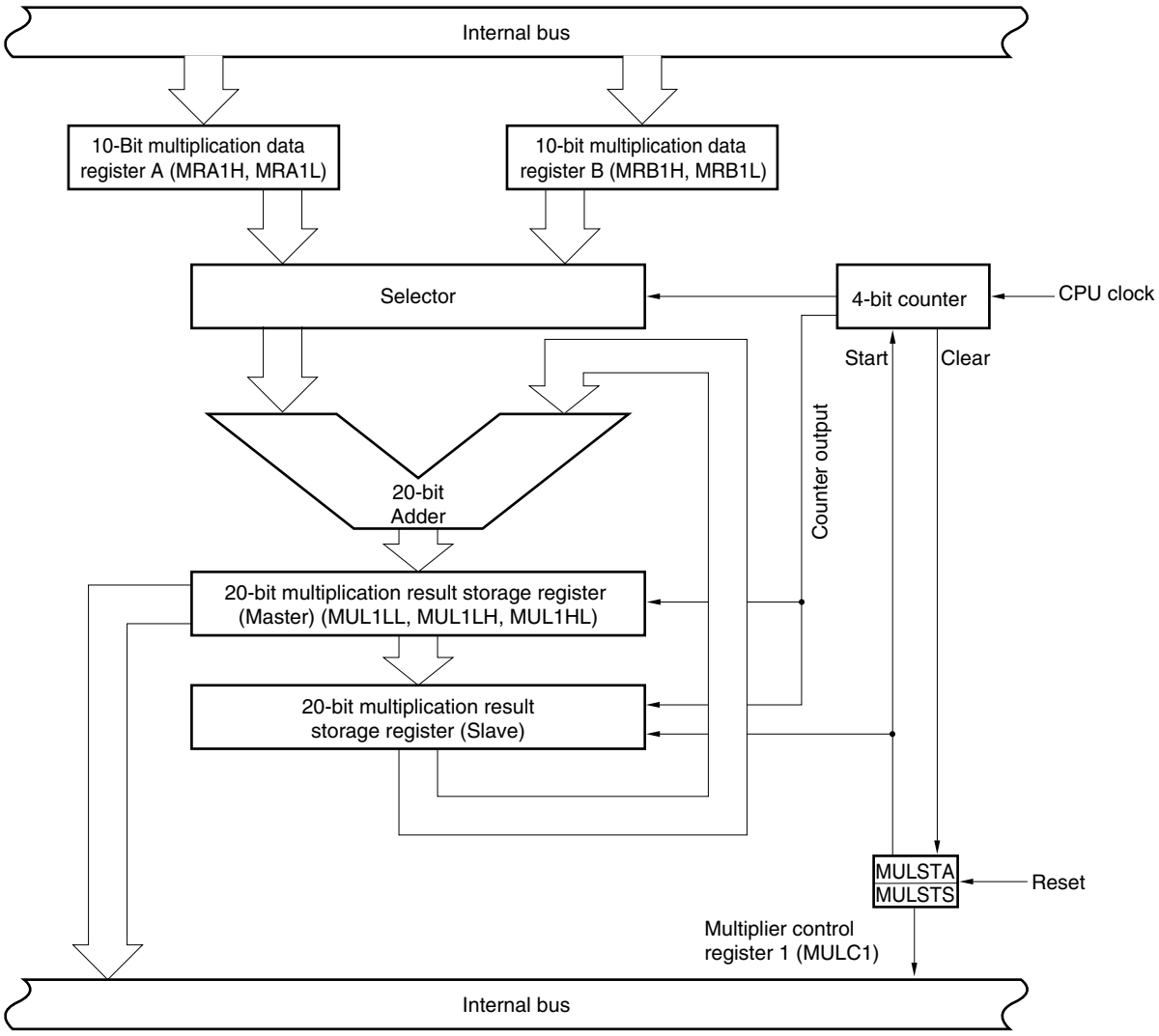
12.1 Multiplier Function

The multiplier has the following function.

- Calculation of 10 bits × 10 bits = 20 bits

12.2 Multiplier Configuration

Figure 12-1. Block Diagram of Multiplier



(1) 20-bit multiplication result storage registers (MUL1LL, MUL1LH, and MUL1HL)

These registers store the 20-bit result of multiplication.

MUL1LL, MUL1LH, and MUL1HL are set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes these registers undefined.

Figure 12-2. Format of 20-Bit Multiplication Result Storage Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MUL1LL	MUL1LL7	MUL1LL6	MUL1LL5	MUL1LL4	MUL1LL3	MUL1LL2	MUL1LL1	MUL1LL0	FFA5H	Undefined	R
MUL1LH	MUL1LH7	MUL1LH6	MUL1LH5	MUL1LH4	MUL1LH3	MUL1LH2	MUL1LH1	MUL1LH0	FFA6H	Undefined	R
MUL1HL	0	0	0	0	MUL1HL3	MUL1HL2	MUL1HL1	MUL1HL0	FFA7H	Undefined	R

(2) 10-bit multiplication data registers (MRA1H, MR1L, MRB1H, and MRB1L)

These are 10-bit multiplication data storage registers. The multiplier multiplies the value of MRA1H and MRA1L by that of MRB1H and MRB1L.

The 10-bit data registers are set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes these registers undefined.

Figure 12-3. Format of 10-Bit Data Register A

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MRA1L	MRA1L7	MRA1L6	MRA1L5	MRA1L4	MRA1L3	MRA1L2	MRA1L1	MRA1L0	FFA1H	Undefined	W
MRA1H	0	0	0	0	0	0	MRA1H1	MRA1H0	FFA2H	Undefined	W

Caution Be sure to clear bits 2 to 7 of MRA1H to 0.

Figure 12-4. Format of 10-Bit Data Register B

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MRB1L	MRB1L7	MRB1L6	MRB1L5	MRB1L4	MRB1L3	MRB1L2	MRB1L1	MRB1L0	FFA3H	Undefined	W
MRB1H	0	0	0	0	0	0	MRB1H1	MRB1H0	FFA4H	Undefined	W

Caution Be sure to clear bits 2 to 7 of MRB1H to 0.

12.3 Register Controlling Multiplier

The multiplier is controlled by the following register.

- Multiplier control register 1 (MULC1)

MULC1 indicates the operating status of the multiplier, as well as controls the multiplier.

The operation of the multiplier ends 20 × f_{CPU} after it starts.

MULC1 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 12-5. Format of Multiplier Control Register 1 in Write Mode

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MULC1	0	0	0	0	0	0	0	MULSTA	FFA0H	00H	W

MULSTA	Multiplier operation start control bit
0	Stop operation
1	Start operation

Figure 12-6. Format of Multiplier Control Register 1 in Read Mode

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MULC1	0	0	0	0	0	0	0	MULSTS	FFA0H	00H	R

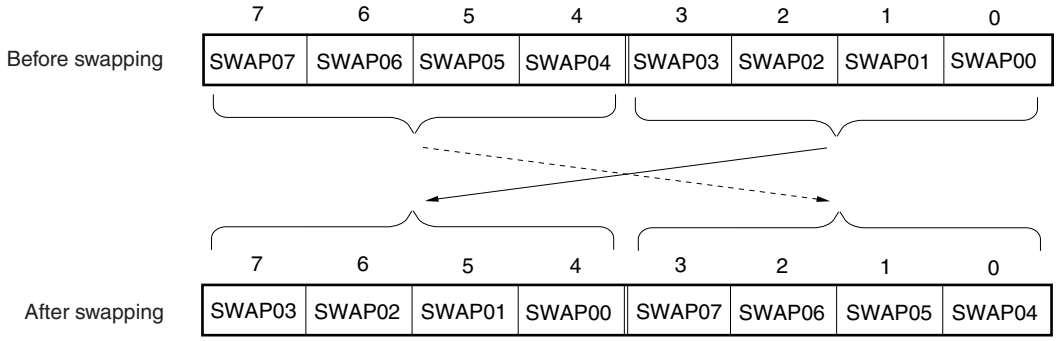
MULSTS	Status of multiplier
0	Operation terminated
1	Operation in progress

CHAPTER 13 SWAPPING (SWAP)

13.1 SWAP Function

By performing four shift operations, it is possible to switch the contents of the higher four bits of swapping function register 0 (SWP0) with the lower four bits. Figure 13-1 shows an example of swapping.

Figure 13-1. Example of Swapping



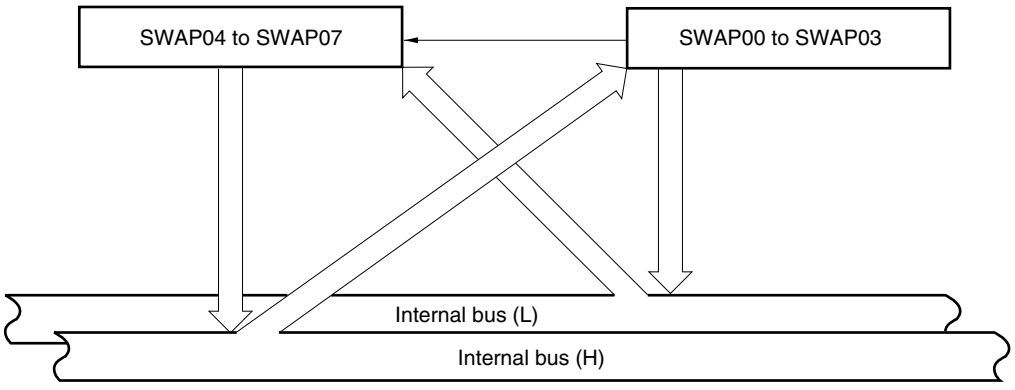
13.2 SWAP Configuration

SWAP consists of the following hardware.

Table 13-1. SWAP Configuration

Item	Configuration
Register	Swapping function register 0 (SWP0)

Figure 13-2. SWAP Block Diagram



(1) Swapping function register 0 (SWP0)

By writing data into SWP0 and subsequently reading it back, the contents of the higher four bits and the lower four bits can be swapped.

SWP0 is set with an 8-bit memory manipulation instruction.

In write mode, $\overline{\text{RESET}}$ input makes SWP0 undefined.

In read mode, $\overline{\text{RESET}}$ input clears SWP0 to 00H.

CHAPTER 14 INTERRUPT FUNCTIONS

14.1 Interrupt Function Types

The following two types of interrupt functions are used.

(1) Non-maskable interrupt

This interrupt is acknowledged even when interrupts are disabled. It does not undergo interrupt priority control and is given top priority over all other interrupt requests.

★ A standby release signal is generated and HALT mode is released.

The only non-maskable interrupts the interrupt request from the watchdog timer.

(2) Maskable interrupt

These interrupts undergo mask control. If two or more interrupts are simultaneously generated, each interrupt has a predetermined priority as shown in Table 14-1.

★ A standby release signal is generated and STOP and HALT modes are released.

Two external interrupt request sources and 11 internal interrupt request sources are available as maskable interrupts.

14.2 Interrupt Sources and Configuration

A total of 14 non-maskable and maskable interrupts are incorporated as interrupt sources.

Table 14-1. Interrupt Sources

Interrupt Type	Priority[Note 1]	Interrupt Source		Internal/External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable interrupt	–	INTWDT	Watchdog timer overflow (when watchdog timer mode 1 is selected)	Internal	0004H	(A)
Maskable interrupt	0	INTWDT	Watchdog timer overflow (when interval timer mode is selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTTM7	Generation of underflow signal for 10-bit inverter control timer	Internal	000AH	(B)
	4	INTSER00	Receive error on serial interface (UART00)		000CH	
	5	INTSR00	Completion of serial interface (UART00) reception		000EH	
	6	INTST00	Completion of serial interface (UART00) transmission		0010H	
	7	INTWT	Watch timer interrupt		0012H	
	8	INTWTI	Interval timer interrupt		0014H	
	9	INTTM80	Generation of match signal for 8-bit timer/event counter 80		0016H	
	10	INTTM81	Generation of match signal for 8-bit timer/event counter 81		0018H	
	11	INTTM82	Generation of match signal for 8-bit timer 82		001AH	
	12	INTAD	A/D conversion completion signal		001CH	

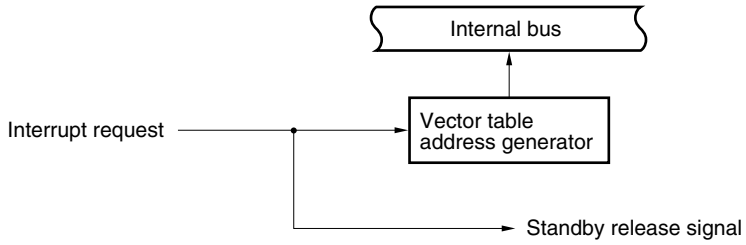
Notes 1. The priority regulates which maskable interrupt has priority when two or more maskable interrupts are requested simultaneously. Zero signifies the highest priority and 12 the lowest.

2. Basic configuration types (A), (B), and (C) correspond to (A), (B), and (C) in Figure 14-1.

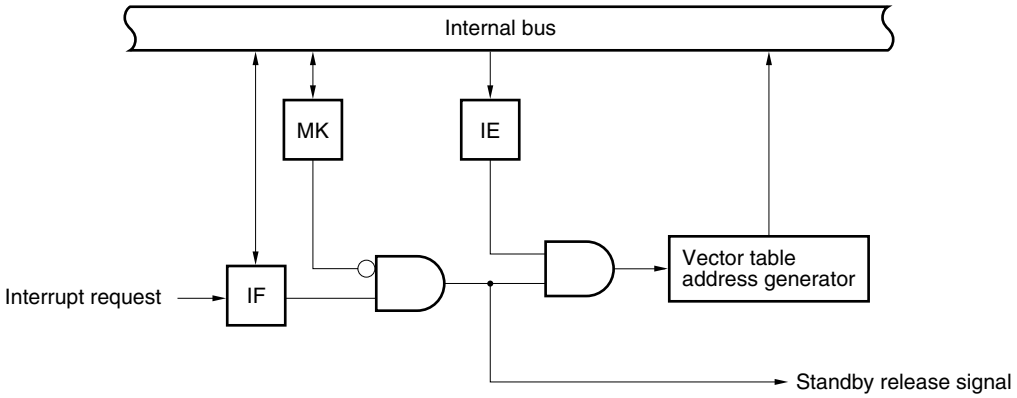
★ **Remark** There are two interrupt sources for the watchdog timer (INTWDT): a non-maskable interrupt (internal) and a maskable interrupt (internal). Either one (but not both) should be selected for actual use.

Figure 14-1. Basic Configuration of Interrupt Function

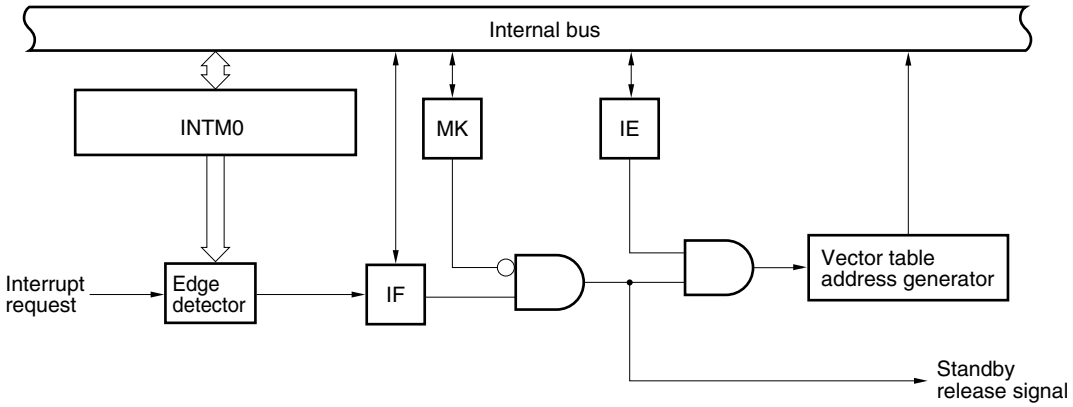
(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt



INTM0: External interrupt mode register 0
IF: Interrupt request flag
IE: Interrupt enable flag
MK: Interrupt mask flag

14.3 Registers Controlling Interrupt Function

The interrupt functions are controlled by the following registers.

- Interrupt request flag registers 0 and 1 (IF0 and IF1)
- Interrupt mask flag registers 0 and 1 (MK0 and MK1)
- External interrupt mode register 0 (INTM0)
- Program status word (PSW)

Table 14-2 lists the interrupt requests, corresponding interrupt request flags, and interrupt mask flags.

Table 14-2. Interrupt Request Signals and Corresponding Flags

Interrupt Request Signal	Interrupt Request Flag	Interrupt Mask Flag
INTWDT	TMIF4	TMMK4
INTP0	PIF0	PMK0
INTP1	PIF1	PMK1
INTTM7	TMIF7	TMMK7
INTSER00	SERIF00	SERMK00
INTSR00	SRIF00	SRMK00
INTST00	STIF00	STMK00
INTWT	WTIF	WTMK
INTWTI	WTIIF	WTIMK
INTTM80	TMIF80	TMMK80
INTTM81	TMIF81	TMMK81
INTTM82	TMIF82	TMMK82
INTAD	ADIF	ADMK

(1) Interrupt request flag registers 0 and 1 (IF0 and IF1)

An interrupt request flag is set to 1 when the corresponding interrupt request is issued, or when the related instruction is executed. It is cleared to 0 when the interrupt request is acknowledged, when a $\overline{\text{RESET}}$ signal is input, or when a related instruction is executed.

IF0 and IF1 are manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears IF0 and IF1 to 00H.

Figure 14-2. Format of Interrupt Request Flag Register

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
IF0	WTIF	STIF00	SRIF00	SERIF00	TMIF7	PIF1	PIF0	TMIF4	FFE0H	00H	R/W

	7	6	5	<4>	<3>	<2>	<1>	<0>			
IF1	0	0	0	ADIF	TMIF82	TMIF81	TMIF80	WTIIF	FFE1H	00H	R/W

××IF	Interrupt request flag
0	No interrupt request signal has been issued.
1	An interrupt request signal has been issued; an interrupt request has been made.

Cautions 1. Be sure to clear bits 5 to 7 of IF1 to 0.

2. The TMIF4 flag can be read- and write-accessed only when the watchdog timer is being used as an interval timer. It must be cleared to 0 if the watchdog timer is used in watchdog timer mode 1 or 2.

3. When port 2 is being used as an output port, and its output level is changed, an interrupt request flag is set, because this port is also used as an external interrupt input. To use port 2 in output mode, therefore, the interrupt mask flag must be set to 1 in advance.

(2) Interrupt mask flag registers 0 and 1 (MK0 and MK1)

The interrupt mask flags are used to enable and disable the corresponding maskable interrupts.

MK0 and MK1 are manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets MK0 and MK1 to FFH.

Figure 14-3. Format of Interrupt Mask Flag Register

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
MK0	WTMK	STMK00	SRMK00	SERMK00	TMMK7	PMK1	PMK0	TMMK4	FFE4H	FFH	R/W

	7	6	5	<4>	<3>	<2>	<1>	<0>			
MK1	1	1	1	ADMK	TMMK82	TMMK81	TMMK80	WTIMK	FFE5H	FFH	R/W

××MK	Interrupt handling control
0	Enable interrupt handling
1	Disable interrupt handling

Cautions 1. Be sure to set bits 5 to 7 of MK1 to 1.

2. The TMMK4 flag can be read- and write-accessed only when the watchdog timer is being used as an interval timer. It must be cleared to 0 if the watchdog timer is used in watchdog timer mode 1 or 2.

3. When port 2 is being used as an output port, and its output level is changed, an interrupt request flag is set, because this port is also used as an external interrupt input. To use port 2 in output mode, therefore, the interrupt mask flag must be set to 1 in advance.

(3) External interrupt mode register 0 (INTM0)

INTM0 is used to specify a valid edge for INTP0 and INTP1.

INTM0 is manipulated with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears INTM0 to 00H.

Figure 14-4. Format of External Interrupt Mode Register 0

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
INTM0	0	0	ES11	ES10	ES01	ES00	0	0	FFECH	00H	R/W

ES11	ES10	INTP1 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES01	ES00	INTP0 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

Cautions 1. Be sure to clear bits 0, 1, 6, and 7 to 0.

2. Before setting INTM0, set the corresponding interrupt mask flag register to 1 to disable interrupts. To enable interrupts, clear to 0 the corresponding interrupt request flag, then the corresponding interrupt mask flag.

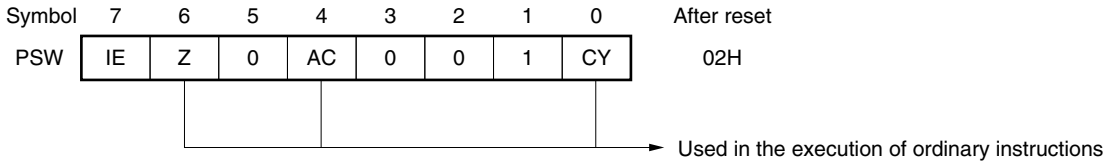
(4) Program status word (PSW)

The program status word is used to hold the instruction execution result and the current status of the interrupt requests. The IE flag, used to enable and disable maskable interrupts, is mapped to the PSW.

The PSW can be read- and write-accessed in 8-bit units, as well as in 1-bit units when using bit manipulation instructions and dedicated instructions (EI and DI). When a vectored interrupt is acknowledged, the PSW is automatically saved to a stack, and the IE flag is reset to 0.

$\overline{\text{RESET}}$ input sets PSW to 02H.

Figure 14-5. Program Status Word Configuration



IE	Whether to enable/disable interrupt acknowledgement
0	Disable
1	Enable

14.4 Interrupt Servicing Operation

14.4.1 Non-maskable interrupt request acknowledgment

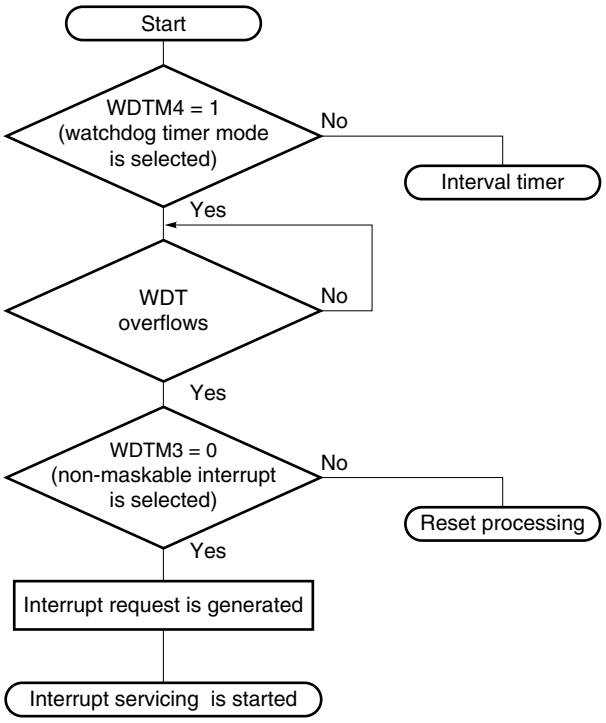
A non-maskable interrupt is unconditionally acknowledged even when interrupts are disabled. It is not subject to interrupt priority control and takes precedence over all other interrupts.

When a non-maskable interrupt request is acknowledged, the PSW and PC are saved to the stack in that order, the IE flag is reset to 0, the contents of the vector table are loaded to the PC, and then program execution branches.

Figure 14-6 shows the flowchart from non-maskable interrupt request generation to acknowledgment. Figure 14-7 shows the timing of non-maskable interrupt request acknowledgment. Figure 14-8 shows the acknowledgment operation if multiple non-maskable interrupts are generated.

Caution During a non-maskable interrupt servicing program execution, do not input another non-maskable interrupt request; if it is input, the servicing program will be interrupted and the new interrupt request will be acknowledged.

Figure 14-6. Flowchart from Non-Maskable Interrupt Request Generation to Acknowledgment



WDTM: Watchdog timer mode register
WDT: Watchdog timer

Figure 14-7. Timing of Non-Maskable Interrupt Request Acknowledgment

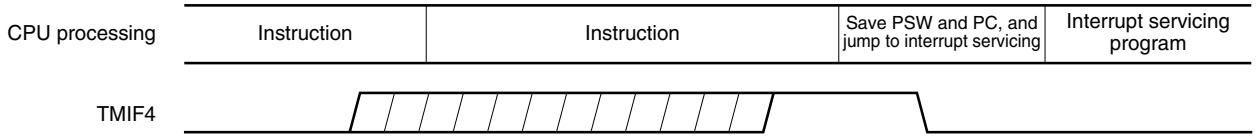
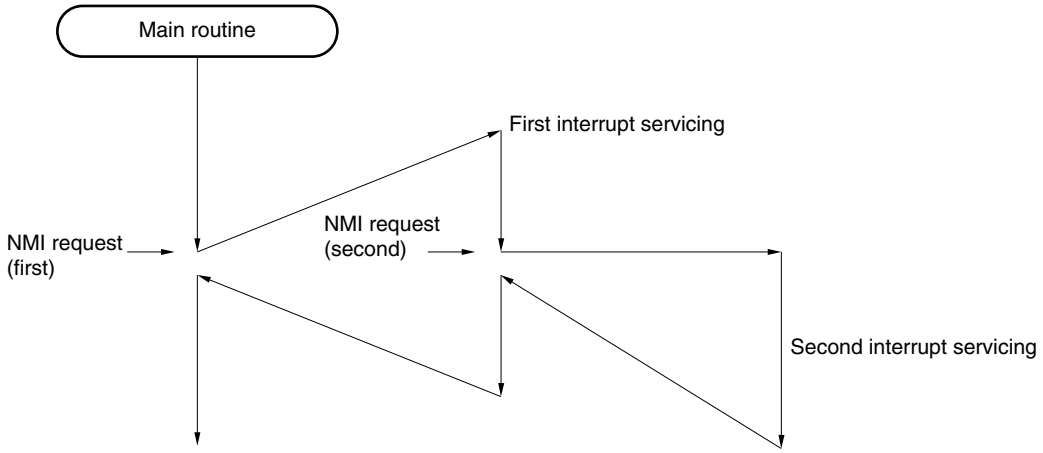


Figure 14-8. Acknowledging Non-Maskable Interrupt Request



14.4.2 Maskable interrupt request acknowledgment

A maskable interrupt can be acknowledged when the interrupt request flag is set to 1 and the corresponding interrupt mask flag is cleared to 0. A vectored interrupt is acknowledged in the interrupt enabled status (when the IE flag is set to 1).

The time required to start the interrupt servicing after a maskable interrupt request has been generated is shown in Table 14-3.

See **Figures 14-10** and **14-11** for the interrupt request acknowledgment timing.

Table 14-3. Time from Generation of Maskable Interrupt Request to Servicing

Minimum Time	Maximum Time[Note]
9 clocks	19 clocks

Note The wait time is maximum when an interrupt request is generated immediately before the BT and BF instructions.

Remark 1 clock: $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

When two or more maskable interrupt requests are generated at the same time, they are acknowledged starting from the interrupt request assigned the highest priority.

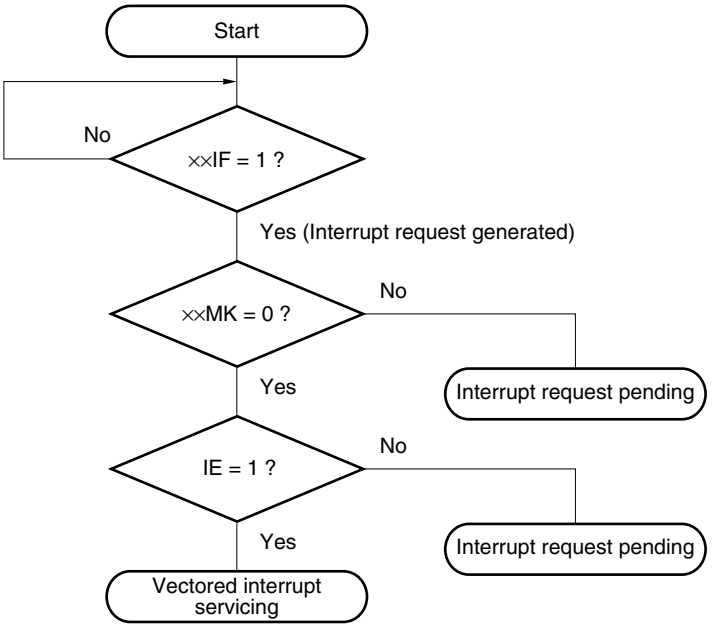
A pending interrupt is acknowledged when the status where it can be acknowledged is set.

Figure 14-9 shows the algorithm of acknowledging interrupt requests.

When a maskable interrupt request is acknowledged, the contents of PSW and PC are saved to the stack in that order, the IE flag is reset to 0, the data in the vector table determined for each interrupt request is loaded to the PC, and execution branches.

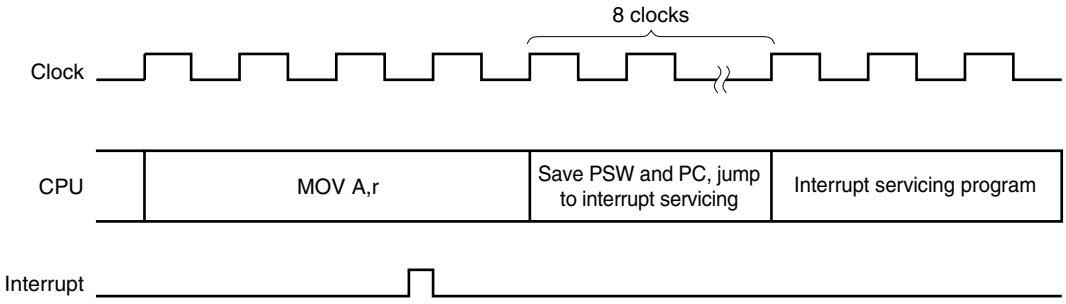
To return from interrupt servicing, use the RETI instruction.

Figure 14-9. Interrupt Request Acknowledgment Program Algorithm



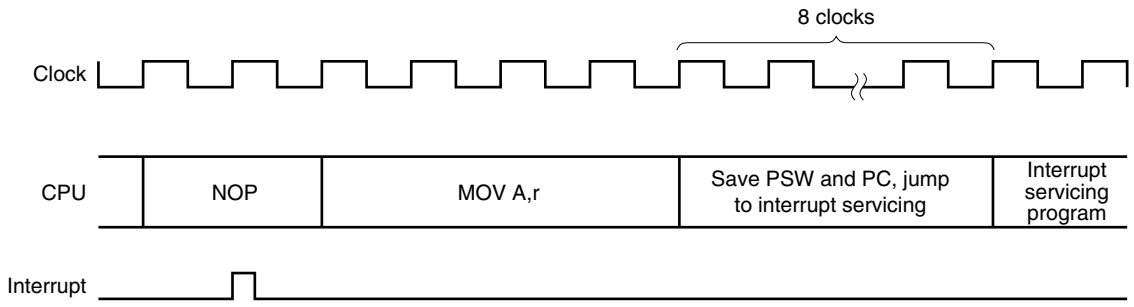
××IF: Interrupt request flag
××MK: Interrupt mask flag
IE: Flag to control maskable interrupt request acknowledgment (1 = Enable, 0 = Disable)

Figure 14-10. Interrupt Request Acknowledgment Timing (Example of MOV A,r)



If an interrupt request flag (××IF) is set before instruction clock n (n = 4 to 10) under execution becomes n − 1, the interrupt is acknowledged after the instruction under execution is complete. Figure 14-10 shows an example of the interrupt request acknowledgment timing for an 8-bit data transfer instruction MOV A,r. Since this instruction is executed for 4 clocks, if an interrupt occurs 3 clocks after the execution starts, the interrupt acknowledgment processing is performed after the MOV A,r instruction is completed.

Figure 14-11. Interrupt Request Acknowledgment Timing (When Interrupt Request Flag Is Generated at Last Clock During Instruction Execution)



If an interrupt request flag (××IF) is set at the last clock of the instruction, the interrupt acknowledgment processing starts after the next instruction is executed.

Figure 14-11 shows an example of the interrupt acknowledgment timing for an interrupt request flag that is set at the second clock of NOP (2-clock instruction). In this case, the MOV A,r instruction after the NOP instruction is executed, and then the interrupt acknowledgment processing is performed.

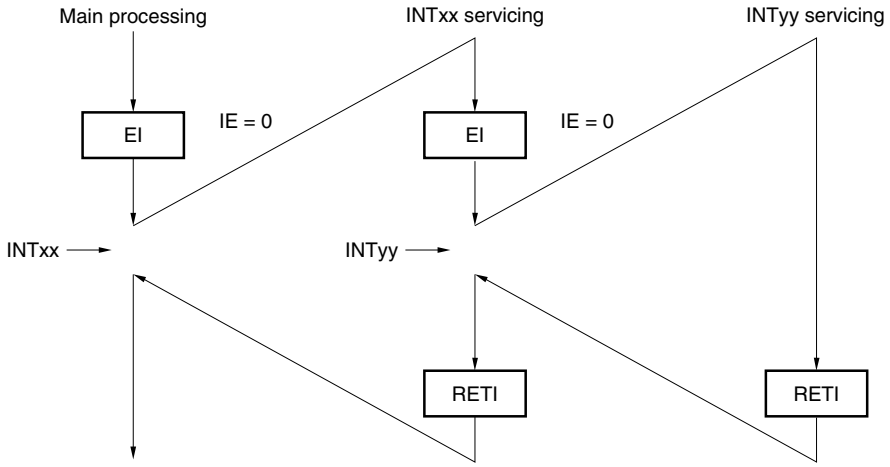
Caution Interrupt requests are held pending while interrupt request flag registers 0 and 1 (IF0 and IF1) or interrupt mask flag registers 0 and 1 (MK0 and MK1) are being accessed.

14.4.3 Multiple interrupt servicing

Multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being serviced can be processed by priority. When two or more interrupts are generated at once, interrupt servicing is performed according to the priority assigned to each interrupt request in advance (see **Table 14-1**).

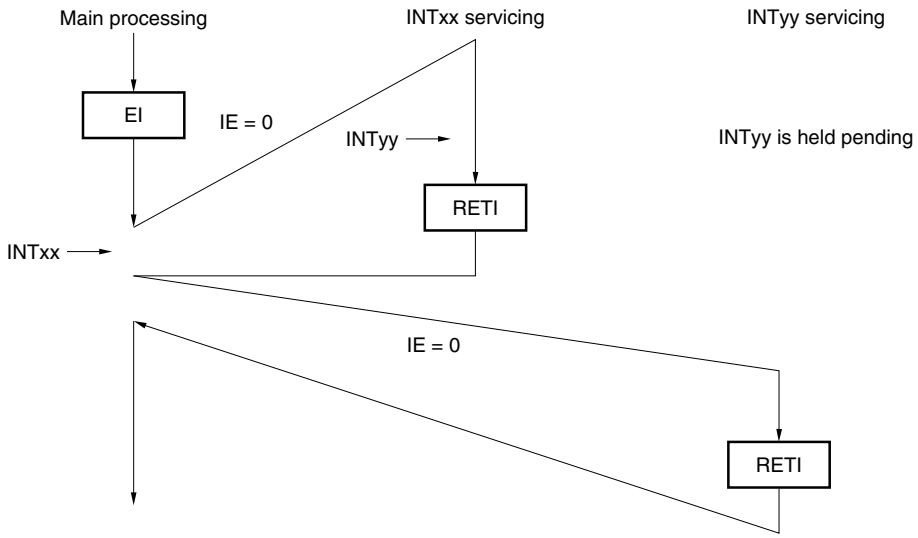
Figure 14-12. Example of Multiple Interrupt Servicing

Example 1. Multiple interrupts are acknowledged



During interrupt INTxx servicing, interrupt request INTyy is acknowledged, and multiple interrupt servicing is performed. The EI instruction is issued before each interrupt request acknowledgment, and the interrupt request acknowledgment enable state is set.

Example 2. Multiple interrupt servicing is not performed because interrupts are not enabled



Because interrupts are not enabled in interrupt INTxx servicing (the EI instruction is not issued), interrupt request INTyy is not acknowledged, and multiple interrupt servicing is not performed. The INTyy request is held pending and acknowledged after the INTxx servicing is performed.

Remark IE = 0: Interrupt request acknowledgment disabled

14.4.4 Interrupt request pending

Some instructions do not acknowledge an interrupt request (maskable interrupt, non-maskable interrupt, and external interrupt) until the completion of the execution of the next instruction even if the interrupt request is generated during the execution. The following shows such instructions (interrupt request pending instructions).

- Manipulation instruction for interrupt request flag registers 0 and 1 (IF0 and IF1)
- Manipulation instruction for interrupt mask flag registers 0 and 1 (MK0 and MK1)

CHAPTER 15 STANDBY FUNCTION

15.1 Standby Function and Configuration

15.1.1 Standby function

The standby function is used to reduce the power consumption of the system and can be effected in the following two modes.

(1) HALT mode

This mode is set when the HALT instruction is executed. HALT mode stops the operation clock of the CPU. The system clock oscillator continues oscillating. This mode does not reduce the power consumption as much as STOP mode, but is useful for resuming processing immediately when an interrupt request is generated, or for intermittent operations.

(2) STOP mode

This mode is set when the STOP instruction is executed. STOP mode stops the system clock oscillator and stops the entire system. The power consumption of the CPU can be substantially reduced in this mode.

STOP mode can be released by an interrupt request, so that this mode can be used for intermittent operations. However, some time is required until the system clock oscillator stabilizes after STOP mode has been released. If processing must be resumed immediately by using an interrupt request, therefore, use HALT mode.

In both modes, the previous contents of the registers, flags, and data memory before setting standby mode are all retained. In addition, the statuses of the output latch of the I/O ports and output buffer are also retained.

Caution To set STOP mode, be sure to stop the operations of the peripheral hardware, and then execute the STOP instruction.

15.1.2 Standby function control register

The wait time after STOP mode is released upon interrupt request generation until the oscillation stabilizes is controlled by the oscillation stabilization time selection register (OSTS).

OSTS is set with an 8-bit memory manipulation instruction.

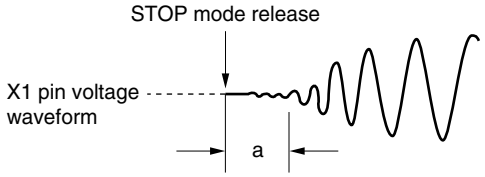
$\overline{\text{RESET}}$ input sets OSTS to 04H. However, the oscillation stabilization time after $\overline{\text{RESET}}$ input is 2^{15}/fx.

Figure 15-1. Format of Oscillation Stabilization Time Selection Register

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

OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection
0	0	0	2^{12}/fx (488 μs)
0	1	0	2^{15}/fx (3.91 ms)
1	0	0	2^{17}/fx (15.6 ms)
Other than above			Setting prohibited

Caution The wait time after STOP mode is released does not include the time from STOP mode release to clock oscillation start ("a" in the figure below), regardless of release by $\overline{\text{RESET}}$ input or by interrupt generation.



Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

15.2 Operation of Standby Function

15.2.1 HALT mode

(1) HALT mode

HALT mode is set by executing the HALT instruction.

The operation statuses in HALT mode are shown in the following table.

Table 15-1. Operation Statuses in HALT Mode

Item		HALT Mode Operation Status
System clock		System clock oscillation enabled Clock supply to CPU disabled
CPU		Operation disabled
Ports (output latches)		Remain in the state existing before the selection of HALT mode
10-bit inverter control timer		Operation enabled
8-bit timer/event counters 80, 81, 82	TM80	Operation enabled
	TM81	Operation enabled
	TM82	Operation enabled
Watch timer		Operation enabled
Watchdog timer		Operation enabled
A/D converter		Operation disabled
Serial interface		Operation enabled
External interrupt		Operation enabled[Note]

Note Maskable interrupt that is not masked

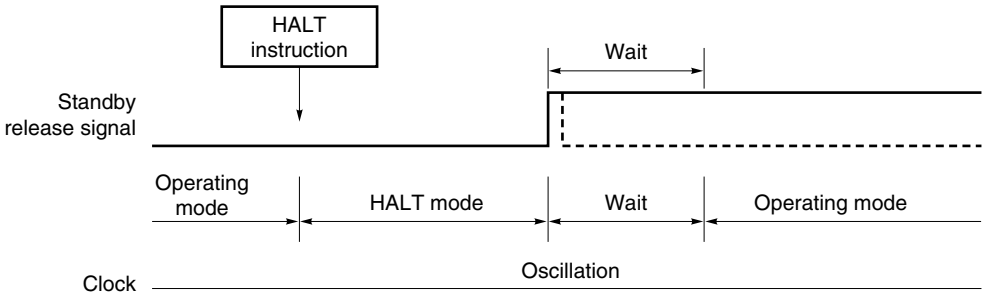
(2) Releasing HALT mode

HALT mode can be released by the following three sources.

(a) Releasing by unmasked interrupt request

HALT mode is released by an unmasked interrupt request. In this case, if the interrupt request is enabled to be acknowledged, vectored interrupt servicing is performed. If the interrupt is disabled, the instruction at the next address is executed.

Figure 15-2. Releasing HALT Mode by Interrupt



Remarks 1. The broken lines indicate the case where the interrupt request that has released standby mode is acknowledged.

2. The wait time is as follows.

- When vectored interrupt servicing is performed: 9 to 10 clocks
- When vectored interrupt servicing is not performed: 1 to 2 clocks

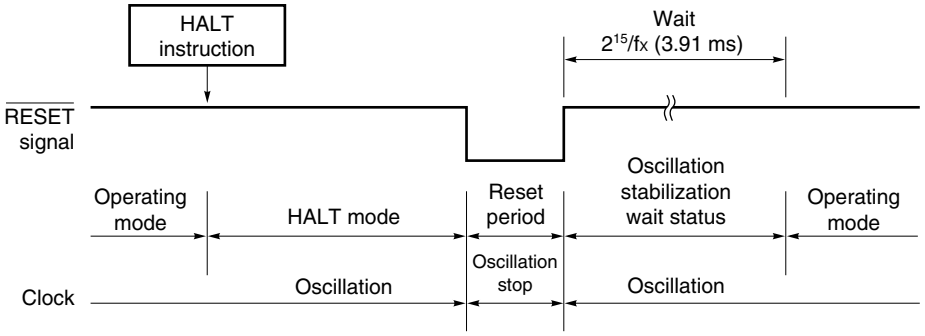
(b) Releasing by non-maskable interrupt request

HALT mode is released regardless of whether the interrupt is enabled or disabled, and vectored interrupt servicing is performed.

(c) Releasing by $\overline{\text{RESET}}$ input

When HALT mode is released by the $\overline{\text{RESET}}$ signal, execution branches to the reset vector address in the same manner as the ordinary reset operation, and program execution is started.

Figure 15-3. Releasing HALT Mode by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

Table 15-2. Operation After Release of HALT Mode

Releasing Source	MK××	IE	Operation
Maskable interrupt request	0	0	Next address instruction executed
	0	1	Interrupt servicing executed
	1	×	HALT mode retained
Non-maskable interrupt request	–	×	Interrupt servicing executed
$\overline{\text{RESET}}$ input	–	–	Reset processing

×: Don't care

15.2.2 STOP mode

(1) Setting and operation status of STOP mode

STOP mode is set by executing the STOP instruction.

Caution Because standby mode can be released by an interrupt request signal, standby mode is released as soon as it is set if there is an interrupt source whose interrupt request flag is set and interrupt mask flag is reset. When STOP mode is set, therefore, HALT mode is set immediately after the STOP instruction has been executed, the wait time set by the oscillation stabilization time selection register (OSTS) elapses, and then the operation mode is set.

The operation statuses in STOP mode are shown in the following table.

Table 15-3. Operation Statuses in STOP Mode

Item		STOP Mode Operation Status
System clock		System clock oscillation disabled
CPU		Operation disabled
Ports (output latches)		Remain in the state existing before the selection of STOP mode
10-bit inverter control timer		Operation disabled
8-bit timer/event counters 80, 81, 82	TM80	Operation enabled only when TI80 is selected as the count clock
	TM81	Operation enabled only when TI81 is selected as the count clock
	TM82	Operation disabled
Watch timer		Operation disabled
Watchdog timer		Operation disabled
A/D converter		Operation disabled
Serial interface		Operation disabled
External interrupt		Operation enabled[Note]

Note Maskable interrupt that is not masked

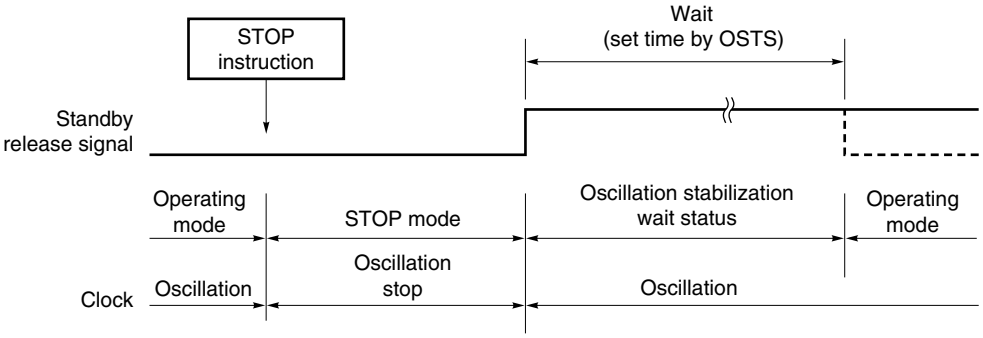
(2) Releasing STOP mode

STOP mode can be released by the following two sources.

(a) Releasing by unmasked interrupt request

STOP mode can be released by an unmasked interrupt request. In this case, if the interrupt is enabled to be acknowledged, vectored interrupt servicing is performed, after the oscillation stabilization time has elapsed. If the interrupt acknowledgment is disabled, the instruction at the next address is executed.

Figure 15-4. Releasing STOP Mode by Interrupt

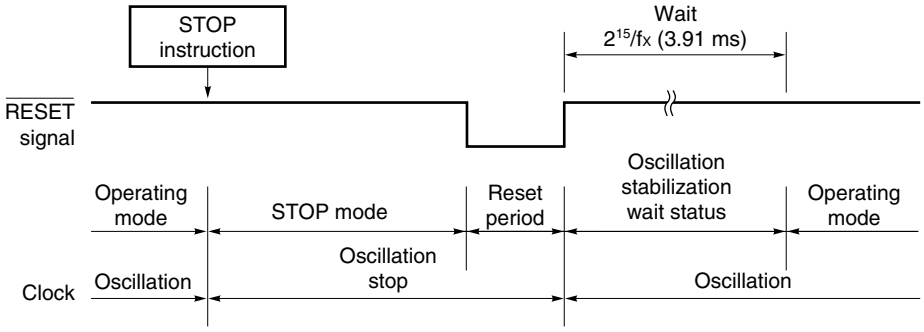


Remark The broken lines indicate the case where the interrupt request that has released standby mode is acknowledged.

(b) Releasing by $\overline{\text{RESET}}$ input

When STOP mode is released by the $\overline{\text{RESET}}$ signal, the reset operation is performed after the oscillation stabilization time has elapsed.

Figure 15-5. Releasing STOP Mode by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

2. The parenthesized values apply to operation at fx = 8.38 MHz.

Table 15-4. Operation After Release of STOP Mode

Releasing Source	MK××	IE	Operation
Maskable interrupt request	0	0	Next address instruction executed
	0	1	Interrupt servicing executed
	1	×	STOP mode retained
$\overline{\text{RESET}}$ input	–	–	Reset processing

×: Don't care

CHAPTER 16 RESET FUNCTION

The following two operations are available to generate reset signals.

(1) External reset input via $\overline{\text{RESET}}$ pin
(2) Internal reset by inadvertent program loop time detected by watchdog timer

External and internal reset have no functional differences. In both cases, program execution starts at the addresses at 0000H and 0001H by reset signal input.

When a low level is input to the $\overline{\text{RESET}}$ pin or the watchdog timer overflows, a reset is applied and the hardware is set to the status shown in Table 16-1. Each pin is high impedance during reset input or during the oscillation stabilization time just after reset is released.

When a high level is input to the $\overline{\text{RESET}}$ pin, the reset is released and program execution is started after the oscillation stabilization time (2^{15}/fx) has elapsed. The reset applied by watchdog timer overflow is automatically released after reset, and program execution is started after the oscillation stabilization time (2^{15}/fx) has elapsed (see **Figures 16-2** to **16-4**).

Cautions 1. For an external reset, input a low level to the $\overline{\text{RESET}}$ pin for 10 μs or more.

2. When STOP mode is released by reset, STOP mode contents are held during reset input. However, the port pins become high impedance.

Figure 16-1. Block Diagram of Reset Function

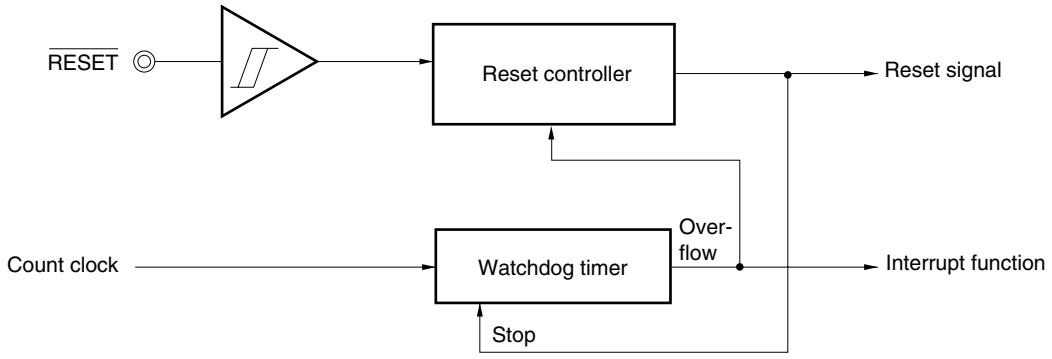


Figure 16-2. Reset Timing by $\overline{\text{RESET}}$ Input

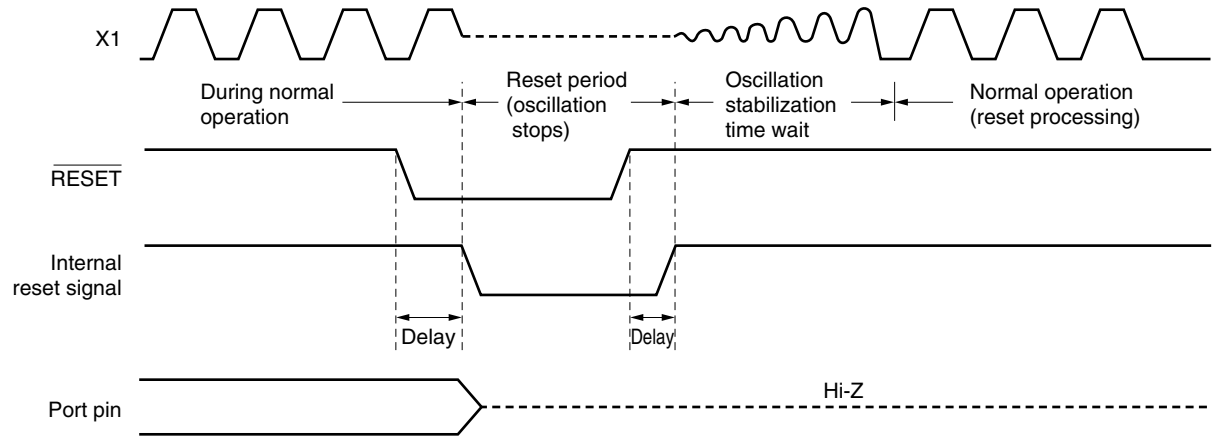


Figure 16-3. Reset Timing by Overflow in Watchdog Timer

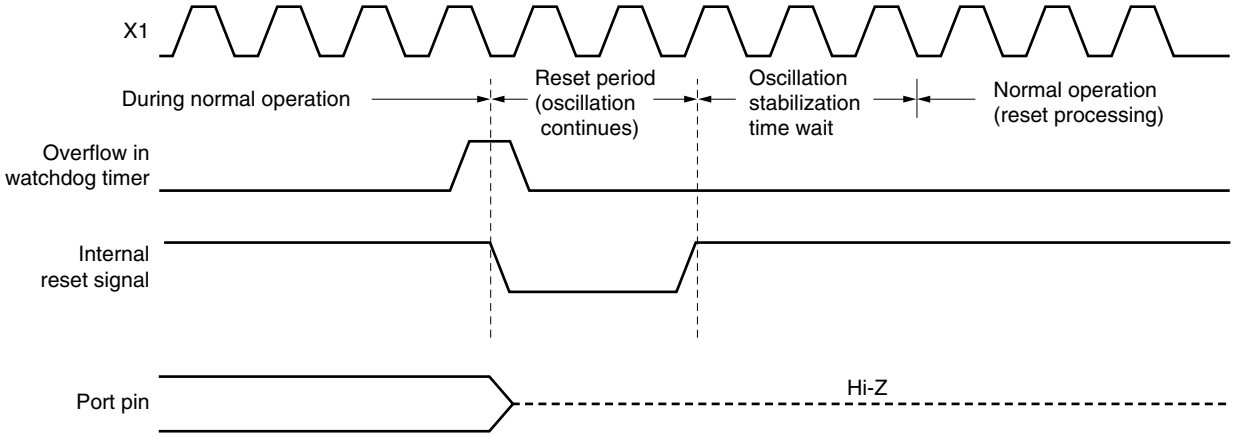


Figure 16-4. Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode

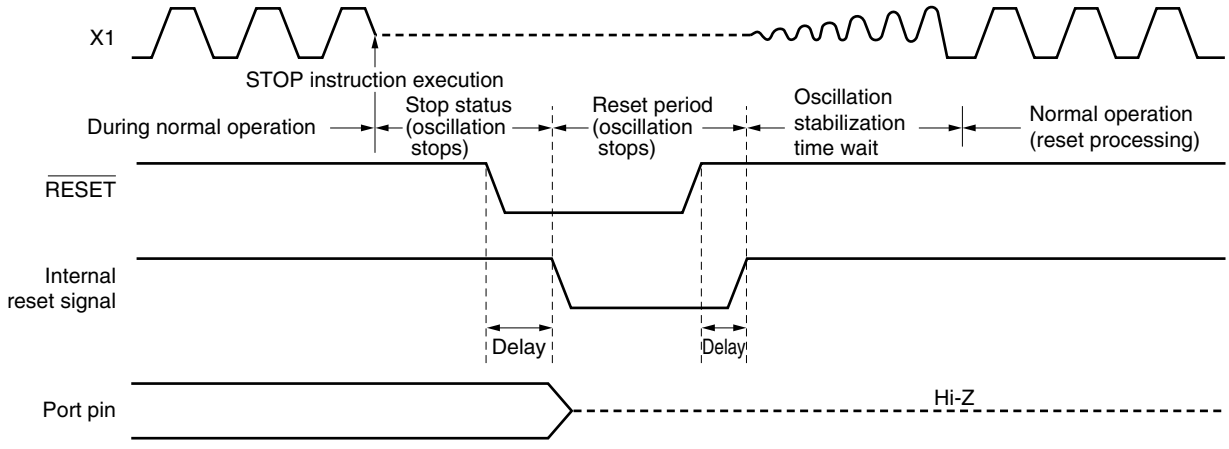


Table 16-1. Statuses of Hardware After Reset (1/2)

Hardware		Status After Reset
Program counter (PC)[Note 1]		Loaded with the contents of the reset vector table (0000H, 0001H)
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined[Note 2]
	General-purpose registers	Undefined[Note 2]
Ports (P0 to P2) (output latches)		00H
Port mode registers (PM0 to PM2)		FFH
Pull-up resistor option registers (PU0, PUB2)		00H
Processor clock control register (PCC)		02H
Oscillation stabilization time selection register (OSTS)		04H
10-bit inverter control timer	Compare registers (CM0 to CM2)	0000H
	Compare register (CM3)	00FFH
	Buffer registers (BFCM0 to BFCM2)	0000H
	Buffer register (BFCM3)	00FFH
	Dead time reload register (DTIME)	FFH
	Control register (TMC7)	00H
	Mode register (TMM7)	00H
8-bit timer/event counters 80, 81, 82	Timer counters (TM80 to TM82)	00H
	Compare registers (CR80 to CR82)	Undefined
	Mode control registers (TMC80 to TMC82)	00H
Watch timer	Mode control register (WTM)	00H
Watchdog timer	Timer clock selection register (TCL2)	00H
	Mode register (WDTM)	00H
A/D converter	Mode register (ADM)	00H
	Conversion result register (ADCRH)	Undefined
	Input selection register (ADS)	00H
Serial interface	Asynchronous serial interface mode register (ASIM00)	00H
	Asynchronous serial interface status register (ASIS00)	00H
	Baud rate generator control register (BRGC00)	00H
	Transmit shift register (TXS00)	Undefined
	Receive buffer register (RXB00)	FFH

Notes 1. While a reset signal is being input, and during the oscillation stabilization period, the contents of the PC will be undefined, while the remainder of the hardware will be the same as after reset.

2. In standby mode, the RAM enters the hold state after a reset.

Table 16-1. Statuses of Hardware After Reset (2/2)

Hardware		Status After Reset
Multiplier	10-bit data registers (MRA1H, MRA1L, MRB1H, MRB1L)	Undefined
	20-bit multiplication result storage registers (MUL1HL, MUL1LH, MUL1LL)	Undefined
	Multiplier control register (MULC1)	00H
Swapping function register (SWP0)		**Note**
Interrupts	Request flag registers (IF0, IF1)	00H
	Mask flag registers (MK0, MK1)	FFH
	External interrupt mode register (INTM0)	00H

Note The status set after a reset differs between read mode and write mode. For details, see **13.2 SWAP Configuration**.

★ # CHAPTER 17 μPD78F9842

The μPD78F9842 features expanded flash memory instead of the internal ROM of the mask ROM versions. The differences between the μPD78F9842 and the mask ROM versions are shown in Table 17-1.

Table 17-1. Differences Between μPD78F9842 and Mask ROM Versions

Item		Flash Memory	Mask ROM	
		μPD78F9842	μPD789841	μPD789842
Internal memory	Flash memory/ROM	16 KB	8 KB	16 KB
	RAM	256 bytes		
IC pin		Not provided	Provided	
V_{PP} pin		Provided	Not provided	
Electrical characteristics		See **CHAPTER 19 ELECTRICAL SPECIFICATIONS**.		

Caution There are differences in the noise immunity and noise radiation between flash memory and mask ROM versions. When pre-producing an application set with the flash memory version and the mass producing it with the mask ROM version, be sure to conduct sufficient evaluations on the commercial sample (CS), not engineering sample (ES), of the mask ROM version.

17.1 Flash Memory Characteristics

Flash memory programming is performed by connecting a dedicated flash programmer (Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4)) to the target system with the μPD78F9842 mounted on the target system (on-board writing). A flash memory program adapter (FA adapter), which is a target board used exclusively for programming, is also provided.

Remark FL-PR3, FL-PR4, and the program adapter are products made by Naito Densei Machida Mfg. Co., Ltd. (TEL +81-45-475-4191).

Programming using flash memory has the following advantages.

- Software can be modified after the microcontroller is solder-mounted on the target system.
- Distinguishing software facilities small-quantity, varied model production
- Easy data adjustment when starting mass production

17.1.1 Programming environment

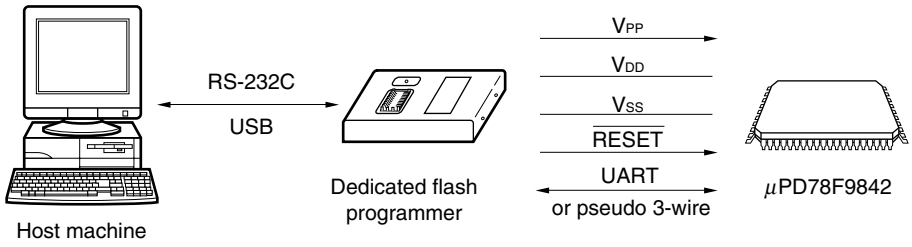
The following shows the environment required for μPD78F9842 flash memory programming.

When Flashpro III (part no. FL-PR3, PG-FP3) or Flashpro IV (part no. FL-PR4, PG-FP4) is used as a dedicated flash programmer, a host machine is required to control the dedicated flash programmer. Communication between the host machine and flash programmer is performed via RS-232C/USB (Rev. 1.1).

For details, refer to the manuals for Flashpro III/Flashpro IV.

Remark USB is supported by Flashpro IV only.

Figure 17-1. Environment for Writing Program to Flash Memory



17.1.2 Communication mode

Use the communication mode shown in Table 17-2 to perform communication between the dedicated flash programmer and μPD78F9842.

Table 17-2. Communication Mode List

Communication Mode	TYPE Setting[Note 1]					Pins Used	Number of V_{PP} Pulses
	COMM PORT	SIO Clock	CPU Clock		Multiple Rate		
			In Flashpro	On Target Board			
UART	UART ch-0 (Async.)	4,800 to 76,800 bps [Notes 2, 4]	5 MHz[Note 5]	4.91 or 5 MHz[Note 2]	1.0	RxD/P22 TxD/P21	8
Pseudo 3-wire	Port A (Pseudo-3 wire)	100 Hz to 1 kHz	1, 2, 4, 5 MHz[Notes 2, 3]	1 to 5 MHz[Note 2]	1.0	P01 P02 P00	12

Notes 1. Selection items for TYPE settings on the dedicated flash programmer (Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4)).

2. The possible setting range differs depending on the voltage. For details, refer to **CHAPTER 19 ELECTRICAL SPECIFICATIONS**.

3. 2 or 4 MHz only for Flashpro III

4. Because signal wave slew also affects UART communication, in addition to the baud rate error, thoroughly evaluate the slew.

5. Only for Flashpro IV. However, when using Flashpro III, be sure to select the clock of the resonator on the board. UART cannot be used with the clock supplied by Flashpro III.

Figure 17-2. Communication Mode Selection Format

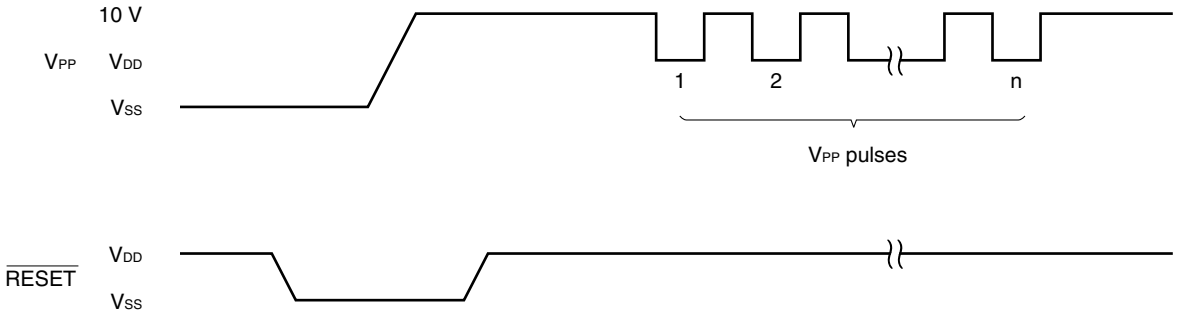
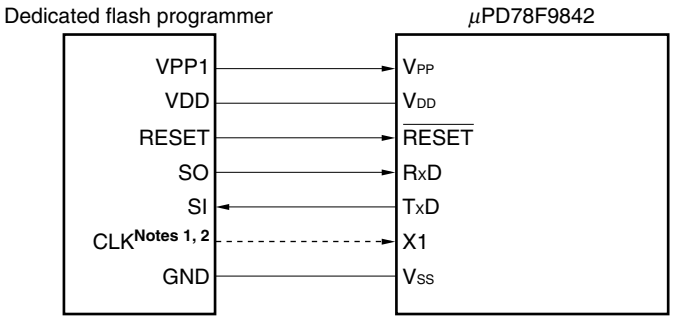
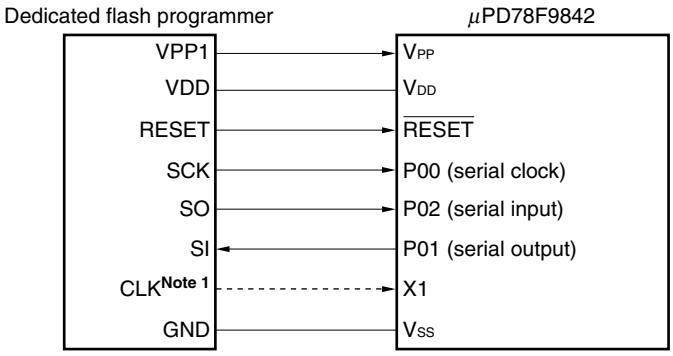


Figure 17-3. Example of Connection with Dedicated Flash Programmer

(a) UART



(b) Pseudo 3-wire (when P0 is used)



Notes 1. When supplying the system clock from a dedicated flash programmer, connect the CLK and X1 pins and cut off the resonator on the board. When using the clock oscillated by the on-board resonator, do not connect the CLK pin.

2. When using UART with Flashpro III, the clock of the resonator connected to the X1 pin must be used, so do not connect the CLK pin.

Caution The V_{DD} pin, if already connected to the power supply, must be connected to the VDD pin of the dedicated flash programmer. When using the power supply connected to the V_{DD} pin, supply voltage before starting programming.

If Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4) is used as a dedicated flash programmer, the following signals are generated for the μPD78F9842. For details, refer to the manual of Flashpro III/Flashpro IV.

Table 17-3. Pin Connection List

Signal Name	I/O	Pin Function	Pin Name	UART	Pseudo 3-Wire
VPP1	Output	Write voltage	V_{PP}	◎	◎
VPP2	–	–	–	×	×
VDD	I/O	V_{DD} voltage generation/voltage monitoring	V_{DD}	◎ Note	◎ Note
GND	–	Ground	V_{SS}	◎	◎
CLK	Output	Clock output	X1	O	O
RESET	Output	Reset signal	$\overline{\text{RESET}}$	◎	◎
SI	Input	Receive signal	TxD, P01	◎	◎
SO	Output	Transmit signal	RxD, P02	◎	◎
SCK	Output	Transfer clock	P00	×	◎
HS	Input	Handshake signal	–	×	×

Note V_{DD} voltage must be supplied before programming is started.

Remark ◎: Pin must be connected.
O: If the signal is supplied on the target board, pin does not need to be connected.
×: Pin does not need to be connected.

17.1.3 On-board pin connections

When programming on the target system, provide a connector on the target system to connect to the dedicated flash programmer.

There may be cases in which an on-board function that switches from the normal operation mode to flash memory programming mode is required.

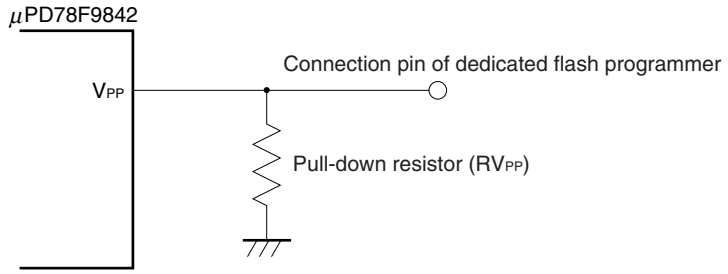
<V_{PP} pin>

Input 0 V to the V_{PP} pin in the normal operation mode. A writing voltage of 10.0 V (TYP.) is supplied to the V_{PP} pin in the flash memory programming mode. Therefore, connect the V_{PP} pin as follows.

(1) Connect a pull-down resistor of RV_{PP} = 10 kΩ to the V_{PP} pin.
(2) Set the jumper on the board to switch the input of V_{PP} pin to the programmer side or directly to GND.

The following shows an example of V_{PP} pin connection.

Figure 17-4. V_{PP} Pin Connection Example



<Serial interface pins>

The following shows the pins used by each serial interface.

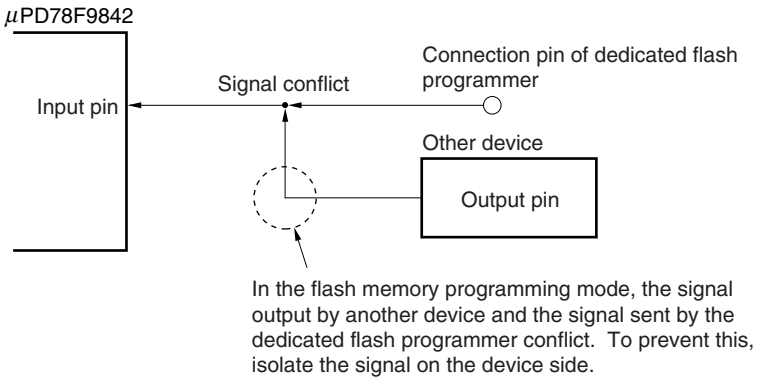
Serial Interface	Pins Used
UART	RxD, TxD
Pseudo 3-wire	P00, P01, P02

Note that signal conflict or malfunction of other devices may occur when an on-board serial interface pin that is connected to another device is connected to the dedicated flash programmer.

(1) Signal conflict

A signal conflict occurs if the dedicated flash programmer (output) is connected to a serial interface pin (input) connected to another device (output). To prevent this signal conflict, isolate the connection with the other device or put the other device in the output high impedance status.

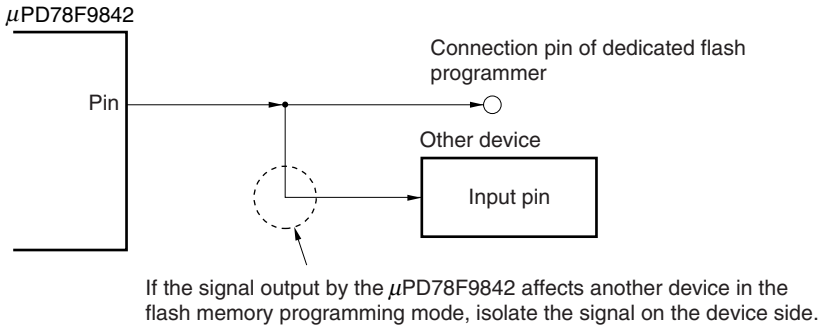
Figure 17-5. Signal Conflict (Serial Interface Input Pin)

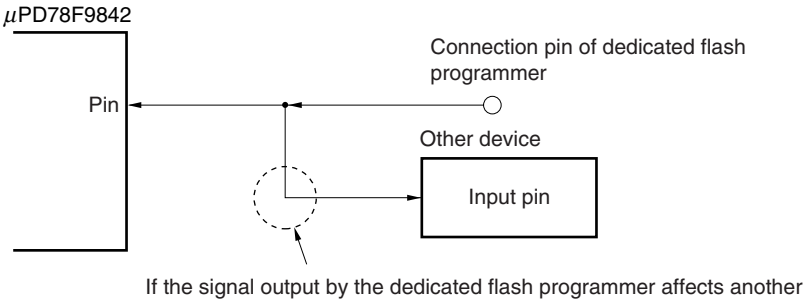


(2) Malfunction of another device

When the dedicated flash programmer (output or input) is connected to a serial interface pin (input or output) connected to another device (input), a signal may be output to the device, causing a malfunction. To prevent such malfunction, isolate the connection with other device or set so that the input signal to the device is ignored.

Figure 17-6. Malfunction of Another Device



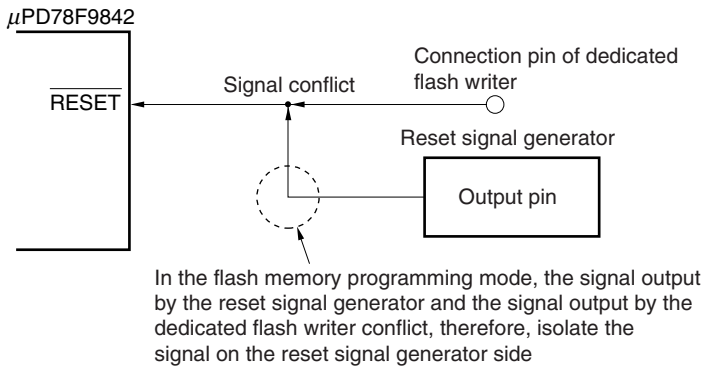


<$\overline{\text{RESET}}$ pin>

When the reset signal of the dedicated flash programmer is connected to the $\overline{\text{RESET}}$ signal connected to the reset signal generator on the board, a signal conflict occurs. To prevent this signal conflict, isolate the connection with the reset signal generator.

If a reset signal is input from the user system in the flash memory programming mode, a normal programming operation will not be performed. Do not input signals other than reset signals from the dedicated flash programmer during this period.

Figure 17-7. Signal Conflict ($\overline{\text{RESET}}$ Pin)



<Port pins>

Shifting to the flash memory programming mode sets all the pins except those used for flash memory programming communication to the status immediately after reset.

Therefore, if the external device does not acknowledge an initial status such as the output high impedance status, connect the external device to V_{DD} or V_{SS} via a resistor.

<Oscillation pins>

When using an on-board clock, connection of X1 and X2 must conform to the methods in the normal operation mode.

When using the clock output of the flash programmer, directly connect it to the X1 pin with the on-board oscillator disconnected, and leave the X2 pin open.

<Power supply>

To use the power output of the flash programmer, connect the V_{DD} and V_{SS} pins to V_{DD} and GND of the flash programmer, respectively.

To use the on-board power supply, connection must conform to that in the normal operation mode. However, because the voltage is monitored by the flash programmer, therefore, V_{DD} of the flash programmer must be connected.

Supply the same power as in normal operation mode for the other power supplies (AV_{DD}, AV_{SS}).

17.1.4 Connection of adapter for flash writing

The following figures show examples of the recommended connection when the adapter for flash writing is used.

Figure 17-8. Wiring Example for Flash Writing Adapter in UART Mode

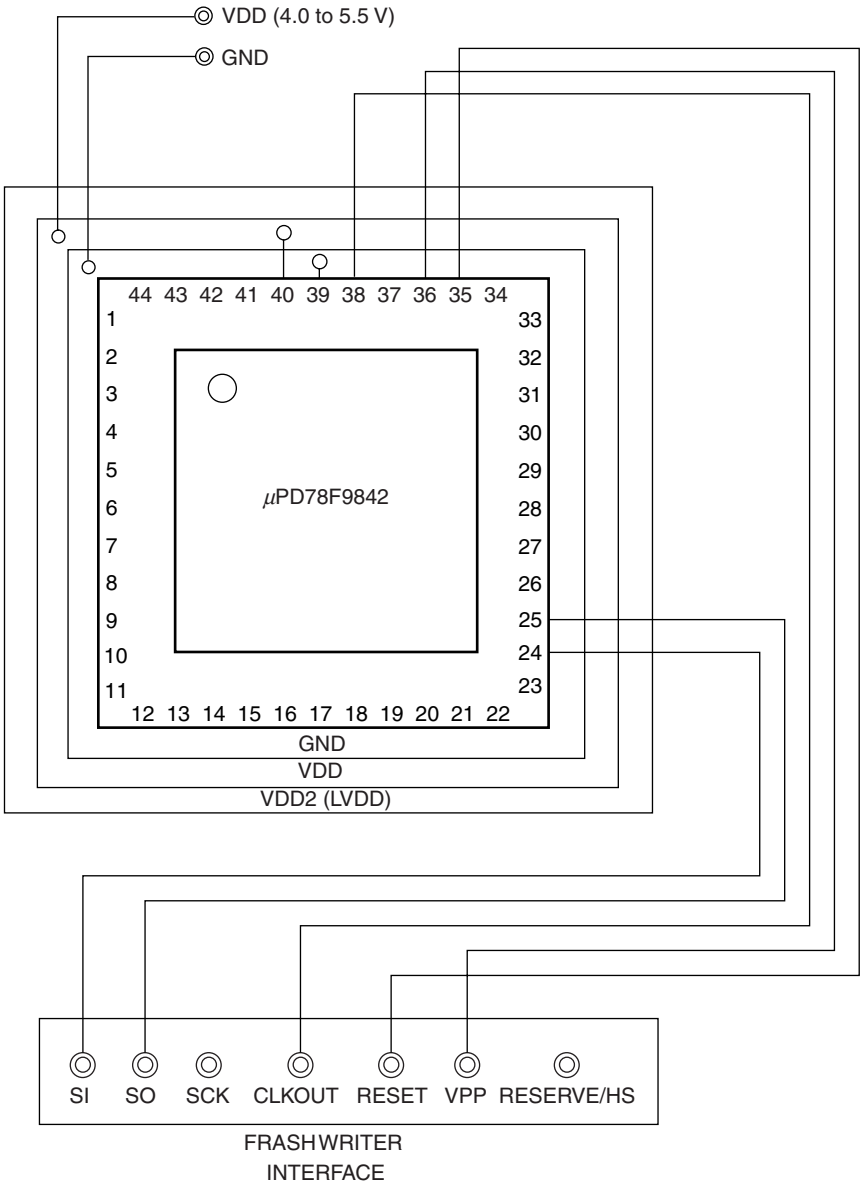
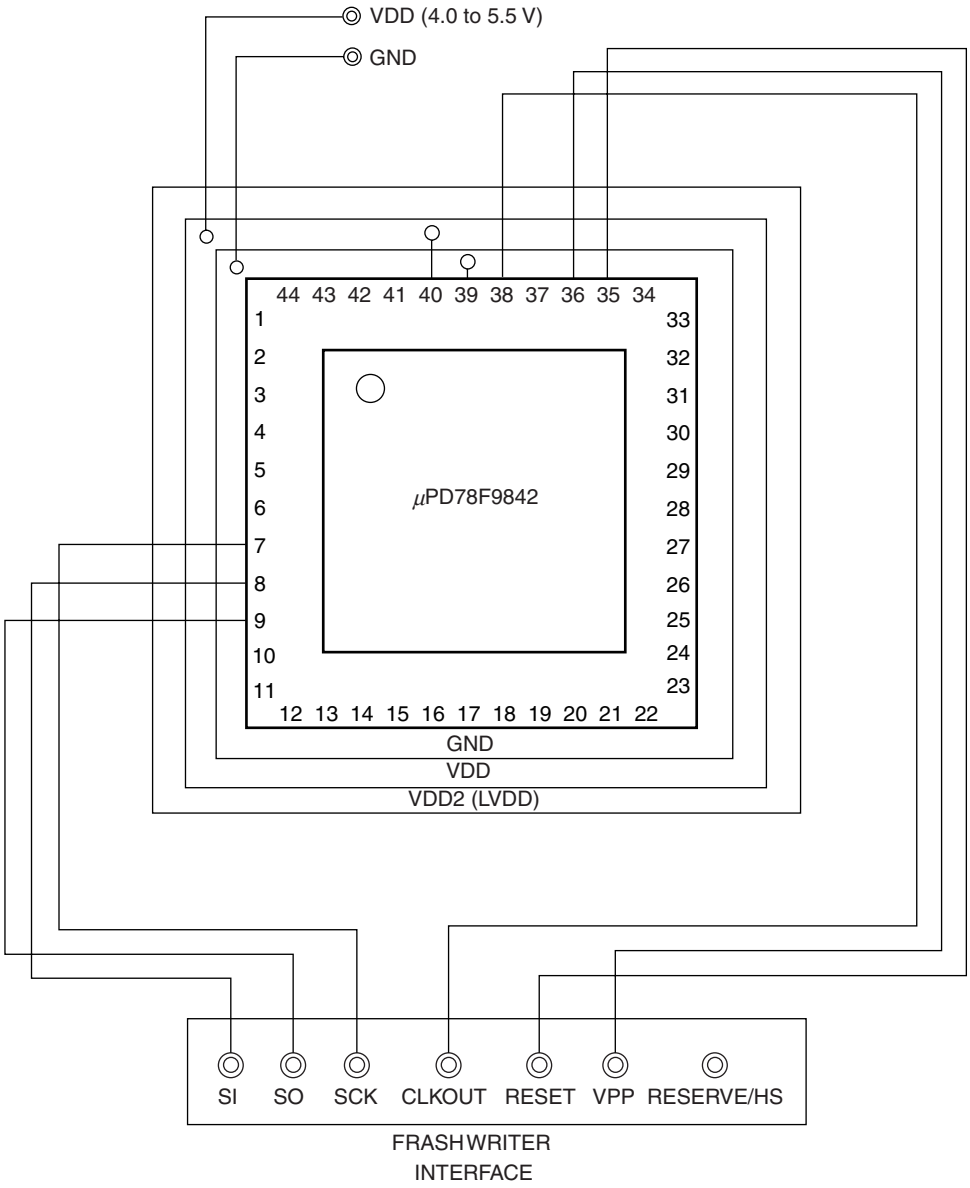


Figure 17-9. Wiring Example for Flash Writing Adapter in Pseudo 3-Wire Mode



CHAPTER 18 INSTRUCTION SET

This chapter lists the instruction set of the μPD789842 Subseries. For the details of the operation and machine language (instruction code) of each instruction, refer to **78K/0S Series Instructions User's Manual (U11047E)**.

18.1 Operation

18.1.1 Operand identifiers and description methods

Operands are described in the "Operand" column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for details). When there are two or more description methods, select one of them. Uppercase letters and the symbols #, !, $, and [ ] are keywords and are described 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, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, $ and [ ] symbols.

For operand register identifiers, 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 for description.

Table 18-1. Operand Identifiers and Description Methods

Identifier	Description Method
r rp sfr	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7) AX (RP0), BC (RP1), DE (RP2), HL (RP3) Special-function register symbol
saddr saddrp	FE20H to FF1FH Immediate data or labels FE20H to FF1FH Immediate data or labels (even addresses only)
addr16 addr5	0000H to FFFFH Immediate data or labels (only even addresses for 16-bit data transfer instructions) 0040H to 007FH Immediate data or labels (even addresses only)
word byte bit	16-bit immediate data or label 8-bit immediate data or label 3-bit immediate data or label

Remark See **Table 3-3** for symbols of special-function registers.

18.1.2 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
IE:	Interrupt request enable flag
NMIS:	Flag indicating non-maskable interrupt servicing in progress
( ):	Memory contents indicated by address or register contents in parentheses
×H, ×L:	Higher 8 bits and lower 8 bits of 16-bit register
∧:	Logical product (AND)
∨:	Logical sum (OR)
∀:	Exclusive logical sum (exclusive OR)
——— :	Inverted data
addr16:	16-bit immediate data or label
jdisp8:	Signed 8-bit data (displacement value)

18.1.3 Description of flag operation column

(Blank):	Unchanged
0:	Cleared to 0
1:	Set to 1
×:	Set/cleared according to the result
R:	Previously saved value is stored

18.2 Operation List

Mnemonic	Operands	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
MOV	r, #byte	3	6	r ← byte			
	saddr, #byte	3	6	(saddr) ← byte			
	sfr, #byte	3	6	sfr ← byte			
	A, r Note 1	2	4	A ← r			
	r, A Note 1	2	4	r ← A			
	A, saddr	2	4	A ← (saddr)			
	saddr, A	2	4	(saddr) ← A			
	A, sfr	2	4	A ← sfr			
	sfr, A	2	4	sfr ← A			
	A, !addr16	3	8	A ← (addr16)			
	!addr16, A	3	8	(addr16) ← A			
	PSW, #byte	3	6	PSW ← byte	×	×	×
	A, PSW	2	4	A ← PSW			
	PSW, A	2	4	PSW ← A	×	×	×
	A, [DE]	1	6	A ← (DE)			
	[DE], A	1	6	(DE) ← A			
	A, [HL]	1	6	A ← (HL)			
	[HL], A	1	6	(HL) ← A			
	A, [HL + byte]	2	6	A ← (HL + byte)			
	[HL + byte], A	2	6	(HL + byte) ← A			
XCH	A, X	1	4	A ↔ X			
	A, r Note 2	2	6	A ↔ r			
	A, saddr	2	6	A ↔ (saddr)			
	A, sfr	2	6	A ↔ sfr			
	A, [DE]	1	8	A ↔ (DE)			
	A, [HL]	1	8	A ↔ (HL)			
	A, [HL + byte]	2	8	A ↔ (HL + byte)			

Notes 1. Except r = A.

2. Except r = A, X.

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by the processor clock control register (PCC).

Mnemonic	Operands	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
MOVW	rp, #word	3	6	rp ← word			
	AX, saddrp	2	6	AX ← (saddrp)			
	saddrp, AX	2	8	(saddrp) ← AX			
	AX, rp Note	1	4	AX ← rp			
	rp, AX Note	1	4	rp ← AX			
XCHW	AX, rp Note	1	8	AX ↔ rp			
ADD	A, #byte	2	4	A, CY ← A + byte	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte	×	×	×
	A, r	2	4	A, CY ← A + r	×	×	×
	A, saddr	2	4	A, CY ← A + (saddr)	×	×	×
	A, !addr16	3	8	A, CY ← A + (addr16)	×	×	×
	A, [HL]	1	6	A, CY ← A + (HL)	×	×	×
	A, [HL + byte]	2	6	A, CY ← A + (HL + byte)	×	×	×
ADDC	A, #byte	2	4	A, CY ← A + byte + CY	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte + CY	×	×	×
	A, r	2	4	A, CY ← A + r + CY	×	×	×
	A, saddr	2	4	A, CY ← A + (saddr) + CY	×	×	×
	A, !addr16	3	8	A, CY ← A + (addr16) + CY	×	×	×
	A, [HL]	1	6	A, CY ← A + (HL) + CY	×	×	×
	A, [HL + byte]	2	6	A, CY ← A + (HL + byte) + CY	×	×	×
SUB	A, #byte	2	4	A, CY ← A – byte	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte	×	×	×
	A, r	2	4	A, CY ← A – r	×	×	×
	A, saddr	2	4	A, CY ← A – (saddr)	×	×	×
	A, !addr16	3	8	A, CY ← A – (addr16)	×	×	×
	A, [HL]	1	6	A, CY ← A – (HL)	×	×	×
	A, [HL + byte]	2	6	A, CY ← A – (HL + byte)	×	×	×

Note Only when rp = BC, DE, or HL.

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by the processor clock control register (PCC).

Mnemonic	Operands	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
SUBC	A, #byte	2	4	A, CY ← A – byte – CY	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte – CY	×	×	×
	A, r	2	4	A, CY ← A – r – CY	×	×	×
	A, saddr	2	4	A, CY ← A – (saddr) – CY	×	×	×
	A, !addr16	3	8	A, CY ← A – (addr16) – CY	×	×	×
	A, [HL]	1	6	A, CY ← A – (HL) – CY	×	×	×
	A, [HL + byte]	2	6	A, CY ← A – (HL + byte) – CY	×	×	×
AND	A, #byte	2	4	A ← A ∧ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ∧ byte	×		
	A, r	2	4	A ← A ∧ r	×		
	A, saddr	2	4	A ← A ∧ (saddr)	×		
	A, !addr16	3	8	A ← A ∧ (addr16)	×		
	A, [HL]	1	6	A ← A ∧ (HL)	×		
	A, [HL + byte]	2	6	A ← A ∧ (HL + byte)	×		
OR	A, #byte	2	4	A ← A ∨ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ∨ byte	×		
	A, r	2	4	A ← A ∨ r	×		
	A, saddr	2	4	A ← A ∨ (saddr)	×		
	A, !addr16	3	8	A ← A ∨ (addr16)	×		
	A, [HL]	1	6	A ← A ∨ (HL)	×		
	A, [HL + byte]	2	6	A ← A ∨ (HL + byte)	×		
XOR	A, #byte	2	4	A ← A ⩛ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ⩛ byte	×		
	A, r	2	4	A ← A ⩛ r	×		
	A, saddr	2	4	A ← A ⩛ (saddr)	×		
	A, !addr16	3	8	A ← A ⩛ (addr16)	×		
	A, [HL]	1	6	A ← A ⩛ (HL)	×		
	A, [HL + byte]	2	6	A ← A ⩛ (HL + byte)	×		

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by the processor clock control register (PCC).

Mnemonic	Operands	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
CMP	A, #byte	2	4	A – byte	×	×	×
	saddr, #byte	3	6	(saddr) – byte	×	×	×
	A, r	2	4	A – r	×	×	×
	A, saddr	2	4	A – (saddr)	×	×	×
	A, !addr16	3	8	A – (addr16)	×	×	×
	A, [HL]	1	6	A – (HL)	×	×	×
	A, [HL + byte]	2	6	A – (HL + byte)	×	×	×
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	×	×	×
INC	r	2	4	r ← r + 1	×	×	
	saddr	2	4	(saddr) ← (saddr) + 1	×	×	
DEC	r	2	4	r ← r – 1	×	×	
	saddr	2	4	(saddr) ← (saddr) – 1	×	×	
INCW	rp	1	4	rp ← rp + 1			
DECW	rp	1	4	rp ← rp – 1			
ROR	A, 1	1	2	$(CY, A_7 \leftarrow A_0, A_{m-1} \leftarrow A_m) \times 1$			×
ROL	A, 1	1	2	$(CY, A_0 \leftarrow A_7, A_{m+1} \leftarrow A_m) \times 1$			×
RORC	A, 1	1	2	$(CY \leftarrow A_0, A_7 \leftarrow CY, A_{m-1} \leftarrow A_m) \times 1$			×
ROLC	A, 1	1	2	$(CY \leftarrow A_7, A_0 \leftarrow CY, A_{m+1} \leftarrow A_m) \times 1$			×
SET1	saddr.bit	3	6	(saddr.bit) ← 1			
	sfr.bit	3	6	sfr.bit ← 1			
	A.bit	2	4	A.bit ← 1			
	PSW.bit	3	6	PSW.bit ← 1	×	×	×
	[HL].bit	2	10	(HL).bit ← 1			
CLR1	saddr.bit	3	6	(saddr.bit) ← 0			
	sfr.bit	3	6	sfr.bit ← 0			
	A.bit	2	4	A.bit ← 0			
	PSW.bit	3	6	PSW.bit ← 0	×	×	×
	[HL].bit	2	10	(HL).bit ← 0			
SET1	CY	1	2	CY ← 1			1
CLR1	CY	1	2	CY ← 0			0
NOT1	CY	1	2	$CY \leftarrow \overline{CY}$			×

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by the processor clock control register (PCC).

Mnemonic	Operands	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
CALL	!addr16	3	6	$(SP - 1) \leftarrow (PC + 3)_H$, $(SP - 2) \leftarrow (PC + 3)_L$, $PC \leftarrow$ addr16, $SP \leftarrow SP - 2$			
CALLT	[addr5]	1	8	$(SP - 1) \leftarrow (PC + 1)_H$, $(SP - 2) \leftarrow (PC + 1)_L$, $PC_H \leftarrow$ (00000000, addr5 + 1), $PC_L \leftarrow$ (00000000, addr5), $SP \leftarrow SP - 2$			
RET		1	6	$PC_H \leftarrow (SP + 1)$, $PC_L \leftarrow (SP)$, $SP \leftarrow SP + 2$			
RETI		1	8	$PC_H \leftarrow (SP + 1)$, $PC_L \leftarrow (SP)$, $PSW \leftarrow (SP + 2)$, $SP \leftarrow SP + 3$, NMIS $\leftarrow 0$	R	R	R
PUSH	PSW	1	2	$(SP - 1) \leftarrow PSW$, $SP \leftarrow SP - 1$			
	rp	1	4	$(SP - 1) \leftarrow rp_H$, $(SP - 2) \leftarrow rp_L$, $SP \leftarrow SP - 2$			
POP	PSW	1	4	$PSW \leftarrow (SP)$, $SP \leftarrow SP + 1$	R	R	R
	rp	1	6	$rp_H \leftarrow (SP + 1)$, $rp_L \leftarrow (SP)$, $SP \leftarrow SP + 2$			
MOVW	SP, AX	2	8	$SP \leftarrow AX$			
	AX, SP	2	6	$AX \leftarrow SP$			
BR	!addr16	3	6	$PC \leftarrow$ addr16			
	$addr16	2	6	$PC \leftarrow PC + 2 +$ jdisp8			
	AX	1	6	$PC_H \leftarrow A$, $PC_L \leftarrow X$			
BC	$saddr16	2	6	$PC \leftarrow PC + 2 +$ jdisp8 if CY = 1			
BNC	$saddr16	2	6	$PC \leftarrow PC + 2 +$ jdisp8 if CY = 0			
BZ	$saddr16	2	6	$PC \leftarrow PC + 2 +$ jdisp8 if Z = 1			
BNZ	$saddr16	2	6	$PC \leftarrow PC + 2 +$ jdisp8 if Z = 0			
BT	saddr.bit, $addr16	4	10	$PC \leftarrow PC + 4 +$ jdisp8 if (saddr.bit) = 1			
	sfr.bit, $addr16	4	10	$PC \leftarrow PC + 4 +$ jdisp8 if sfr.bit = 1			
	A.bit, $addr16	3	8	$PC \leftarrow PC + 3 +$ jdisp8 if A.bit = 1			
	PSW.bit, $addr16	4	10	$PC \leftarrow PC + 4 +$ jdisp8 if PSW.bit = 1			
BF	saddr.bit, $addr16	4	10	$PC \leftarrow PC + 4 +$ jdisp8 if (saddr.bit) = 0			
	sfr.bit, $addr16	4	10	$PC \leftarrow PC + 4 +$ jdisp8 if sfr.bit = 0			
	A.bit, $addr16	3	8	$PC \leftarrow PC + 3 +$ jdisp8 if A.bit = 0			
	PSW.bit, $addr16	4	10	$PC \leftarrow PC + 4 +$ jdisp8 if PSW.bit = 0			
DBNZ	B, $addr16	2	6	$B \leftarrow B - 1$, then $PC \leftarrow PC + 2 +$ jdisp8 if $B \neq 0$			
	C, $addr16	2	6	$C \leftarrow C - 1$, then $PC \leftarrow PC + 2 +$ jdisp8 if $C \neq 0$			
	saddr, $addr16	3	8	$(saddr) \leftarrow (saddr) - 1$, then $PC \leftarrow PC + 3 +$ jdisp8 if $(saddr) \neq 0$			
NOP		1	2	No Operation			
EI		3	6	IE $\leftarrow$ 1 (Enable Interrupt)			
DI		3	6	IE $\leftarrow$ 0 (Disable Interrupt)			
HALT		1	2	Set HALT Mode			
STOP		1	2	Set STOP Mode			

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by the processor clock control register (PCC).

18.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, INC, DEC, ROR, ROL, RORC, ROLC, PUSH, POP, DBNZ

1st Operand \ 2nd Operand	#byte	A	r	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL + byte]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV[Note] XCH[Note] ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV 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											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											
[HL + byte]		MOV											

Note Except r = A.

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

1st Operand \ 2nd Operand	#word	AX	rp[Note]	saddrp	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	
rp	MOVW	MOVW[Note]				INCW DECW PUSH POP
saddrp		MOVW				
SP		MOVW				

Note Only when rp = BC, DE, or HL.

(3) Bit manipulation instructions

SET1, CLR1, NOT1, BT, BF

1st Operand \ 2nd Operand	$addr16	None
A.bit	BT BF	SET1 CLR1
sfr.bit	BT BF	SET1 CLR1
saddr.bit	BT BF	SET1 CLR1
PSW.bit	BT BF	SET1 CLR1
[HL].bit		SET1 CLR1
CY		SET1 CLR1 NOT1

(4) Call instructions/branch instructions

CALL, CALLT, BR, BC, BNC, BZ, BNZ, DBNZ

1st Operand \ 2nd Operand	AX	!addr16	[addr5]	$addr16
Basic instructions	BR	CALL BR	CALLT	BR BC BNC BZ BNZ
Compound instructions				DBNZ

(5) Other instructions

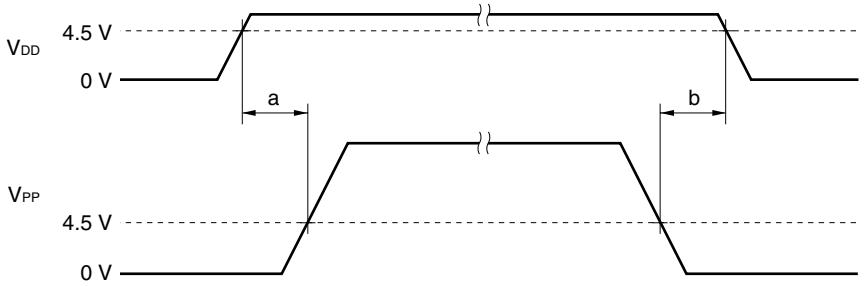
RET, RETI, NOP, EI, DI, HALT, STOP

Absolute Maximum Ratings (T_A = 25°C)

Parameter	Symbol	Conditions	Rated Value	Unit
Power supply voltage	V_{DD}		–0.3 to +6.5	V
	V_{PP}	μPD78F9842 only, **Note**	–0.3 to +10.5	V
Input voltage	V_I		–0.3 to V_{DD} + 0.3	V
Output voltage	V_O		–0.3 to V_{DD} + 0.3	V
Output current, high	I_{OH}	Per pin	–10	mA
		Total for all pins	–30	mA
Output current, low	I_{OL}	Per pin	30	mA
		Total for all pins	160	mA
Operating ambient temperature	T_A	In normal operation	–40 to +85	°C
		During flash memory programming	10 to 40	°C
Storage temperature	T_{stg}		–65 to +150	°C

Note Make sure that the following conditions of the V_{PP} voltage application timing are satisfied when the flash memory is written.

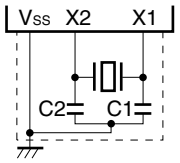
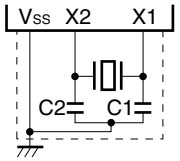
- **When supply voltage rises**
  V_{PP} must exceed V_{DD} 10 μs or more after V_{DD} has reached the lower-limit value (4.5 V) of the operating voltage range (see a in the figure below).
- **When supply voltage drops**
  V_{DD} must be lowered 10 μs or more after V_{PP} falls below the lower-limit value (4.5 V) of the operating voltage range of V_{DD} (see b in the figure below).



Caution Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

Remark Unless otherwise specified, the characteristics of alternate-function pins are the same as those of port pins.

System Clock Oscillator Characteristics (T_A = –40 to +85°C, V_{DD} = 4.0 to 5.5 V)

Resonator	Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Ceramic resonator		Oscillator frequency (fx)[Note 1]	V_{DD} = oscillation voltage range	8.0	8.38	8.5	MHz
		Oscillation stabilization time[Note 2]	Time after the V_{DD} has reached the minimum value in the oscillation voltage range			4	ms
Crystal oscillator		Oscillator frequency (fx) [Note 1]		8.0	8.38	8.5	MHz
		Oscillation stabilization time[Note 2]				10	ms

Notes 1. Only the characteristics of the oscillator are indicated. See the description of the AC characteristics for the instruction execution time.

2. Time required for oscillation to stabilize once a reset sequence ends or STOP mode is released. Use a resonator that will become stable within the oscillation wait time.

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

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with any other signal lines.**
- **Do not route the wiring in the vicinity of a line through which a high fluctuating current flows.**
- **Always make the ground of the capacitor the same potential as V_{SS}.**
- **Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

Remark For the resonator selection and oscillator constant, customers are required to either evaluate the oscillation themselves or apply to the resonator manufacturer for evaluation.

DC Characteristics (T_A = –40 to +85°C, V_{DD} = 4.0 to 5.5 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Output current, high	I_{OH}	Per pin				–1	mA
		Total for all pins				–15	mA
Output current, low	I_{OL}	Per pin				10	mA
		Total for all pins				80	mA
Input voltage, high	V_{IH1}	P00 to P07, P10 to P17, P60 to P67		0.7V_{DD}		V_{DD}	V
	V_{IH2}	$\overline{\text{RESET}}$, P20 to P25		0.8V_{DD}		V_{DD}	V
	V_{IH3}	X1, X2		V_{DD} – 0.1		V_{DD}	V
Input voltage, low	V_{IL1}	P00 to P07, P10 to P17, P60 to P67		0		0.3V_{DD}	V
	V_{IL2}	$\overline{\text{RESET}}$, P20 to P25		0		0.2V_{DD}	V
	V_{IL3}	X1, X2		0		0.1	V
Output voltage, high	V_{OH}	I_{OH} = –1mA		V_{DD} – 1.0			V
Output voltage, low	V_{OL}	I_{OL} = 10mA				1.0	V
Input leakage current, high	I_{LIH1}	V_{IN} = V_{DD}	Pins other than X1 and X2			3	μA
	I_{LIH2}		X1, X2			20	μA
Input leakage current, low	I_{ILI1}	V_{IN} = 0 V	Pins other than X1 and X2			–3	μA
	I_{ILI2}		X1, X2			–20	μA
Output leakage current, high	I_{LOH}	V_{OUT} = V_{DD}				3	μA
Output leakage current, low	I_{LOL}	V_{OUT} = 0 V				–3	μA
Software pull-up resistance	R	V_{IN} = 0 V		50	100	200	kΩ
Power supply current[Note 1]	I_{DD1}	8.38 MHz crystal oscillation operating mode[Note 2]			5.5	16.5	mA
	I_{DD2}	8.38 MHz crystal oscillation HALT mode			1.2	3.6	mA
	I_{DD3}	STOP mode			0.1	30	μA
	I_{DD4}	8.38 MHz crystal oscillation A/D operating mode			6.0	18.0	mA

Notes 1. The power supply current does not include the current flowing through the on-chip pull-up resistors.

2. During high-speed mode operation (when the processor clock control register (PCC) is cleared to 00H.)

Remark Unless otherwise specified, the characteristics of alternate function pins are the same as those of port pins.

AC Characteristics

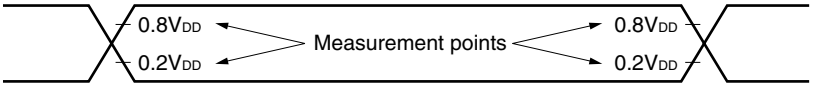
(1) Basic operations (T_A = –40 to +85°C, V_{DD} = 4.0 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Cycle time (minimum instruction execution time)	T_{CY}	When PCC is set to 00H	0.24		0.25	μs
		When PCC is set to 02H	0.94		1.00	μs
TI80, TI81 input frequency	f_{TI}		0		4.0	MHz
TI80, TI81 input high/low-level width	f_{TIH}, f_{TIL}		0.1			μs
Interrupt input high/low-level width	f_{INTH}, f_{INTL}	INTP0, INTP1	10			μs
$\overline{\text{RESET}}$ input low-level width	f_{RSL}		10			μs

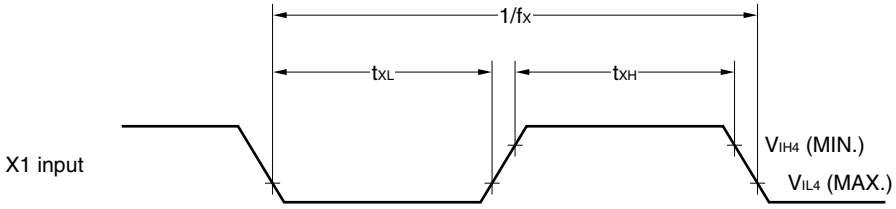
(2) Serial interface (UART) (T_A = –40 to +85°C, V_{DD} = 4.0 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate		At fx = 8.38 MHz operation			115200	bps

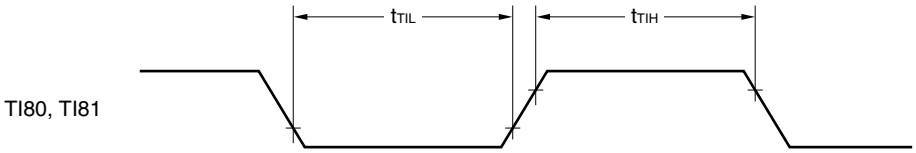
AC Timing Measurement Points (Except X1 Input)



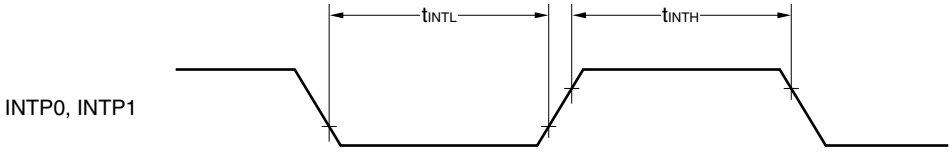
Clock Timing



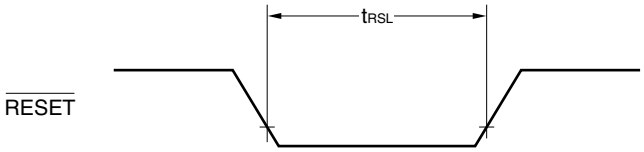
TI Timing



Interrupt Input Timing



$\overline{\text{RESET}}$ Input Timing



A/D Converter Characteristics (T_A = –40 to +85°C, V_{DD} = 4.0 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution			8	8	8	bit
Overall error[Note]					1.5	LSB
Conversion time	t_{CONV}		14			μs
Analog input voltage	V_{IAN}		0		V_{DD}	V

Note Excludes quantization error (±1/2 LSB).

Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics (T_A = –40 to +85°C)

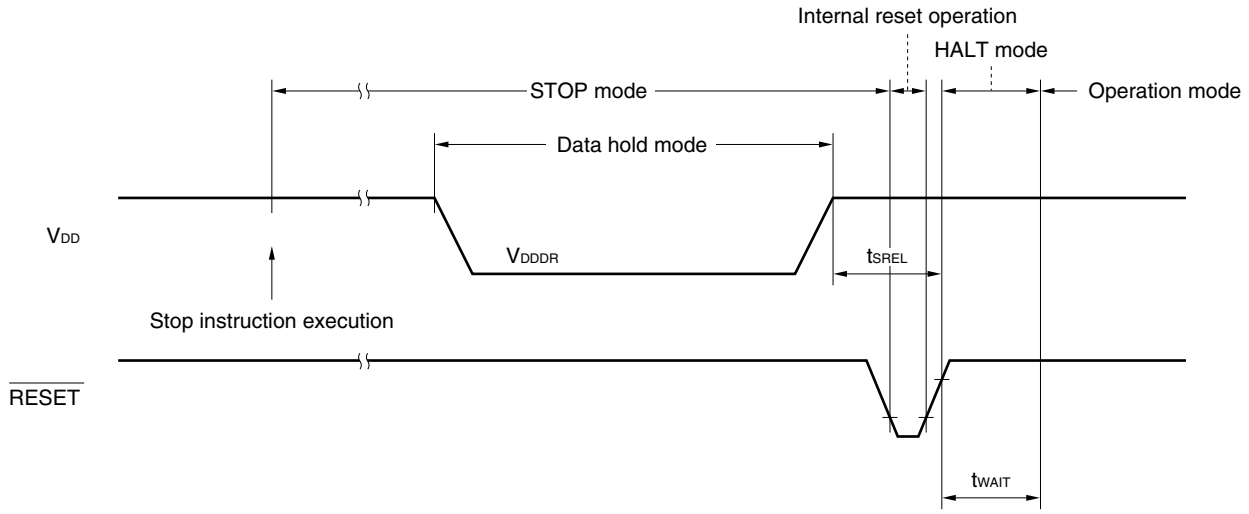
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data retention supply voltage	V_{DDDR}		4.0		5.5	V
Release signal set time	t_{SREL}		0			μs
Oscillation stabilization wait time[Note 1]	t_{WAIT}	Cleared by $\overline{\text{RESET}}$		2^{15}/fx		ms
		Cleared by an interrupt request		**Note 2**		ms

Notes 1. The oscillation stabilization time is a period in which the operation of the CPU is stopped in order to avoid unstable operation at the start of oscillation.

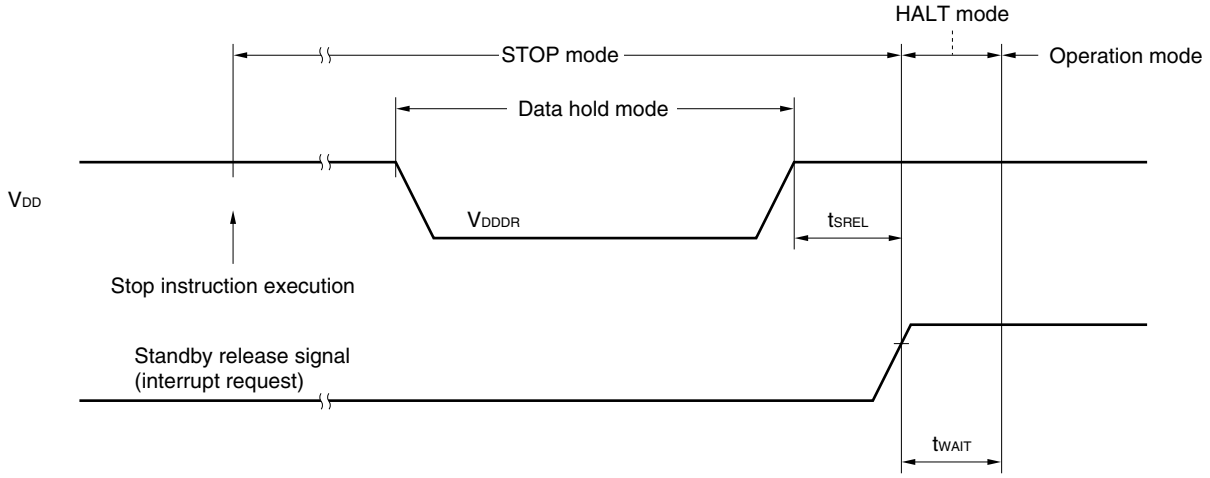
2. The typical (TYP) value can be selected from 2^{12}/fx, 2^{15}/fx, or 2^{17}/fx by bits 0 to 2 (OSTS0 to OSTS2) of the oscillation stabilization time selection register.

Remark fx: System clock oscillation frequency

Data Retention Timing (STOP Mode Release by $\overline{\text{RESET}}$)



Data Retention Timing (Standby Release Signal: STOP Mode Release by Interrupt Signal)

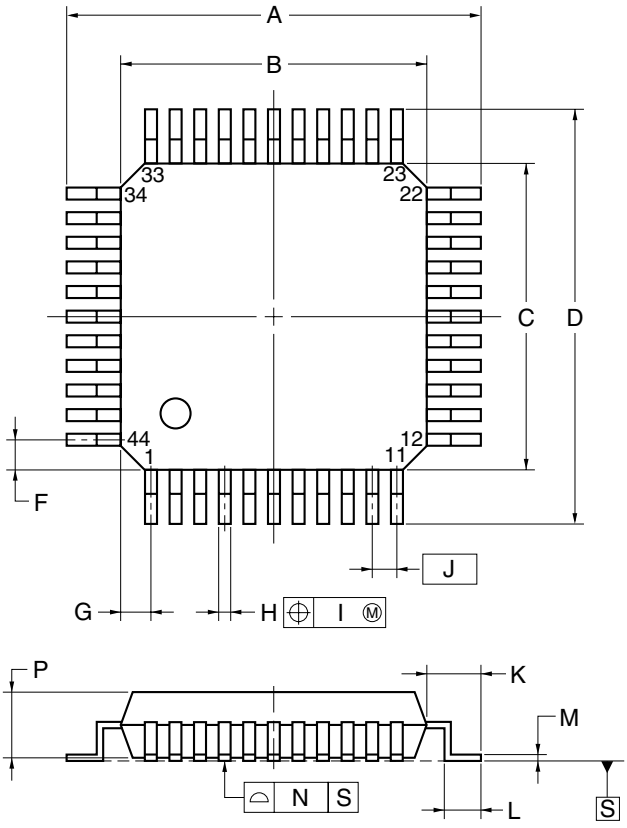


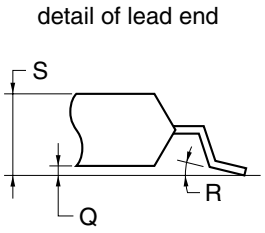
Flash Memory Programming Characteristics

Basic characteristics (T_A = 10 to 40°C, V_{DD} = 4.0 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Clock frequency	f_X		8.0		8.5	MHz
Power supply voltage	V_{PPL}	During V_{PP} low-level detection	0		$0.2V_{DD}$	V
	V_{PPH}	During V_{PP} high-level detection	$0.8V_{DD}$	V_{DD}	$1.2V_{DD}$	V
	V_{PP}	During V_{PP} high-voltage detection	9.7	10.0	10.3	V
V_{DD} power supply current	I_{DD}				50	mA
V_{PP} power supply current	I_{PP}	V_{PP} = 10 V			100	mA
Write time	T_{WRT}	1 byte		50	500	μs
Number of rewrites	C_{WRT}				20	Times
Erase time	T_{ERASE}			6		s

44-PIN PLASTIC QFP (10x10)





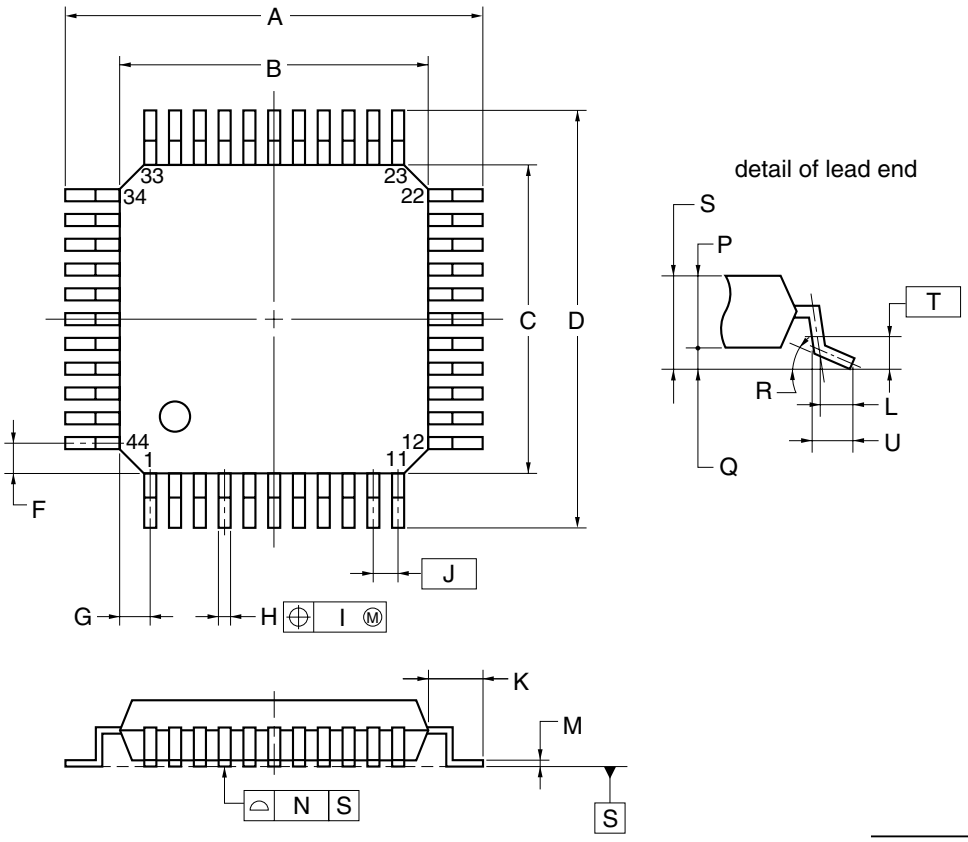
NOTE

Each lead centerline is located within 0.16 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	13.2±0.2
B	10.0±0.2
C	10.0±0.2
D	13.2±0.2
F	1.0
G	1.0
H	$0.37^{+0.08}_{-0.07}$
I	0.16
J	0.8 (T.P.)
K	1.6±0.2
L	0.8±0.2
M	$0.17^{+0.06}_{-0.05}$
N	0.10
P	2.7±0.1
Q	0.125±0.075
R	$3^{\circ}{}^{+7^{\circ}}_{-3^{\circ}}$
S	3.0 MAX.

S44GB-80-3BS-2

44 PIN PLASTIC LQFP (10x10)



NOTE

Each lead centerline is located within 0.20 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	12.0±0.2
B	10.0±0.2
C	10.0±0.2
D	12.0±0.2
F	1.0
G	1.0
H	$0.37^{+0.08}_{-0.07}$
I	0.20
J	0.8 (T.P.)
K	1.0±0.2
L	0.5
M	$0.17^{+0.03}_{-0.06}$
N	0.10
P	1.4±0.05
Q	0.1±0.05
R	$3^{\circ}{}^{+4^{\circ}}_{-3^{\circ}}$
S	1.6 MAX.
T	0.25 (T.P.)
U	0.6±0.15

S44GB-80-8ES-2

★ CHAPTER 21 RECOMMENDED SOLDERING CONDITIONS

The μPD789842 Subseries[Note] should be soldered and mounted under the following recommended conditions.

For soldering methods and conditions other than those recommended below, contact an NEC Electronics sales representative.

For technical information, see the following website.

Semiconductor Device Mount Manual (http://www.necel.com/pkg/en/mount/index.html)

Note Evaluation of the soldering conditions for the GB-3BS-MTX type is not yet complete.

Table 21-1. Surface Mounting Type Soldering Conditions

(1) μPD789841GB-×××-8ES
μPD789842GB-×××-8ES
μPD78F9842GB-8ES

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C, Time: 30 seconds max. (at 210°C or higher), Count: Two times or less	IR35-00-2
VPS	Package peak temperature: 215°C, Time: 40 seconds max. (at 200°C or higher), Count: Two times or less	VP15-00-2
Wave soldering	Solder bath temperature: 260°C max., Time: 10 seconds max., Count: Once, Preheating temperature: 120°C max. (package surface temperature)	WS60-00-1
Partial heating	Pin temperature: 350°C max., Time: 3 seconds max. (per pin row)	–

(2) μPD789841GB-×××-8ES-A
μPD789842GB-×××-8ES-A
μPD78F9842GB-8ES-A

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 260°C, Time: 60 seconds max. (at 220°C or higher), Count: Three times or less, Exposure limit: 7 days[Note] (after that, prebake at 125°C for 20 to 72 hours)	IR60-207-3
Wave soldering	For details, contact an NEC Electronics sales representative.	–
Partial heating	Pin temperature: 350°C max., Time: 3 seconds max. (per pin row)	–

Note After opening the dry pack, store it at 25°C or less and 65% RH or less for the allowable storage period.

Caution Do not use different soldering methods together (except for partial heating).

Remark Products that have the part numbers suffixed by "-A" are lead-free products.

★

APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for development of systems using the μPD789842 Subseries. Figure A-1 shows development tools.

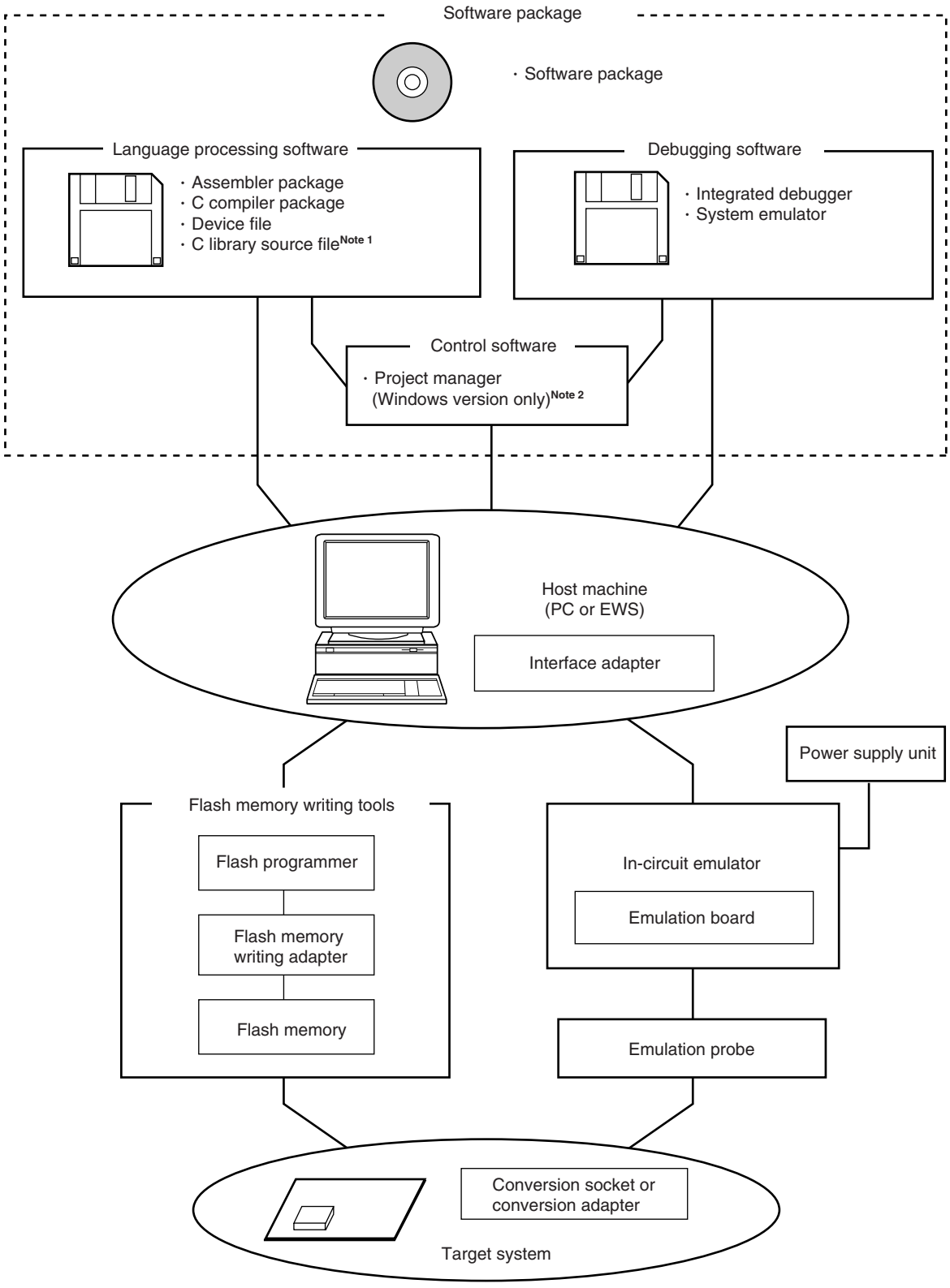
- Support of PC98-NX series

  Unless specified otherwise, the products supported by IBM PC/AT™ compatibles can be used in the PC98-NX series. When using the PC98-NX series, refer to the explanation of IBM PC/AT compatibles.

- Windows

  Unless specified otherwise, “Windows” indicates the following operating systems.
  - Windows 3.1
  - Windows 95, 98, 2000
  - Windows NT™ Ver. 4.0

Figure A-1. Development Tools



Notes 1. The C library source file is not included in the software package.

2. The project manager is included in the assembler package and is available only for Windows.

A.1 Software Package

SP78K0S Software package	Various software tools for 78K/0S development are integrated in one package. The following tools are included. RA78K0S, CC78K0S, ID78K0S-NS, SM78K0S, various device files
	Part number: μS××××SP78K0S

Remark ×××× in the part number differs depending on the operating system used.

μS×××× SP78K0S

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT and compatibles	Japanese Windows	CD-ROM
BB17		English Windows	

A.2 Language Processing Software

RA78K0S Assembler package	Program that converts program written in mnemonic into object codes that can be executed by a microcontroller. In addition, automatic functions to generate a symbol table and optimize branch instructions are also provided. Used in combination with a device file (DF789842) (sold separately). **<Caution when used in PC environment>** The assembler package is a DOS-based application but may be used in the Windows environment by using the project manager of Windows (included in the package).
	Part number: μS××××RA78K0S
CC78K0S C compiler package	Program that converts program written in C language into object codes that can be executed by a microcontroller. Used in combination with an assembler package (RA78K0S) and device file (DF789842) (both sold separately). **<Caution when used in PC environment>** The C compiler package is a DOS-based application but may be used in the Windows environment by using the project manager of Windows (included in the assembler package).
	Part number: μS××××CC78K0S
DF789842[Note 1] Device file	File containing the information inherent to the device. Used in combination with other tools (RA78K0S, CC78K0S, ID78K0S-NS, SM78K0S) (all sold separately).
	Part number: μS××××DF789842
CC78K0S-L[Note 2] C library source file	Source file of functions constituting the object library included in the C compiler package. Necessary for changing the object library included in the C compiler package according to the customer's specifications. Since this is a source file, its working environment does not depend on any particular operating system.
	Part number: μS××××CC78K0S-L

Notes 1. DF789842 is a common file that can be used with RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S.

2. CC78K0S-L is not included in the software package (SP78K0S).

Remark ××××in the part number differs depending on the host machine and operating system used.

μS××××RA78K0S
μS××××CC78K0S

××××	Host Machine	OS	Supply Media
AB13	PC-9800 series, IBM PC/AT and compatibles	Japanese Windows	3.5” 2HD FD
BB13		English Windows	
AB17		Japanese Windows	CD-ROM
BB17		English Windows	
3P17	HP9000 series 700™	HP-UX™ (Rel.10.10)	
3K17	SPARCstation™	SunOS™ (Rel.4.1.4), Solaris™ (Rel.2.5.1)	

μS××××DF789842
μS××××CC78K0S-L

××××	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT and compatibles	Japanese Windows	3.5” 2HD FD
BB13		English Windows	
3P16	HP9000 series 700	HP-UX (Rel.10.10)	DAT
3K13	SPARCstation	SunOS (Rel.4.1.4), Solaris (Rel.2.5.1)	3.5” 2HD FD
3K15			1/4” CGMT

A.3 Control Software

Project manager	Control software provided for efficient user program development in the Windows environment. The project manager allows a series of tasks required for user program development to be performed, including starting the editor, building, and starting the debugger. <**Caution**> The project manager is included in the assembler package (RA78K0S). It cannot be used in an environment other than Windows.

A.4 Flash Memory Writing Tools

Flashpro III (FL-PR3, PG-FP3) Flashpro IV (FL-PR4, PG-FP4) Flash programmer	Flash programmer dedicated to microcontrollers incorporating flash memory.
FA-44GB FA-44GB-8ES Flash memory writing adapter	Flash memory writing adapter. Used in connection with Flashpro III or Flashpro IV. FA-44GB: For 44-pin plastic QFP (GB-3BS type) FA-44GB-8ES: For 44-pin plastic LQFP (GB-8ES type)

Remark FL-PR3, FL-PR4, FA-44GB, and FA-44GB-8ES are products of Naito Densei Machida Mfg. Co., Ltd.
For further information, contact: Naito Densei Machida Mfg. Co., Ltd. (+81-45-475-4191)

A.5 Debugging Tools (Hardware)

IE-78K0S-NS In-circuit emulator	In-circuit emulator for debugging hardware and software of application system using the 78K/0S Series. Can be used with an integrated debugger (ID78K0S-NS). Used in combination with an AC adapter, emulation probe, and interface adapter for connecting the host machine.
IE-78K0S-NS-A In-circuit emulator	In-circuit emulator with enhanced functions of the IE-78K0S-NS. The debug function is further enhanced by adding a coverage function and enhancing the tracer and timer functions.
IE-70000-MC-PS-B AC adapter	Adapter for supplying power from a 100 to 240 VAC outlet.
IE-70000-98-IF-C Interface adapter	Adapter required when using a PC-9800 series (except notebook type) as the host machine (C bus supported).
IE-70000-CD-IF-A PC card interface	PC card and interface cable required when using a notebook type PC as the host machine (PCMICA socket supported).
IE-70000-PC-IF-C Interface adapter	Adapter required when using an IBM PC/AT or compatible as the host machine (ISA bus supported).
IE-70000-PCI-IF-A Interface adapter	Adapter required when using a personal computer incorporating the PCI bus as the host machine.
IE-789842-NS-EM1 Emulation board	Emulation board for emulating the peripheral hardware inherent to the device. Used in combination with an in-circuit emulator.
NP-44GB-TQ NP-H44GB-TQ Emulation probe	Probe for connecting the in-circuit emulator and target system. Used in combination with TGB-044SAP.
TGB-044SAP Conversion adapter	Conversion adapter used to connect a target system board designed to allow mounting a 44-pin plastic QFP (GB-3BS type) or 44-pin plastic LQFP (GB-8ES type) and the NP-44GB-TQ/NP-H44GB-TQ.

Remarks 1. NP-44GB-TQ and NP-H44GB-TQ are products of Naito Densei Machida Mfg. Co., Ltd.
For further information, contact: Naito Densei Machida Mfg. Co., Ltd. (+81-45-475-4191)

2. TGB-044SAP is a product made by TOKYO ELETECH CORPORATION.
For further information, contact: Daimaru Kogyo, Ltd.
Tokyo Electronics Department (TEL +81-3-3820-7112)
Osaka Electronics Department (TEL +81-6-6244-6672)

A.6 Debugging Tools (Software)

ID78K0S-NS Integrated debugger	This debugger supports the in-circuit emulators IE-78K0S-NS and IE-78K0S-NS-A for the 78K/0S Series. The ID78K0S-NS is Windows-based software. It has improved C-compatible debugging functions and can display the results of tracing with the source program using an integrating window function that associates the source program, disassemble display, and memory display with the trace result. Used in combination with a device file (DF789842) (sold separately).
	Part number: μS××××ID78K0S-NS
SM78K0S System simulator	This is a system simulator for the 78K/0S Series. The SM78K0S is Windows-based software. It can be used to debug the target system at C source level of assembler level while simulating the operation of the target system on the host machine. Using SM78K0S, the logic and performance of the application can be verified independently of hardware development. Therefore, the development efficiency can be enhanced and the software quality can be improved. Used in combination with a device file (DF789842) (sold separately).
	Part number: μS××××SM78K0S
DF789842[Note] Device file	File containing the information inherent to the device. Used in combination with other tools (RA78K0S, CC78K0S, ID78K0S-NS, SM78K0S) (all sold separately).
	Part number: μS××××DF789842

Note DF789842 is a common file that can be used with RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S.

Remark ×××× in the part number differs depending on the operating system used and the supply medium.

μS××××ID78K0S-NS
μS××××SM78K0S

××××	Host Machine	OS	Supply Medium
AB13	PC-9800 series, IBM PC/AT and compatibles	Japanese Windows	3.5” 2HD FD
BB13		English Windows	
AB17		Japanese Windows	CD-ROM
BB17		English Windows	

★ APPENDIX B NOTES ON TARGET SYSTEM DESIGN

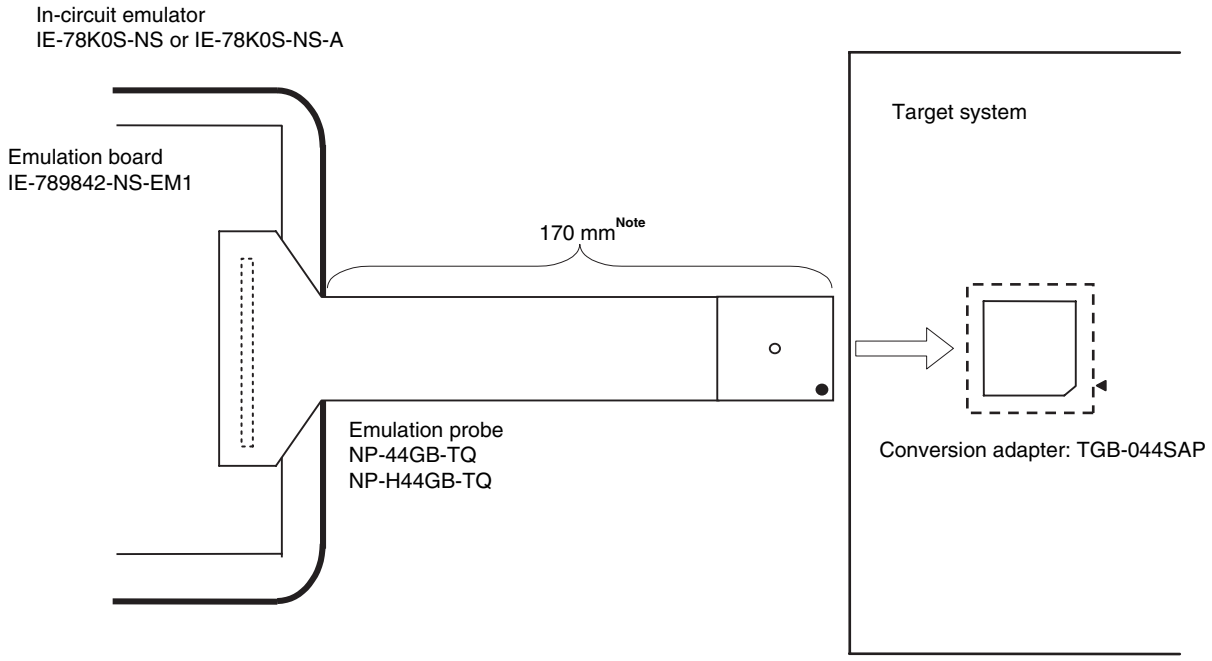
The following shows the conditions when connecting the emulation probe and conversion adapter. Consider the shape of the components to be mounted on the target system and follow the configurations below when designing the system.

Among the products described in this appendix, NP-44GB-TQ and NP-H44GB-TQ are products of Naito Densei Machida Mfg. Co., Ltd. and TGB-044SAP is a product of TOKYO ELETECH CORPORATION.

Table B-1. Distance Between IE System and Conversion Adapter

Emulation Probe	Conversion Adapter	Distance Between IE System and Conversion Adapter
NP-44GB-TQ	TGB-044SAP	170 mm
NP-H44GB-TQ		370 mm

Figure B-1. Distance Between IE System and Conversion Adapter



Note The above distance shows when the NP-44GB-TQ is used. When the NP-H44GB-TQ is used, the distance is 370 mm.

Figure B-2. Connection Conditions of Target System (NP-44GB-TQ)

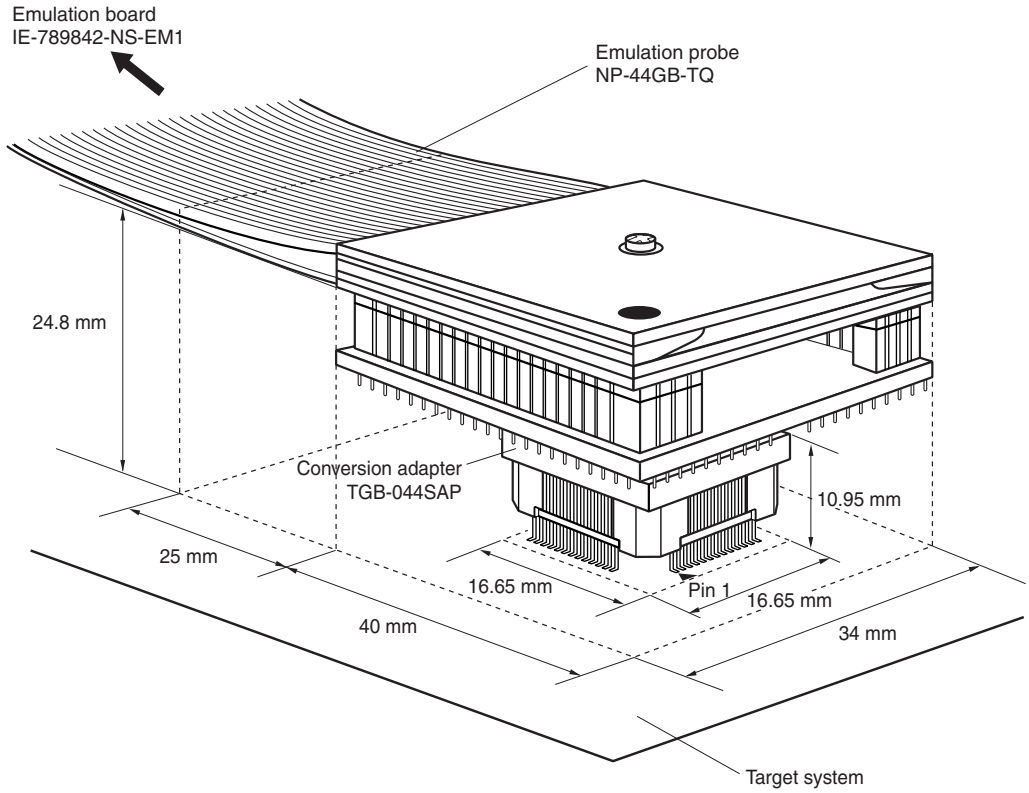
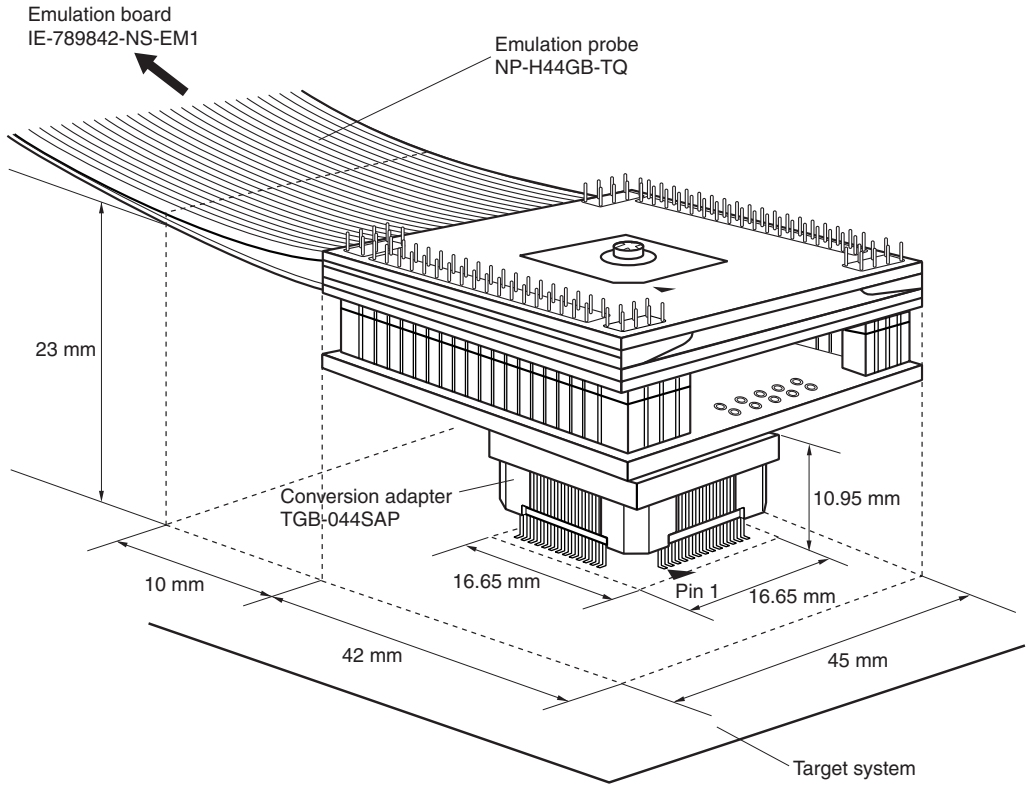


Figure B-3. Connection Conditions of Target System (NP-H44GB-TQ)



APPENDIX C REGISTER INDEX

C.1 Register Name Index (Alphabetic Order)

C.2 Register Symbol Index (Alphabetic Order)

APPENDIX D REVISION HISTORY

★ D.1 Major Revisions in This Edition

Page	Description
U13776JJ2V0UD00 → U13776JJ3V0UD00	
p.24	**CHAPTER 2 PIN FUNCTIONS** • Modification of description in **2.2.12 V_{PP} (μPD78F9842 only)**
pp.37, 38, 47, 51, 52	**CHAPTER 3 CPU ARCHITECTURE** • Modification of **Figure 3-10 Data to Be Saved to Stack Memory** • Modification of **Figure 3-11 Data to Be Restored from Stack Memory** • Modification of **[Description example]** in **3.4.2 Short direct addressing** • Addition of **[Illustration]** in **3.4.6 Based addressing** • Addition of **[Illustration]** in **3.4.7 Stack addressing**
pp.64, 68	**CHAPTER 5 CLOCK GENERATOR** • Modification of **Figure 5-2 Format of Processor Clock Control Register** • Modification of description in **5.5 Clock Generator Operation**
pp.91, 97	**CHAPTER 7 8-BIT TIMER/EVENT COUNTERS 80, 81, 82** • Addition of description in **7.3 (4) Port mode register 2 (PM2)** • Modification of description in **7.5 Notes on Using 8-Bit Timer/Event Counters 80, 81, 82**
pp.101, 103	**CHAPTER 8 WATCH TIMER** • Addition of **Caution 2** in **Figure 8-2 Format of Watch Timer Mode Control Register** • Addition of **Caution** in **Figure 8-3 Watch Timer/Interval Timer Operation Timing**
pp.118, 121	**CHAPTER 10 A/D CONVERTER** • Modification of **Figure 10-6 Software-Started A/D Conversion** • Addition of **(8) Input impedance of ANI0 to ANI7 pins** in **10.5 Notes on Using A/D Converter**
p.139	**CHAPTER 11 SERIAL INTERFACE** • Modification of **Caution** in **11.4.2 (2) (d) Reception**
pp.146, 147	**CHAPTER 14 INTERRUPT FUNCTIONS** • Addition of description in **14.1 Interrupt Function Types** • Addition of **Remark** in **Table 14-1 Interrupt Sources**
p.173	Total revision of **CHAPTER 17 μPD78F9842**
p.193	Addition of **CHAPTER 19 ELECTRICAL SPECIFICATIONS**
p.201	Addition of **CHAPTER 20 PACKAGE DRAWINGS**
p.203	Addition of **CHAPTER 21 RECOMMENDED SOLDERING CONDITIONS**
p.204	Total revision of description in **APPENDIX A DEVELOPMENT TOOLS**
p.210	Addition of **APPENDIX B NOTES ON TARGET SYSTEM DESIGN**
p.201 in 2nd edition	Deletion of **APPENDIX B EMBEDDED SOFTWARE**
U13776JJ3V0UD00 → U13776JJ3V1UD00	
p.14	Addition of lead-free products to **1.3 Ordering Information**
p.203	Addition of lead-free products to **CHAPTER 21 RECOMMENDED SOLDERING CONDITIONS**

D.2 Revisions up to Previous Edition

The revision history is described below. The “Applied to” column indicates the chapter in each edition.

Edition	Major Revisions from Previous Edition	Applied to
2nd edition	Change of μPD789841, 789842, and 78F9842 status from under development to development completed	Throughout
	Deletion of flash programmer Flashpro II	
	Addition of handling of AV_{DD} and AV_{SS} pins to **Table 2-1 Type of I/O Circuit for Each Pin and Handling of Unused Pins**	**CHAPTER 2 PIN FUNCTIONS**
	Addition of caution description on rewriting CR8n in **7.2 (1) 8-bit compare register 8n (CR8n)**	**CHAPTER 7 8-BIT TIMER/EVENT COUNTER**
	Modification of description of operations in **7.4.1 Operation as interval timer**	
	Modification of description of operations in **7.4.2 Operation as external event counter**	
	Modification of description of operations in **7.4.3 Operation as square-wave output**	
	Addition of **17.1.4 Example of settings for Flashpro III (PG-FP3)**	**CHAPTER 17 μPD78F9842**
	Revision of **APPENDIX A DEVELOPMENT TOOLS**	**APPENDIX A DEVELOPMENT TOOLS**