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

Preliminary User's Manual

μPD789871 Subseries

8-Bit Single-Chip Microcontrollers

μPD789870
μPD789871
μPD78F9872

Document No. U14938EJ1V0UM00 (1st edition)
Date Published January 2001 N CP(K)

[MEMO]

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

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

Licence not needed: μPD78F9872

The customer must judge the need for license: μPD789870, 789871

- **The information contained in this document is being issued in advance of the production cycle for the device. The parameters for the device may change before final production or NEC Corporation, at its own discretion, may withdraw the device prior to its production.**
- **Not all devices/types available in every country. Please check with local NEC 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 Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.
- NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.
- 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 the customer's equipment shall be done under the full responsibility of the customer. NEC Corporation assumes no responsibility for any losses incurred by the customer or third parties arising from the use of these circuits, software, and information.
- While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.
- NEC devices are classified into the following three quality grades:
  "Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated "quality assurance program" for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device before using it in a particular application.
  - Standard: Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots
  - Special: Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support)
  - Specific: Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.

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

Regional Information

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

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

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

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

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

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

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

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

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

NEC Electronics (France) S.A.
Madrid Office
Madrid, Spain
Tel: 91-504-2787
Fax: 91-504-2860

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

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

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

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

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

NEC do Brasil S.A.
Electron Devices Division
Guarulhos-SP Brasil
Tel: 55-11-6462-6810
Fax: 55-11-6462-6829

J00.7

INTRODUCTION

Target Readers

This manual is intended for users who wish to understand the functions of the μPD789871 Subseries and to design and develop application systems and programs using these microcontrollers.

Purpose

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

Organization

The μPD789871 Subseries User's Manual is divided into two parts: this manual and instructions (common to the 78K/0S Series).

μPD789871 Subseries User's Manual (This manual)	78K/0S Series User's Manual Instructions
• Pin functions • Internal block functions • Interrupt • Other internal peripheral functions	• CPU function • Instruction set • Instruction description

How to Read This Manual

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

☐ To understand the overall functions in general
→ Read this manual in the order of the **CONTENTS**.
☐ How to interpret register formats
→ The name of a bit whose number is encircled is reserved for the assembler and is defined for the C compiler by the header file **sfrbit.h**.
☐ To learn the detailed functions of a register whose register name is known
→ Refer to **APPENDIX C REGISTER INDEX**.
☐ To learn the details of the instruction functions of the 78K/0S Series
→ Refer to **78K/0S Series User's Manual Instructions (U11047E)**.

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.
	English
μPD789870, 789871 Preliminary Product Information	U14916E
μPD78F9872 Preliminary Product Information	U14880E
μPD789871 Subseries User's Manual	This manual
78K/0S Series User's Manual Instruction	U11047E
78K/0, 78K/0S Series Application Note Flash Memory Write	U14458E

Documents Related to Development Tools (User's Manuals)

Document Name		Document No.
		English
RA78K0S Assembler Package	Operation	U11622E
	Assembly Language	U11599E
	Structured Assembly Language	U11623E
CC78K0S C Compiler	Operation	U11816E
	Language	U11817E
SM78K0S System Simulator Windows™ Based	Reference	U11489E
SM78K Series System Simulator	External components user open interface specification	U10092E
ID78K0S-NS Integrated Debugger Windows Based	Reference	U12901E
IE-78K0S-NS In-Circuit Emulator		U13549E
IE-789872-NS-EM1 Emulation Board		To be prepared

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

Document Related to Embedded Software (User's Manual)

Document Name		Document No.
		English
78K/0S Series OS MX78K0S	Basics	U12938E

Other Documents

Document Name	Document No.
	English
SEMICONDUCTOR SELECTION GUIDE Products & Packages (CD-ROM)	X13769X
Semiconductor Device Mounting Technology Manual	C10535E
Quality Grades on NEC Semiconductor Devices	C11531E
NEC Semiconductor Device Reliability/Quality Control System	C10983E
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892E

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

CONTENTS

LIST OF FIGURES (1/3)

LIST OF FIGURES (2/3)

LIST OF FIGURES (3/3)

LIST OF TABLES (1/2)

LIST OF TABLES (2/2)

CHAPTER 1 GENERAL

1.1 Features

- ROM and RAM capacities

Item / Part Number	Program Memory (ROM)		Data Memory	
			Internal High-Speed RAM	LCD Display RAM
μPD789870	ROM	4 KB	512 bytes	96 bytes
μPD789871		8 KB		
μPD78F9872	Flash memory	16 KB		

- Minimum instruction execution time can be changed from high-speed (0.4 μs: @ 5.0 MHz operation with main system clock) to ultra-low-speed (122 μs: @ 32.768 kHz operation with subsystem clock)
- I/O ports: 33
- Timer: 5 channels
  - 8-bit remote control timer: 1 channel
  - 8-bit timer: 2 channels
  - Watch timer: 1 channel
  - Watchdog timer: 1 channel
- Serial interface: 1 channel
- VFD controller/driver: Display output total 25
- Vectored interrupt sources: 12
- Power supply voltage: V_{DD} = 2.7 to 5.5 V (in normal operation)
  V_{DD} = 4.5 to 5.5 V (in VFD operation)
- Operating ambient temperature: T_A = –40 to +85°C

1.2 Applications

Products with a front panel such as DVD, VCD, and S-VCD.

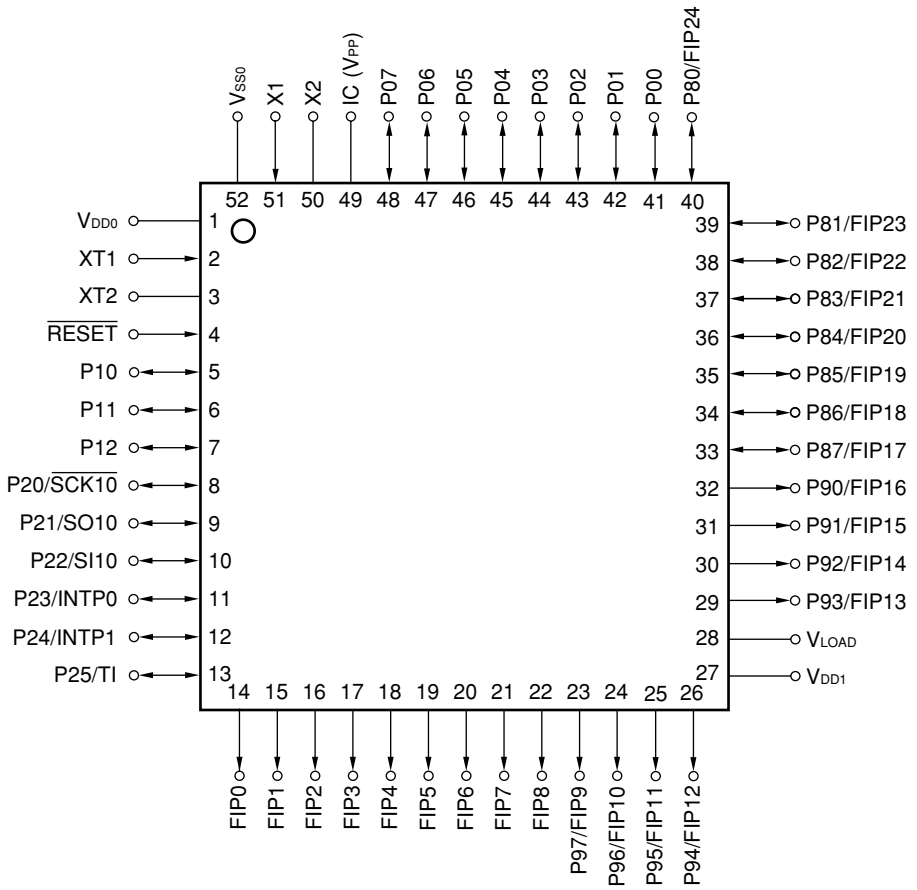
1.3 Ordering Information

Part Number	Package	Internal ROM
μPD789870GB-×××-8ET	52-pin plastic LQFP (10 × 10)	Mask ROM
μPD789871GB-×××-8ET	52-pin plastic LQFP (10 × 10)	Mask ROM
μPD78F9872GB-8ET	52-pin plastic LQFP (10 × 10)	Flash memory

Remark ××× indicates ROM code suffix.

1.4 Pin Configuration (Top View)

- 52-pin plastic LQFP (10 × 10)
  μPD789870GB-×××-8ET
  μPD789871GB-×××-8ET
  μPD78F9872GB-8ET



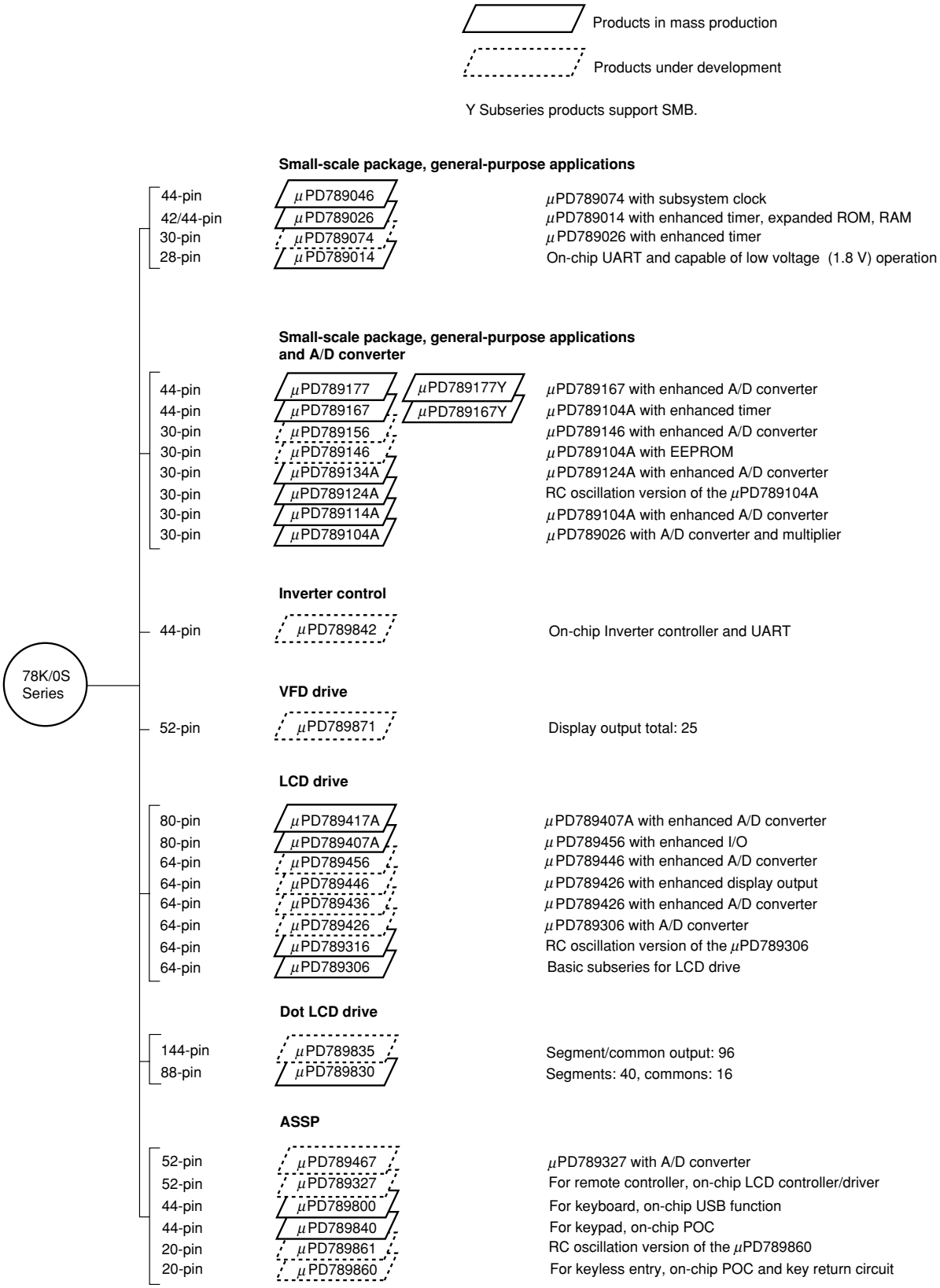
Caution Connect the IC (Internally Connected) pin directly to V_{SS0}.

Remark The parenthesized values apply to μPD78F9872.

FIP0 to FIP24:	Fluorescent Indicator Panel
IC:	Internally Connected
INTP0, INTP1:	Interrupt from Peripherals
P00 to P07:	Port 0
P10 to P12:	Port 1
P20 to P25:	Port 2
P80 to P87:	Port 8
P90 to P97:	Port 9
RESET:	Reset
SCK10:	Serial Clock
SI10:	Serial Input
SO10:	Serial Output
TI:	Remote Control Timer Input
V_{DD0},V_{DD1} :	Power Supply
V_{LOAD}:	Negative Power Supply
V_{PP}:	Programming Power Supply
V_{SS0}:	Ground
X1, X2:	Crystal (Main System Clock)
XT1, XT2:	Crystal (Subsystem Clock)

1.5 78K/0S Series Lineup

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

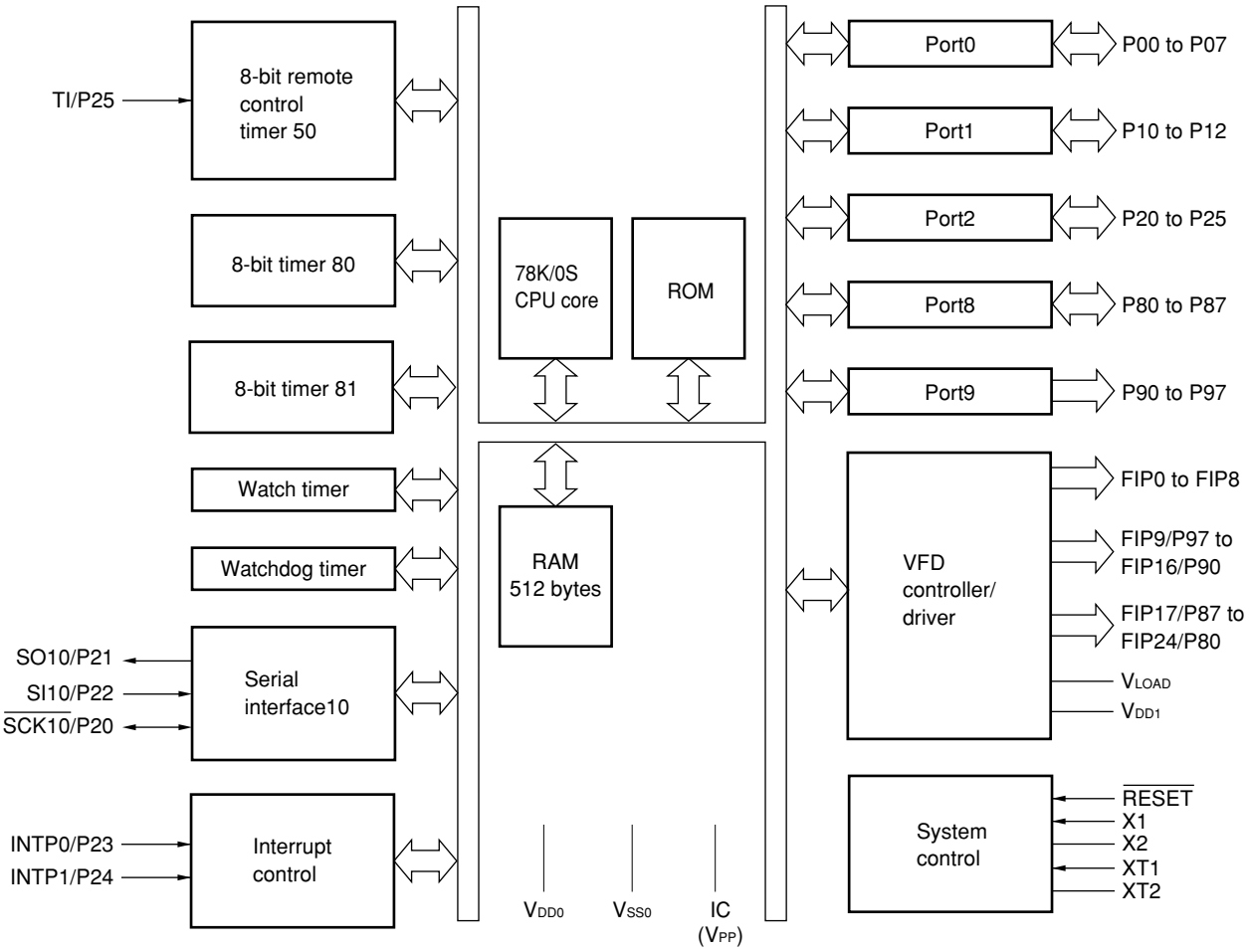


The major functional differences among the subseries are listed below.

Subseries Name \ Function		ROM Capacity	8-Bit	16-Bit	Watch	WDT	8-Bit A/D	10-Bit A/D	Serial Interface	I/O	V_{DD} MIN. Value	Remarks
Small-scale package general-purpose applications	μPD789046	16 K	1 ch	1 ch	1 ch	1 ch	–	–	1 ch (UART: 1 ch)	34	1.8 V	–
	μPD789026	4 K to 16 K			–							
	μPD789074	2 K to 8 K								24		
	μPD789014	2 K to 4 K	2 ch	–						22		
Small-scale package general-purpose applications and A/D converter	μPD789177	16 K to 24 K	3 ch	1 ch	1 ch		–	8 ch	1 ch (UART: 1 ch)	31		–
	μPD789167						8 ch	–				
	μPD789156	8 K to 16 K	1 ch		–		–	4 ch		20		On-chip EEPROM
	μPD789146						4 ch	–				
	μPD789134A	2 K to 8 K					–	4 ch				RC-oscillation version
	μPD789124A						4 ch	–				
	μPD789114A						–	4 ch				–
	μPD789104A						4 ch	–				
Inverter control	μPD789842	8 K to 16 K	3 ch	**Note**	1 ch	1 ch	8 ch	–	1 ch (UART: 1 ch)	30	4.0 V	–
For LCD driving	μPD789871	4 K to 8 K	3 ch	–	1 ch	1 ch	–	–	1 ch	33	2.7 V	–
	μPD789417A	12K to 24 K	3 ch	1 ch	1 ch	1 ch	–	7 ch	1 ch (UART: 1 ch)	43	1.8 V	–
	μPD789407A						7 ch	–				
	μPD789456	12K to 16 K	2 ch				–	6 ch		30		
	μPD789446						6 ch	–				
	μPD789436						–	6 ch		40		
	μPD789426						6 ch	–				
	μPD789316	8 K to 16 K					–		2 ch (UART: 1 ch)	23		RC-oscillation version
	μPD789306											–
For Dot LCD driving	μPD789835	12K to 60 K	6 ch	–	1 ch	1 ch	3 ch	–	1 ch (UART: 1 ch)	28	1.8 V	–
	μPD789830	24 K	1 ch	1 ch			–			30	2.7 V	
ASSP	μPD789467	4 K to 24 K	2 ch	–	1 ch	1 ch	1 ch	–	–	18	1.8 V	On-chip LCD
	μPD789327						–		1 ch	21		
	μPD789800	8 K			–				2 ch (USB: 1 ch)	31	4.0 V	–
	μPD789840						4 ch		1 ch	29	2.8 V	
	μPD789861	4 K					–		–	14	1.8 V	RC-oscillation version, on-chip EEPROM
	μPD789860											On-chip EEPROM

Note 10-bit timer: 1 channel

1.6 Block Diagram



Remarks 1. The internal ROM capacity varies depending on the product.

2. The parenthesized values apply to μPD78F9872.

An outline of the timer is shown below.

		8-Bit Remote Control Timer	8-Bit Timer	Watch Timer	Watchdog Timer
Operation mode	Interval timer	–	1 channel	1 channel[Note 1]	1 channel[Note 2]
Function	Pulse width measurement	1 input	–	–	–
	Interrupt sources	3	2	1	1

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

2. The watchdog timer has the watchdog timer and interval timer function. However, use the watchdog timer by selecting either the watchdog timer function or interval timer function.

1.7 Overview of Functions

Item		μPD789870	μPD789871	μPD78F9872
Internal memory	ROM	Mask ROM		Flash memory
		4 KB	8 KB	16 KB
	High-speed RAM	512 bytes		
	VFD display RAM	96 bytes		
Minimum instruction execution time		• 0.4 μs/1.6 μs (@ 5.0 MHz operation with main system clock) •122 μs (@ 32.768 kHz operation with subsystem clock)		
General-purpose registers		8 bits × 8 registers		
Instruction set		• 16-bit operations • Bit manipulations (such as set, reset, and test)		
I/O ports		Total: 33 • CMOS I/O: 17 • P-ch open-drain I/O: 8 • P-ch open-drain output: 8		
VFD controller/driver Display output total:		25		
Serial interfaces		3-wire serial I/O: 1 channel		
Timers		• 8-bit remote control timer: 1 channel • 8-bit timer: 2 channels • Watch timer: 1 channel • Watchdog timer: 1 channel		
Vectored interrupt sources	Maskable	Internal: 8, external: 4		
	Non-maskable	Internal: 1		
Power supply voltage		V_{DD} = 2.7 to 5.5 V (in normal operation) V_{DD} = 4.5 to 5.5 V (in VFD operation)		
Operating ambient temperature		T_A = –40 to +85°C		
Package		52-pin plastic LQFP (10 × 10)		

CHAPTER 2 PIN FUNCTIONS

2.1 List of Pin Functions

(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, an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).	Input	—
P10 to P12	I/O	Port 1. 3-bit I/O port. Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of 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. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register B2 (PUB2).	Input	$\overline{\text{SCK10}}$
P21				SO10
P22				SI10
P23				INTP0
P24				INTP1
P25				TI
P80 to P87	I/O	Port 8. P-ch open-drain 8-bit high-tolerance I/O port. For mask ROM versions, use of a pull-down resistor for V_{LOAD} can be specified in 1-bit units by a mask option (when used as a general-purpose I/O port, the pull-down resistor is connected to V_{SS0}).	Output	FIP24 to FIP17
P90 to P97	Output	Port 9 P-ch open-drain 8-bit high-tolerance output port. Mask ROM versions include an on-chip pull-down resistor (connected to V_{LOAD}).	Output	FIP16 to FIP9

(2) Non-port pins

Pin Name	I/O	Function	After Reset	Alternate Function
INTP0	Input	External interrupt input for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified	Input	P23
INTP1				P24
SI10	Input	Serial interface serial data input	Input	P22
SO10	Output	Serial interface serial data output	Input	P21
$\overline{SCK10}$	I/O	Serial interface serial clock input/output	Input	P20
TI	Input	Timer input to 8-bit remote control timer	Input	P25
FIP0 to FIP8	Output	VFD controller/driver high-tolerance high current output	Output	—
FIP9 to FIP16				P97 to P90
FIP17 to FIP24[Note]				P87 to P80
X1	Input	Connecting crystal resonator for main system clock oscillation	—	—
X2	—		—	—
XT1	Input	Connecting crystal resonator for Subsystem clock oscillation	—	—
XT2	—		—	—
V_{LOAD}	—	Connecting pull-down resistor of VFD controller/driver	—	—
V_{DD0}	—	Positive power supply for ports	—	—
V_{DD1}	—	Positive power supply for VFD controller/driver	—	—
V_{SS0}	—	Ground potential	—	—
$\overline{RESET}$	Input	System reset input	Input	—
IC	—	Internally connected. Connect directly to V_{SS0}.	—	—
V_{PP}	—	Sets flash memory programming mode. Applies high voltage when a program is written or verified. Connect directly to V_{SS0} in normal operation mode.	—	—

Note Pins set as P-ch open-drain I/O port by a mask option cannot be used as VFD controller/driver.

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 in input or output port mode in 1-bit units by using port mode register 0 (PM0). When used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 0 (PU0) in port units.

2.2.2 P10 to P12 (Port 1)

These pins constitute a 3-bit I/O port and can be set in input or output port mode in 1-bit units by port mode register 1 (PM1). When used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 0 (PU0) in port units.

2.2.3 P20 to P25 (Port 2)

These pins constitute a 6-bit I/O port. In addition, these pins enable timer input serial interface data I/O, and external interrupt input.

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

(1) Port mode

In this mode, port 2 functions as a 6-bit I/O port. Port 2 can be set in the input or output mode in 1-bit units by port mode register 2 (PM2). When used as an input port, use of an on-chip pull-up resist can be specified by pull-up resistor option register B2 (PUB2) in 1-bit units.

(2) Control mode

In this mode, P20 to P25 function as the timer input, serial interface data I/O, and external interrupt input.

(a) TI

This is the timer input pin of 8-bit remote control timer.

(b) SI10, SO10

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

(c) $\overline{\text{SCK10}}$

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

(d) 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.

2.2.4 P80 to P87 (Port 8)

These pins constitute an 8-bit P-ch open-drain I/O port. In addition, they also function as VFD controller/driver outputs. Port 8 can be specified in the following operation mode in 1-bit units.

(1) Port mode

In this mode, port 8 functions as an 8-bit P-ch open-drain I/O Port. Port 8 can be set in the input or output port mode in 1-bit units by port mode register 8 (PM8).

(2) Control mode

In this mode, these pins function as VFD controller/driver outputs (FIP17 to FIP24). For mask ROM versions, use of a pull-down resistor for V_{LOAD} can be specified by a mask option.

2.2.5 P90 to P97 (Port 9)

These pins constitute an 8-bit P-ch open-drain I/O port. In addition, they also function as VFD controller/driver outputs. Port 9 can be specified in the following operation modes in 1-bit units.

(1) Port mode

In this mode, port 9 functions as a P-ch open-drain I/O port. Port 9 can be set in the input or output port mode in 1-bit units by port mode register 9 (PM9).

(2) Control mode

In this mode, these pins function as VFD controller/driver outputs (FIP9 to FIP16). Mask ROM versions include an on-chip pull-down resistor (connected to V_{LOAD}).

2.2.6 FIP0 to FIP8

These pins are output pins for the VFD controller/driver.

2.2.7 $\overline{\text{RESET}}$

This pin inputs an active-low system reset signal.

2.2.8 X1, X2

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

2.2.9 XT1, XT2

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

2.2.10 V_{DD0}

This is the positive power supply pin.

2.2.11 V_{DD1}

This is the positive power supply pin for VFD controller/driver.

2.2.12 V_{LOAD}

This pin is used to connect a pull-down resistor of VFD controller/driver.

2.2.13 V_{SS0}

This is the ground pin.

2.2.14 V_{PP} (μPD78F9872 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.

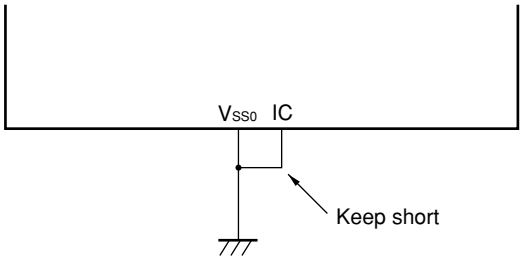
Directly connect this pin to V_{SS0} in the normal operation mode.

2.2.15 IC (mask ROM version only)

The IC (Internally Connected) pin is used to set the μPD789870 and μPD789871 in the test mode before shipment. In the normal operation mode, directly connect this pin to the V_{SS0} pin with as short a wiring length as possible.

If a potential difference is generated between the IC pin and V_{SS0} pin due to a long wiring length, or an external noise superimposed on the IC pin, the user program may not run correctly.

- Directly connect the IC pin to the V_{SS0} pin.



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

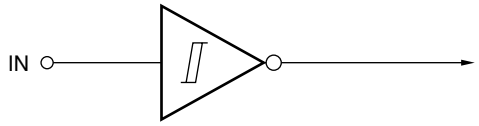
The I/O circuit type for each pin and recommended connection of unused pins are shown in Table 2-1. For the input/output circuit configuration of each type, see Figure 2-1.

Table 2-1. Types of Pin I/O Circuits

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
P00 to P07	5-H	I/O	Input: Independently connect to V_{DD0} or V_{SS0} via a resistor. Output: Leave open.
P10 to P12			
P20/$\overline{SCK10}$	8-C		
P21/SO10	5-H		
P22/SI10	8-C		
P23/INTP0			
P24/INTP1			
P25/TI			
FIP0 to FIP8	14-F	Output	Leave open.
FIP9/P97 to FIP16/P90			
FIP17/P87 to FIP24/P80 (Mask ROM version)	15-G	I/O	
FIP17/P87 to FIP24/P80 (Flash memory version)	15-H		
$\overline{RESET}$	2	Input	—
IC	—	—	Connect directly to V_{SS0}.
V_{PP}	—	—	

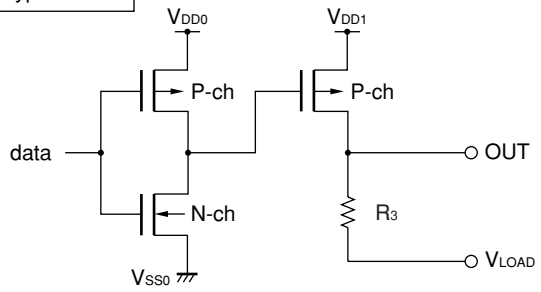
Figure 2-1. Pin I/O Circuits

Type 2



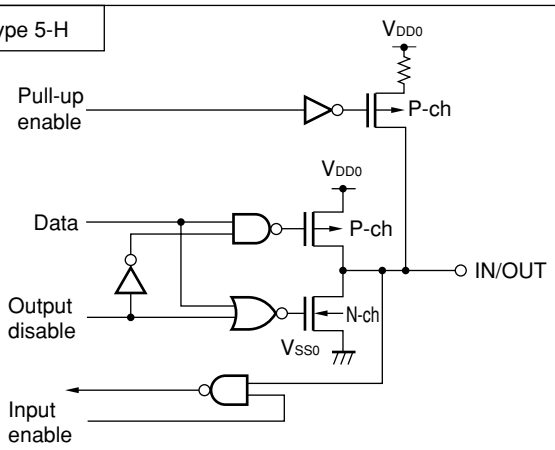
Schmitt-triggered input with hysteresis characteristics.

Type 14-F

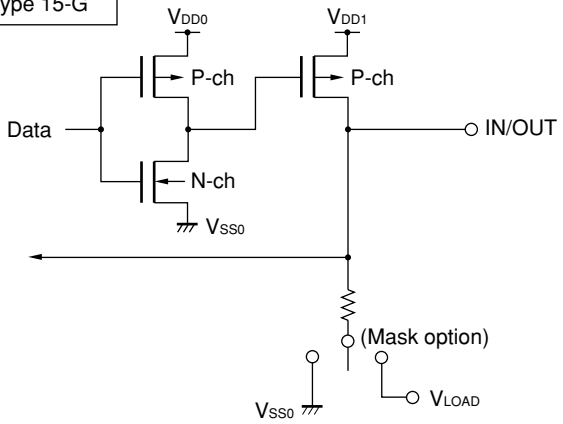


Remark μPD78F9872 does not include an on-chip pull-down resistor at FIP9/P97 to FIP16/P90.

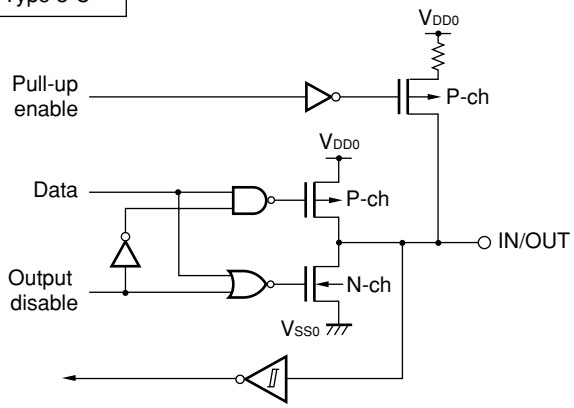
Type 5-H



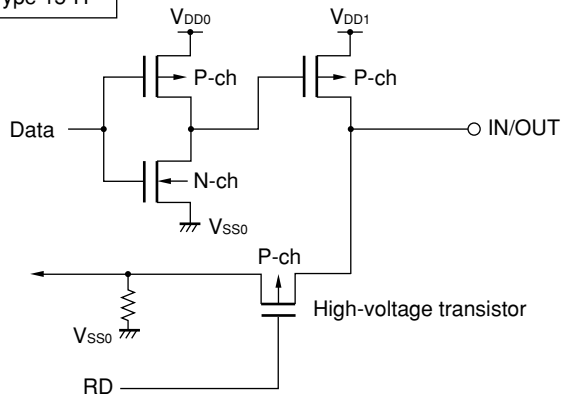
Type 15-G



Type 8-C



Type 15-H



CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Space

The μPD789871 Subseries can access 64 KB of memory space. Figures 3-1 to 3-3 show the memory maps.

Figure 3-1. Memory Map (μPD789870)

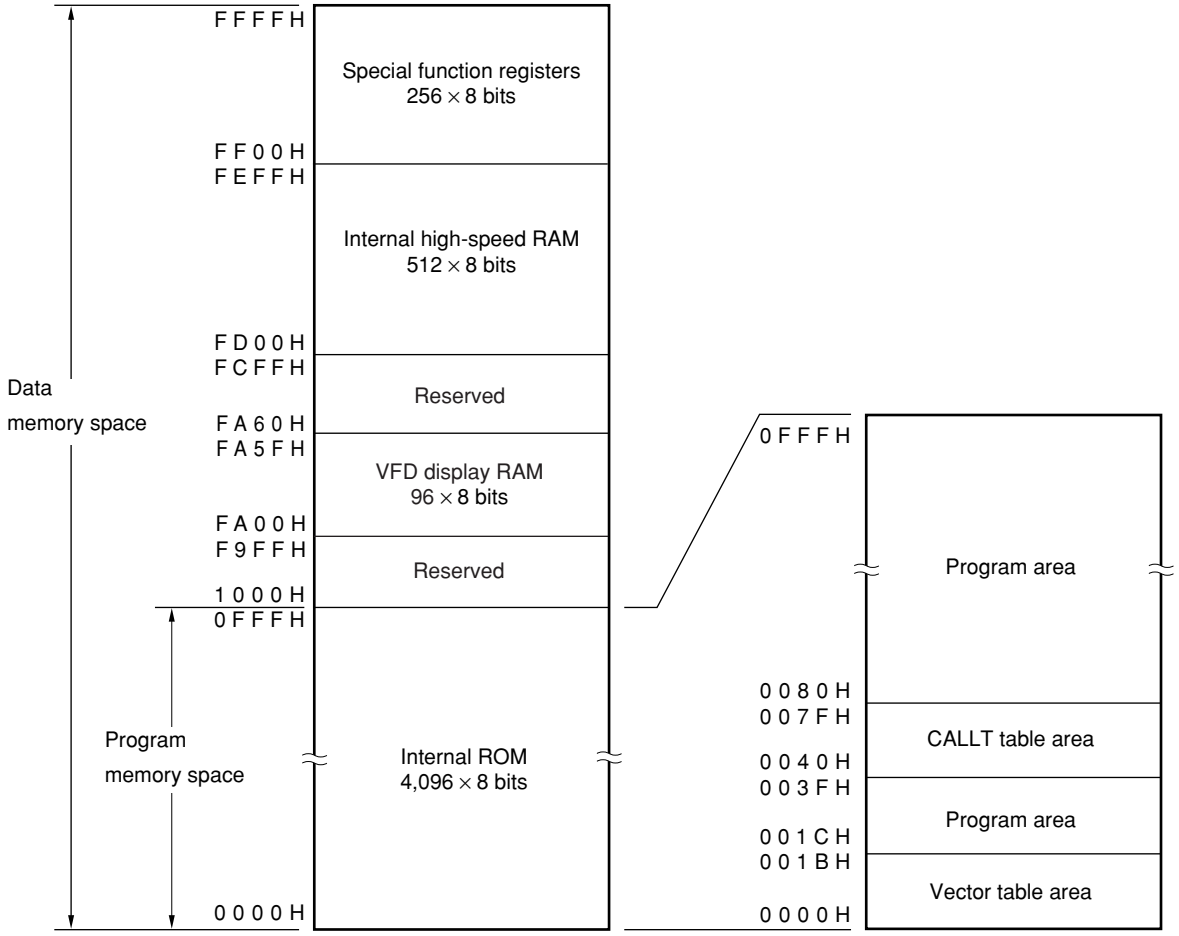


Figure 3-2. Memory Map (μPD789871)

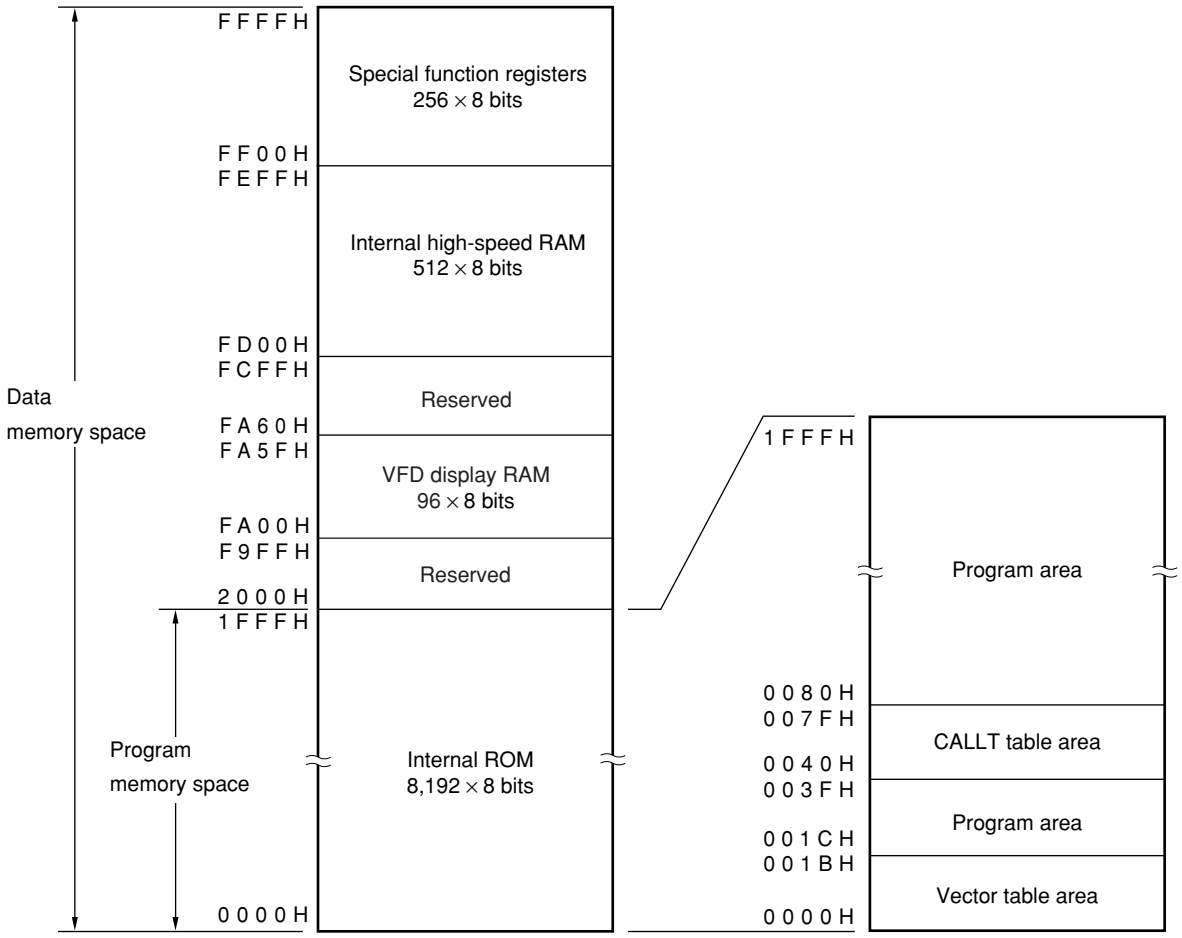
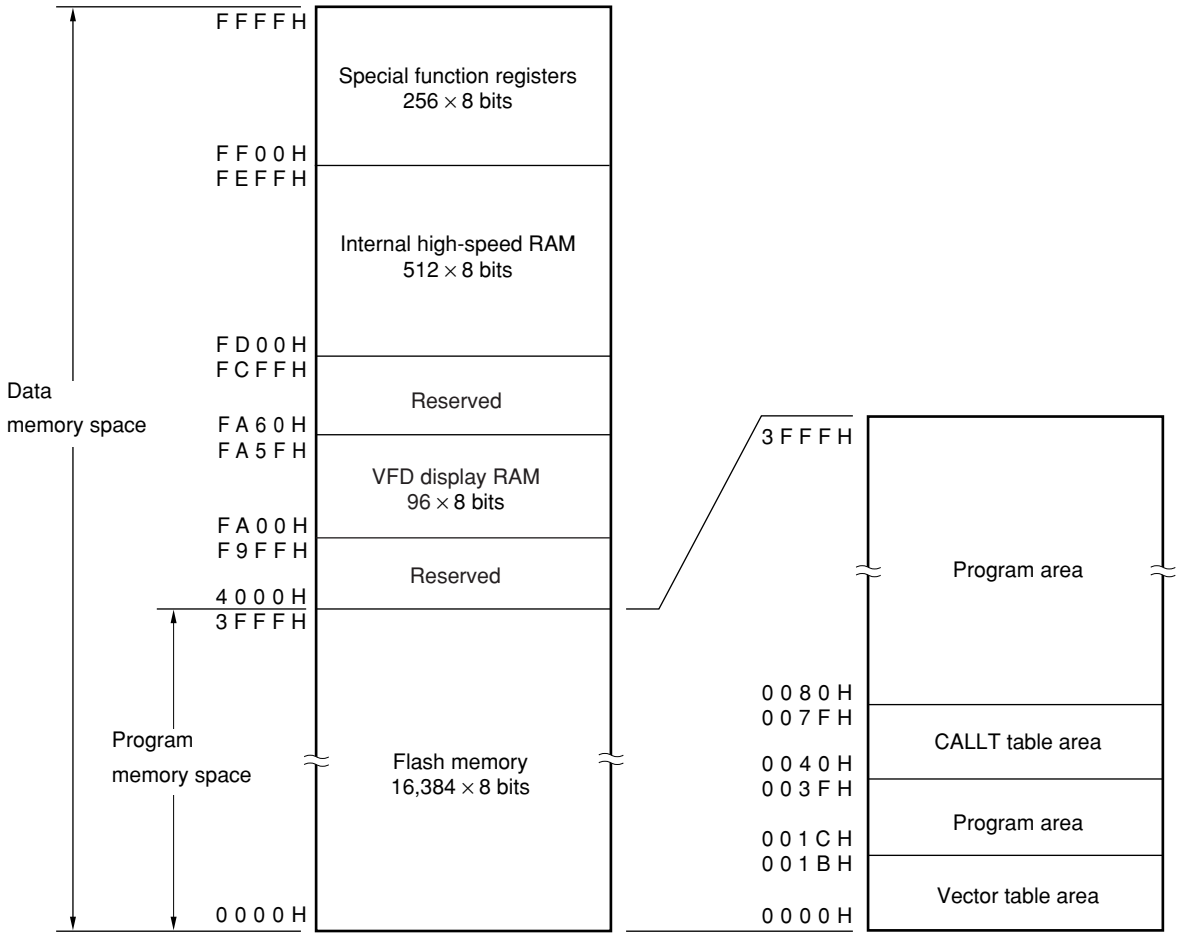


Figure 3-3. Memory Map (μPD78F9872)



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 μPD789871 Subseries provides the following internal ROMs (or flash memory) containing the following capacities.

Table 3-1. Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
μPD789870	Mask ROM	4,096 × 8 bits
μPD789871		8,192 × 8 bits
μPD78F9872	Flash memory	16,384 × 8 bits

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

(1) Vector table area

A 28-byte area of addresses 0000H to 001BH is reserved as a vector table area. This area stores program start addresses to be used when branching by the $\overline{\text{RESET}}$ input or an 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	INTKS
0004H	INTWDT	0012H	INTCSI10
0006H	INTP0	0014H	INTTM80
0008H	INTP1	0016H	INTTM81
000AH	INTTM50	0018H	INTWT
000CH	INTTM51	001AH	INTWTI
000EH	INTTM52		

(2) CALLT instruction table area

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

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

The μPD789871 Subseries provides following internal RAMs.

(1) Internal high-speed RAM

An Internal high-speed RAM is incorporated in the area of FD00H to FEFFH. This RAM can also be used as a stack.

(2) VFD display RAM

A VFD display RAM is allocated to the area of FA00H to FA5FH (96 bytes). This RAM can also be used as an ordinary RAM.

3.1.3 Special function register (SFR) area

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

3.1.4 Data memory addressing

The μPD789871 Subseries provides a variety of addressing modes which take account of memory manipulability, etc. Especially at addresses corresponding to data memory area (FE00H to FEFFH), particular addressing modes are possible to meet the functions of the special function registers (SFRs) and general-purpose registers. Figures 3-4 to 3-6 show the data memory addressing modes.

Figure 3-4. Data Memory Addressing (μPD789870)

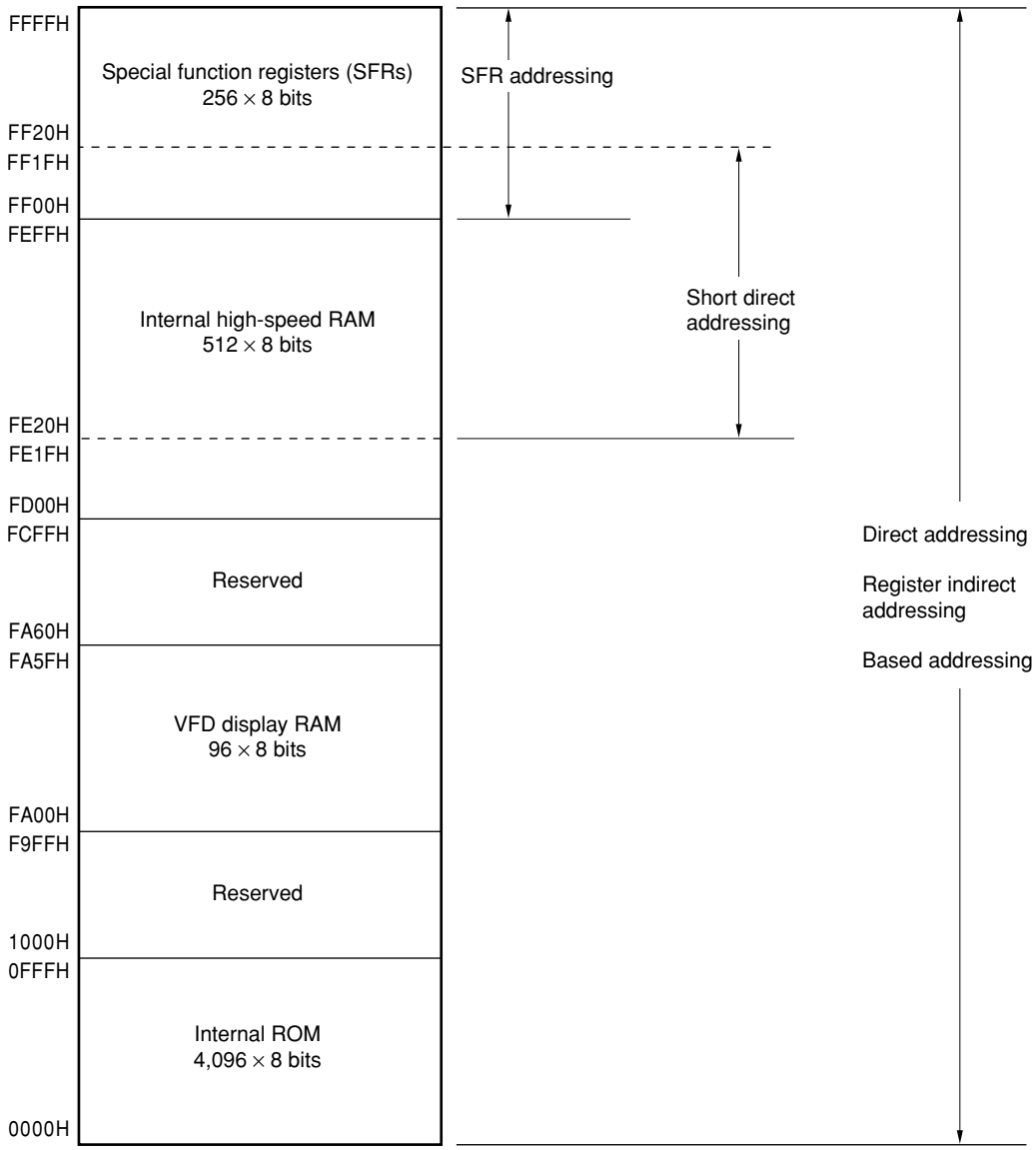


Figure 3-5. Data Memory Addressing (μPD789871)

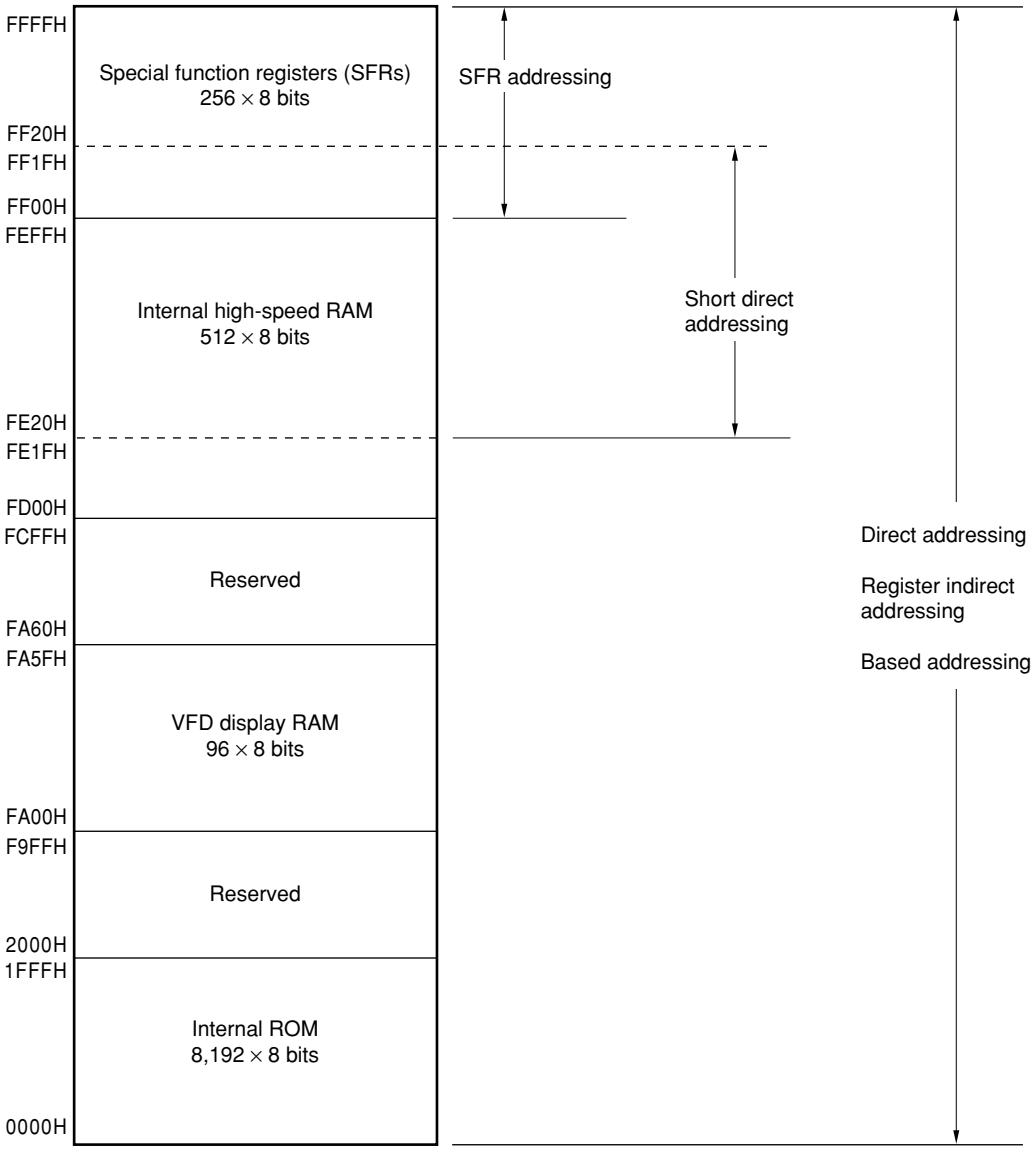
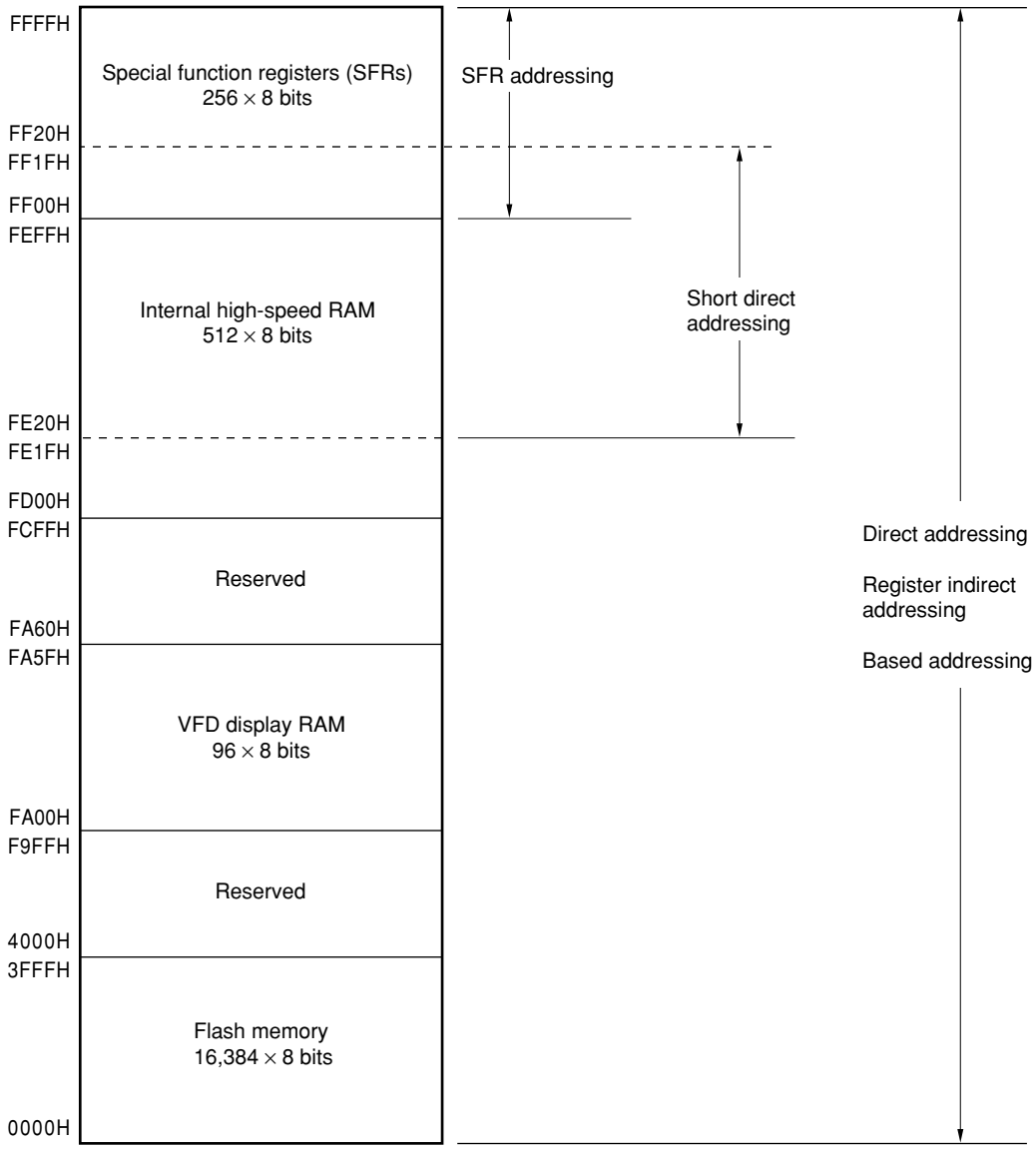


Figure 3-6. Data Memory Addressing (μPD78F9872)



3.2 Processor Registers

The μPD789871 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 control registers.

(1) Program counter (PC)

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

In normal operation, the PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data or register contents is 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/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 interrupt request acknowledge operations of CPU.

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

When IE = 1, the IE is set to interrupt enabled (EI) status and interrupt request acknowledgement is controlled with an interrupt mask flag for various interrupt sources.

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

(b) Zero flag (Z)

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

(c) 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 (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 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/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

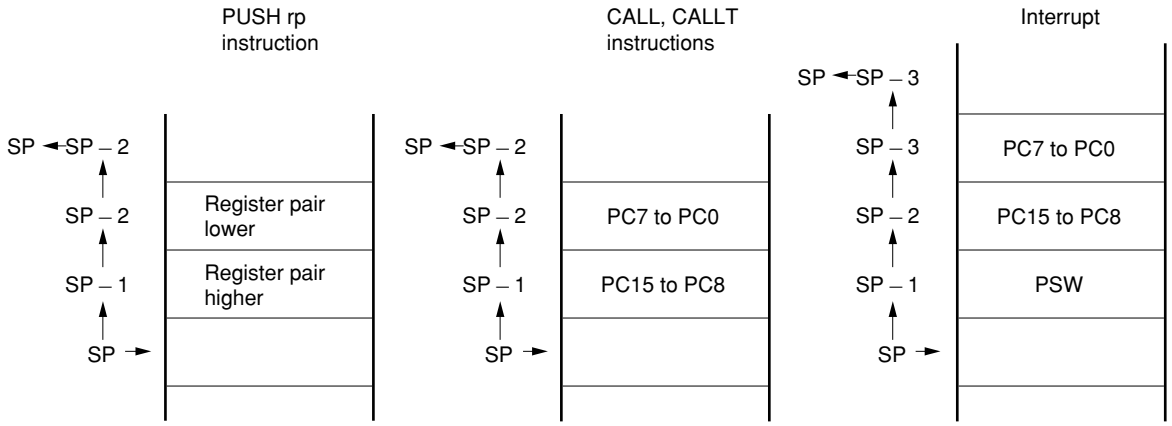
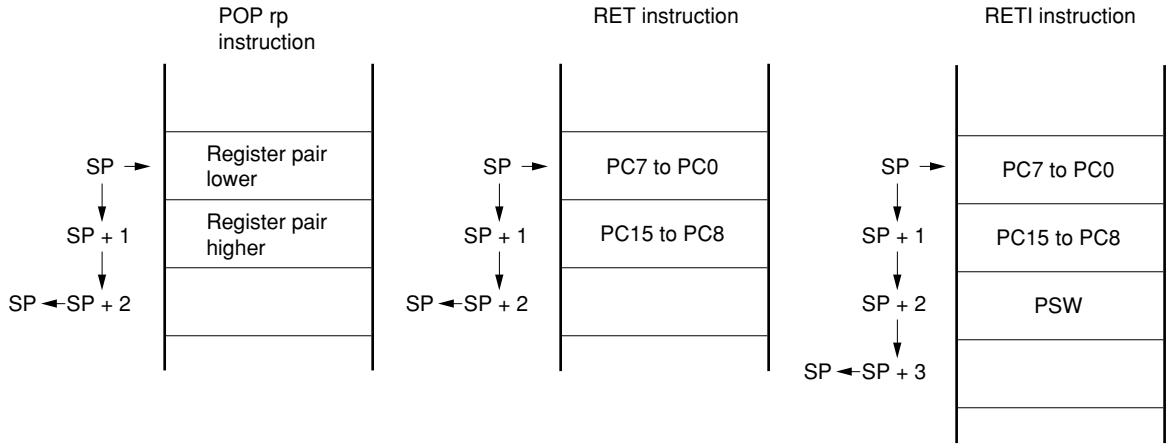


Figure 3-11. Data to Be Restored from Stack Memory



3.2.2 General-purpose registers

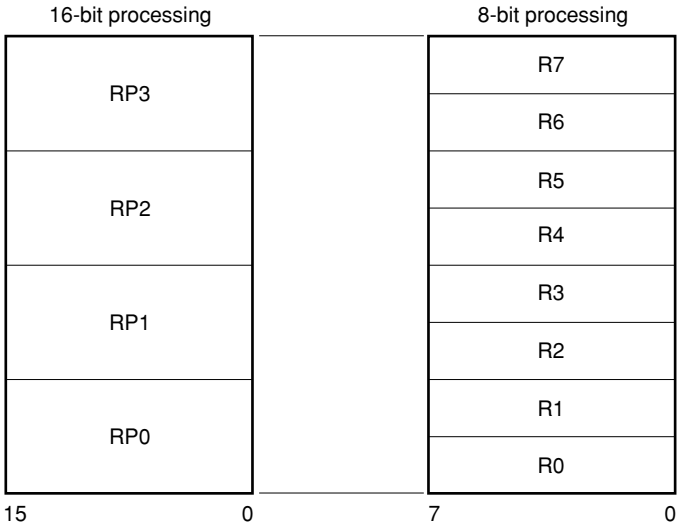
The general-purpose registers consist of eight 8-bit registers (X, A, C, B, E, D, L, and H).

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

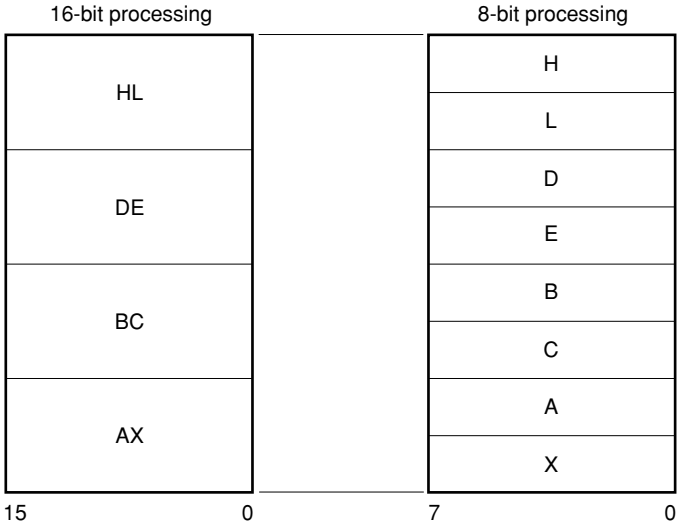
They can be described in terms of functional names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

Figure 3-12. General-Purpose Register Configuration

(a) Absolute names



(b) Functional names



3.2.3 Special function registers (SFRs)

Unlike general-purpose registers, special function registers have their own functions and are allocated to the 256-byte area FF00H to FFFFH.

Special function registers can be manipulated, like general-purpose registers, with operation, transfer, and bit manipulation instructions. The bit units in which one register can be manipulated (1, 8, and 16) differ depending on the special function register type.

Each bit unit for manipulation can be specified as follows.

- 1-bit manipulation
  A symbol reserved by assembler is described as the operand (sfr.bit) of a 1-bit manipulation instruction. This manipulation can also be specified with an address.

- 8-bit manipulation
  A symbol reserved by assembler is described as the operand (sfr) of an 8-bit manipulation instruction. This manipulation can also be specified with an address.

- 16-bit manipulation
  A symbol reserved by assembler is described 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 the reserved words of the assembler, and have already been defined in the header file called “sfrbit.h” of the C compiler. Therefore, these symbols can be used as instruction operands if 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

- Bit units for manipulation
  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 Register List

Address	Special Function Register (SFR) Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FF00H	Port 0	P0	R/W	√	√	—	00H
FF01H	Port 1	P1		√	√	—	
FF02H	Port 2	P2		√	√	—	
FF08H	Port 8	P8		√	√	—	
FF09H	Port 9	P9		√	√	—	
FF20H	Port mode register 0	PM0		√	√	—	FFH
FF21H	Port mode register 1	PM1		√	√	—	
FF22H	Port mode register 2	PM2		√	√	—	
FF32H	Pull-up register option register B2	PUB2		√	√	—	00H
FF42H	Watchdog time clock select register	WDCS		—	√	—	
FF4AH	Watch time 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	√	√	—	
FF58H	Remote control timer control register 50	TMC50		√	√	—	
FF5AH	Remote control timer capture register 50	CP50	R	—	√	—	
FF5BH	Remote control timer capture register 51	CP51		—	√	—	
FF72H	Serial operation mode register 11	CSIM10	R/W	√	√	—	
FF74H	Transmit/receive shift register 10	SIO10		—	√	—	Undefined
FFA0H	Display mode register 0	DSPM0		√	√	—	10H
FFA1H	Display mode register 1	DSPM1		√	√	—	01H
FFA2H	Display mode register 2	DSPM2		√	√	—	00H
FFE0H	Interrupt request flag register 0	IF0		√	√	—	
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
FFF0H	Suboscillation mode register	SCKM		√	√	—	
FFF2H	Subclock control register	CSS		√	√	—	
FFF7H	Pull-up register option register 0	PU0		√	√	—	
FFF9H	Watchdog timer mode register	WDTM		√	√	—	
FFFAH	Oscillation stabilization timer select register	OSTS		—	√	—	04H
FFFBH	Processor clock control register	PCC		√	√	—	02H

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 and branched by the following addressing (For details of each instruction, refer to **78K/0S Series User's Manual Instruction (U11047E)**).

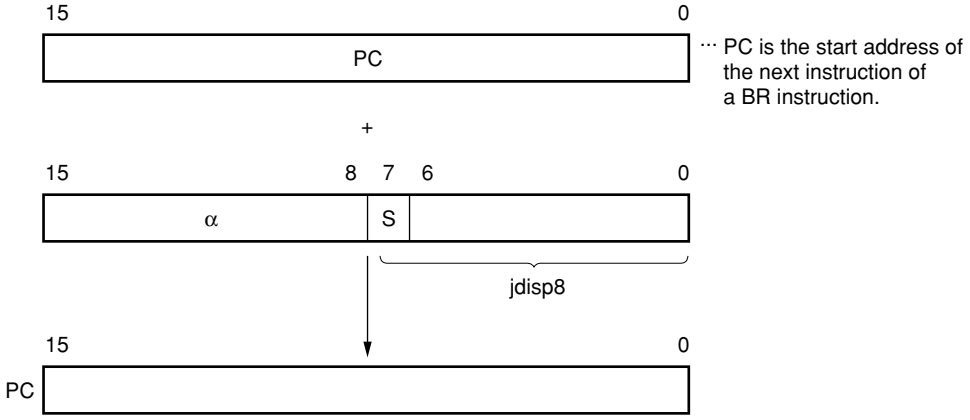
3.3.1 Relative addressing

[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]

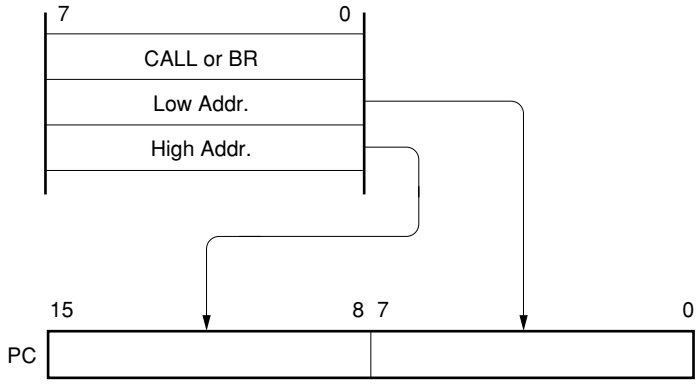
Immediate data in the instruction word is transferred to the program counter (PC) and branched.

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

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

[Illustration]

In case of CALL !addr16, BR !addr16 instruction



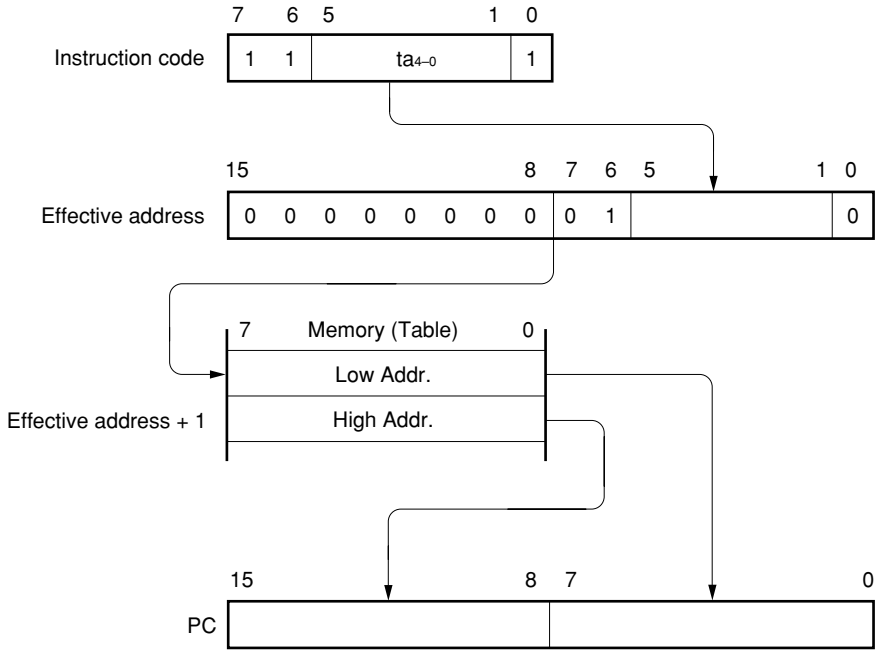
3.3.3 Table indirect addressing

[Function]

Table contents (branch destination address) of the particular location to be addressed by the lower 5-bit 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 branch to all the memory spaces.

[Illustration]



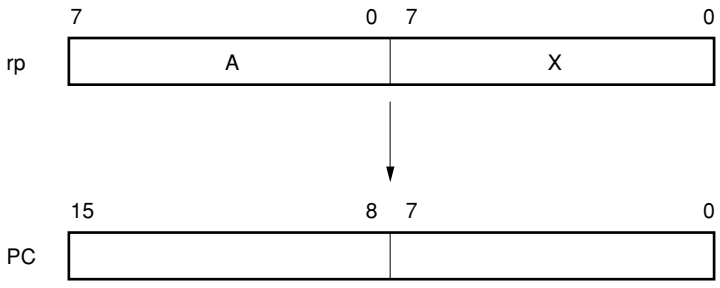
3.3.4 Register addressing

[Function]

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

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

[Illustration]



3.4 Operand Address Addressing

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

3.4.1 Direct addressing

[Function]

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

[Operand format]

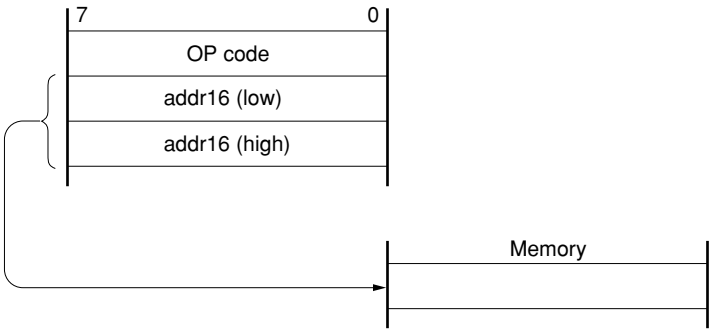
Identifier	Description
addr16	Label or 16-bit immediate data

[Description example]

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

Instruction code	0	0	1	0	1	0	0	1	OP code
	0	0	0	0	0	0	0	0	00H
	1	1	1	1	1	1	1	0	FEH

[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 a special function register (SFR) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.

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

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

[Operand format]

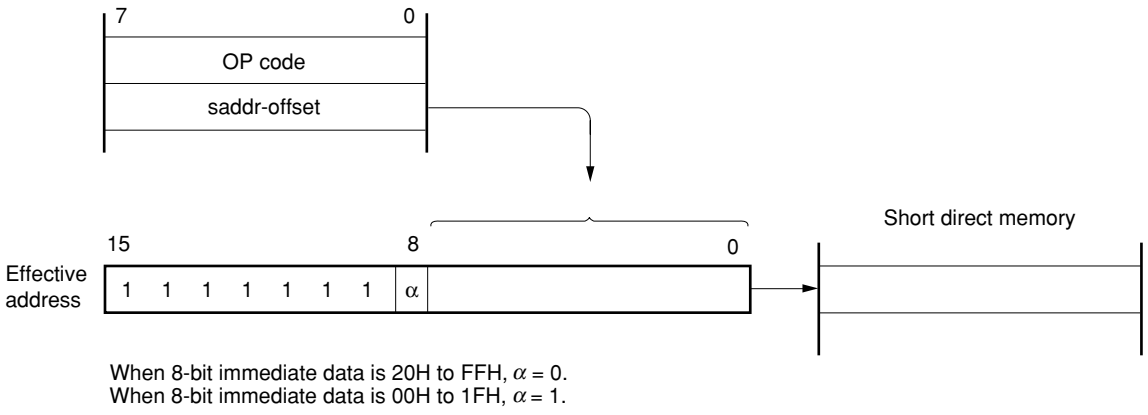
Identifier	Description
saddr	Label or FE20H to FF1FH immediate data
saddrp	Label or FE20H to FF1FH immediate data (even address only)

[Description example]

MOV FE90H, #50H; When setting saddr to FE90H and the immediate data to 50H

Instruction code	1 1 1 1 0 1 0 1	OP code
	1 0 0 1 0 0 0 0	90H (saddr-offset)
	0 1 0 1 0 0 0 0	50H (immediate data)

[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]

Memory-mapped special function registers (SFRs) are 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, SFRs mapped at FF00H to FF1FH can be accessed with short direct addressing.

[Operand format]

Identifier	Description
sfr	Special function register name

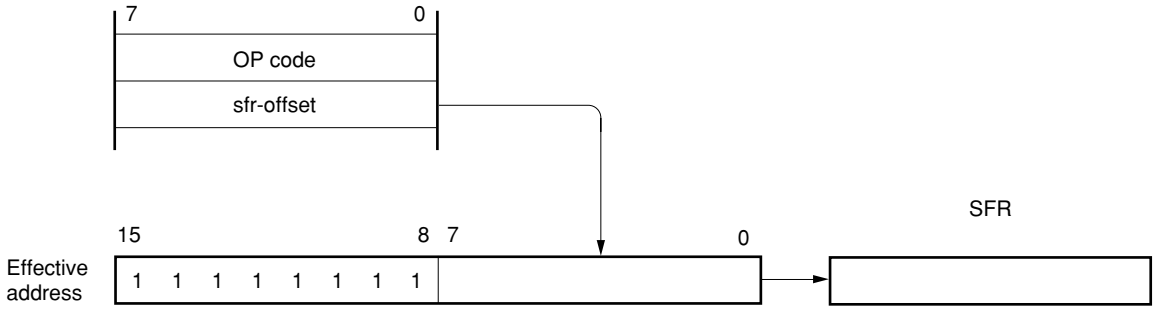
[Description example]

MOV PM0, A; When selecting PM0 for sfr

Instruction code 1 1 1 0 0 1 1 1

0 0 1 0 0 0 0 0

[Illustration]



3.4.4 Register addressing

[Function]

General-purpose registers are accessed as operands. 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 3 bits in the instruction code.

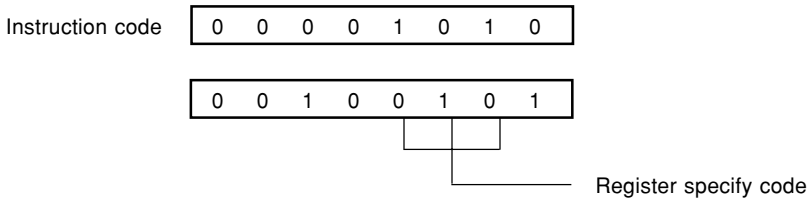
[Operand format]

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

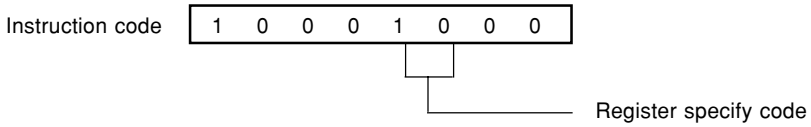
'r' and 'rp' can be described with absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

[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]

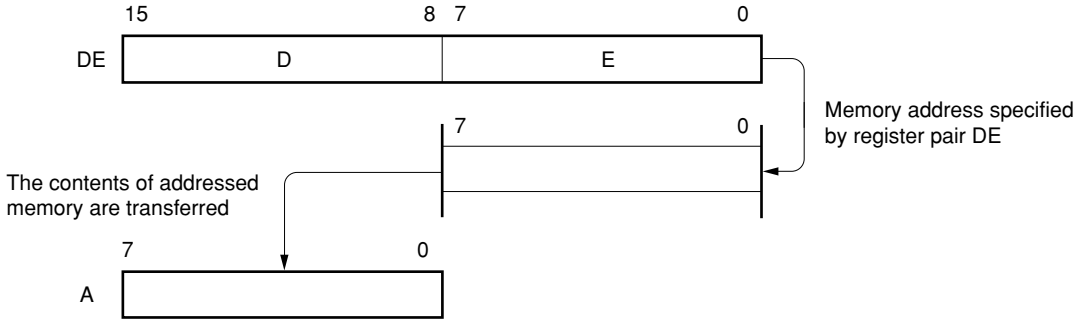
Identifier	Description
—	[DE], [HL]

[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

Instruction code	0	0	1	0	1	1	0	1
	0	0	0	1	0	0	0	0

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/reset upon generation of an interrupt request.

Stack addressing enables to address the internal high-speed RAM area only.

[Description example]

In the case of PUSH DE

Instruction code	1	0	1	0	1	0	1	0

CHAPTER 4 PORT FUNCTIONS

4.1 Functions of Ports

The μPD789871 Subseries provides the ports shown in Figure 4-1, enabling various methods of control.

Numerous other functions are provided that can be used in addition to the digital I/O port function. For more information on these additional functions, refer to **CHAPTER 2 PIN FUNCTIONS**.

Figure 4-1. Port Types

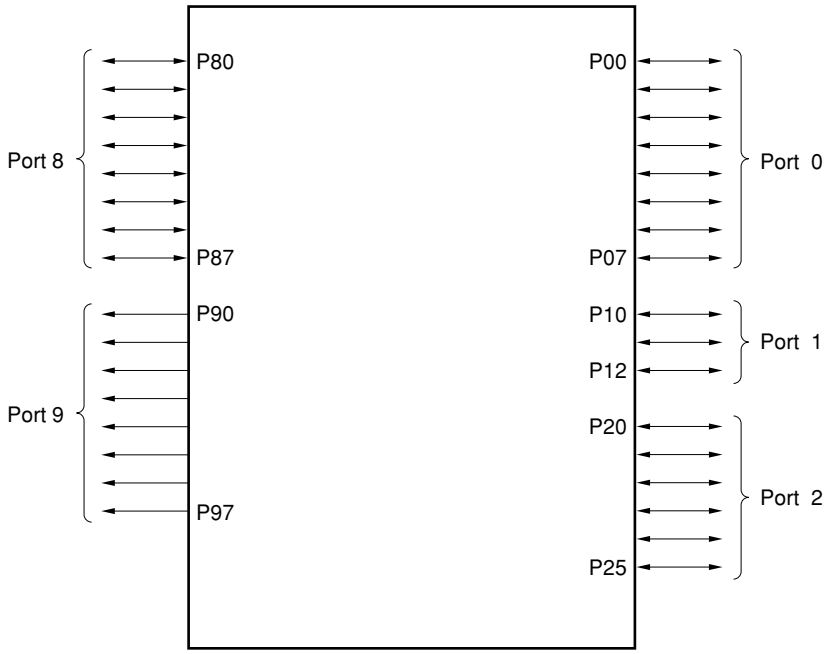


Table 4-1. Port Functions

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, an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).	Input	—
P10 to P12	I/O	Port 1. 3-bit I/O port. Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of 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. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register B2 (PUB2).	Input	$\overline{\text{SCK10}}$
P21				SO10
P22				SI10
P23				INTP0
P24				INTP1
P25				TI
P80 to P87	I/O	Port 8. P-ch open-drain 8-bit high-tolerance I/O port. For mask ROM versions, use of a pull-down resistor for V_{LOAD} can be specified in 1-bit units by a mask option (when used as a general-purpose I/O port, the pull-down resistor is connected to V_{SS0}).	Output	FIP24 to FIP17
P90 to P97	Output	Port 9 P-ch open-drain 8-bit high-tolerance output port. Mask ROM versions include an on-chip pull-down resistor (connected to V_{LOAD}).	Output	FIP16 to FIP9

4.2 Port Configuration

A port consists of the following hardware.

Table 4-2. Configuration of Port

Parameter	Configuration
Control register	Port mode register (PMm: m = 0 to 2) Pull-up resistor option register 0 (PU0) Pull-up option register B2 (PUB2)
Port	Total: 33 CMOS I/O: 17 P-ch open-drain I/O: 8 P-ch open-drain output: 8
Pull-up resistor	17 (software control)
Pull-down resistor	• Mask ROM versions: 16 (8 of these can be specified by the mask option) • Flash memory versions: None

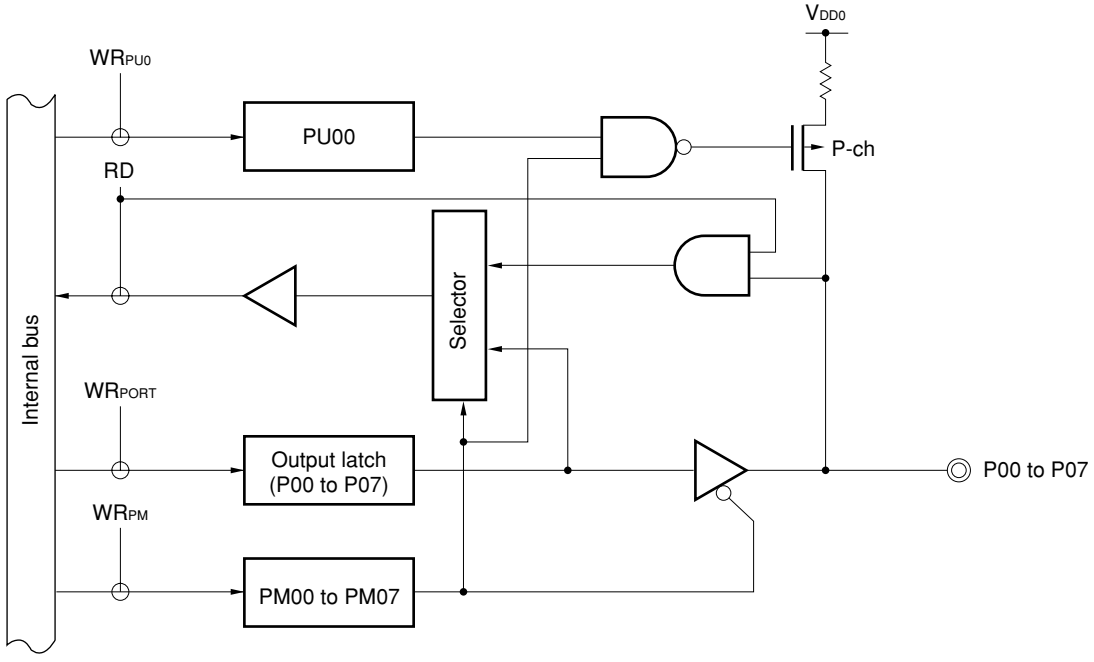
4.2.1 Port 0

This is a 8-bit I/O port with output latches. Port 0 can be specified as input or output mode in 1-bit units by using port mode register 0 (PM0). When pins P00 to P07 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 the 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

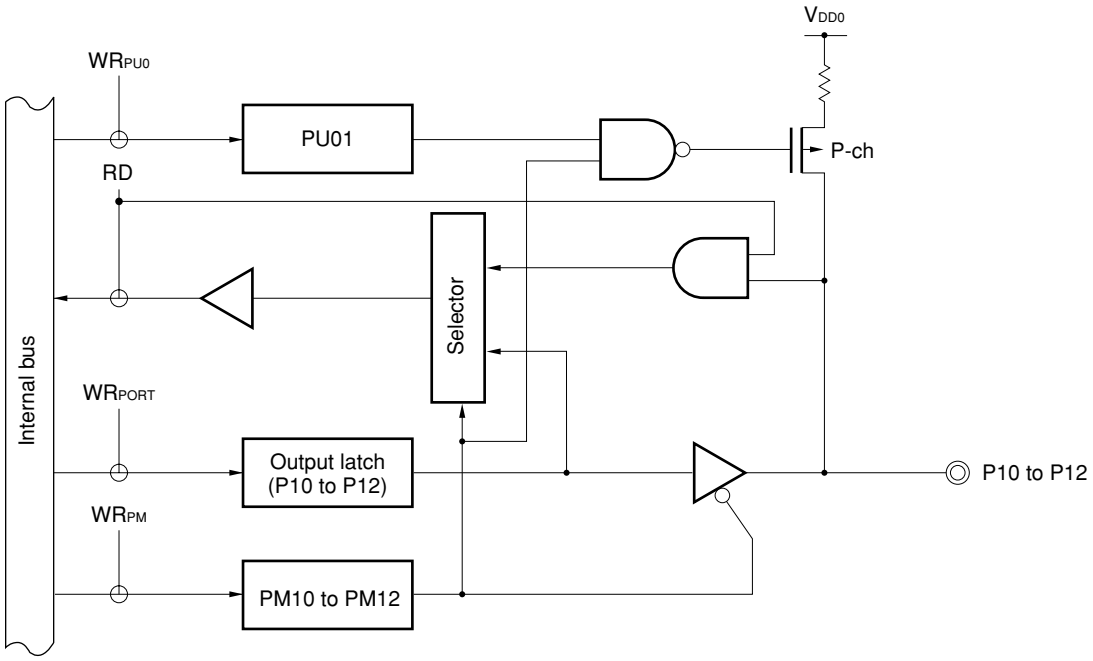
4.2.2 Port 1

This is a 3-bit I/O port with output latches. Port 1 can be specified as input or output mode in 1-bit units by using port mode register 1 (PM1). When pins P10 to P12 are used as input port pins, on-chip pull-up resistors can be connected in 3-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 the block diagram of port 1.

Figure 4-3. Block Diagram of P10 to P12



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 output latches. Port 2 can be specified as input or output mode in 1-bit units by using port mode register 2 (PM2). Use of on-chip pull-up resistors can be specified for pins P20 to P25 in 1-bit units by using pull-up resistor option register B2 (PUB2).

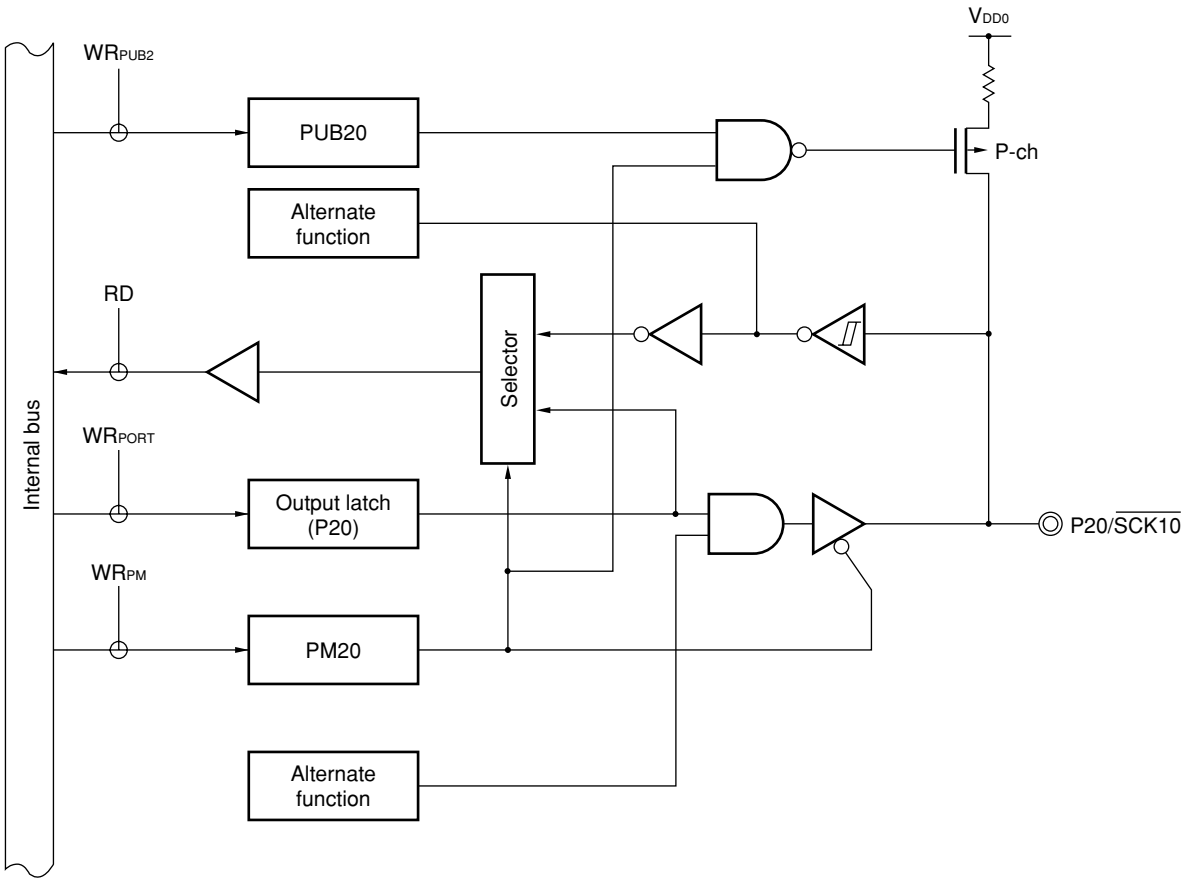
The port is also used as a serial interface I/O, remote control timer input, and external interrupt input.

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

Figures 4-4 to 4-6 show block diagrams of port 2.

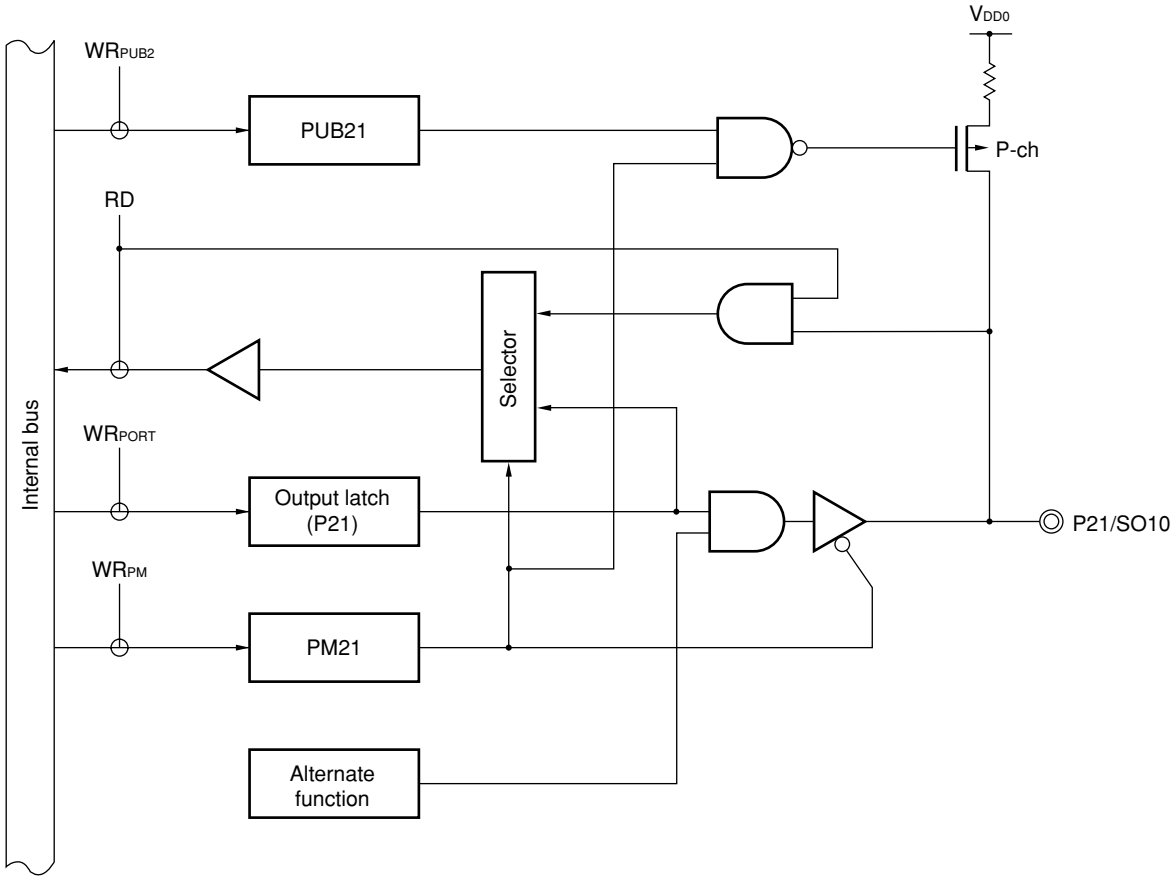
Caution When using the pins of port 2 as the serial interface, the I/O or output latch must be set according to the function to be used. For how to set the latches, see Table 10-2 Serial Interface 10 Operating Mode Settings.

Figure 4-4. Block Diagram of P20



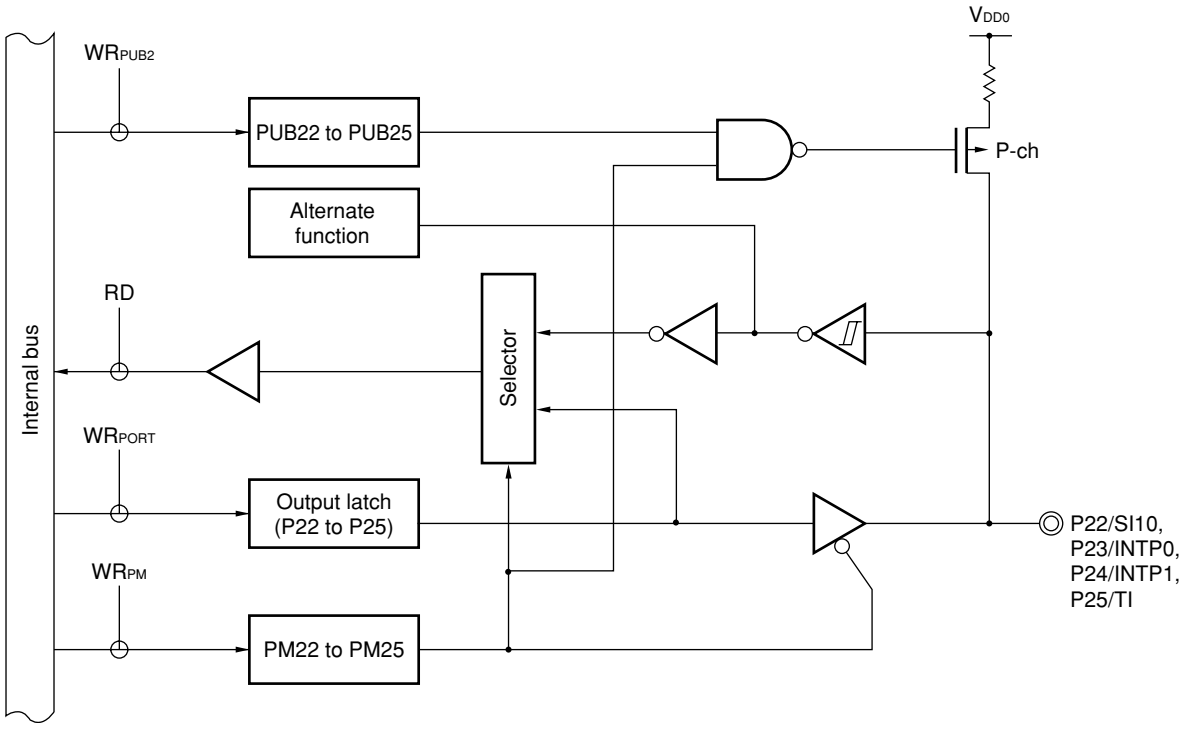
PUB2: Pull-up resistor option register B2
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

Figure 4-5. Block Diagram of P21



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

Figure 4-6. Block Diagram of P22 to P25



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

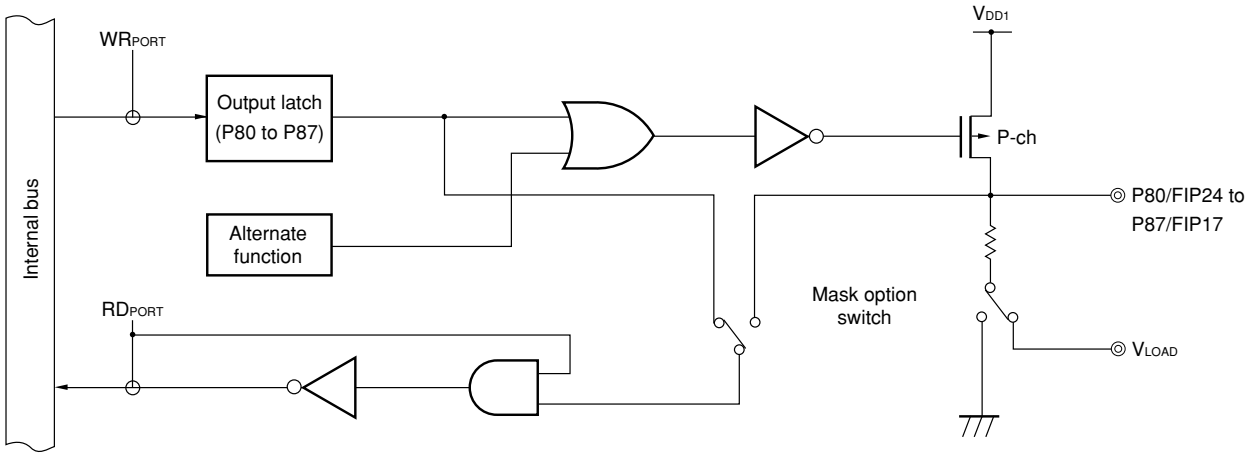
4.2.4 Port 8

This is an 8-bit P-ch open-drain I/O port with output latches. For mask ROM versions, use of a pull-down resistor for V_{LOAD} can be specified by a mask option.

$\overline{RESET}$ input sets port 8 to input mode.

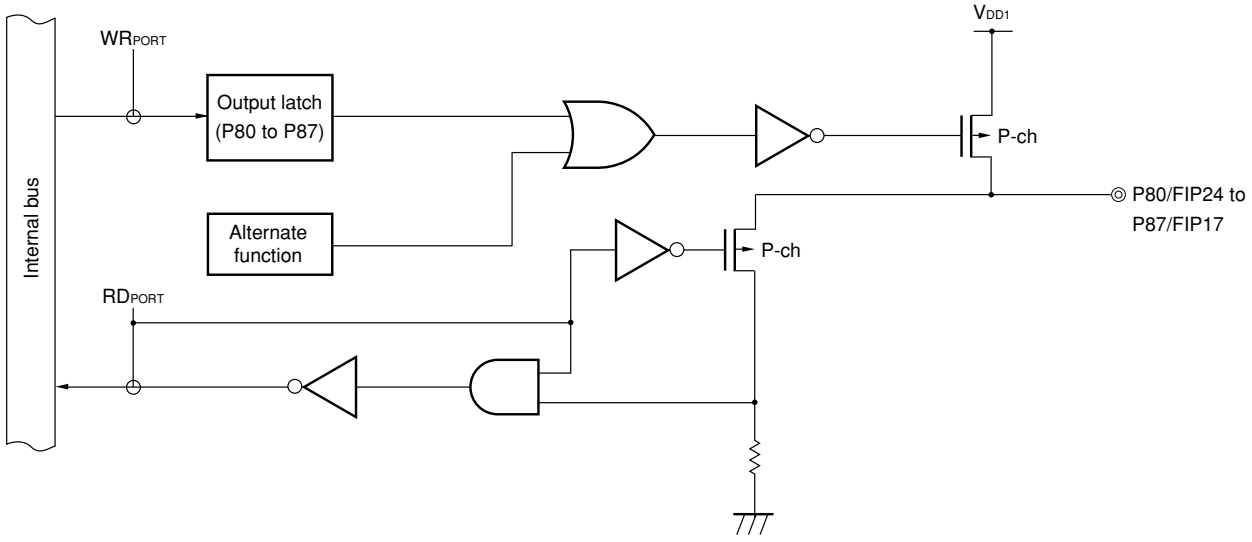
Figures 4-7 and 4-8 show block diagrams of port 8.

Figure 4-7. Block Diagram of P80 to P87 (μPD789870, 789871)



PM: Port mode register
RD: Port 8 read signal
WR: Port 8 write signal

Figure 4-8. Block Diagram of P80 to P87 (μPD78F9872)



PM: Port mode register
RD: Port 8 read signal
WR: Port 8 write signal

4.2.5 Port 9

This is an 8-bit P-ch open-drain output port. Mask ROM versions include an on-chip pull-down resistor (connected to V_{LOAD}).

$\overline{RESET}$ input sets port 9 to output mode.

Figures 4-9 and 4-10 show block diagrams of port 9.

Figure 4-9. Block Diagram of P90 to P97 (μPD789870, 789871)

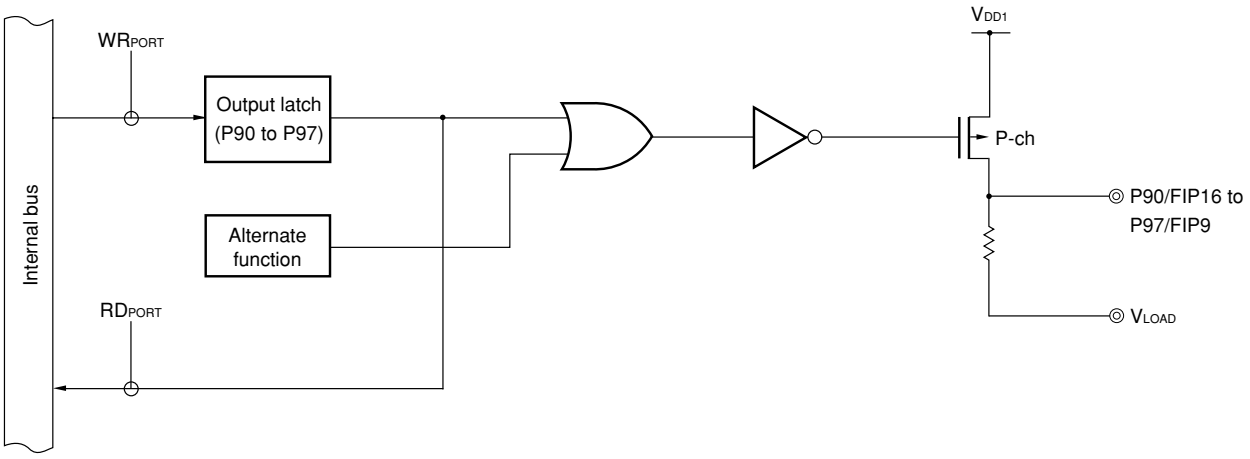
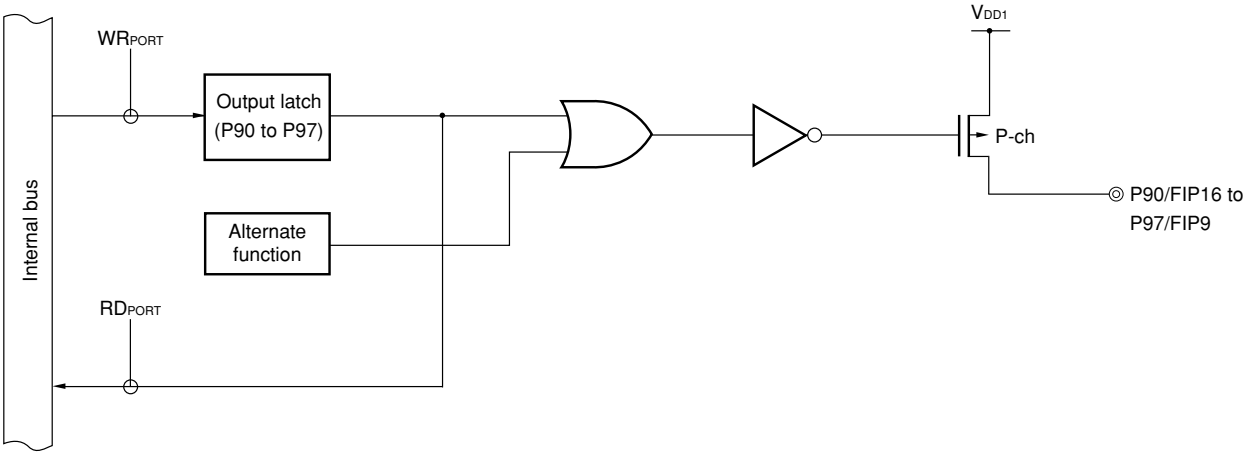


Figure 4-10. Block Diagram of P90 to P97 (μPD78F9872)



4.3 Port Function Control Registers

The following three types of registers 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)

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

Port mode registers are independently set with a 1-bit or 8-bit memory manipulation instruction.

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

When port pins are used as alternate-function pins, set the port mode register and output latch according to Table 4-3.

Caution As port 2 has an alternate function as external interrupt input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. When the output mode is used, therefore, the interrupt mask flag should be set to 1 beforehand.

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

Pin Name	Alternate Function		PM××	P××
	Name	Input/Output		
P23	INTP0	Input	1	×
P24	INTP1	Input	1	×
P25	TI	Input	1	×

Caution When Port 2 is used for serial interface pin, the I/O latch or output latch must be set according to its function. For the setting method, refer to Table 10-2 Serial Interface 10 Operating Mode Settings.

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

Figure 4-11. Port Mode Register Format

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	1	1	1	1	1	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)

Pull-up resistor option register (PU0) sets whether to use an on-chip pull-up resistor at each port or not. At a port where use of on-chip pull-up resistors has been specified by PU0, a pull-up resistor can be internally used only for the bits set in input mode. On-chip pull-up resistors cannot be used for the bits set in output mode, regardless of setting of PU0, or 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 4-12. Pull-Up Resistor Option Register 0 Format

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 used
1	On-chip pull-up resistor used

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

Pull-up register option register B2 (PUB2) sets whether to use an on-chip resistor at each pin of port 2. At a pin for which use of on-chip pull-up resistors has been specified by PUB2, a pull-up resistor can be internally used only for the bits set in input mode. On-chip pull-up resistors cannot be used for the bits set in output mode, regardless of the setting of PUB2, or when the pins are used as alternate-function output pins.

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

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

Figure 4-13. Pull-Up Resistor Option Register B2 Format

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

PUB2n	P2n on-chip pull-up resistor selection (n = 0 to 5)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

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.

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.

The data once written to the output latch is retained until new data is written to 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.

The data once written to the output latch is retained until new data is written to the output latch.

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 input/output port, therefore, the contents of the output latch of the pin that is set in the input mode and not subject to manipulation become undefined.

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.

The data once written to the output latch is retained until new data is written to 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.

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 input/output port, therefore, the contents of the output latch of the pin that is set in the input mode and not subject to manipulation become undefined.

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 two types of system clock oscillators are used.

- **Main system clock (ceramic/crystal) oscillator**
  This circuit oscillates at 1.0 to 5.0 MHz. Oscillation can be stopped by executing the STOP instruction or setting the processor clock control register (PCC).
- **Subsystem clock oscillator**
  This circuit oscillates at 32.768 kHz. Oscillation can be stopped by the suboscillation mode register (SCKM).

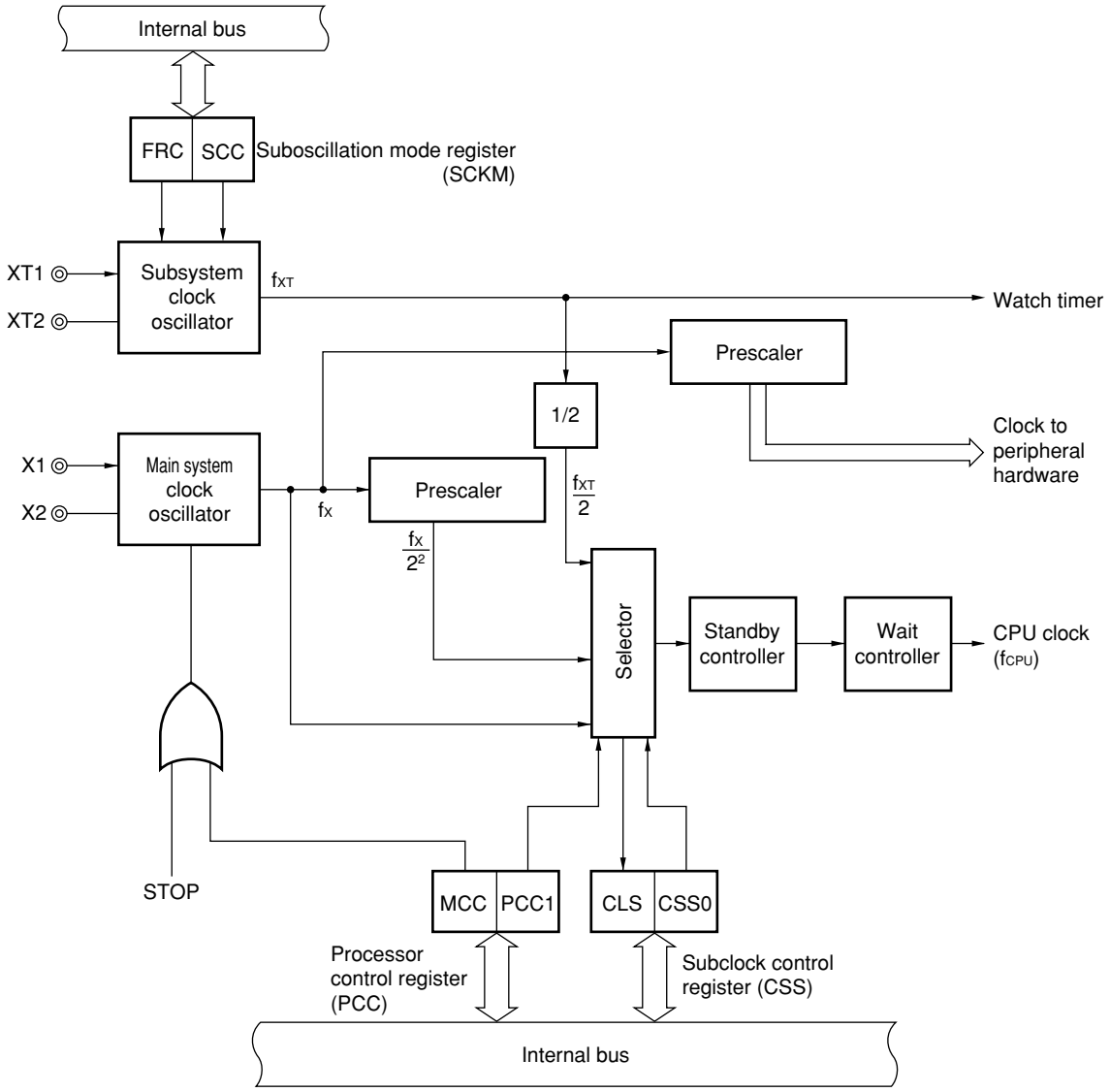
5.2 Clock Generator Configuration

The clock generator includes the following hardware.

Table 5-1. Configuration of Clock Generator

Item	Configuration
Control registers	Processor clock control register (PCC) Suboscillation mode register (SCKM) Subclock control register (CSS)
Oscillators	Main system clock oscillator Subsystem clock oscillator

Figure 5-1. Block Diagram of Clock Generator



5.3 Register Controlling Clock Generator

The clock generator is controlled by the following registers.

- Processor clock control register (PCC)
- Suboscillation mode register (SCKM)
- Subclock control register (CSS)

(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 the PCC to 02H.

Figure 5-2. Processor Clock Control Register Format

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

MCC	Control of main system clock oscillator operation
0	Operation enabled
1	Operation stopped

CSS0	PCC1	CPU clock (f_{CPU}) selection[Note]
0	0	f_X (0.2 μs)
0	1	$f_X/2^2$ (0.8 μs)
1	0	$f_{XT}/2$ (61 μs)
1	1	

Note The CPU clock is selected according to a combination of the PCC1 flag in the processor clock control register (PCC and the CSS0 flag in the subclock control register (CSS) (refer to **5.3 (3) subclock control register (CSS)**).

Cautions 1. Bit 0 and 2 to 6 must be set to 0.

2. The MCC can be set only when the subsystem clock has been selected as the CPU clock.

Remarks 1. f_X: Main system clock oscillation frequency

2. f_{XT}: Subsystem clock oscillation frequency

3. The parenthesized values apply to operation at f_X = 5.0 MHz or f_{XT} = 32.768 kHz.

4. Minimum instruction execution time: $2f_{CPU}$

- f_{CPU} = 0.2 μs: 0.4 μs
- f_{CPU} = 0.8 μs: 1.6 μs
- f_{CPU} = 61 μs: 122 μs

(2) Suboscillation mode register (SCKM)

SCKM selects a the feedback resistor for the subsystem clock, and controls the oscillation of the clock.

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

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

Figure 5-3. Suboscillation Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
SCKM	0	0	0	0	0	0	FRC	SCC	FFF0H	00H	R/W

FRC	Feedback resistor selection
0	On-chip feedback resistor used
1	On-chip feedback resistor not used

SCC	Control of subsystem clock oscillator operation
0	Operation enabled
1	Operation stopped

Caution Bit 2 to 7 must be set to 0.

(3) Subclock control register (CSS)

CSS specifies whether the main system or subsystem clock oscillator is to be selected. It also specifies the CPU clock operation status.

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

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

Figure 5-4. Subclock Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
CSS	0	0	CLS	CSS0	0	0	0	0	FFF2H	00H	R/W[Note]

CLS	CPU clock operation status
0	Operation based on the divided main system clock output
1	Operation based on the subsystem clock output

CSS0	Selection of the main system or subsystem clock oscillator
0	Divided output from the main system clock oscillator
1	Output from the subsystem clock oscillator

Note Bit 5 read only.

Caution Bits 0 to 3, 6, and 7 must be set to 0.

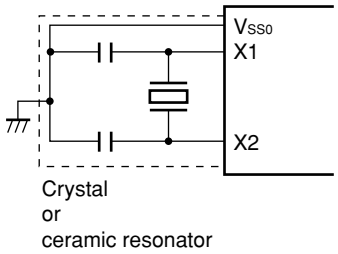
5.4 System Clock Oscillators

5.4.1 Main system clock oscillator

The main system clock oscillator is oscillated by the crystal or ceramic resonator (5.0 MHz TYP.) connected across the X1 and X2 pins.

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

Figure 5-5. External Circuit of Main System Clock Oscillator

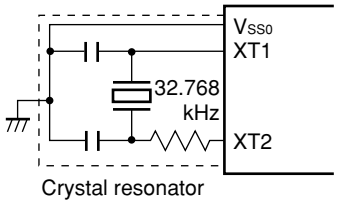


5.4.2 Subsystem clock oscillator

The subsystem clock oscillator by the crystal resonator (32.768 kHz TYP.) connected across the XT1 and XT2 pins.

Figure 5-6 shows the external circuit of the subsystem clock oscillator.

Figure 5-6. External Circuit of Subsystem Clock Oscillator



Caution When using the main system or subsystem clock oscillator, wire as follows in the area enclosed by the broken lines in Figures 5-5 and 5-6 to avoid an adverse effect from wiring capacitance.

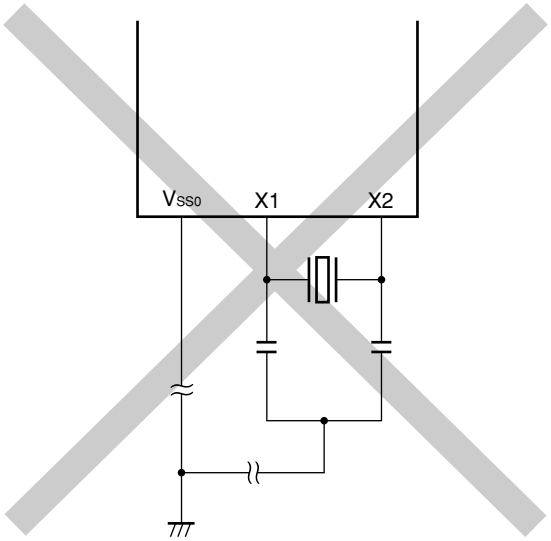
- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS0}. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

When using the subsystem clock, particular care is required because the subsystem clock oscillator is designed as a low-amplitude circuit for reducing current consumption.

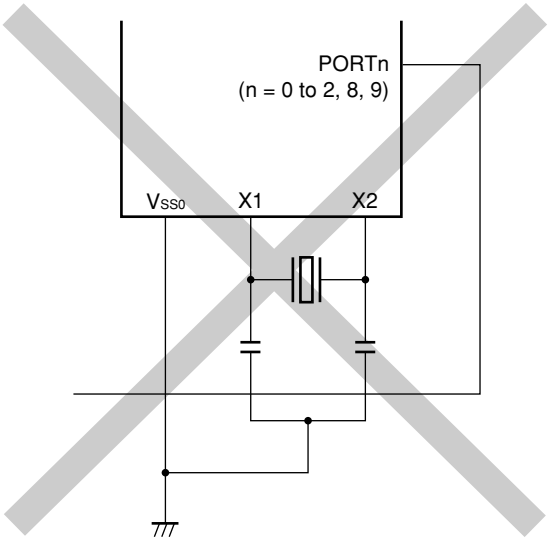
Figure 5-7 shows examples of incorrect resonator connection.

Figure 5-7. Examples of Incorrect Resonator Connection (1/2)

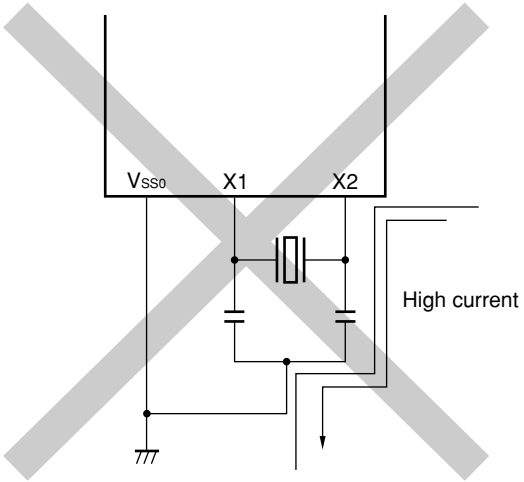
(a) Tool long wiring



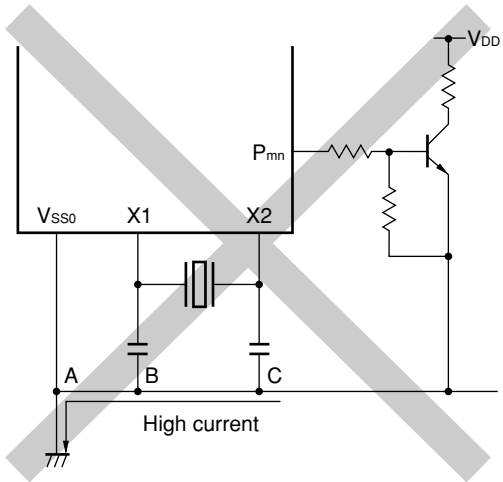
(b) Crossed signal line



(c) Wiring near high fluctuating current



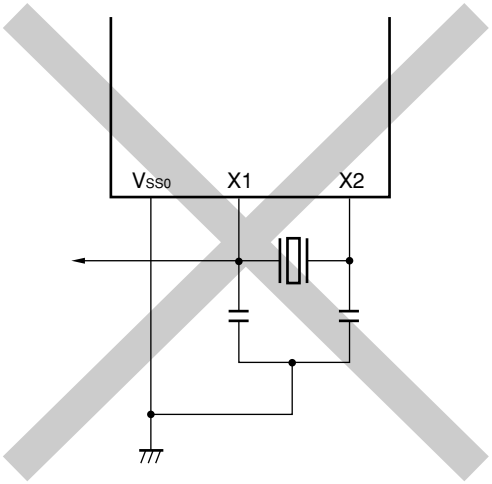
(d) Current flowing through ground line of oscillator (potential at points A, B, and C fluctuates)



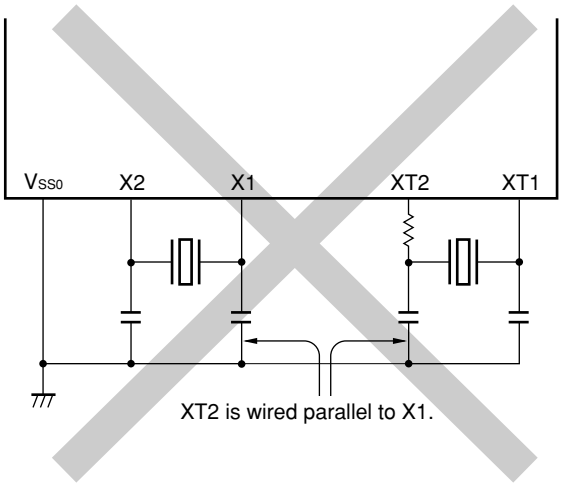
Remark When using the subsystem clock, read X1 and X2 as XT1 and XT2, respectively, and connect a resistor to the XT2 in series.

Figure 5-7. Examples of Incorrect Resonator Connection (2/2)

(e) Signal is fetched



(f) Parallel and near signal lines of main system clock and subsystem clock



Remark When using the subsystem clock, read X1 and X2 as XT1 and XT2, respectively, and connect a resistor to the XT2 in series.

Caution If the X1 wire is in parallel with the XT2 wire, crosstalk noise may occur between the X1 and XT2, resulting in a malfunction.
To avoid this, do not lay the X1 and XT2 wires in parallel.

5.4.3 Frequency divider

The frequency divider divides the output of the main system clock oscillator (fx) to generate various clocks.

5.4.4 When no subsystem clock is used

If a subsystem clock is not necessary, for example, for low-power consumption operation or clock operation, handle the XT1 and XT2 pins as follows:

XT1: Connect to V_{SS0}
XT2: Leave open

In this case, however, a small leakage current flows via the on-chip feedback resistor in the subsystem clock oscillator when the main system clock is stopped. To avoid this, set bit 1 (FRC) of the suboscillation mode register (SCKM) so that the on-chip feedback resistor will not be used. Also in this case, handle the XT1 and XT2 pins as stated above.

5.5 Clock Generator Operation

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

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

The operation and function of the clock generator is determined by the processor clock control register (PCC), suboscillation mode register (SCKM), and subclock control register (CSS), as follows.

(a) The low-speed mode $2f_{CPU}$ (1.6 μs : at 5.0 MHz operation) of the main system clock is selected when the $\overline{\text{RESET}}$ signal is generated (PCC = 02H). While a low level is input to the $\overline{\text{RESET}}$ pin, oscillation of the main system clock is stopped.

(b) Three types of CPU clocks f_{CPU} (0.2 μs and 0.8 μs: main system clock (at 5.0 MHz operation), 61 μs: subsystem clock (at 32.768 kHz operation)) can be selected by the PCC, SCKM, and CSS settings.

(c) Two standby modes, STOP and HALT, can be used with the main system clock selected. In a system where no subsystem clock is used, setting SCKM bit 1 (FRC) so that the on-chip feedback resistor cannot be used reduces current consumption in STOP mode. In a system where a subsystem clock is used, setting SCKM bit 0 to 1 can cause the subsystem clock to stop oscillation.

(d) CSS bit 4 (CSS0) can be used to select the subsystem clock so that low current consumption operation is used (122 μs : at 32.768 kHz operation).

(e) With the subsystem clock selected, it is possible to cause the main system clock to stop oscillating using bit 7 (MCC) of PCC. The HALT mode can be used, but the STOP mode cannot.

(f) The clock pulse for the peripheral hardware is generated by dividing the frequency of the main system clock, but the subsystem clock pulse is only supplied to the watch timer. The watch timer can therefore keep running even during standby. The other hardware stops when the main system clock stops because if runs based on the main system clock (except for the external input clock operations).

5.6 Changing Setting of System Clock and CPU Clock

5.6.1 Time required for switching between system clock and CPU clock

The CPU clock can be selected by using bit 1 (PCC1) of the processor clock control register (PCC) and bit 4 (CSS0) of the subclock control register (CSS).

Actually, the specified clock is not selected immediately after the setting of PCC has been changed, and 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					
CSS0	PCC1	CSS0	PCC1	CSS0	PCC1	CSS0	PCC1
		0	0	0	1	1	×
0	0			4 clocks		2fx/fxT clocks (306 clocks)	
	1	2 clocks				fx/2fxT clocks (76 clocks)	
1	×	2 clocks		2 clocks			

Remarks 1. Two clocks are the minimum instruction execution time of the CPU clock before switching.

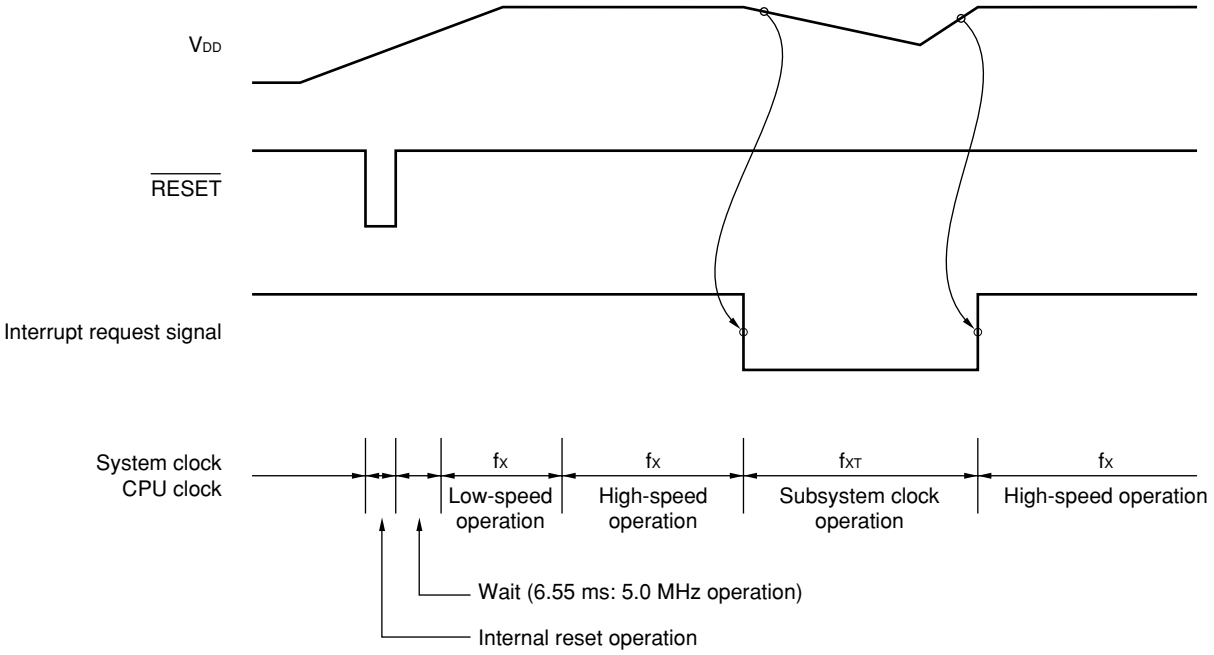
2. The parenthesized values apply to operation at fx = 5.0 MHz or fxT = 32.768 kHz.

3. ×: don't care

5.6.2 Switching between system clock and CPU clock

The following figure illustrates how the CPU clock and system clock switch.

Figure 5-8. 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 reset state is released when the $\overline{\text{RESET}}$ pin is made high, and the main system clock starts oscillating. At this time, the oscillation stabilization time ($2^{15}/f_X$) is automatically secured.
After that, the CPU starts instruction execution at the low speed of the main system clock (1.6 μs: at 5.0 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, bit 1 (PCC1) of the processor clock control register (PCC) and bit 4 (CSS0) of the subclock control register (CSS) are rewritten so that high-speed operation can be selected.

<3> A drop of the V_{DD} voltage is detected with an interrupt request signal. The clock is switched to the subsystem clock (at this moment, the subsystem clock must be in the oscillation stabilization state).

<4> A recover of the V_{DD} voltage is detected with an interrupt request signal. Bit 7 (MCC) of PCC is set to 0, and then the main system clock starts oscillating. After the time required for the oscillation to stabilize has elapsed, PCC1 and CSS0 are rewritten so that high-speed operation can be selected again.

Caution When the main system clock is stopped and the device is operating on the subsystem clock, wait until the oscillation stabilization time has been secured by the program before switching back to the main system clock.

CHAPTER 6 8-BIT REMOTE CONTROL TIMER

6.1 8-Bit Remote Control Timer Functions

The 8-bit remote control timer has a pulse width measurement function with a resolution of 8 bits.

Pulse width is measured from a difference in count value when the valid edge has been detected while the timer operates in the free-running mode.

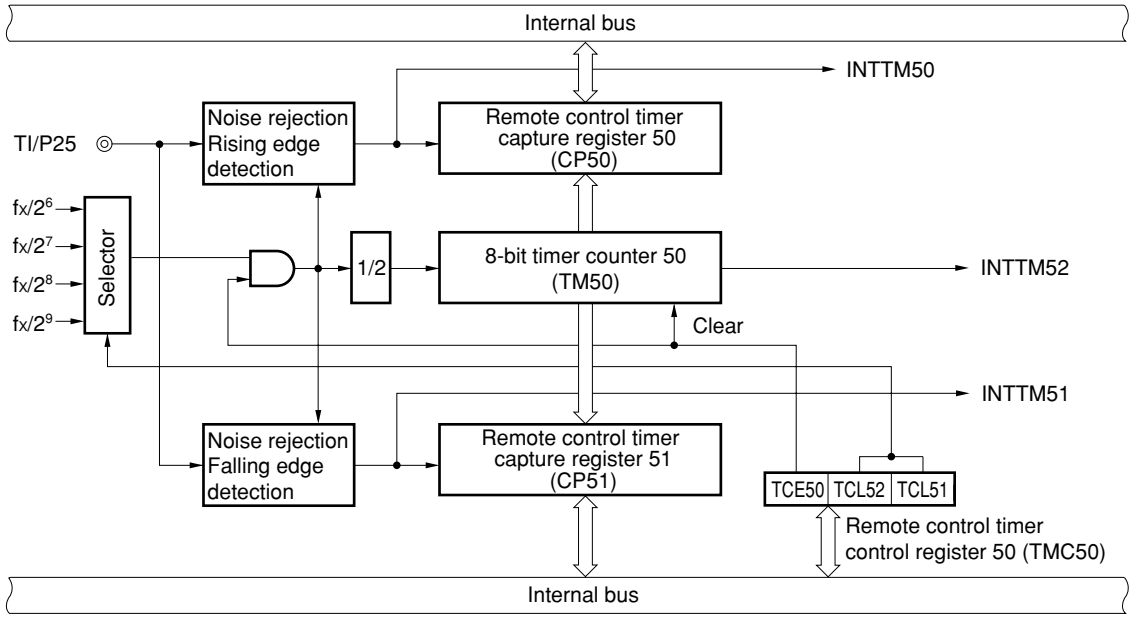
6.2 8-Bit Remote Control Timer Configuration

The 8-bit remote control timer consists of the following hardware.

Table 6-1. Configuration of 8-Bit Remote Control Timer

Item	Configuration
Timer counter	8-bit timer (TM50)
Register	Remote control timer capture register: × 2 (CP50 and CP51)
Control register	Remote control timer control register 50 (TMC50)

Figure 6-1. Block Diagram of 8-Bit Remote Control Timer



(1) Remote control timer capture registers (CP50 and CP51)

These 8-bit registers capture the contents of the 8-bit timer counter 50 (TM50).

The capture operation is performed in synchronization with the valid edge input to the TI pin (capture trigger).

The contents of CP50 are retained until the next rising edge of the TI pin is detected. The contents of CP51 are retained until the next falling edge of the TI pin is detected.

CP50 and CP51 can be read by using an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears CP50 and CP51 to 00H.

(2) 8-bit timer counter 50 (TM50)

This 8-bit register counts the count pulse.

$\overline{\text{RESET}}$ input or clearing the TCE50 bit clears TM50 to 00H.

6.3 Registers Controlling 8-Bit Remote Control Timer

The following register controls the 8-bit remote control timer.

(1) Remote control timer control register 50 (TMC50)

This register enables or disables the operation of the 8-bit timer counter 50 (TM50), and sets the count clock.

TMC50 is set by using a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TMC50 to 00H.

Figure 6-2. Remote Control Timer Control Register 50 Format

Symbol	< 7 >	6	5	4	3	2	1	0	Address	After reset	R/W
TMC50	TCE50	0	0	0	0	0	TCL52	TCL51	FF58H	00H	R/W

TCE50	TM50 count operation control
0	Clears counter to 0 and stops operation
1	Starts count operation

TCL52	TCL51	Count clock selection
0	0	$f_X/2^9$ (9.8 kHz)
0	1	$f_X/2^8$ (19.5 kHz)
1	0	$f_X/2^7$ (39.1 kHz)
1	1	$f_X/2^6$ (78.1 kHz)

Caution Be sure to clear bits 2 to 6 to 0.

Remarks 1. f_X: Main system clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 5.0 MHz

6.4 Operation of 8-Bit Remote Control Timer

The 8-bit remote control timer operates as a pulse width measuring circuit.

The width of a high-level or low-level external pulse input to the TI pin is measured by operating the 8-bit timer counter 50 (TM50) in the free-running mode.

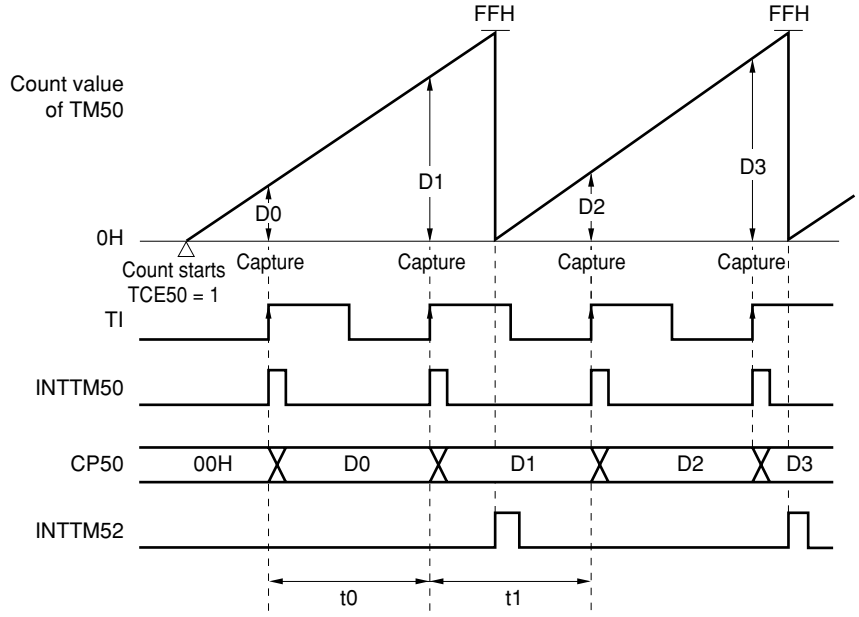
Noise with short pulse width can be detected since the detection of the valid edge is sampled every 2 cycles of the count clock selected by TCL51 and TCL52, and the capture operation is not performed until the valid level has been detected two times. Therefore, the pulse width input to the TI pin must be 5 or more of the count clock set by TCL51 and TCL52, regardless of whether the level is high or low. If the pulse width is less than 5 clocks, it cannot be detected, and the capture operation is not performed.

The value of timer register 50 (TM50) being counted is loaded to and retained in the capture registers (CP50 and CP51) in synchronization with the valid edge of the pulse input to the TI pin, as shown in Figure 6-3.

Figure 6-3 shows the timing of pulse width measurement.

Figure 6-3. Pulse Width Measurement Timing (1/2)

(1) To measure pulse width in synchronization with rising edge



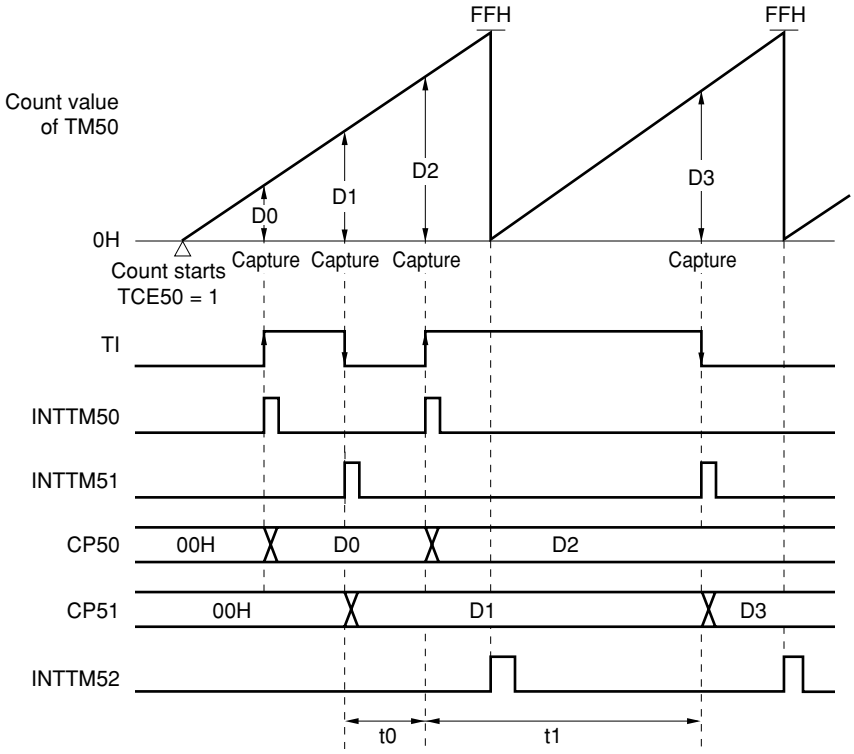
Remark t0 = (D1 − D0) × 1/f_{COUNT}

t1 = (100H − D1 + D2) × 1/f_{COUNT}

f_{COUNT}: Count clock frequency set by TCL51 and TCL52

Figure 6-3. Pulse Width Measurement Timing (2/2)

(2) Measure pulse width in synchronization with both rising and falling edges



Remark t0 = (D2 – D1) × 1/f_{COUNT}

t1 = (100H – D2 + D3) × 1/f_{COUNT}

f_{COUNT}: Count clock frequency set by TCL51 and TCL52

CHAPTER 7 8-BIT TIMER

7.1 8-Bit Timer Functions

The 8-bit timers (TM80 and TM81) enable to use the interval function.

(1) 8-bit interval timer

(1) 8-bit interval timer

When the 8-bit timer is used as an interval timer, it generates an interrupt at any time intervals set in advance.

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

Minimum Interval Time	Maximum Interval Time	Resolution
$2^2/f_X$ (0.8 μs)	$2^{10}/f_X$ (204.8 μs)	$2^2/f_X$ (0.8 μs)
$2^4/f_X$ (3.2 μs)	$2^{12}/f_X$ (819.2 μs)	$2^4/f_X$ (3.2 μs)
$2^6/f_X$ (12.8 μs)	$2^4/f_X$ (3.28 ms)	$2^6/f_X$ (12.8 μs)

Remarks 1. f_X: Main system clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 5.0 MHz.

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

Minimum Interval Time	Maximum Interval Time	Resolution
$2/f_X$ (0.4 μs)	$2^8/f_X$ (51.2 μs)	$2/f_X$ (0.4 μs)
$2^5/f_X$ (6.4 μs)	$2^{13}/f_X$ (1.64 ms)	$2^5/f_X$ (6.4 μs)
$2^7/f_X$ (25.6 μs)	$2^{15}/f_X$ (6.55 ms)	$2^7/f_X$ (25.6 μs)

Remarks 1. f_X: Main system clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 5.0 MHz.

7.2 8-Bit Timer Configuration

The 8-bit timer consists of the following hardware.

Table 7-3. Configuration of 8-Bit Timer

Item	Configuration
Timer counter	8 bits × 2 (TM80 and TM81)
Register	Compare register: 8 bits × 2 (CR80 and CR81)
Control register	8-bit timer mode control registers 80 and 81 (TMC80 and TMC81)

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

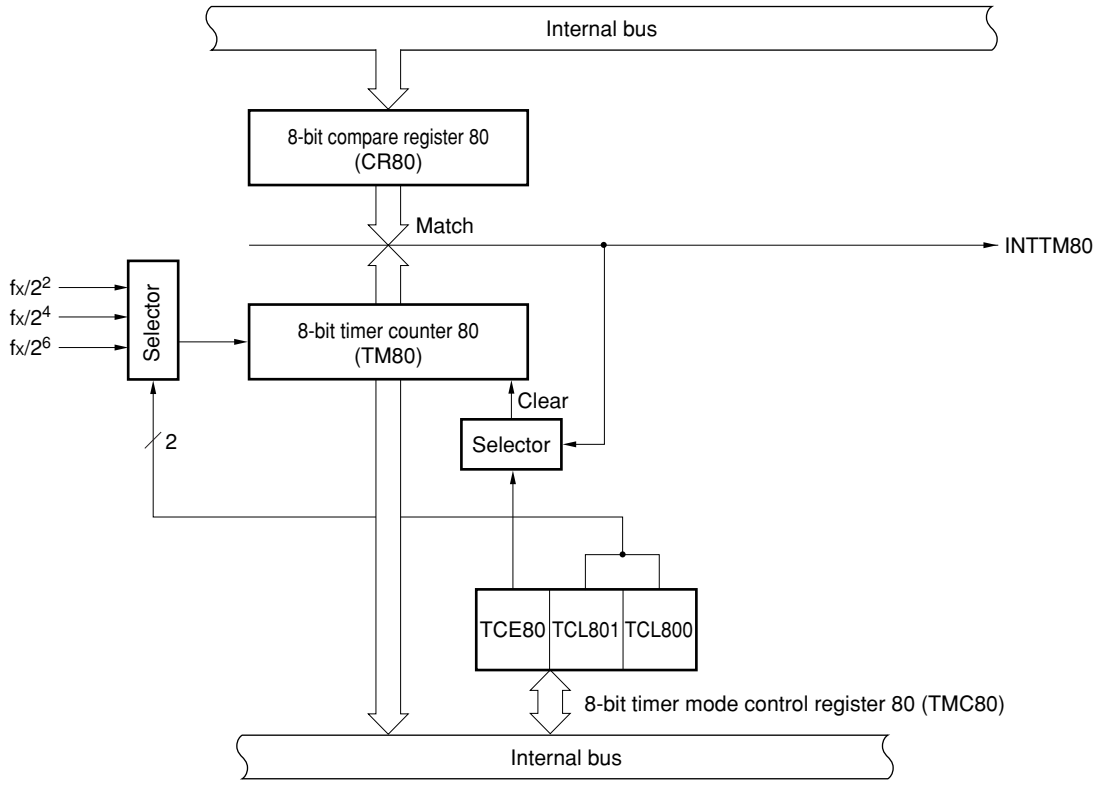
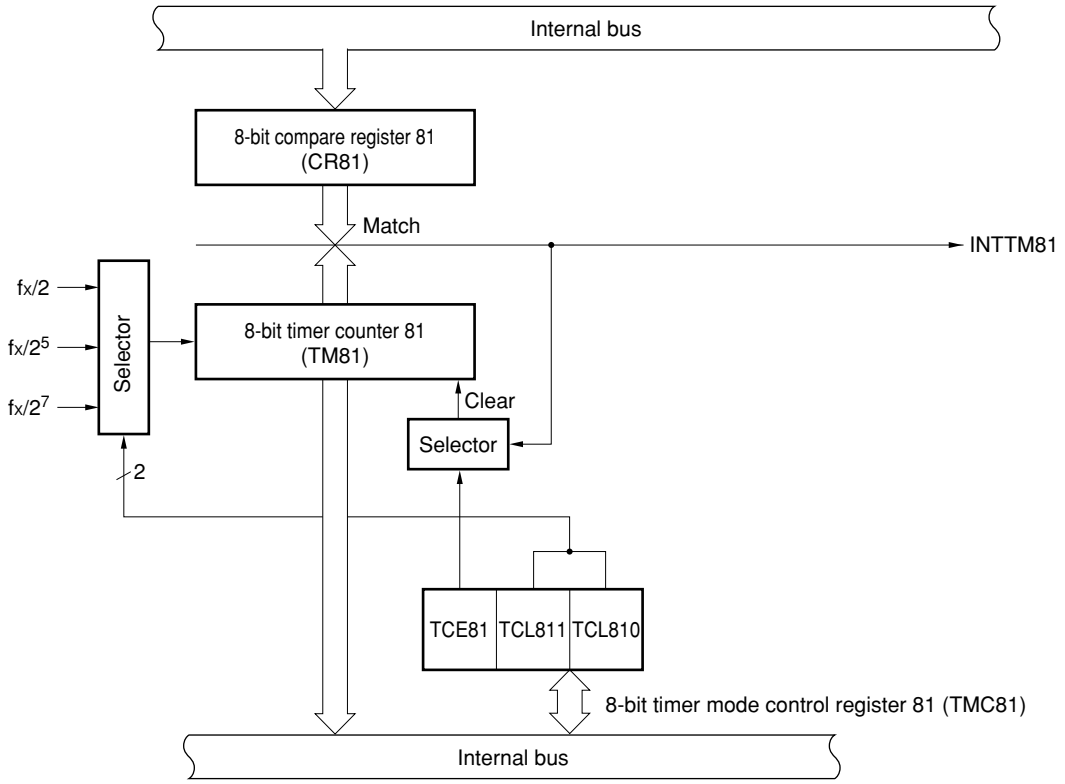


Figure 7-2. Block Diagram of 8-Bit Timer 81



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

This is an 8-bit register to compare the value set to CR8n with 8-bit timer counter 8n (TM8n) count value, and if they match, generates an interrupt request (INTTM8n).

CR8n is set with an 8-bit memory manipulation instruction. The 00H to FFH values can be set.

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

Caution Be sure to set CR8n after the timer operation is stopped.

Remark n = 0 or 1

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

This is an 8-bit register to count pulses.

TM8n is read with an 8-bit memory manipulation instruction.

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

Remark n = 0 or 1

7.3 Registers Controlling 8-Bit Timer

The following register is used to control the 8-bit timer.

- 8-bit timer mode control registers 80 and 81 (TMC80 and TMC81)

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

This register enables/stops operation of 8-bit timer counter 80 (TM80) and sets the counter clock of 8-bit timer 80.

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

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

Figure 7-3. 8-Bit Timer Mode Control Register 80 Format

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
TMC80	TCE80	0	0	0	0	TCL801	TCL800	0	FF53H	00H	R/W

TCE80	8-bit timer counter 80 operation control
0	Operation stopped (TM80 cleared to 0)
1	Operation enabled

TCL801	TCL800	8-bit timer 80 count clock selection
0	0	$f_X/2^2$ (1.25 MHz)
0	1	$f_X/2^4$ (312.5 kHz)
1	0	$f_X/2^6$ (78.1 kHz)
1	1	Setting prohibited

Caution Be sure to set the count clock after the timer operation is stopped (TCE80 = 0). Refer to 7.4 8-Bit Timer Operation for details.

Remarks 1. f_X: Main system clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 5.0 MHz.

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

This register enables/stops operation of 8-bit timer counter 81 (TM81), sets the count clock of 8-bit timer 81, and controls the operation of the output control circuit.

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

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

Figure 7-4. 8-Bit Timer Mode Control Register 81 Format

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
TMC81	TCE81	0	0	0	0	TCL811	TCL810	0	FF57H	00H	R/W

TCE81	8-bit timer counter 81 operation control
0	Operation stopped (TM81 cleared to 0)
1	Operation enabled

TCL811	TCL810	8-bit timer 81 count clock selection
0	0	$f_X/2$ (2.5 MHz)
0	1	$f_X/2^5$ (156.3 kHz)
1	0	$f_X/2^7$ (39.1 kHz)
1	1	Setting prohibited

Caution Be sure to set the count clock after the timer operation is stopped (TCE81 = 0). Refer to 7.4 8-Bit Timer Operation for details.

Remarks 1. f_X: Main system clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 5.0 MHz.

7.4 8-Bit Timer Operation

7.4.1 Operation as interval timer

Interval timer repeatedly generates an interrupt at time intervals specified by the count value set to 8-bit compare register 8n (CR8n) in advance.

To operate 8-bit timer counter as an interval timer, the following settings are required.

<1> Disable operation of the 8-bit timer counter 8n (TM8n) by setting TCE8n (bit 7 of the 8-bit timer mode control register 8n (TMC8n)) to 0.
<2> Set the count clock of the 8-bit timer 8n (see **Tables 7-5** and **7-6**).
<3> Set count values to CR8n.
<4> Enable operation of TM8n by setting TCE0n to 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 0 and TM8n continues counting. At the same time, an interrupt request signal (INTTM8n) is generated.

Tables 7-5 and 7-6 show interval time, and Figures 7-6 and 7-7 show the interval timer operation timing.

Caution When the TMC8n count clock is set and the operation of TM8n is enabled simultaneously by an 8-bit memory manipulation instruction, an error of more than 1 clock may occur in 1 cycle after the timer has been started. Therefore, be sure to follow the settings above when the 8-bit timer is operating as an internal timer.

Remark n = 0 or 1

Table 7-4. Interval Time of 8-Bit Timer 80

TCL801	TCL800	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	2^2/fx (0.8 μs)	2^{10}/fx (204.8 μs)	2^2/fx (0.8 μs)
0	1	2^4/fx (3.2 μs)	2^{12}/fx (819.2 μs)	2^4/fx (3.2 μs)
1	0	2^6/fx (12.8 μs)	2^{14}/fx (3.28 ms)	2^6/fx (12.8 μs)
1	1	Setting prohibited		

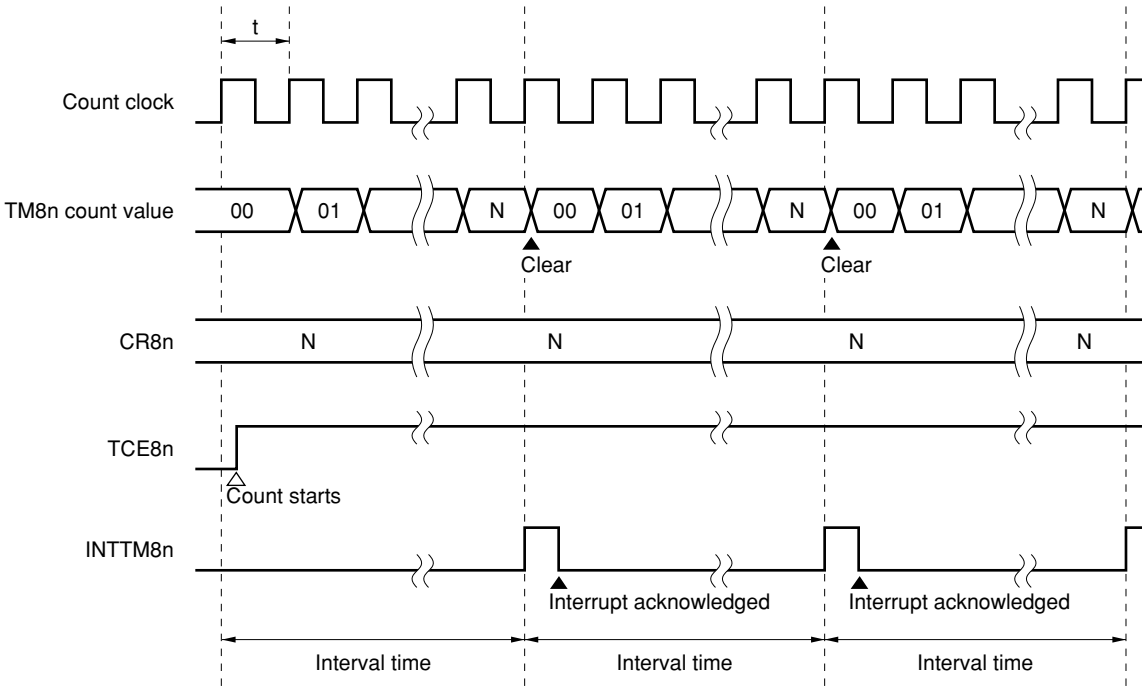
Remarks 1. fx: Main system clock oscillation frequency
2. The parenthesized values apply to operation at fx = 5.0 MHz.

Table 7-5. Interval Time of 8-Bit Timer 81

TCL811	TCL810	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	2/fx (0.4 μs)	2^8/fx (51.2 μs)	2/fx (0.4 μs)
0	1	2^5/fx (6.4 μs)	2^{13}/fx (1.64 ms)	2^5/fx (6.4 μs)
1	0	2^7/fx (25.6 μs)	2^{15}/fx (6.55 ms)	2^7/fx (25.6 μs)
1	1	Setting prohibited		

Remarks 1. fx: Main system clock oscillation frequency
2. The parenthesized values apply to operation at fx = 5.0 MHz.

Figure 7-5. Interval Timer Operation Timing



Caution An error of up to 1 clock may occur in the period until the match signal is generated after the timer starts.
This is because 8-bit timer counter 8n (TM8n) starts operation asynchronously with the count pulse.

Remarks 1. Interval time = (N + 1) × t where N = 00H to FFH
2. n = 0 or 1

CHAPTER 8 WATCH TIMER

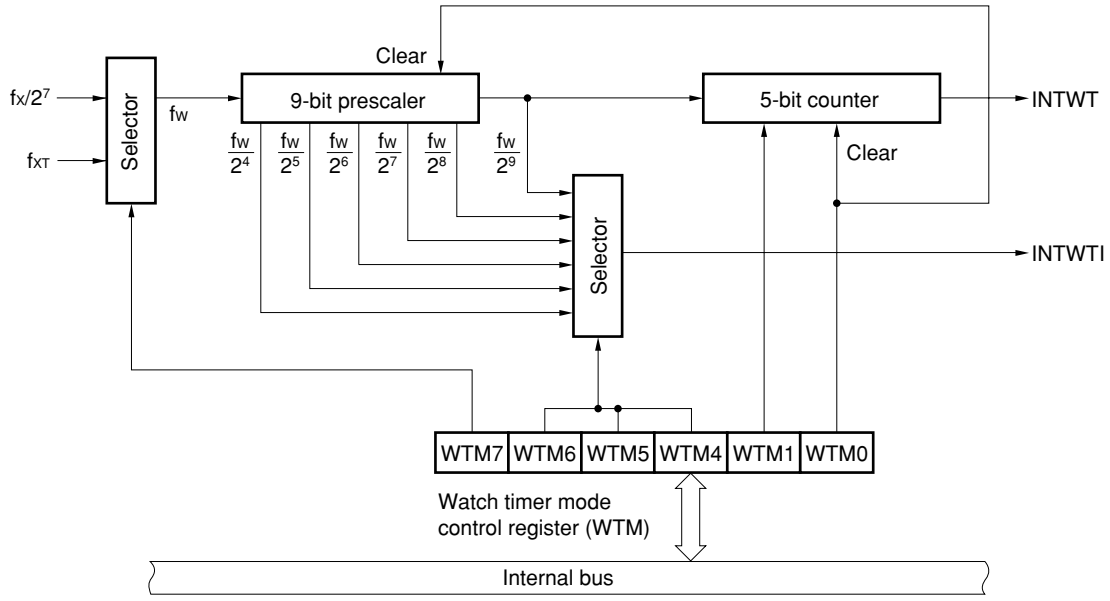
8.1 Watch Timer Functions

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. Watch Timer Block Diagram



(1) Watch timer

The 4.19 MHz main system clock or 32.768 kHz subsystem clock is used to generate an interrupt request (INTWT) at 0.5-second intervals.

Caution When the main system clock is operating at 5.0 MHz, it cannot be used to generate a 0.5-second interval. In this case, the subsystem clock, which operates at 32.768 kHz, should be used instead.

(2) Interval timer

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

Table 8-1. Interval Time of Interval Timer

Interval	During f_X = 5.0 MHz Operation	During f_X = 4.19 MHz Operation	During f_{XT} = 32.768 kHz Operation
$2^4 \times 1/fw$	409.6 μs	489 μs	488 μs
$2^5 \times 1/fw$	819.2 μs	978 μs	977 μs
$2^6 \times 1/fw$	1.64 ms	1.96 ms	1.95 ms
$2^7 \times 1/fw$	3.28 ms	3.91 ms	3.91 ms
$2^8 \times 1/fw$	6.55 ms	7.82 ms	7.81 ms
$2^9 \times 1/fw$	13.1 ms	15.6 ms	15.6 ms

Remarks 1. fw: Watch timer clock frequency ($f_X/2^7$ or f_{XT})
2. f_X: Main system clock oscillation frequency
3. f_{XT}: Subsystem clock oscillation frequency

8.2 Watch Timer Configuration

The watch timer includes the following hardware.

Table 8-2. Configuration of Watch Timer

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 watch timer mode control register (WTM) is used to control the watch timer.

- Watch timer mode control register (WTM)
  WTM selects a 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. Watch Timer Mode Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
WTM	WTM7	WTM6	WTM5	WTM4	0	0	WTM1	WTM0	FF4AH	00H	R/W

WTM7	Watch timer count clock selection
0	$f_X/2^7$ (39.1 kHz)
1	f_{XT} (32.768 kHz)

WTM6	WTM5	WTM4	Prescaler interval selection
0	0	0	$2^4/f_W$ (488 μs)
0	0	1	$2^5/f_W$ (977 μs)
0	1	0	$2^6/f_W$ (1.95 ms)
0	1	1	$2^7/f_W$ (3.91 ms)
1	0	0	$2^8/f_W$ (7.81 ms)
1	0	1	$2^9/f_W$ (15.6 ms)
Other than above			Setting prohibited

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

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

Remarks 1. f_W: Watch timer clock frequency ($f_X/2^7$ or f_{XT})

2. f_X: Main system clock oscillation frequency

3. f_{XT}: Subsystem clock oscillation frequency

4. The parenthesized values apply to operation at f_X = 5.0 MHz or at $f_{XT} = f_W$ = 32.768 kHz.

8.4 Watch Timer Operation

8.4.1 Operation as watch timer

The main system clock (4.19 MHz) or subsystem clock (32.768 kHz) is used as a watch timer with 0.5-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.

When the interval timer also operates at the same time, only the watch timer can be started from 0 seconds by setting WTM1 to 0. However, an error of up to $2^9 \times 1/f_W$ seconds may occur for the first overflow of the watch timer (INTWT) after a 0-second start because the 9-bit prescaler is not cleared in this case.

8.4.2 Operation as interval timer

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

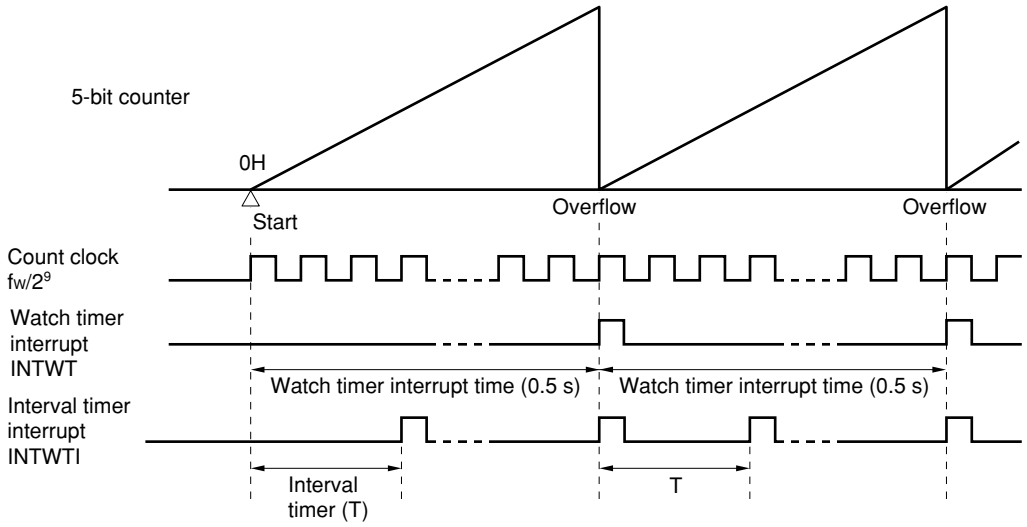
The interval time can be selected by bits 4 to 6 (WTM4 to WTM6) of the watch timer mode control register (WTM).

Table 8-3. Interval Time of Interval Timer

WTM6	WTM5	WTM4	Interval	During f_X = 5.0 MHz Operation	During f_X = 4.19 MHz Operation	During f_{XT} = 32.768 kHz Operation
0	0	0	$2^4 \times 1/f_W$	409.6 μs	489 μs	488 μs
0	0	1	$2^5 \times 1/f_W$	819.2 μs	978 μs	977 μs
0	1	0	$2^6 \times 1/f_W$	1.64 ms	1.96 ms	1.95 ms
0	1	1	$2^7 \times 1/f_W$	3.28 ms	3.91 ms	3.91 ms
1	0	0	$2^8 \times 1/f_W$	6.55 ms	7.82 ms	7.81 ms
1	0	1	$2^9 \times 1/f_W$	13.1 ms	15.6 ms	15.6 ms
Other than above			Setting prohibited			

Remark f_X: Main system clock oscillation frequency
f_{XT}: Subsystem clock oscillation frequency
f_W: Watch timer clock frequency

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



Caution When operation of the watch timer and 5-bit counter has been enabled by setting the watch timer mode control register (WTM) (setting WTM0 (bit 0 of WTM) to 1), the time until the first interrupt request after this setting will not be exactly the same as the time set by WTM3 (bit 3 of WTM). This is because the 5-bit counter starts counting one cycle after the output of the 9-bit prescaler. The INTWT signal will be generated at the set time from its second generation.

Remarks 1. f_W: Watch timer clock frequency

2. The parenthesized values apply to operation at f_W = 32.768 kHz.

CHAPTER 9 WATCHDOG TIMER

The watchdog timer can generate non-maskable interrupts, maskable interrupts and $\overline{\text{RESET}}$ with arbitrary preset intervals.

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).

(1) Watchdog timer

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

Table 9-1. Runaway Detection Time of Watchdog Timer

Runaway Detection Time	At fx = 5.0 MHz Operation
$2^{11} \times 1/fx$	410 μs
$2^{13} \times 1/fx$	1.64 ms
$2^{15} \times 1/fx$	6.55 ms
$2^{17} \times 1/fx$	26.2 ms

fx: Main system clock oscillation frequency

(2) Interval timer

The interval timer generates an interrupt at a given interval set in advance.

Table 9-2. Interval Time

Interval Time	At fx = 5.0 MHz Operation
$2^{11} \times 1/fx$	410 μs
$2^{13} \times 1/fx$	1.64 ms
$2^{15} \times 1/fx$	6.55 ms
$2^{17} \times 1/fx$	26.2 ms

fx: Main system clock oscillation frequency

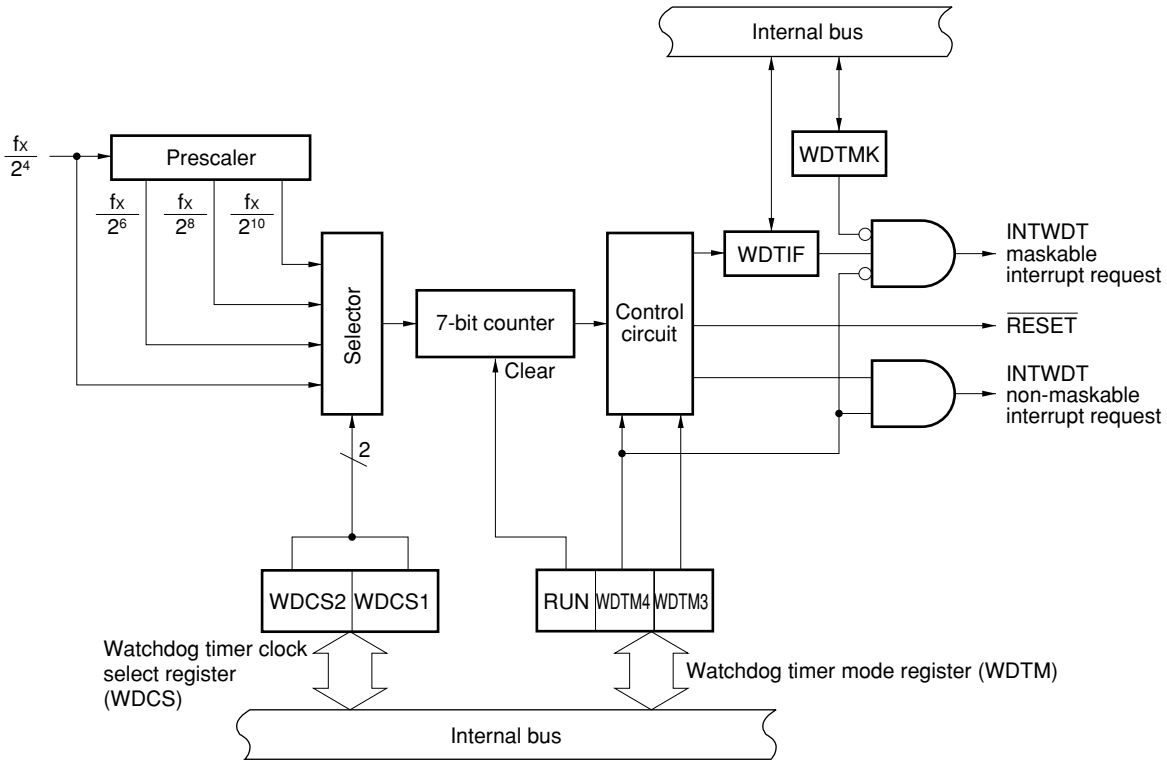
9.2 Watchdog Timer Configuration

The watchdog timer consists of the following hardware.

Table 9-3. Configuration of Watchdog Timer

Item	Configuration
Control register	Watchdog timer clock select register (WDCS) Watchdog timer mode register (WDTM)

Figure 9-1. Block Diagram of Watchdog Timer



9.3 Registers Controlling Watchdog Timer

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

- Watchdog timer clock select register (WDCS)
- Watchdog timer mode register (WDTM)

(1) Watchdog timer clock select register (WDCS)

This register sets the watchdog timer count clock.

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

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

Figure 9-2. Watchdog Timer Clock Select Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
WDCS	0	0	0	0	0	WDCS2	WDCS1	0	FF42H	00H	R/W

WDCS2	WDCS1	Watchdog timer count clock selection	Interval time
0	0	$f_X/2^4$ (312.5 kHz)	$2^{11}/f_X$ (410 μs)
0	1	$f_X/2^6$ (78.1 kHz)	$2^{13}/f_X$ (1.64 ms)
1	0	$f_X/2^8$ (19.5 kHz)	$2^{15}/f_X$ (6.55 ms)
1	1	$f_X/2^{10}$ (4.88 kHz)	$2^{17}/f_X$ (26.2 ms)
Other than above		Setting prohibited	

Remark f_X: Main system clock oscillation frequency

(2) Watchdog timer mode register (WDTM)

This register sets an operation mode of the watchdog timer, and enables/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. Watchdog Timer Mode Register Format

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	Stops counting
1	Clears counter and starts 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 (1), it cannot be cleared (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 (1), they cannot be cleared (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 watchdog timer clock select register (WDCS).

2. In watchdog timer mode 1 or 2, set WDTM4 to 1 after confirming WDTIF (bit 0 of interrupt request flag 0) has been set to 0. When watchdog timer mode 1 or 2 is selected under the condition where WDTIF is 1, a non-maskable interrupt occurs at the completion of rewriting.

9.4 Watchdog Timer Operation

9.4.1 Operation as watchdog timer

The watchdog timer operates to detect a runaway when bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 1.

The count clock (runaway detection time interval) of the watchdog timer can be selected by bits 1 and 2 (WDCS1 and WDCS2) of watchdog timer clock select register (WDCS). By setting bit 7 (RUN) of WDTM to 1, the watchdog timer is started. Set RUN to 1 within the set runaway 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 runaway detection time is exceeded, the system is reset or a non-maskable interrupt is generated by the value of bit 3 (WDTM3) of WDTM.

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

Cautions 1. The actual runaway detection time may be up to 0.8% shorter than the set time.

2. When the subsystem clock is specified as the CPU clock, the count operation of the watchdog timer stops. Therefore, even if the main system clock is oscillating at that time, watchdog timer operation stops.

Table 9-4. Runaway Detection Time of Watchdog Timer

WDCS2	WDCS1	At f_X = 5.0 MHz Operation
0	0	410 μs
0	1	1.64 ms
1	0	6.55 ms
1	1	26.2 ms

f_X: Main system clock oscillation frequency

9.4.2 Operation as interval timer

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

Select a count clock (or interval time) by setting bits 1 and 2 (WDCS1 and WDCS2) of watchdog timer clock select register (WDCS). The watchdog timer starts operation as an interval timer when the RUN bit (bit 7 of WDTM) is set to 1.

In the interval timer mode, the interrupt mask flag (WDTMK) is valid, and a maskable interrupt (INTWDT) can be generated. The priority of INTWDT is set as the highest of all the maskable interrupts.

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

WDCS2	WDCS1	At fx = 5.0 MHz Operation
0	0	410 μs
0	1	1.64 ms
1	0	6.55 ms
1	1	26.2 ms

fx: Main system clock oscillation frequency

CHAPTER 10 SERIAL INTERFACE 10

10.1 Serial Interface 10 Functions

The serial interface 10 has the following two modes.

- Operation stop mode
- 3-wire serial I/O mode

(1) Operation stop mode

This mode is used when serial transfer is not carried out. It reduces power consumption.

(2) 3-wire serial I/O mode (MSB first/LSB first selectable)

In this mode, 8-bit data transfer is carried out with three lines, one for serial clock ($\overline{SCK10}$) and two for serial data (SI10 and SO10).

The 3-wire serial I/O mode supports simultaneous transmit and receive operation, reducing data transfer processing time.

It is possible to select the start bit of 8-bit data to be transmitted between the MSB and the LSB, thus allowing connection to devices with either start bit.

The 3-wire serial I/O mode is effective for connecting display controllers and peripheral I/Os such as the 75XL Series, 78K Series, and 17K Series that have internal conventional clock synchronous serial interface.

10.2 Serial Interface 10 Configuration

Serial interface 10 has the following hardware configuration.

Table 10-1. Configuration of Serial Interface 10

Item	Configuration
Register	Transmit/receive shift register 10 (SIO10)
Control register	Serial operating mode register 10 (CSIM10)

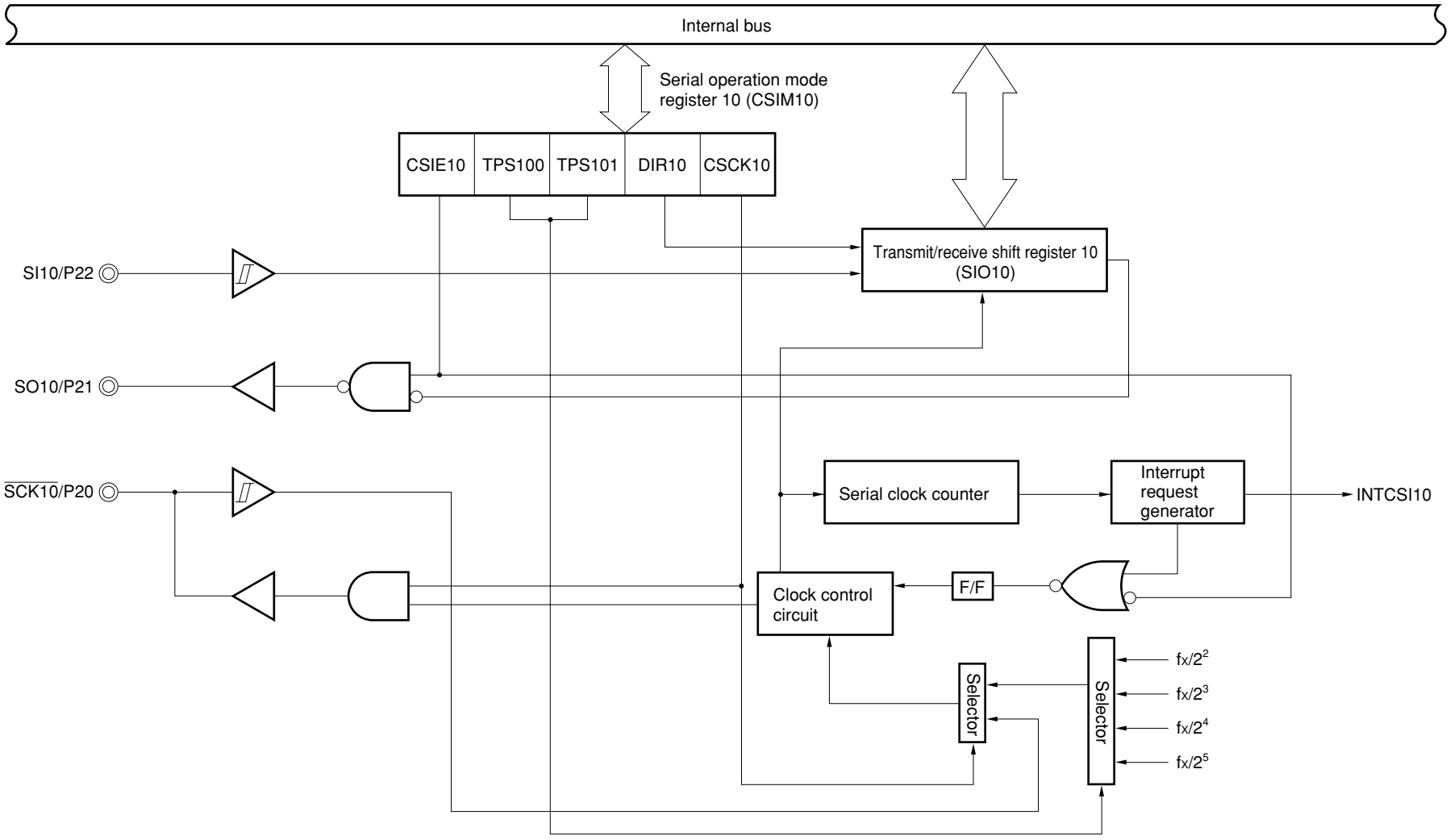
(1) Transmit/receive shift register 10 (SIO10)

This is an 8-bit register used for parallel-to-serial conversion and to perform serial data transmission/reception in synchronization with serial clocks.

SIO10 is set with an 8-bit memory manipulation instruction.

$\overline{RESET}$ input makes SIO10 undefined.

Figure 10-1. Block Diagram of Serial Interface 10



10.3 Register Controlling Serial Interface 10

The following register is used to control serial interface 10.

- Serial operation mode register 10 (CSIM10)

(1) Serial operation mode register 10 (CSIM10)

This register is used to control serial interface 10 operation and set the serial clock and start bit.

CSIM10 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears CSIM10 to 00H.

Figure 10-2. Serial Operation Mode Register 10 Format

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM10	CSIE10	0	TPS101	TPS100	0	DIR10	CSCK10	0	FF72H	00H	R/W

CSIE10	Operation control in 3-wire serial I/O mode
0	Operation stopped
1	Operation enabled

TPS101	TPS100	Count clock selection during operation enable in 3-wire serial I/O mode
0	0	$f_X/2^2$ (1.25 MHz)
0	1	$f_X/2^3$ (625 kHz)
1	0	$f_X/2^4$ (313 kHz)
1	1	$f_X/2^5$ (157 kHz)

DIR10	Start bit specification
0	MSB
1	LSB

CSCK10	Clock selection in 3-wire serial I/O mode
0	Input clock to $\overline{\text{SCK10}}$ pin from external
1	Internal clock selected by TPS100, TPS101

Caution Bits 0, 3, and 6 must be set to 0.

Remarks 1. f_X: Main system clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 50 MHz.

Table 10-2. Settings of Serial Interface 10 Operation Mode

(1) Operation stop mode

CSIM10			PM22	P22	PM21	P21	PM20	P20	Start	Shift	P22/SI10	P21/SO10	P20/$\overline{SCK10}$
CSIE10	DIR10	CSCK10							Bit	Clock	Pin Function	Pin Function	Pin Function
0	×	×	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	–	–	P22	P21	P20
Other than above									Setting prohibited				

(2) 3-wire serial I/O mode

CSIM10			PM22	P22	PM21	P21	PM20	P20	Start	Shift	P22/SI10	P21/SO10	P20/$\overline{SCK10}$
CSIE10	DIR10	CSCK10							Bit	Clock	Pin Function	Pin Function	Pin Function
1	0	0	1Note 2	×Note 2	0	1	1	×	MSB	External clock	SI10Note 2	SO10 (CMOS output)	$\overline{SCK10}$ input
		1					0	1		External clock			$\overline{SCK10}$ output
1	1	0					1	×	LSB	External clock			$\overline{SCK10}$ input
		1					0	1		External clock			$\overline{SCK10}$ output
Other than above									Setting prohibited				

Notes 1. Can be used as port function.

2. If used only for transmission, can be used as P22 (CMOS I/O port).

Remark ×: don't care

10.4 Serial Interface 10 Operation

Serial interface 10 provides the following two types of modes.

- Operation stop mode
- 3-wire serial I/O mode

10.4.1 Operation stop mode

In the operation stop mode, serial transfer is not executed; therefore, the power consumption can be reduced. P20/$\overline{SCK10}$, P21/SO10, and P22/SI10 pins can be used as normal I/O ports.

(1) Register setting

Operation stop mode is set by serial operation mode register 10 (CSIM10).

Serial operation mode register 10 (CSIM10)

CSIM10 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{RESET}$ input clears CSIM10 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM10	CSIE10	0	TPS101	TPS100	0	DIR10	CSCK10	0	FF72H	00H	R/W

CSIE10	Operation control in 3-wire serial I/O mode
0	Operation stopped
1	Operation enabled

Caution Bits 0, 3, and 6 must be set to 0.

10.4.2 3-wire serial I/O mode

The 3-wire serial I/O mode is useful for connection of peripheral I/Os and display controllers, etc., which incorporate a conventional clocked serial interface, such as the 75XL Series, 78K Series, 17K Series, etc.

Communication is performed using three lines: the serial clock ($\overline{SCK10}$), serial output (SO10), and serial input (SI10).

(1) Register setting

3-wire serial I/O mode settings are performed using serial operation mode register 10 (CSIM10).

(a) Serial operation mode register 10 (CSIM10)

CSIM10 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{RESET}$ input clears CSIM10 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM10	CSIE10	0	TPS101	TPS100	0	DIR10	CSCK10	0	FF72H	00H	R/W

CSIE10	Operation control in 3-wire serial I/O mode
0	Operation stopped
1	Operation enabled

TPS101	TPS100	Count clock selection during operation enabled in 3-wire serial I/O mode
0	0	$f_X/2^2$ (1.25 MHz)
0	1	$f_X/2^3$ (625 kHz)
1	0	$f_X/2^4$ (313 kHz)
1	1	$f_X/2^5$ (157 kHz)

DIR10	Start bit specification
0	MSB
1	LSB

CSCK10	Clock selection in 3-wire serial I/O mode
0	Input clock to $\overline{SCK10}$ pin from external
1	Count clock selected by TPS100, TPS101

Caution Bits 0, 3, and 6 must be set to 0.

Remarks 1. f_X: Main system clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 5.0 MHz.

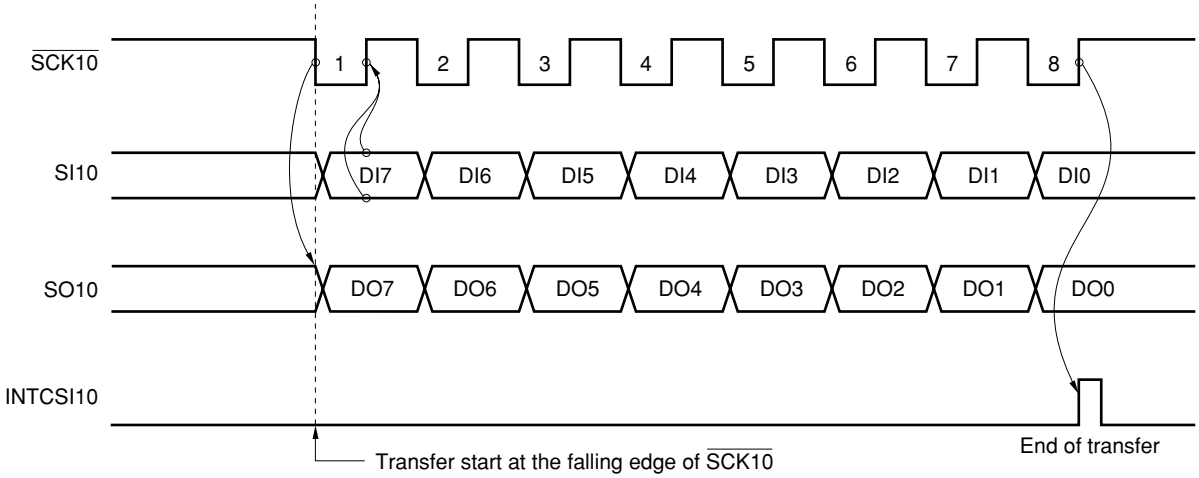
(2) Communication operation

In the 3-wire serial I/O mode, data transmission/reception is performed in 8-bit units. Data is transmitted/received in 1-bit units in synchronization with the serial clock.

Transmit/receive shift register 10 (SIO10) shift operations are performed in synchronization with the fall of the serial clock ($\overline{SCK10}$). Then transmit data is held in the SO10 latch and output from the SO10 pin. Also, receive data input to the SI10 pin is latched in input bits of SIO10 on the rise of $\overline{SCK10}$.

At the end of an 8-bit transfer, the operation of SIO10 stops automatically, and the interrupt request signal (INTCSI10) is generated.

Figure 10-3. 3-Wire Serial I/O Mode Timing



Cautions 1. When data is written to SIO10 in the serial operation disabled status (CSIE10 = 0), the data cannot be transmitted or received.

2. When data is written to SIO10 in the serial operation disabled status (CSIE10 = 0) and then serial operation is enabled (CSIE10 = 1), the data cannot be transmitted or received.

3. Once data has been written to SIO10 with the serial clock selected (CSCK10 = 0), overwriting the data does not update the contents of SIO10.

4. When CSIM10 is operated during data transmission/reception, data cannot be transmitted or received normally.

5. When SIO10 is operated during data transmission/reception, the data cannot be transmitted or received normally.

(3) Transfer start

Serial transfer is started by setting transfer data to the transmit/receive shift register 10 (SIO10) when the following two conditions are satisfied.

- Bit 7 (CSIE10) of serial operation mode register 10 (CSIM10) = 1
- Internal serial clock is stopped or $\overline{SCK10}$ is a high level after 8-bit serial transfer.

An end of 8-bit transfer stops the serial transfer automatically and generates the interrupt request signal (INTCSI10).

CHAPTER 11 VFD CONTROLLER/DRIVER

11.1 VFD Controller/Driver Functions

The on-chip VFD controller/driver of the μPD789871 Subseries has the following functions.

(1) Can output display signals (DMA operation) by automatically reading display data.
(2) The pins not used for VFD display can be used as I/O port or output port pins (FIP9 to FIP24 pins only).
(3) Luminance can be adjusted in 8 steps by VFD display mode register 1 (DSPM1).
(4) Hardware for key scan application
    • Generates an interrupt signal (INTKS) indicating key scan timing.
    • Timing in which key scan data is output can be detected by key scan flag (KSF).
    • Whether key scan timing is inserted or not can be selected.
(5) High-tolerance output buffer that can directly drive VFD
(6) The FIP17 to FIP24 pins can be connected to pull-down resistors (connected to V_{LOAD}) using mask option (mask ROM version only). The μPD78F9872 does not have on-chip pull-down resistors).

Of the 25 VFD output pins of the μPD789871 Subseries, FIP9 to FIP24 are alternate function pins with port function. FIP0 to FIP8 are output-only pins.

FIP9 to FIP24 can be used as port pins when VFD display is disabled by bit 7 (DSPEN) of the display mode register 0 (DSPM0). Even when VFD display is enabled, the VFD output pins not used for display signal output can be used as port pins.

Table 11-1. VFD Output Pins and Alternate-Function Pins for Ports

FIP Pin Name	Multiplexed Port Name	I/O
FIP9 to FIP16	P97 to P90	Output-only port
FIP17 to FIP24	P87 to P80	I/O port

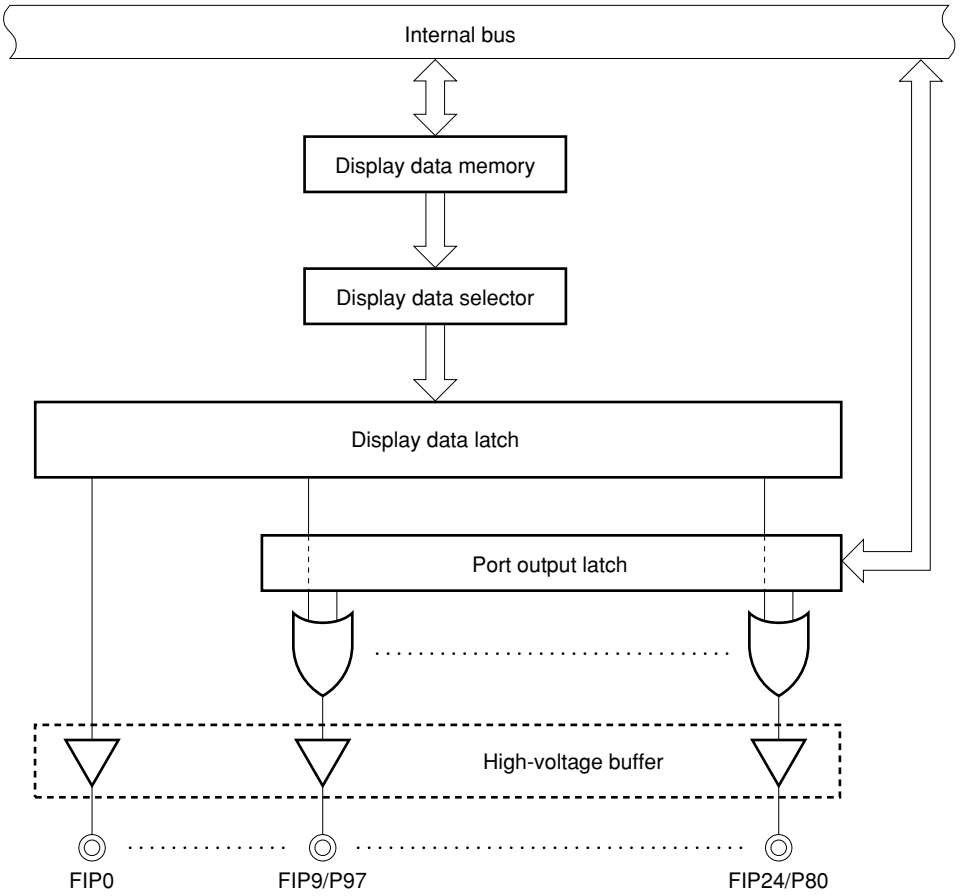
11.2 VFD Controller/Driver Configuration

The VFD controller/driver consists of the following hardware.

Table 11-2. Configuration of VFD Controller/Driver

Item	Configuration
Display	25
Control register	Display mode register 0 (DSPM0) Display mode register 1 (DSPM1) Display mode register 2 (DSPM2)

Figure 11-1. Block Diagram of VFD Controller/Driver



11.3 Registers Controlling VFD Controller/Driver

11.3.1 Control registers

The following three types of registers control the VFD controller/driver.

- Display mode register 0 (DSPM0)
- Display mode register 1 (DSPM1)
- Display mode register 2 (DSPM2)

(1) Display mode register 0 (DSPM0)

DSPM0 performs the following setting.

- Enables or disables display
- Number of VFD output pins

DSPM0 is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input sets DSPM0 to 10H.

Figure 11-2. Display Mode Register 0 Format

Symbol	< 7 >	6	5	4	3	2	1	0	Address	After reset	R/W
DSPM0	DSPEN	0	FOUT5	FOUT4	FOUT3	FOUT2	FOUT1	FOUT0	FFA0H	10H	R/W

DSPEN	Enables or disables VFD display
0	Disables
1	Enables

FOUT5	FOUT4	FOUT3	FOUT2	FOUT1	FOUT0	Number of VFD output pins
0	1	0	0	0	0	17
0	1	0	0	0	1	18
0	1	0	0	1	0	19
0	1	0	0	1	1	20
0	1	0	1	0	0	21
0	1	0	1	0	1	22
0	1	0	1	1	0	23
0	1	0	1	1	1	24
0	1	1	0	0	0	25
Other than above						Setting prohibited

Cautions 1. Be sure to set bit 6 to 0.

2. Do not write data to the bits other than DSPEN when bit 7 (DSPEN) = 1.

3. Be sure to set the output latch of the alternate-function port of a pin used for VFD output to 0.

(2) Display mode register 1 (DSPM1)

DSPM1 performs the following setting.

- Blanking width of VFD output signal
- Number of display patterns

DSPM1 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets DSPM1 to 01H.

Figure 11-3. Display Mode Register 1 Format

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DSPM1	FBLK2	FBLK1	FBLK0	FPAT4	FPAT3	FPAT2	FPAT1	FPAT0	FFA1H	01H	R/W

FBLK2	FBLK1	FBLK0	Blanking width of VFD output signal
0	0	0	1/16
0	0	1	2/16
0	1	0	4/16
0	1	1	6/16
1	0	0	8/16
1	0	1	10/16
1	1	0	12/16
1	1	1	14/16

FPAT4	FPAT3	FPAT2	FPAT1	FPAT0	Number of display patterns
0	0	0	0	1	2
0	0	0	1	0	3
0	0	0	1	1	4
0	0	1	0	0	5
0	0	1	0	1	6
0	0	1	1	0	7
0	0	1	1	1	8
0	1	0	0	0	9
0	1	0	0	1	10
0	1	0	1	0	11
0	1	0	1	1	12
0	1	1	0	0	13
0	1	1	0	1	14
0	1	1	1	0	15
0	1	1	1	1	16
Other than above					Setting prohibited

Caution Do not write data to the display mode register 1 (DSPM1) when bit 7 (DSPEN) of the display mode register 0 (DSPM0) is 1.

(3) Display mode register 2 (DSPM2)

DSPM2 performs the following setting. It also indicates the status of the display timing/key scan.

- Insertion of key scan timing
- Display cycle (T_{DSP})

DSPM2 is set with a 1-bit or 8-bit memory manipulation instruction. However, only bit 7 (KSF) can be read by a 1-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears DSPM2 to 00H.

Figure 11-4. Display Mode Register 2 Format

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
DSPM2	KSF	KSM	0	0	0	0	FCYC1	FCYC0	FFA2H	00H	R/W

KSF	Status of key scan cycle
0	Other than key scan cycle
1	Key scan cycle

KSM	Key scan cycle insertion selection
0	Not inserted
1	Inserted

FCYC1	FCYC0	Display cycle
0	0	$2^{12}/f_X$ (819 μs)
0	1	$2^{11}/f_X$ (410 μs)
1	0	$2^{10}/f_X$ (205 μs)
1	1	Setting prohibited

Cautions 1. Be sure to set bits 2 to 5 to 0.

2. Do not write data to the display mode register 2 (DSPM2) when bit 7 (DSPEN) of the display mode register 0 (DSPM0) is 1.

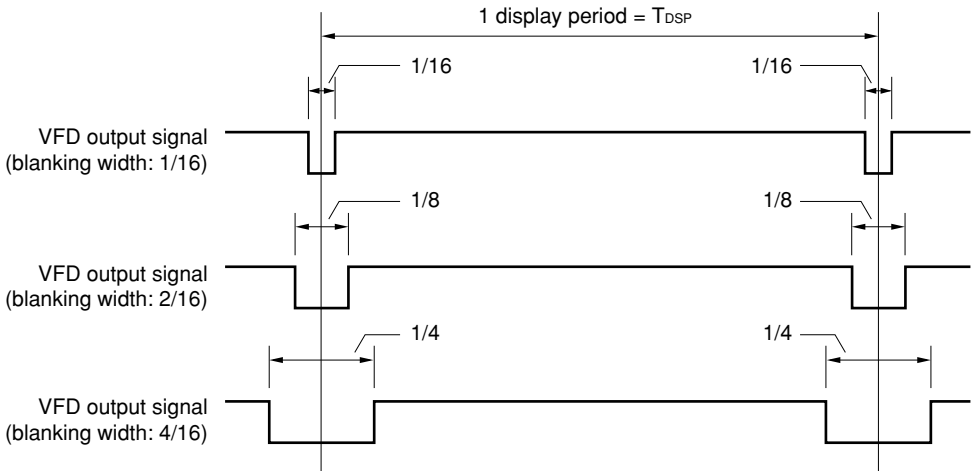
Remarks 1. f_X: Main system clock oscillation frequency

2. The parenthesized values apply to operation at f_X = 5.0 MHz.

11.3.2 One display period and blanking width

The VFD output signals are blanked equally at the beginning and end of the display period by the blanking width set by bits 0 to 2 (FBLK0 to FBLK2) of the display mode register 1 (DSPM1).

Figure 11-5. Blanking Width of VFD Output Signal



11.4 Display Data Memory

The display data memory is a 96-byte RAM area that stores data to be displayed, and is mapped to addresses FA00H to FA5FH.

The VFD controller reads the data stored in the display data memory independently of the CPU operation for VFD display (DMA operation).

The area of the display data memory not used for display can be used as a normal RAM area.

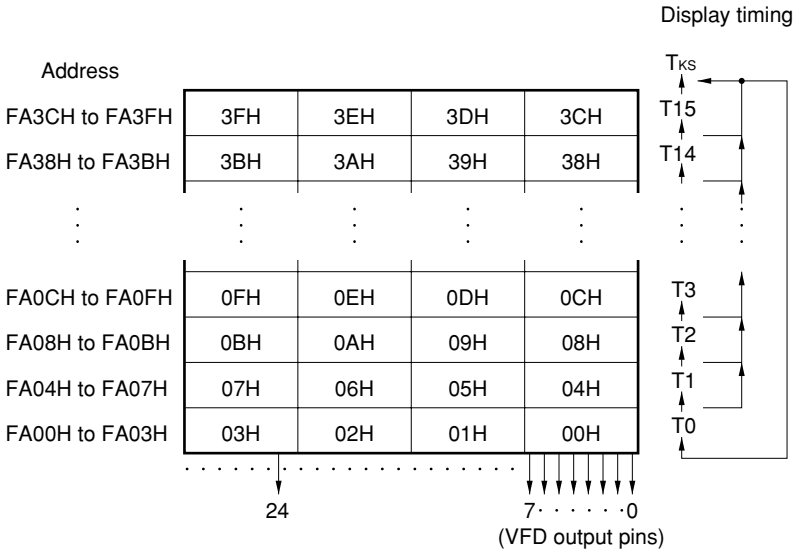
At key scan timing (T_{KS}), all the VFD output pins are cleared to 0, and the data of the output latches of ports 8 and 9 are output to FIP9/P97 to FIP24/P80.

The address location of the display data memory is as follows:

(1) With 25 VFD output pins and 16 patterns

The addresses of the display data memory corresponding to the data output at each display timing (T0 to T15) are as shown in Figure 11-6 (for example, T0 = FA00H to FA03H, and T1 = FA04H to FA07H). When 25 VFD output pins (FIP0 to FIP24) are used, one block of display data consists of 4 bytes. VFD output pins 0 (FIP0) to 24 (FIP24) correspond to one block of display data sequentially, starting from the least significant bit toward the most significant bit.

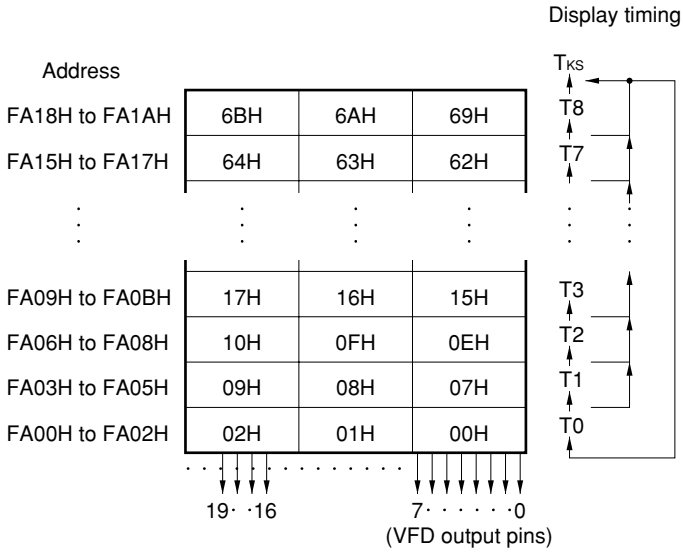
Figure 11-6. Relationship Between Address Location of Display Data Memory and VFD Output (with 25 VFD Output Pins and 16 Patterns)



(2) With 20 VFD output pins and 9 patterns

The addresses of the display data memory corresponding to the data output at each display timing (T0 to T8) are as shown in Figure 11-7 (for example, T0 = FA00H to FA02H, and T1 = FA03H to FA05H). When 20 VFD output pins (FIP0 to FIP19) are used, one block of display data consists of 3 bytes. VFD output pins 0 (FIP0) to 19 (FIP19) correspond to one block of display data sequentially, starting from the least significant bit toward the most significant bit.

Figure 11-7. Relationship Between Address Location of Display Data Memory and VFD Output (with 20 VFD Output Pins and 9 Patterns)



11.5 Key Scan Flag and Key Scan Data

11.5.1 Key scan flag

The key scan flag (KSF) is set to 1 during key scan timing, and is automatically reset to 0 at display timing.

KSF is mapped to bit 7 of the display mode register 2 (DSPM2) and can be tested in 1-bit units. It cannot be written, however.

By testing KSF, it can be determined whether key scan timing is in progress, and whether key input data is correct can be checked.

Whether key scan timing is inserted or not can be selected by using the key scan timing insertion specification flag (KSM) (bit 6 of the display mode register 2 (DSPM2)).

11.5.2 Key scan data

Data stored in ports 8 and 9 are output from the FIP9 to FIP24 pins during key scan timing.

Caution If scanning is performed in such a manner that both a segment and a digit turn ON during key scan timing, the display may flicker.

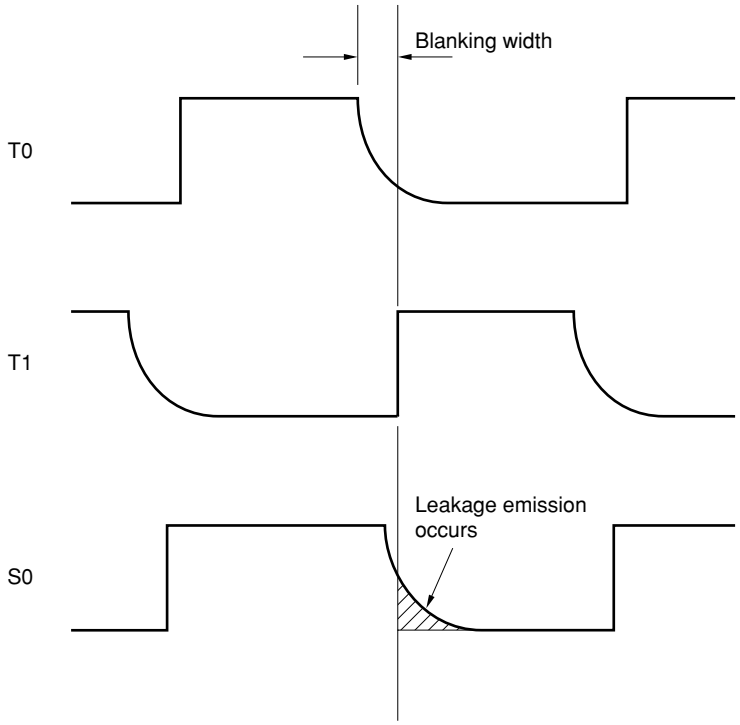
11.6 Leakage Emission of Fluorescent Indicator Panel

Leakage emission may take place when a fluorescent indicator panel is driven by the μPD789871 Subseries. The possible causes of this leakage emission are as follows.

(1) Short blanking time

Figure 11-8 shows the signal waveforms of a 2-digit display where the first digit T0 lights and the second digit T1 remains dark. If the blanking time is too short as shown in this figure, the T1 signal rises before the segment signal is deasserted, causing leakage emission. Generally, the blanking time must be about 20 μs. Determine the set value of the display mode register 1 (DSPM1), taking this into consideration.

Figure 11-8. Leakage Emission Because of Short Blanking Time



(2) Segment-grid capacitance of fluorescent indicator panel

Even if a sufficiently long blanking time is ensured as shown in Figure 11-10, leakage emission may still occur. This is because the fluorescent indicator panel has a capacitance between the grid and segment, as indicated by C_{SG} in Figure 11-9, and the timing signal pin is raised via C_{SG} when segment signal turns on. If the voltage of the timing signal rises beyond the cutoff voltage (E_K) as shown in Figure 11-10, leakage emission occurs. This whisker-like voltage changes with the values of C_{SG} and internal pull-down resistor (R_L). The greater the value of C_{SG}, and the greater the value of R_L, the higher this voltage, increasing the possibility of the occurrence of leakage emission.

The value of C_{SG} differs depending on the display area of the fluorescent indicator panel. The larger the area, the higher the C_{SG}.

Therefore, the value of the pull-down resistor differs depending on the size of the fluorescent indicator panel, in order to prevent leakage emission.

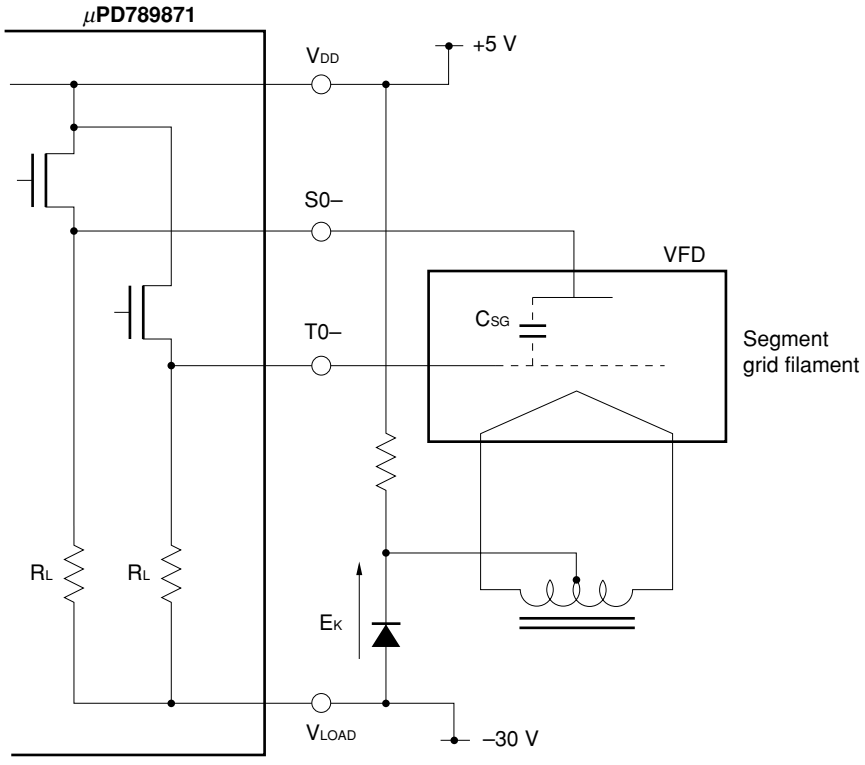
Because the value of the pull-down resistor that can be connected by mask option is relatively high, the leakage emission may not be suppressed by the internal pull-down resistor alone.

In case sufficient display quality cannot be obtained, deepen the back bias (increase E_K), attach a filter to the fluorescent indicator panel, or connect an external pull-down resistor of several 10 kΩ to the timing signal pin.

The likelihood of leakage emission caused by C_{SG} occurring changes depending on the duty cycle of the whisker voltage vis-a-vis the total display cycle. The fewer the number of display digits, the greater the likelihood of occurrence of leakage emission.

Lowering the display luminance also has an effect of suppressing the leakage emission.

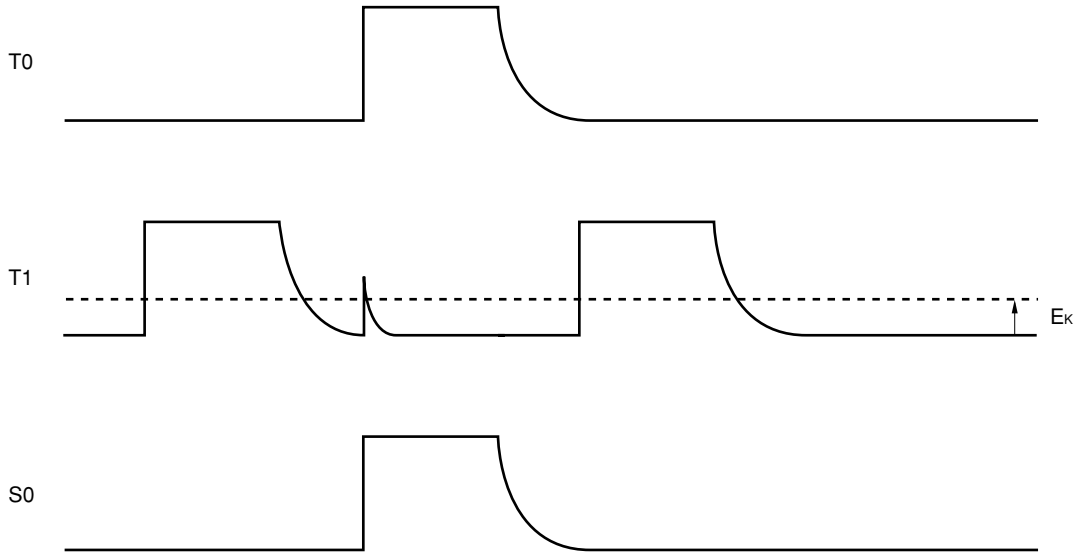
Figure 11-9. Leakage Emission Caused by C_{SG}



E_K : Cutoff voltage

R_L : On-chip pull-down resistor

Figure 11-10. Leakage Emission Caused by C_{SG}



11.7 Calculation of Total Power Dissipation

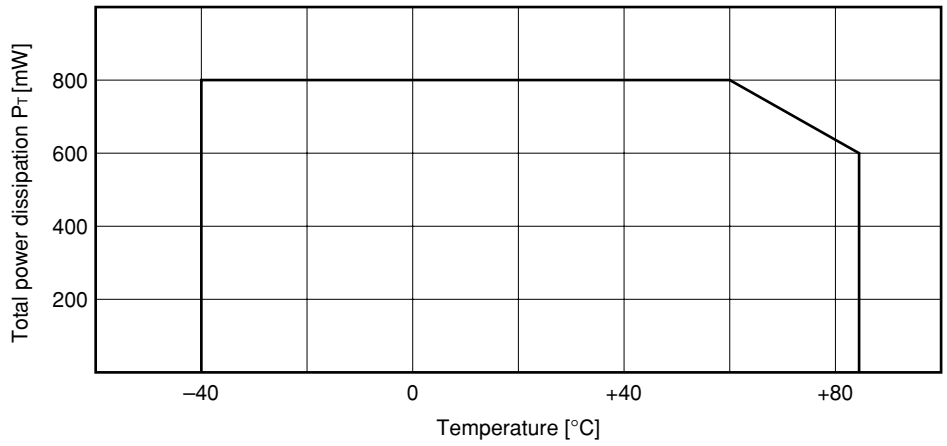
The following three power consumption are available for the μPD789871 Subseries. The sum of the three power consumption should be less than the total power dissipation P_T (refer to **Figure 11-11**) (80% or less of ratings is recommended).

<1> CPU power consumption: Calculate V_{DD} (MAX.) × I_{DD} (MAX.).

<2> Output pin power consumption: Power consumption when maximum current flows into each VFD output pin.

<3> Pull-down resistor power consumption: Power consumption by pull-down resistor incorporated in VFD output pin.

Figure 11-11. Total Power Dissipation P_T (T_A = –40 to +85°C)



The following is how to calculate total power dissipation for the example in Figure 11-12.

Example Assume the following conditions:

V_{DD} = 5.5 V, 5.0 MHz oscillation

Supply current (I_{DD}) = 15 mA

VFD output: 11 grids × 10 segments (Blanking width = 1/16: when FBLK0 to FBLK2 = 000B)

Maximum current at the grid pin is 15 mA.

Maximum current at the segment pin is 5 mA.

At the key scan timing, VFD output pin is OFF.

VFD output voltage: grid $V_{OD} = V_{DD} – 2$ V (voltage drop of 2 V)

segments $V_{OD} = V_{DD} – 0.5$ V (voltage drop of 0.5 V)

Fluorescent indicator panel voltage (V_{LOAD}) = –35 V

Mask option pull-down resistor = 30 kΩ

By placing the above conditions in calculation <1> to <3>, the total dissipation can be worked out.

<1> CPU power consumption: 5.5 V × 15 mA = 82.5 mW

<2> Output pin power consumption:

$$\text{Grid} \quad (V_{DD} - V_{OD}) \times \frac{\text{Total current value of each grid}}{\text{Number of grids} + 1} \times (1 - \text{Blanking width}) =$$

$$2\ \text{V} \times \frac{15\ \text{mA} \times 11\ \text{grids}}{11\ \text{grids} + 1} \times \left(1 - \frac{1}{16}\right) = 25.8\ \text{mW}$$

$$\text{Segment} \quad (V_{DD} - V_{OD}) \times \frac{\text{Total segment current value of illuminated dots}}{\text{Number of grids} + 1} \times (1 - \text{Blanking width}) =$$

$$0.5\ \text{V} \times \frac{5\ \text{mA} \times 31\ \text{dots}}{11\ \text{grids} + 1} \times \left(1 - \frac{1}{16}\right) = 6.1\ \text{mW}$$

<3> Pull-down resistor power consumption:

$$\text{Grid} \quad \frac{(V_{DD} - V_{LOAD})^2}{\text{Pull-down resistor value}} \times \frac{\text{Number of grids}}{\text{Number of grids} + 1} \times (1 - \text{Blanking width}) =$$

$$\frac{(5.5\ \text{V} - 2\ \text{V} - (-35\ \text{V}))^2}{30\ \text{k}\Omega} \times \frac{11\ \text{grids}}{11\ \text{grids} + 1} \times \left(1 - \frac{1}{16}\right) = 42.5\ \text{mW}$$

$$\text{Segment} \quad \frac{(V_{OD} - V_{LOAD})^2}{\text{Pull-down resistor value}} \times \frac{\text{Number of illuminated dots}}{\text{Number of grids} + 1} \times (1 - \text{Blanking width}) =$$

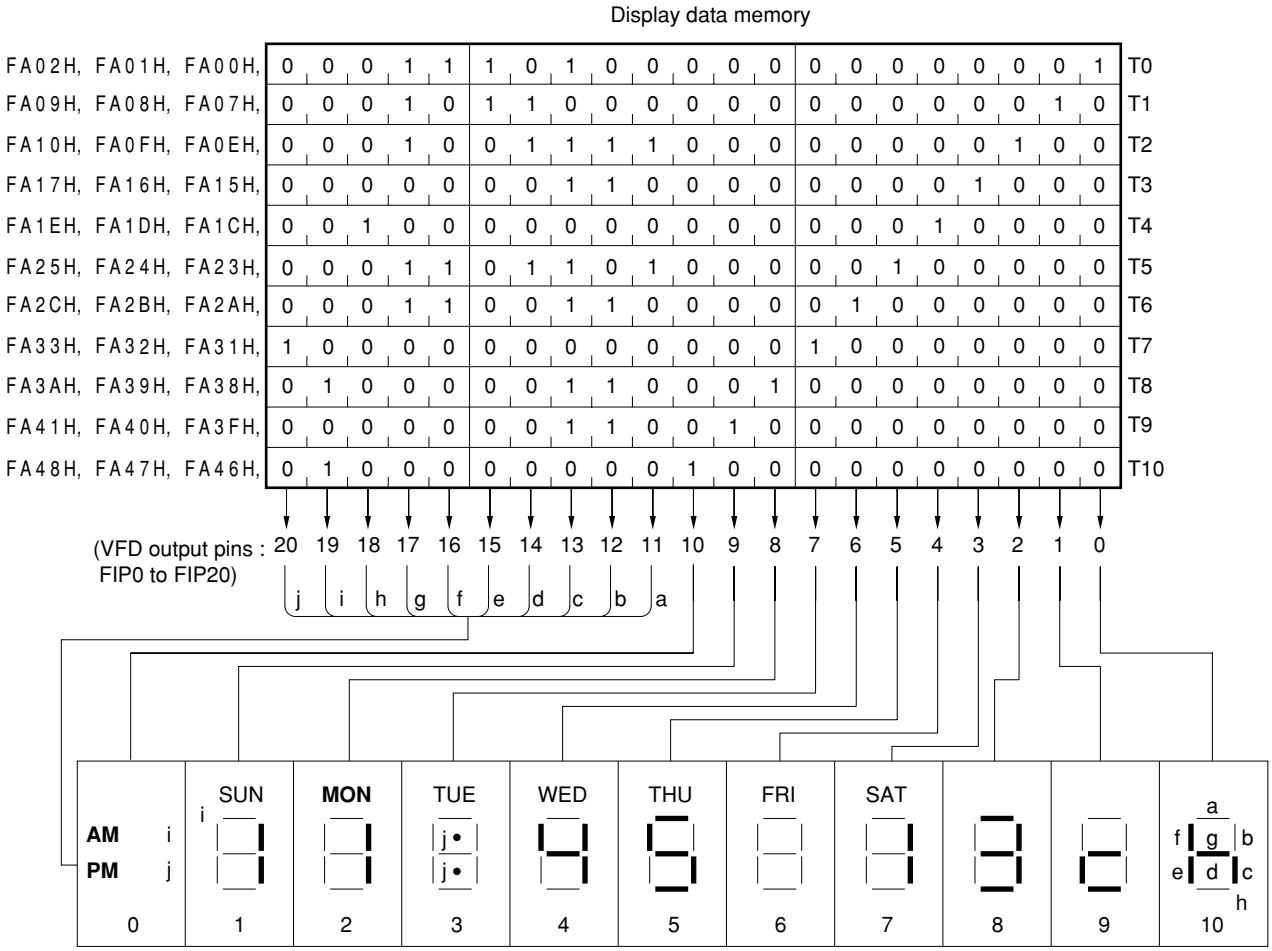
$$\frac{(5.5\ \text{V} - 2\ \text{V} - (-35\ \text{V}))^2}{30\ \text{k}\Omega} \times \frac{31\ \text{dots}}{11\ \text{grids} + 1} \times \left(1 - \frac{1}{16}\right) = 129.2\ \text{mW}$$

Total power consumption = <1> + <2> + <3> = 82.5 + 25.8 + 6.1 + 42.5 + 129.2 = 286.1 mW

In this example, the total power consumption do not exceed the rating of the total power dissipation, so there is no problem in power consumption.

However, when the total power consumption exceeds the rating of the total power dissipation, it is necessary to lower the power consumption. To reduce power consumption, reduce the number of pull-down resistors.

Figure 11-12. Relationship Between Display Data Memory Contents and VFD Output with 10 Segments-11 Digits Displayed



CHAPTER 12 INTERRUPT FUNCTIONS

12.1 Interrupt Function Types

The following two types of interrupt functions are used.

(1) Non-maskable interrupt

This interrupt is acknowledged unconditionally. It does not undergo interrupt priority control and is given top priority over all other interrupt requests.

A standby release signal is generated.

The non-maskable interrupt has one source of interrupt from the watchdog timer.

(2) Maskable interrupt

These interrupts undergo mask control. If two or more interrupts with the same priority are simultaneously generated, each interrupt has a predetermined priority as shown in Table 12-1.

A standby release signal is generated.

The maskable interrupt has four sources of external interrupts and seven sources of internal interrupts.

12.2 Interrupt Sources and Configuration

There are total of 12 non-maskable and maskable interrupts in the interrupt sources (see **Table 12-1**).

Table 12-1. Interrupt Source List

Interrupt Type	Priority[Note 1]	Interrupt Source		Internal /External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	—	INTWDT	Watchdog timer overflow (watchdog timer mode 1 selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (interval timer mode selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTTM50	TI pin input rising edge detection		000AH	(D)
	4	INTTM51	TI pin input falling edge detection		000CH	
	5	INTTM52	8-bit remote control timer over flow signal	Internal	000EH	(B)
	6	INTKS	Key scan timing		0010H	
	7	INTCSI10	End of serial interface 10 transmission/ reception		0012H	
	8	INTTM80	Generation of 8-bit timer 80 match signal		0014H	
	9	INTTM81	Generation of 8-bit timer 81 match signal		0016H	
	10	INTWT	Watch timer interrupt		0018H	
	11	INTWTI	Watch timer interval timer interrupt		001AH	

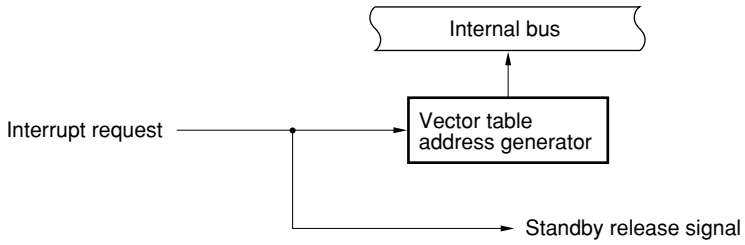
Notes 1. Priority is the priority applicable when two or more maskable interrupts are simultaneously generated. 0 is the highest priority and 11 is the lowest priority.

2. Basic configuration types A to C correspond to A to C in Figure 12-1.

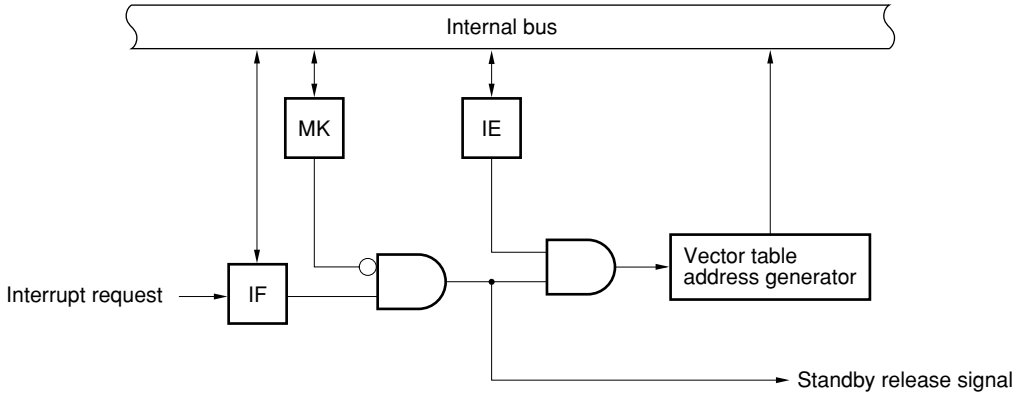
Remark As the interrupt source of the watchdog timer (INTWDT), either a non-maskable interrupt or a maskable interrupt (internal) can be selected.

Figure 12-1. Basic Configuration of Interrupt Function

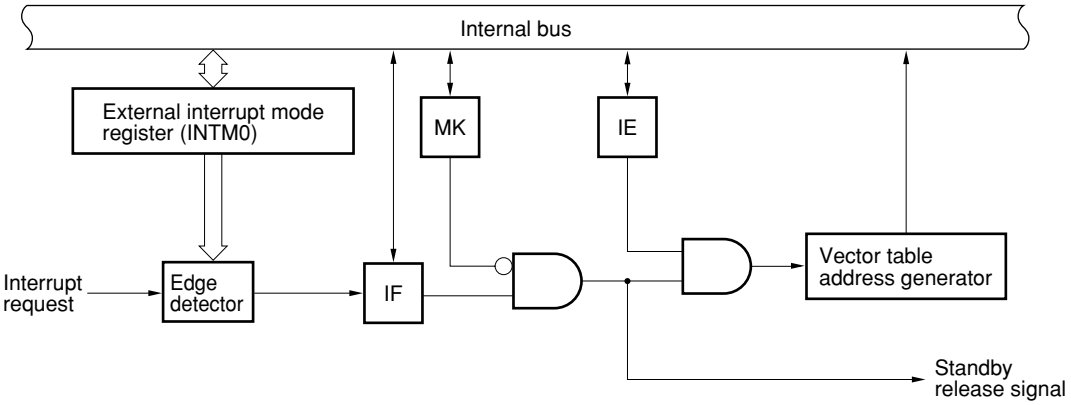
(A) Internal non-maskable interrupt



(B) Internal maskable interrupt

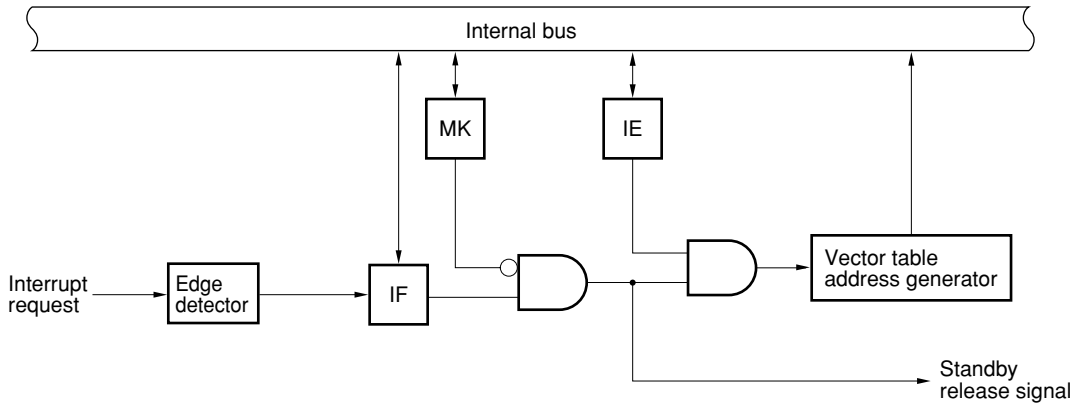


(C) External maskable interrupt (INTP0, INTP1)



IF: Interrupt request flag
IE: Interrupt enable flag
MK: Interrupt mask flag

(D) External maskable interrupt (INTTM50, INTTM51)



IF: Interrupt request flag
IE: Interrupt enable flag
MK: Interrupt mask flag

12.3 Interrupt Function Control Registers

The following four registers are used to control the interrupt functions.

- Interrupt request flag registers (IF0, IF1)
- Interrupt mask flag registers (MK0, MK1)
- External interrupt mode register (INTM0)
- Program status word (PSW)

Table 12-2 gives a listing of interrupt request flag and interrupt mask flag names corresponding to interrupt requests.

Table 12-2. Flags Corresponding to Interrupt Request Signal

Interrupt Request Signal Name	Interrupt Request Flag	Interrupt Mask Flag
INTWDT	WDTIF	WDTMK
INTP0	PIF0	PMK0
INTP1	PIF1	PMK1
INTTM50	TMIF50	TMMK50
INTTM51	TMIF51	TMMK51
INTTM52	TMIF52	TMMK52
INTKS	KSIF	KSMK
INTCSI10	CSIIF10	CSIMK10
INTTM80	TMIF80	TMMK80
INTTM81	TMIF81	TMMK81
INTWT	WTIF	WTMK
INTWTI	WTIIF	WTIMK

(1) Interrupt request flag registers (IF0, IF1)

The interrupt request flag is set to 1 when the corresponding interrupt request is generated or an instruction is executed. It is cleared to 0 when an instruction is executed upon acknowledgement of an interrupt request or upon $\overline{\text{RESET}}$ input.

IF0 and IF1 are set with a 1-bit or 8-bit memory manipulation instruction.

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

Figure 12-2. Interrupt Request Flag Register Format

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
IF0	CSIIF10	KSIF	TMIF52	TMIF51	TMIF50	PIF1	PIF0	WDTIF	FFE0H	00H	R/W

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	After reset	R/W
IF1	0	0	0	0	WTIIF	WTIF	TMIF81	TMIF80	FFE1H	00H	R/W

××IF×	Interrupt request flag
0	No interrupt request signal is generated
1	Interrupt request signal is generated; Interrupt request state

Cautions 1. WDTIF flag is R/W enabled only when the watchdog timer is used as an interval timer. If the watchdog timer mode 1 and 2 are used, set the WDTIF flag to 0.

2. Because port 2 has an alternate function as the external interrupt input, when the output level is changed by specifying the output mode of the port function, an interrupt request flag is set. Therefore, the interrupt mask flag should be set to 1 before using the output mode.

(2) Interrupt mask flag registers (MK0, MK1)

The interrupt mask flag is used to enable/disable the corresponding maskable interrupt service.

MK0 and MK1 are set with a 1-bit or 8-bit memory manipulation instruction.

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

Figure 12-3. Interrupt Mask Flag Register Format

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
MK0	CSIMK10	KSMK	TMMK52	TMMK51	TMMK50	PMK1	PMK0	WDTMK	FFE4H	FFH	R/W

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	After reset	R/W
MK1	1	1	1	1	WTIMK	WTMK	TMMK81	TMMK80	FFE5H	FFH	R/W

××MK×	Interrupt servicing control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Cautions 1. If the WDTMK flag is read when the watchdog timer is used in watchdog timer mode 1 and 2, its value becomes undefined.

2. Because port 2 has an alternate function as the external interrupt input, when the output level is changed by specifying the output mode of the port function, an interrupt request flag is set. Therefore, the interrupt mask flag should be set to 1 before using the output mode.

(3) External interrupt mode register 0 (INTM0)

This register is used to set the valid edge of INTP0 and INTP1.

INTM0 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears INTM0 to 00H.

Figure 12-4. External Interrupt Mode Register 0 Format

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 set bits 0, 1, 6, and 7 to 0.

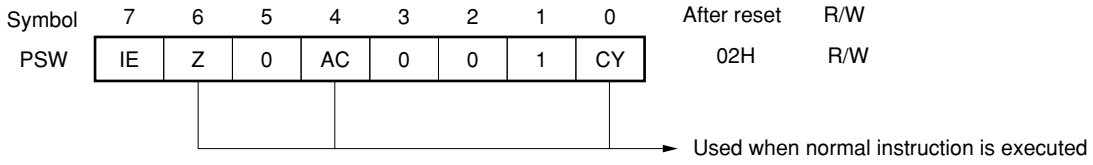
2. Before setting the INTM0 register, be sure to set the corresponding interrupt mask flag (××MK× = 1) to disable interrupts. After setting the INTM0 register, clear the interrupt request flag (××IF× = 0), then clear the interrupt mask flag (××MK× = 0), which will enable interrupts.

(4) Program status word (PSW)

The program status word is a register used to hold the instruction execution result and the current status for interrupt requests. The IE flag to set maskable interrupt enable/disable is mapped.

This register can be read/written in 8-bit units and can carry out operations using a bit manipulation and dedicated instructions (EI, DI). When a vectored interrupt request is acknowledged, PSW is automatically saved into a stack, and the IE flag is reset to 0. It is reset from the stack by the RETI and POP PSW instructions. $\overline{\text{RESET}}$ input sets PSW to 02H.

Figure 12-5. Program Status Word Configuration



IE	Interrupt acknowledge enable/disable
0	Disable
1	Enable

12.4 Interrupt Processing Operation

12.4.1 Non-maskable interrupt request acknowledgement operation

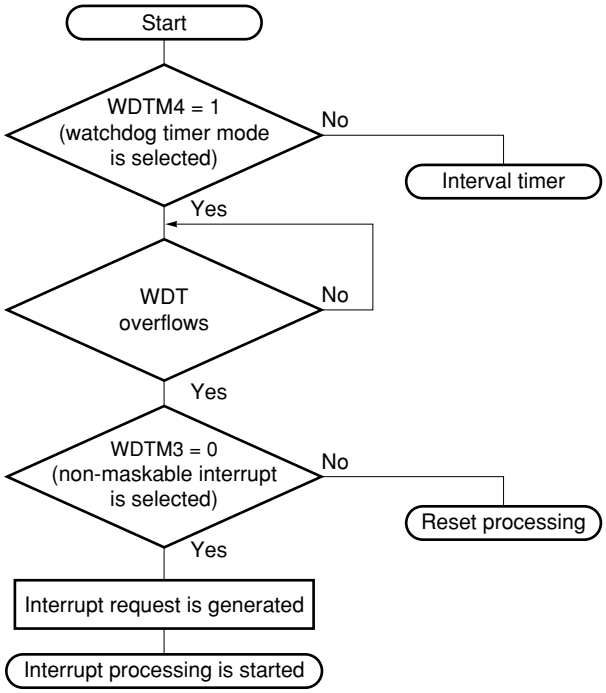
The non-maskable interrupt request is unconditionally acknowledged even when interrupts are disabled. It is not subject to interrupt priority control and takes precedence over all other interrupts.

When the non-maskable interrupt request is acknowledged, 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 12-6 shows the flowchart from non-maskable interrupt request generation to acknowledgement. Figure 12-7 shows the timing of non-maskable interrupt request acknowledgement. Figure 12-8 shows the acknowledgement operation if multiple non-maskable interrupts are generated.

Caution During a non-maskable interrupt service program execution, do not input another non-maskable interrupt request; if it is input, the service program will be interrupted and the new interrupt request will be acknowledged.

Figure 12-6. Flowchart from Non-Maskable Interrupt Request Generation to Acknowledgement



WDTM: Watchdog timer mode register
WDT: Watchdog timer

Figure 12-7. Timing of Non-Maskable Interrupt Request Acknowledgement

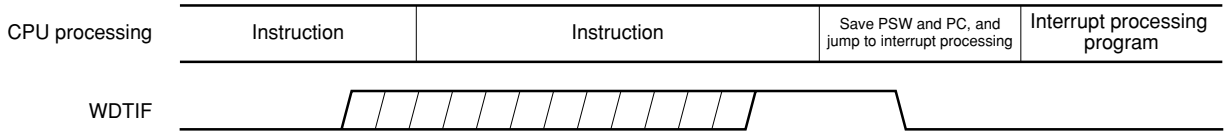
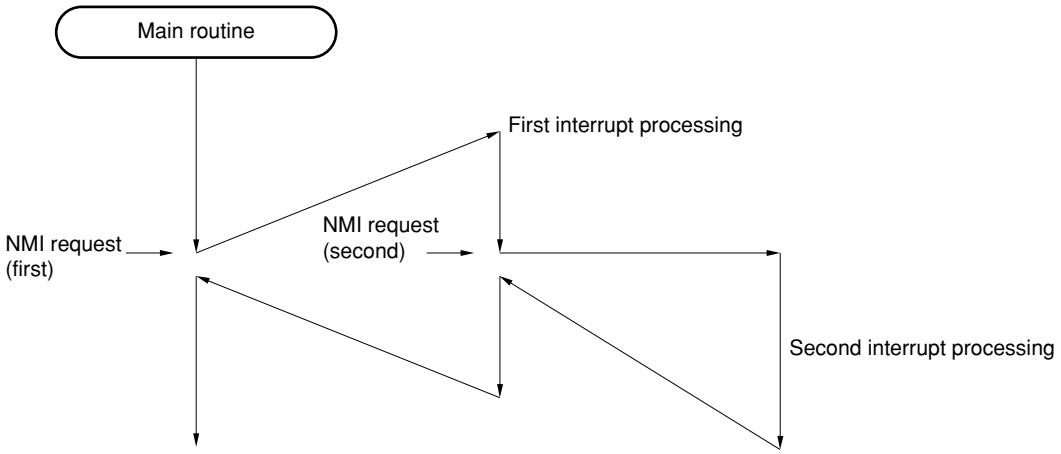


Figure 12-8. Acknowledging Non-Maskable Interrupt Request



12.4.2 Maskable interrupt request acknowledgement operation

A maskable interrupt request 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 request is acknowledged in the interrupt enabled status (when the IE flag is set to 1).

The time required to start the interrupt processing after a maskable interrupt request has been generated is shown in Table 12-3.

Refer to Figures 12-10 and 12-11 for the interrupt request acknowledgement timing.

Table 12-3. Time from Generation of Maskable Interrupt Request to Processing

Minimum Time	Maximum Time[Note]
9 clocks	19 clocks

Note The wait time is maximum when an interrupt request is generated immediately before BT and BF instruction.

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.

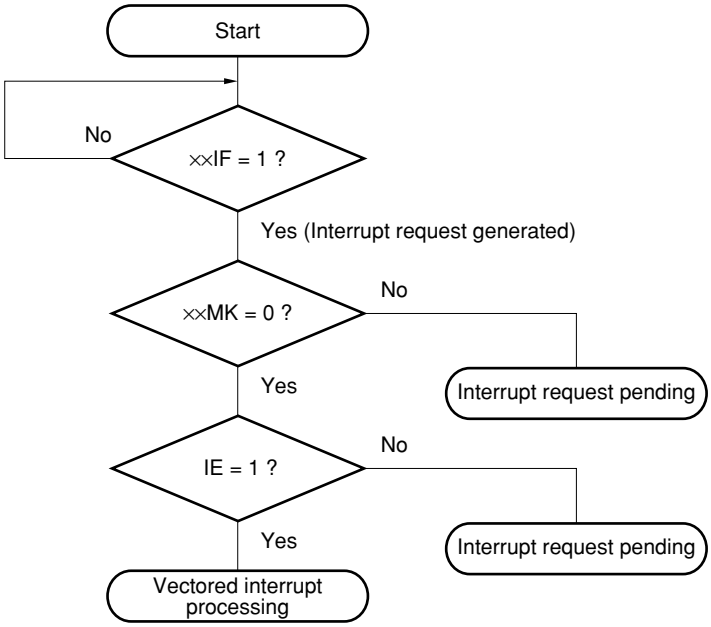
A pending interrupt is acknowledged when the status where it can be acknowledged is set.

Figure 12-9 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, and the data in the vector table determined for each interrupt request is loaded to the PC, and execution branches.

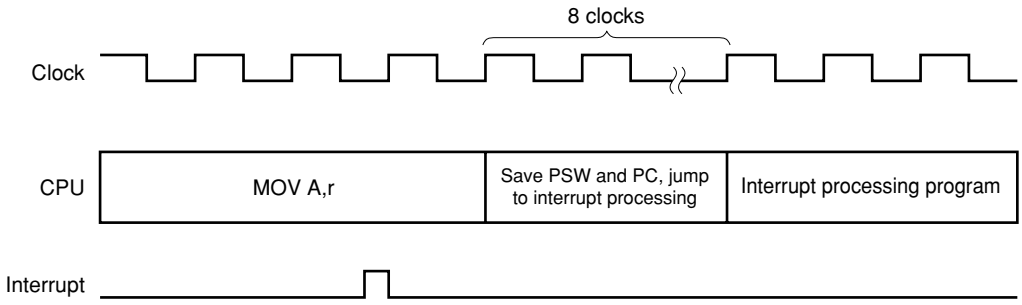
To return from interrupt processing, use the RETI instruction.

Figure 12-9. Interrupt Acknowledgement Program Algorithm



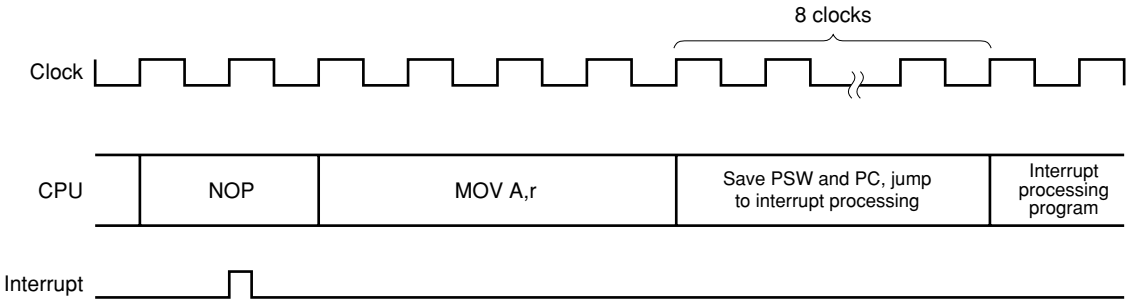
××IF: Interrupt request flag
××MK: Interrupt mask flag
IE: Flag to control maskable interrupt request acknowledgement (1 = enable, 0 = disable)

Figure 12-10. Interrupt Request Acknowledgement Timing (Example of MOV A,r)



If an interrupt request flag (××IF) is set before an instruction clock n (n = 4 to 10) under execution becomes n–1, the interrupt is acknowledged after the instruction under execution is complete. Figure 12-10 shows an example of the interrupt request acknowledgement timing for an 8-bit data transfer instruction MOV A,r. Since this instruction is executed for 4 clocks, if an interrupt occurs for 3 clocks after the execution starts, the interrupt acknowledgement processing is performed after the MOV A,r instruction is completed.

Figure 12-11. Interrupt Request Acknowledgement Timing (When Interrupt Request Flag Is Generated at the Last Clock During Instruction Execution)



If an interrupt request flag (××IF) is set at the last clock of the instruction, the interrupt acknowledgement processing starts after the next instruction is executed.

Figure 12-11 shows an example of the interrupt acknowledgement 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 acknowledgement processing is performed.

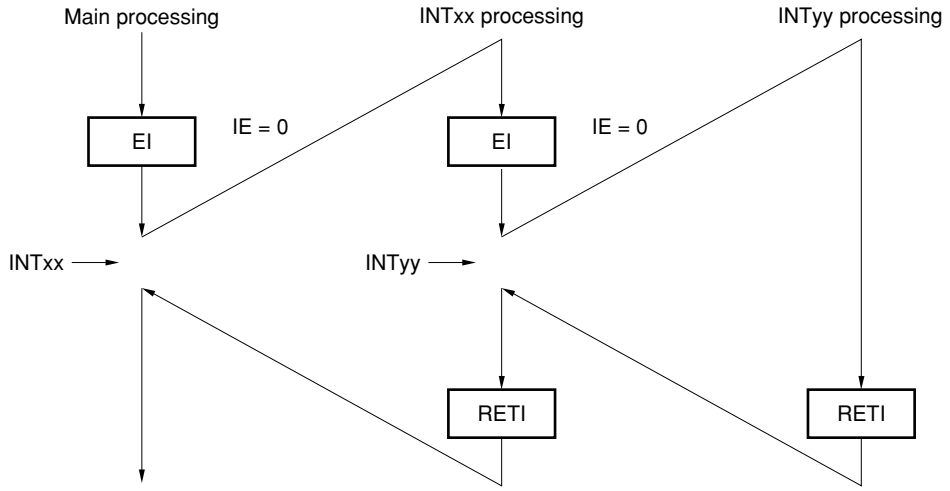
Caution Interrupt requests are kept pending while the interrupt request flag register (IF0, IF1) or the interrupt mask flag register (MK0, MK1) is being accessed.

12.4.3 Multiple interrupt processing

Multiple interrupt processing in which another interrupt is acknowledged while an interrupt is being processed can be processed by priority. When the priority is controlled by the default priority and two or more interrupts are generated at once, interrupt processing is performed according to the priority assigned to each interrupt request in advance (refer to **Table 12-1**).

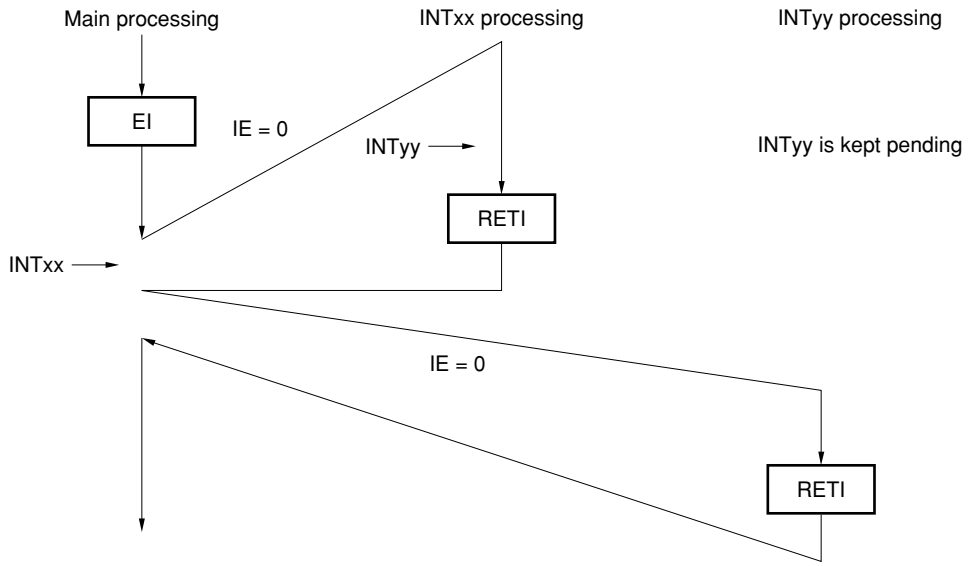
Figure 12-12. Example of Multiple Interrupts

Example 1. Multiple interrupts are acknowledged



During interrupt INTxx servicing, interrupt request INTyy is acknowledged, and a multiple interrupt is generated. An EI instruction is issued before each interrupt request acknowledgement, and the interrupt request acknowledgement enable state is set.

Example 2. A multiple interrupt is not generated because interrupts are not enabled



Because interrupts are not enabled in interrupt INTxx servicing (an EI instruction is not issued), interrupt request INTyy is not acknowledged, and a multiple interrupt is not generated. The INTyy request is kept pending and acknowledged after the INTxx processing is performed.

IE = 0: Interrupt request acknowledgement disabled

12.4.4 Interrupt request pending

Some instructions may hold pending the acknowledgement of an instruction request until the completion of the execution of the next instruction even if the interrupt request (maskable interrupt, non-maskable interrupt, and external interrupt) is generated during the execution. The following shows such instructions.

- Manipulation instruction for the interrupt request flag register (IF0, IF1)
- Manipulation instruction for the interrupt mask flag register (MK0, MK1)

CHAPTER 13 STANDBY FUNCTION

13.1 Standby Function and Configuration

13.1.1 Standby function

The standby function is 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. The HALT mode stops the operation clock of the CPU. The system clock oscillation circuit continues oscillating. This mode does not reduce the power consumption as much as the 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. The STOP mode stops the main system clock oscillation circuit and stops the entire system. The power consumption of the CPU can be substantially reduced in this mode.

The low voltage (V_{DD} = 1.8 V) of the data memory can be retained. Therefore, this mode is useful for retaining the contents of the data memory at an extremely low current.

The STOP mode can be released by an interrupt request, so that this mode can be used for intermittent operation. However, some time is required until the system clock oscillation circuit stabilizes after the STOP mode has been released. If processing must be resumed immediately by using an interrupt request, therefore, use the HALT mode.

In both modes, the previous contents of the registers, flags, and data memory before setting the 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 the STOP mode, be sure to stop the operations of the peripheral hardware, and then execute the STOP instruction.

13.1.2 Register controlling standby function

The wait time after the STOP mode is released upon interrupt request until the oscillation stabilizes is controlled with the oscillation stabilization time select register (OSTS).

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

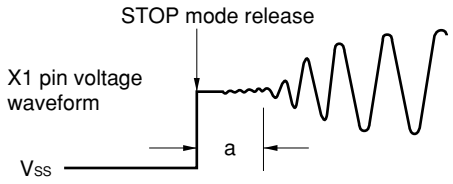
$\overline{\text{RESET}}$ input sets OSTS to 04H. However, the oscillation stabilization time after $\overline{\text{RESET}}$ input is 2^{15}/fx, instead of 2^{17}/fx.

Figure 13-1. Oscillation Stabilization Time Select Register Format

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 (819 μs)
0	1	0	2^{15}/fx (6.55 ms)
1	0	0	2^{17}/fx (26.2 ms)
Other than above			Setting prohibited

Caution The wait time after the 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: Main system clock oscillation frequency

2. Values in parentheses apply to operation with fx = 5.0 MHz.

13.2 Operation of Standby Function

13.2.1 HALT mode

(1) HALT mode

The HALT mode is set by executing the HALT instruction.

The operation status in the HALT mode is shown in the following table.

Table 13-1. HALT Mode Operating Status

Item	Operating Status in HALT Mode During Main System Clock Operation		Operating Status in HALT Mode During Subsystem Clock Operation	
	Subsystem Clock Operating	Subsystem Clock Stopped	Main System Clock Operating	Main System Clock Stopped
Main system clock	Oscillation enabled			Oscillation stopped
CPU	Operation stopped			
Port (output latch)	Retains the status before setting the HALT mode			
8-bit remote control timer 50	Operable			Operation stopped
8-bit timer 80	Operable			Operation stopped
8-bit timer 81	Operable			Operation stopped
Watch timer	Operable	Operable[Note 1]	Operable	Operable[Note 2]
Watchdog timer	Operable		Operation stopped	
Serial interface 10	Operable			Operable[Note 3]
VFD controller/driver	Operation stopped (retains output data)			
External interrupt	Operable[Note 4]			

Notes 1. Operable when the main system clock is selected.

2. Operable when the subsystem clock is selected.

3. Operable only when the external clock is selected.

4. Maskable interrupt that is not masked.

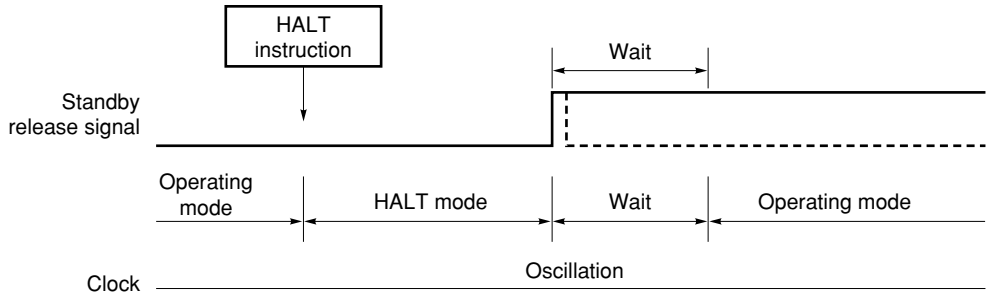
(2) Releasing HALT mode

The HALT mode can be released by the following three types of sources.

(a) Releasing by unmasked interrupt request

The HALT mode is released by an unmasked interrupt request. In this case, if the interrupt request is able to be acknowledged, vectored interrupt processing is performed. If the interrupt is disabled, the instruction at the next address is executed.

Figure 13-2. Releasing HALT Mode by Interrupt



Remarks 1. The broken lines indicate the case where the interrupt request that has released the standby mode is acknowledged.

2. The wait time is as follows:

- When vectored interrupt processing is performed: 9 to 10 clocks
- When vectored interrupt processing is not performed: 1 to 2 clocks

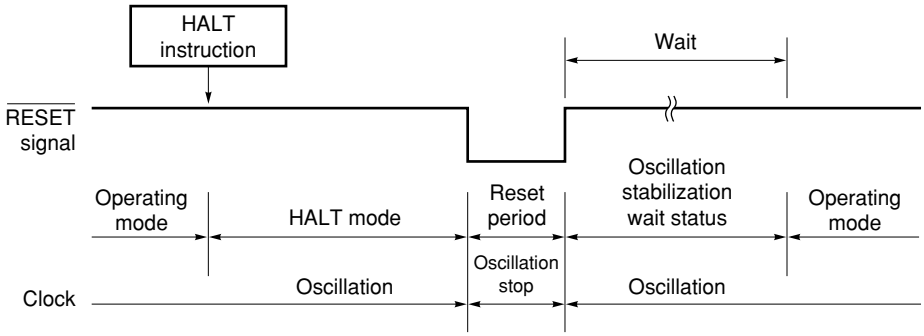
(b) Releasing by non-maskable interrupt request

The HALT mode is released regardless of whether the interrupt is enabled or disabled, and vectored interrupt processing is performed.

(c) Releasing by $\overline{\text{RESET}}$ input

When the 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 13-3. Releasing HALT Mode by $\overline{\text{RESET}}$ Input



Remark fx: Main system clock oscillation frequency

Table 13-2. Operation After Release of HALT Mode

Releasing Source	MK××	IE	Operation
Maskable interrupt request	0	0	Executes next address instruction
	0	1	Executes interrupt processing
	1	×	Retains HALT mode
Non-maskable interrupt request	—	×	Executes interrupt processing
$\overline{\text{RESET}}$ input	—	—	Reset processing

×: don't care

13.2.2 STOP mode

(1) Setting and operation status of STOP mode

The STOP mode is set by executing the STOP instruction.

Cautions 1. When the STOP mode is set, the X2 or CL2 pin is internally pulled up to V_{DD} to suppress the current leakage of the oscillation circuit block. Therefore, do not use the STOP mode in a system where the external clock is used as the system clock.

2. Because the standby mode can be released by an interrupt request signal, the 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 the STOP mode is set, therefore, the HALT mode is set immediately after the STOP instruction has been executed, the wait time set by the oscillation stabilization time select register (OSTS) elapses, and then an operation mode is set.

The operation status in the STOP mode is shown in the following table.

Table 13-3. STOP Mode Operating Status

Item	Operating Status in STOP Mode During Main System Clock Operation	
	Subsystem Clock Operating	Subsystem Clock Stopped
Main system clock	Oscillation stopped	
CPU	Operation stopped	
Port (output latch)	Retains the status before setting the STOP mode	
8-bit remote control timer 50	Operation stopped	
8-bit timer 80	Operation stopped	
8-bit timer 81	Operation stopped	
Watch timer	Operable[Note 1]	Operation stopped
Watchdog timer	Operation stopped	
Serial interface 10	Operable[Note 2]	
VFD controller/driver	Operation stopped (retains output data)	
External interrupt	Operable[Note 3]	

Notes 1. Operable when the subsystem clock is selected.

2. Operable only when the external clock is selected.

3. Maskable interrupt that is not masked.

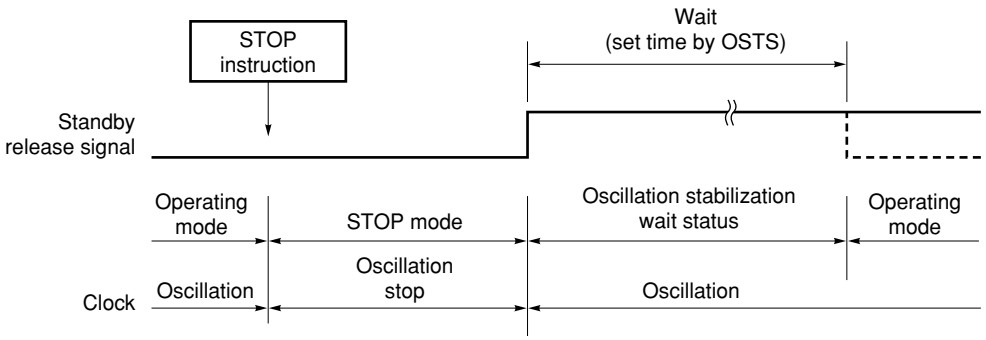
(2) Releasing STOP mode

The STOP mode can be released by the following two types of sources.

(a) Releasing by unmasked interrupt request

The STOP mode can be released by an unmasked interrupt request. In this case, if the interrupt is able to be acknowledged, vectored interrupt processing is performed, after the oscillation stabilization time has elapsed. If the interrupt is disabled, the instruction at the next address is executed.

Figure 13-4. Releasing STOP Mode by Interrupt

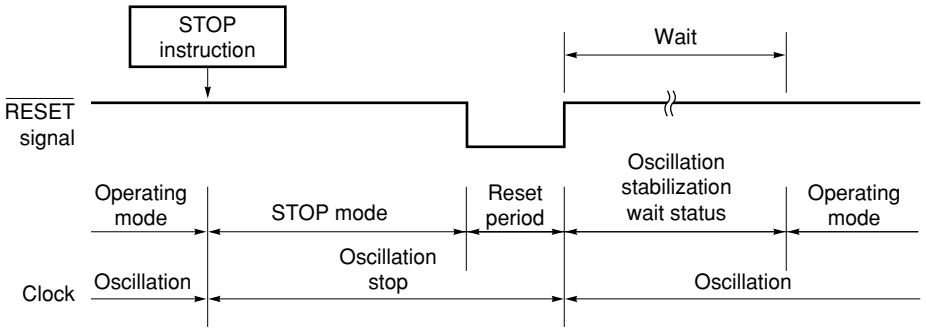


Remark The broken lines indicate the case where the interrupt request that has released the standby mode is acknowledged.

(b) Releasing by $\overline{\text{RESET}}$ input

When the STOP mode is released by the $\overline{\text{RESET}}$ signal, the reset operation is performed after the oscillation stabilization time has elapsed.

Figure 13-5. Releasing STOP Mode by $\overline{\text{RESET}}$ Input



Remark fx: Main system clock oscillation frequency

Table 13-4. Operation After Release of STOP Mode

Releasing Source	MK××	IE	Operation
Maskable interrupt request	0	0	Executes next address instruction
	0	1	Executes interrupt processing
	1	×	Retains STOP mode
$\overline{\text{RESET}}$ input	—	—	Reset processing

×: don't care

CHAPTER 14 RESET FUNCTION

The following two operations are available to generate reset signals.

(1) External reset input with $\overline{\text{RESET}}$ pin
(2) Internal reset by program runaway time detection with watchdog timer

External and internal reset have no functional differences. In both cases, program execution starts at the addresses 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 each hardware item is set to the status shown in Table 14-1. Each pin has a high impedance during reset input or during the oscillation stabilization time just after reset clear.

When a high level is input to the $\overline{\text{RESET}}$ pin, the reset is cleared and program execution is started after the oscillation stabilization time has elapsed. The reset applied by the watchdog timer overflow is automatically cleared after reset, and program execution is started after the oscillation stabilization time has elapsed (see **Figures 14-2** to **14-4**).

Cautions 1. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin.
2. When the STOP mode is cleared by reset, the STOP mode contents are held during reset input. However, the port pins become high impedance.

Figure 14-1. Block Diagram of Reset Function

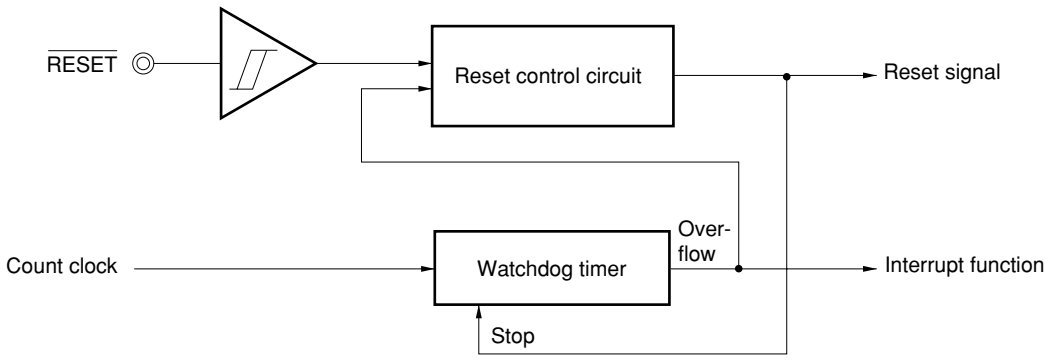


Figure 14-2. Reset Timing by $\overline{\text{RESET}}$ Input

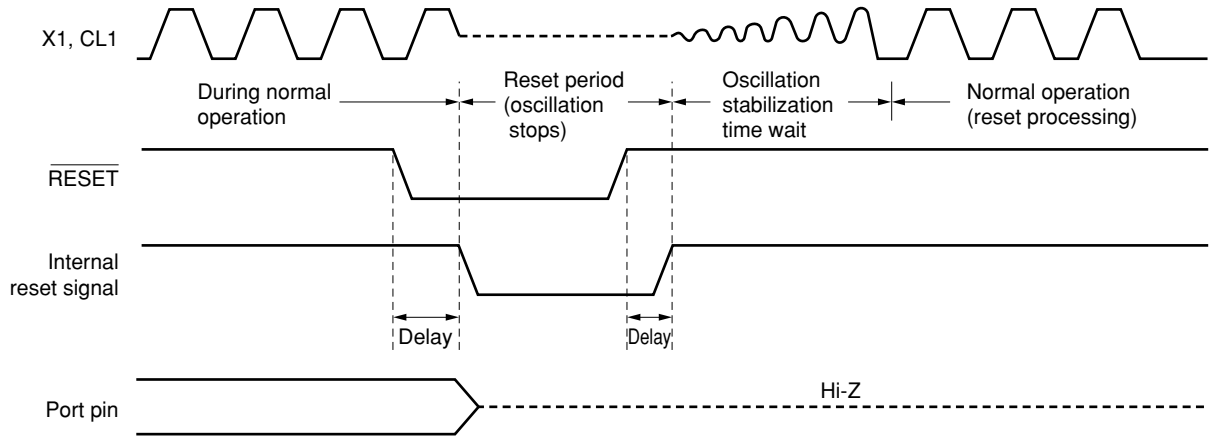


Figure 14-3. Reset Timing by Overflow in Watchdog Timer

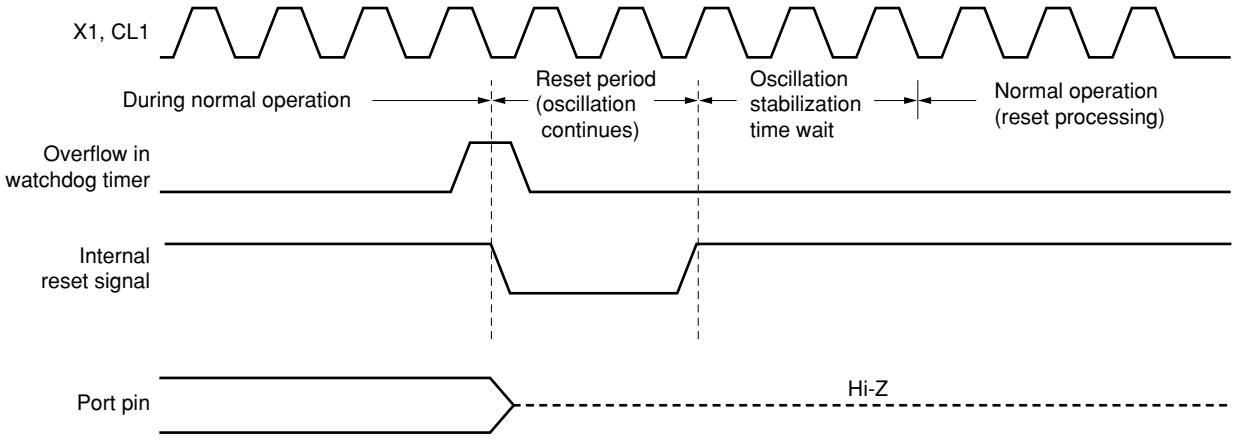


Figure 14-4. Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode

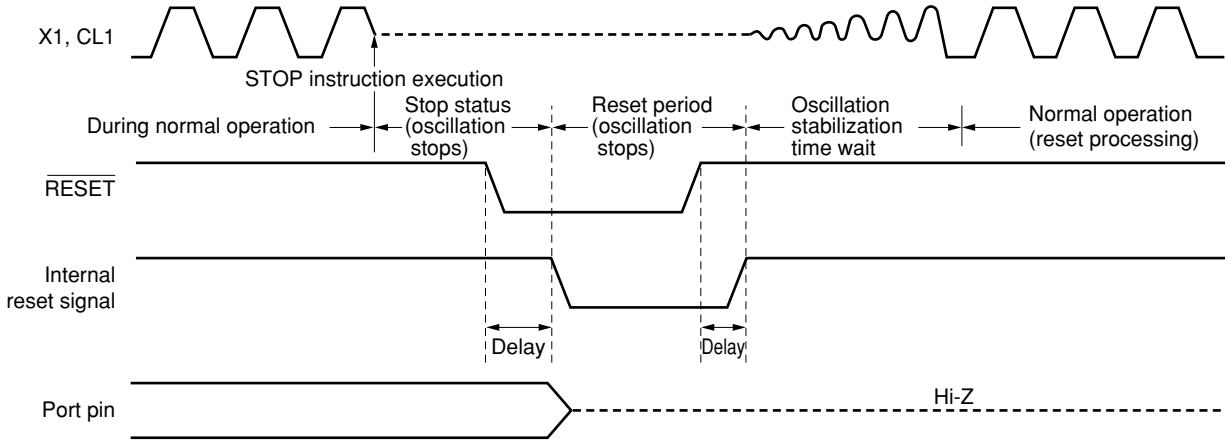


Table 14-1. Hardware Status After Reset

Hardware		Status after Reset
Program counter (PC)[Note 1]		The contents of reset vector tables (0000H and 0001H) are set.
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined[Note 2]
	General-purpose register	Undefined[Note 2]
Port (P0 to P2, P8, P9) (Output latch)		00H
Port mode register (PM0 to PM2)		FFH
Pull-up resistor option register 0 (PU0)		00H
Pull-up resistor option register B2 (PUB2)		00H
Processor clock control register (PCC)		02H
Oscillation stabilization time select register (OSTS)		04H
8-bit remote control timer	Control register (TMC50)	00H
	Capture register (CP50, CP51)	00H
8-bit timer	Timer counter (TM80, TM81)	00H
	Compare register (CR80, CR81)	00H
	Mode control register (TMC80, TMC81)	00H
Watch timer	Mode control register (WTM)	00H
Watchdog timer	Timer clock select register (WDCS)	00H
	Mode register (WDTM)	00H
Serial interface 10	Mode register (CSIM10)	00H
	Transmit/receive shift register 10 (SIO10)	Undefined
VFD controller/driver	Display mode register 0 (DSPM0)	10H
	Display mode register 1 (DSPM1)	01H
	Display mode register 2 (DSPM2)	00H
Interrupt	Request flag register (IF0, IF1)	00H
	Mask flag register (MK0, MK1)	FFH
	External Interrupt mode register (INTM0)	00H

Notes 1. During reset input and oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined.
All other hardware remains unchanged after reset.

2. If the reset signal is input in the standby mode, the status before reset is retained even after reset.

CHAPTER 15 μPD78F9872

The μPD78F9872 is a version with flash memory instead of the internal ROM of the mask ROM version in the μPD789871 Subseries. The differences between the flash memory and the mask ROM versions are shown in Table 15-1.

Table 15-1. Differences Between Flash Memory and Mask ROM Versions

Item		Flash Memory	Mask ROM	
		μPD78F9872	μPD789870	μPD789871
Internal memory	ROM	16 KB (flash memory)	4 KB	8 KB
	High-speed RAM	512 bytes		
VFD display RAM		96 bytes		
IC pin		Not provided	Provided	
V_{PP} pin		Provided	Not provided	
Pull-down resister of FIP0 to FIP8		Provided		
On-chip pull-down resister of P80/FIP24 to P87/FIP17 (connected to V_{LOAD} using mask option)		Not provided	Provided	
Pull-down resister of P90/FIP16 to P97/FIP9		Not provided	Provided	
Electrical specifications		There are differences between flash memory versions and mask ROM versions.		

Caution There are differences in noise immunity and noise radiation between the flash memory versions and mask ROM versions. When pre-producing an application set with the flash memory version and then mass-producing it with the mask ROM version, be sure to conduct sufficient evaluations for the commercial samples (not engineering samples) of the mask ROM versions.

15.1 Flash Memory Programming

The on-chip program memory in the μPD78F9872 is flash memory.

The flash memory can be written with the μPD78F9116A mounted on the target system (on-board). Connect the dedicated flash programmer (Flashpro III (part number: FL-PR3, PG-FP3)) to the host machine and target system to write the flash memory.

Remark FL-PR3 is made by Naito Densei Machida Mfg. Co., Ltd.

15.1.1 Selecting communication mode

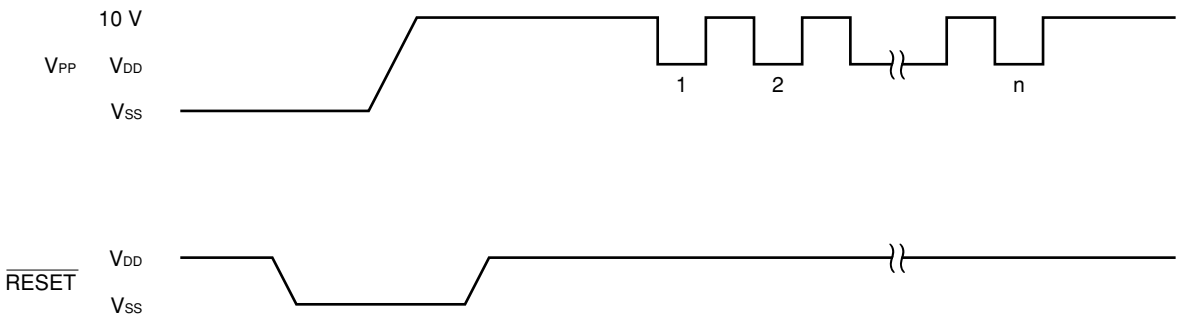
The flash memory is written by using Flashpro III and by means of serial communication. Select a communication mode from those listed in Table 15-2. To select a communication mode, the format shown in Figure 15-1 is used. Each communication mode is selected by the number of V_{PP} pulses shown in Table 15-2.

Table 15-2. Communication Mode

Communication Mode	Pins Used	Number of V_{PP} Pulses
3-wire serial I/O	$\overline{SCK10}$/P20 SO10/P21 SI10/P22	0

Caution Be sure to select a communication mode based on the V_{PP} pulse number shown in Table 15-2.

Figure 15-1. Communication Mode Selection Format



15.1.2 Function of flash memory programming

By transmitting/receiving commands and data in the selected communication mode, operations such as writing to the flash memory are performed. Table 15-3 shows the major functions of flash memory programming.

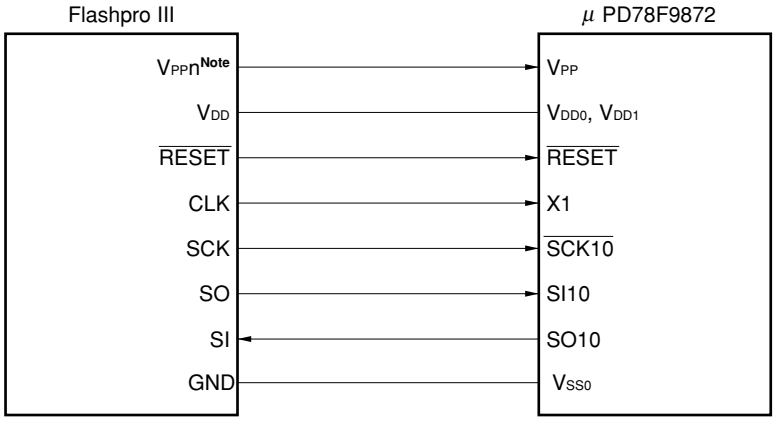
Table 15-3. Functions of Flash Memory Programming

Function	Description
Batch erase	Erases all contents of memory
Batch blank check	Checks erased state of entire memory
Data write	Write to flash memory based on write start address and number of data written (number of bytes)
Batch verify	Compares all contents of memory with input data

15.1.3 Flashpro III connection example

A connection example between the Flashpro III and the μPD78F9872 is shown in Figure 15-2.

Figure 15-2. Flashpro III Connection in 3-Wire Serial I/O Mode



Note n = 1, 2

15.1.4 Example of settings for Flashpro III (PG-FP3)

Make the following setting when writing to flash memory using Flashpro III (PG-FP3).

<1> Load the parameter file.
<2> Select serial mode and serial clock using the type command.
<3> An example of the settings for the PG-FP3 is shown below.

Table 15-4. Example of Settings for PG-FP3

Communication Mode	Example of Setting for PG-FP3		Number of V_{PP} Pulses[Note]
3-wire serial I/O	COMM PORT	SIO-ch0	0
	CPU CLK	On Target Board	
		In Flashpro	
	On Target Board	4.1943 MHz	
	SIO CLK	1.0 MHz	
	In Flashpro	4.0 MHz	
	SIO CLK	1.0 MHz	

Note The number of V_{PP} pulses supplied from Flashpro III when serial communication is initialized. The pins to be used for communication are determined according to the number of these pulses.

Remark COMM PORT: Selection of serial port
SIO CLK: Selection of serial clock frequency
CPU CLK: Selection of source of CPU clock to be input

- Mask option of FIP17/P87 to FIP24/P80
  An on-chip pull-down resistor for V_{LOAD} can be specified by a mask option.

Caution The flash memory versions do not provide the on-chip pull-down resistor of P80 to P87.

CHAPTER 17 INSTRUCTION SET

This chapter lists the instruction set of the μPD789871 Subseries. For the details of the operation and machine language (instruction code) of each instruction, refer to **78K/0S Series User's Manual Instruction (U11047E)**.

17.1 Operation

17.1.1 Operand identifiers and description methods

Operands are described in "Operand" column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for detail). When there are two or more description methods, select one of them. Alphabetic letters in capitals and symbols, #, !, $, and [ ] are key words 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 parenthesis in the table below, R0, R1, R2, etc.) can be used for description.

Table 17-1. 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 Refer to **Table 3-3 Special Function Register List** for symbols of special function registers.

17.1.2 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 parenthesis
$\times_H$, $\times_L$: Higher 8 bits and lower 8 bits of 16-bit register
$\wedge$: Logical product (AND)
$\vee$: Logical sum (OR)
$\veebar$: Exclusive logical sum (exclusive OR)
$\overline{\ }$: Inverted data
addr16: 16-bit immediate data or label
jdisp8: Signed 8-bit data (displacement value)

17.1.3 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 restored

17.2 Operation List

Mnemonic	Operands	Byte	Clock	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	Byte	Clock	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	Byte	Clock	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	Byte	Clock	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	Byte	Clock	Operation	Flag
					Z AC CY
CALL	!addr16	3	6	(SP – 1) ← (PC + 3)H, (SP – 2) ← (PC + 3)L, PC ← addr16, SP ← SP – 2	
CALLT	[addr5]	1	8	(SP – 1) ← (PC + 1)H, (SP – 2) ← (PC + 1)L, PCH ← (00000000, addr5 + 1), PCL ← (00000000, addr5), SP ← SP – 2	
RET		1	6	PCH ← (SP + 1), PCL ← (SP), SP ← SP + 2	
RETI		1	8	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3, NMIS ← 0	R R R
PUSH	PSW	1	2	(SP – 1) ← PSW, SP ← SP – 1	
	rp	1	4	(SP – 1) ← rpH, (SP – 2) ← rpL, SP ← SP – 2	
POP	PSW	1	4	PSW ← (SP), SP ← SP + 1	R R R
	rp	1	6	rpH ← (SP + 1), rpL ← (SP), SP ← SP + 2	
MOVW	SP, AX	2	8	SP ← AX	
	AX, SP	2	6	AX ← SP	
BR	!addr16	3	6	PC ← addr16	
	$addr16	2	6	PC ← PC + 2 + jdisp8	
	AX	1	6	PCH ← A, PCL ← X	
BC	$addr16	2	6	PC ← PC + 2 + jdisp8 if CY = 1	
BNC	$addr16	2	6	PC ← PC + 2 + jdisp8 if CY = 0	
BZ	$saddr16	2	6	PC ← PC + 2 + jdisp8 if Z = 1	
BNZ	$saddr16	2	6	PC ← PC + 2 + jdisp8 if Z = 0	
BT	saddr.bit,$addr16	4	10	PC ← PC + 4 + jdisp8 if (saddr.bit) = 1	
	sfr.bit,$addr16	4	10	PC ← PC + 4 + jdisp8 if sfr.bit = 1	
	A.bit,$addr16	3	8	PC ← PC + 3 + jdisp8 if A.bit = 1	
	PSW.bit,$addr16	4	10	PC ← PC + 4 + jdisp8 if PSW.bit = 1	
BF	saddr.bit,$addr16	4	10	PC ← PC + 4 + jdisp8 if (saddr.bit) = 0	
	sfr.bit,$addr16	4	10	PC ← PC + 4 + jdisp8 if sfr.bit = 0	
	A.bit,$addr16	3	8	PC ← PC + 3 + jdisp8 if A.bit = 0	
	PSW.bit,$addr16	4	10	PC ← PC + 4 + jdisp8 if PSW.bit = 0	
DBNZ	B,$addr16	2	6	B ← B – 1, then PC ← PC + 2 + jdisp8 if B ≠ 0	
	C,$addr16	2	6	C ← C – 1, then PC ← PC + 2 + jdisp8 if C ≠ 0	
	saddr,$addr16	3	8	(saddr) ← (saddr) – 1, then PC ← PC + 3 + jdisp8 if (saddr) ≠ 0	
NOP		1	2	No Operation	
EI		3	6	IE ← 1 (Enable Interrupt)	
DI		3	6	IE ← 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).

17.3 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

APPENDIX A DEVELOPMENT TOOLS

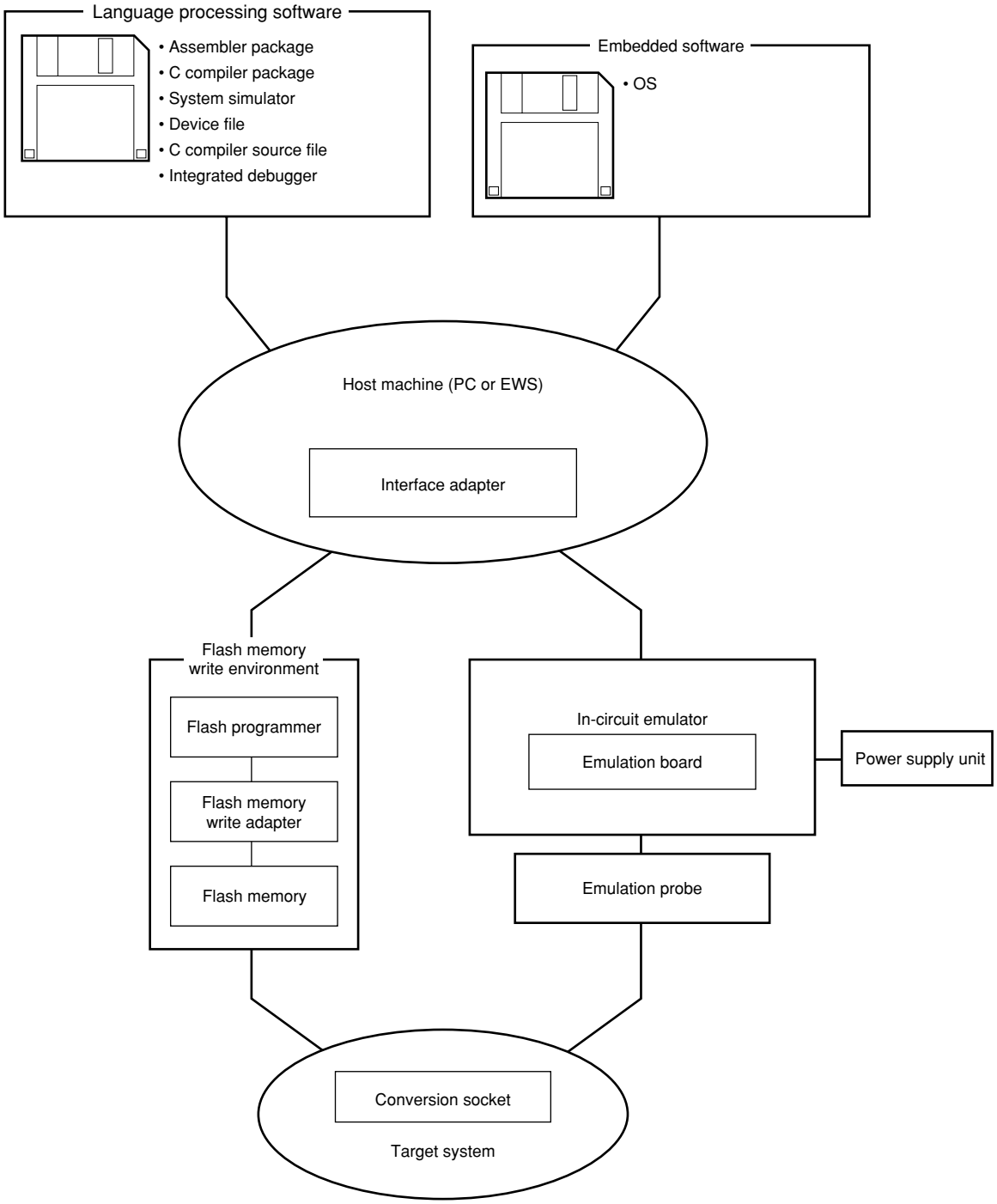
The following development tools are available for the development of systems that employ the μPD789871 Subseries.

Figure A-1 shows the development tool configuration.

- **Support of the PC98-NX Series**
  Unless otherwise specified, the μPD789104A/114A/124A/134A Subseries supported by IBM PC/AT™ and compatibles can be used for the PC98-NX Series. When using the PC98-NX Series, refer to the descriptions of the IBM PC/AT and compatibles.

- **Windows**
  Unless otherwise specified, "Windows" indicates the following OSs.
  Windows 3.1
  Windows 95
  Windows NT™ Ver. 4.0

Figure A-1. Development Tool Configuration



A.1 Language Processing Software

RA78K0S Assembler package	A program that converts a program written in mnemonic into object codes that can be executed by microcontrollers. In addition, automatic functions to generate symbol tables and optimize branch instructions are also provided. Used in combination with a device file (DF789872) (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 assembler package).
	Part number: μS××××RA78K0S
CC78K0S C compiler package	A program that converts a program written in C language into object codes that can be executed by microcontrollers. Used in combination with an assembler package (RA78K0S) and device file (DF789872) (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
DF789872[Note] Device file	File containing the information inherent to the device. Used in combination with RA78K0S, CC78K0S, and SM78K0S (all sold separately).
	Part number: μS××××DF789872
CC78K0S-L C compiler source file	Source file of functions for generating the object library included in the C compiler package. Necessary for changing the object library included in the C compiler package according to 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

Note DF789872 is a common file that can be used with RA78K0S, CC78K0S, and SM78K0S.

Remark ×××× in the part number differs depending on the host machine and operating system to be used.

μS××××RA78K0S
μS××××CC78K0S
μS××××DF789872
μS××××CC78K0S-L

××××	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows[Note]	3.5" 2HD FD
AB13	IBM PC/AT and compatibles	Japanese Windows[Note]	3.5" 2HC FD
3P16	HP9000 Series 700™	HP-UX™ (Rel.10.10)	DAT (DDS)
3K13	SPARCstation™	SunOS™ (Rel.4.1.1)	3.5" 2HC FD
3K15		Solaris™ (Rel.2.5.1)	1/4" CGMT
3R13	NEWS™ (RISC)	NEWS-OS™ (Rel.6.1)	3.5" 2HC FD

Note Also operates in the DOS environment

A.2 Flash Memory Writing Tools

Flashpro III (Part No. FL-PR3, PG-FP3) Flash programmer	Dedicated flash programmer for microcontrollers incorporating flash memory
FA-52GB Flash memory writing adapter	Adapter for writing to flash memory and connected to Flashpro III.

Remark The FL-PR3 and FA-52GB are products made by Naito Densei Machida Mfg. Co., Ltd. (TEL +81-44-822-3813).

A.3 Debugging Tools

A.3.1 Hardware

IE-78K0S-NS In-circuit emulator	In-circuit emulator for debugging the hardware and software of an application system using the 78K/0S Series. Supports an integrated debugger (ID78K0S-NS). Used in combination with an AC adapter, emulation probe, and interface adapter for connecting the host machine.
IE-70000-MC-PS-B AC adapter	Adapter for supplying power from an AC 100 to 240 V outlet.
IE-70000-98-IF-C Interface adapter	Adapter necessary when using a PC-9800 series PC (except notebook type) as the host machine of the IE-78K0S-NS (C bus supported)
IE-70000-CD-IF-A PC card interface	PC card and interface cable necessary when using a notebook PC as the host machine of the IE-78K0S-NS (PCMCIA socket supported)
IE-70000-PC-IF-C Interface adapter	Interface adapter necessary when using an IBM PC/AT or compatible as the host machine of the IE-78K0S-NS (ISA bus supported)
IE-70000-PCI-IF Interface adapter	Adapter necessary when using a personal computer incorporating the PCI bus as the host machine of the IE-78K0S-NS
IE-789872-NS-EM1 Emulation board	Board for emulating the peripheral hardware inherent to the device. Used in combination with an in-circuit emulator.
NP-52GB Emulation probe	Probe for connecting the in-circuit emulator and target system. This is for a 30-pin plastic SSOP (MC-5A4 type).
NGS-30 Conversion socket	Conversion socket to connect the NP-36GS and a target system board on which a 30-pin plastic SSOP (MC-5A4 type) can be mounted.

Remark The NP-52GB, and NGS-30 are products made by Naito Densei Machida Mfg. Co., Ltd. For details of these products, contact Naito Densei Machida Mfg. Co., Ltd. (TEL +81-44-822-3813).

A.3.2 Software

ID78K0S-NS Integrated debugger (Supports in-circuit emulator IE-78K0S-NS)	Control program for debugging the 78K/0S Series. This program provides a graphical user interface. It runs on Windows for personal computer users and on OSF/Motif™ for engineering work station users, and has visual designs and operationability that comply with these operating systems. In addition, it has a powerful debug function that supports C language. Therefore, trace results can be displayed at a C language level by the window integration function that links the source program, disassembled display, and memory display, to the trace result. This software also allows users to add other function extension modules such as a task debugger and system performance analyzer to improve the debug efficiency for programs using a real-time operating system. Used in combination with a device file (DF789872) (sold separately).
	Part number: μS××××ID78K0S-NS

Remark ×××× in the part number differs depending on the host machine and operating system to be used.

μS××××ID78K0S-NS

××××	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows[Note]	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows[Note]	3.5" 2HC FD

Note Also operates in the DOS environment.

SM78K0S System simulator	Debugs the program at C source level or assembler level while simulating operation of the target system on the host machine. SM78K0S runs in Windows. By using SM78K0S, the logic and performance of an application can be verified independently of hardware development even when the in-circuit emulator is not used. This enhances development efficiency and improves software quality. Used in combination with a device file (DF789872) (sold separately).
	Part number: μS××××SM78K0S
DF789872[Note] Device file	File containing the information inherent to the device. Used in combination with the RA78K0S, CC78K0S, and SM78K0S (all sold separately).
	Part number: μS××××DF789872

Note DF789872 is a common file that can be used with the RA78K0S, CC78K0S, and SM78K0S.

Remark ×××× in the part number differs depending on the host machine and operating system to be used.

μS××××SM78K0S

××××	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows[Note]	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows[Note]	3.5" 2HC FD

Note Also operates in the DOS environment.

APPENDIX B EMBEDDED SOFTWARE

The following embedded software products are available for efficient program development and maintenance of the μPD789871 Subseries.

MX78K0S OS	MX78K0S is a subset OS that is based on the μITRON specification. Supplied with the MX78K0S nucleus. The MX78K0S OS controls tasks, events, and time. In task control, the MX78K0S OS controls task execution order, and then perform the switching process to a task to be executed. <Caution when used in a PC environment> The MX78K0S is a DOS-based application. Use this software in the DOS pane when running it on Windows.
	Part number: μS××××MX78K0S

Remark ×××× in the part number differ depending on the host machine and OS used.

μS××××MX78K0S

××××	Host Machine	OS	Supply Media
AA13	PC-9800 Series	Japanese Windows[Note]	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows[Note]	3.5" 2HD FD
BB13		English Windows[Note]	

Note Can also be operated in the DOS environment.

APPENDIX C REGISTER INDEX

C.1 Register Name Index (Alphabetic Order)

[V]

[W]

C.2 Register Symbol Index (Alphabetic Order)

NEC

Facsimile Message

From:

Name

Company

Tel. FAX

Address

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

Thank you for your kind support.

North America
NEC Electronics Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288
1-408-588-6130

Europe
NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

South America
NEC do Brasil S.A.
Fax: +55-11-6462-6829

Hong Kong, Philippines, Oceania
NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

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

Taiwan
NEC Electronics Taiwan Ltd.
Fax: 02-2719-5951

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Japan
NEC Semiconductor Technical Hotline
Fax: 044-435-9608

I would like to report the following error/make the following suggestion:

Document title: _____

Document number: _____ Page number: _____

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	❑	❑	❑	❑
Technical Accuracy	❑	❑	❑	❑
Organization	❑	❑	❑	❑