NEC

User's Manual

μPD789426, 789436, 789446
789456 Subseries

8-Bit Single-Chip Microcontrollers

μPD789425
μPD789426
μPD789435
μPD789436
μPD78F9436

μPD789445
μPD789446
μPD789455
μPD789456
μPD78F9456

Document No. U15075EJ1V0UM00 (1st edition)
Date Published November 2000 N CP(K)

© NEC Corporation 2000
Printed in Japan

[MEMO]

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

EEPROM is a trademark of NEC Corporation.
Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.
PC/AT is a trademark of International Business Machines Corporation.
HP9000 series 700 and HP-UX are trademarks of Hewlett-Packard Company.
SPARCstation is a trademark of SPARC International, Inc.
Solaris and SunOS are trademarks of Sun Microsystems, Inc.
OSF/Motif is a trademark of Open Software Foundation, Inc.
NEWS and NEWS-OS are trademarks of Sony Corporation.
TRON is an abbreviation of The Realtime Operating system Nucleus.
ITRON is an abbreviation of Industrial TRON.

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

License not needed:	μPD78F9436, 78F9456
The customer must judge the need for license:	μPD789425, 789426, 789435, 789436, 789445, 789446, 789455, 789456

- **The information in this document is current as of September, 2000. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC's data sheets or data books, etc., for the most up-to-date specifications of NEC semiconductor products. Not all products and/or types are available in every country. Please check with an NEC sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without prior written consent of NEC. NEC assumes no responsibility for any errors that may appear in this document.
- NEC does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC semiconductor products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC 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 customer's equipment shall be done under the full responsibility of customer. NEC assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC endeavours to enhance the quality, reliability and safety of NEC semiconductor products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC semiconductor products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment, and anti-failure features.
- NEC semiconductor products are classified into the following three quality grades:
  "Standard", "Special" and "Specific". The "Specific" quality grade applies only to semiconductor products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of a semiconductor product depend on its quality grade, as indicated below. Customers must check the quality grade of each semiconductor product before using it in a particular application.
  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots
  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support)
  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.
  The quality grade of NEC semiconductor products is "Standard" unless otherwise expressly specified in NEC's data sheets or data books, etc. If customers wish to use NEC semiconductor products in applications not intended by NEC, they must contact an NEC sales representative in advance to determine NEC's willingness to support a given application.
  (Note)
  (1) "NEC" as used in this statement means NEC Corporation and also includes its majority-owned subsidiaries.
  (2) "NEC semiconductor products" means any semiconductor product developed or manufactured by or for NEC (as defined above).

M8E 00.4

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

[MEMO]

INTRODUCTION

Target Readers

This manual is intended to give user engineers an understanding of the functions of the μPD789426, 789436, 789446, and 789456 Subseries to design and develop its application systems and programs.

Target products:

- μPD789426 Subseries: μPD789425, 789426
- μPD789436 Subseries: μPD789435, 789436
- μPD789446 Subseries: μPD789445, 789446
- μPD789456 Subseries: μPD789455, 789456

Purpose

This manual is designed to deepen your understanding of the following functions using the following organization.

Organization

Two manuals are available for the μPD789426, 789436, 789446, and 789456 Subseries:

This manual and the instruction manual (common to the 78K/0S Series).

μPD789426, 789436, 789446, and 789456 Subseries User's Manual	78K/0S Series User's Manual Instructions
• Pin functions • Internal block functions • Interrupts • Other internal peripheral functions	• CPU function • Instruction set • Instruction description

How to Use This Manual

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

- To understand the overall functions of the μPD789426, 789436, 789446, and 789456 Subseries
  → Read this manual in the order of the **CONTENTS**.
- How to read register formats
  → The name of a bit whose number is enclosed in brackets 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
  → See **APPENDIX C REGISTER INDEX**.
- To learn the details of the instruction functions of the 78K/0S series
  → Refer to **78K/0S Series Instructions User's Manual (U11047E)** separately available.

Conventions

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

Related Documents

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

Documents Related to Devices

Document Name	Document No.	
	Japanese	English
μPD789425, 789426, 789435, 789436, 789445, 789446, 789455, 789456 Preliminary Product Information	U14493J	U14493E
μPD78F9436, 78F9456 Preliminary Product Information	To be prepared	To be prepared
μPD789426, 789436, 789446, 789456 Subseries User's Manual	U15075J	This manual
78K/0S Series Instructions User's Manual	U11047J	U11047E
78K/0, 78K/0S Series Flash Memory Write Application Note	U14458J	U14458E

Documents Related to Development Tools (User's Manuals)

Document Name		Document No.	
		Japanese	English
RA78K0S Assembler Package	Operation	U11622J	U11622E
	Assembly Language	U11599J	U11599E
	Structured Assembly Language	U11623J	U11623E
CC78K/0S C Compiler	Operation	U11816J	U11816E
	Language	U11817J	U11817E
SM78K0S, SM78K0 System Simulator Ver. 2.10 or Later Windows™ Based	Operation	U14611J	To be prepared
SM78K Series System Simulator	External Part User Open Interface Specifications	U10092J	U10092E
ID78K0S-NS Integrated Debugger Windows Based	Reference	U12901J	U12901E
ID78K0-NS, ID78K0S-NS Integrated Debugger Ver. 2.20 or Later Windows Based	Operation	U14910J	To be prepared
IE-78K0S-NS		U13549J	U13549E
IE-789456-NS-EM1		To be prepared	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.	
		Japanese	English
78K/0S Series OS MX78K0S	Fundamental	U12938J	U12938E

Other Related Documents

Document Name	Document No.	
	Japanese	English
SEMICONDUCTOR SELECTION GUIDE Products & Packages (CD-ROM)	X13769X	
Semiconductor Device Mounting Technology Manual	C10535J	C10535E
Quality Grades on NEC Semiconductor Devices	C11531J	C11531E
NEC Semiconductor Device Reliability/Quality Control System	C10983J	C10983E
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892J	C11892E
Guide to Microcomputer-Related Products by Third Parties	U11416J	—

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

[MEMO]

CONTENTS

LIST OF FIGURES (1/5)

LIST OF FIGURES (2/5)

LIST OF FIGURES (3/5)

LIST OF FIGURES (4/5)

LIST OF FIGURES (5/5)

LIST OF TABLES (1/2)

LIST OF TABLES (2/2)

[MEMO]

CHAPTER 1 GENERAL

1.1 Features

- ROM and RAM capacities

Item / Part Number	Program Memory (ROM)		Data Memory	
			Internal High-Speed RAM	LCD Display RAM
μPD789425, 789435	Mask ROM	8 KB	512 bytes	5 bytes
μPD789426, 789436		16 KB		
μPD78F9436	Flash memory	16 KB		
μPD789445, 789455	Mask ROM	12 KB		15 bytes
μPD789446, 789456		16 KB		
μPD78F9456	Flash memory			

- 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: 40 (μPD789426, 789436 Subseries)
  30 (μPD789446, 789456 Subseries)
- Timer: 5 channels
  - 16-bit timer: 1 channel
  - 8-bit timer: 2 channels
  - Watch timer: 1 channel
  - Watchdog timer: 1 channel
- A/D converter:
  8-bit resolution: 6 channels (μPD789426, 789446 Subseries)
  10-bit resolution: 6 channels (μPD789436, 789456 Subseries)
- Serial interface: 1 channel
- LCD controller/driver
  Segment signals: 5, common signals: 4 (μPD789426, 789436 Subseries)
  Segment signals: 15, common signals: 4 (μPD789446, 789456 Subseries)
- Vectored interrupt sources: 15
- Power supply voltage: V_{DD} = 1.8 to 5.5 V
- Operating ambient temperature: T_A = –40 to +85°C

1.2 Applications

Portable audio, cameras, healthcare equipment, etc.

1.3 Ordering Information

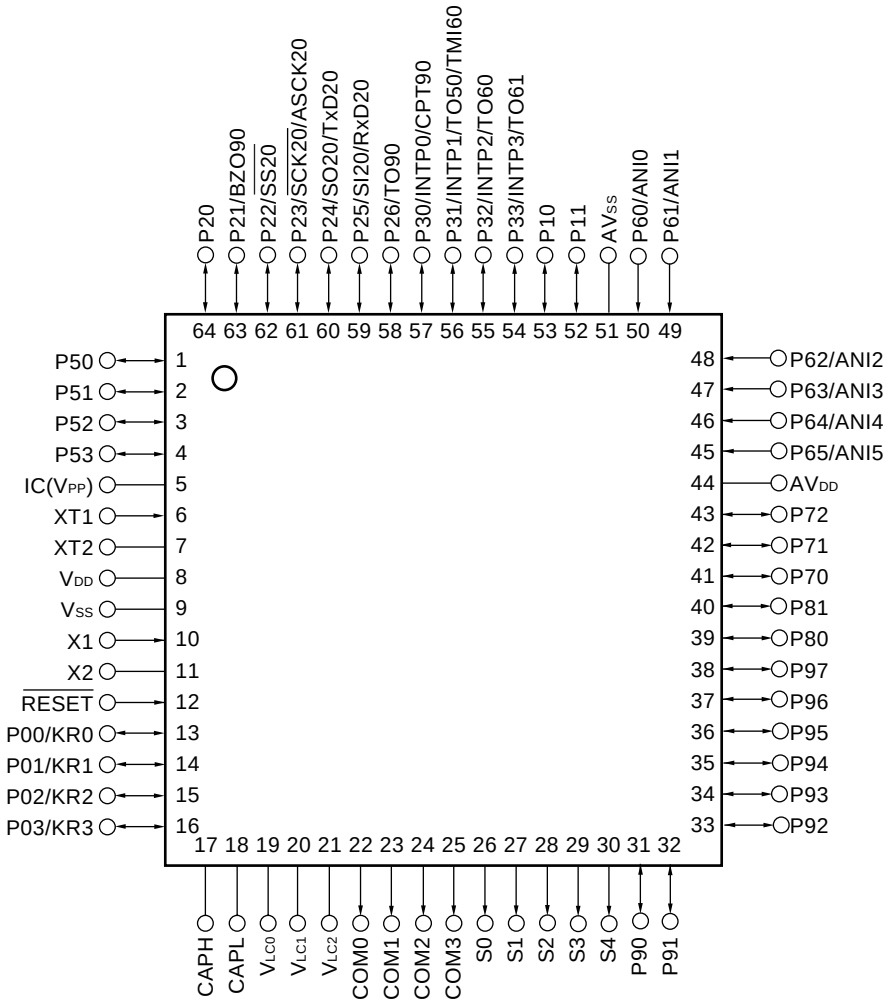
Part Number	Package	Internal ROM
μPD789425GK-×××-9ET	64-pin plastic TQFP (12 × 12 mm)	Mask ROM
μPD789426GK-×××-9ET	64-pin plastic TQFP (12 × 12 mm)	Mask ROM
μPD789435GK-×××-9ET	64-pin plastic TQFP (12 × 12 mm)	Mask ROM
μPD789436GK-×××-9ET	64-pin plastic TQFP (12 × 12 mm)	Mask ROM
μPD789445GK-×××-9ET	64-pin plastic TQFP (12 × 12 mm)	Mask ROM
μPD789446GK-×××-9ET	64-pin plastic TQFP (12 × 12 mm)	Mask ROM
μPD789455GK-×××-9ET	64-pin plastic TQFP (12 × 12 mm)	Mask ROM
μPD789456GK-×××-9ET	64-pin plastic TQFP (12 × 12 mm)	Mask ROM
μPD78F9436GK-9ET	64-pin plastic TQFP (12 × 12 mm)	Flash memory
μPD78F9456GK-9ET	64-pin plastic TQFP (12 × 12 mm)	Flash memory

Remark ××× indicates ROM code suffix.

1.4 Pin Configuration (Top View)

1.4.1 Pin configuration of μPD789426, 789436 Subseries (Top view)

64-pin plastic TQFP (fine pitch) (12 × 12)
μPD789425GK-×××-9ET
μPD789426GK-×××-9ET
μPD789435GK-×××-9ET
μPD789436GK-×××-9ET
μPD78F9436GK-9ET



Cautions 1. Connect the IC (Internally Connected) pin directly to V_{SS}.

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

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

Remark The parenthesized values apply to the μPD78F9436.

1.4.2 Pin configuration of μPD789446, 789456 Subseries (Top view)

64-pin plastic TQFP (fine pitch) (12 × 12)

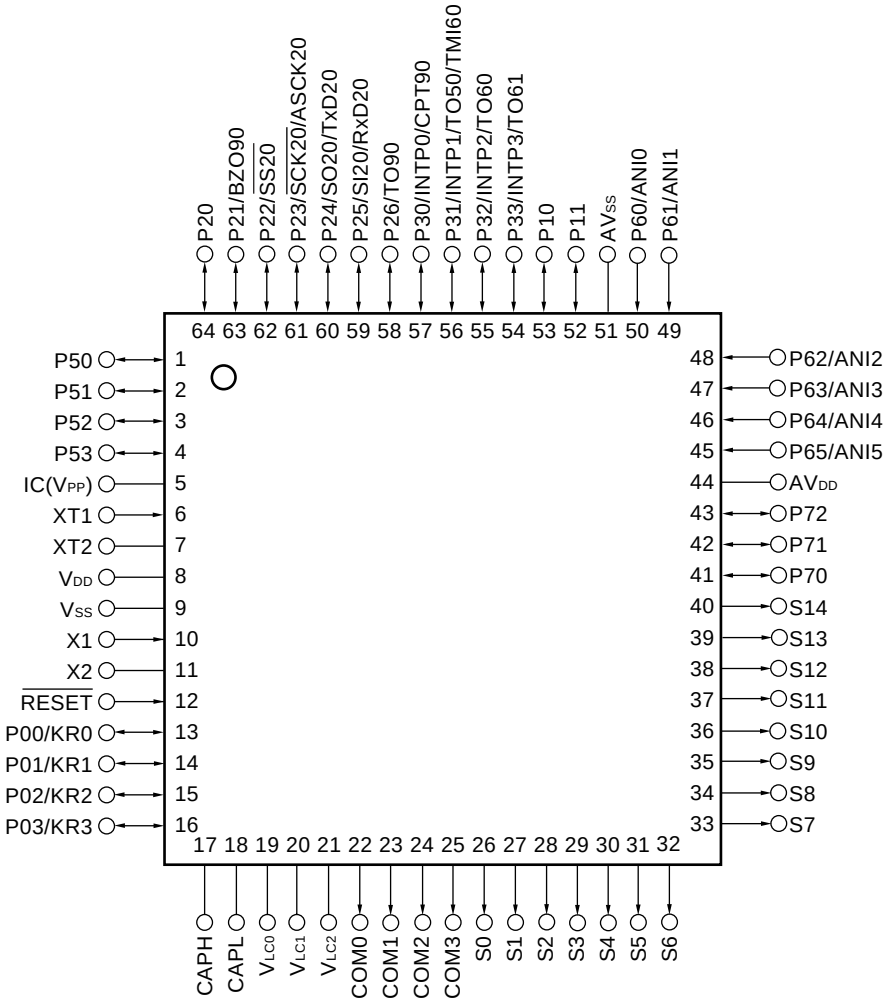
μPD789445GK-×××-9ET

μPD789446GK-×××-9ET

μPD789455GK-×××-9ET

μPD789456GK-×××-9ET

μPD78F9456GK-9ET



Cautions 1. Connect the IC (Internally Connected) pin directly to VSS.

2. Connect the AVDD pin to VDD.

3. Connect the AVSS pin to VSS.

Remark The parenthesized values apply to the μPD78F9456.

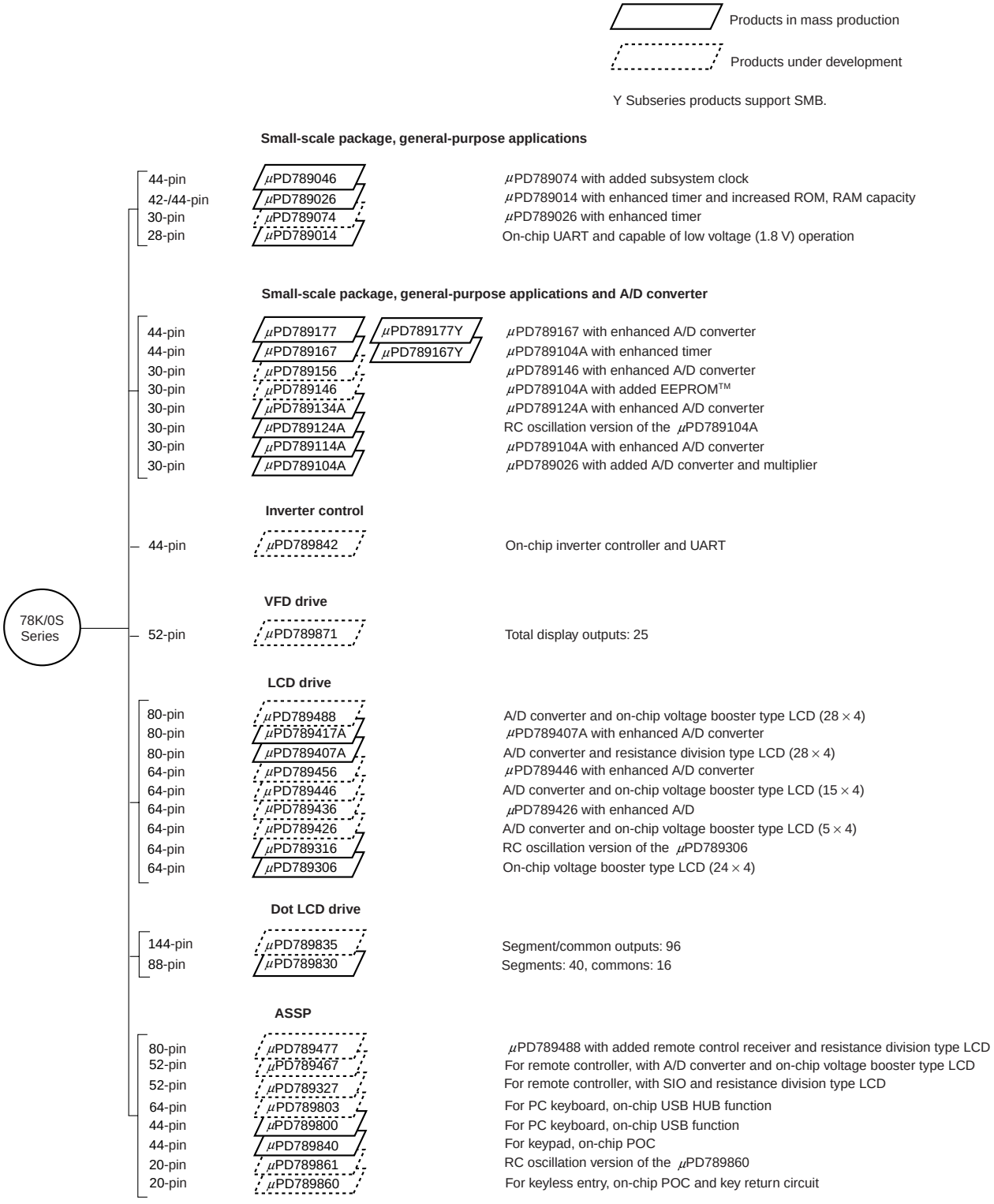
ANI0 to ANI5:	Analog input	P90 to P97[Note 1]:	Port 9
ASCK20:	Asynchronous serial input	$\overline{\text{RESET}}$:	Reset
AV_{DD}:	Analog power supply	RxD20:	Receive data
AV_{SS}:	Analog ground	$\overline{\text{SS20}}$:	Serial chip select
BZO90:	Buzzer output	S0 to S4, S5 to S14[Note 2]:	Segment output
CAPH, CAPL:	LCD power supply capacitance control	$\overline{\text{SCK20}}$:	Serial clock
COM0 to COM3:	Common output	SI20:	Serial input
CPT90:	Capture trigger input	SO20:	Serial output
IC:	Internally connected	TMI60:	Timer input
INTP0 to INTP3:	External interrupt input	TO90, TO50, TO60,	
KR0 to KR3:	Key return	TO61:	Timer output
P00 to P03:	Port 0	TxD20:	Transmit data
P10, P11:	Port 1	V_{DD}:	Power supply
P20 to P26:	Port 2	V_{LC0} to V_{LC2}:	LCD power supply
P30 to P33:	Port 3	V_{PP}:	Programming power supply
P50 to P53:	Port 5	V_{SS}:	Ground
P60 to P65:	Port 6	X1, X2:	Crystal (main system clock)
P70 to P72:	Port 7	XT1, XT2:	Crystal (subsystem clock)
P80, P81[Note 1]:	Port 8		

Notes 1. μPD789426, 789436 Subseries only

2. μPD789446, 789456 Subseries only

1.5 78K/0S Series Lineup

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



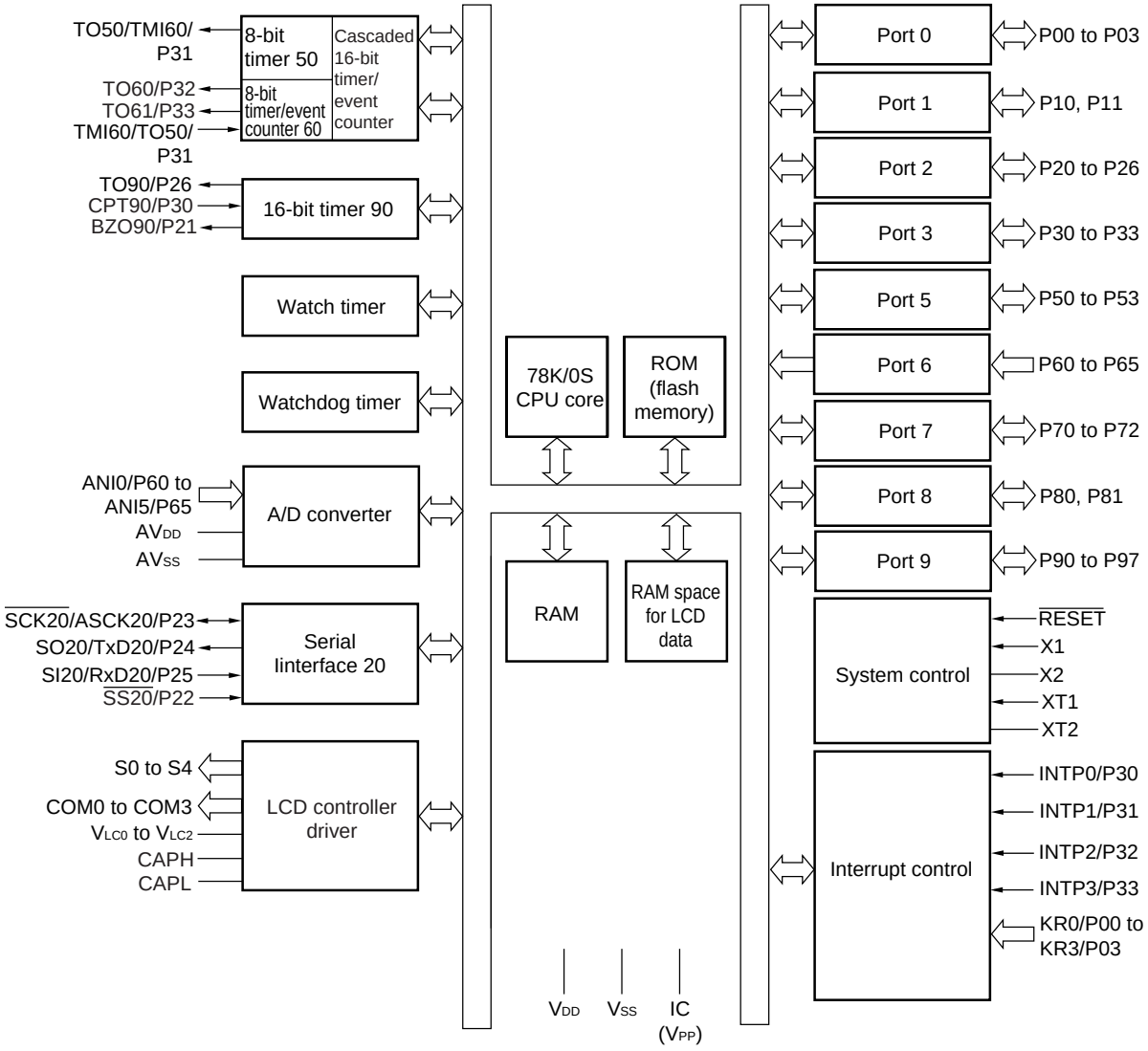
The major functional differences among the subseries are listed below.

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	–
VFD drive	μPD789871	4 K to 8 K	3 ch	–	1 ch	1 ch	–	–	1 ch	33	2.7 V	–
LCD drive	μPD789488	32 K	3 ch	1 ch	1 ch	1 ch	–	8 ch	2 ch (UART: 1 ch)	45	1.8 V	–
	μPD789417A	12 K to 24 K						7 ch	1 ch (UART: 1 ch)	43		
	μPD789407A						7 ch	–				
	μPD789456	12 K 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											–
Dot LCD drive	μPD789835	24 K 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	μPD789477	24 K	3 ch	1 ch	1 ch	1 ch	8 ch	–	2 ch (UART: 1 ch)	45	1.8 V	On-chip LCD
	μPD789467	4 K to 24 K	2 ch	–			1 ch		–	18		
	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

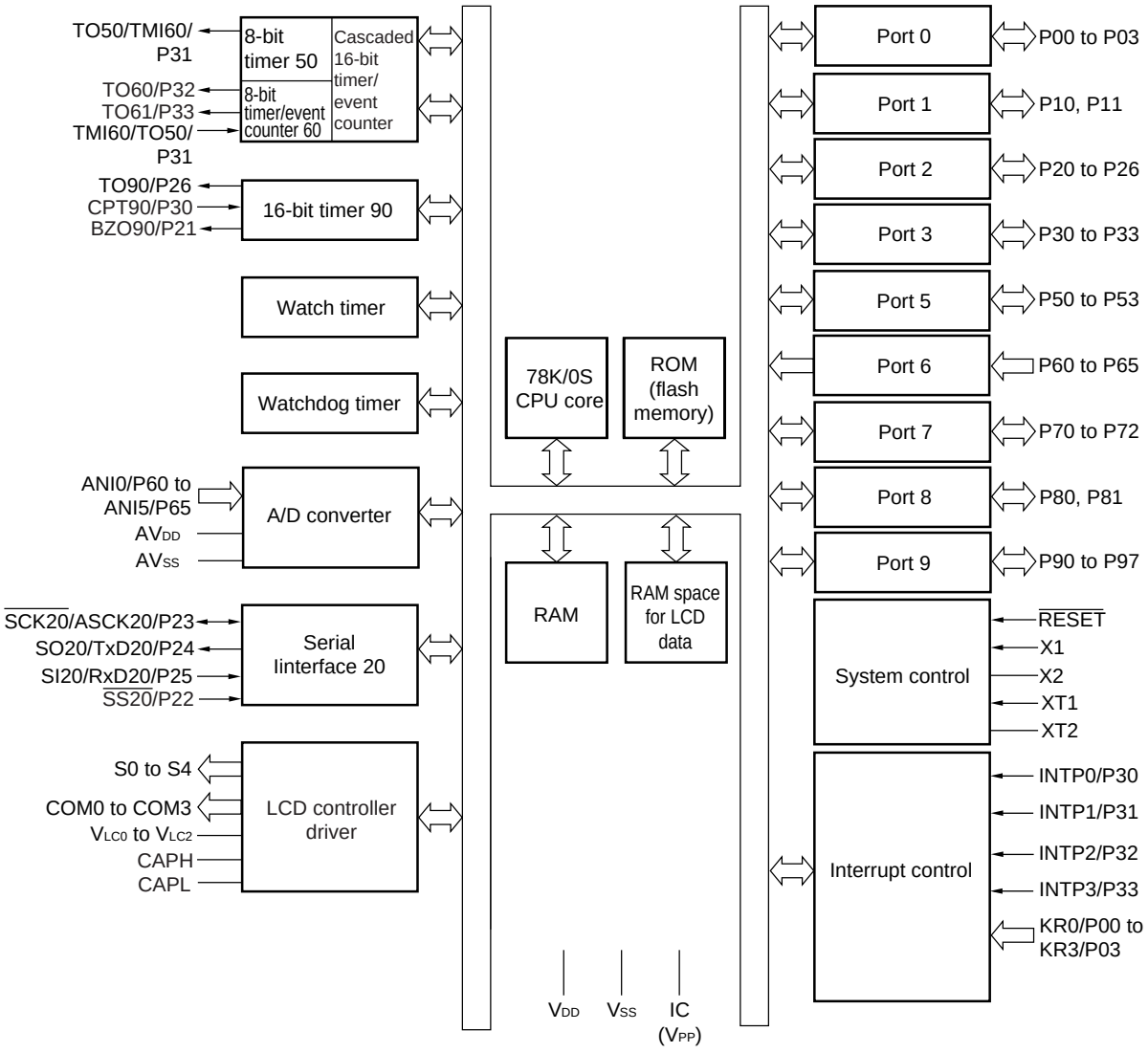
1.6.1 Block diagram of μPD789426, 789436 Subseries



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

2. The parenthesized values apply to the μPD78F9436.

1.6.2 Block diagram of μPD789446, 789456 Subseries



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

2. The parenthesized values apply to the μPD78F9456.

1.7 Overview of Functions

Item		μPD789425, 789435	μPD789426, 789436	μPD78F9436	μPD789445, 789455	μPD789446, 789456	μPD78F9456
Internal memory	ROM	Mask ROM		Flash memory	Mask ROM		Flash memory
		12 KB	16 KB		12 KB	16 KB	
	High-speed RAM	512 bytes					
	LCD display RAM	5 bytes			15 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: 40 • CMOS I/O: 30 • CMOS input: 6 • N-ch open-drain: 4			Total: 30 • CMOS I/O: 20 • CMOS input: 6 • N-ch open-drain: 4		
Timers		• 16-bit timer: 1 channel • 8-bit timer: 2 channels • Watch timer: 1 channel • Watchdog timer: 1 channel					
A/D converter		• 8-bit resolution × 6 channels (μPD789426, 789446 Subseries) • 10-bit resolution × 6 channels (μPD789436, 789456 Subseries)					
Serial interfaces		Switchable between 3-wire serial I/O mode and UART mode: 1 channel					
LCD controller/driver		• Segment signal outputs: 5 max. • Common signal outputs: 4 max.			• Segment signal outputs: 15 max. • Common signal outputs: 4 max.		
Vectored interrupt sources	Maskable	Internal: 9, external: 5					
	Non-maskable	Internal: 1					
Power supply voltage		V_{DD} = 1.8 to 5.5 V					
Operating ambient temperature		T_A = –40 to +85°C					
Package		64-pin plastic TQFP (fine pitch) (12 × 12)					

An outline of the timer is shown below.

		16-Bit Timer	8–Bit Timer 50	8-Bit Timer 60	Watch Timer	Watchdog Timer
Operation mode	Interval timer	–	1 channel	1 channel	1 channel[Note 1]	1 channel[Note 2]
	External event counter	–	–	1 channel	–	–
Function	Timer outputs	1	1	2	–	–
	Square-wave outputs	–	1	2	–	–
	Capture	1 input	–	–	–	–
	Interrupt sources	1	1	1	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 functions. However, use the watchdog timer by selecting either the watchdog timer function or interval timer function.

[MEMO]

2.1 List of Pin Functions

(1) Port pins (1/2)

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P03	I/O	Port 0. 4-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) or key return mode register 00 (KRM00).	Input	KR0 to KR3
P10 to P13	I/O	Port 1. 4-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. 7-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	–
P21				BZO90
P22				$\overline{SS}$20
P23				$\overline{SCK}$20/ASCK20
P24				SO20/TxD20
P25				SI20/RxD20
P26				TO90
P30	I/O	Port 3. 4-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 B3 (PUB3).	Input	INTP0/CPT90
P31				INTP1/TO50/ TMI60
P32				INTP2/TO60
P33				INTP3/TO61
P50 to P53	I/O	Port 5. 4-bit N-ch open-drain I/O port. Input/output can be specified in 1-bit units. For a mask ROM version, an on-chip pull-up resistor can be specified by the mask option.	Input	–
P60 to P65	Input	Port 6. 6-bit input port.	Input	ANI0 to ANI5
P70 to P72	I/O	Port 7. 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 B7 (PUB7).	Input	–

(1) Port pins (2/2)

Pin Name	I/O	Function	After Reset	Alternate Function
P80, P81[Note]	I/O	Port 8. 2-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 B8 (PUB8).	Input	–
P90 to P97[Note]	I/O	Port 9. 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 B9 (PUB9).	Input	–

Note μPD789426, 789436 Subseries only

(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	P30/CPT90
INTP1				P31/TO50/TMI60
INTP2				P32/TO60
INTP3				P33/TO61
KR0 to KR3	Input	Key return signal detection	Input	P00 to P03
$\overline{SS20}$	Input	Serial interface (SIO20) chip select	Input	P22
$\overline{SCK20}$	I/O	Serial interface 20 serial clock input/output	Input	P23/ASCK20
SI20	Input	Serial interface 20 of SIO20 serial data input	Input	P25/RxD20
SO20	Output	Serial interface 20 of SIO20 serial data output	Input	P24/TxD20
ASCK20	Input	Serial clock input for asynchronous serial interface	Input	P23/$\overline{SCK20}$
RxD20	Input	Serial data input for asynchronous serial interface	Input	P25/SI20
TxD20	Output	Serial data output for asynchronous serial interface	Input	P24/SO20
TO90	Output	16-bit timer (TM90) output	Input	P26
CPT90	Input	Capture edge input	Input	P30/INTP0
TO50	Output	8-bit timer (TM50) output	Input	P31/INTP1/TMI40
TO60	Output	8-bit timer (TM60) output	Input	P32/INTP2
TO61	Output		Input	P33/INTP33
TMI60	Input	External count clock input to timer 40	Input	P31/INTP1/TO50
ANI0 to ANI5	Input	A/D converter analog input	Input	P60 to P65
S0 to S4	Output	LCD controller/driver segment signal output	Output	–
S5 to S14[Note]	Output		Output	–
COM0 to COM3	Output	LCD controller/driver common signal output	Output	–
V_{LC0} to V_{LC2}	–	LCD driving voltage	–	–
CAPH	–	Capacitor connection pin for LCD drive	–	–
CAPL	–		–	–
X1	Input	Connecting crystal resonator for main system clock oscillation	–	–
X2	–		–	–
XT1	Input	Connecting crystal resonator for subsystem clock oscillation	–	–
XT2	–		–	–
$\overline{RESET}$	Input	System reset input	Input	–
V_{DD}	–	Positive power supply	–	–
V_{SS}	–	Ground potential	–	–
AV_{DD}	–	A/D converter analog potential	–	
AV_{SS}	–	A/D converter analog ground potential	–	–
IC	–	Internally connected. Connect directly to V_{SS}.	–	–
V_{PP}	–	Sets flash memory programming mode. Applies high voltage when a program is written or verified. Connect directly to V_{SS} in normal operation mode.	–	–

Note μPD789446, 789456 Subseries only

2.2 Description of Pin Functions

2.2.1 P00 to P03 (Port 0)

These pins constitute a 4-bit I/O port. In addition, these pins enable key return signal detection.
Port 0 can be specified in the following operation modes in 1-bit units.

(1) Port mode

These pins constitute a 4-bit I/O port and can be set in the input or output port mode in 1-bit units by 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) Control mode

In this mode, P00 to P03 function as key return signal detection pins (KR0 to KR3).

2.2.2 P10, P11 (Port 1)

These pins constitute a 2-bit I/O port and can be set in the 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 P26 (Port 2)

These pins constitute a 7-bit I/O port. In addition, these pins enable buzzer output, timer output, serial interface data I/O, and serial clock I/O.
Port 2 can be specified in the following operation modes in 1-bit units.

(1) Port mode

In this mode, P20 to P26 function as a 7-bit I/O port. Port 2 can be set in the input or output port mode in 1-bit units by port mode register 2 (PM2). When used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register B2 (PUB2) in 1-bit units.

(2) Control mode

In this mode, P20 to P26 function as the buzzer output, timer output, serial interface data I/O, and serial clock I/O.

(a) Buzzer output

This is the buzzer output pin of 16-bit timer 90.

(b) TO90

This is the timer output pin of 16-bit timer 90.

(c) SI20, SO20

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

(d) $\overline{\text{SCK20}}$

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

(e) RxD20, TxD20

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

(f) ASCK20

This is the serial clock input pin of the asynchronous serial interface.

Caution When using P20 to P26 as serial interface pins, the I/O mode and output latch must be set according to the functions to be used. For the details of the setting, refer to Table 12-2 Settings of Serial Interface 20 Operating Mode.

2.2.4 P30 to P33 (Port 3)

These pins constitute a 4-bit I/O port. In addition, they also function as timer I/O and external interrupt input. Port 3 can be specified in the following operation mode in 1-bit units.

(1) Port mode

In this mode, P30 to P33 functions as a 4-bit I/O port. Port 3 can be set in the input or output port mode in 1-bit units by port mode register 3 (PM3). When used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register B3 (PUB3) in 1-bit units.

(2) Control mode

In this mode, P30 to P33 function as timer I/O and external interrupt input.

(a) TMI60

This is the external clock input pin to timer 60.

(b) TO50, TO60, TO61

These are the timer output pins of timer 50 and timer 60

(c) CPT90

This is the capture edge input pin of 16-bit timer 90.

(d) INTP0 to INTP3

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

2.2.5 P50 to P53 (Port 5)

These pins function as a 4-bit N-ch open-drain I/O port. Port 5 can be set in the input or output port mode in 1-bit units by port mode register 5 (PM5). In the mask ROM version, use of an on-chip pull-up resistor can be specified by a mask option.

2.2.6 P60 to P65 (Port 6)

This is a 6-bit input-only port. In addition to a general-purpose input port function, it has an A/D converter input function.

(1) Port mode

In this mode, P60 to P65 function as 6-bit input-only port.

(2) Control mode

In this mode, P60 to P65 function as analog inputs (ANI0 to ANI5) of A/D converter.

2.2.7 P70 to P72 (Port 7)

These pins constitute a 3-bit I/O port. Port 7 can be set in the input or output mode in 1-bit units by port mode register 7 (PM7). When used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register B7 (PUB7) in port units.

2.2.8 P80, P81 (Port 8)[Note]

These pins constitute a 2-bit I/O port. Port 8 can be set in the input or output mode in 1-bit units by port mode register 8 (PM8). When used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register B8 (PUB8) in port units.

Note Only the μPD789426 and μPD789436 Subseries.

2.2.9 P90 to P97 (Port 9)[Note]

These pins constitute an 8-bit I/O port. Port 9 can be set in the input or output mode in 1-bit units by port mode register 9 (PM9). When used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register B9 (PUB9) in port units.

Note Only the μPD789426 and μPD789436 Subseries.

2.2.10 S0 to S14[Note]

These pins are segment signal output pins for the LCD controller/driver.

Note S0 to S4 in the case of the μPD789426 and 789436 Subseries

2.2.11 COM0 to COM3

These pins are common signal output pins for the LCD controller/driver.

2.2.12 V_{LC0} to V_{LC2}

These pins are power supply voltage pins to drive the LCD.

2.2.13 CAPH, CAPL

These pins are capacitor connection pins to drive the LCD.

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

This pin inputs an active-low system reset signal.

2.2.15 X1, X2

These pins are used to connect a crystal resonator for main system clock oscillation.
To supply an external clock, input the clock to X1 and input the inverted signal to X2.

2.2.16 XT1, XT2

These pins are used to connect a crystal resonator for subsystem clock oscillation.
To supply an external clock, input the clock to XT1 and input the inverted signal to XT2.

2.2.17 V_{DD}

This is the positive power supply pin.

2.2.18 V_{SS}

This is the ground pin.

2.2.19 V_{PP} (μPD78F9436, 78F9456 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.

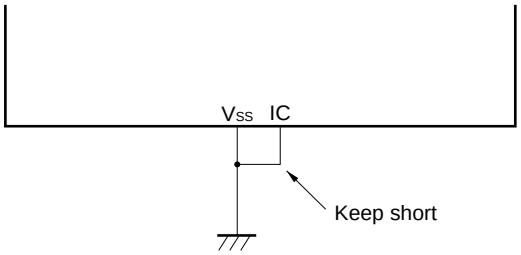
Directly connect this pin to V_{SS} in the normal operation mode.

2.2.20 IC (mask ROM version only)

The IC (Internally Connected) pin is used to set the μPD789426, 789436, 789446, and 789456 Subseries in the test mode before shipment. In the normal operation mode, directly connect this pin to the V_{SS} pin with as short a wiring length as possible.

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

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



2.3 Pin Input/Output Circuits and Recommended Connection of Unused Pins

The input/output circuit type of 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.

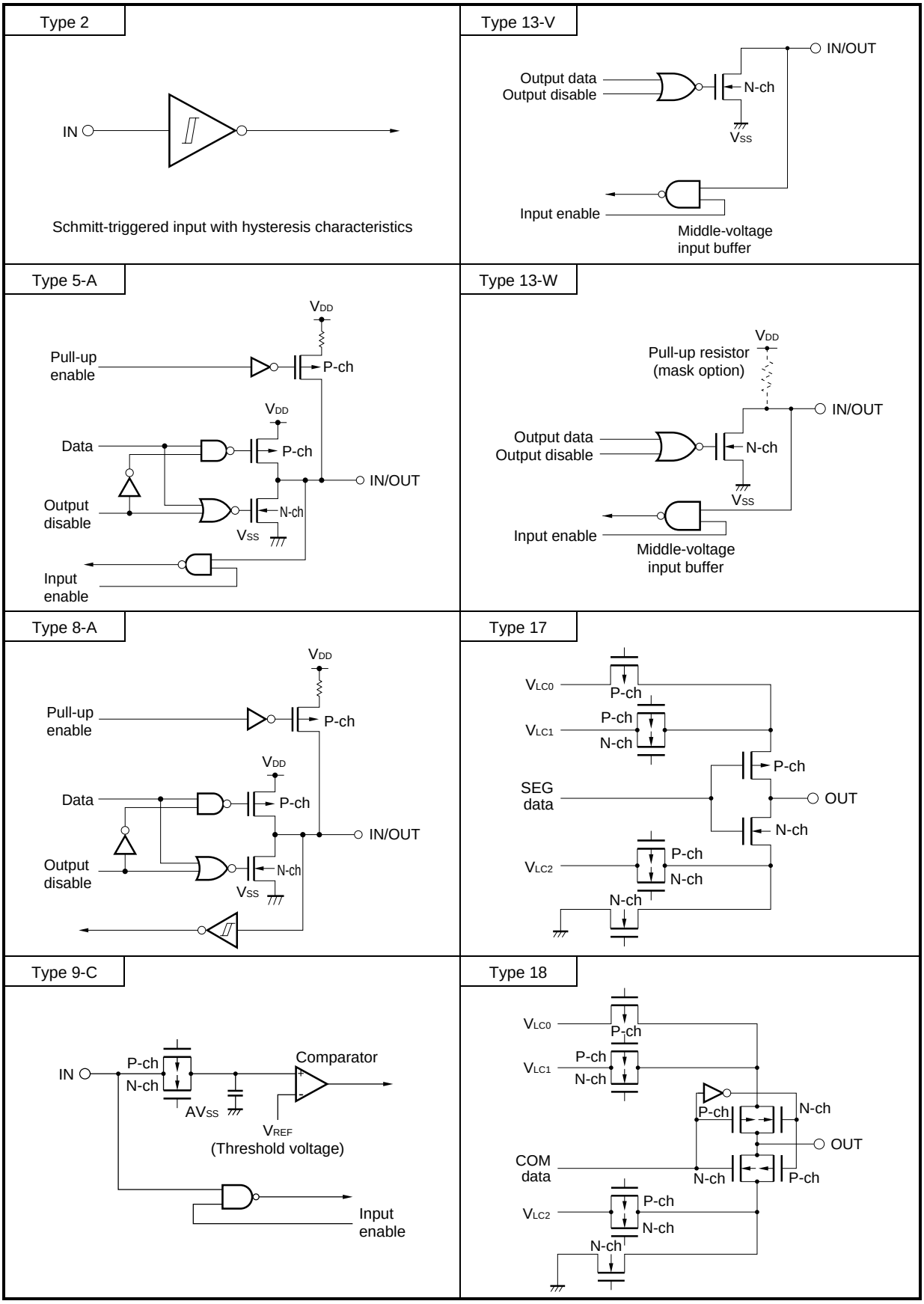
Table 2-1. Types of Pin Input/Output Circuits

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
P00/KR0 to P03/KR3	8-A	I/O	Input: Independently connect to V_{DD} or V_{SS} via a resistor. Output: Leave open.
P10, P11	5-A		
P20	8-A		
P21/BZO90			
P22/$\overline{SS20}$			
P23/$\overline{SCK20}$/ASCK20			
P24/SO20/TxD20			
P25/SI20/RxD20			
P26/TO90			
P30/INPT0/CPT90			Input: Independently connect to V_{SS} via a resistor. Output: Leave open.
P31/INPT1/TO50/TMI60			
P32/INPT2/TO60			
P33/INPT3/TO61			
P50 to P53 (Mask ROM version)	13-W		Input: Independently connect to V_{DD} via a resistor. Output: Leave open.
P50 to P53 (Flash memory version)	13-V		
P60/ANI0 to P65/ANI5	9-C	Input	Connect directly to V_{DD} or V_{SS}.
P70 to P72	5-A	I/O	Input: Independently connect to V_{DD} or V_{SS} via a resistor. Output: Leave open.
P80, P81[Note 1]			
P90 to P97[Note 1]			
S0 to S4[Note 1]	17	Output	Leave open.
S0 to S14[Note 2]			
COM0 to COM3	18	–	
V_{LC0} to V_{LC2}	–		
CAPH, CAPL			
XT1		Input	Connect to V_{SS}.
XT2		–	Leave open.
AV_{SS}			Connect to V_{SS}.
AV_{DD}			Connect to V_{DD}.
$\overline{RESET}$	2	Input	–
IC	–	–	Connect directly to V_{SS}.
V_{PP}			

Notes 1. When using the μPD789426 and 789436 Subseries

2. When using the μPD789446 and 789456 Subseries

Figure 2-1. Pin Input/Output Circuits



[MEMO]

3.1 Memory Space

The μPD789426, 789436, 789446, and 789456 Subseries can access 64 KB of memory space. Figures 3-1 through 3-6 show the memory maps.

Figure 3-1. Memory Map (μPD789425, 789435)

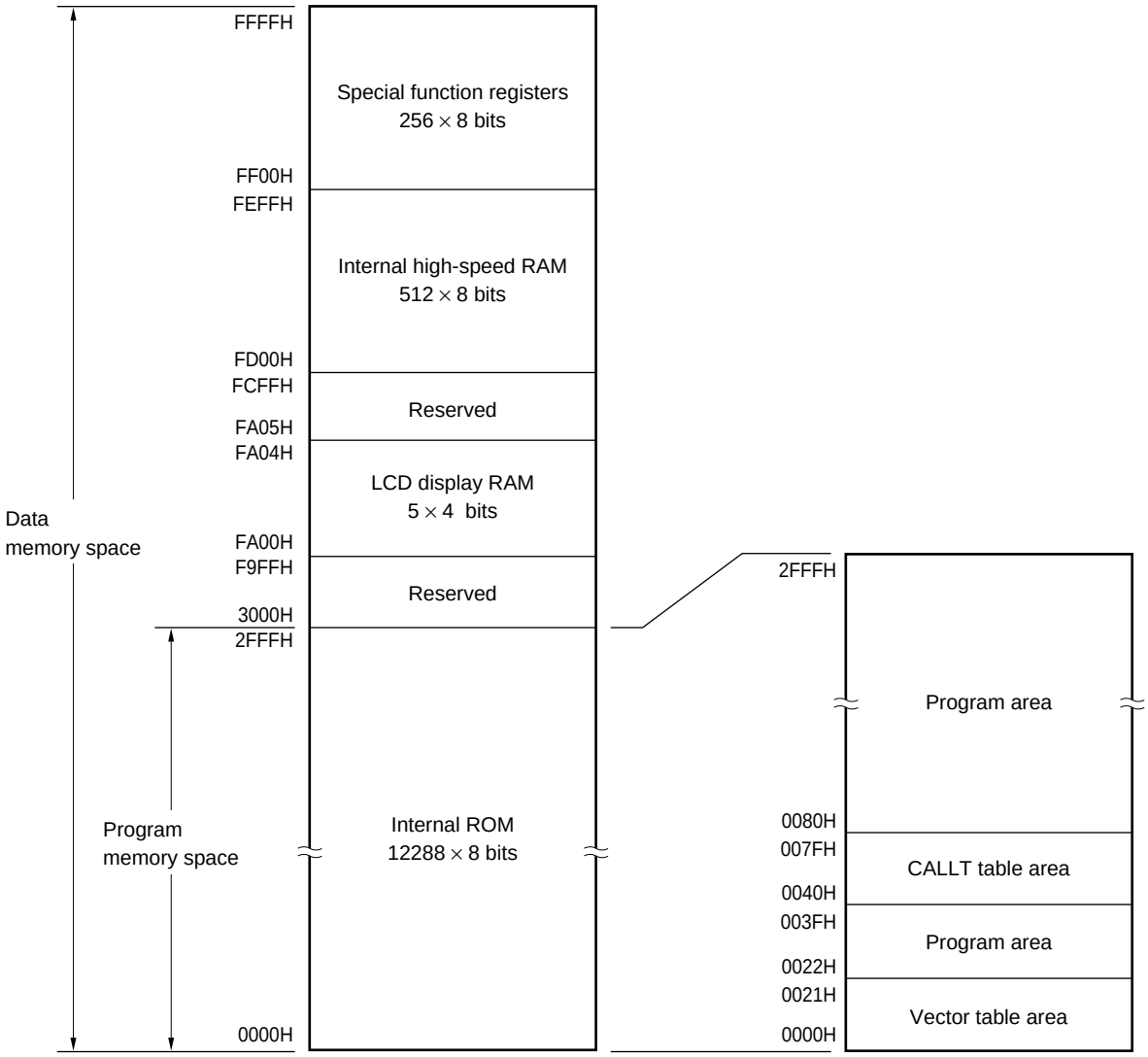


Figure 3-2. Memory Map (μPD789426, 789436)

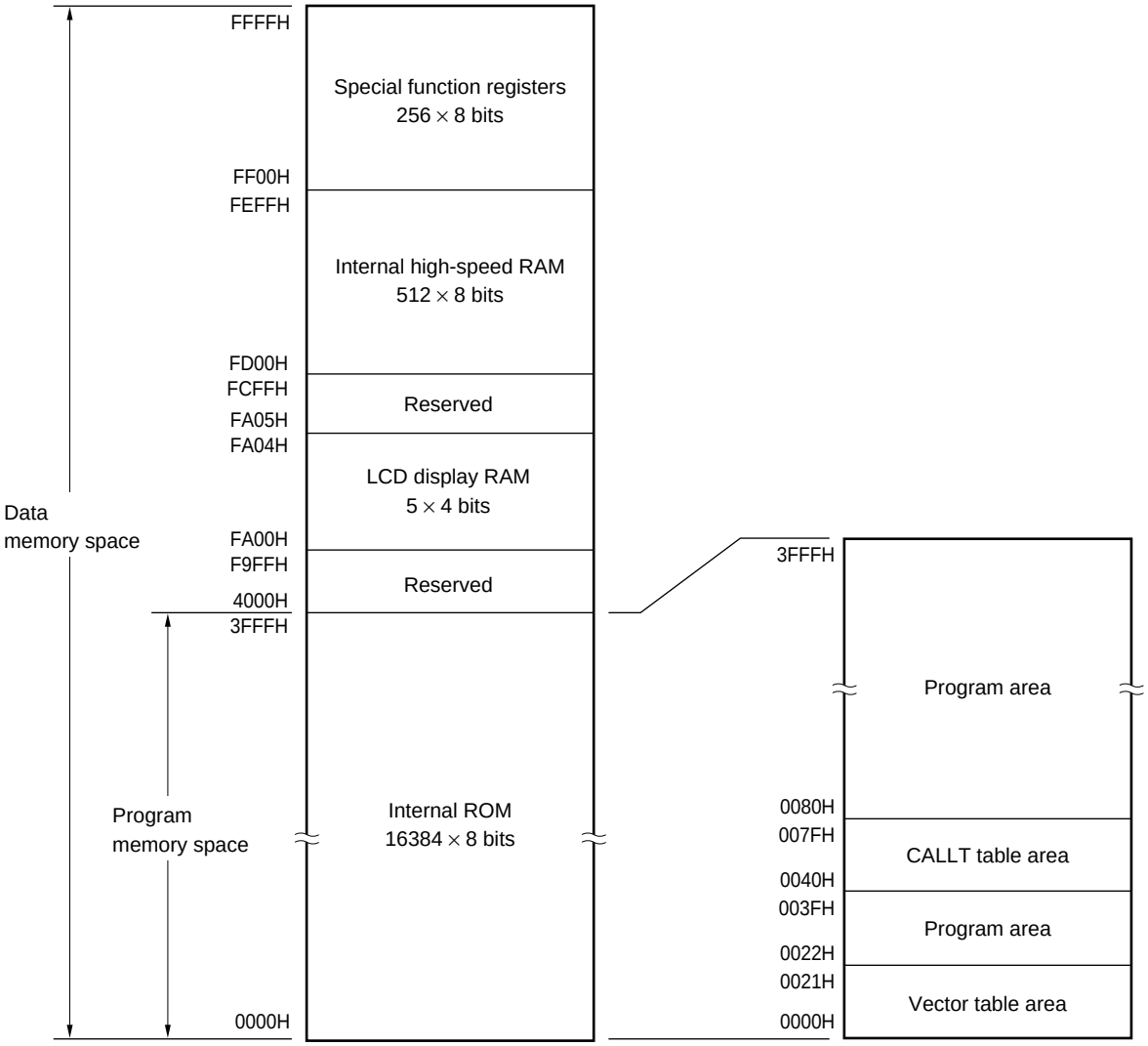


Figure 3-3. Memory Map (μPD78F9436)

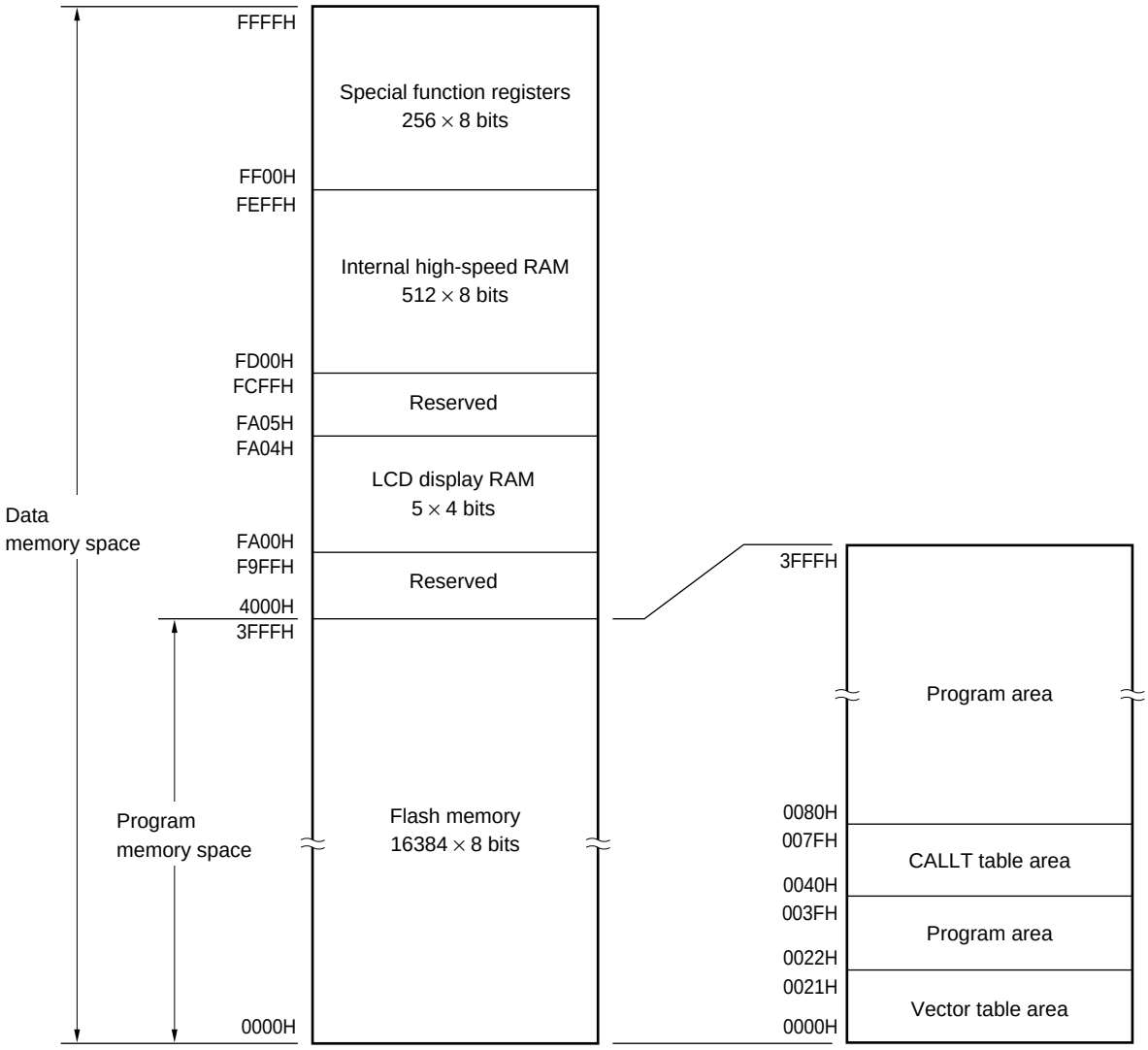


Figure 3-4. Memory Map (μPD789445, 789455)

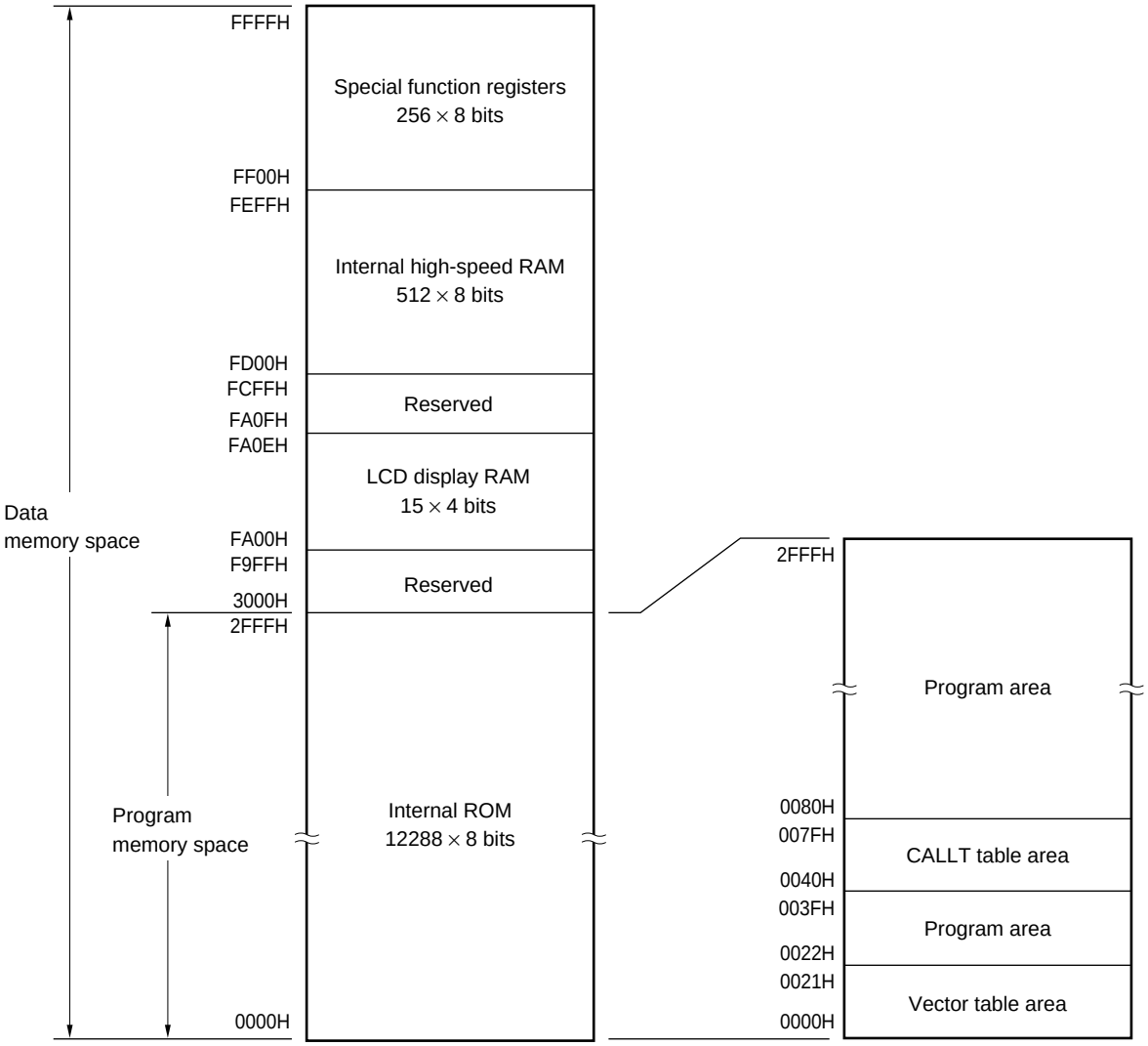


Figure 3-5. Memory Map (μPD789446, 789456)

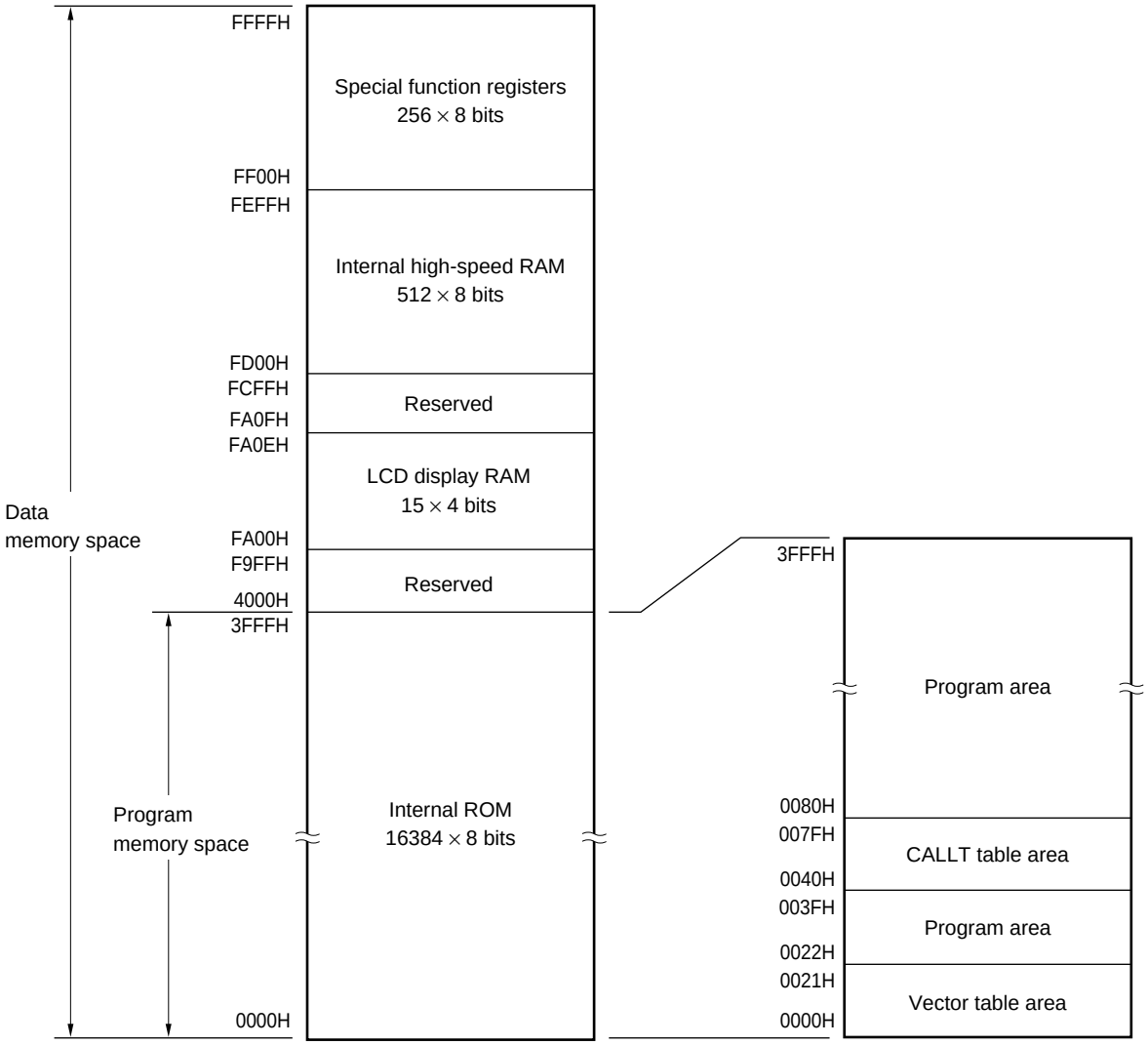
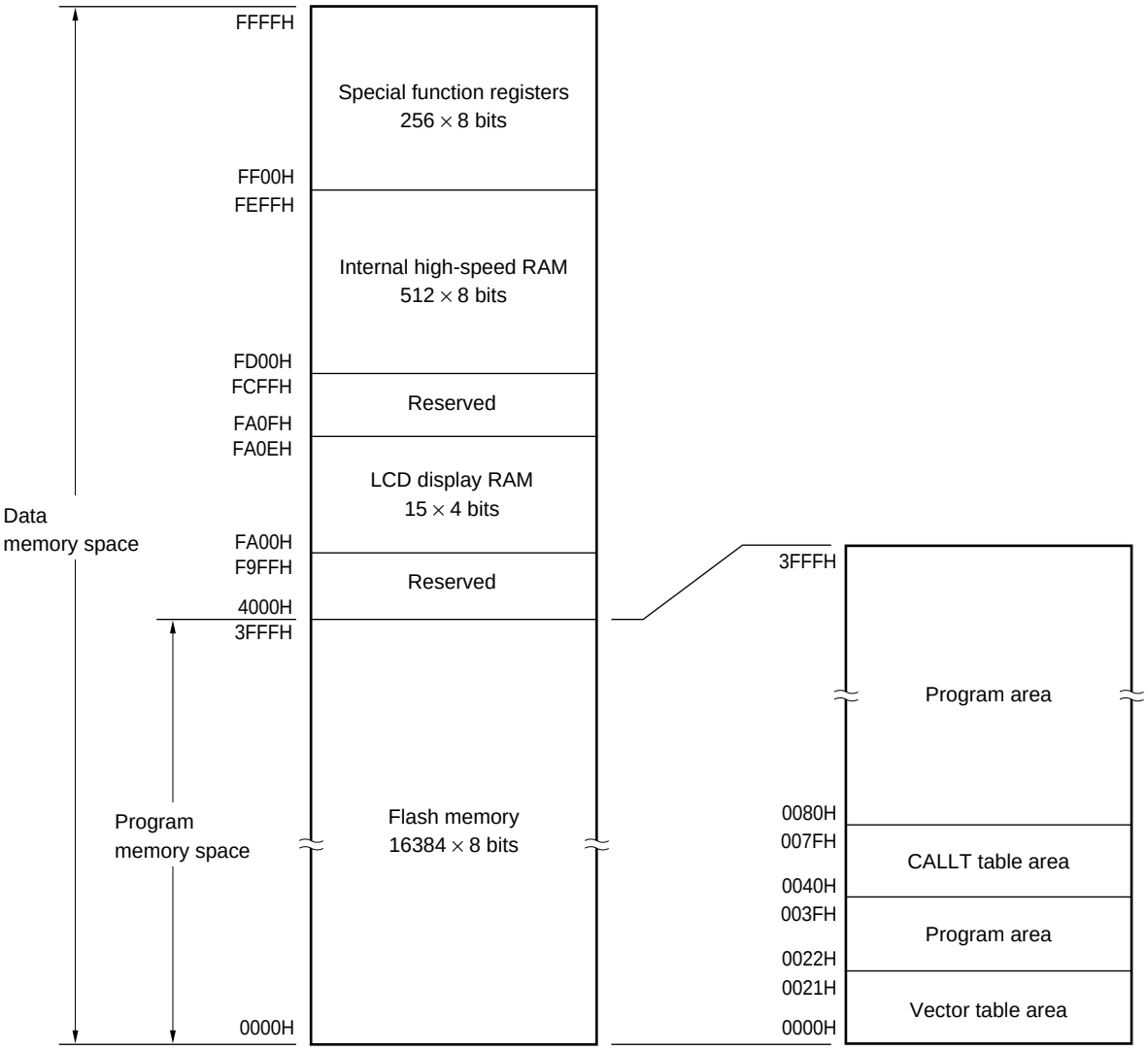


Figure 3-6. Memory Map (μPD78F9456)



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 μPD789426, 789436, 789446, and 789456 Subseries provide internal ROM (or flash memory) with the following capacity for each product.

Table 3-1. Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
μPD789425, 789435, 789445, 789455	Mask ROM	12288 × 8 bits
μPD789426, 789436, 789446, 789456		16384 × 8 bits
μPD78F9436, 78F9456	Flash memory	16384 × 8 bits

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

(1) Vector table area

The 34-byte area of addresses 0000H to 0021H 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	0014H	INTWTI
0004H	INTWDT	0016H	INTTM90
0006H	INTP0	0018H	INTTM50
0008H	INTP1	001AH	INTTM60
000AH	INTP2	001CH	INTAD0
000CH	INTP3	001EH	INTWT
000EH	INTSR20/INTCSI20	0020H	INTKR00
0012H	INTST20		

(2) CALLT instruction table area

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

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

The μPD789426, 789436, 789446, and 789456 Subseries products incorporate the following RAM.

(1) Internal high-speed RAM

Internal high-speed RAM is incorporated in the area between FD00H and FEFFH.

The internal high-speed RAM is also used as a stack.

(2) LCD display RAM

LCD display RAM is incorporated.

The LCD display RAM can also be used as ordinary RAM.

Each subseries incorporates LCD display RAM with the following capacity.

Table 3-3. LCD Display RAM Capacity

Subseries Name	Area	Capacity
μPD789426, 789436 Subseries	FA00H to FA04H	5 × 4 bits
μPD789446, 789456 Subseries	FA00H to FA0EH	15 × 4 bits

3.1.3 Special function register (SFR) area

Special function registers (SFRs) of on-chip peripheral hardware are allocated in the area between FF00H to FFFFH (see **Table 3-4**).

3.1.4 Data memory addressing

The μPD789426, 789436, 789446, and 789456 Subseries are provided with a variety of addressing modes to make memory manipulation as efficient as possible. At the addresses corresponding to data memory area (FD00H to FFFFH) especially, specific addressing modes that correspond to the particular function an area, such as the special function registers are available. Figures 3-7 through 3-12 show the data memory addressing modes.

Figure 3-7. Data Memory Addressing (μPD789425, 789435)

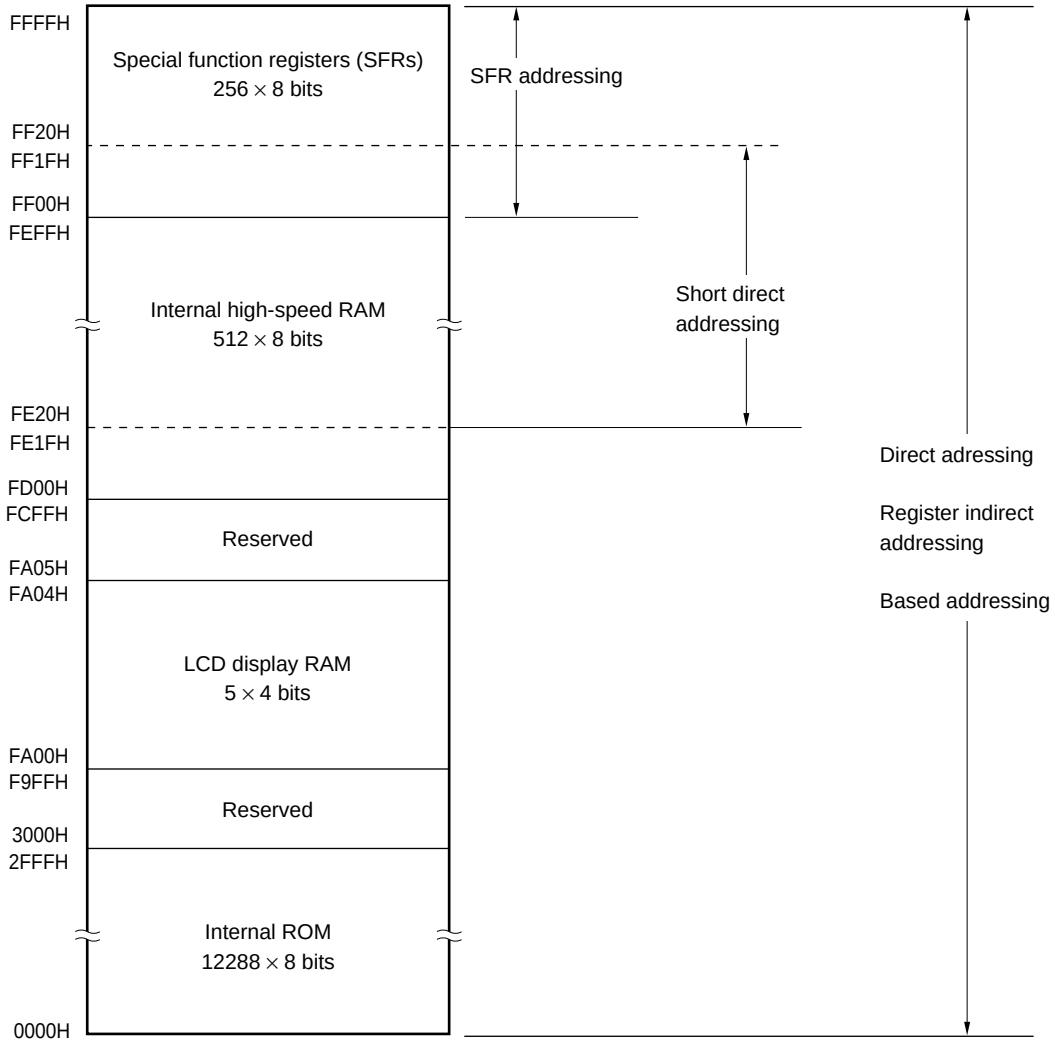


Figure 3-8. Data Memory Addressing (μPD789426, 789436)

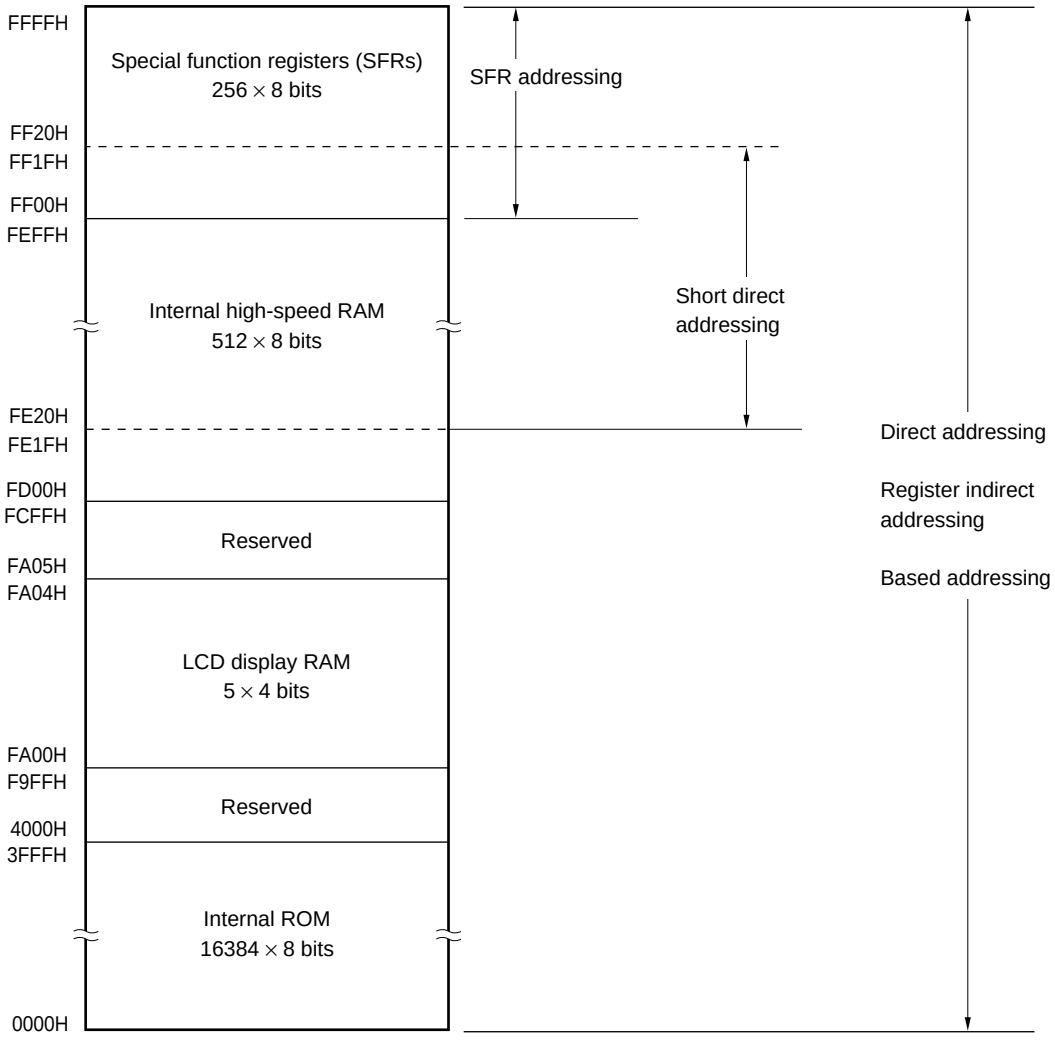


Figure 3-9. Data Memory Addressing (μPD78F9436)

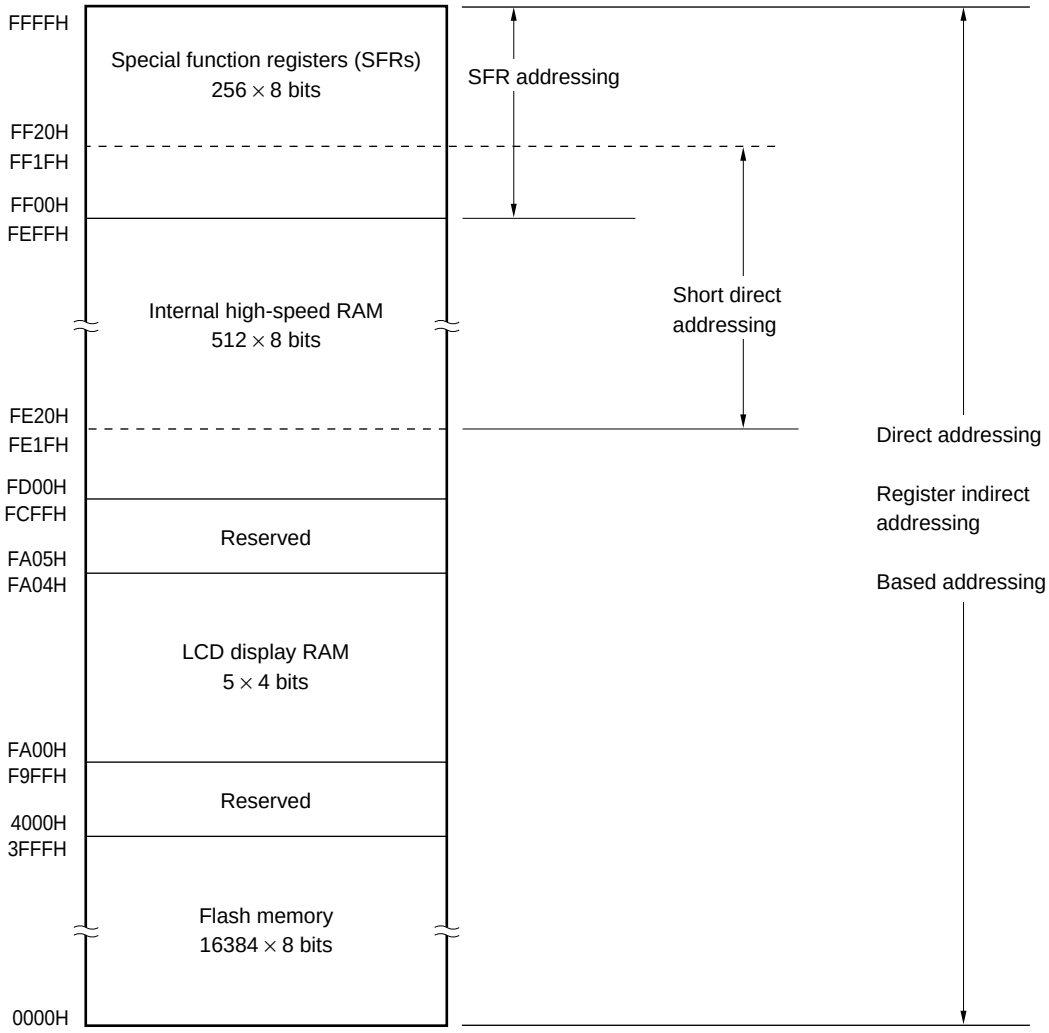


Figure 3-10. Data Memory Addressing (μPD789445, 789455)

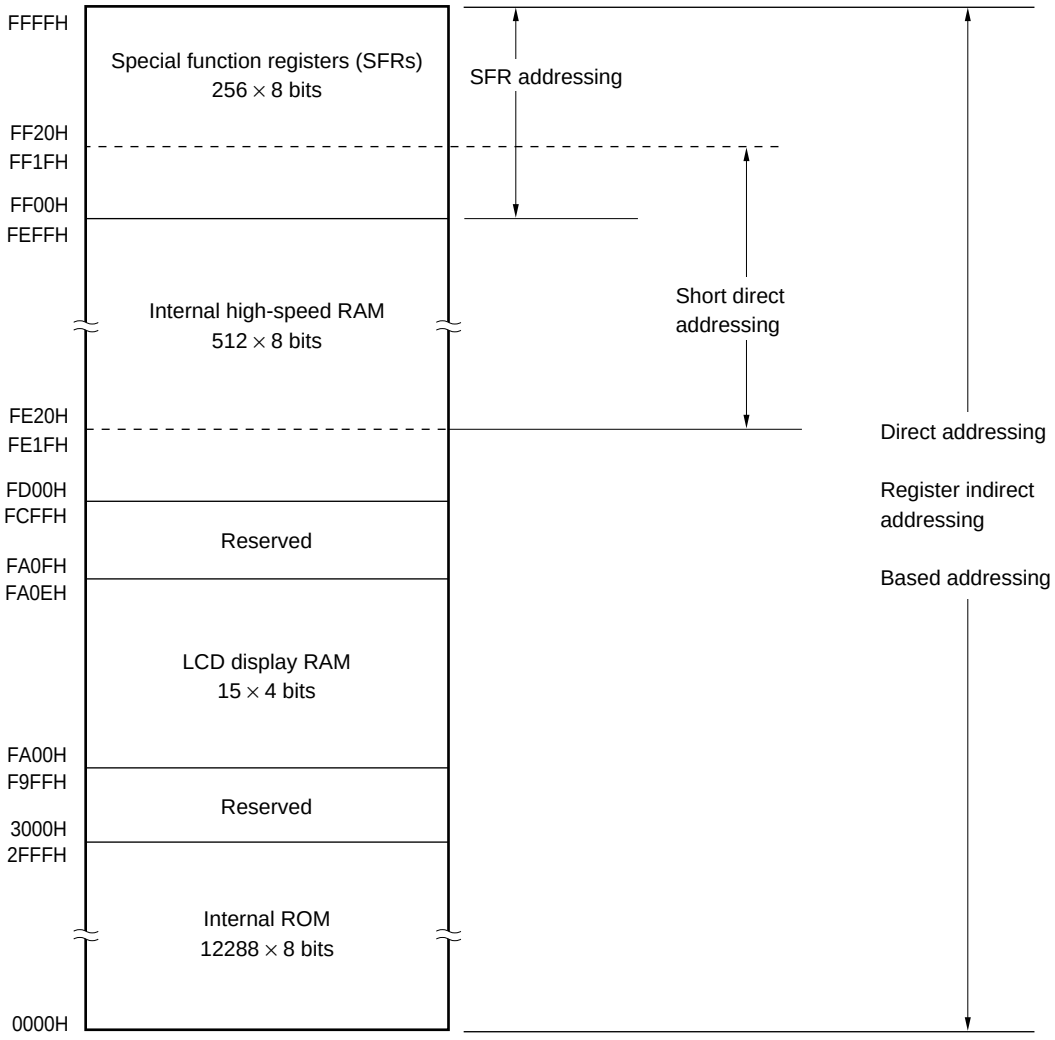


Figure 3-11. Data Memory Addressing (μPD789446, 789456)

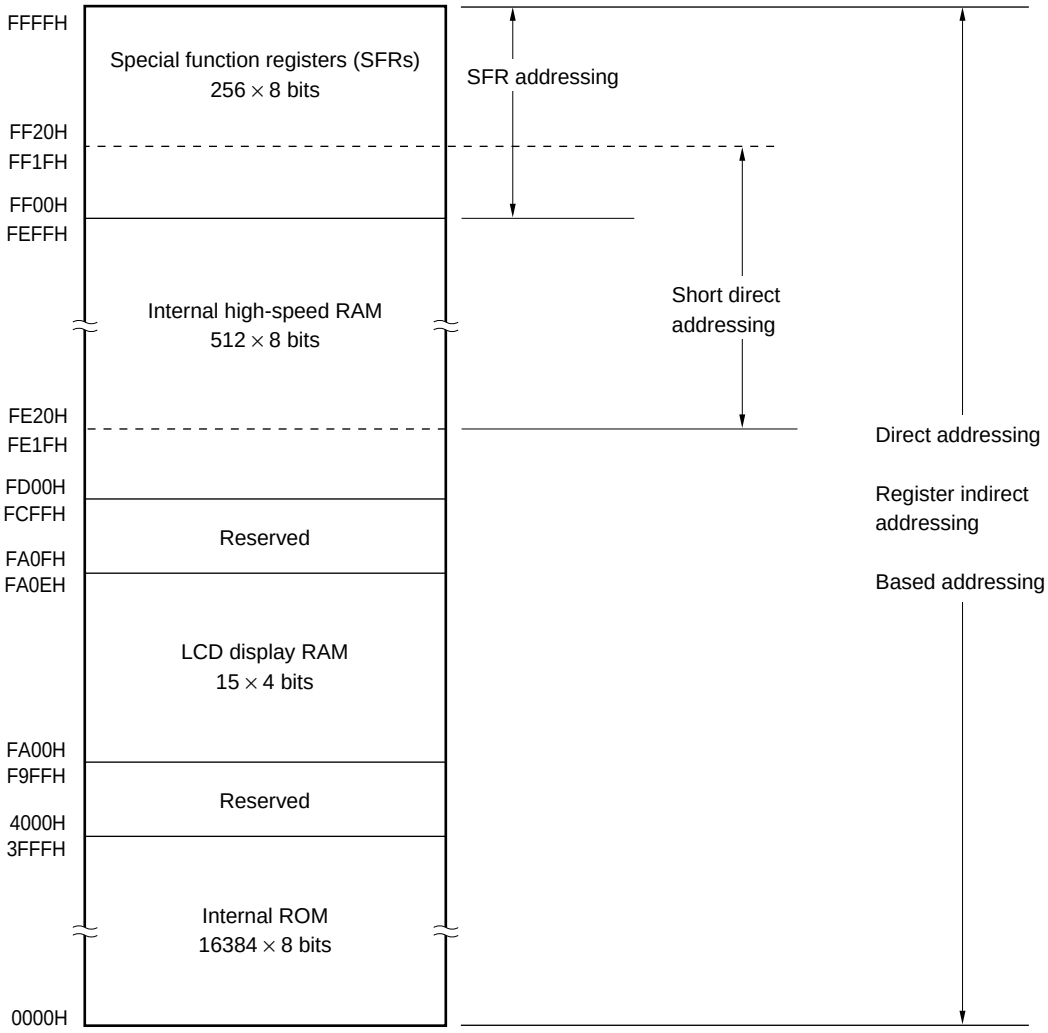
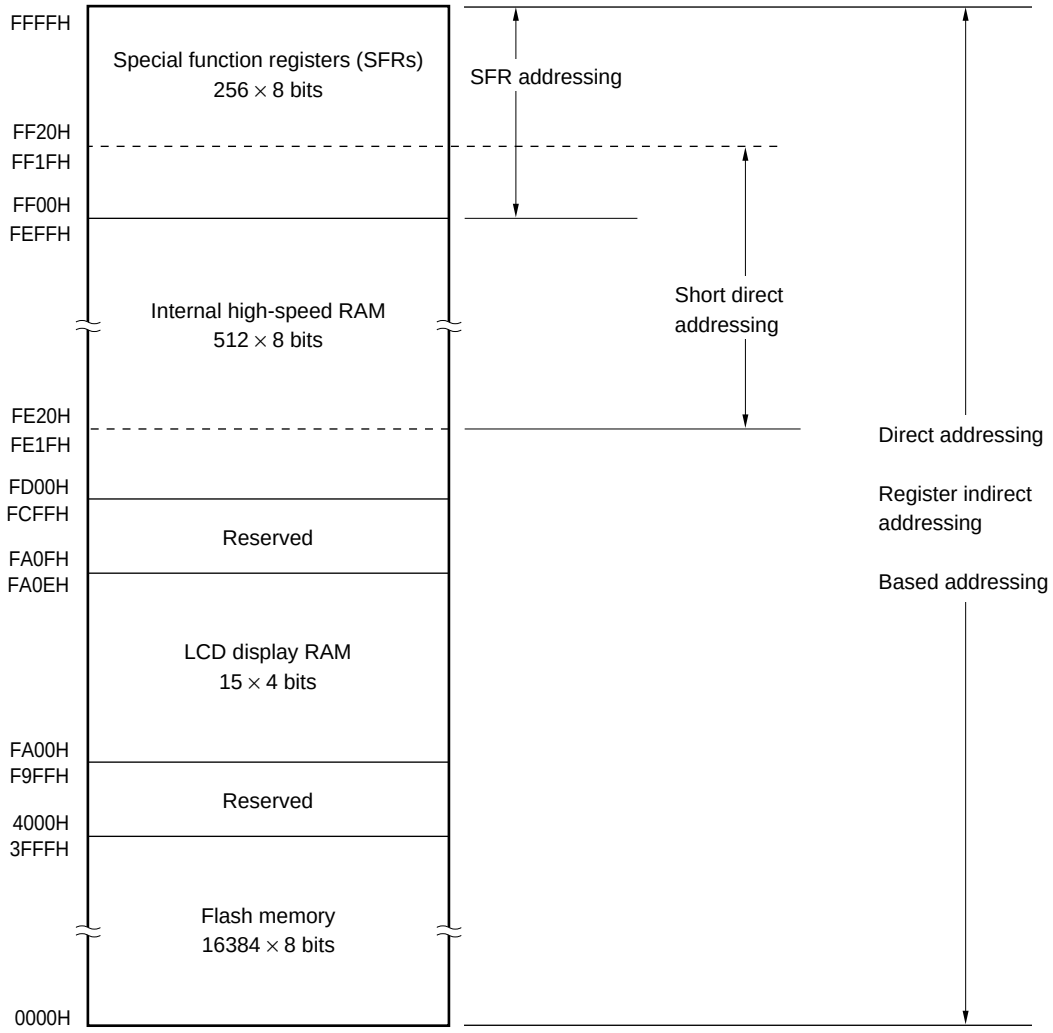


Figure 3-12. Data Memory Addressing (μPD78F9456)



3.2 Processor Registers

The μPD789426, 789436, 789446, and 789456 Subseries provide 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 that holds the address information of the next program to be executed.

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

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

Figure 3-13. 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.

The program status word contents are automatically stacked upon interrupt request generation or PUSH PSW instruction execution and are automatically restored upon execution of the RETI and POP PSW instructions.

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

Figure 3-14. 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 acknowledgement operations of the CPU.

When 0, IE is set to the interrupt disable status (DI), and interrupt requests other than non-maskable interrupt are all disabled.

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

IE 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 (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 3-15. 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 3-16 and 3-17.

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

Figure 3-16. Data to Be Saved to Stack Memory

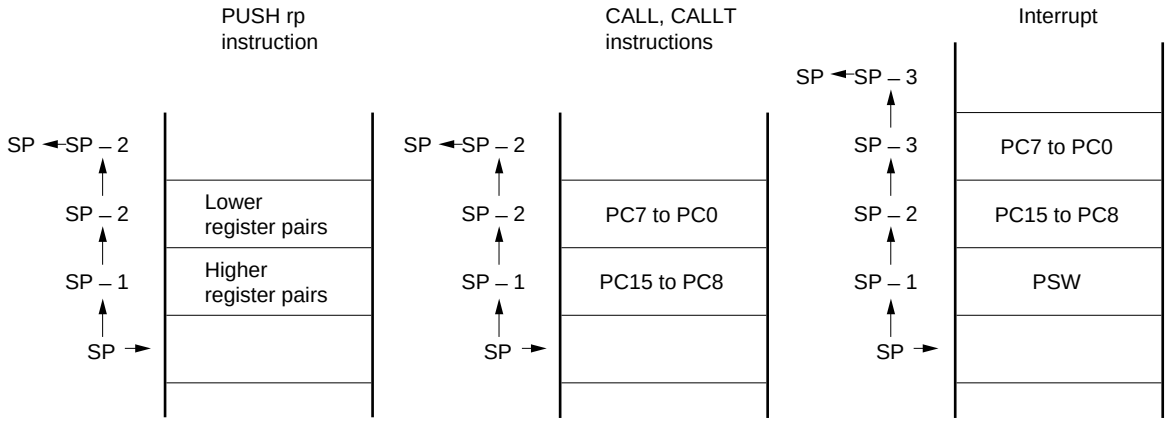
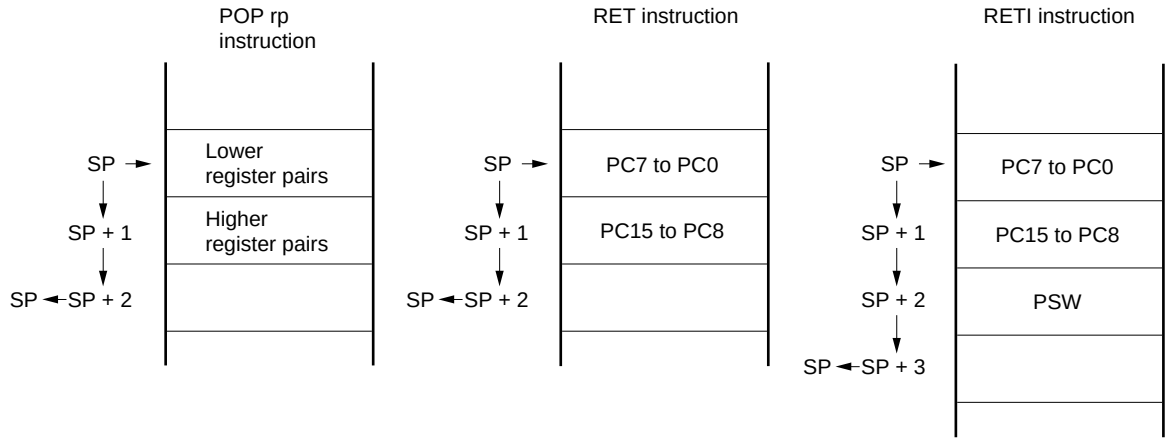


Figure 3-17. Data to Be Restored from Stack Memory



3.2.2 General-purpose registers

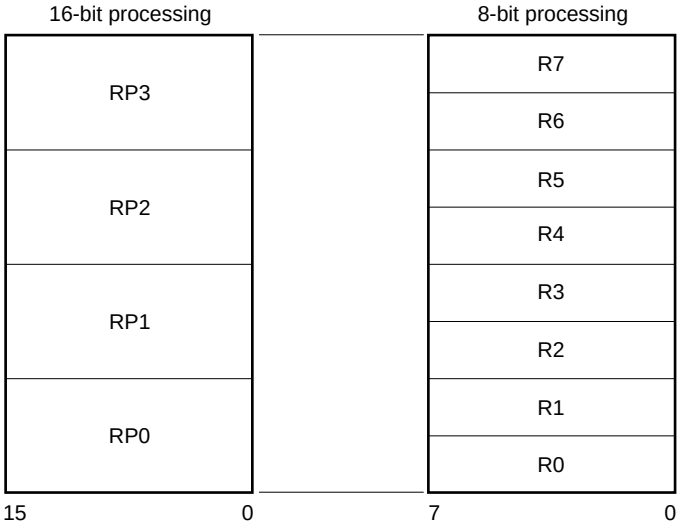
The general-purpose registers consist of eight 8-bit registers (X, A, C, B, E, D, L, and H).

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

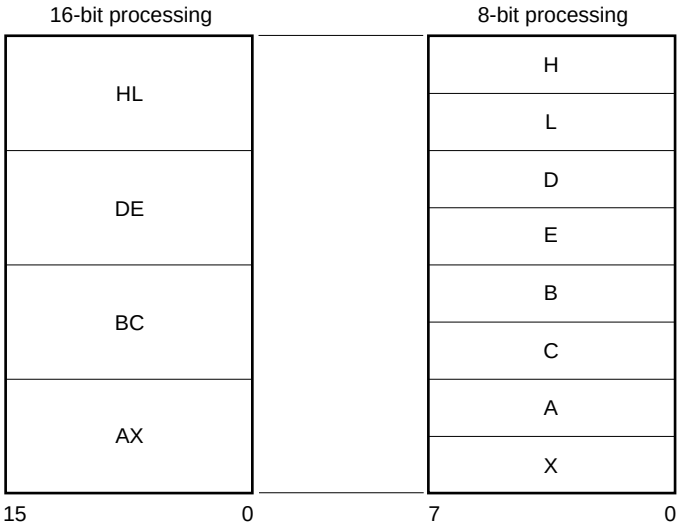
General-purpose registers can be described in terms of function names (X, A, C, B, E, D, L, H, AX, BC, DE, or HL) or absolute names (R0 to R7 and RP0 to RP3).

Figure 3-18. General-Purpose Register Configuration

(a) Absolute names



(b) Function names



3.2.3 Special function registers (SFRs)

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

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

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

The manipulatable bits can be specified as follows.

- 1-bit manipulation
  Describes a symbol reserved by the assembler for the 1-bit manipulation instruction operand (sfr.bit). This manipulation can also be specified with an address.

- 8-bit manipulation
  Describes a symbol reserved by the assembler for the 8-bit manipulation instruction operand (sfr). This manipulation can also be specified with an address.

- 16-bit manipulation
  Describes a symbol reserved by the assembler for the 16-bit manipulation instruction operand. When addressing an address, describe an even address.

Table 3-4 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 an 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 manipulation unit
  Indicates the bit units (1, 8, 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 3-4. Special Function Register List (1/2)

Address	Special Function Register (SFR) Name	Symbol		R/W	Bit Manipulation Unit			After Reset
					1 Bit	8 Bits	16 Bits	
FF00H	Port 0	P0		R/W	√	√	–	00H
FF01H	Port 1	P1			√	√	–	
FF02H	Port 2	P2			√	√	–	
FF03H	Port 3	P3			√	√	–	
FF05H	Port 5	P5			√	√	–	
FF06H	Port 6	P6		R	√	√	–	
FF07H	Port 7	P7		R/W	√	√	–	
FF08H	Port 8 Note 1	P8			√	√	–	
FF09H	Port 9 Note 1	P9			√	√	–	
FF0CH	8-bit compare register 60	CR60	CR6 Note 2	W	–	√	√ Notes 3, 4	Undefined
FF0DH	8-bit compare register 50	CR50			–	√		
FF0EH	8-bit timer counter 60	TM60	TM6 Note 2	R	–	√	√ Notes 3, 4	00H
FF0FH	8-bit timer counter 50	TM50			–	√		
FF10H	Transmit shift register 20	TXS20	SIO20	W	–	√	–	FFH
	Receive buffer register 20	RXB20		R	–	√	–	Undefined
FF14H	A/D conversion result register 0	ADCR0 Note 5		R	–	√	√ Notes 3	0000H
FF15H								
FF16H	16-bit compare register 90	CR90 Note 2		W	–	–	√ Notes 3, 4	FFFFH
FF17H								
FF18H	16-bit timer counter 90	TM90 Note 2		R	–	–	√ Notes 3, 4	0000H
FF19H								
FF1AH	16-bit capture register 90	TCP90 Note 2			–	–	√ Note 3	Undefined
FF1BH								
FF20H	Port mode register 0	PM0		R/W	√	√	–	FFH
FF21H	Port mode register 1	PM1			√	√	–	
FF22H	Port mode register 2	PM2			√	√	–	
FF23H	Port mode register 3	PM3			√	√	–	
FF25H	Port mode register 5	PM5			√	√	–	

Notes 1. μPD789426 and 789436 Subseries only.

2. Name of SFR dedicated for 16-bit access.

3. Only in short direct addressing, 16-bit access is possible.

4. These are 16-bit access dedicated registers, however, 8-bit access is possible. When performing 8-bit access, access using direct addressing.

5. When used as an 8-bit A/D converter (μPD789426 and 789446 Subseries), only 8-bit access is possible. In this case, the address is FF15H.

When used as a 10-bit A/D converter (μPD789436 and 789456 Subseries), only 16-bit access is possible. When the μPD78F9436, a flash memory version of the μPD789425 or μPD789426, is used, this register can be accessed in 8-bit units. However, only an object file assembled with the μPD789425 or μPD789426 can be used. The same is also true for the μPD78F9456, a flash memory version of the μPD789445 or μPD789446: this register can be accessed in 8-bit units, but only an object file assembled with the μPD789445 or μPD789446 can be used.

Table 3-4. Special Function Register List (2/2)

Address	Special Function Register (SFR) Name	Symbol	R/W	Bit Manipulation Unit			After Reset
				1 Bit	8 Bits	16 Bits	
FF27H	Port mode register 7	PM7	R/W	√	√	–	FFH
FF28H	Port mode register 8[Note]	PM8		√	√	–	
FF29H	Port mode register 9[Note]	PM9		√	√	–	
FF32H	Pull-up resistor option register B2	PUB2		√	√	–	00H
FF33H	Pull-up resistor option register B3	PUB3		√	√	–	
FF37H	Pull-up resistor option register B7	PUB7		√	√	–	
FF38H	Pull-up resistor option register B8[Note]	PUB8		√	√	–	
FF39H	Pull-up resistor option register B9[Note]	PUB9		√	√	–	
FF42H	Watchdog timer clock select register	WDCS		–	√	–	
FF48H	16-bit timer mode control register 90	TMC90		√	√	–	
FF49H	Buzzer output control register 90	BZC90		√	√	–	
FF4AH	Watch timer mode control register	WTM		√	√	–	
FF4CH	8-bit compare register H60	CRH60	W	–	√	–	Undefined
FF4DH	8-bit timer mode control register 50	TMC50	R/W	√	√	–	00H
FF4EH	8-bit timer mode control register 60	TMC60		√	√	–	
FF4FH	Carrier generator output control register 60	TCA60	W	√	√	–	
FF70H	Asynchronous serial interface mode register 20	ASIM20	R/W	√	√	–	
FF71H	Asynchronous serial interface status register 20	ASIS20	R	√	√	–	
FF72H	Serial operation mode register 20	CSIM20	R/W	√	√	–	
FF73H	Baud rate generator control register 20	BRGC20		–	√	–	
FF80H	A/D converter mode register 0	ADM0		√	√	–	
FF84H	Analog input channel specification register 0	ADS0		√	√	–	
FFB0H	LCD display mode register 0	LCDM0		√	√	–	
FFB2H	LCD clock control register 0	LCDC0		√	√	–	
FFB3H	LCD voltage amplification control register 0	LCDVA0		√	√	–	
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
FFEDH	External interrupt mode register 1	INTM1		–	√	–	
FFF0H	Suboscillation mode register	SCKM		√	√	–	
FFF2H	Subclock control register	CSS		√	√	–	
FFF5H	Key return mode register 00	KRM00		√	√	–	
FFF7H	Pull-up resistor option register 0	PU0		√	√	–	
FFF9H	Watchdog timer mode register	WDTM		√	√	–	
FFFAH	Oscillation stabilization time select register	OSTS		–	√	–	04H
FFFBH	Processor clock control register	PCC		√	√	–	02H

Note μPD789426 and 789436 Subseries only.

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 Instructions User's Manual (U11047E)**).

3.3.1 Relative addressing

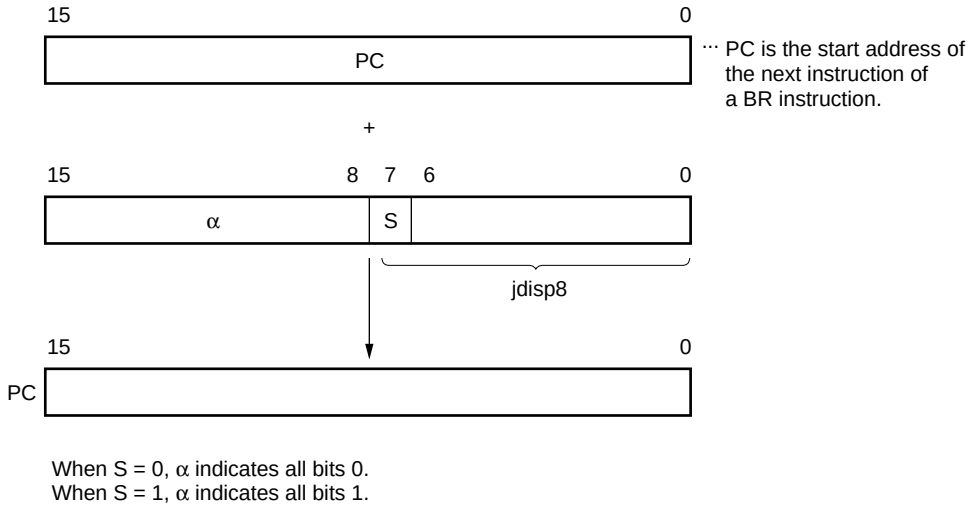
[Function]

The value obtained by adding 8-bit immediate data (displacement value: jdisp8) of an instruction code to the start address of the following instruction is transferred to the program counter (PC) and branched. The displacement value is treated as signed two's complement data (–128 to +127) and bit 7 becomes a sign bit.

This means that information is relatively branched to a location between –128 and +127, from the start address of the next instruction when relative addressing is used.

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]

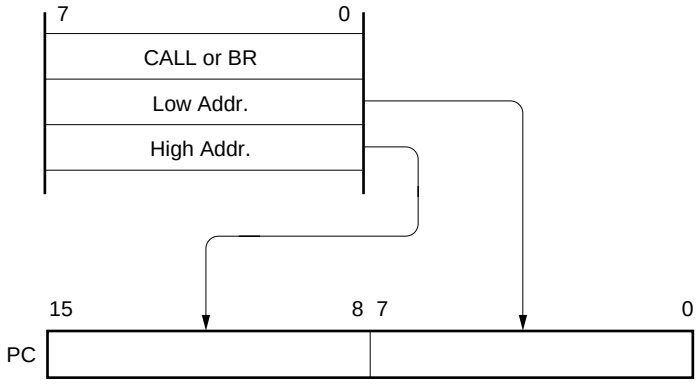
Immediate data in the instruction word is transferred to the program counter (PC) and branched.

This function is carried out when the CALL !addr16 or BR !addr16 instruction is executed.

CALL !addr16 and BR !addr16 instructions can be branched to any location in the memory space.

[Illustration]

In case of CALL !addr16 and BR !addr16 instructions



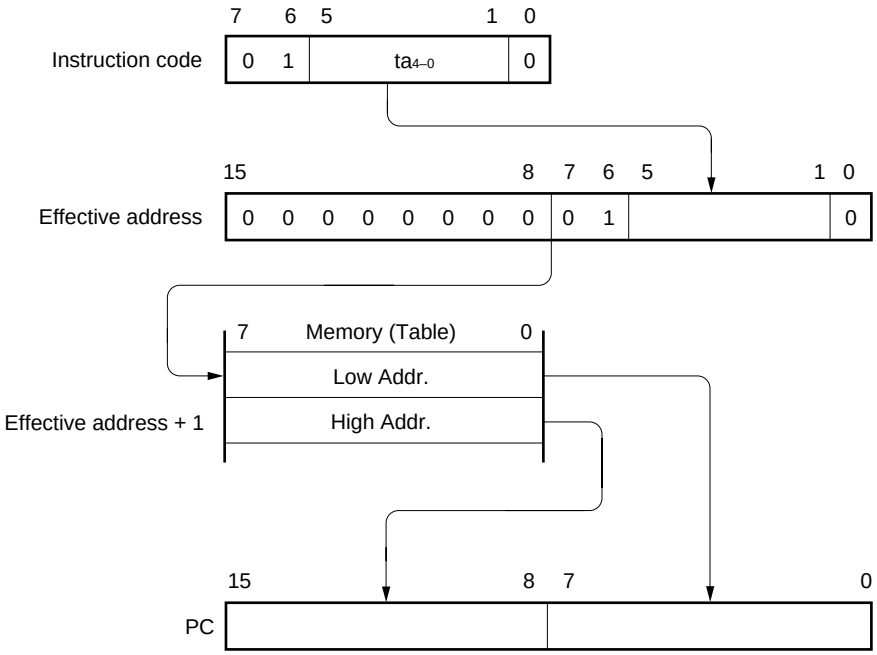
3.3.3 Table indirect addressing

[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.

This function is carried out when the CALLT [addr5] instruction is executed. The instruction enables a branch to any location in the memory space by referring to the addresses stored in the memory table at 40H to 7FH.

[Illustration]



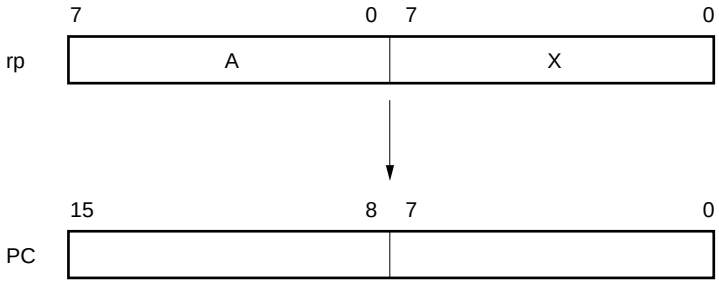
3.3.4 Register addressing

[Function]

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

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

[Illustration]



3.4 Operand Address Addressing

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

[Operand format]

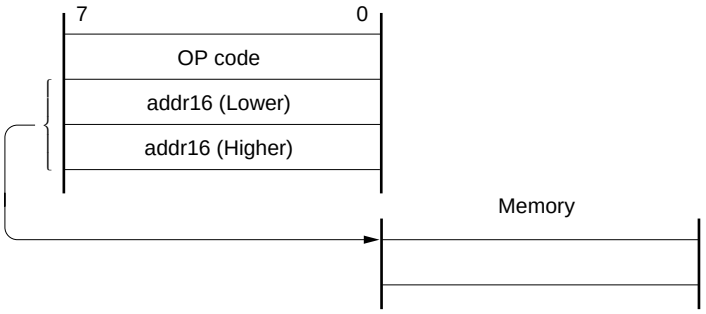
Identifier	Description
addr16	Label or 16-bit immediate data

[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 is the 256-byte space FE20H to FF1FH where the addressing is applied. Internal high-speed RAM and special function registers (SFRs) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.

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

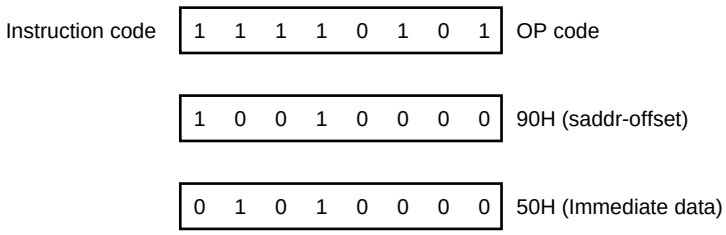
When 8-bit immediate data is at 20H to FFH, bit 8 of an effective address is set to 0. When it is at 00H to 1FH, bit 8 is set to 1. See **[Illustration]** below.

[Operand format]

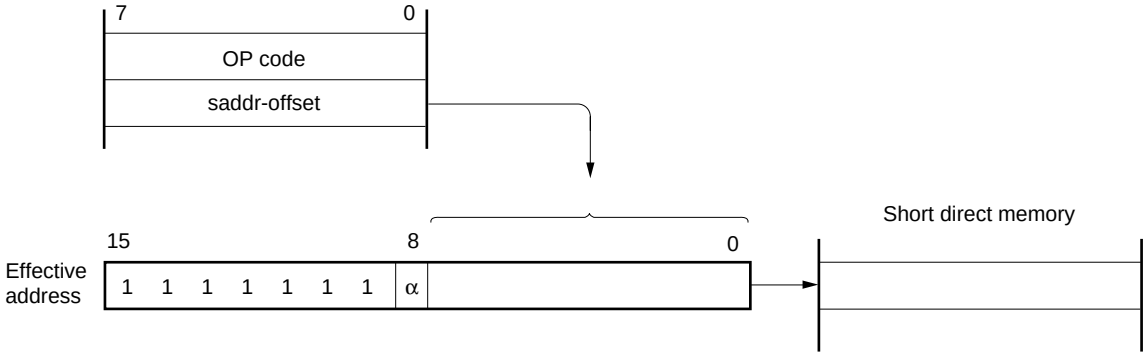
Identifier	Description
saddr	Label or FE20H to FF1FH immediate data
saddrp	Label or FE20H to FF1FH immediate data (even address only)

[Description example]

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



[Illustration]



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

3.4.3 Special function register (SFR) addressing

[Function]

The memory-mapped special function registers (SFRs) are addressed with 8-bit immediate data in an instruction word.

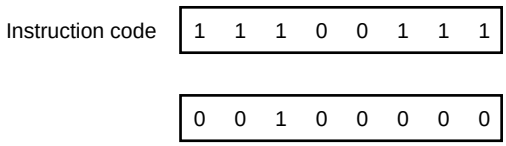
This addressing is applied to the 256-byte space FF00H to FFFFH. However, the SFRs mapped at FF00H to FF1FH can also be accessed with short direct addressing.

[Operand format]

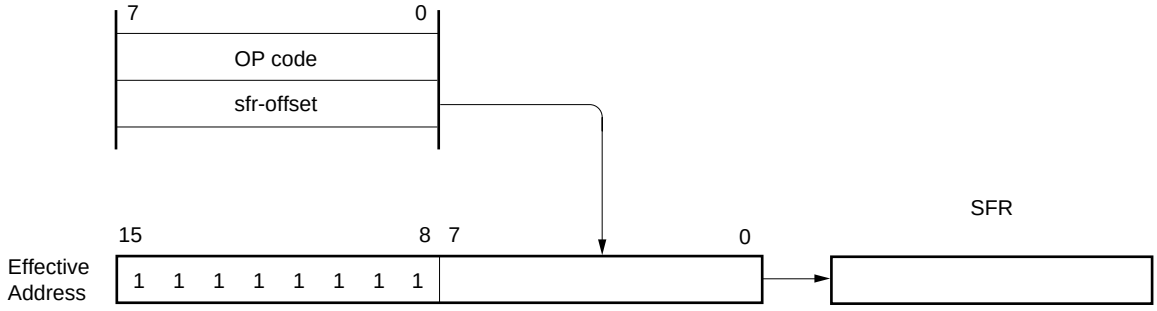
Identifier	Description
sfr	Special function register name

[Description example]

MOV PM0, A; When selecting PM0 for sfr



[Illustration]



3.4.4 Register addressing

[Function]

In the register addressing mode, general-purpose registers are accessed as operands. The general-purpose register to be accessed is specified by a register specification code or 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.

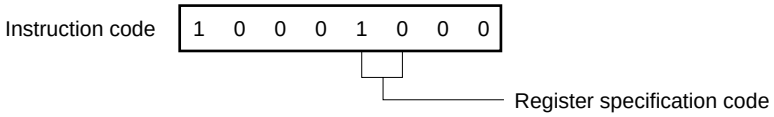
[Operand format]

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

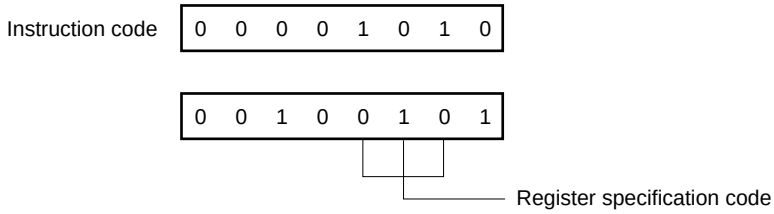
r and rp can be described with absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

[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]

In the register indirect addressing mode, memory is manipulated according to the contents of a register pair specified as an operand. The register pair to be accessed is specified by the register pair specification code in an instruction code.

This addressing can be carried out for all the memory spaces.

[Operand format]

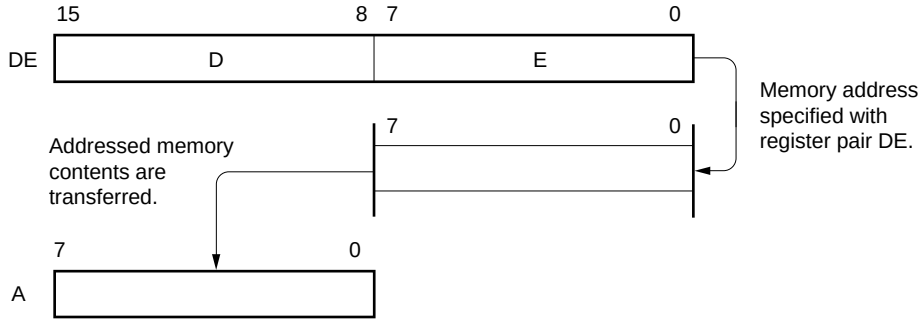
Identifier	Description
–	[DE], [HL]

[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/restored upon generation of an interrupt request.

Only the internal high-speed RAM area can be addressed using stack addressing.

[Description example]

In the case of PUSH DE

Instruction code	1	0	1	0	1	0	1	0

CHAPTER 4 PORT FUNCTIONS

4.1 Port Functions

The μPD789426, 789436, 789446, and 789456 Subseries provide the ports shown in Figures 4-1 and 4-2, enabling various methods of control.

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

Figure 4-1. Port Types (μPD789426, 789436 Subseries)

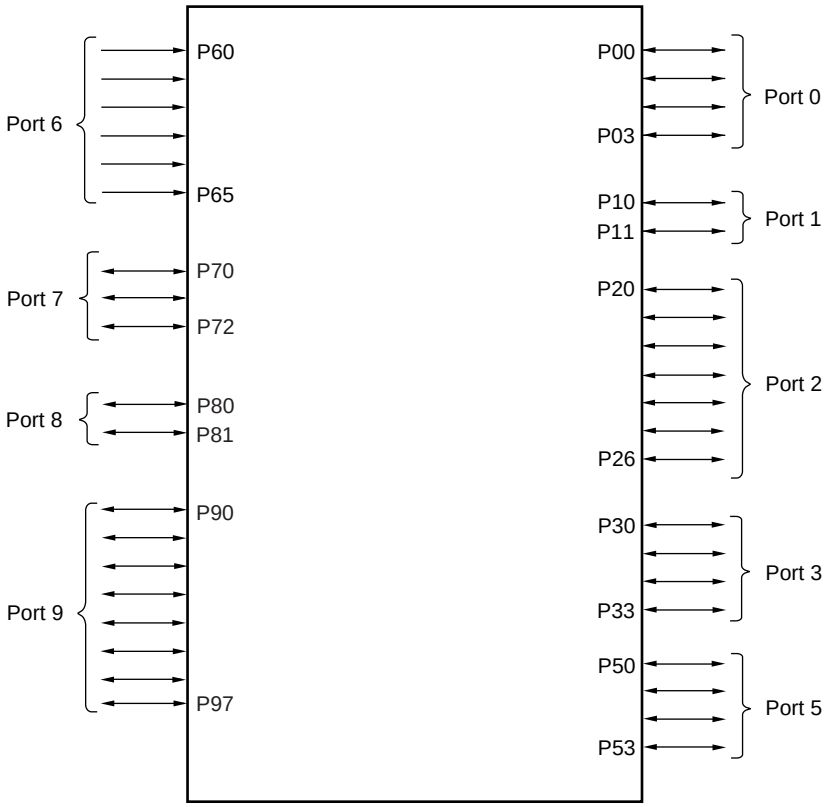


Figure 4-2. Port Types (μPD789446, 789456 Subseries)

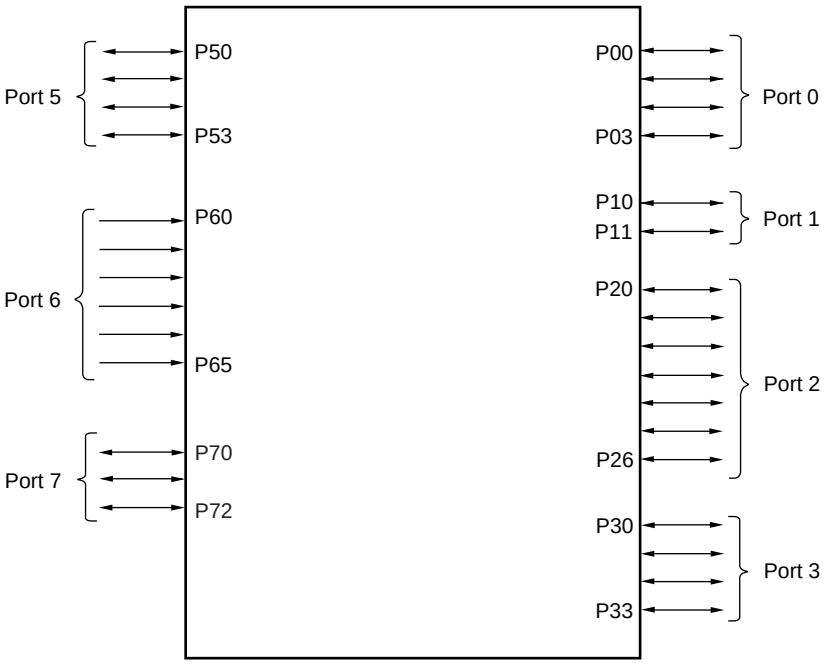


Table 4-1. Port Functions (1/2)

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P03	I/O	Port 0. 4-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) or key return mode register 00 (KRM00).	Input	KR0 to KR3
P10, P11	I/O	Port 1. 2-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 pull-up resistor option register 0 (PU0).	Input	–
P20	I/O	Port 2. 7-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	–
P21				BZO90
P22				$\overline{SS20}$
P23				$\overline{SCK20}$/ASCK20
P24				SO20/TxD20
P25				SI20/RxD20
P26				TO90
P30	I/O	Port 3. 4-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 B3 (PUB3).	Input	INTP0/CPT90
P31				INTP1/TO50/TMI60
P32				INTP2/TO60
P33				INTP3/TO61
P50 to P53	I/O	Port 5. 4-bit I/O port. Input/output can be specified in 1-bit units. For a mask ROM version, an on-chip pull-up resistor can be specified by a mask option.	Input	–
P60 to P65	Input	Port 6. 6-bit input port.	Input	ANI0 to ANI5
P70 to P72	I/O	Port 7. 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 B7 (PUB7).	Input	–

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

Pin Name	I/O	Function	After Reset	Alternate Function
P80, P81[Note]	I/O	Port 8. 2-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 B8 (PUB8).	Input	–
P90 to P97[Note]	I/O	Port 9. 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 B9 (PUB9).	Input	–

Note μPD789426, 789436 Subseries only

4.2 Port Configuration

Ports have the following hardware configuration.

Table 4-2. Configuration of Port

Item		Configuration
Control registers		Port mode register (PMm: m = 0 to 3, 5, 7 to 9) Pull-up resistor option register (PU0, PUB2, PUB3, PUB7 to PUB9)
Ports	μPD789426, 789436 Subseries	Total: 40 (CMOS I/O: 30, CMOS input: 6, N-ch open-drain I/O: 4)
	μPD789446, 789456 Subseries	Total: 30 (CMOS I/O: 20, CMOS input: 6, N-ch open-drain I/O: 4)
Pull-up resistors	μPD789426, 789436 Subseries	Total: 34 (software control: 30, mask option specification: 4)
	μPD789446, 789456 Subseries	Total: 24 (software control: 20, mask option specification: 4)

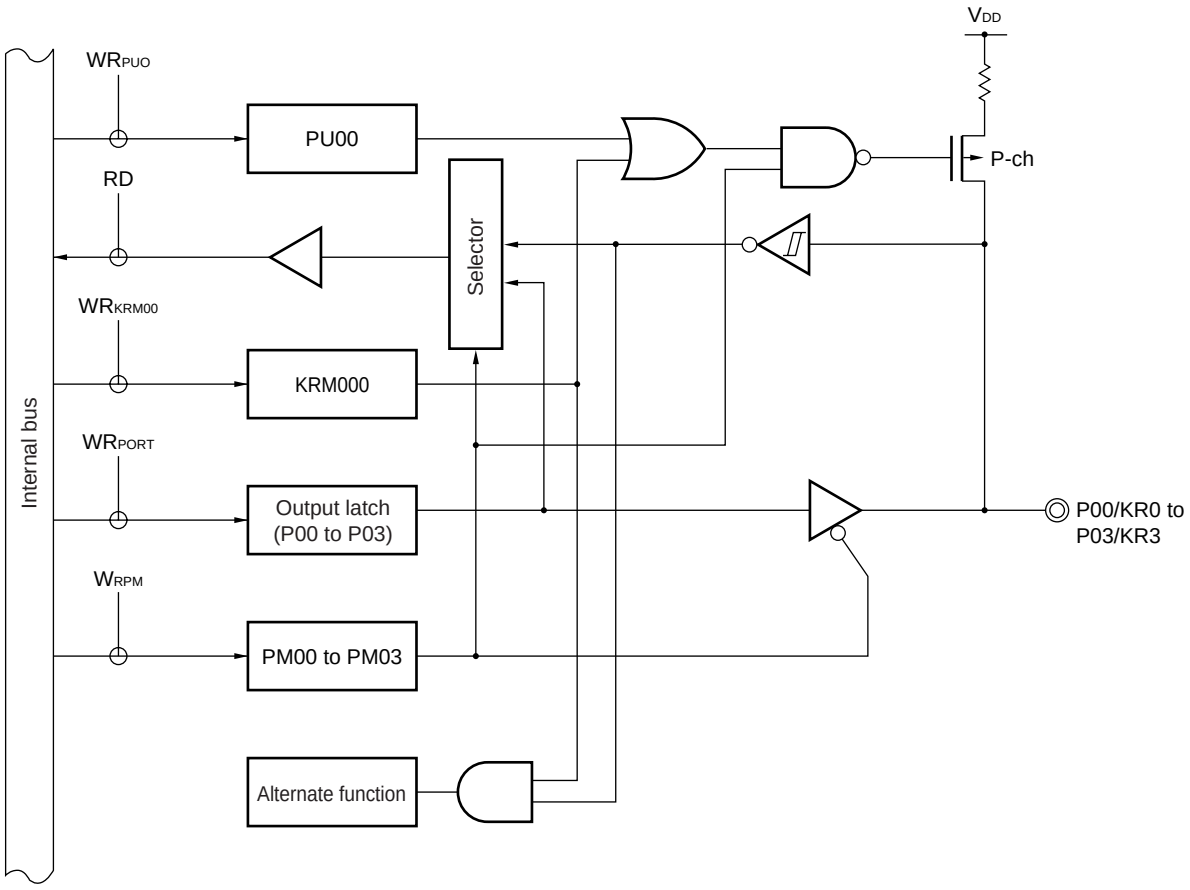
4.2.1 Port 0

This is a 4-bit I/O port with an output latch. Port 0 can be specified in the input or output mode in 1-bit units by using the port mode register 0 (PM0). When the P00 to P03 pins are used as input port pins, on-chip pull-up resistors can be connected in 4-bit units by using pull-up resistor option register 0 (PU0).

Port 0 is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-3 shows a block diagram of port 0.

Figure 4-3. Block Diagram of P00 to P03



KRM00: Key return mode register 00
PU0: Pull-up resistor option register 0
PM: Port mode register
RD: Port 0 read signal
WR: Port 0 write signal

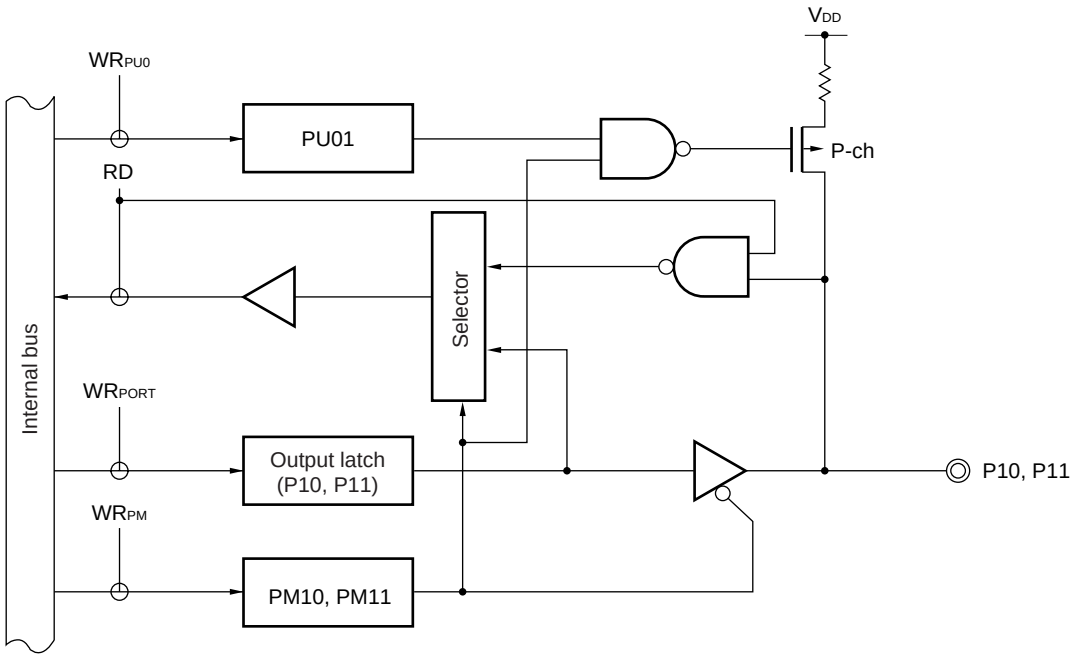
4.2.2 Port 1

This is a 2-bit I/O port with an output latch. Port 1 can be specified in the input or output mode in 1-bit units by using port mode register 1 (PM1). When using the P10 and P11 pins as input port pins, on-chip pull-up resistors can be connected in 2-bit units by using pull-up resistor option register 0 (PU0).

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

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

Figure 4-4. Block Diagram of P10 and P11



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 7-bit I/O port with an output latch. Port 2 can be specified in the input or output mode in 1-bit units by using port mode register 2 (PM2). When using the P20 to P26 pins as input port pins, on-chip pull-up resistors can be connected in 1-bit units by using pull-up resistor option register B2 (PUB2).

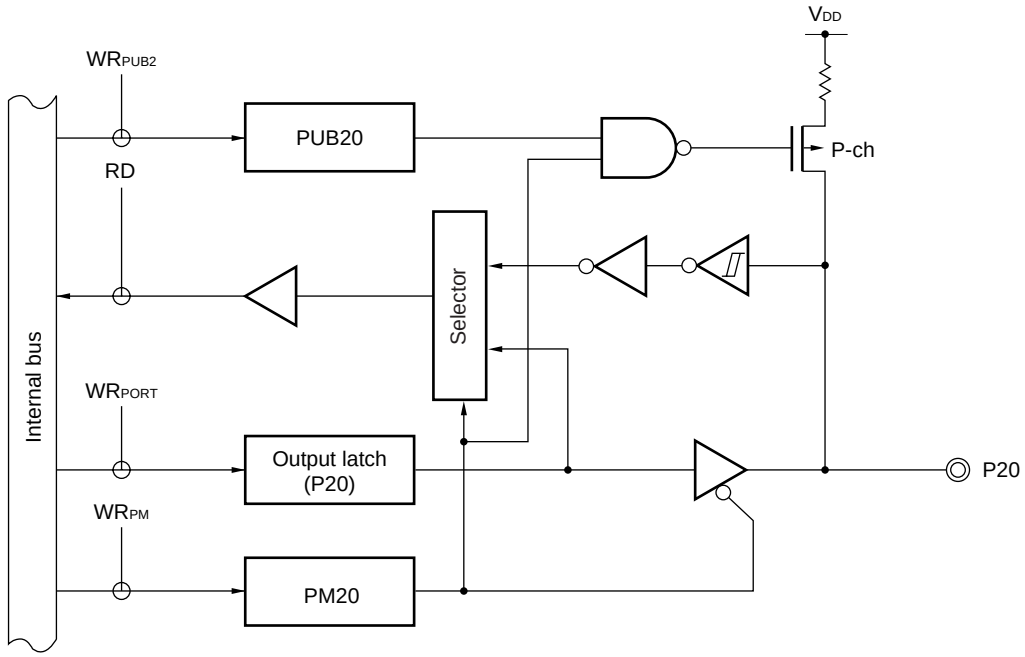
The port is also used as the serial interface I/O, buzzer output, and timer output.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figures 4-5 to 4-10 show block diagrams of port 2.

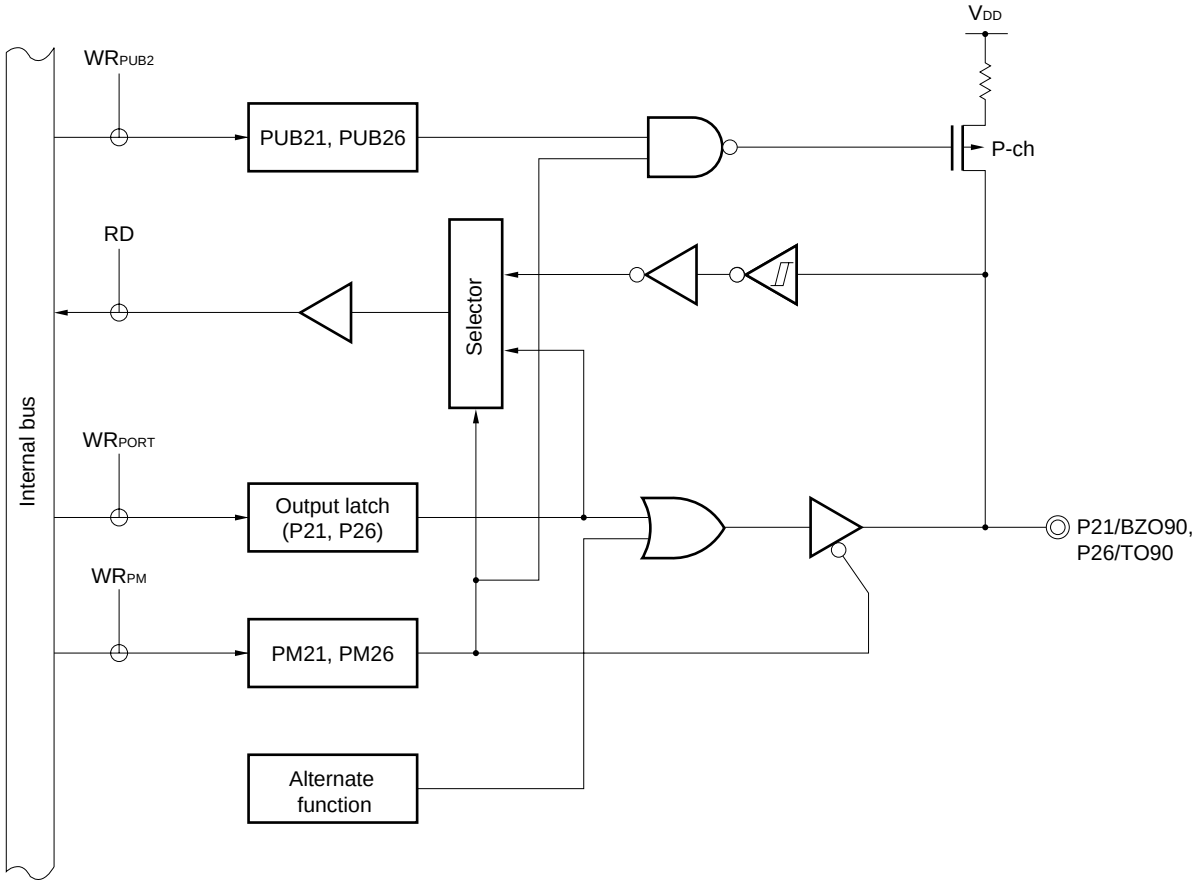
Caution When using the pins of port 2 as the serial interface, the I/O or output latch must be set according to the function to be used. For how to set the latches, see Figure 12-2 Settings of Serial Interface 20 Operating Mode.

Figure 4-5. Block Diagram of P20



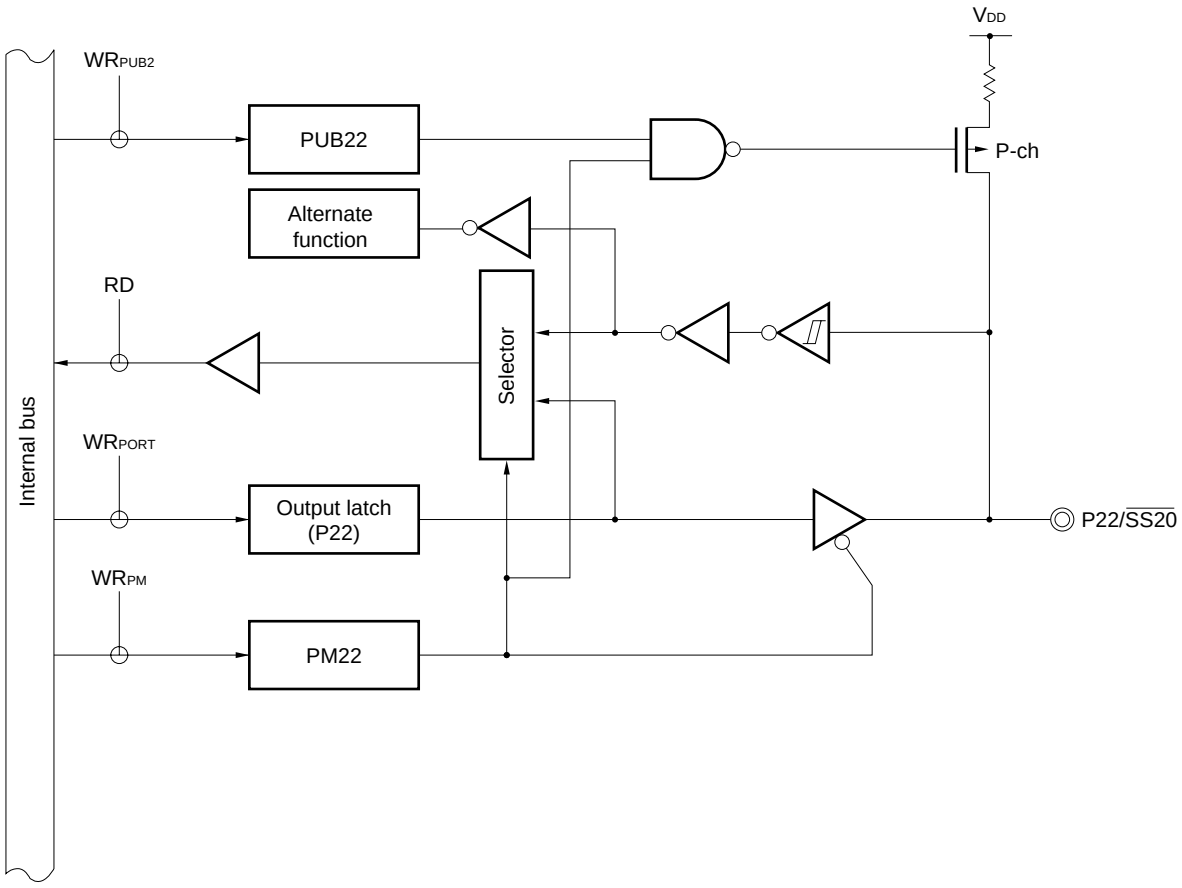
PUB2: Pull-up resistor option register B2
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

Figure 4-6. Block Diagram of P21 and P26



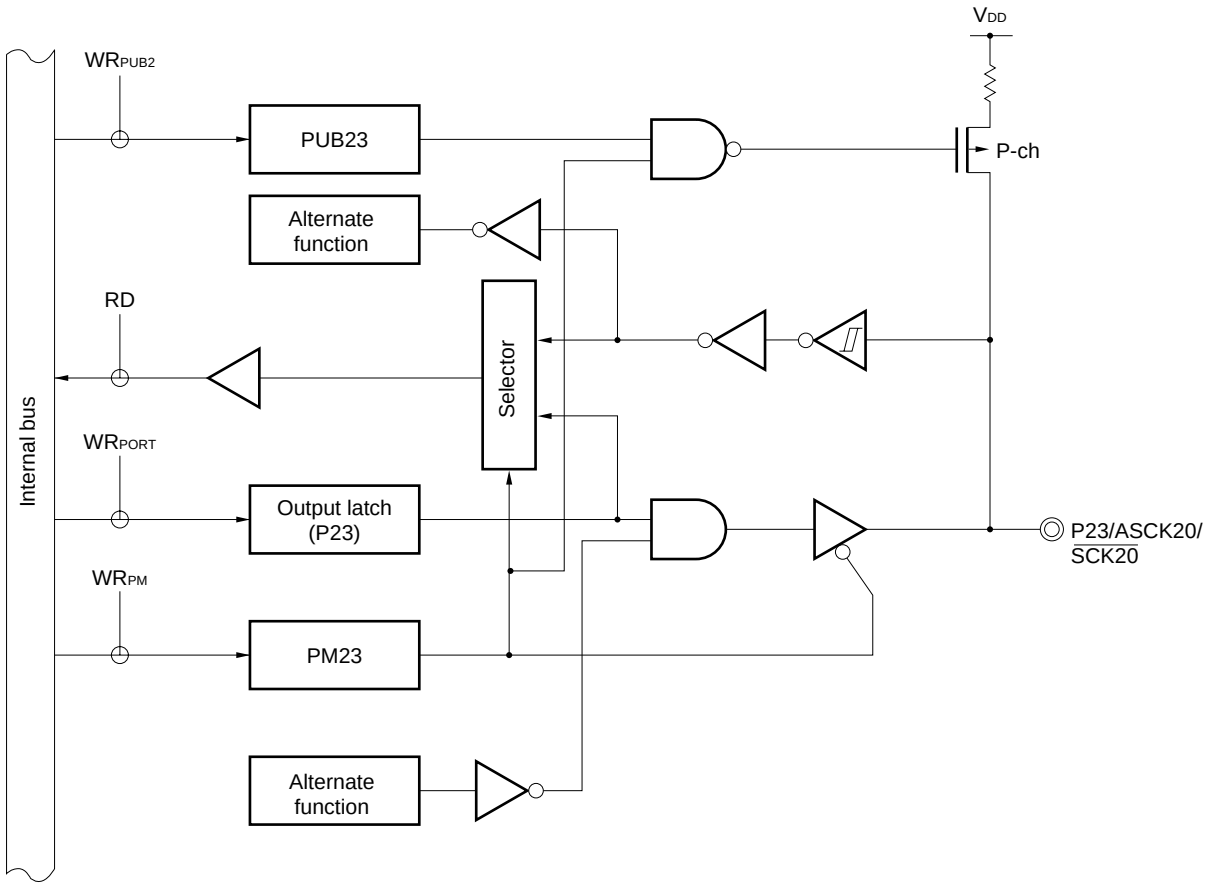
PUB2: Pull-up resistor option register B2
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

Figure 4-7. Block Diagram of P22



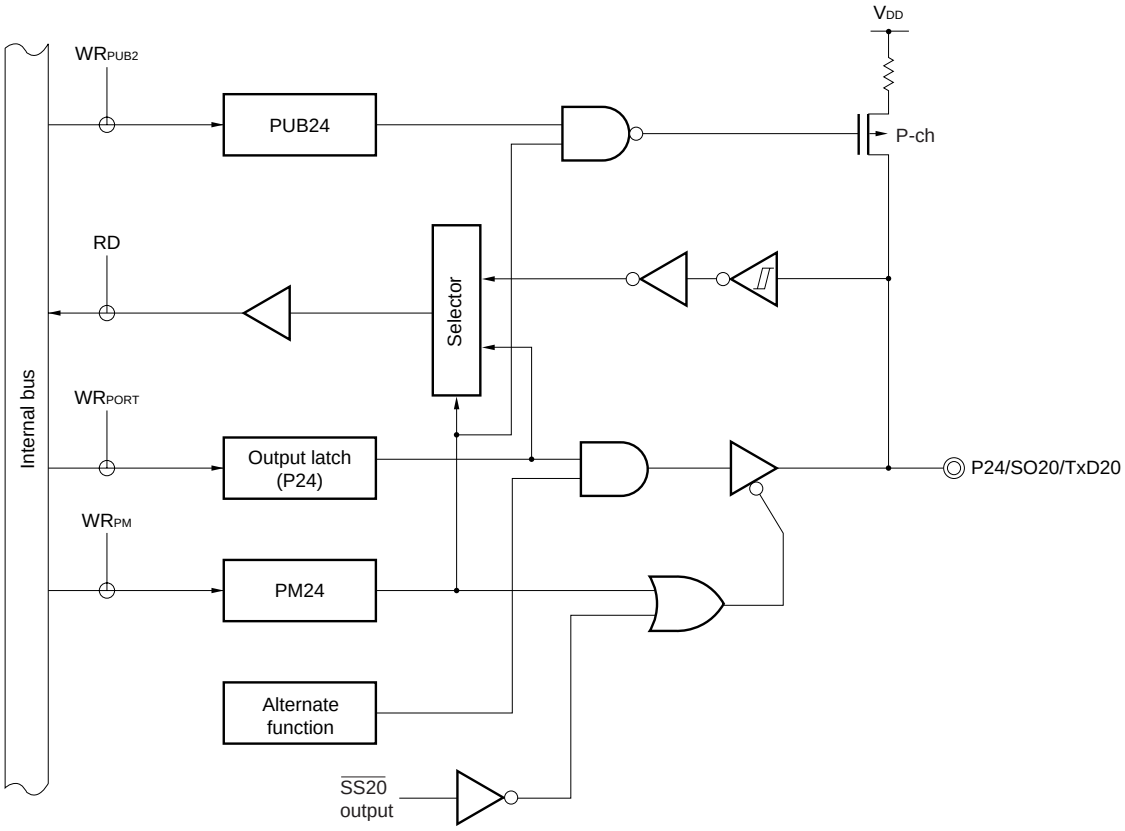
PUB2: Pull-up resistor option register B2
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

Figure 4-8. Block Diagram of P23



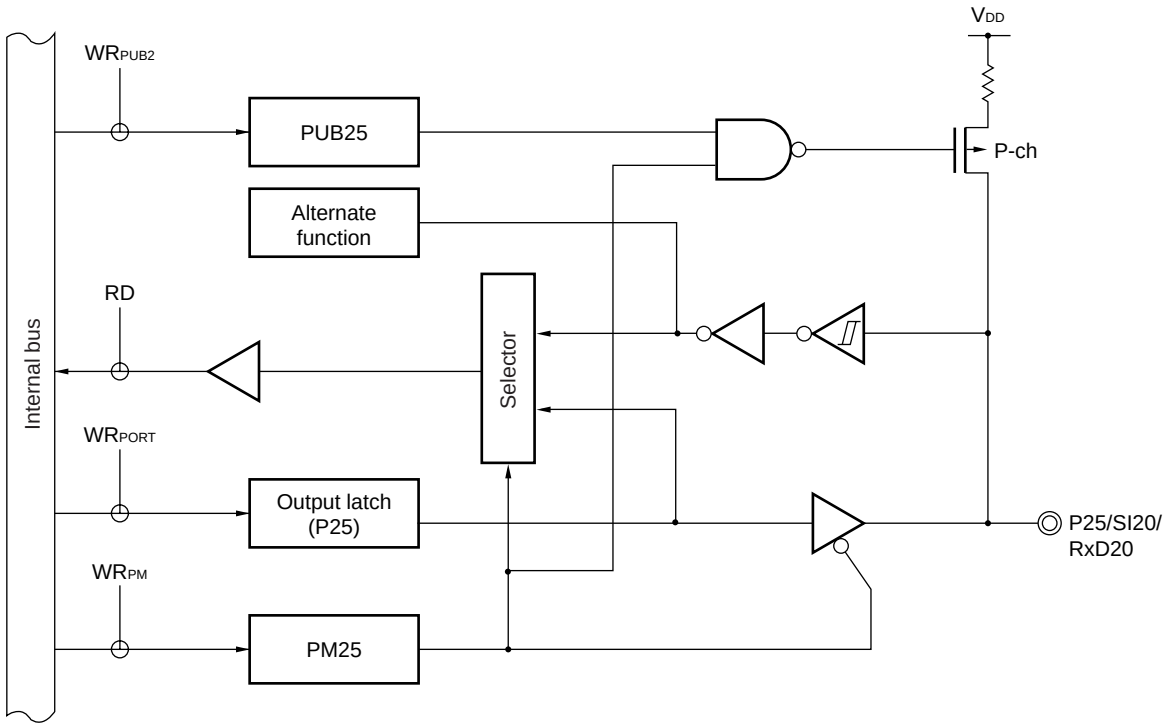
PUB2: Pull-up resistor option register B2
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

Figure 4-9. Block Diagram of P24



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

Figure 4-10. Block Diagram of 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 3

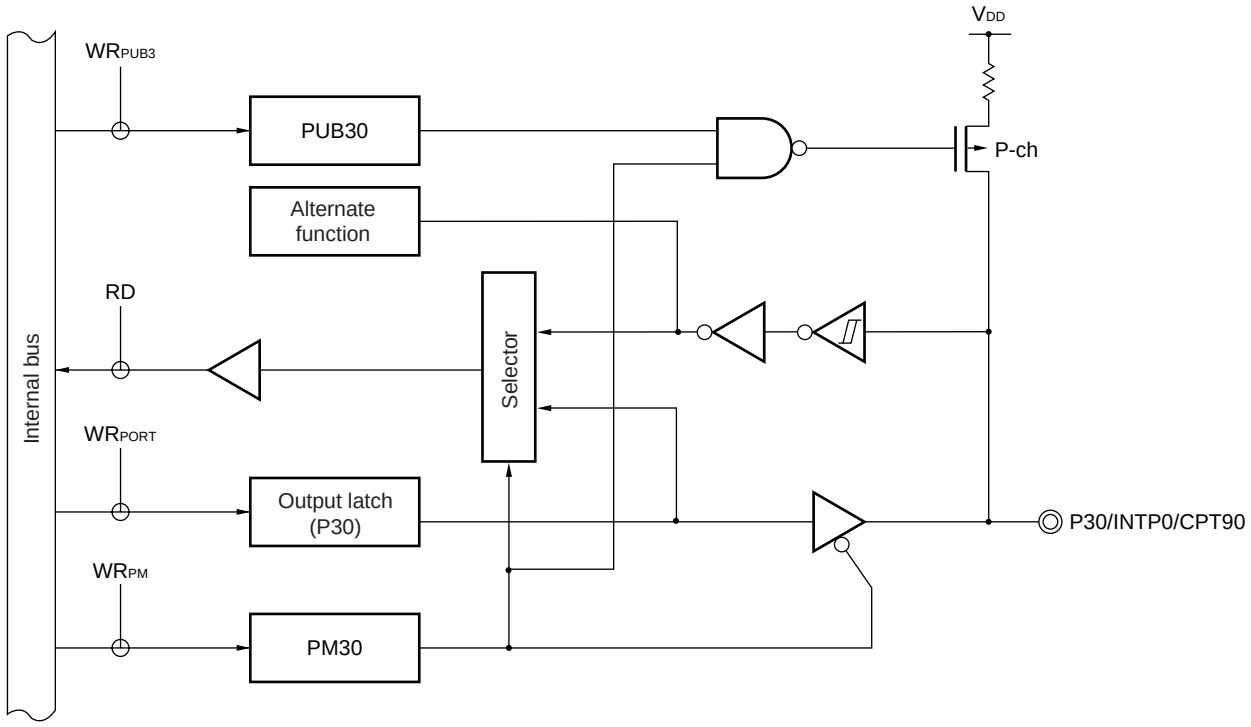
This is a 4-bit I/O port with an output latch. Port 3 can be specified in the input or output mode in 1-bit units by using port mode register 3 (PM3). When using the P30 to P33 pins as input port pins, on-chip pull-up resistors can be connected in 1-bit units by using pull-up resistor option register B3 (PUB3).

This port is also used as an external interrupt input, capture input, and timer I/O.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

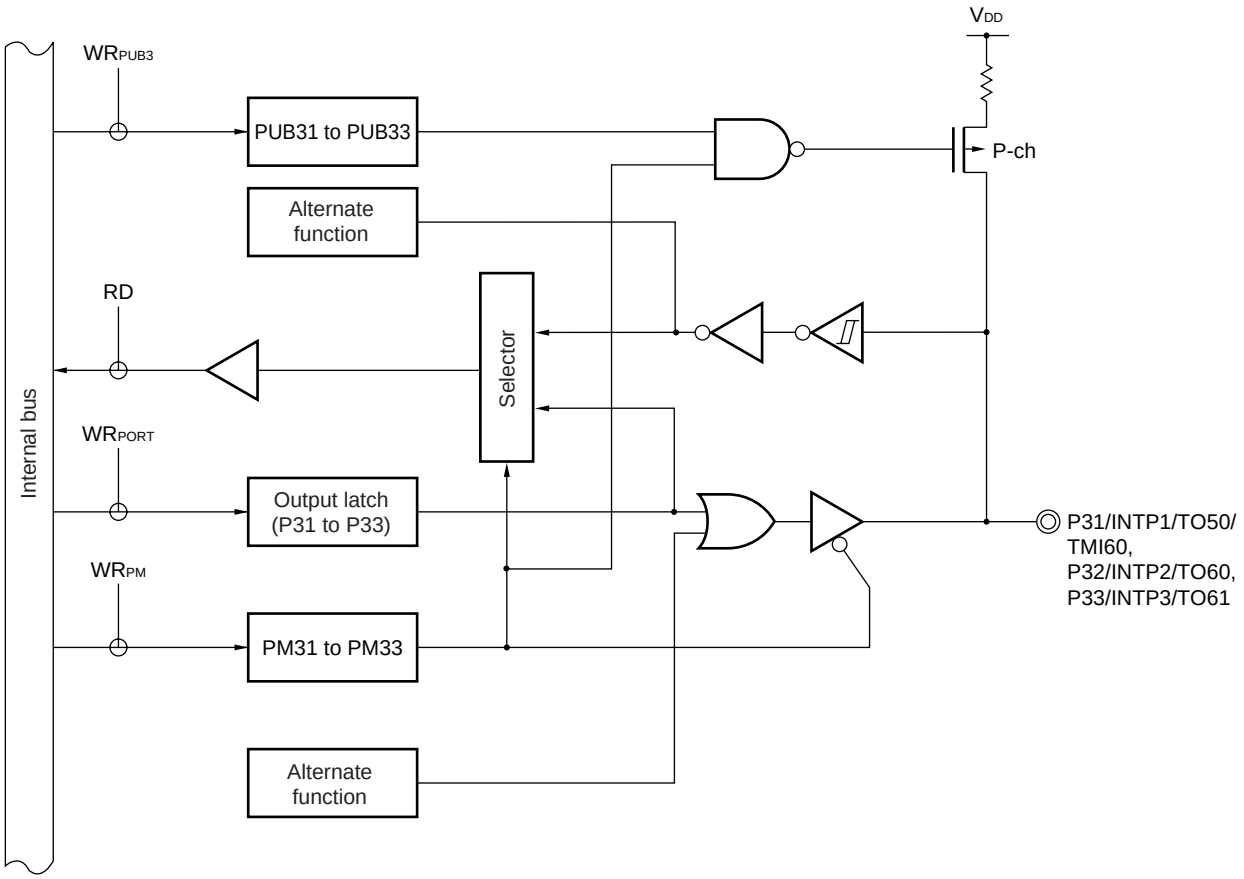
Figures 4-11 and 4-12 show block diagrams of port 3.

Figure 4-11. Block Diagram of P30



PUB3: Pull-up resistor option register B3
PM: Port mode register
RD: Port 3 read signal
WR: Port 3 write signal

Figure 4-12. Block Diagram of P31 to P33



PUB3: Pull-up resistor option register B3
PM: Port mode register
RD: Port 3 read signal
WR: Port 3 write signal

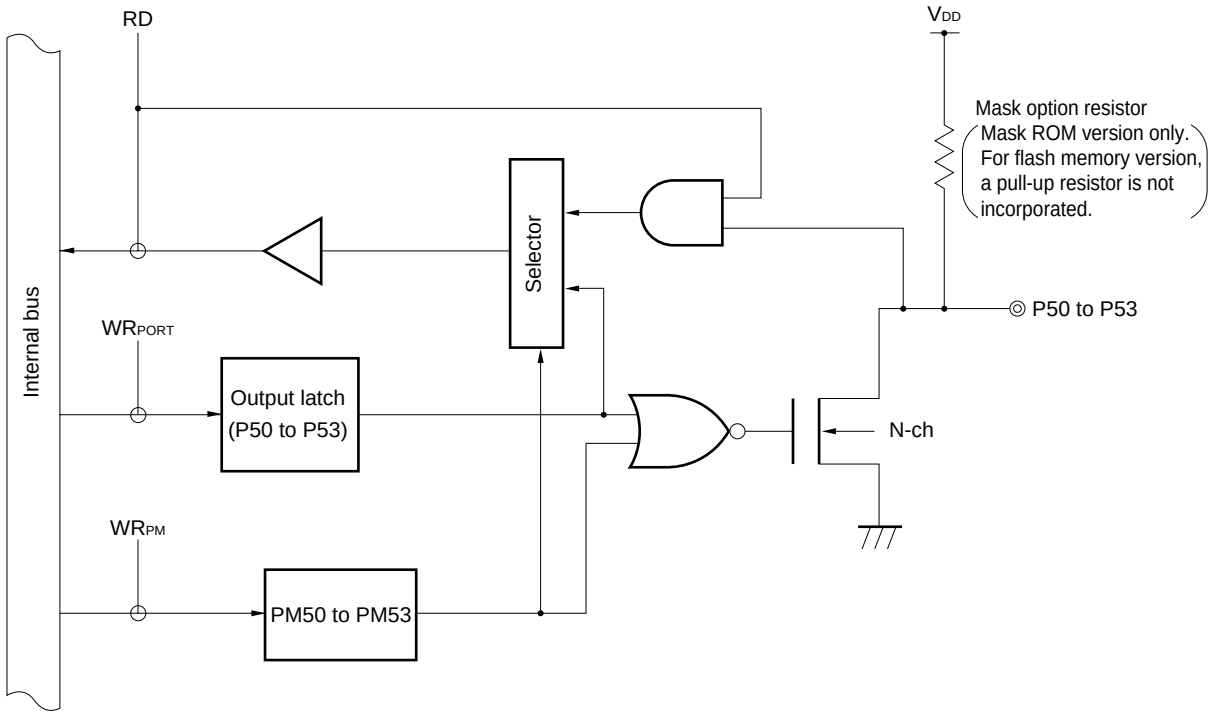
4.2.5 Port 5

This is a 4-bit N-ch open-drain I/O port with an output latch. Port 5 can be specified in the input or output mode in 1-bit units by using port mode register 5 (PM5). For a mask ROM version, use of an on-chip pull-up resistor can be specified by a mask option.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-13 shows a block diagram of port 5.

Figure 4-13. Block Diagram of P50 to P53



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

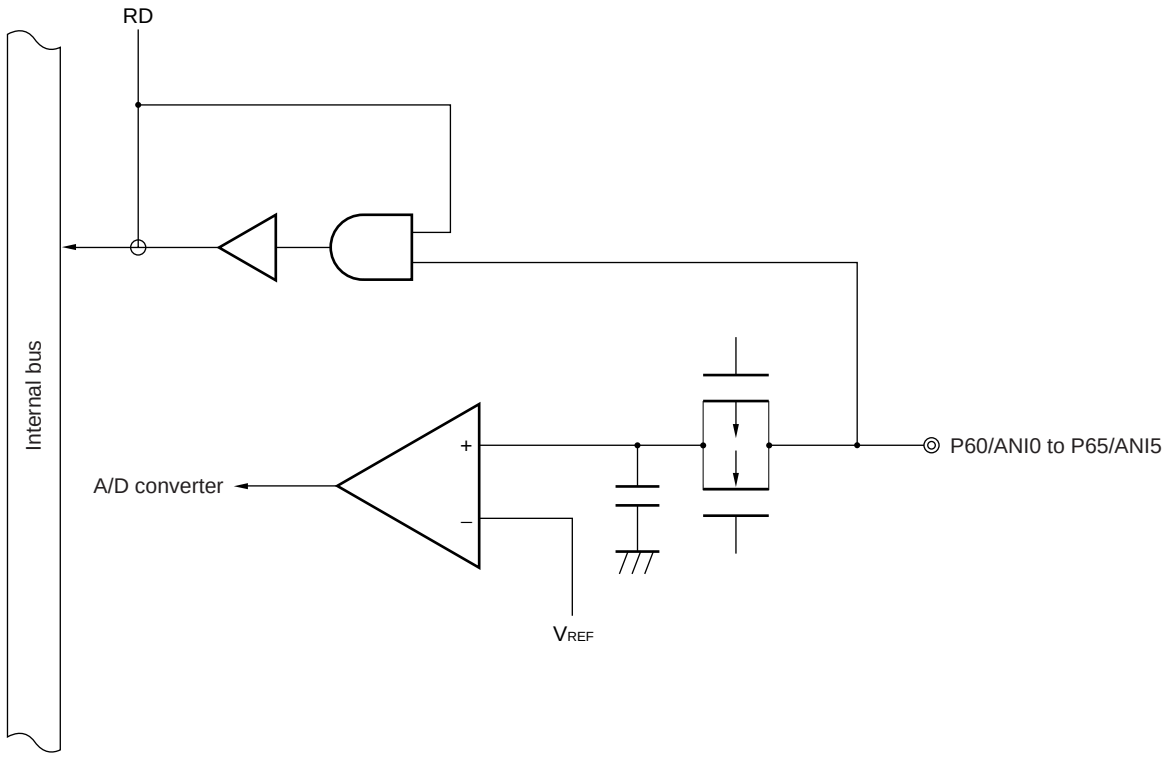
4.2.6 Port 6

This is an 8-bit input-only port.

This port is also used as the analog input of an A/D converter.

Figure 4-14 shows a block diagram of Port 6.

Figure 4-14. Block Diagram of Port 6



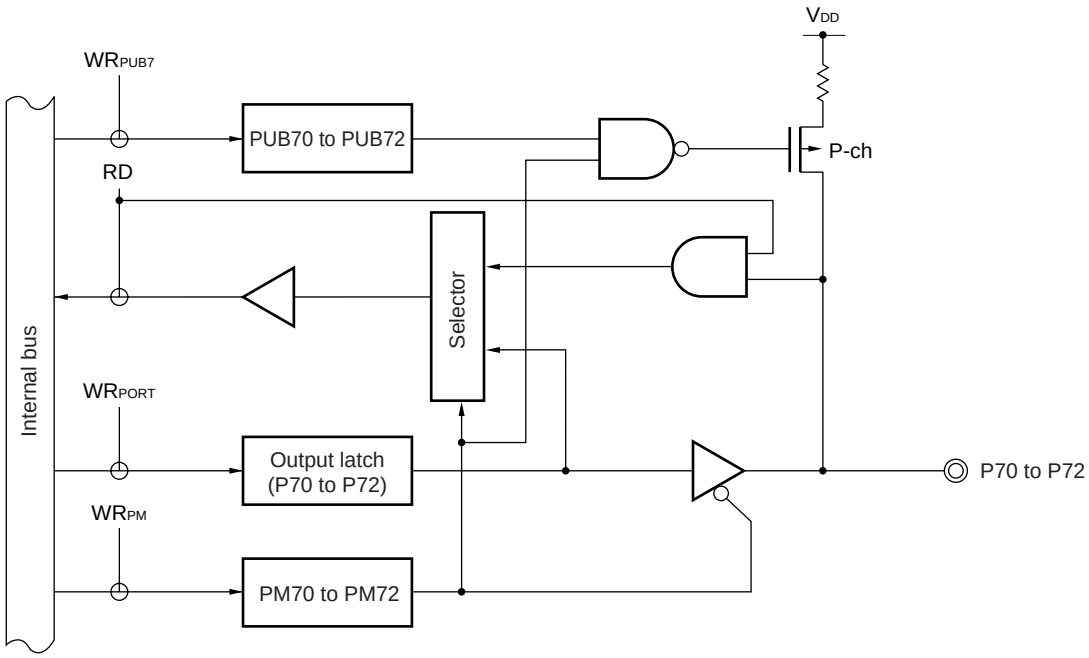
4.2.7 Port 7

This is a 3-bit I/O port with an output latch. Port 7 can be specified in the input or output mode in 1-bit units by using port mode register 7 (PM7). When using the P70 to P72 pins as input port pins, on-chip pull-up resistors can be connected in 1-bit units by using pull-up resistor option register B7 (PUB7).

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-15 shows a block diagram of Port 7.

Figure 4-15. Block Diagram of P70 to P72



PUB7: Pull-up resistor option register B7
PM: Port mode register
RD: Port 7 read signal
WR: Port 7 write signal

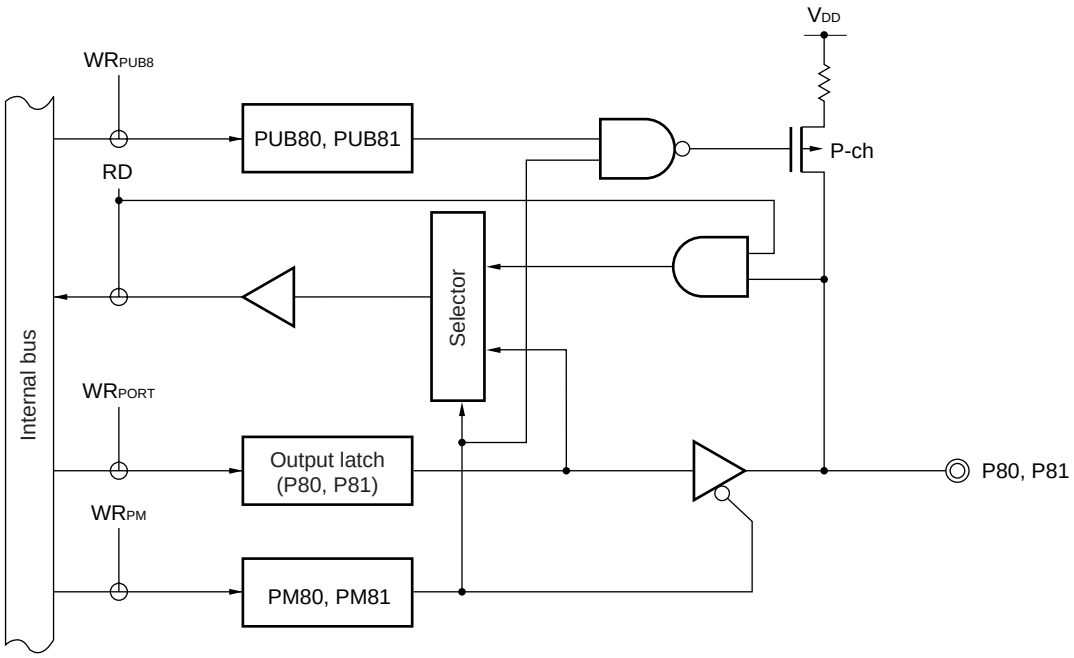
4.2.8 Port 8 (μPD789426, 789436 Subseries only)

This is a 2-bit I/O port with an output latch. Port 8 can be specified in the input or output mode in 1-bit units by using port mode register 8 (PM8). When using pins P80 and P81 as input port pins, on-chip pull-up resistors can be connected in 1-bit units by using pull-up resistor option register B8 (PUB8).

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-16 shows a block diagram of port 8.

Figure 4-16. Block Diagram of P80 and P81



PUB8: Pull-up resistor option register B8
PM: Port mode register
RD: Port 8 read signal
WR: Port 8 write signal

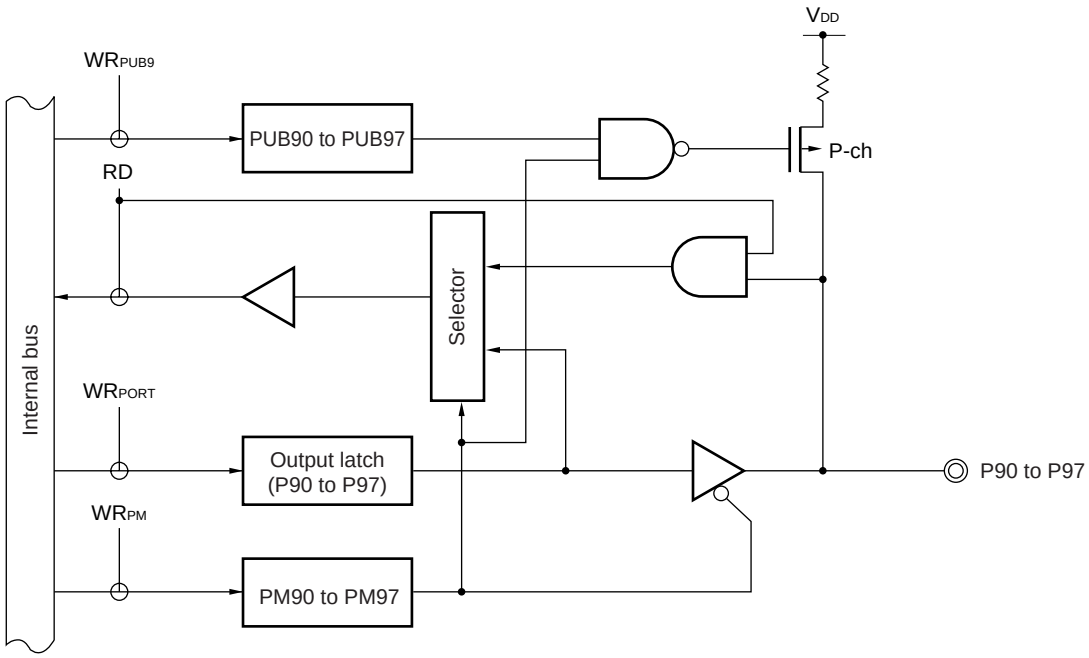
4.2.9 Port 9 (μPD789426, 789436 Subseries only)

This is an 8-bit I/O port with an output latch. Port 9 can be specified in the input or output mode in 1-bit units by using port mode register 9 (PM9). When using the pins of this port as input port pins, on-chip pull-up resistors can be connected in 1-bit units by using pull-up resistor option register B9 (PUB9).

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-17 shows a block diagram of port 9.

Figure 4-17. Block Diagram of P90 to P97



PUB9: Pull-up resistor option register B9
PM: Port mode register
RD: Port 9 read signal
WR: Port 9 write signal

4.3 Registers Controlling Port Function

The ports are controlled by the following two types of registers.

- Port mode registers (PM0 to PM3, PM5, PM7 to PM9)
- Pull-up resistor option registers (PU0, PUB2, PUB3, PUB7 to PUB9)

(1) Port mode registers (PM0 to PM3, PM5, PM7 to PM9)

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

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

$\overline{\text{RESET}}$ input sets the 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 3 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 preset to 1.

Figure 4-18. Format of Port Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	1	1	1	1	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM1	1	1	1	1	PM13	PM12	PM11	PM10	FF21H	FFH	R/W
PM2	1	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W
PM3	1	1	1	1	PM33	PM32	PM31	PM30	FF23H	FFH	R/W
PM5	1	1	1	1	PM53	PM52	PM51	PM50	FF25H	FFH	R/W
PM7	1	1	1	1	1	PM72	PM71	PM70	FF27H	FFH	R/W
PM8[Note]	1	1	1	1	1	1	PM81	PM80	FF28H	FFH	R/W
PM9[Note]	PM97	PM96	PM95	PM94	PM93	PM92	PM91	PM90	FF29H	FFH	R/W

PMmn	Pmn pin input/output mode selection (m = 0 to 3, 5, 7 to 9, n = 0 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

Note Incorporated only in the μPD789426 and 789436 Subseries.

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

Pin Name	Alternate Function		PMxx	Pxx
	Name	I/O		
P00 to P03	KR0 to KR3	Input	1	x
P26	TO90	Output	0	0
P30	INTP0	Input	1	x
	CPT90	Input	1	x
P31	INTP1	Input	1	x
	TO50	Output	0	0
	TMI60	Input	1	x
P32	INTP2	Input	1	x
	TO60	Output	0	0
P33	INTP3	Input	1	x
	TO61	Output	0	0
P60 to P65	ANI0 to ANI5	Input	1	x

Caution When port 2 is used as a serial interface pin, the I/O latch or output latch must be set according to its function. For the setting method, see Table 12-2 Settings of Serial Interface 20 Operating Mode.

Remark x: don't care
PMxx: Port mode register
Pxx: Port output latch

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

Pull-up resistor option register 0 (PU0) sets whether on-chip pull-up registers are used on ports 0 and 1 or not.

On the port specified to use an on-chip pull-up resistor by PU0, the pull-up resistor can be internally used only for the bits set in the input mode. No on-chip pull-up resistors can be used for the bits set in the output mode regardless of the setting of PU0. This also applies to cases when the pins are used for alternate functions.

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

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

Figure 4-19. Format of Pull-Up Resistor Option Register 0

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

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

Caution Bits 2 to 7 must be set to 0.

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

Pull-up resistor option register B2 (PUB2) sets whether on-chip pull-up resistors on P20 to P26 are used or not.

On the port specified to use an on-chip pull-up resistor by PUB2, the pull-up resistor can be internally used only for the bits set in the input mode. No on-chip pull-up resistors can be used for the bits set in the output mode regardless of the setting of PUB2. This also applies to cases when the pins are used for alternate functions.

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

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

Figure 4-20. Format of Pull-Up Resistor Option Register B2

Symbol	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
PUB2	0	PUB26	PUB25	PUB24	PUB23	PUB22	PUB21	PUB20	FF32H	00H	R/W

PUB2n	P2n on-chip pull-up resistor selection (n = 0 to 6)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

(4) Pull-up resistor option register B3 (PUB3)

Pull-up resistor option register B3 (PUB3) sets whether on-chip pull-up resistors on P30 to P33 are used or not.

On the port specified to use an on-chip pull-up resistor by PUB3, the pull-up resistor can be internally used only for the bits set in the input mode. No on-chip pull-up resistors can be used for the bits set in the output mode regardless of the setting of PUB3. This also applies to cases when the pins are used for alternate functions.

PUB3 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PUB3 to 00H.

Figure 4-21. Format of Pull-Up Resistor Option Register B3

Symbol	7	6	5	4	<3>	<2>	<1>	<0>	Address	After reset	R/W
PUB3	0	0	0	0	PUB33	PUB32	PUB31	PUB30	FF33H	00H	R/W

PUB3n	P3n on-chip pull-up resistor selection (n = 0 to 3)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

(5) Pull-up resistor option register B7 (PUB7)

Pull-up resistor option register B7 (PUB7) sets whether on-chip pull-up resistors on P70 to P72 are used or not. On the port specified to use an on-chip pull-up resistor by PUB7, the pull-up resistor can be internally used only for bits set in the input mode. No on-chip pull-up resistors can be used for the bits set in the output mode regardless of the setting of PUB7. This also applies to when the pins are used for alternate function.

PUB7 is set with a 1-bit or 8-bit memory manipulation instructions.

$\overline{\text{RESET}}$ input sets PUB7 to 00H.

Figure 4-22. Format of Pull-Up Resistor Option Register B7

Symbol	7	6	5	4	3	<2>	<1>	<0>	Address	After reset	R/W
PUB7	0	0	0	0	0	PUB72	PUB71	PUB70	FF37H	00H	R/W

PUB7n	P7n on-chip pull-up resistor selection (n = 0 to 2)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

(6) Pull-up resistor option register B8 (PUB8)[Note]

Pull-up resistor option register B8 (PUB8) sets whether on-chip pull-up resistors on P80 and P81 are used or not. On the port specified to use an on-chip pull-up resistor by PUB8, the pull-up resistor can be internally used only for bits set in the input mode. No on-chip pull-up resistors can be used for the bits set in the output mode regardless of the setting of PUB8. This also applies to when the pins are used for alternate functions.

PUB8 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PUB8 to 00H.

Note Incorporated only in the μPD789426 and 789436 Subseries.

Figure 4-23. Format of Pull-Up Resistor Option Register B8

Symbol	7	6	5	4	3	2	<1>	<0>	Address	After reset	R/W
PUB8	0	0	0	0	0	0	PUB81	PUB80	FF38H	00H	R/W

PUB8n	P8n on-chip pull-up resistor selection (n = 0, 1)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

(7) Pull-up resistor option register B9 (PUB9)[Note]

Pull-up resistor option register B9 (PUB9) sets whether on-chip pull-up resistors on P90 to P97 are used or not. On the port specified to use an on-chip pull-up resistor by PUB9, the pull-up resistor can be internally used only for bits set in the input mode. No on-chip pull-up resistors can be used for the bits set in the output mode regardless of the setting of PUB9. This also applies to when the pins are used for alternate function.

PUB9 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PUB9 to 00H.

Note Incorporated only in the μPD789426 and 789436 Subseries.

Figure 4-24. Format of Pull-Up Resistor Option Register B9

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
PUB9	PUB97	PUB96	PUB95	PUB94	PUB93	PUB92	PUB91	PUB90	FF39H	00H	R/W

PUB9n	P9n on-chip pull-up resistor selection (n = 0 to 7)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

4.4 Port Function Operation

The operation of a port differs depending on whether the port is set in the 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.

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.

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 status of an 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.

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 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).

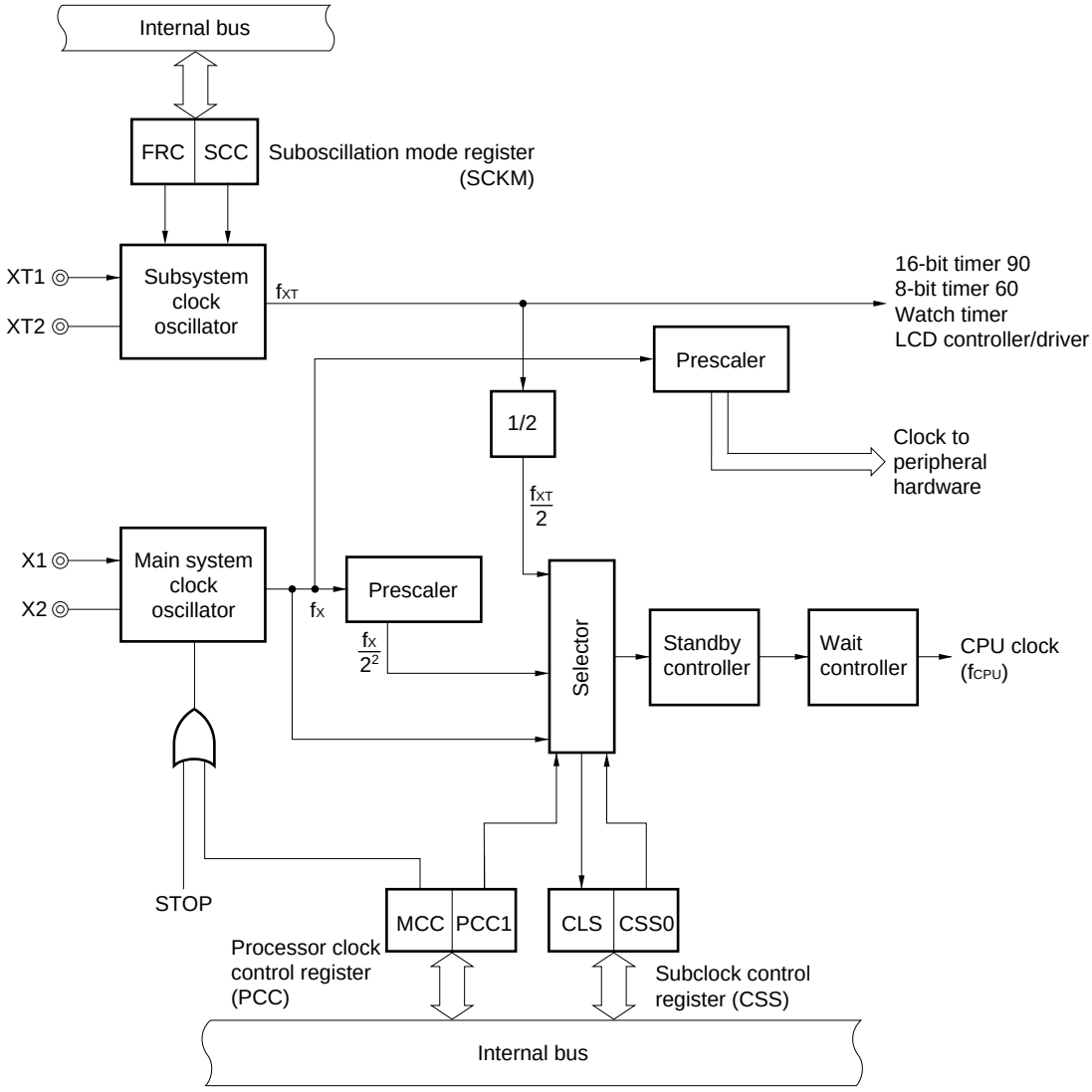
5.2 Clock Generator Configuration

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 Registers 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 CPU clock selection and the division ratio.

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

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

Figure 5-2. Format of Processor Clock Control Register

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
PCC	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 disabled

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. Bits 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 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 sets SCKM to 00H.

Figure 5-3. Format of Suboscillation Mode Register

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 disabled

Caution Bits 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 sets CSS to 00H.

Figure 5-4. Format of Subclock Control Register

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 output of the (divided) main system clock
1	Operation based on the subsystem clock

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 is 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

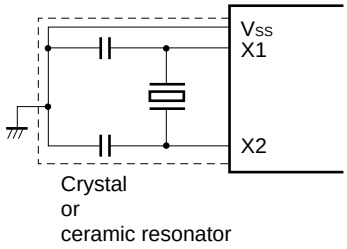
The main system clock oscillator is oscillated by the crystal or ceramic resonator (5.0 MHz TYP.) connected across the X1 and X2 pins.

An external clock can also be input to the circuit. In this case, input the clock signal to the X1 pin, and input the inverted signal to the X2 pin.

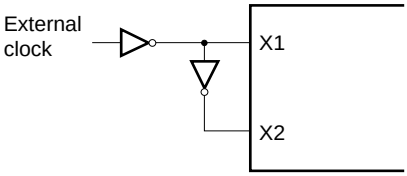
Figure 5-5 shows the external circuit of the main system clock oscillator.

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

(a) Crystal or ceramic oscillation



(b) External clock



5.4.2 Subsystem clock oscillator

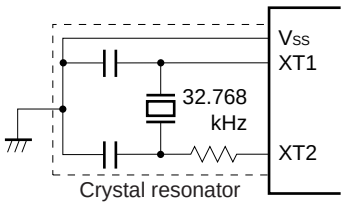
The subsystem clock oscillator is oscillated by the crystal resonator (32.768 kHz TYP.) connected across the XT1 and XT2 pins.

An external clock can also be input to the circuit. In this case, input the clock signal to the XT1 pin, and input the inverted signal to the XT2 pin.

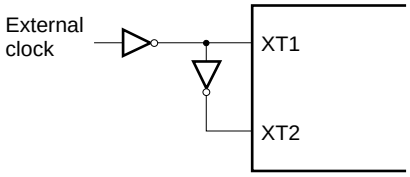
Figure 5-6 shows the external circuit of the subsystem clock oscillator.

Figure 5-6. External Circuit of Subsystem Clock Oscillator

(a) Crystal oscillation



(b) External clock



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.

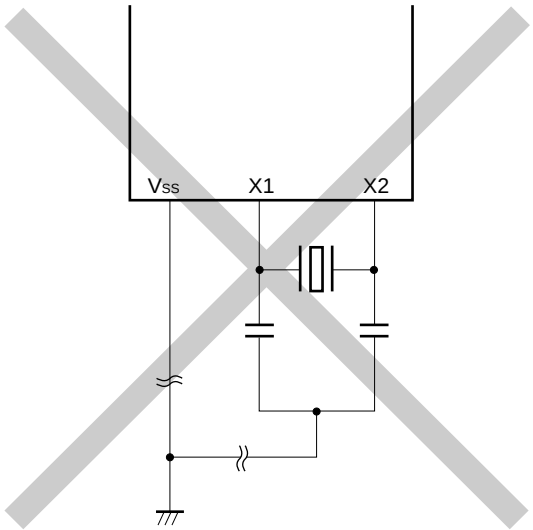
- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS}. Do not ground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

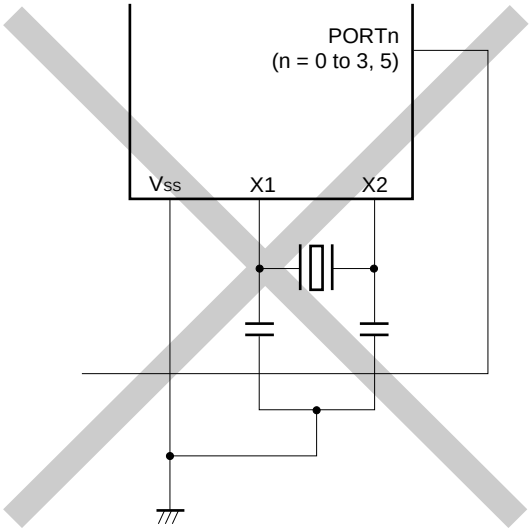
Figure 5-7 shows examples of incorrect resonator connection.

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

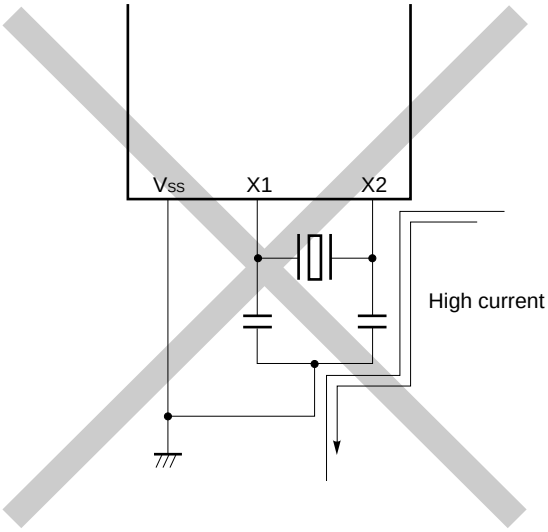
(a) Too long wiring



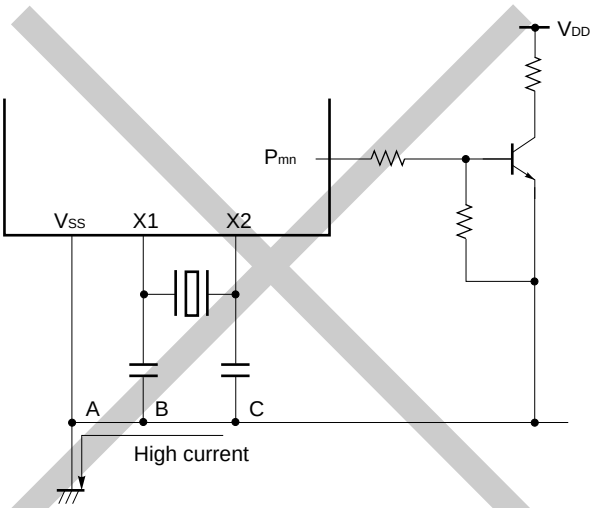
(b) Crossed signal line



(c) Wiring near high fluctuating current



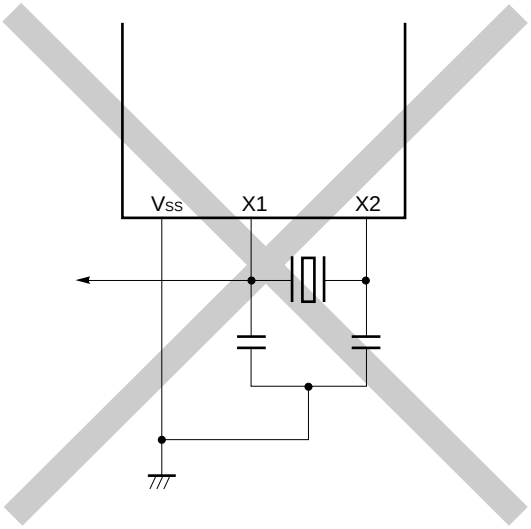
(d) Current flowing through ground line of oscillator (potential at points A, B, and C fluctuates)



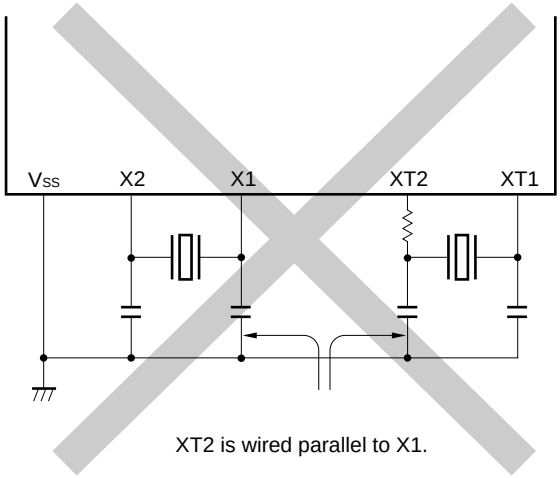
Remark When using the subsystem clock, read X1 and X2 as XT1 and XT2, respectively, and connect a resistor to XT2 in series.

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 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 Divider circuit

The divider circuit divides the output of the main system clock oscillator (f_X) 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_{SS}
XT2: Leave open

In this case, however, a small current leaks 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 operation 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 bit 1 (FRC) of the SCKM 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 16-bit timer, 8-bit timer, watch timer, and LCD controller/driver. The 16-bit timer, 8-bit timer, watch timer, and LCD controller/driver can therefore keep running even during standby. The other hardware stops when the main system clock stops because it runs based on the main system clock (except for 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	x
0	0			4 clocks		$2f_X/f_{XT}$ clocks (306 clocks)	
	1	2 clocks				$f_X/2f_{XT}$ clocks (76 clocks)	
1	x	2 clocks		2 clocks			

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

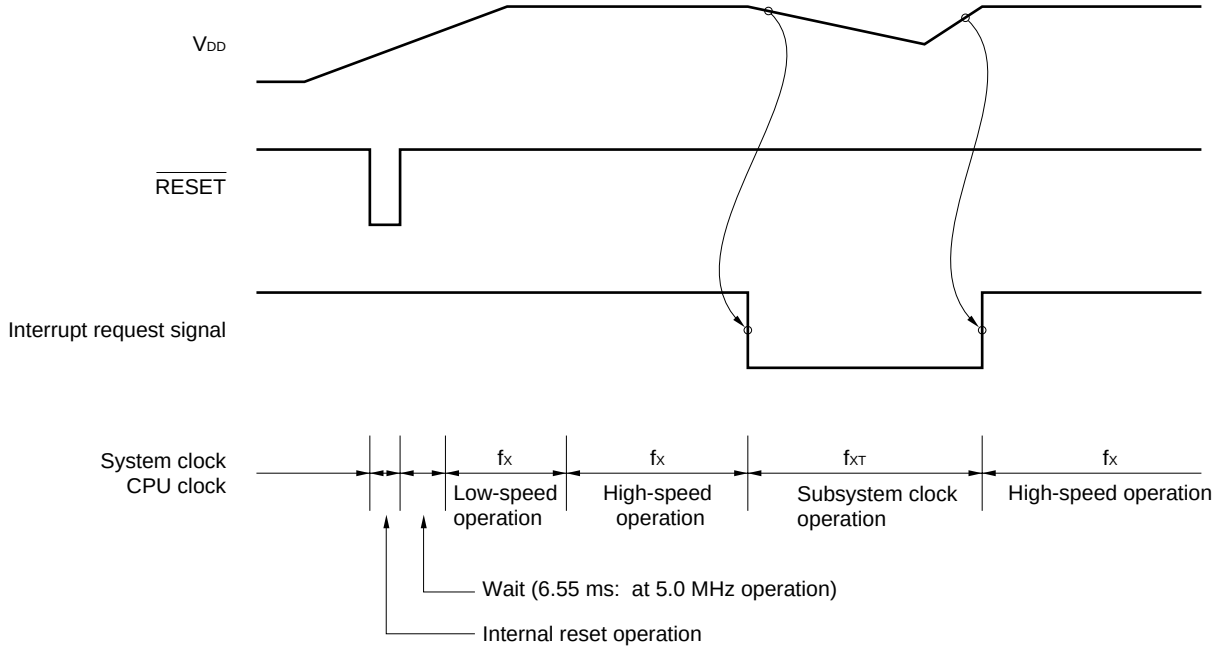
2. The parenthesized values apply to operation at f_X = 5.0 MHz or f_{XT} = 32.768 kHz.

3. x: 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 effect of resetting is released when the $\overline{\text{RESET}}$ pin is later 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 slow 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 status).

<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 16-BIT TIMER

6.1 16-Bit Timer Functions

The 16-bit timer has the following functions.

- Timer interrupt
- Timer output
- Buzzer output
- Count value capture

(1) Timer interrupt

An interrupt is generated when a count value and compare value matches.

(2) Timer output

Timer output can be controlled when a count value and compare value matches.

(3) Buzzer output

Buzzer output can be controlled by software.

(4) Count value capture

A count value of 16-bit timer counter 90 (TM90) is latched into a capture register synchronizing with the capture trigger and retained.

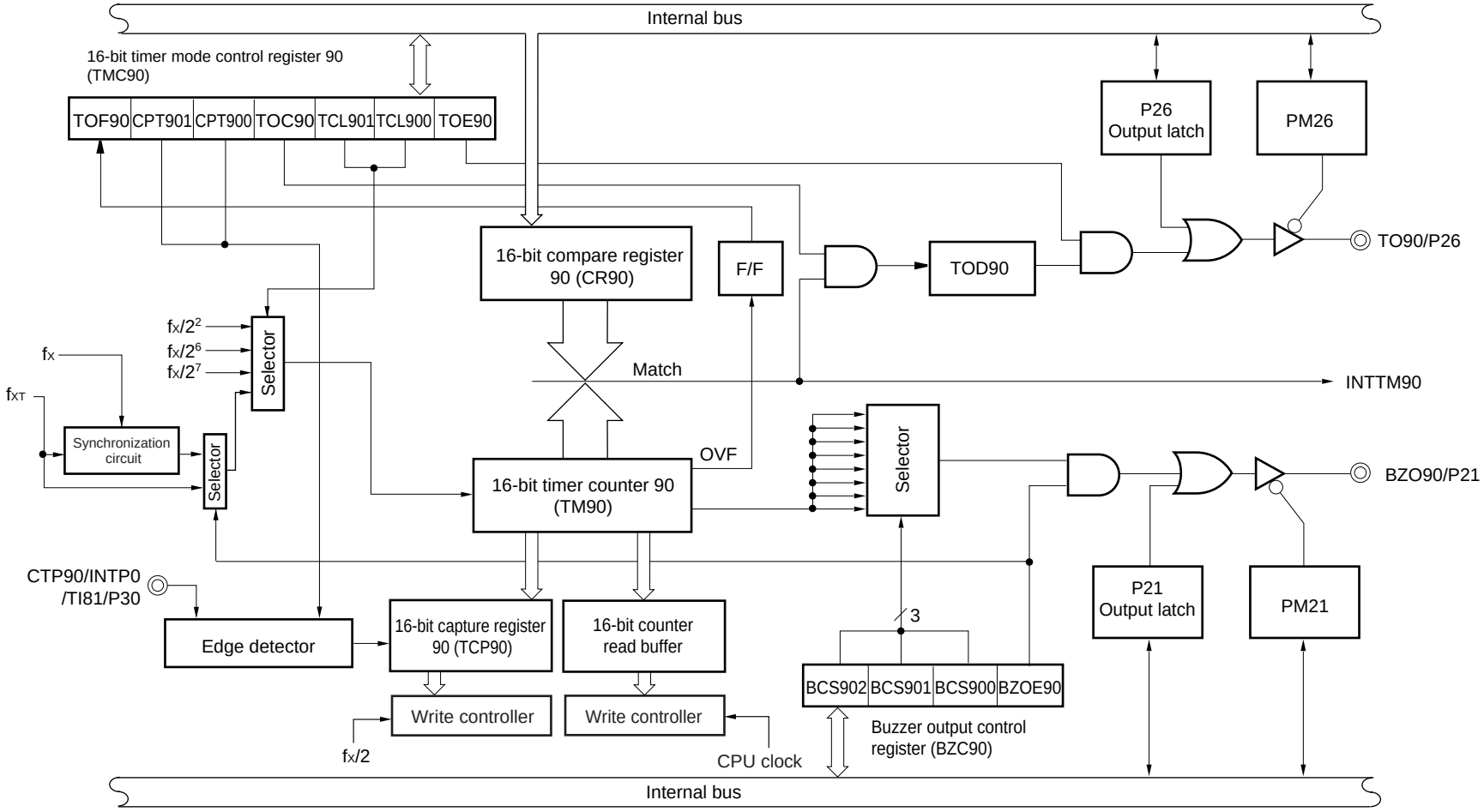
6.2 16-Bit Timer Configuration

The 16-bit timer includes the following hardware.

Table 6-1. 16-Bit Timer Configuration

Item	Configuration
Timer counters	16 bits × 1 (TM90)
Registers	Compare register: 16 bits × 1 (CR90) Capture register: 16 bits × 1 (TCP90)
Timer outputs	1 (TO90)
Control registers	16-bit timer mode control register 90 (TMC90) Buzzer output control register 90 (BZC90) Port mode register 2 (PM2)

Figure 6-1. Block Diagram of 16-Bit Timer



(1) 16-bit compare register 90 (CR90)

A value specified in CR90 is compared with the count in 16-bit timer register 90 (TM90). If they match, an interrupt request (INTTM90) is issued by CR90.

CR90 is set with an 8-bit or 16-bit memory manipulation instruction. Any value from 0000H to FFFFH can be set.

$\overline{\text{RESET}}$ input sets CR90 to FFFFH.

Cautions 1. CR90 is designed to be manipulated with a 16-bit memory manipulation instruction. However, it can also be manipulated with an 8-bit memory manipulation instruction. When an 8-bit memory manipulation instruction is used to set CR90, it must be accessed by direct addressing.

2. To overwrite CR90 during a count operation, it is necessary to disable interrupts in advance, using interrupt mask flag register 1 (MK1). It is also necessary to disable inversion of the timer output data, using 16-bit timer mode control register 90 (TMC90). If the value in CR90 is rewritten in the interrupt-enabled state, an interrupt request may occur at the moment of rewrite.

(2) 16-bit timer counter 90 (TM90)

TM90 is used to count the number of pulses.

The contents of TM90 are read with an 8-bit or 16-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TM90 to 0000H.

Cautions 1. The count becomes undefined when STOP mode is deselected, because the count operation is performed before oscillation stabilizes.

2. TM90 is designed to be manipulated with a 16-bit memory manipulation instruction. However, it can also be manipulated with an 8-bit memory manipulation instruction. When an 8-bit memory instruction is used to manipulate TM90, it must be accessed by direct addressing.

3. When an 8-bit memory manipulation instruction is used to manipulate TM90, the lower and higher bytes must be read as a pair, in this order.

(3) 16-bit capture register 90 (TCP90)

TCP90 captures the contents of TM90.

It is set with an 8-bit or 16-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes TCP90 undefined.

Caution TCP90 is designed to be manipulated with a 16-bit memory manipulation instruction. However, it can also be manipulated with an 8-bit memory manipulation instruction. When an 8-bit memory manipulation instruction is used to manipulate TCP90, it must be accessed by direct addressing.

(4) 16-bit counter read buffer 90

This buffer is used to latch and hold the count value for TM90.

6.3 Registers Controlling 16-Bit Timer

The 16-bit timer is controlled by the following three registers.

- 16-bit timer mode control register 90 (TMC90)
- Buzzer output control register 90 (BZC90)
- Port mode register 2 (PM2)

(1) 16-bit timer mode control register 90 (TMC90)

16-bit timer mode control register 90 (TMC90) controls the setting of a count clock, capture edge, etc.

TMC90 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC90 to 00H.

Figure 6-2. Format of 16-Bit Timer Mode Control Register 90

Symbol	<7>	<6>	5	4	3	2	1	<0>	Address	After reset	R/W
TMC90	TOD90	TOF90	CPT901	CPT900	TOC90	TCL901	TCL900	TOE90	FF48H	00H	R/W[Note]

TOD90	Timer output data
0	Timer output data is "0"
1	Timer output data is "1"

TOF90	Overflow flag control
0	Reset or cleared by software
1	Set when the 16-bit timer overflows

CPT901	CPT900	Capture edge selection
0	0	Capture operation disabled
0	1	Captured at the rising edge of the CPT90 pin
1	0	Captured at the falling edge of the CPT90 pin
1	1	Captured at both the rising and falling edges of the CPT90 pin

TOC90	Timer output data inversion control
0	Inversion disabled
1	Inversion enabled

TCL901	TCL900	16-bit timer counter 90 count clock selection
0	0	$f_X/2^2$ (1.25 MHz)
0	1	$f_X/2^6$ (78.1 kHz)
1	0	$f_X/2^7$ (39.1 kHz)
1	1	f_{XT} (32.768 kHz)

TOE90	16-bit timer counter 90 output control
0	Output disabled (port mode)
1	Output enabled

Note Bit 7 is read-only.

Caution Disable interrupts in advance by using the interrupt mask flag register (MK1) to change the data of TCL901 and TCL900. Also, prevent the timer output data from being inverted by setting TOC90 to1.

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.

(2) Buzzer output control register 90 (BZC90)

This register selects a buzzer frequency based on fcl selected with the count clock select bits (TCL901 and TCL900), and controls the output of the square wave.

BZC90 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BZC90 to 00H.

Figure 6-3. Format of Buzzer Output Control Register 90

Symbol	7	6	5	4	3	2	1	<0>	Address	After reset	R/W
BZC90	0	0	0	0	BCS902	BCS901	BCS900	BZOE90	FF49H	00H	R/W[Note]

BCS902	BCS901	BCS900	Buzzer frequency			
			fcl = $f_X/2^2$	fcl = $f_X/2^6$	fcl = $f_X/2^7$	fcl = f_{XT}
0	0	0	fcl/2^4 (78.1 kHz)	fcl/2^4 (4.88 kHz)	fcl/2^4 (2.44 kHz)	fcl/2^4 (2.05 kHz)
0	0	1	fcl/2^5 (39.1 kHz)	fcl/2^5 (2.44 kHz)	fcl/2^5 (1.22 kHz)	fcl/2^5 (1.02 kHz)
0	1	0	fcl/2^8 (4.88 kHz)	fcl/2^8 (305 Hz)	fcl/2^8 (153 Hz)	fcl/2^8 (128 Hz)
0	1	1	fcl/2^9 (2.44 kHz)	fcl/2^9 (153 Hz)	fcl/2^9 (76 Hz)	fcl/2^9 (64 Hz)
1	0	0	fcl/2^{10} (1.22 kHz)	fcl/2^{10} (76 Hz)	fcl/2^{10} (38 Hz)	fcl/2^{10} (32 Hz)
1	0	1	fcl/2^{11} (610 Hz)	fcl/2^{11} (38 Hz)	fcl/2^{11} (19 Hz)	fcl/2^{11} (16 Hz)
1	1	0	fcl/2^{12} (305 Hz)	fcl/2^{12} (19 Hz)	fcl/2^{12} (10 Hz)	fcl/2^{12} (8 Hz)
1	1	1	fcl/2^{13} (153 Hz)	fcl/2^{13} (10 Hz)	fcl/2^{13} (5 Hz)	fcl/2^{13} (4 Hz)

BZOE90	Buzzer port output control
0	Disables buzzer port output.
1	Enables buzzer port output.

Note Bits 4 to 7 must be set to 0.

Caution If the subclock is selected as the count clock (TCL901 = 1, TCL900 = 1: see Figure 6-2 Format of 16-Bit Timer Mode Control Register 90), the subclock is not synchronized when buzzer port output is enabled. In this case, the capture function and TM90 read function are disabled. In addition, the count value of TM90 is undefined.

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.

(3) Port mode register 2 (PM2)

PM2 is used to set each bit of port 3 to input or output.

When pin P26/ITO90 is used for timer output, reset the output latch of P26 and PM26 to 0; when pin P21/BZO90 is used for buzzer output, reset the output latch of P26 and PM26 to 0.

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

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

Figure 6-4. Format of Port Mode Register 2

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM2	1	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W

PM2n	P2n pin I/O mode (n =1, 6)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

6.4 16-Bit Timer Operation

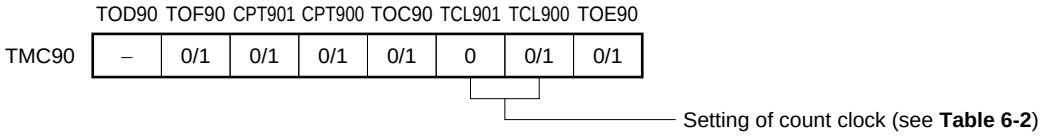
6.4.1 Operation as timer interrupt

In the timer interrupt function, interrupts are repeatedly generated at the count value preset in 16-bit compare register 90 (CR90) based on the intervals of the value set in TCL901 and TCL900.

To operate the 16-bit timer as a timer interrupt, the following settings are required.

- Set count values in CR90
- Set 16-bit timer mode control register 90 (TMC90) as shown in Figure 6-5.

Figure 6-5. Settings of 16-Bit Timer Mode Control Register 90 for Timer Interrupt Operation



Caution If both the CPT901 and CPT900 flags are set to 0, the capture operation is prohibited.

When the count value of 16-bit timer counter 90 (TM90) matches the value set in CR90, counting of TM90 continues and an interrupt request signal (INTTM90) is generated.

Table 6-2 shows interval time, and Figure 6-6 shows timing of timer interrupt operation.

Caution When rewriting the value in CR90 during a count operation, be sure to execute the following processing.

<1> Set interrupt disabled (set TMMK90 (bit 4 of interrupt mask flag register 1 (MK1)) to 1).
<2> Disable inversion control of timer output data (set TOC90 to 0)

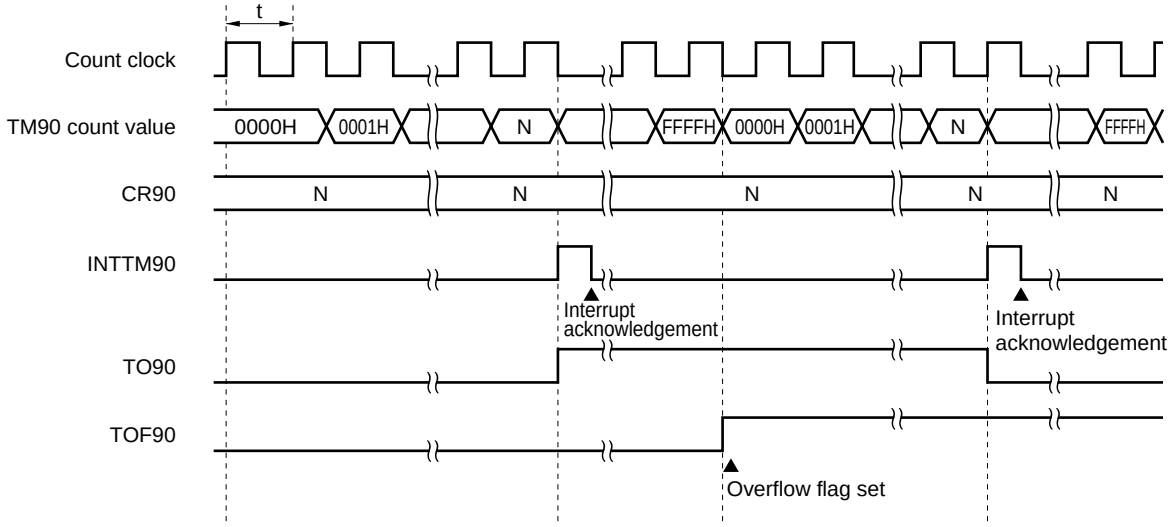
If the value in CR90 is rewritten in the interrupt-enabled state, an interrupt request may occur at the moment of rewrite.

Table 6-2. Interval Time of 16-Bit Timer

TCL901	TCL900	Count Clock	Interval Time
0	0	$2^2/f_X$ (0.8 μs)	$2^{18}/f_X$ (52.4 ms)
0	1	$2^6/f_X$ (12.8 μs)	$2^{22}/f_X$ (838.9 ms)
1	0	$2^7/f_X$ (25.6 μs)	$2^{23}/f_X$ (1.68 s)
1	1	$1/f_{XT}$ (30.5 μs)	$2^{16}/f_{XT}$ (2.0 s)

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.

Figure 6-6. Timing of Timer Interrupt Operation



Remark N = 0000H to FFFFH

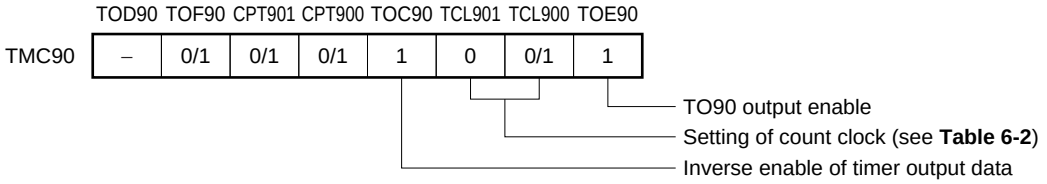
6.4.2 Operation as timer output

Timer outputs are repeatedly generated at the count value preset in 16-bit compare register 90 (CR90) based on the intervals of the value set in TCL901 and TCL900.

To operate the 16-bit timer as a timer output, the following settings are required.

- Set P26 to output mode (PM26 = 0).
- Reset output latch of P26 to 0.
- Set the count value in CR90.
- Set 16-bit timer mode control register 90 (TMC90) as shown in Figure 6-7.

Figure 6-7. Settings of 16-Bit Timer Mode Control Register 90 for Timer Output Operation

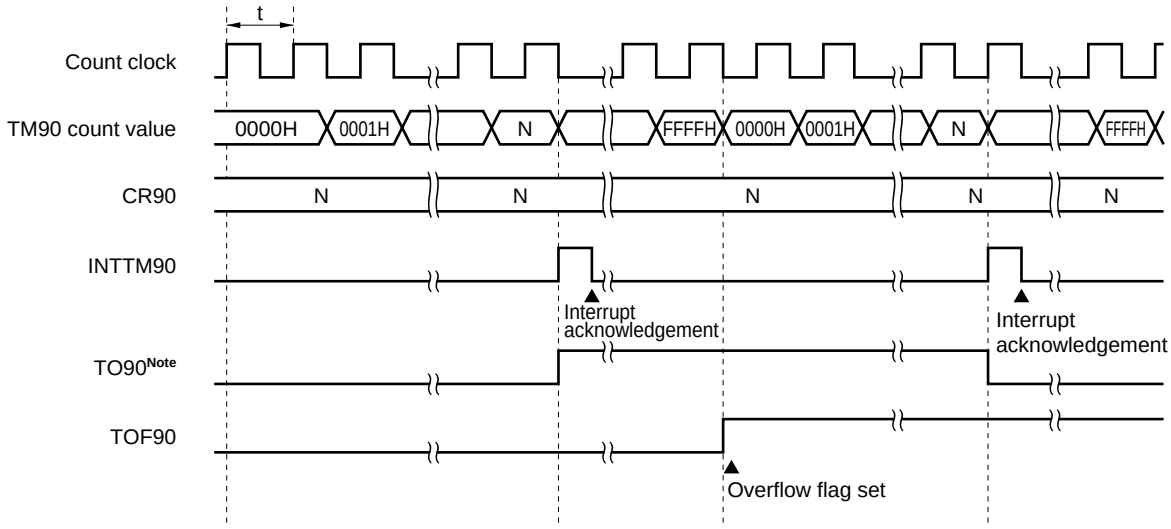


Caution If both the CPT901 flag and CPT900 flag are set to 0, the capture operation is prohibited.

When the count value of 16-bit timer counter 90 (TM90) matches the value set in CR90, the output status of the TO90/P26 pin is inverted. This enables timer output. At that time, TM90 counting continues and an interrupt request signal (INTTM90) is generated.

Figure 6-8 shows the timing of timer output (see Table 6-2 for the interval time of the 16-bit timer).

Figure 6-8. Timer Output Timing



Note The initial value of TO90 becomes low level when output is enabled (TOE90 = 1).

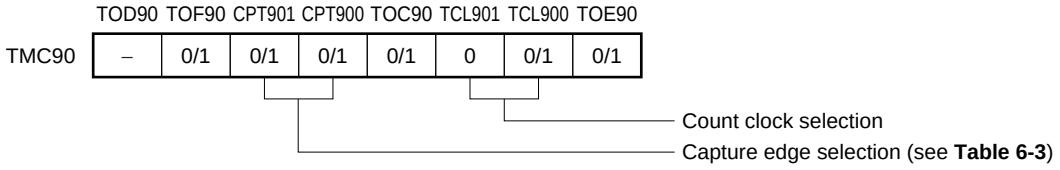
Remark N = 0000H to FFFFH

6.4.3 Capture operation

The capture operation consists of latching the count value of 16-bit timer register 90 (TM90) into a capture register in synchronization with a capture trigger, and retaining the count value.

Set TMC90 as shown in Figure 6-9 to allow the 16-bit timer to start the capture operation.

Figure 6-9. Settings of 16-Bit Timer Mode Control Register 90 for Capture Operation



16-bit capture register 90 (TCP90) starts a capture operation after a CPT90 capture trigger edge is detected, and latches and retains the count value of 16-bit timer register 90. The TCP90 fetches the count value within 2 clocks and retains the count value until the next capture edge detection.

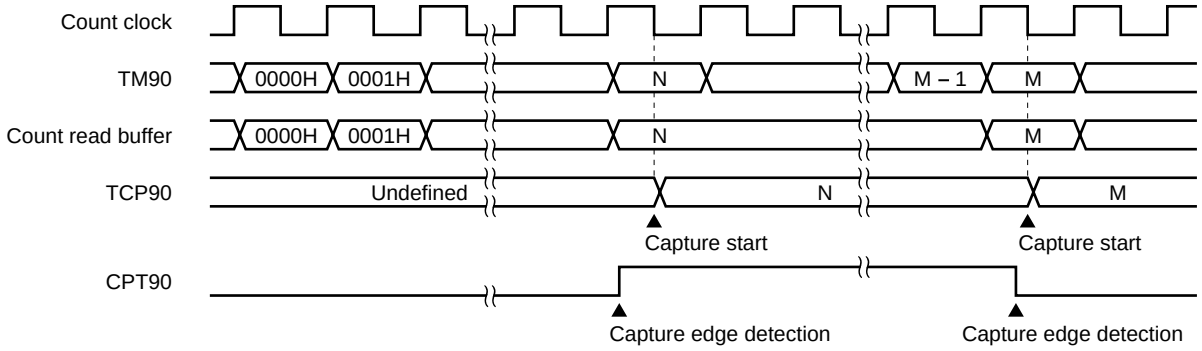
Table 6-3 and Figure 6-10 show the settings of the capture edge and the capture operation timing, respectively.

Table 6-3. Settings of Capture Edge

CPT901	CPT900	Capture Edge Selection
0	0	Capture operation prohibited
0	1	CPT90 pin rising edge
1	0	CPT90 pin falling edge
1	1	CPT90 pin both edges

Caution Because TCP90 is rewritten when a capture trigger edge is detected during TCP90 read, disable the capture trigger edge detection during TCP90 read.

Figure 6-10. Capture Operation Timing (Both Edges of CPT90 Pin Are Specified)



6.4.4 16-bit timer counter 90 readout

The count value of 16-bit timer counter 90 (TM90) is read out using a 16-bit manipulation instruction.

TM90 readout is performed through a counter read buffer. The counter read buffer latches the TM90 count value, and the buffer operation is held pending at the CPU clock falling edge after the read signal of the TM90 lower byte rises, and the count value is retained. The retained counter read buffer value can be read out as the count value.

Cancellation of the pending state is performed at the CPU clock falling edge after the read signal of the TM90 higher byte falls.

$\overline{\text{RESET}}$ input sets TM90 to 0000H and TM90 starts freerunning.

Figure 6-11 shows the timing of 16-bit timer counter 90 readout.

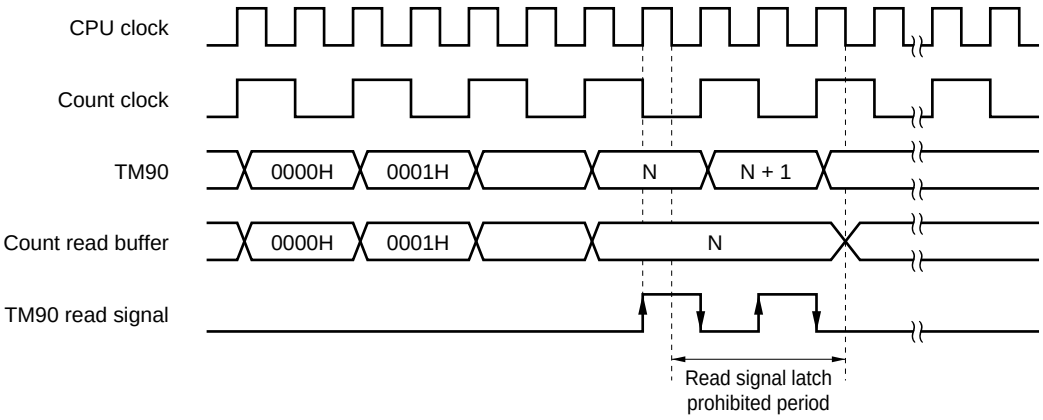
Cautions 1. The count value after releasing stop becomes undefined because the count operation is executed during the oscillation stabilization time.

2. Though TM90 is designed for a 16-bit transfer instruction, an 8-bit transfer instruction can also be used.

When using an 8-bit transfer instruction, execute it by direct addressing.

3. When using an 8-bit transfer instruction, execute in the order from lower byte to higher byte in pairs. If only the lower byte is read, the pending state of the counter read buffer is not canceled, and if only the higher byte is read, an undefined count value is read.

Figure 6-11. 16-Bit Timer Counter 90 Readout Timing



6.4.5 Buzzer output operation

The buzzer frequency is set using buzzer output control register 90 (BZC90) based on the count clock selected with TCL901 and TCL900 of TMC90 (source clock). A square wave of the set buzzer frequency is output.

Table 6-4 shows the buzzer frequency.

Set the 16-bit timer as follows to use it for buzzer output:

- Set P21 to output mode (PM21 = 0).
- Reset output latch of P21 to 0.
- Set a count clock using TCL901 and TCL900.
- Set BZC90 as shown in Figure 6-12.

Figure 6-12. Settings of Buzzer Output Control Register 90 for Buzzer Output Operation

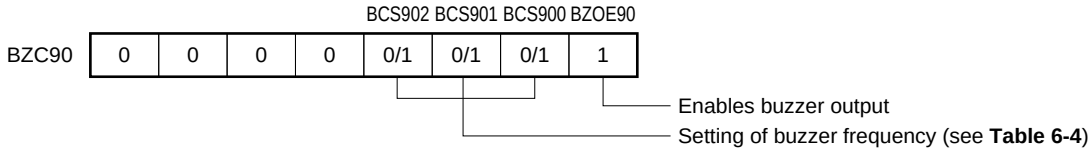


Table 6-4. Buzzer Frequency of 16-Bit Timer

BCS902	BCS901	BCS900	Buzzer Frequency			
			$fcl = f_X/2^2$	$fcl = f_X/2^6$	$fcl = f_X/2^7$	$fcl = f_{XT}$
0	0	0	$fcl/2^4$ (78.1 kHz)	$fcl/2^4$ (4.88 kHz)	$fcl/2^4$ (2.44 kHz)	$fcl/2^4$ (2.05 kHz)
0	0	1	$fcl/2^5$ (39.1 kHz)	$fcl/2^5$ (2.44 kHz)	$fcl/2^5$ (1.22 kHz)	$fcl/2^5$ (1.02 kHz)
0	1	0	$fcl/2^8$ (4.88 kHz)	$fcl/2^8$ (305 Hz)	$fcl/2^8$ (153 Hz)	$fcl/2^8$ (128 Hz)
0	1	1	$fcl/2^9$ (2.44 kHz)	$fcl/2^9$ (153 Hz)	$fcl/2^9$ (76 Hz)	$fcl/2^9$ (64 Hz)
1	0	0	$fcl/2^{10}$ (1.22 kHz)	$fcl/2^{10}$ (76 Hz)	$fcl/2^{10}$ (38 Hz)	$fcl/2^{10}$ (32 Hz)
1	0	1	$fcl/2^{11}$ (610 Hz)	$fcl/2^{11}$ (38 Hz)	$fcl/2^{11}$ (19 Hz)	$fcl/2^{11}$ (16 Hz)
1	1	0	$fcl/2^{12}$ (305 Hz)	$fcl/2^{12}$ (19 Hz)	$fcl/2^{12}$ (10 Hz)	$fcl/2^{12}$ (8 Hz)
1	1	1	$fcl/2^{13}$ (153 Hz)	$fcl/2^{13}$ (10 Hz)	$fcl/2^{13}$ (5 Hz)	$fcl/2^{13}$ (4 Hz)

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.

6.5 Notes on Using 16-Bit Timer

Usable functions differ according to the settings of the count clock selection, CPU clock operation, system clock oscillation status, and BZOE90 (bit 0 of buzzer output control register 90 (BZC90)).

Refer to the following table.

Count Clock	CPU Clock	System Clock		BZOE90	Capture	TM90 Read	Buzzer Output	Timer Output	Timer Interrupt
		Main System Clock	Subsystem Clock						
$f_X/2^2$, $f_X/2^6$, $f_X/2^7$	Main	Oscillating	Oscillating/Stopped	1/0	√	√ Note 1	**Note 2**	√	√
		Stopped			×	×	×	×	×
	Sub	Oscillating	Oscillating		√	×	**Note 2**	√	√
		Stopped			×	×	×	×	×
f_{XT}	Main	Oscillating	Oscillating	0	√	√	×	√	√
				1	×	×	√	√	√
			Stopped	1/0	×	×	×	×	×
		Stopped (STOP mode)	Oscillating	0	×	×	×	×	×
				1	×	×	√	√	√
			Stopped	1/0	×	×	×	×	×
	Sub	Oscillating	Oscillating	0	√	√	×	√	√
				1	×	×	√	√	√
		Stopped		0	×	×	×	×	×
				1	×	×	√	√	√

Notes 1. TM90 is enabled only when the CPU clock is in high-speed mode.

2. Output is enabled when BZOE90 = 1.

Cautions 1. The capture function uses $f_X/2$ for control (refer to Figure 6-1 Block Diagram of 16-Bit Timer). Therefore, the capture function cannot be used when the main system clock is stopped.

2. The read function of TM90 uses the CPU clock for control (refer to Figure 6-1), and reads an undefined value when the CPU clock is slower than the count clock (values are not guaranteed). When reading TM90, set the count clock to the same speed as the CPU clock (when the CPU clock is the main system clock, high-speed mode is set), or select a clock slower than the CPU clock.

3. When the subsystem clock is selected as the count clock and BZOE90 is set to 0, the subsystem clock selected as the TM90 count clock is one that has been synchronized with the main system clock (refer to Figure 6-1). Therefore, when the main system clock oscillation is stopped, the timer operation is stopped because the clock supplied to the 16-bit timer is stopped (timer interrupt is not generated).

Moreover, when the subsystem clock is selected as the count clock and BZOE90 is set to 1, the capture and TM90 read values are not guaranteed because the subsystem clock is not synchronized. Therefore, be sure to set BZOE90 to 0 when using the capture and TM90 read functions (when the subsystem clock is selected as the count clock, buzzer output, capture, and TM90 read functions cannot be used at the same time).

Make the following settings when stopping the main system clock oscillation to support low current consumption and releasing the HALT mode.

Count clock:	Subsystem clock
CPU clock:	Subsystem clock
Main system clock:	Oscillation stopped
BZOE90:	1 (Buzzer output enabled)

At this time, when the setting of P21, the buzzer output alternate function pin, is “PM21 = 0, P21 = 0”, a square wave of the buzzer frequency is output from P21. To avoid outputting the buzzer frequency, make either of the following settings.

- Set P21 to input mode (PM21 = 1)
- If P21 cannot be set to input mode, set the port latch value of P21 to 1 (P21 = 1) (In this case, a high level is output from P21)

CHAPTER 7 8-BIT TIMER

7.1 8-Bit Timer Functions

An 8-bit timer (one channel, timer 50) and an 8-bit timer/event counter (one channel, timer 60) are incorporated in the μPD789426, 789436, 799446, 789456 Subseries. The operation modes listed in the following table can be set via mode register settings.

Table 7-1. Operation Modes

Mode \ Channel	Timer 50	Timer 60
8-bit timer counter mode (Discrete mode)	Available	Available
16-bit timer counter mode (Cascade connection mode)	Available	
Carrier generator mode	Available	
PWM output mode	Available (Free-running mode)	Available (Pulse generator mode)

(1) 8-bit timer counter mode (discrete mode)

The following functions can be used in this mode.

- Interval timer with 8-bit resolution
- External event counter with 8-bit resolution (timer 40 only)
- Square wave output with 8-bit resolution

(2) 16-bit timer counter mode (cascade connection mode)

Operation as a 16-bit timer/event counter is enabled during cascade connection mode.

The following functions can be used in this mode.

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

(3) Carrier generator mode

The carrier clock generated by timer 60 is output in cycles set by timer 50.

(4) PWM output mode

(a) Timer 50: Free-running mode

The timer output status inverts repeatedly due to a match between TM50 and CR50 and TM50 overflow, and pulses of any duty ratio are output.

(b) Timer 60: Pulse generator mode

The timer output status inverts repeatedly due to the settings of TM60, CR60, and CRH60, and pulses of any duty ratio are output (either P32/INTP2/TO60 or P33/INTP3/TO61 can be selected as the timer output pin using software).

7.2 8-Bit Timer Configuration

The 8-bit timer includes the following hardware.

Table 7-2. 8-Bit Timer Configuration

Item	Configuration
Timer counters	8 bits × 2 (TM50, TM60)
Registers	Compare registers: 8 bits × 3 (CR50, CR60, CRH60)
Timer outputs	3 (TO50, TO60, TO61)
Control registers	8-bit timer mode control register 50 (TMC50) 8-bit timer mode control register 60 (TMC60) Carrier generator output control register 60 (TCA60) Port mode register 3 (PM3)

Figure 7-1. Block Diagram of Timer 50

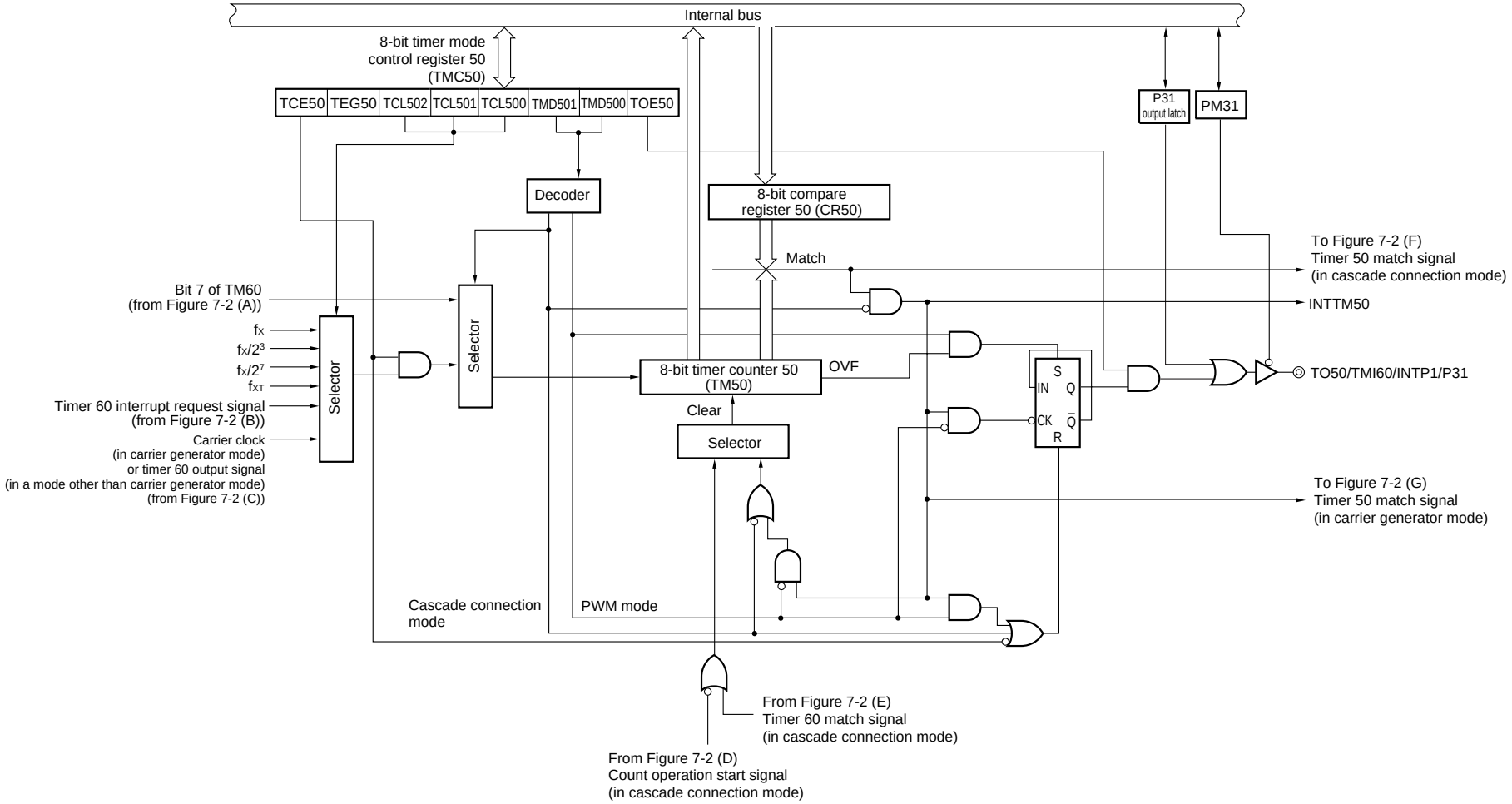
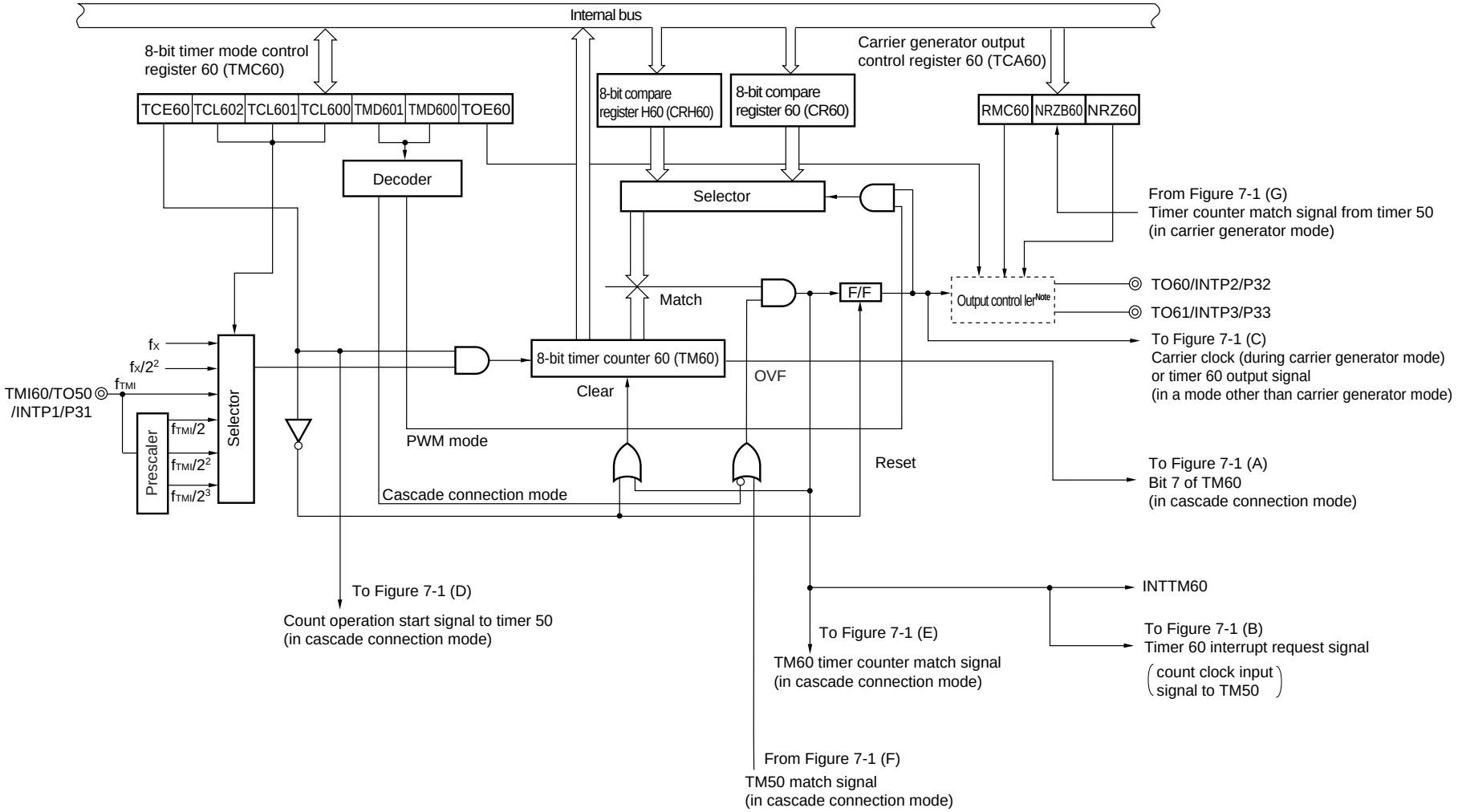
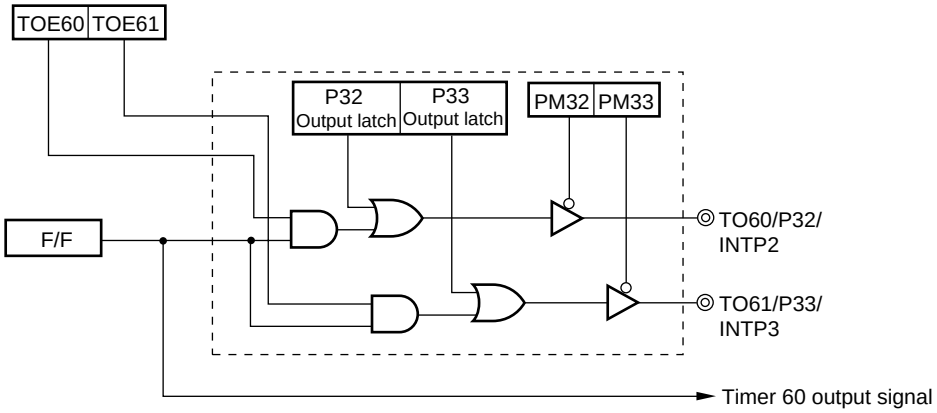


Figure 7-2. Block Diagram of Timer 60



Note For details, see **Figure 7-3**.

Figure 7-3. Block Diagram of Output Controller (Timer 60)



(1) 8-bit compare register 50 (CR50)

This 8-bit register is used to continually compare the value set to CR50 with the count value in 8-bit timer counter 50 (TM50) and to generate an interrupt request (INTTM50) when a match occurs.

CR50 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes CR50 undefined.

Cautions 1. If the CR50 is overwritten during timer operation in the PWM output mode (TMD501 = 1, TMD500 = 0), a high level may be output for 1 cycle immediately after. If this waveform poses a problem for the application, either <1> stop the timer when overwriting the CR50, or <2> overwrite the CR50 with the TOE50 in a cleared status.

2. If the valid edge of the count clock is selected for both edges in the PWM output mode (TEG50 = 1), do not set 00H, 01H, and FFH to the CR50. If the rising edge is selected (TEG50 = 0), do not set 00H to CR50.

(2) 8-bit compare register 60 (CR60)

This 8-bit register is used to continually compare the value set to CR60 with the count value in 8-bit timer counter 60 (TM60) and to generate an interrupt request (INTTM60) when a match occurs. When connected to TM50 via a cascade connection and used as a 16-bit timer/event counter, the interrupt request (INTTM60) occurs only when matches occur simultaneously between CR50 and TM50 and between CR60 and TM60 (INTTM50 does not occur).

CR60 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes CR60 undefined.

(3) 8-bit compare register H60 (CRH60)

In PWM output mode, the high-level width of timer output is set by writing a value to CRH60.

CRH60 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes CRH60 undefined.

(4) 8-bit timer counters 50 and 60 (TM50 and TM60)

These are 8-bit registers that are used to count the count pulse.

TM50 and TM60 are read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TM50 and TM60 to 00H.

TM50 and TM60 are cleared to 00H under the following conditions.

(a) Discrete mode

(i) TM50

- After reset
- When TCE50 (bit 7 of 8-bit timer mode control register 50 (TMC50)) is cleared to 0
- When a match occurs between TM50 and CR50
- When the TM50 count value overflows

(ii) TM60

- After reset
- When TCE60 (bit 7 of 8-bit timer mode control register 60 (TMC60)) is cleared to 0
- When a match occurs between TM60 and CR60
- When the TM60 count value overflows

(b) Cascade connection mode (TM50 and TM60 are simultaneously cleared to 00H)

- After reset
- When the TCE60 flag is cleared to 0
- When matches occur simultaneously between TM50 and CR50 and between TM60 and CR60
- When the TM50 and TM60 count values overflow simultaneously

(c) Carrier generator mode

(i) TM50

- After reset
- When the TCE50 flag is cleared to 0
- When a match occurs between TM50 and CR50

(ii) TM60

- After reset
- When the TCE60 flag is cleared to 0
- When a match occurs between TM60 and CR60
- When a match occurs between TM60 and CRH60

(d) PWM output mode

(i) TM50

- After reset
- When the TCE50 flag is cleared to 0
- When a match occurs between TM50 and CR50
- When the TM50 count value overflows

(ii) TM60

- Reset
- When the TCE60 flag is cleared to 0
- When a match occurs between TM60 and CRH60
- When the TM60 count value overflows

7.3 Registers Controlling 8-Bit Timer

The 8-bit timer is controlled by the following four registers.

- 8-bit timer mode control register 50 (TMC50)
- 8-bit timer mode control register 60 (TMC60)
- Carrier generator output control register 60 (TCA60)
- Port mode register 3 (PM3)

(1) 8-bit timer mode control register 50 (TMC50)

8-bit timer mode control register 50 (TMC50) is used to control the timer 50 count clock setting and the operation mode setting.

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

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

Figure 7-4. Format of 8-Bit Timer Mode Control Register 50

Symbol	<7>	<6>	5	4	3	2	1	<0>	Address	After reset	R/W
TMC50	TCE50	TEG50	TCL502	TCL501	TCL500	TMD501	TMD500	TOE50	FF4DH	00H	R/W

TCE50	Control of TM50 count operation[Note 1]
0	Clears TM50 count value and stops operation
1	Starts count operation

TEG50	Valid edge selection for TM50 count clock
0	Counts at the rising edge of the count clock
1	Counts at both edges of the count clock[Note 2]

TCL502	TCL501	TCL500	Selection of timer 50 count clock
0	0	0	f_X (5.0 MHz)
0	0	1	$f_X/2^3$ (625 kHz)
0	1	0	$f_X/2^7$ (39.1 kHz)
0	1	1	f_{XT} (32.768 kHz)
1	0	0	Timer 60 match signal
1	0	1	Carrier clock (in carrier generator mode) or timer 60 output signal (in a mode other than carrier generator mode)
Other than above			Setting prohibited

TMD501	TMD500	TMD601	TMD600	Selection of operation mode for timer 50 and timer 60[Note 2]
0	0	0	0	Discrete mode (8-bit timer counter mode)
0	1	0	1	Cascade connection mode (16-bit timer counter mode)
0	0	1	1	Carrier generator mode
1	0	1	0	Timer 50: PWM free-running mode Timer 60: PWM pulse generator mode
Other than above				Setting prohibited

TOE50	Control of timer output
0	Output disabled
1	Output enabled

Notes 1. Since the count operation is controlled by TCE40 (bit 7 of TMC40) in cascade connection mode, any setting for TCE30 is ignored.

2. The selection of both edges is valid only in the PWM output mode. In 8-bit counter mode or cascade connection mode, counting is done using the rising edge even if TEG50 is set to “1”.

3. The operation mode selection is set to both the TMC30 register and TMC40 register.

Cautions 1. In cascade connection mode, the output signal of timer 60 is forcibly selected as the count clock.

2. When operating TMC50, be sure to perform settings in the following order.

<1> Stop TM50 count operation.

<2> Set the operation mode and the count clock.

<3> Start count operation.

Remarks 1. f_X: Main system clock oscillation frequency (ceramic/crystal oscillation)

2. f_{CC}: Main system clock oscillation frequency (RC oscillation)

(2) 8-bit timer mode control register 60 (TMC60)

8-bit timer mode control register 60 (TMC60) is used to control the timer 60 count clock setting and the operation mode setting.

TMC60 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC60 to 00H.

Figure 7-5. Format of 8-Bit Timer Mode Control Register 60

Symbol	<7>	<6>	5	4	3	2	1	<0>	Address	After reset	R/W
TMC60	TCE60	TOE61	TCL602	TCL601	TCL600	TMD601	TMD600	TOE60	FF4EH	00H	R/W

TCE60	Control of TM60 count operation[Note 1]
0	Clears TM60 count value and stops operation (the count value is also cleared for TM50 in cascade connection mode)
1	Starts count operation (the count operation is also started for TM50 in cascade connection mode)

TCL602	TCL601	TCL600	Selection of timer 60 count clock
0	0	0	f_X (5.0 MHz)
0	0	1	$f_X/2^2$ (1.25 MHz)
0	1	0	f_{TMI} (external input clock)
0	1	1	$f_{TMI}/2$ (external input clock)
1	0	0	$f_{TMI}/2^2$ (external input clock)
1	0	1	$f_{TMI}/2^3$ (external input clock)
Other than above			Setting prohibited

TMD501	TMD500	TMD601	TMD600	Selection of operation mode for timer 50 and timer 60[Note 2]
0	0	0	0	Discrete mode (8-bit timer counter mode)
0	1	0	1	Cascade connection mode (16-bit timer counter mode)
0	0	1	1	Carrier generator mode
1	0	1	0	Timer 50: PWM free-running mode Timer 60: PWM pulse generator mode
Other than above				Setting prohibited

TOE61	TOE60	Control of timer output
0	0	Output disabled
0	1	Output enabled only for TO60
1	0	Output enabled only for TO61
1	1	Setting prohibited

Notes 1. Since the count operation is controlled by TCE60 (bit 7 of TMC60) in cascade connection mode, any setting for TCE50 is ignored.

2. The operation mode selection is set to both the TMC50 register and TMC60 register.

Caution When operating the TMC60, be sure to perform settings in the following order.

<1> Stop the TM60 count operation.

<2> Set the operation mode and the count clock.

<3> Start count operation.

Remarks 1. f_X: Main system clock oscillation frequency

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

(3) Carrier generator output control register 60 (TCA60)

This register is used to set the timer output data in carrier generator mode.

TCA60 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TCA60 to 00H.

Figure 7-6. Format of Carrier Generator Output Control Register 60

Symbol	7	6	5	4	3	<2>	<1>	<0>	Address	After reset	R/W
TCA60	0	0	0	0	0	RMC60	NRZB60	NRZ60	FF4FH	00H	W

RMC60	Control of remote control output
0	When NRZB60 = 1, a carrier pulse is output. When NRZB60 = 0, a low level is output.
1	When NRZB60 = 1, high-level signal is output. When NRZB60 = 0, a low level is output.

NRZB60	This is the bit that stores the next data to be output to NRZ60. When a match signal occurs (for a match with timer 50), the data is output to NRZ60. Input the required value to NRZ60 by program beforehand.

NRZ60	No return zero data
0	Outputs low-level signal (carrier clock is stopped)
1	Outputs carrier pulse

Caution TCA60 cannot be set with a 1-bit memory manipulation instruction. Be sure to use an 8-bit memory manipulation instruction to set TCA60.

(4) Port mode register 3 (PM3)

This register is used to set the I/O mode of port 3 in 1-bit units.

When using the P31/TO50/INTP1/TMI60 pin as a timer output, set the PM31 and P31 output latch to 0.

When using the P32/TO60/INTP2 pin as a timer output, set the PM32 and P32 output latch to 0.

When using the P33/TO61/INTP3 pin as a timer output, set the PM33 and P33 output latch to 0.

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

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

Figure 7-7. Format of Port Mode Register 3

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

PM3n	I/O mode of P3n pin (n = 0 to 3)
0	Output mode (output buffer is ON)
1	Input mode (output buffer is OFF)

7.4 8-Bit Timer Operation

7.4.1 Operation as 8-bit timer counter

Timer 50 and timer 60 can be independently used as 8-bit timer counters.
The following modes can be used for the 8-bit timer counter.

- Interval timer with 8-bit resolution
- External event counter with 8-bit resolution (timer 60 only)
- Square wave output with 8-bit resolution

(1) Operation as interval timer with 8-bit resolution

The interval timer with 8-bit resolution repeatedly generates an interrupt at a time interval specified by the count value preset in 8-bit compare register n0 (CRn0).
To operate 8-bit timer n0 as an interval timer, settings must be made in the following sequence.

<1> Disable operation of 8-bit timer counter n0 (TMn0) (TCEn0 = 0).
<2> Disable timer output of TOn0 (TOEn0 = 0).
<3> Set a count value in CRn0.
<4> Set the operation mode of timer n0 to 8-bit timer counter mode (see Figures 7-4 and 7-5).
<5> Set the count clock for timer n0 (see Tables 7-3 to 7-6).
<6> Enable the operation of TMn0 (TCEn0 = 1).

When the count value of 8-bit timer counter n0 (TMn0) matches the value set in CRn0, TMn0 is cleared to 00H and continues counting. At the same time, an interrupt request signal (INTTMn0) is generated.
Tables 7-3 to 7-6 show interval time, and Figures 7-8 to 7-13 show the timing of the interval timer operation.

Caution Be sure to stop the timer operation before overwriting the count clock with different data.

Remark n = 5, 6

Table 7-3. Interval Time of Timer 50

TCL502	TCL501	TCL500	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	$1/f_X$ (0.2 μs)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (0.2 μs)
0	0	1	$2^3/f_X$ (1.6 μs)	$2^{11}/f_X$ (409.6 μs)	$2^3/f_X$ (1.6 μs)
0	1	0	$2^7/f_X$ (25.6 μs)	$2^{15}/f_X$ (6.55 ms)	$2^7/f_X$ (25.6 μs)
0	1	1	$1/f_{XT}$ (30.5 μs)	$2^8/f_{XT}$ (7.81 ms)	$1/f_{XT}$ (30.5 μs)
1	0	0	Input cycle of timer 60 match signal	Input cycle of timer 60 match signal $\times$ 8	Input cycle of timer 60 match signal
1	0	1	Input cycle of timer 60 output	Input cycle of timer 60 output $\times$ 8	Input cycle of timer 60

Remarks 1. f_X: Main system clock oscillation frequency

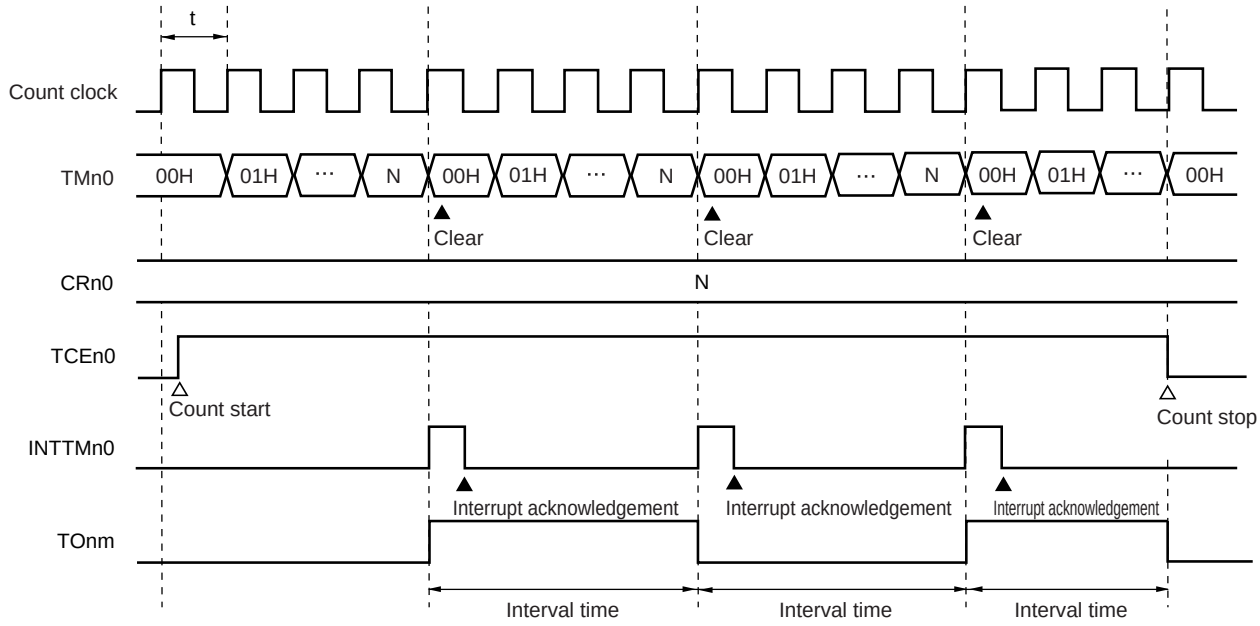
2. f_{XT}: Subsystem clock oscillation frequency

Table 7-4. Interval Time of Timer 60

TCL602	TCL601	TCL600	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	$1/f_X$ (0.2 μs)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (0.2 μs)
0	0	1	$2/f_X$ (0.4 μs)	$2^9/f_X$ (1.02 μs)	$2/f_X$ (0.4 μs)
0	1	0	f_{TMI} input cycle	f_{TMI} input cycle $\times 2^8$	f_{TMI} input cycle
0	1	1	$f_{TMI}/2$ input cycle	$f_{TMI}/2$ input cycle $\times 2^8$	$f_{TMI}/2$ input cycle
1	0	0	$f_{TMI}/2^2$ input cycle	$f_{TMI}/2^2$ input cycle $\times 2^8$	$f_{TMI}/2^2$ input cycle
1	0	1	$f_{TMI}/2^3$ input cycle	$f_{TMI}/2^3$ input cycle $\times 2^8$	$f_{TMI}/2^3$ input cycle

Remark f_X: Main system clock oscillation frequency

Figure 7-8. Timing of Interval Timer Operation with 8-Bit Resolution (Basic Operation)

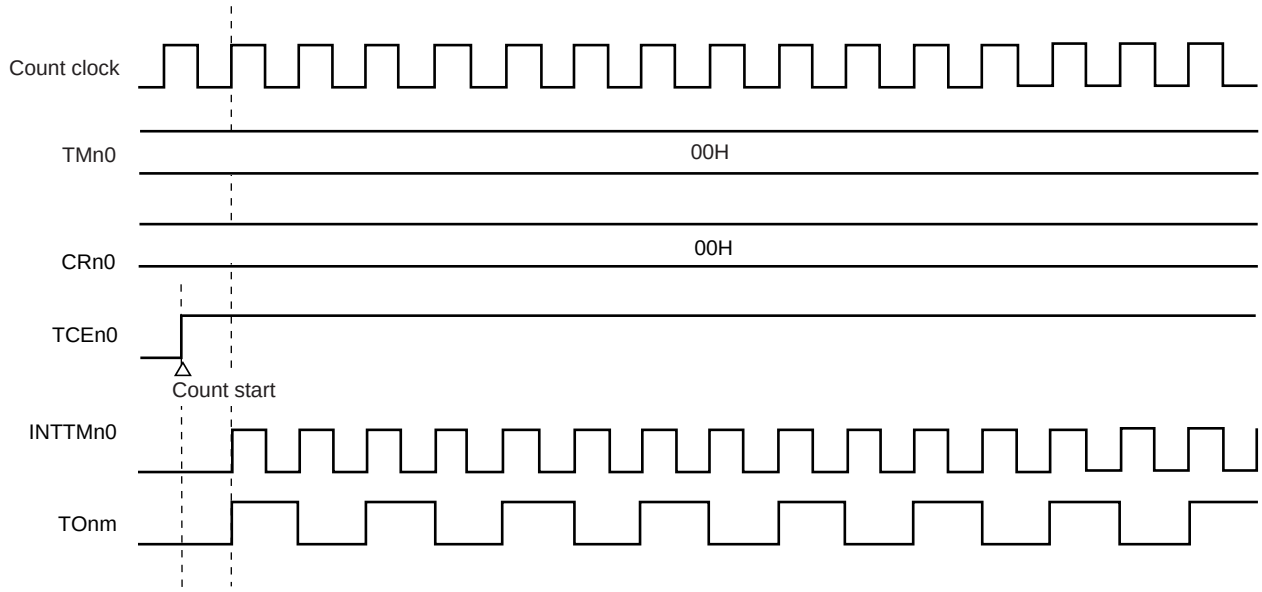


Remarks 1. Interval time = (N + 1) × t: N = 00H to FFH

2. n = 5, 6

nm = 50, 60, 61

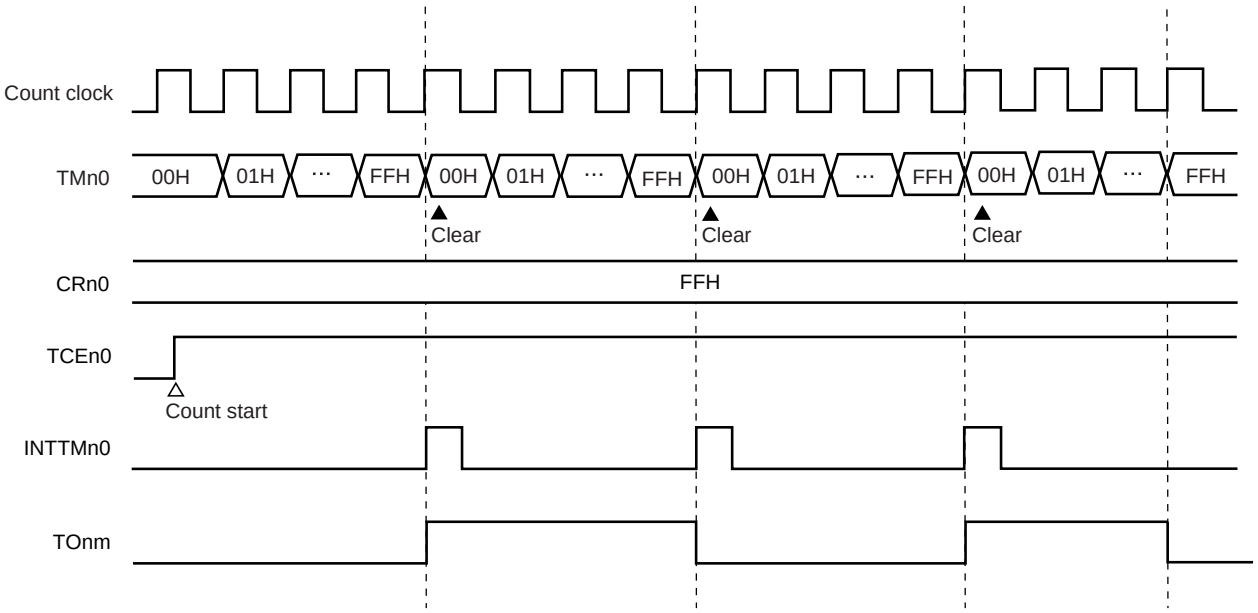
Figure 7-9. Timing of Interval Timer Operation with 8-Bit Resolution (When CRn0 Is Set to 00H)



Remark n = 5, 6

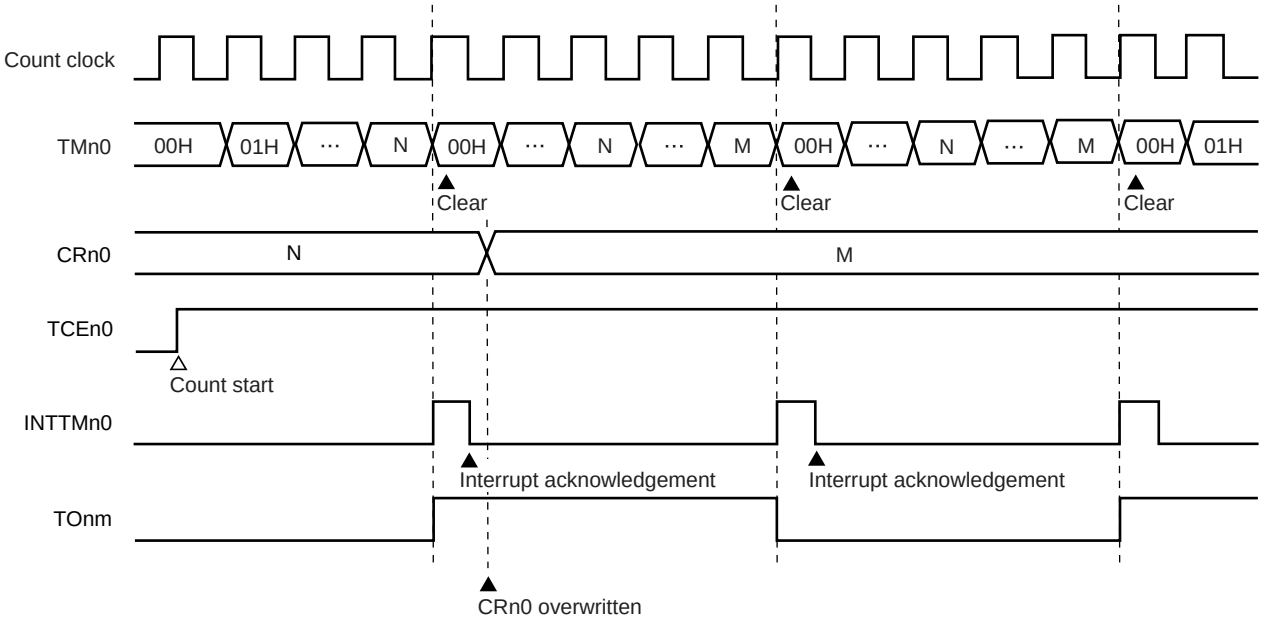
nm = 50, 60, 61

Figure 7-10. Timing of Interval Timer Operation with 8-Bit Resolution (When CRn0 Is Set to FFH)



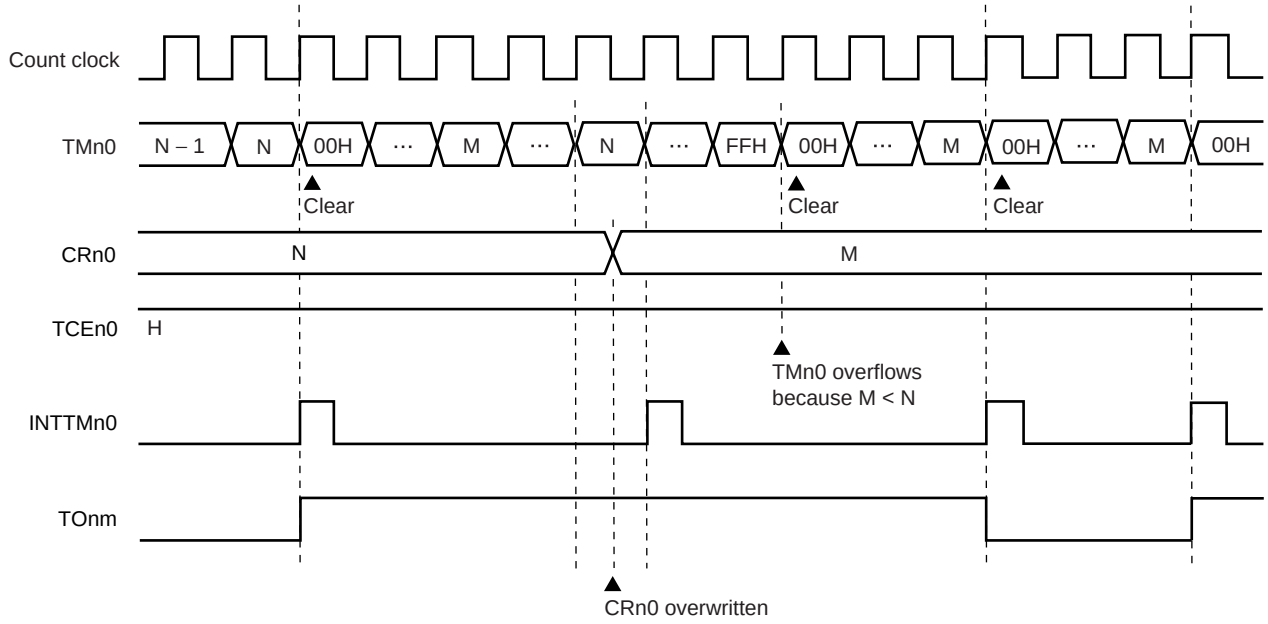
Remark n = 5, 6
nm = 50, 60, 61

Figure 7-11. Timing of Interval Timer Operation with 8-Bit Resolution (When CRn0 Changes from N to M (N < M))



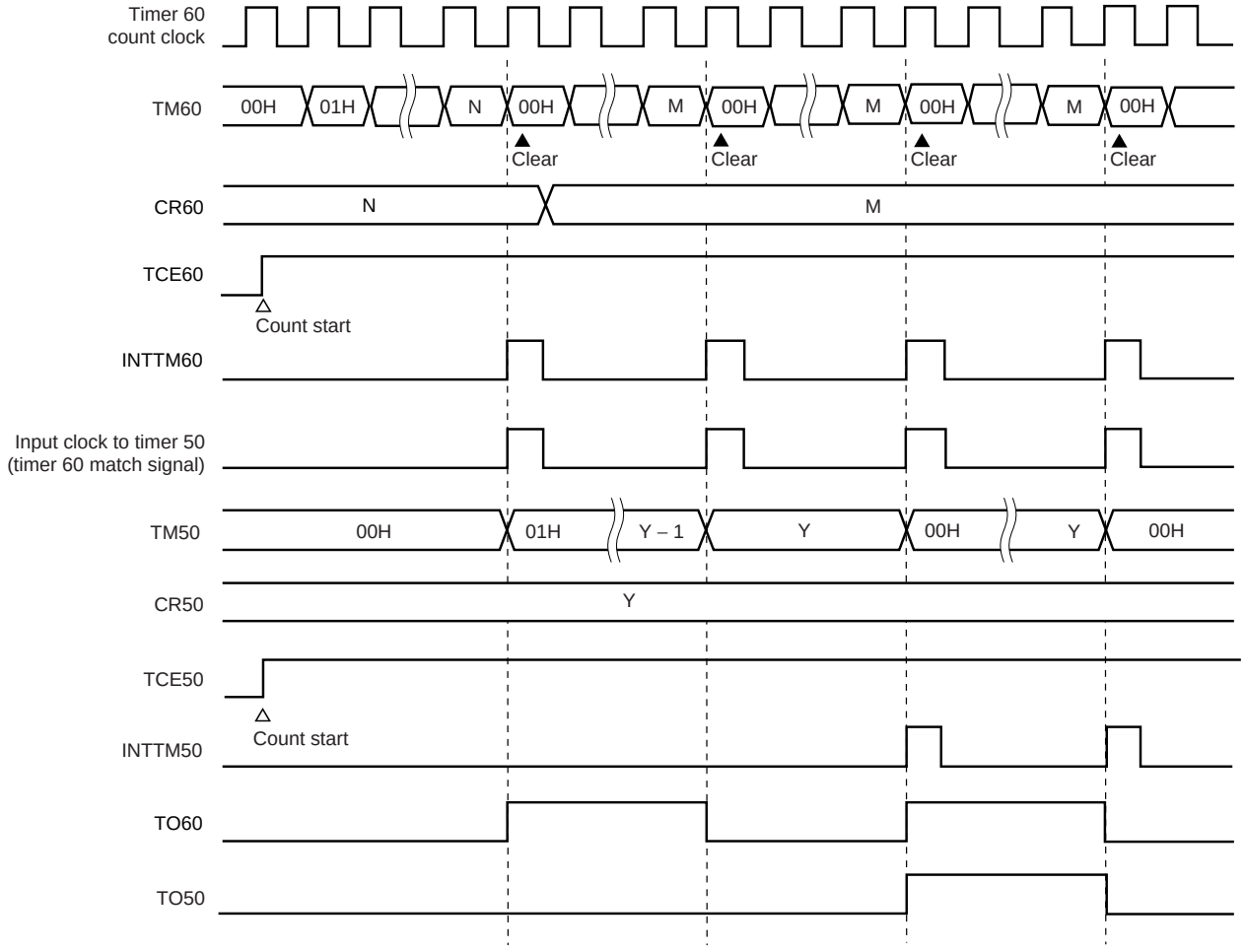
Remark n = 5, 6
nm = 50, 60, 61

Figure 7-12. Timing of Interval Timer Operation with 8-Bit Resolution (When CRn0 Changes from N to M (N > M))



Remark n = 5, 6
nm = 50, 60, 61

Figure 7-13. Timing of Interval Timer Operation with 8-Bit Resolution (When Timer 60 Match Signal Is Selected for Timer 50 Count Clock)



(2) Operation as external event counter with 8-bit resolution (timer 60 only)

The external event counter counts the number of external clock pulses input to the TMI60/P31/INTP1/TO50 pin by using 8-bit timer counter 60 (TM60).

To operate timer 60 as an external event counter, settings must be made in the following sequence.

<1> Disable operation of 8-bit timer counter 60 (TM60) (TCE60 = 0).
<2> Disable timer output of TO60 (TOE60 = 0).
<3> Set P31 to input mode (PM31 = 1).
<4> Select the external input clock for timer 60 (see Table 7-5).
<5> Set the operation mode of timer 60 to 8-bit timer counter mode (see Figures 7-4 and 7-5).
<6> Set a count value in CR60.
<7> Enable the operation of TM60 (TCE60 = 1).

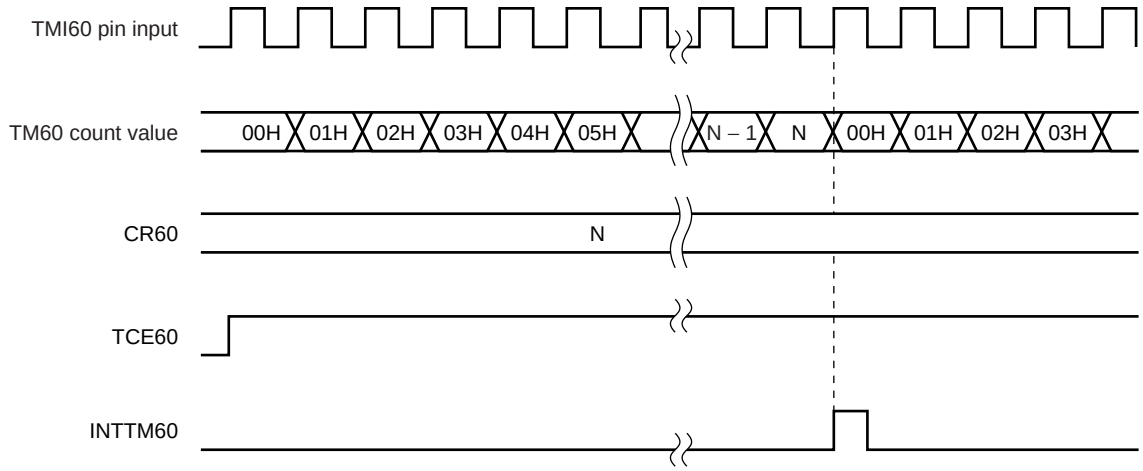
Each time the valid edge is input, the value of TM60 is incremented.

When the count value of TM60 matches the value set in CR60, TM60 is cleared to 00H and continues counting. At the same time, an interrupt request signal (INTTM60) is generated.

Figure 7-14 shows the timing of the external event counter operation.

Caution Be sure to stop the timer operation before overwriting the count clock with different data.

Figure 7-14. Timing of Operation of External Event Counter with 8-Bit Resolution



Remark N = 00H to FFH

(3) Operation as square-wave output with 8-bit resolution

Square waves of any frequency can be output at an interval specified by the value preset in 8-bit compare register n0 (CRn0).

To operate timer n0 for square-wave output, settings must be made in the following sequence.

<1> When using timer 50, set P31 to output mode (PM31 = 0).
When using timer 60, set P32 to output mode (PM32 = 0) or set P33 to output mode (PM33 = 0) (When TO61 is selected as timer output).
<2> Set the output latches of P31, P32, and P33 to 0.
<3> Disable operation of timer counter n0 (TMn0) (TCEn0 = 0).
<4> Set a count clock for timer n0 and enable output of TOn0 (TOEn0 = 1)[Note].
<5> Set a count value in CRn0.
<6> Enable the operation of TMn0 (TCEn0 = 1).

When the count value of TMn0 matches the value set in CRn0, the TOn0 pin output will be inverted. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurs, TMn0 is cleared to 00H and continues counting. At the same time, an interrupt request signal (INTTMn0) is generated.

The square-wave output is cleared to 0 by setting TCEn0 to 0.

Tables 7-5 and 7-6 show the square-wave output range, and Figure 7-15 shows the timing of square-wave output.

Note In the case of timer 60, either TO60 or TO61 can be selected as the timer output pin. If TO61 is selected, set TOE61 = 1.

Caution Be sure to stop the timer operation before overwriting the count clock with different data.

Remark n = 5, 6

Table 7-5. Square-Wave Output Range of Timer 50 (During f_X = 5.0 MHz Operation)

TCL502	TCL501	TCL500	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	0	$1/f_X$ (0.2 μs)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (0.2 μs)
0	0	1	$2^3/f_X$ (1.6 μs)	$2^{11}/f_X$ (409.6 μs)	$2^3/f_X$ (1.6 μs)
0	1	0	$2^7/f_X$ (25.6 μs)	$2^{15}/f_X$ (6.55 ms)	$2^7/f_X$ (25.6 μs)
0	1	1	$1/f_{XT}$ (30.5 μs)	$2^8/f_{XT}$ (7.81 ms)	$1/f_{XT}$ (30.5 μs)
1	0	0	Input cycle of timer 60 match signal	Input cycle of timer 60 match signal × 8	Input cycle of timer 60 match signal
1	0	1	Input cycle of timer 60 output	Input cycle of timer 60 output × 8	Input cycle of timer 60

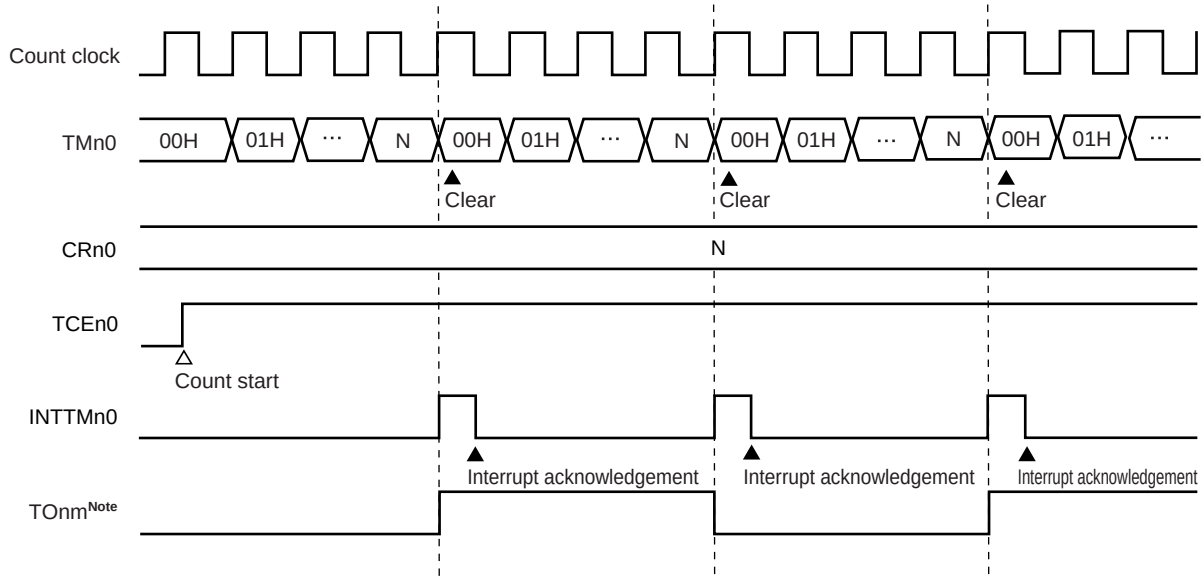
Remarks 1. f_X: Main system clock oscillation frequency
2. f_{XT}: Subsystem clock oscillation frequency

Table 7-6. Square-Wave Output Range of Timer 60 (During f_X = 5.0 MHz Operation)

TCL602	TCL601	TCL600	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	0	$1/f_X$ (0.2 μs)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (0.2 μs)
0	0	1	$2/f_X$ (0.4 μs)	$2^9/f_X$ (1.02 ms)	$2/f_X$ (0.4 μs)
0	1	0	f_{TMI} input cycle	f_{TMI} input cycle $\times 2^8$	f_{TMI} input cycle
0	1	1	$f_{TMI}/2$ input cycle	$f_{TMI}/2$ input cycle $\times 2^8$	$f_{TMI}/2$ input cycle
1	0	0	$f_{TMI}/2^2$ input cycle	$f_{TMI}/2^2$ input cycle $\times 2^8$	$f_{TMI}/2^2$ input cycle
1	0	1	$f_{TMI}/2^3$ input cycle	$f_{TMI}/2^3$ input cycle $\times 2^8$	$f_{TMI}/2^3$ input cycle

Remark f_X: Main system clock oscillation frequency

Figure 7-15. Timing of Square-Wave Output with 8-Bit Resolution



Note The initial value of TOnm is low level when output is enabled (TOEnm = 1).

Remark n = 5, 6
nm = 50, 60, 61

7.4.2 Operation as 16-bit timer counter

Timer 50 and timer 60 can be used as a 16-bit timer counter using cascade connection. In this case, 8-bit timer counter 50 (TM50) is the higher 8 bits and 8-bit timer counter 60 (TM60) is the lower 8 bits. 8-bit timer 60 controls reset and clear.

The following modes can be used for the 16-bit timer counter.

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

(1) Operation as interval timer with 16-bit resolution

The interval timer with 16-bit resolution repeatedly generates an interrupt at a time interval specified by the count value preset in 8-bit compare register 50 (CR50) and 8-bit compare register 60 (CR60).

To operate as an interval timer with 16-bit resolution, settings must be made in the following sequence.

<1> Disable operation of 8-bit timer counter 50 (TM50) and 8-bit timer counter 60 (TM60) (TCE50 = 0, TCE60 = 0).
<2> Disable timer output of TO60 (TOE60 = 0).
<3> Set the count clock for timer 60 (see Tables 7-5 and 7-6).
<4> Set the operation mode of timer 50 and 8-bit timer 60 to 16-bit timer counter mode (see Figures 7-4 and 7-5).
<5> Set a count value in CR50 and CR60.
<6> Enable the operation of TM50 and TM60 (TCE60 = 1[Note]).

Note Start and clear of the timer in the 16-bit timer counter mode are controlled by TCE60 (the value of TCE50 is invalid).

When the count values of TM50 and TM60 match the values set in CR50 and CR60 respectively, both TM50 and TM60 are simultaneously cleared to 00H and counting continues. At the same time, an interrupt request signal (INTTM60) is generated (INTTM50 is not generated).

Table 7-7 shows interval time, and Figure 7-16 shows the timing of the interval timer operation.

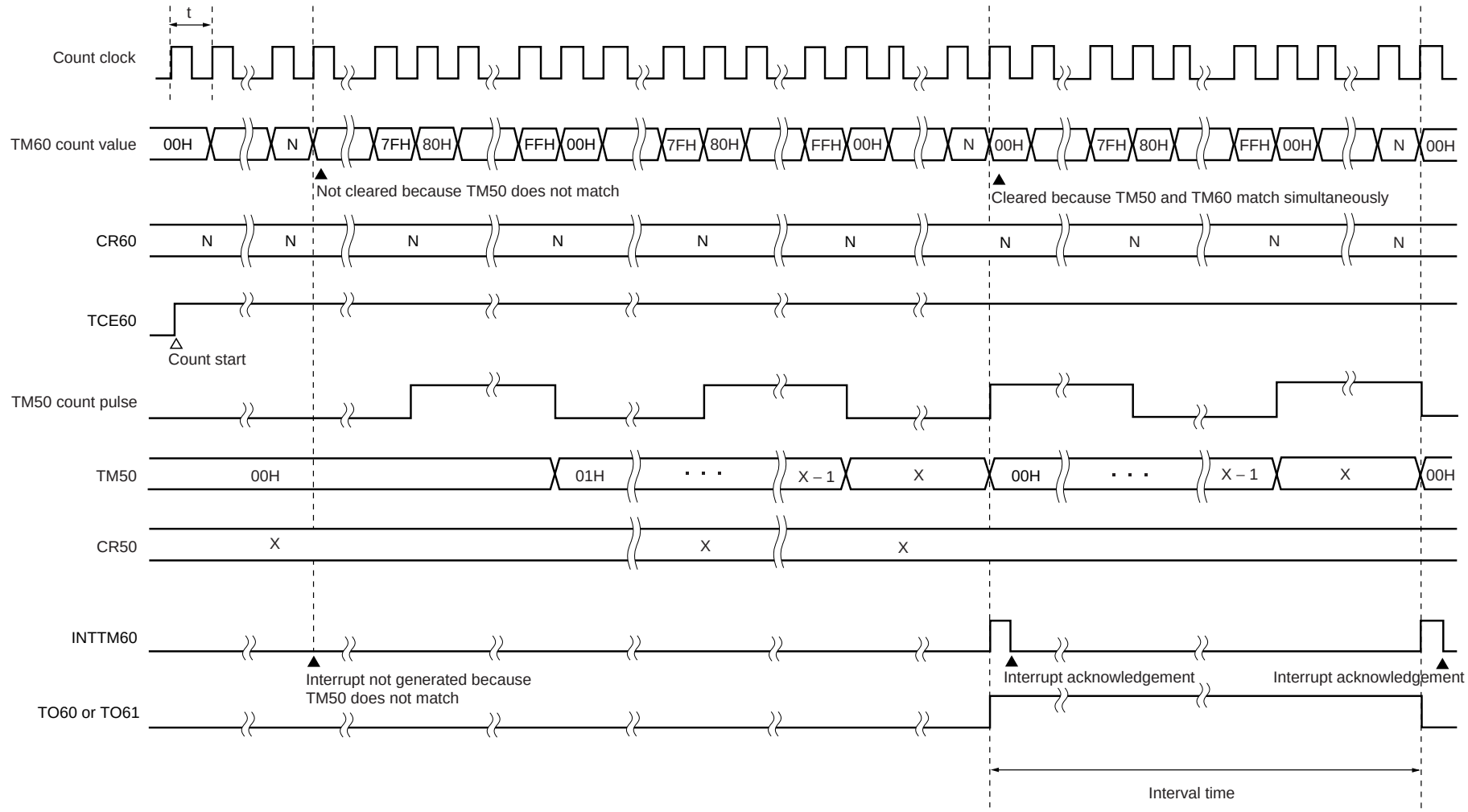
Cautions 1. Be sure to stop the timer operation before overwriting the count clock with different data.
2. In the 16-bit timer counter mode, TO50 cannot be used. Be sure to set TOE50 = 0 to disable TO50 output.

Table 7-7. Interval Time with 16-Bit Resolution (During f_X = 5.0 MHz Operation)

TCL602	TCL601	TCL600	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	$1/f_X$ (0.2 μs)	$2^{16}/f_X$ (13.1 ms)	$1/f_X$ (0.2 μs)
0	0	1	$2/f_X$ (0.4 μs)	$2^{17}/f_X$ (26.2 ms)	$2/f_X$ (0.4 μs)
0	1	0	f_{TMI} input cycle	f_{TMI} input cycle $\times 2^{16}$	f_{TMI} input cycle
0	1	1	$f_{TMI}/2$ input cycle	$f_{TMI}/2$ input cycle $\times 2^{16}$	$f_{TMI}/2$ input cycle
1	0	0	$f_{TMI}/2^2$ input cycle	$f_{TMI}/2^2$ input cycle $\times 2^{16}$	$f_{TMI}/2^2$ input cycle
1	0	1	$f_{TMI}/2^3$ input cycle	$f_{TMI}/2^3$ input cycle $\times 2^{16}$	$f_{TMI}/2^3$ input cycle

Remark f_X: Main system clock oscillation frequency

Figure 7-16. Timing of Interval Timer Operation with 16-Bit Resolution



Remark Interval time = $(256X + N + 1) \times t$: X = 00H to FFH, N = 00H to FFH

(2) Operation as external event counter with 16-bit resolution

The external event counter counts the number of external clock pulses input to the TMI60/P31/INTP1/TO50 pin by TM50 and TM60.

To operate as an external event counter with 16-bit resolution, settings must be made in the following sequence.

<1> Disable operation of TM50 and TM60 (TCE50 = 0, TCE60 = 0).
<2> Disable timer output of TO60 (TOE60 = 0).
<3> Set P31 to input mode (PM31 = 1).
<4> Select the external input clock for timer 60 (see Tables 7-5 and 7-6).
<5> Set the operation mode of timer 50 and 8-bit timer 60 to 16-bit timer counter mode (see Figures 7-4 and 7-5).
<6> Set a count value in CR50 and CR60.
<7> Enable the operation of TM50 and TM60 (TCE60 = 1[Note]).

Note Start and clear of the timer in the 16-bit timer counter mode are controlled by TCE60 (the value of TCE50 is invalid).

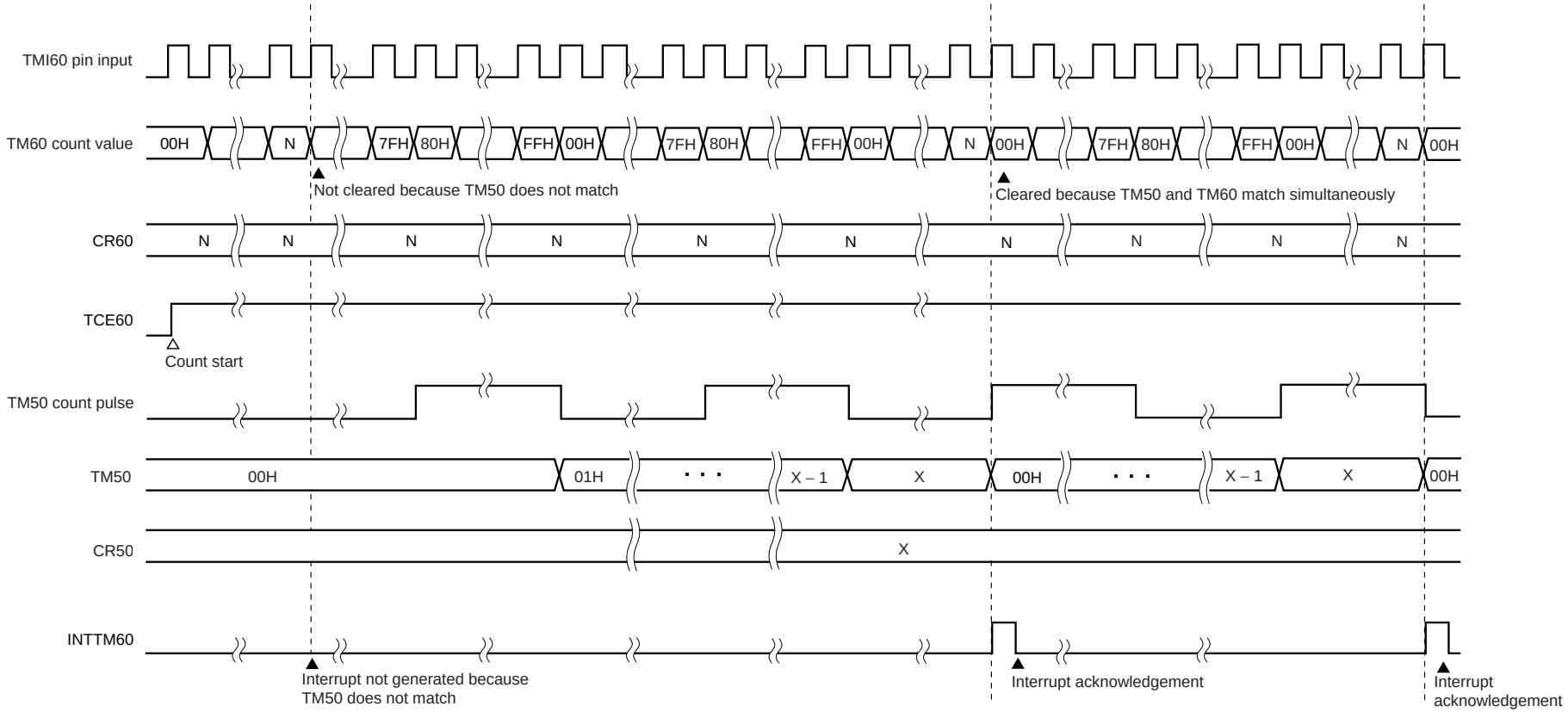
Each time the valid edge is input, the values of TM50 and TM60 are incremented.

When the count values of TM50 and TM60 simultaneously match the values set in CR50 and CR60 respectively, both TM50 and TM60 are cleared to 00H and counting continues. At the same time, an interrupt request signal (INTTM60) is generated (INTTM50 is not generated).

Figure 7-17 shows the timing of the external event counter operation.

Caution Be sure to stop the timer operation before overwriting the count clock with different data.

Figure 7-17. Timing of External Event Counter Operation with 16-Bit Resolution



Remark X = 00H to FFH, N = 00H to FFH

(3) Operation as square-wave output with 16-bit resolution

Square waves of any frequency can be output at an interval specified by the count value preset in CR50 and CR60.

To operate as a square-wave output with 16-bit resolution, settings must be made in the following sequence.

<1> Disable operation of TM50 and TM60 (TCE50 = 0, TCE60 = 0).
<2> Disable output of TO50 and TO60 (TOE50 = 0, TOE60 = 0).
<3> Set a count clock for timer 60.
<4> Select either TO60 or TO61 as the timer output pin.
  If TO60 is selected: Set P32 to the output mode (PM32 = 0), set the P32 output latch to 0, and set TO60 to output enable (TO60 = 1). (Use of TO50 is prohibited.)
  If TO61 is selected: Set P33 to the output mode (PM33 = 0), set the P33 output latch to 0, and set TO61 to output enable (TOE61 = 1). (Use of TO50 is prohibited.)
<5> Set count values in CR50 and CR60.
<6> Enable the operation of TM60 (TCE60 = 1[Note]).

Note Start and clear of the timer in the 16-bit timer counter mode are controlled by TCE60 (the value of TCE50 is invalid).

When the count values of TM50 and TM60 simultaneously match the values set in CR50 and CR60 respectively, the TO60 pin output will be inverted. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurs, TM50 and TM60 are cleared to 00H and counting continues. At the same time, an interrupt request signal (INTTM60) is generated (INTTM50 is not generated).

The square-wave output is cleared to 0 by setting TCE60 to 0.

Table 7-8 shows the square wave output range, and Figure 7-18 shows timing of square wave output.

Cautions 1. Be sure to stop the timer operation before overwriting the count clock with different data.

2. In the 16-bit timer counter mode, TO50 cannot be used. Be sure to set TOE50 = 0 to disable TO50 output.

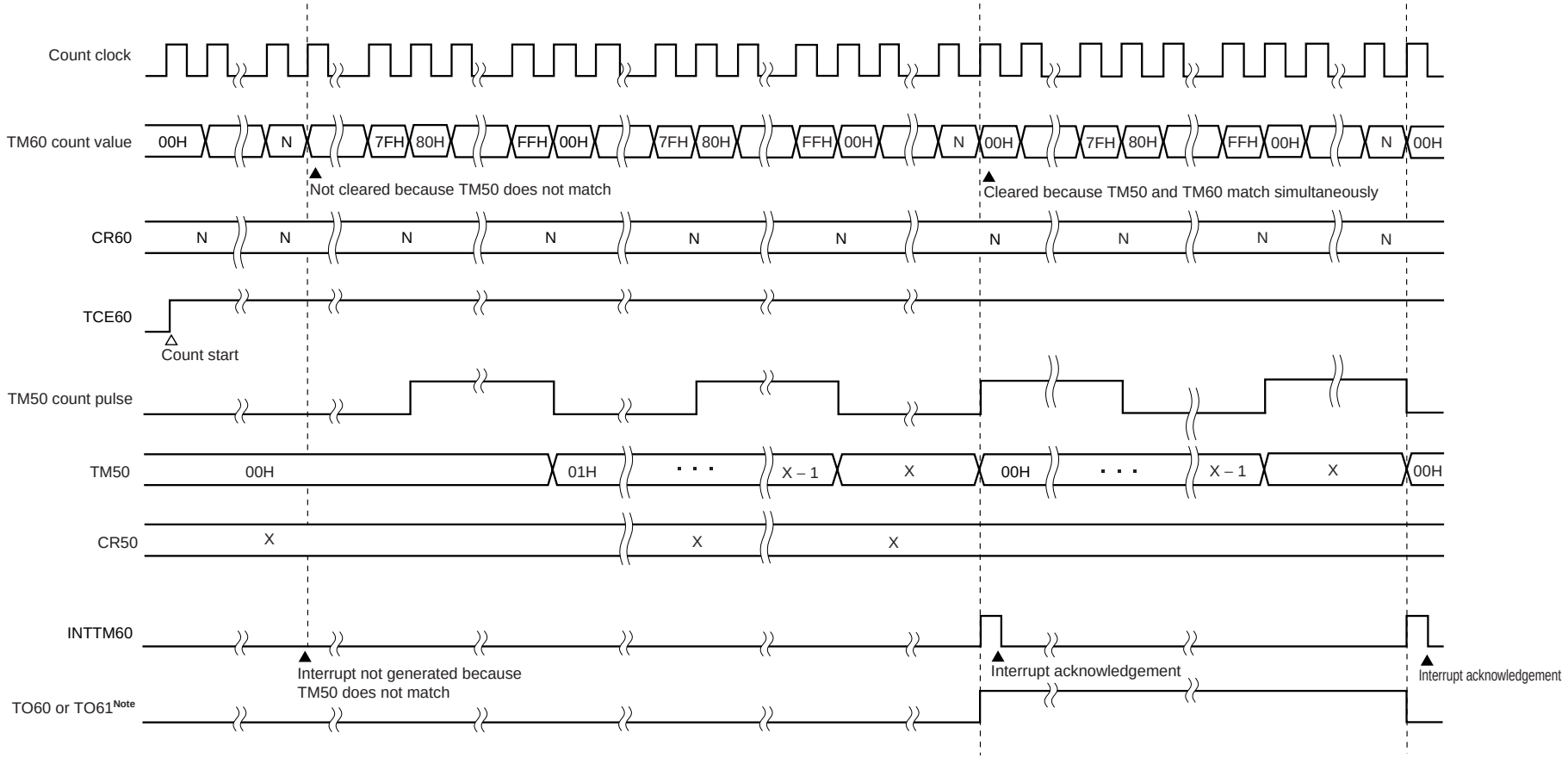
Remark Items in parentheses are for when the TO61 pin is selected for timer output.

Table 7-8. Square-Wave Output Range with 16-Bit Resolution (During f_X = 5.0 MHz Operation)

TCL602	TCL601	TCL600	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	0	$1/f_X$ (0.2 μs)	$2^{16}/f_X$ (13.1 ms)	$1/f_X$ (0.2 μs)
0	0	1	$2/f_X$ (0.4 μs)	$2^{17}/f_X$ (26.2 ms)	$2/f_X$ (0.4 μs)
0	1	0	f_{TMI} input cycle	f_{TMI} input cycle $\times 2^{16}$	f_{TMI} input cycle
0	1	1	$f_{TMI}/2$ input cycle	$f_{TMI}/2$ input cycle $\times 2^{16}$	$f_{TMI}/2$ input cycle
1	0	0	$f_{TMI}/2^2$ input cycle	$f_{TMI}/2^2$ input cycle $\times 2^{16}$	$f_{TMI}/2^2$ input cycle
1	0	1	$f_{TMI}/2^3$ input cycle	$f_{TMI}/2^3$ input cycle $\times 2^{16}$	$f_{TMI}/2^3$ input cycle

Remark f_X: Main system clock oscillation frequency

Figure 7-18. Timing of Square-Wave Output with 16-Bit Resolution



Note The initial value of TO60 or TO61 is low level when output is enabled.

Remark X = 00H to FFH, N = 00H to FFH

7.4.3 Operation as carrier generator

An arbitrary carrier clock generated by TM60 can be output in the cycle set in TM50.

To operate timer 50 and timer 60 as carrier generators, settings must be made in the following sequence.

<1> Disable operation of TM50 and TM60 (TCE50 = 0, TCE60 = 0).

<2> Disable timer output of TO50 and TO60 (TOE50 = 0, TOE60 = 0).

<3> Set count values in CR50, CR60, and CRH60.

<4> Set the operation mode of timer 50 and timer 60 to carrier generator mode (see Figures 7-4 and 7-5).

<5> Set the count clock for timer 50 and timer 60.

<6> Set remote control output to carrier pulse (RMC60 (bit 2 of carrier generator output control register 60 (TCA60)) = 0).
Input the required value to NRZB60 (bit 1 of TCA60) by program.
Input a value to NRZ60 (bit 0 of TCA60) before it is reloaded from NRZB60.

<7> Select either TO60 or TO61 as the timer output pin.
If TO60 is selected: Set P32 to the output mode (PM32 = 0), set the P32 output latch to 0, and set TOE60 to output enable (TOE60 = 1).
If TO61 is selected: Set P33 to the output mode (PM33 = 0), set the P33 output latch to 0, and set TOE61 to output enable (TOE60 = 1).

<8> Enable the operation of TM50 and TM60 (TCE50 = 1, TCE60 = 1).

The operation of the carrier generator is as follows.

<1> When the count value of TM60 matches the value set in CR60, an interrupt request signal (INTTM60) is generated and output of timer 60 is inverted, which makes the compare register switch from CR60 to CRH60.

<2> After that, when the count value of TM60 matches the value set in CRH60, an interrupt request signal (INTTM60) is generated and output of timer 60 is inverted again, which makes the compare register switch from CRH60 to CR60.

<3> The carrier clock is generated by repeating <1> and <2> above.

<4> When the count value of TM50 matches the value set in CR50, an interrupt request signal (INTTM50) is generated. The rising edge of INTTM50 is the data reload signal of NRZB60 and is transferred to NRZ60.

<5> When NRZ60 is 1, a carrier clock is output from the TO60 pin (or the TO61 pin).

Cautions 1. TCA60 cannot be set with a 1-bit memory manipulation instruction. Be sure to use an 8-bit memory manipulation instruction.

2. When setting the carrier generator operation again after stopping it once, reset NRZB60 because the previous value is not retained. In this case also a 1-bit memory manipulation instruction cannot be used. Be sure to use an 8-bit memory manipulation instruction.

Figures 7-19 to 7-21 show the operation timing of the carrier generator.

Figure 7-19. Timing of Carrier Generator Operation (When CR60 = N, CRH60 = M (M > N))

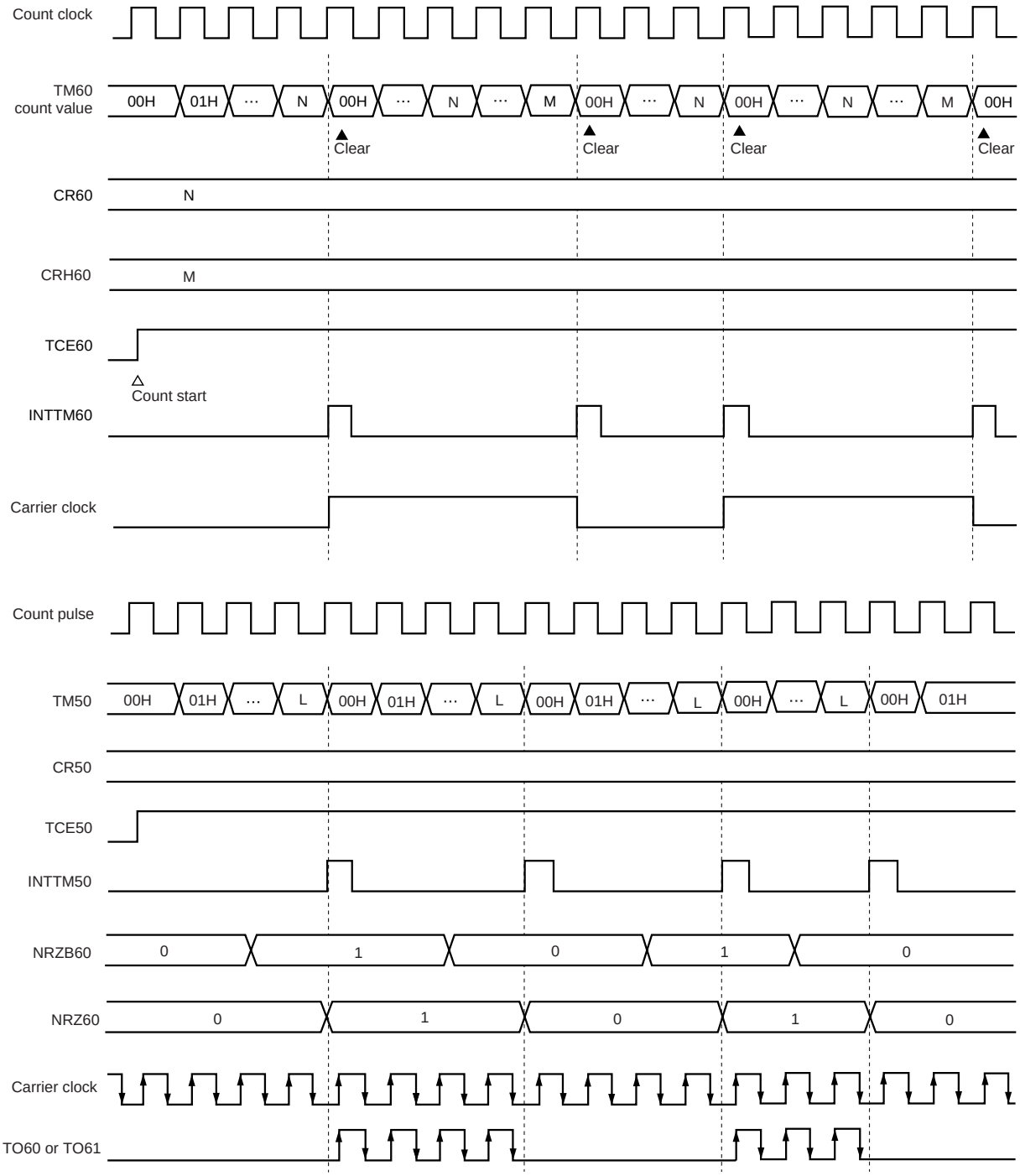


Figure 7-20. Timing of Carrier Generator Operation
(When CR60 = N, CRH60 = M (M < N), Phases of Carrier Clock and NRZ60 Are Asynchronous)

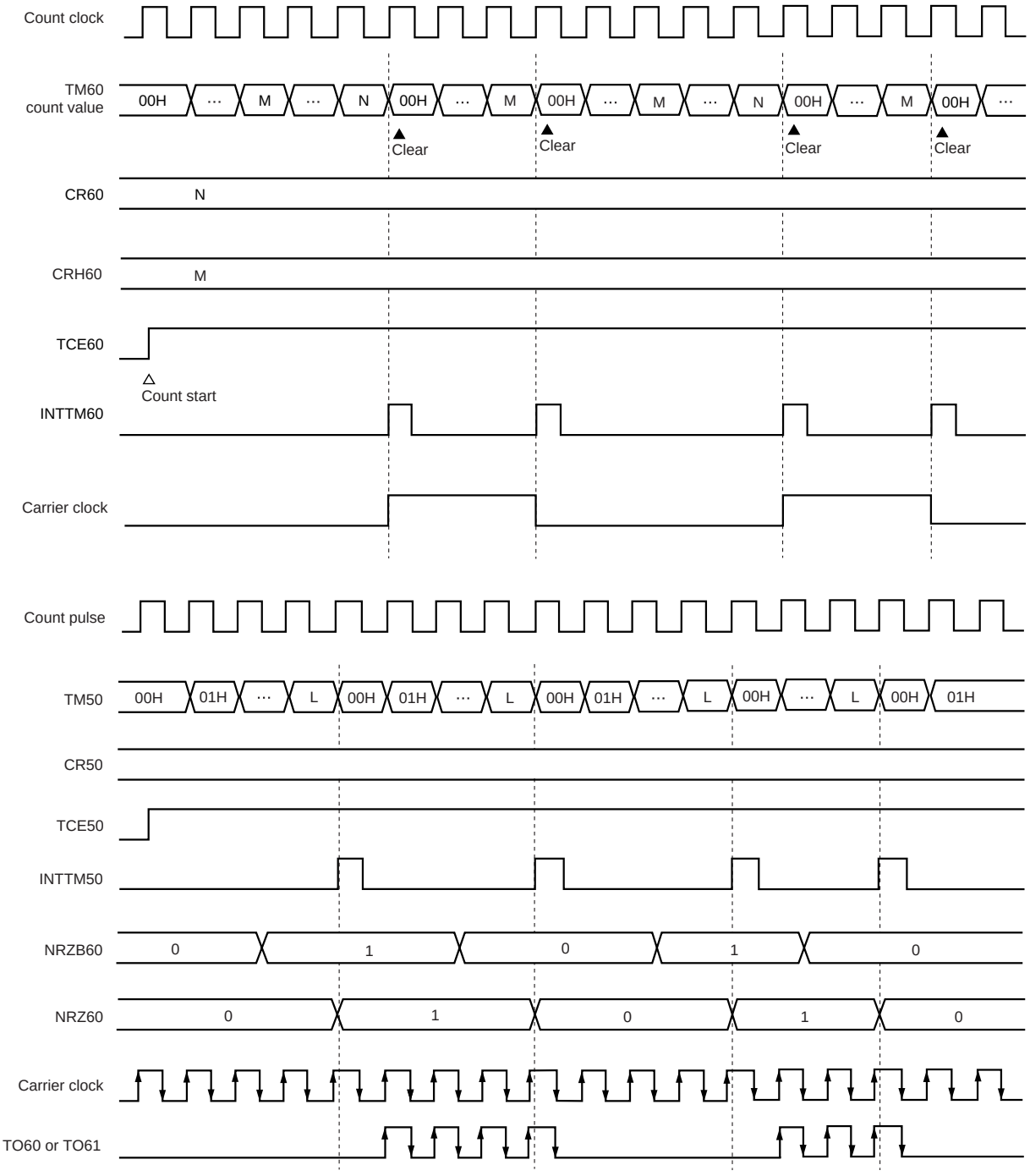
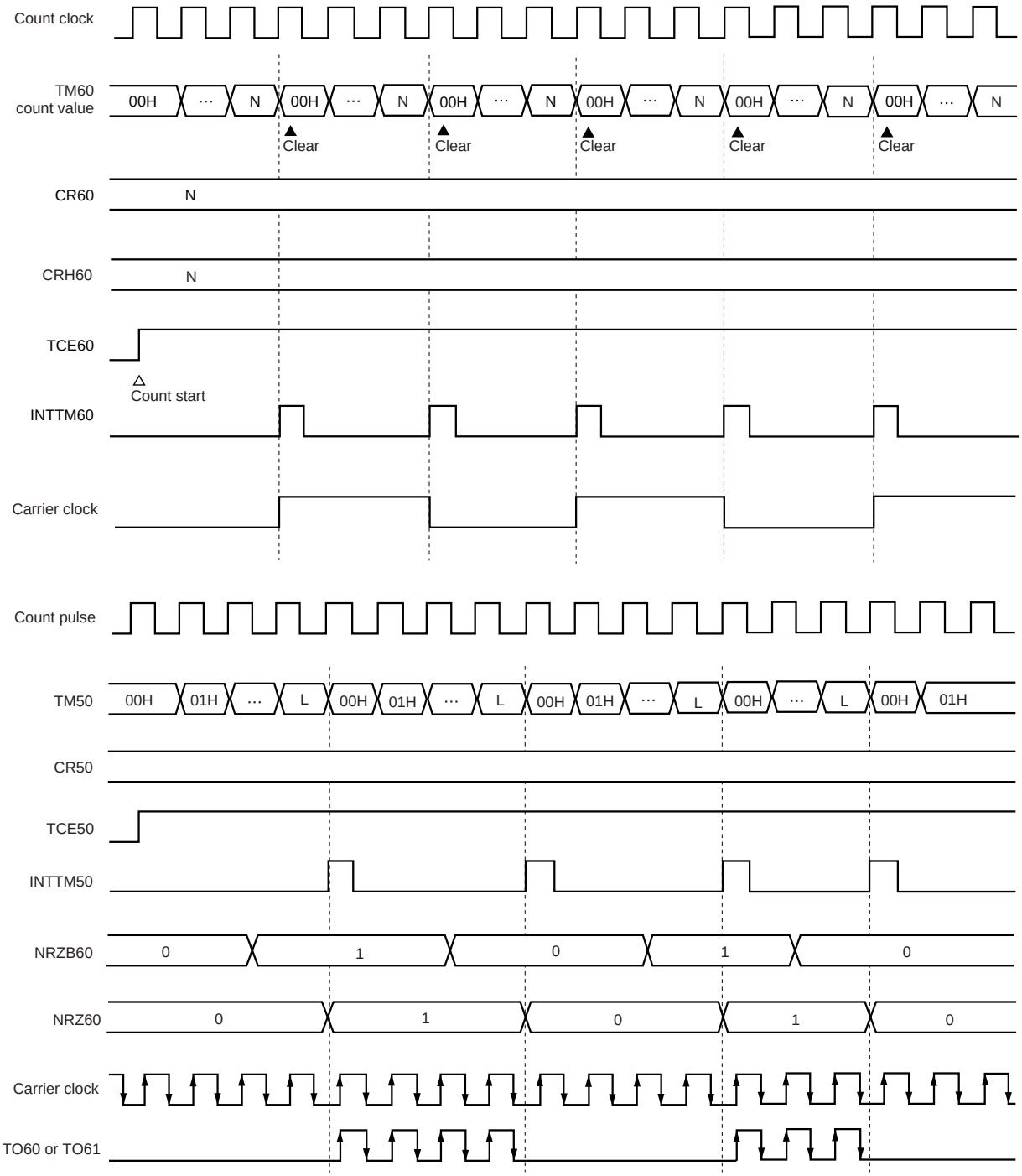


Figure 7-21. Timing of Carrier Generator Operation (When CR60 = CRH60 = N)



7.4.4. PWM free-running mode operation (timer 50)

In the PWM free-running mode, TO50 becomes high level when TM50 overflows, and TO50 becomes low level when CR50 and TM50 match. It is thus possible to output a pulse with any duty ratio.

To operate timer 50 in the PWM free-running mode, setting must be made in the following sequence.

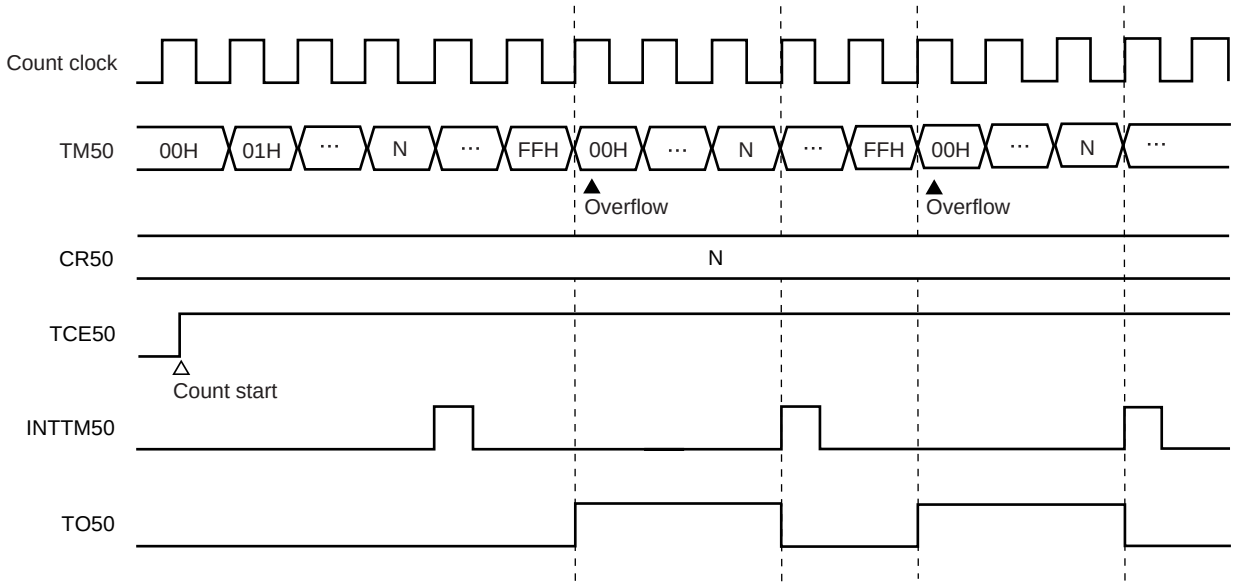
<1> Disable operation of TM50 (TCE50 = 0).
<2> Disable timer output of TO50 (TOE50 = 0).
<3> Set a count value to CR50.
<4> Set the operation mode of timer 50 to the PWM free-running mode. (see Figure 7-4.)
<5> Set the count clock for timer 50.
<6> Set P31 to the output mode (PM31 = 0) and the P31 output latch to 0 and enable timer output of TO50 (TOE50 = 1).
<7> Enable the operation of TM50 (TCE50 = 1).

The operation in the PWM free-running mode is as follows.

<1> When the count value of TM50 matches the value set in CR50, an interrupt request signal (INTTM50) is generated and a low level is output by the TO50. The TM50 continues counting without being cleared.
<2> TO50 outputs a high level when the TM50 overflows.

A pulse of any duty is output by repeating the above procedure. Figures 7-22 to 7-25 show the operation timing in the PWM free-running mode.

Figure 7-22. Operation Timing in PWM Free-Running Mode (When Rising Edge Is Selected)



Caution When the rising edge is selected, do not set the CR50 to 00H. If the CR50 is set to 00H, PWM output may not be performed normally.

Figure 7-23. Operation Timing When Overwriting CR50 (When Rising Edge Is Selected) (1/2)

(1) When setting CR50 > TM50 after overflow

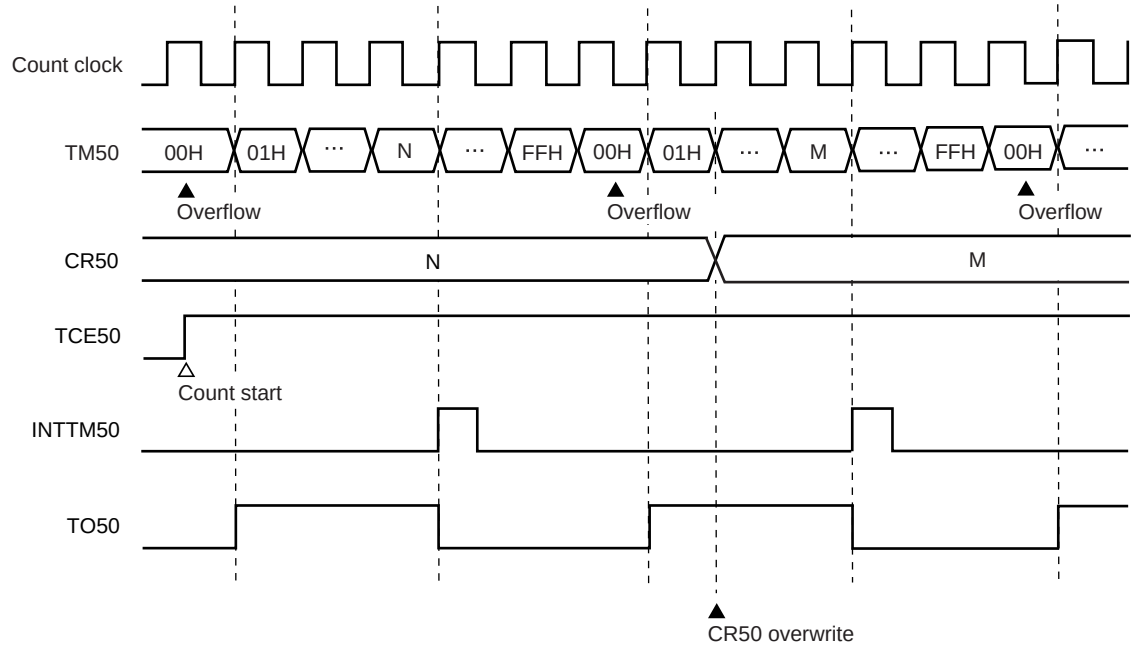


Figure 7-23. Operation Timing When Overwriting CR50 (When Rising Edge Is Selected) (2/2)

(2) When setting CR50 < TM50 after overflow

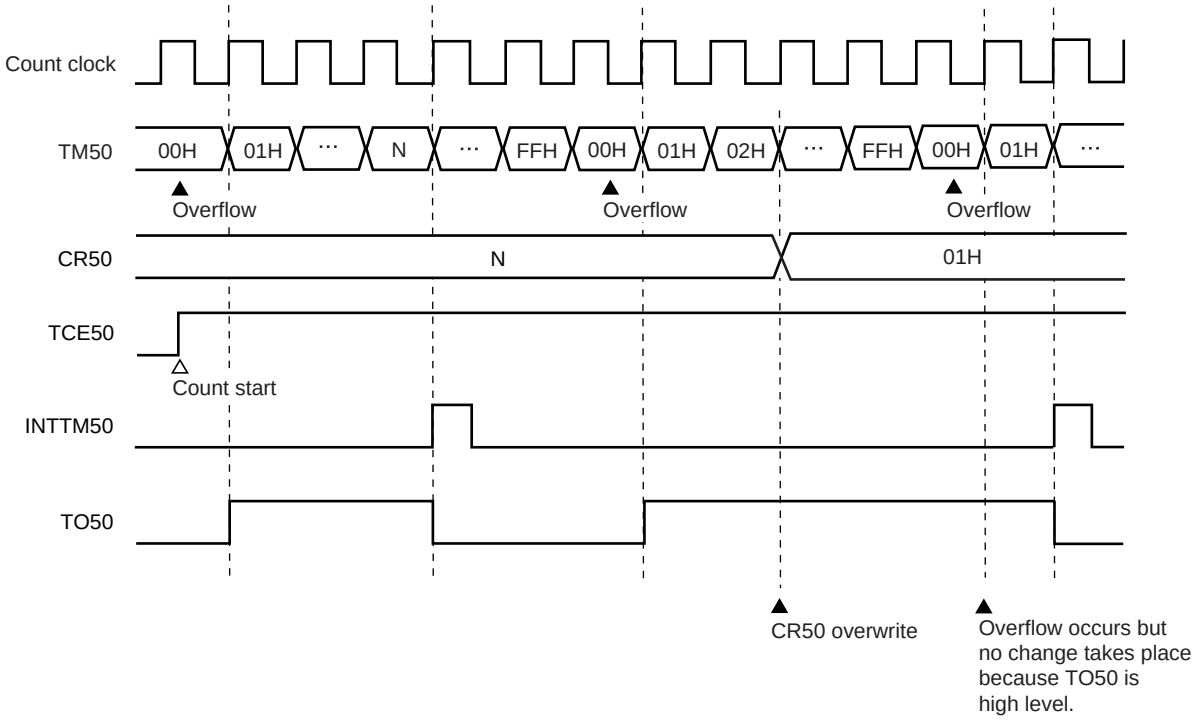


Figure 7-24. Operation Timing in PWM Free-Running Mode (When Both Edges Are Selected) (1/2)

(1) CR50 = Even number

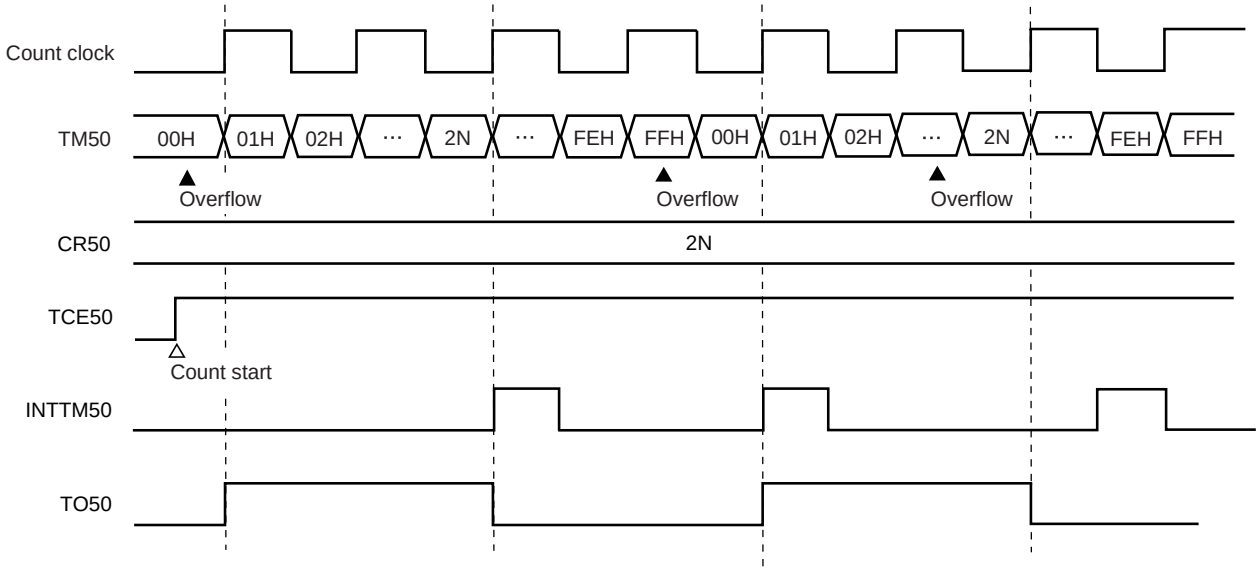
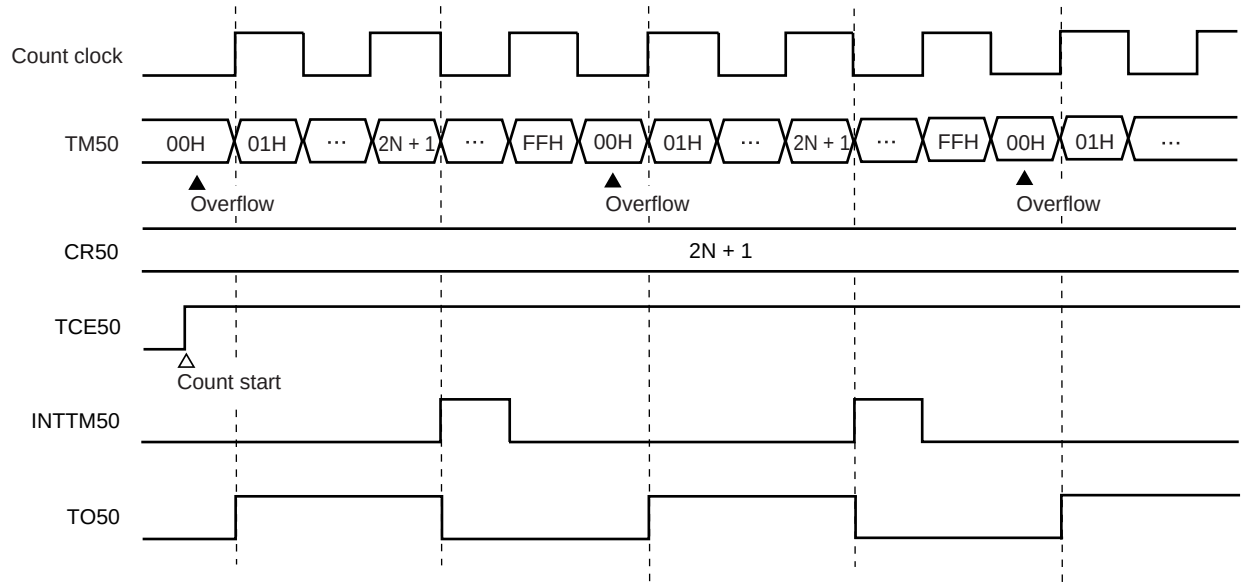


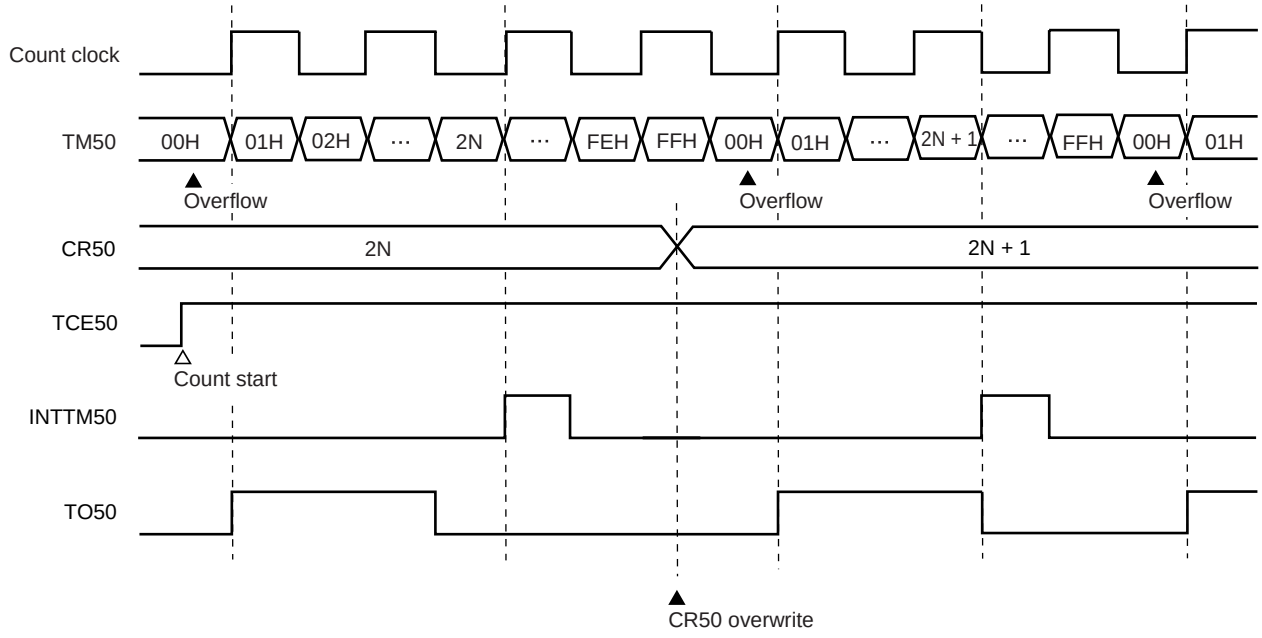
Figure 7-24. Operation Timing in PWM Free-Running Mode (When Both Edges Are Selected) (2/2)

(2) When CR50 = Odd number



Caution When both edges are selected, do not set CR50 to 00H, 01H, and FFH. If the CR50 is set to these values, PWM output may not be performed normally.

Figure 7-25. Operation Timing in PWM Free-Running Mode (When Both Edges Are Selected) (When CR50 Is Overwritten)



7.4.5 Operation as PWM output (timer 60)

In the PWM pulse generator mode, a pulse of any duty ratio can be output by setting a low-level width using CR60 and a high-level width using CRH60.

To operate timer 60 in PWM output mode, settings must be made in the following sequence.

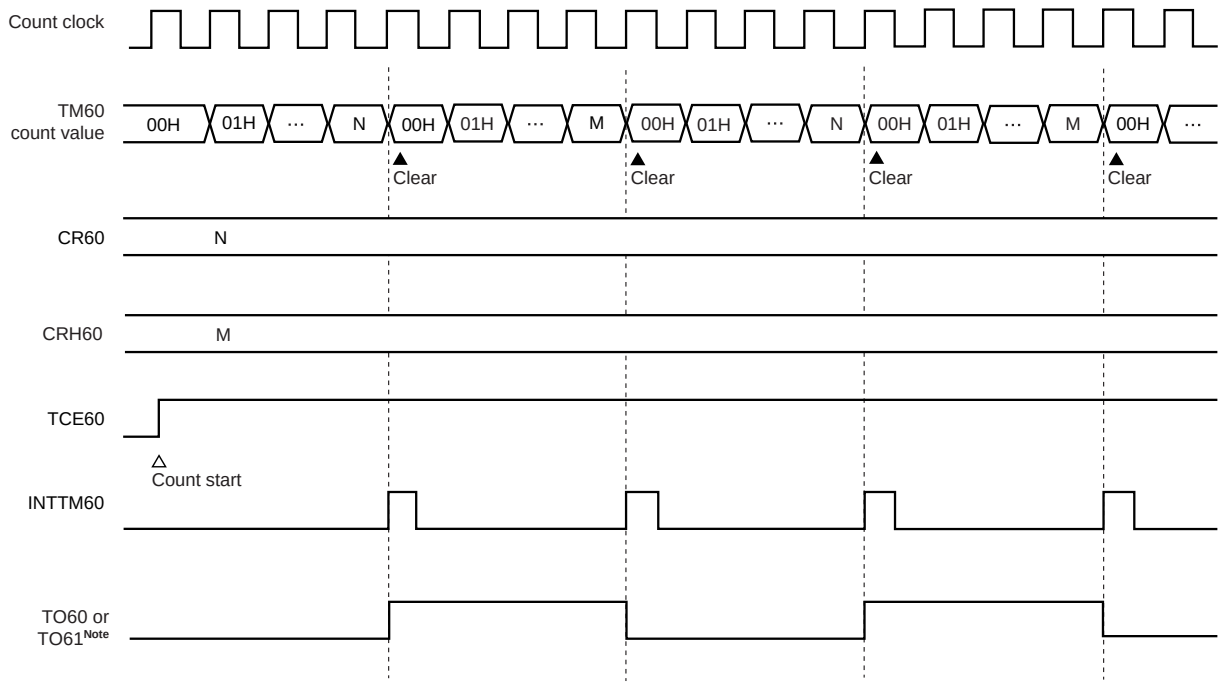
<1> Disable operation of TM60 (TCE60 = 0).
<2> Disable timer output of TO60 (TOE60 = 0).
<3> Set count values in CR60 and CRH60.
<4> Set the operation mode of timer 60 to the PWM pulse generator mode (see Figure 7-5).
<5> Set the count clock for timer 60.
<6> Set P32 to the output mode (PM32 = 0) and the P32 output latch to 0 and enable timer output of TO60 (TOE60 = 1).
<7> Enable the operation of TM60 (TCE60 = 1).

The operation in the PWM output mode is as follows.

<1> When the count value of TM60 matches the value set in CR60, an interrupt request signal (INTTM60) is generated and output of timer 60 is inverted, which makes the compare register switch from CR60 to CRH60.
<2> A match between TM60 and CR60 clears the TM60 value to 00H and then counting starts again.
<3> After that, when the count value of TM60 matches the value set in CRH60, an interrupt request signal (INTTM60) is generated and output of timer 60 is inverted again, which makes the compare register switch from CRH60 to CR60.
<4> A match between TM60 and CRH60 clears the TM60 value to 00H and then counting starts again.

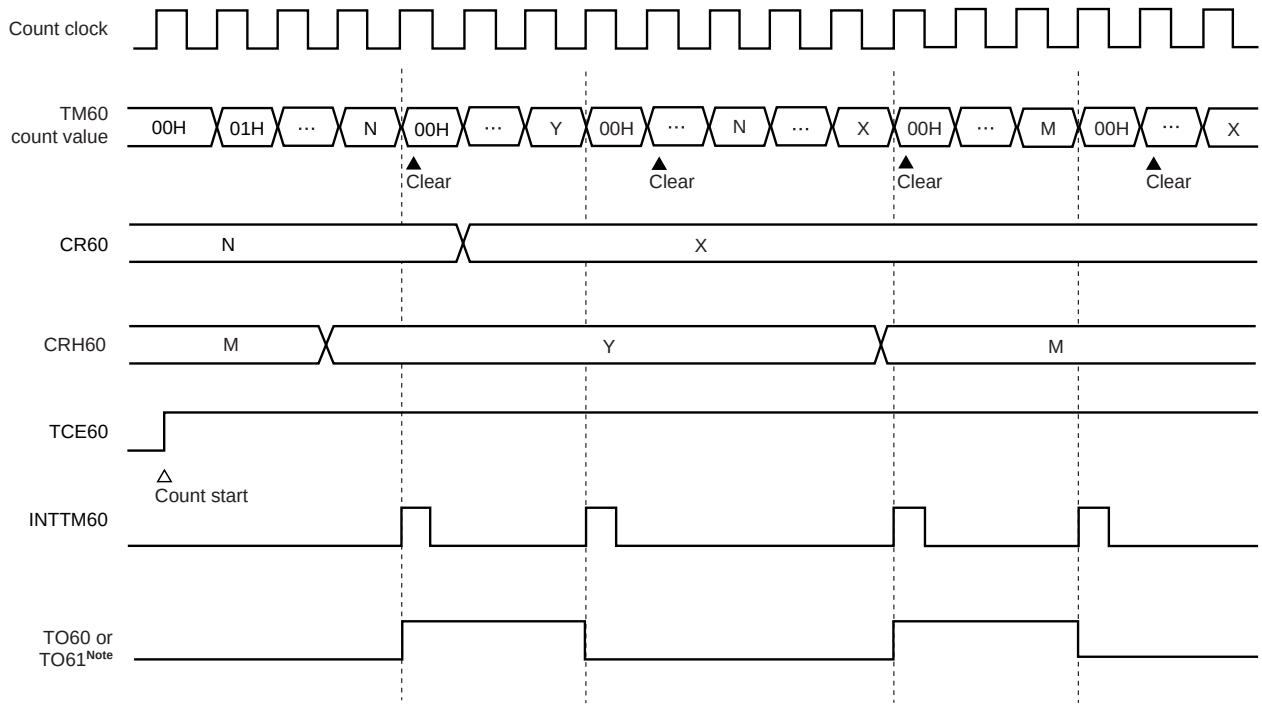
A pulse of any duty ratio is output by repeating <1> to <4> above. Figures 7-26 and 7-27 show the operation timing in the PWM output mode.

Figure 7-26. PWM Pulse Generator Mode Timing (Basic Operation)



Note The initial value of TO60 is low level when output is enabled (TOE60 = 1).

Figure 7-27. PWM Output Mode Timing (When CR60 and CRH60 Are Overwritten)



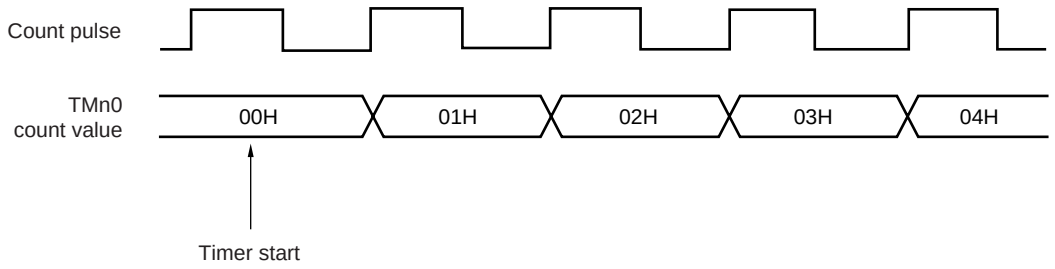
Note The initial value of TO60 is low level when output is enabled (TOE60 = 1).

7.5 Notes on Using 8-Bit Timer

(1) Error on starting timer

An error of up to 1 clock is included in the time between the timer being started and a match signal being generated. This is because 8-bit timer counter n0 (TMn0) is started asynchronously to the count pulse.

Figure 7-28. Start Timing of 8-Bit Timer Counter



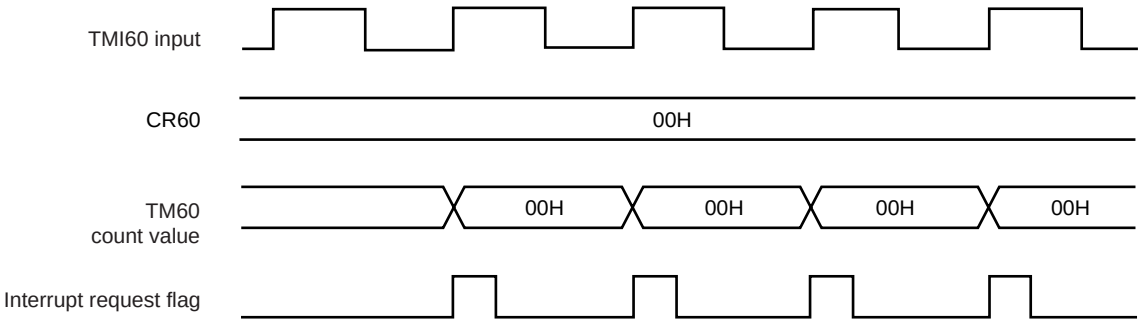
Remark n = 5, 6

(2) Setting of 8-bit compare register n0

8-bit compare register n0 (CRn0) can be set to 00H.

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

Figure 7-29. Timing of Operation as External Event Counter (8-Bit Resolution)



CHAPTER 8 WATCH TIMER

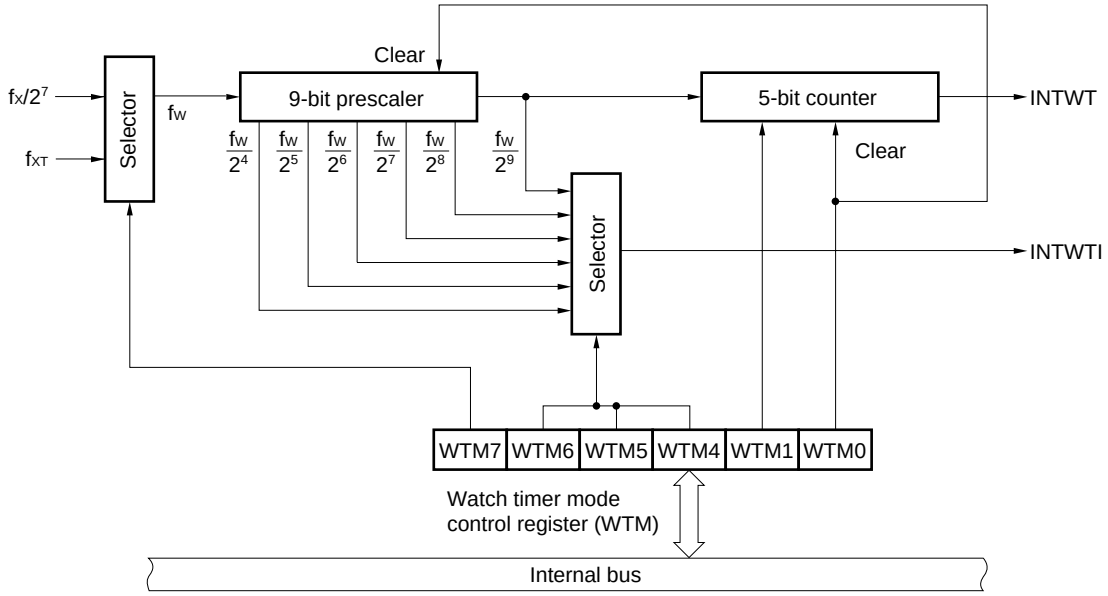
8.1 Watch Timer Functions

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 is a block diagram of the watch timer.

Figure 8-1. Block Diagram of Watch Timer



(1) Watch timer

The 4.19 MHz main system clock or 32.768 kHz subsystem clock is used to issue 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 (INTWTI) at specified intervals.

Table 8-1. Interval Generated Using the Interval Timer

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

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

8.2 Watch Timer Configuration

The watch timer includes the following hardware.

Table 8-2. Watch Timer Configuration

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

8.3 Watch Timer Control Register

The watch timer is controlled by the watch timer mode control register (WTM).

- **Watch timer mode control register (WTM)**
  WTM selects a count clock for the watch timer and specifies whether to enable operation 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 sets WTM to 00H.

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

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
WTM	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 stop
1	Started

WTM0	Watch timer operation
0	Operation disabled (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_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 to enable the watch timer to operate at 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.

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

8.4.2 Operation as interval timer

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

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

Table 8-3. Interval Time of Interval Timer

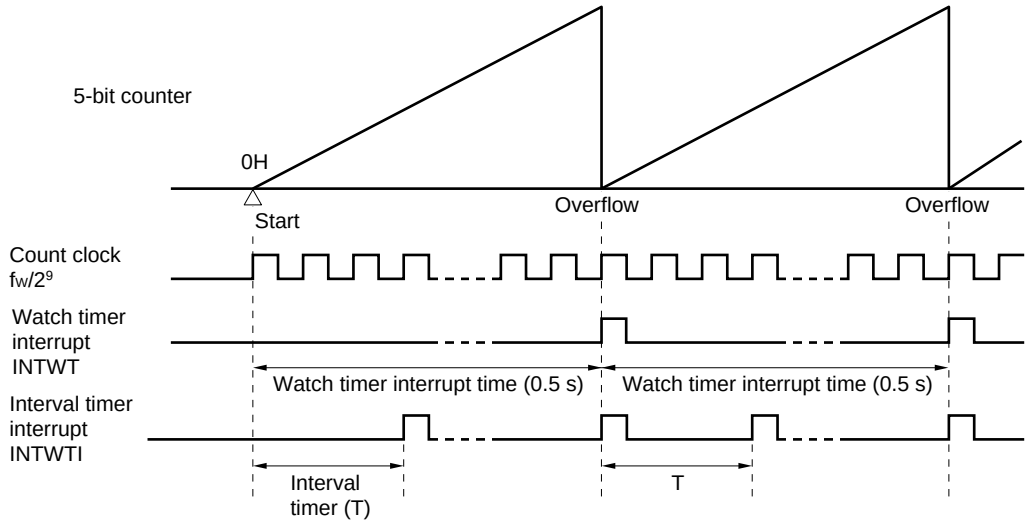
WTM6	WTM5	WTM4	Interval	At f_X = 5.0 MHz	At f_X = 4.19 MHz	At f_{XT} = 32.768 kHz
0	0	0	$2^4 \times 1/f_W$	409.6 μs	489 μs	488 μs
0	0	1	$2^5 \times 1/f_W$	819.2 μs	978 μs	977 μs
0	1	0	$2^6 \times 1/f_W$	1.64 ms	1.96 ms	1.95 ms
0	1	1	$2^7 \times 1/f_W$	3.28 ms	3.91 ms	3.91 ms
1	0	0	$2^8 \times 1/f_W$	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			

Remarks 1. f_X: Main system clock oscillation frequency

2. f_{XT}: Subsystem clock oscillation frequency

3. 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 operation is enabled by setting bit 0 (WTM0) of the watch mode timer mode control register (WTM) to 1, the interval until the first interrupt request (INTWT) is generated after the register is set does not exactly match the specification made with WTM3 (bit 3 of WTM). This is because there is a delay of one 9-bit prescaler output cycle until the 5-bit counter starts counting. Subsequently, however, the INTWT signal is generated at the specified intervals.

Remarks 1. f_W: Watch timer clock frequency

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

[MEMO]

CHAPTER 9 WATCHDOG TIMER

9.1 Watchdog Timer Functions

The watchdog timer has the following functions.

- Watchdog timer
- Interval timer

Caution Select the watchdog timer mode or interval timer mode by using the watchdog timer mode register (WDTM).

(1) Watchdog timer

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

Table 9-1. Watchdog Timer Runaway Detection Time

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

f_X: Main system clock oscillation frequency

(2) Interval timer

The interval timer generates an interrupt at an arbitrary preset interval.

Table 9-2. Interval Time

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

f_X: Main system clock oscillation frequency

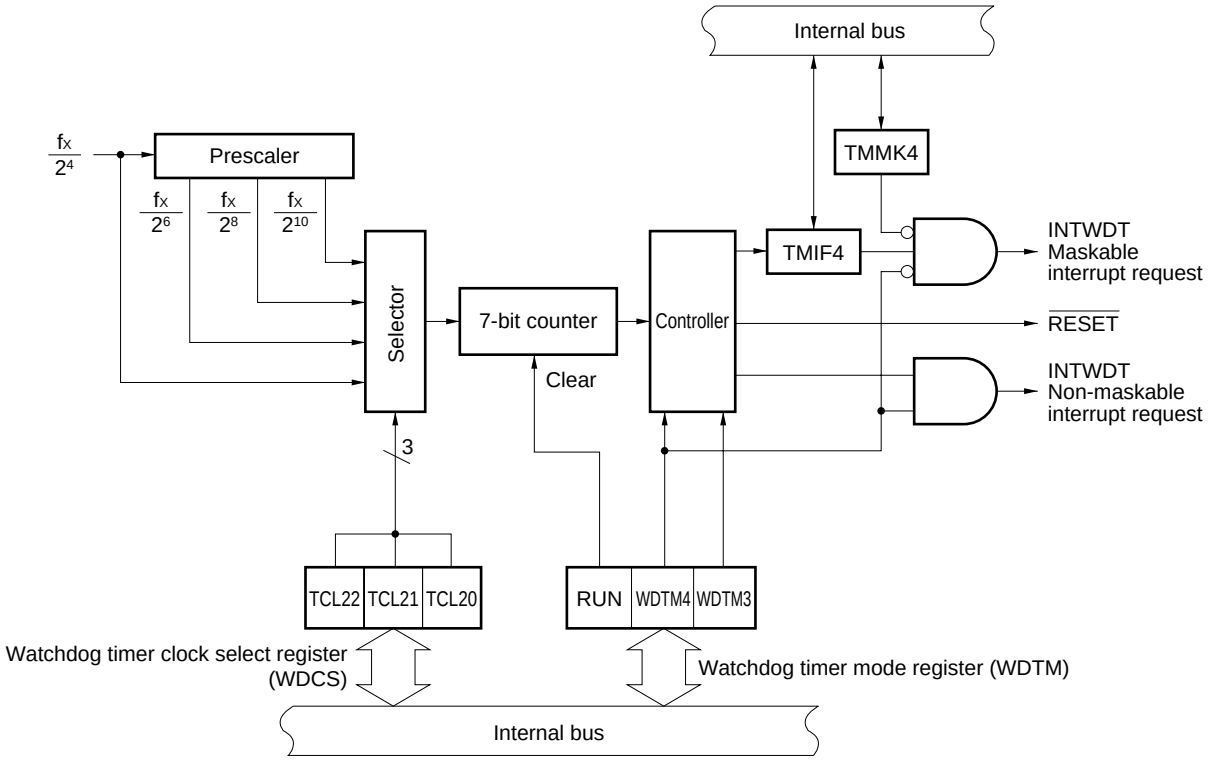
9.2 Watchdog Timer Configuration

The watchdog timer includes the following hardware.

Table 9-3. Configuration of Watchdog Timer

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

Figure 9-1. Block Diagram of Watchdog Timer



9.3 Watchdog Timer Control Registers

The watchdog timer is controlled by the following two registers.

- 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.

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

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

Figure 9-2. Format of Watchdog Timer Clock Select Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
WDCS	0	0	0	0	0	WDCS2	WDCS1	WDCS0	FF42H	00H	R/W

WDCS2	WDCS1	WDCS0	Watchdog timer count clock selection	Interval
0	0	0	$f_X/2^4$ (312.5 kHz)	$2^{11}/f_X$ (410 μs)
0	1	0	$f_X/2^6$ (78.1 kHz)	$2^{13}/f_X$ (1.64 ms)
1	0	0	$f_X/2^8$ (19.5 kHz)	$2^{15}/f_X$ (6.55 ms)
1	1	0	$f_X/2^{10}$ (4.88 kHz)	$2^{17}/f_X$ (26.2 ms)
Other than above			Setting prohibited	

Remarks 1. f_X : Main system clock oscillation frequency

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

(2) Watchdog timer mode register (WDTM)

This register sets the 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 sets WDTM to 00H.

Figure 9-3. Format of Watchdog Timer Mode Register

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

RUN	Watchdog timer operation selection[Note 1]
0	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 (Generates a maskable interrupt upon overflow occurrence.)[Note 3]
1	0	Watchdog timer mode 1 (Generates a non-maskable interrupt upon overflow occurrence.)
1	1	Watchdog timer mode 2 (Starts reset operation upon overflow occurrence.)

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 operation 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 the watchdog timer clock select register (WDCS).

2. To set watchdog timer mode 1 or 2, set WDTM4 to 1 after confirming TMIF4 (bit 0 of the interrupt request flag register 0 (IF0)) is set to 0. When watchdog timer mode 1 or 2 is selected with TMIF4 set to 1, a non-maskable interrupt is generated upon the completion of rewriting WDTM4.

9.4 Watchdog Timer Operation

9.4.1 Operation as watchdog timer

The watchdog timer detects a program 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 0 to 2 (WDCS0 to 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, a system reset signal or a non-maskable interrupt is generated, depending on the value of bit 3 (WDTM3) of WDTM.

The watchdog timer continues operation in HALT mode, but stops in STOP mode. Therefore, first set RUN to 1 to clear the watchdog timer before executing 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 selected as the CPU clock, the watchdog timer count operation is stopped. Even when the main system clock continues oscillating in this case, watchdog timer count operation is stopped.

Table 9-4. Watchdog Timer Runaway Detection Time

WDCS2	WDCS1	WDCS0	Runaway Detection Time	At f_X = 5.0 MHz
0	0	0	$2^{11} \times 1/f_X$	410 μs
0	1	0	$2^{13} \times 1/f_X$	1.64 ms
1	0	0	$2^{15} \times 1/f_X$	6.55 ms
1	1	0	$2^{17} \times 1/f_X$	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 the watchdog timer mode register (WDTM) are set to 0 and 1, respectively, the watchdog timer operates as an interval timer that repeatedly generates an interrupt at intervals specified by a preset count value.

Select a count clock (or interval) by setting bits 0 to 2 (WDCS0 to WDCS2) of the 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 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 HALT mode, but stops in STOP mode. Therefore, first set RUN to 1 to clear the interval timer before executing the STOP instruction.

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

2. The interval time may be up to 0.8% shorter than the set time when WDTM has just been set.

Table 9-5. Interval Time of Interval Timer

WDCS2	WDCS1	WDCS0	Interval	At f_X = 5.0 MHz
0	0	0	$2^{11} \times 1/f_X$	410 μs
0	1	0	$2^{13} \times 1/f_X$	1.64 ms
1	0	0	$2^{15} \times 1/f_X$	6.55 ms
1	1	0	$2^{17} \times 1/f_X$	26.2 ms

f_X: Main system clock oscillation frequency

CHAPTER 10 8-BIT A/D CONVERTER (μPD789426 AND 789446 SUBSERIES)

10.1 8-Bit A/D Converter Functions

The 8-bit A/D converter is an 8-bit resolution converter used to convert analog inputs into digital signals. This converter can control six channels (ANI0 to ANI5) of analog inputs.

A/D conversion can only be started by software.

One of analog inputs ANI0 to ANI5 is selected for A/D conversion. A/D conversion is performed repeatedly, with an interrupt request (INTAD0) being issued each time A/D conversion is complete.

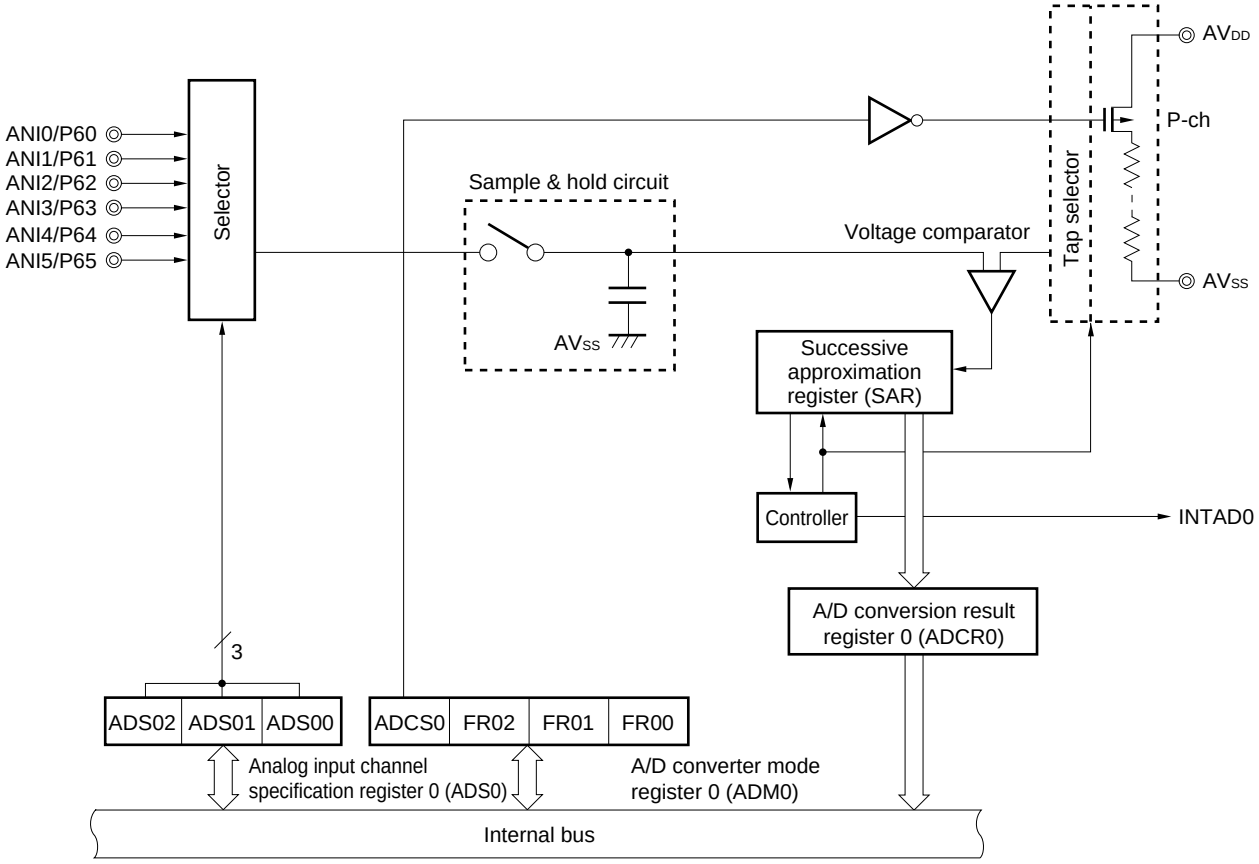
10.2 8-Bit A/D Converter Configuration

The 8-bit A/D converter includes the following hardware.

Table 10-1. Configuration of 8-Bit A/D Converter

Item	Configuration
Analog inputs	6 channels (ANI0 to ANI5)
Registers	Successive approximation register (SAR) A/D conversion result register 0 (ADCR0)
Control registers	A/D converter mode register 0 (ADM0) Analog input channel specification register 0 (ADS0)

Figure 10-1. Block Diagram of 8-Bit A/D Converter



(1) Successive approximation register (SAR)

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

Upon receiving all the bits, down to the least significant bit (LSB), that is, upon the completion of A/D conversion, the SAR sends its contents to A/D conversion result register 0 (ADCR0).

(2) A/D conversion result register 0 (ADCR0)

ADCR0 holds the result of A/D conversion. Each time A/D conversion ends, the conversion result in the successive approximation register is loaded into ADCR0, which is an 8-bit register.

ADCR0 can be read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes ADCR0 undefined.

(3) Sample & hold circuit

The sample & hold circuit samples consecutive analog inputs from the input circuit, one by one, and sends them to the voltage comparator. The sampled analog input voltage is held during A/D conversion.

(4) Voltage comparator

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

(5) Series resistor string

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

(6) ANI0 to ANI5

Pins ANI0 to ANI5 are the 6-channel analog input pins for the A/D converter. They are used to receive the analog signals for A/D conversion.

Caution Do not supply pins ANI0 to ANI5 with voltages that fall outside the rated range. If a voltage greater than AV_{DD} or less than AV_{SS} (even if within the absolute maximum rating) is applied to any of these pins, the conversion value for the corresponding channel will be undefined. Furthermore, the conversion values for the other channels may also be affected.

(7) AV_{SS} pin

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

(8) AV_{DD} pin

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

10.3 8-Bit A/D Converter Control Registers

The 8-bit A/D converter is controlled by the following two registers.

- A/D converter mode register 0 (ADM0)
- Analog input channel specification register 0 (ADS0)

(1) A/D converter mode register 0 (ADM0)

ADM0 specifies the conversion time for analog inputs. It also specifies whether to enable conversion.

ADM0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ADM0 to 00H.

Figure 10-2. Format of A/D Converter Mode Register 0

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
ADM0	ADCS0	0	FR02	FR01	FR00	0	0	0	FF80H	00H	R/W

ADCS0	A/D conversion control
0	Conversion disabled
1	Conversion enabled

FR02	FR01	FR00	A/D conversion time selection[Note 1]
0	0	0	144/fx (28.8 μs)
0	0	1	120/fx (24 μs)
0	1	0	96/fx (19.2 μs)
1	0	0	72/fx (14.4 μs)
1	0	1	60/fx (Setting prohibited[Note 2])
1	1	0	48/fx (Setting prohibited[Note 2])
Other than above			Setting prohibited

Notes 1. The specifications of FR02, FR01, and FR00 must be such that the A/D conversion time is at least 14 μs.

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

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

2. The result of conversion performed immediately after setting ADCS0 is undefined.

3. The conversion result may be undefined after clearing ADCS0.

Remarks 1. fx: Main system clock oscillation frequency

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

(2) Analog input channel specification register 0 (ADS0)

ADS0 specifies the port used to input the analog voltage to be converted to a digital signal.

ADS0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ADS0 to 00H.

Figure 10-3. Format of Analog Input Channel Specification Register 0

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADS0	0	0	0	0	0	ADS02	ADS01	ADS00	FF84H	00H	R/W

ADS02	ADS01	ADS00	Analog input channel specification
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
Other than above			Setting prohibited

Caution Bits 3 to 7 must be set to 0.

10.4 8-Bit A/D Converter Operation

10.4.1 Basic operation of 8-bit A/D converter

<1> Select a channel for A/D conversion, using analog input channel specification register 0 (ADS0).

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

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

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

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

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

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

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

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

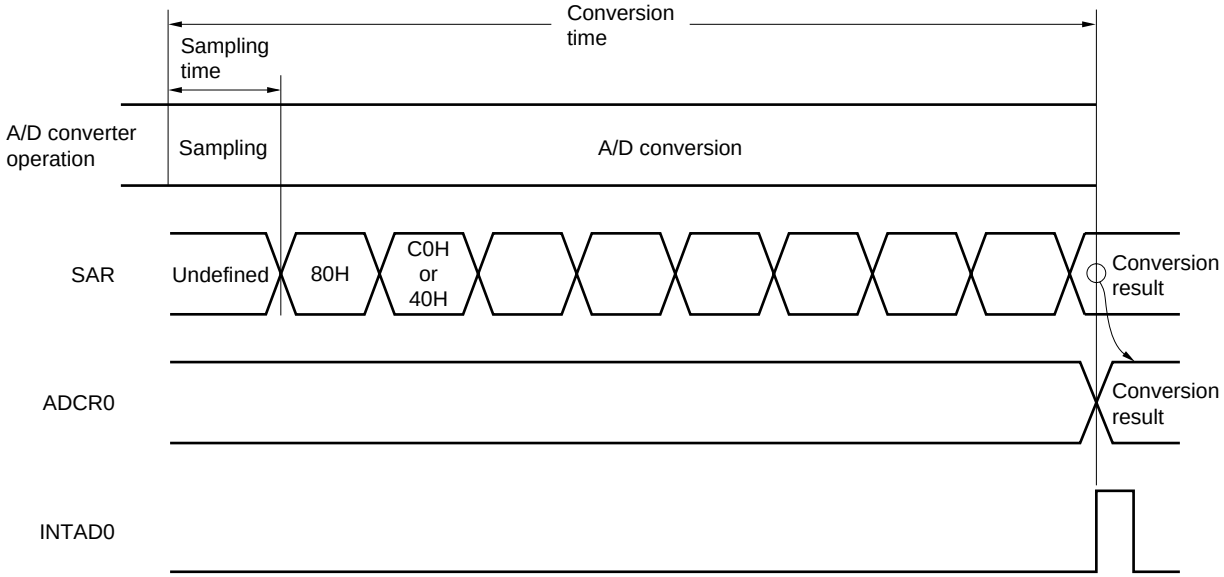
<7> Comparison is repeated until bit 0 of SAR is reached.

<8> When comparison is completed for all of the 8 bits, a significant digital result is left in SAR. This value is sent to and latched in A/D conversion result register 0 (ADCR0). At the same time, it is possible to generate an A/D conversion end interrupt request (INTAD0).

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

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

Figure 10-4. Basic Operation of 8-Bit A/D Converter



A/D conversion continues until bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) is reset (0) by software.

If an attempt is made to write to ADM0 or analog input channel specification register 0 (ADS0) during A/D conversion, the ongoing A/D conversion is canceled. In this case, A/D conversion is restarted from the beginning, if ADCS0 is set (1).

$\overline{\text{RESET}}$ input makes A/D conversion result register 0 (ADCR0) undefined.

10.4.2 Input voltage and conversion result

The relationships between the analog input voltage at the analog input pins (ANI0 to ANI5) and the A/D conversion result (A/D conversion result register 0 (ADCR0)) are represented by:

$$ADCR0 = INT\left(\frac{V_{IN}}{AV_{DD}} \times 256 + 0.5\right)$$

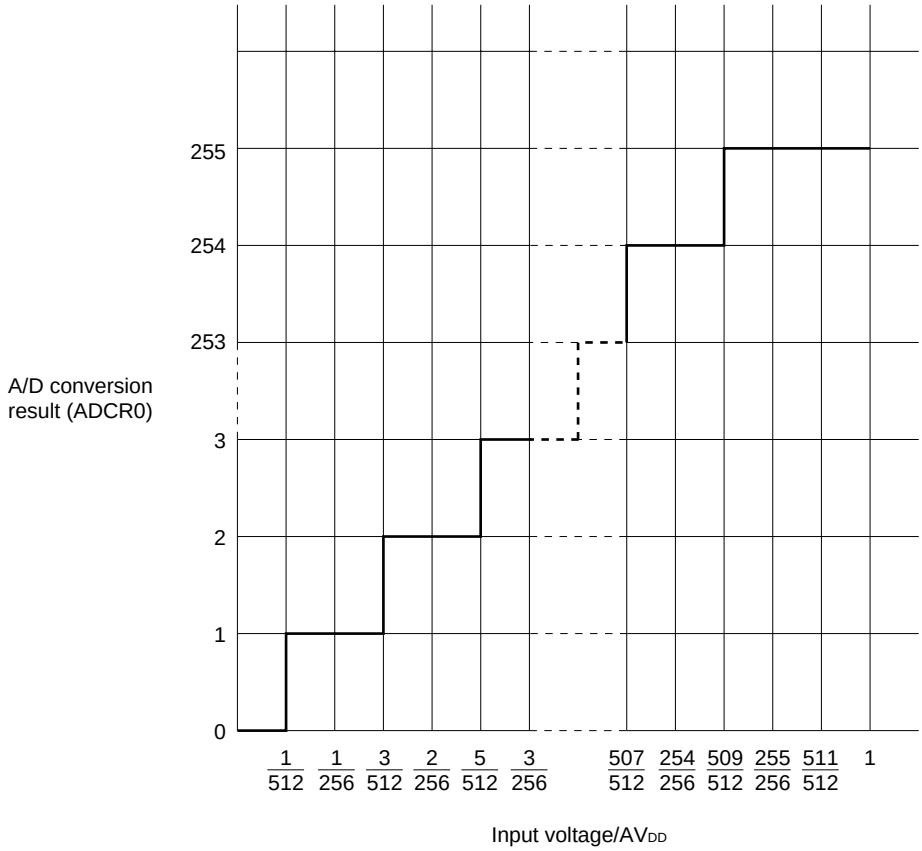
or

$$(ADCR0 - 0.5) \times \frac{AV_{DD}}{256} \leq V_{IN} < (ADCR0 + 0.5) \times \frac{AV_{DD}}{256}$$

INT( ): Function that returns the integer part of a parenthesized value
V_{IN}: Analog input voltage
AV_{DD}: Supply voltage for the A/D converter
ADCR0: Value in A/D conversion result register 0 (ADCR0)

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

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



10.4.3 Operation mode of 8-bit A/D converter

The A/D converter is initially in select mode. In this mode, analog input channel specification register 0 (ADS0) is used to select an analog input channel from ANI0 to ANI5 for A/D conversion.

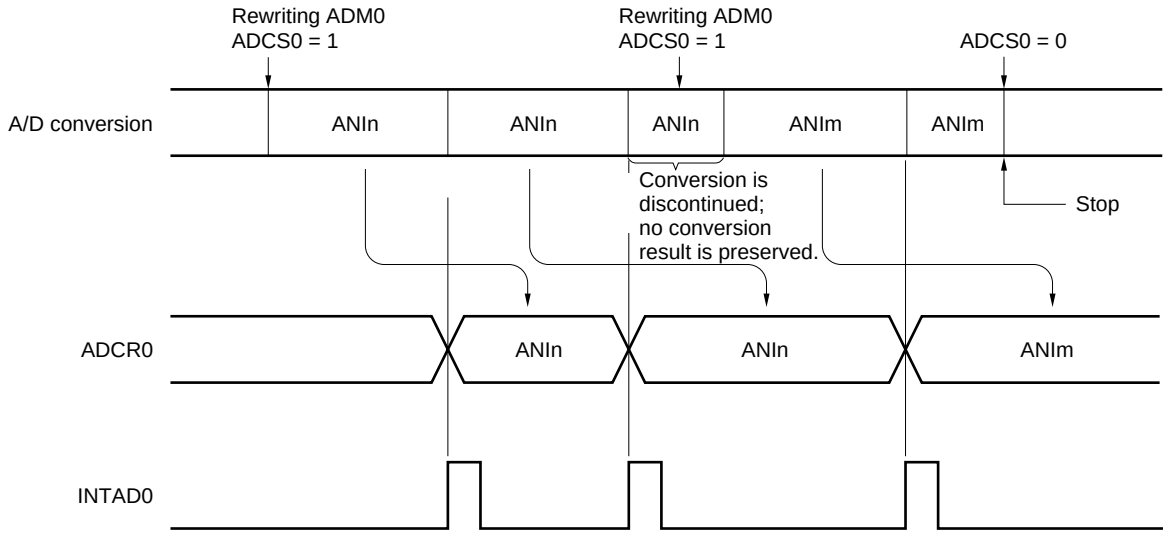
A/D conversion can be started only by software, that is, by setting A/D converter mode register 0 (ADM0).

The A/D conversion result is saved to A/D conversion result register 0 (ADCR0). At the same time, an interrupt request signal (INTAD0) is generated.

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

  Setting bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) to 1 triggers A/D conversion for a voltage applied to the analog input pin specified in analog input channel specification register 0 (ADS0). Upon completion of A/D conversion, the conversion result is saved to A/D conversion result register 0 (ADCR0). At the same time, an interrupt request signal (INTAD0) is generated. Once A/D conversion is activated, and completed, another session of A/D conversion is started. A/D conversion is repeated until new data is written to ADM0. If data where ADCS0 is 1 is written to ADM0 again during A/D conversion, the ongoing session of A/D conversion is discontinued, and a new session of A/D conversion begins for the new data. If data where ADCS0 is 0 is written to ADM0 again during A/D conversion, A/D conversion is stopped immediately.

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



Remarks 1. n = 0 to 5

2. m = 0 to 5

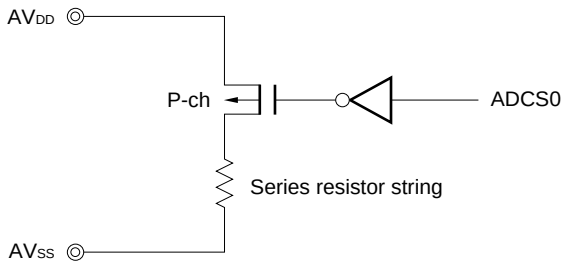
10.5 Cautions Related to 8-Bit A/D Converter

(1) Current consumption in standby mode

In standby mode, the A/D converter stops operation. Stopping conversion (bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) = 0) can reduce the current consumption.

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

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



(2) Input range for pins ANI0 to ANI5

Be sure to keep the input voltage at ANI0 to ANI5 within the rating. If a voltage not lower than AV_{DD} or not higher than AV_{SS} (even within the absolute maximum rating) is input into a conversion channel, the conversion output of the channel becomes undefined, which may affect the conversion output of the other channels.

(3) Conflict

<1> Conflict between writing to A/D conversion result register 0 (ADCR0) at the end of conversion and reading from ADCR0 using instruction

Reading from ADCR0 takes precedence. After reading, the new conversion result is written to ADCR0.

<2> Conflict between writing to ADCR0 at the end of conversion and writing to A/D converter mode register 0 (ADM0) or analog input channel specification register 0 (ADS0)

Writing to ADM0 or ADS0 takes precedence. ADCR0 is not written to. No A/D conversion end interrupt request signal (INTAD0) is generated.

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

The first A/D conversion value immediately after A/D conversion has been started is undefined. Poll the A/D conversion end interrupt request (INTAD0) and drop the first conversion result.

(5) Timing of undefined A/D conversion result

The A/D conversion value may become undefined if the timing of the completion of A/D conversion and that to stop the A/D conversion operation conflict. Therefore, read the A/D conversion result while the A/D conversion operation is in progress. To read the A/D conversion result after the A/D conversion operation has been stopped, stop the A/D conversion operation before the next conversion operation is completed. Figures 10-8 and 10-9 show the timing at which the conversion result is read.

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

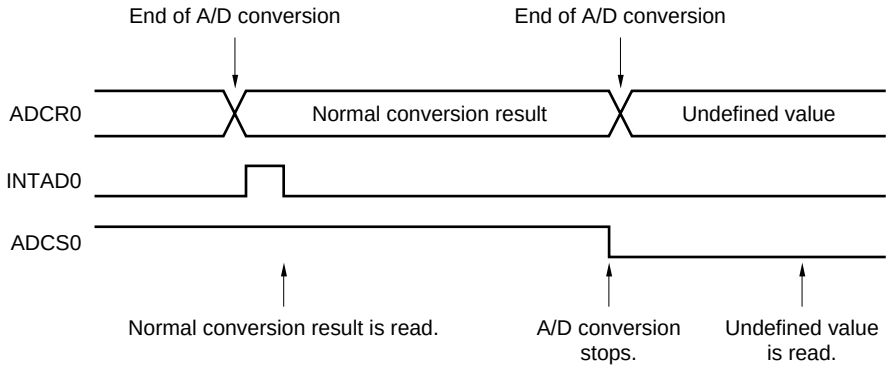
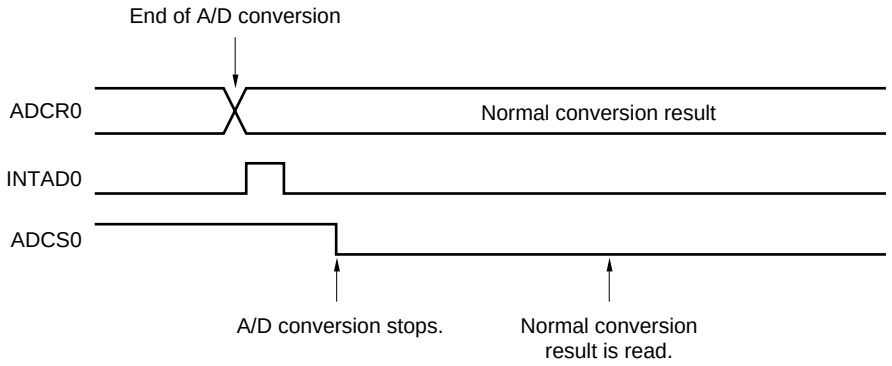


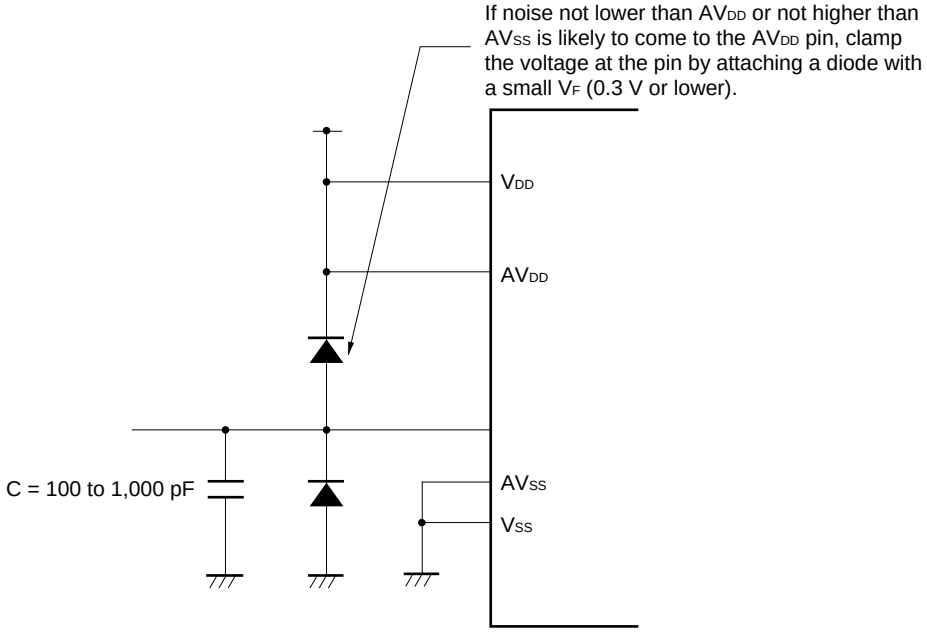
Figure 10-9. Conversion Result Read Timing (If Conversion Result Is Normal)



(6) Noise prevention

To maintain a resolution of 8 bits, watch for noise to the AV_{DD} and ANI0 to ANI5 pins. The higher the output impedance of the analog input source, the larger the effect by noise. To reduce noise, attach an external capacitor to the relevant pins as shown in Figure 10-10.

Figure 10-10. Analog Input Pin Treatment



(7) ANI0 to ANI5

The analog input pins (ANI0 to ANI5) are alternate-function pins. They are also used as port pins (P60 to P65).

If any of ANI0 to ANI5 has been selected for A/D conversion, do not execute input instructions for the ports; otherwise the conversion resolution may be reduced.

If a digital pulse is applied to a pin adjacent to the analog input pins during A/D conversion, coupling noise may occur that prevents an A/D conversion result from being obtained as expected. Avoid applying a digital pulse to pins adjacent to the analog input pins during A/D conversion.

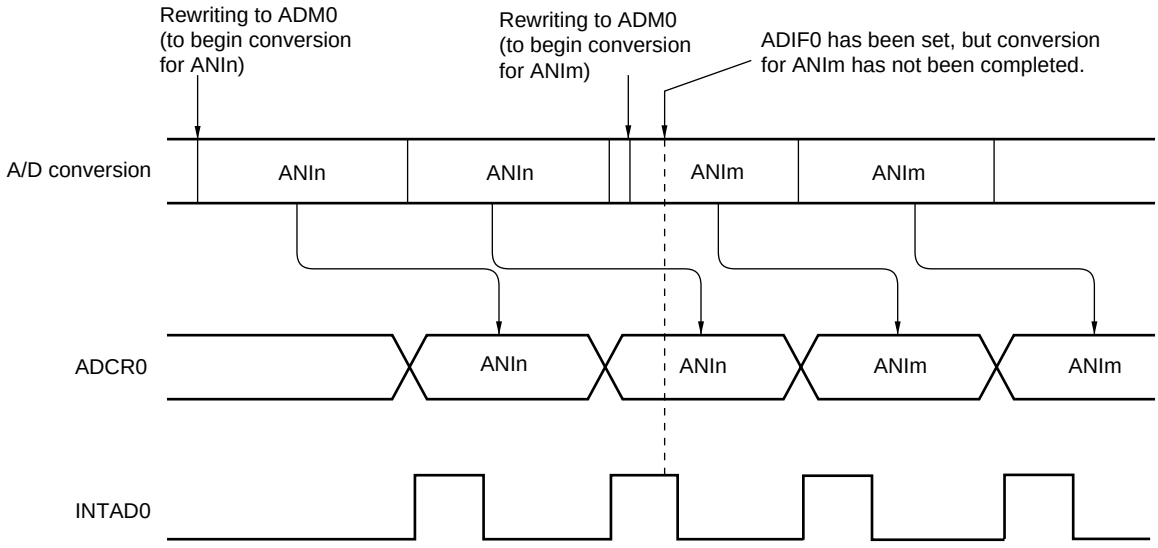
(8) Interrupt request flag (ADIF0)

Changing the contents of A/D converter mode register 0 (ADM0) does not clear the interrupt request flag (ADIF0).

If the analog input pins are changed during A/D conversion, therefore, the A/D conversion result and the conversion end interrupt request flag may reflect the previous analog input immediately before writing to ADM0 occurs. In this case, ADIF0 may already be set if it is read-accessed immediately after ADM0 is write-accessed, even when A/D conversion has not been completed for the new analog input.

In addition, when A/D conversion is restarted, ADIF0 must be cleared beforehand.

Figure 10-11. A/D Conversion End Interrupt Request Generation Timing



Remarks 1. n = 0 to 5

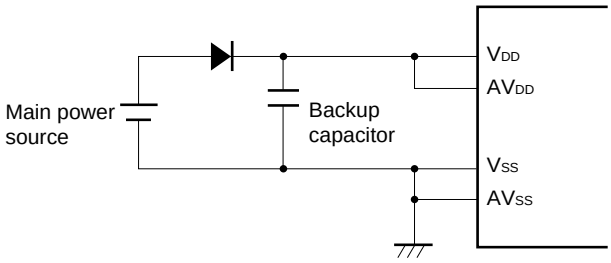
2. m = 0 to 5

(9) AV_{DD} pin

The AV_{DD} pin is used to supply power to the analog circuit. It is also used to supply power to the ANI0 to ANI5 input circuit.

If your application is designed to be changed to backup power, the AV_{DD} pin must be supplied with the same voltage level as the V_{DD} pin, as shown in Figure 10-12.

Figure 10-12. AV_{DD} Pin Handling



(10) AV_{DD} pin input impedance

A series resistor string of several ten of kΩ is connected between the AV_{DD} and AV_{SS} pins. Consequently, if the output impedance of the reference voltage supply is high, the reference voltage supply will form a parallel connection with the series resistor string, creating a large reference voltage differential.

[MEMO]

CHAPTER 11 10-BIT A/D CONVERTER (μPD789436 AND 789456 SUBSERIES)

11.1 10-Bit A/D Converter Functions

The 10-bit A/D converter is a 10-bit resolution converter used to convert analog inputs into digital signals. This converter can control six channels (ANI0 to ANI5) of analog inputs.

A/D conversion can only be started by software.

One of analog inputs ANI0 to ANI5 is selected for A/D conversion. A/D conversion is performed repeatedly, with an interrupt request (INTAD0) being issued each time A/D conversion is complete.

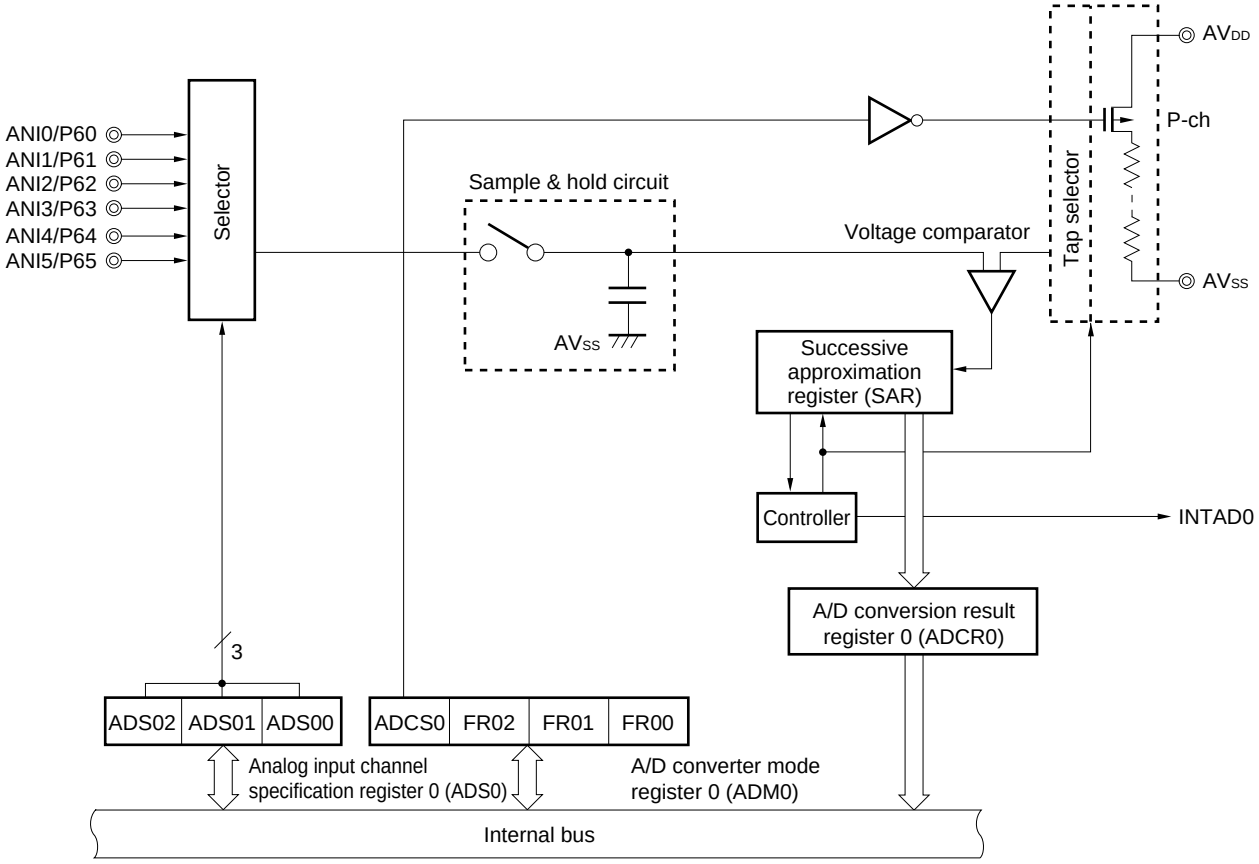
11.2 10-Bit A/D Converter Configuration

The 10-bit A/D converter includes the following hardware.

Table 11-1. Configuration of 10-Bit A/D Converter

Item	Configuration
Analog inputs	6 channels (ANI0 to ANI5)
Registers	Successive approximation register (SAR) A/D conversion result register 0 (ADCR0)
Control registers	A/D converter mode register 0 (ADM0) Analog input channel specification register 0 (ADS0)

Figure 11-1. Block Diagram of 10-Bit A/D Converter



(1) Successive approximation register (SAR)

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

Upon receiving all the bits, down to the least significant bit (LSB), that is, upon the completion of A/D conversion, the SAR sends its contents to A/D conversion result register 0 (ADCR0).

(2) A/D conversion result register 0 (ADCR0)

ADCR0 holds the result of A/D conversion. Each time A/D conversion ends, the conversion result in the successive approximation register is loaded into ADCR0, which is a 10-bit register.

ADCR0 can be read with a 16-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes ADCR0 undefined.

	ADCR0H (FF14H)										ADCR0L (FF14H)						Address	After reset	R/W
Symbol											0	0	0	0	0	0	FF14H, FF15H	0000H	R

Caution When the μPD78F9436, a flash memory version of the μPD789425 or μPD789426, is used, this register can be accessed in 8-bit units. However, only an object file assembled with the μPD789425 or μPD789426 can be used. The same is also true for the μPD78F9456, a flash memory version of the μPD789445 or μPD789446: This register can be accessed in 8-bit units, but only an object file assembled with the μPD789445 or μPD789446 can be used.

(3) Sample & hold circuit

The sample & hold circuit samples consecutive analog inputs from the input circuit, one by one, and sends them to the voltage comparator. The sampled analog input voltage is held during A/D conversion.

(4) Voltage comparator

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

(5) Series resistor string

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

(6) ANI0 to ANI5

Pins ANI0 to ANI5 are the 6-channel analog input pins for the A/D converter. They are used to receive the analog signals for A/D conversion.

Caution Do not supply pins ANI0 to ANI5 with voltages that fall outside the rated range. If a voltage greater than AV_{DD} or less than AV_{SS} (even if within the absolute maximum rating) is applied to any of these pins, the conversion value for the corresponding channel will be undefined. Furthermore, the conversion values for the other channels may also be affected.

(7) AV_{SS} pin

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

(8) AV_{DD} pin

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

11.3 10-Bit A/D Converter Control Registers

The 10-bit A/D converter is controlled by the following two registers.

- A/D converter mode register 0 (ADM0)
- Analog input channel specification register 0 (ADS0)

(1) A/D converter mode register 0 (ADM0)

ADM0 specifies the conversion time for analog inputs. It also specifies whether to enable conversion.

ADM0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ADM0 to 00H.

Figure 11-2. Format of A/D Converter Mode Register 0

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
ADM0	ADCS0	0	FR02	FR01	FR00	0	0	0	FF80H	00H	R/W

ADCS0	A/D conversion control
0	Conversion disabled
1	Conversion enabled

FR02	FR01	FR00	A/D conversion time selection[Note 1]
0	0	0	144/fx (28.8 μs)
0	0	1	120/fx (24 μs)
0	1	0	96/fx (19.2 μs)
1	0	0	72/fx (14.4 μs)
1	0	1	60/fx (Setting prohibited[Note 2])
1	1	0	48/fx (Setting prohibited[Note 2])
Other than above			Setting prohibited

Notes 1. The specifications of FR02, FR01, and FR00 must be such that the A/D conversion time is at least 14 μs.

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

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

2. The result of conversion performed immediately after setting ADCS0 is undefined.

3. The conversion result may be undefined after clearing ADCS0.

Remarks 1. fx: Main system clock oscillation frequency

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

(2) Analog input channel specification register 0 (ADS0)

ADS0 specifies the port used to input the analog voltage to be converted to a digital signal.

ADS0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ADS0 to 00H.

Figure 11-3. Format of Analog Input Channel Specification Register 0

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADS0	0	0	0	0	0	ADS02	ADS01	ADS00	FF84H	00H	R/W

ADS02	ADS01	ADS00	Analog input channel specification
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
Other than above			Setting prohibited

Caution Bits 3 to 7 must be set to 0.

11.4 10-Bit A/D Converter Operation

11.4.1 Basic operation of 10-bit A/D converter

<1> Select a channel for A/D conversion, using analog input channel specification register 0 (ADS0).

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

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

<4> Bit 9 of the successive approximation register (SAR) is set. The series resistor string tap voltage at the tap selector is set to half of AV_{DD}.

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

<6> Bit 8 of SAR is set automatically, and comparison shifts to the next stage. The next tap voltage of the series resistor string is selected according to bit 9, which reflects the previous comparison result, as follows:

- Bit 9 = 1: Three quarters of AV_{DD}
- Bit 9 = 0: One quarter of AV_{DD}

The tap voltage is compared with the analog input voltage. Bit 8 is set or reset according to the result of comparison.

- Analog input voltage ≥ tap voltage: Bit 8 = 1
- Analog input voltage < tap voltage: Bit 8 = 0

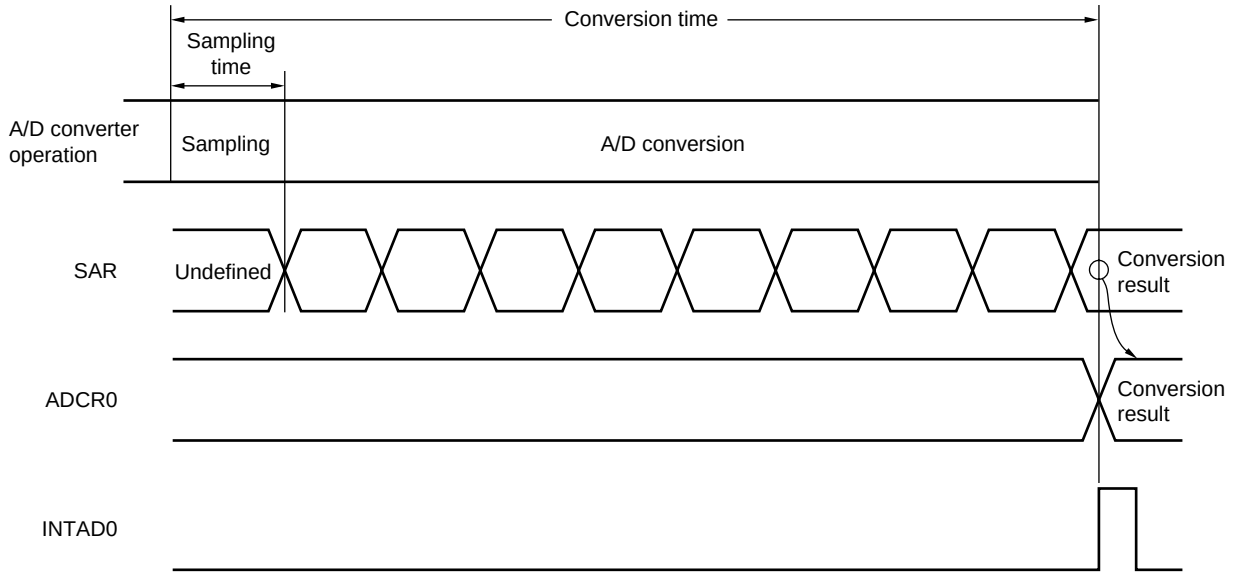
<7> Comparison is repeated until bit 0 of SAR is reached.

<8> When comparison is completed for all of the 10 bits, a significant digital result is left in SAR. This value is sent to and latched in A/D conversion result register 0 (ADCR0). At the same time, it is possible to generate an A/D conversion end interrupt request (INTAD0).

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

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

Figure 11-4. Basic Operation of 10-Bit A/D Converter



A/D conversion continues until bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) is reset (0) by software.

If an attempt is made to write to ADM0 or analog input channel specification register 0 (ADS0) during A/D conversion, the ongoing A/D conversion is canceled. In this case, A/D conversion is restarted from the beginning, if ADCS0 is set (1).

$\overline{\text{RESET}}$ input makes A/D conversion result register 0 (ADCR0) undefined.

11.4.2 Input voltage and conversion result

The relationships between the analog input voltage at the analog input pins (ANI0 to ANI5) and the A/D conversion result (A/D conversion result register 0 (ADCR0)) are represented by:

$$ADCR0 = INT\left(\frac{V_{IN}}{AV_{DD}} \times 1{,}024 + 0.5\right)$$

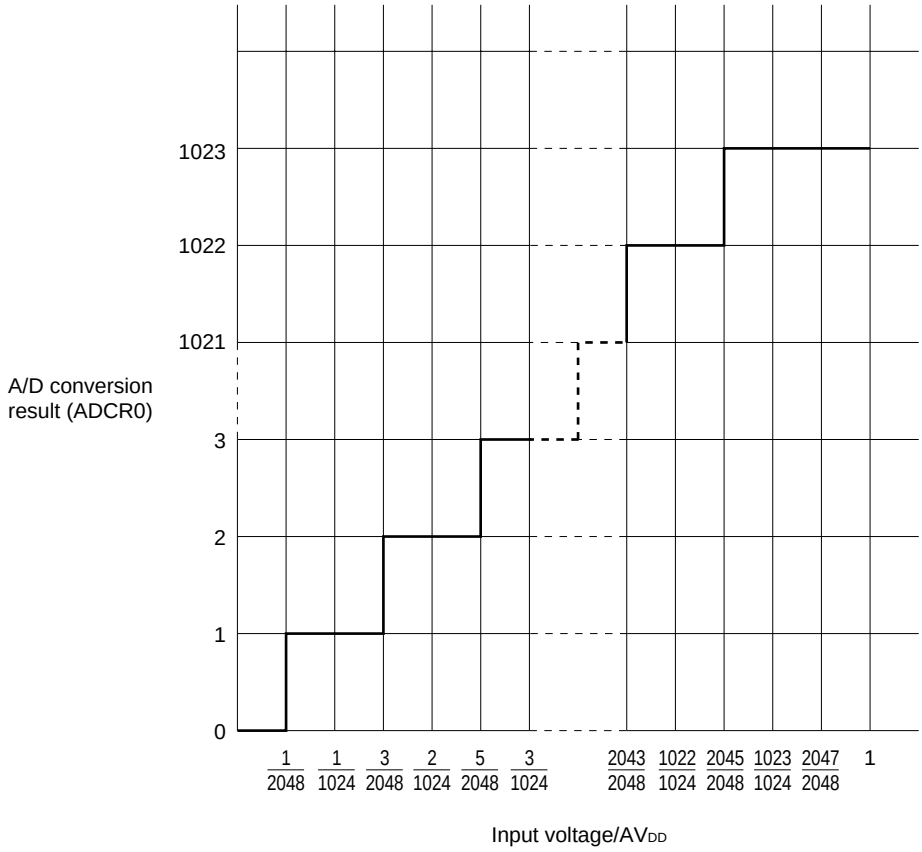
or

$$(ADCR0 - 0.5) \times \frac{AV_{DD}}{1{,}024} \leq V_{IN} < (ADCR0 + 0.5) \times \frac{AV_{DD}}{1{,}024}$$

INT( ): Function that returns the integer part of a parenthesized value
V_{IN}: Analog input voltage
AV_{DD}: Supply voltage for the A/D converter
ADCR0: Value in A/D conversion result register 0 (ADCR0)

Figure 11-5 shows the relationship between the analog input voltage and the A/D conversion result.

Figure 11-5. Relationship Between Analog Input Voltage and A/D Conversion Result



11.4.3 Operation mode of 10-bit A/D converter

The A/D converter is initially in select mode. In this mode, analog input channel specification register 0 (ADS0) is used to select an analog input channel from ANI0 to ANI5 for A/D conversion.

A/D conversion can be started only by software, that is, by setting A/D converter mode register 0 (ADM0).

The A/D conversion result is saved to A/D conversion result register 0 (ADCR0). At the same time, an interrupt request signal (INTAD0) is generated.

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

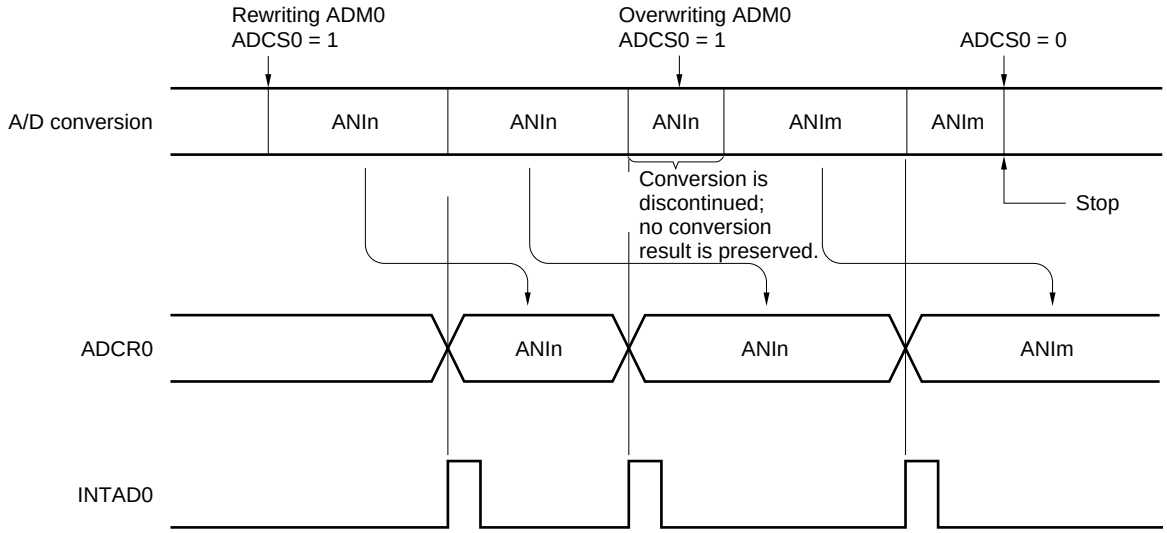
  Setting bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) to 1 triggers A/D conversion for a voltage applied to the analog input pin specified in A/D input selection register 0 (ADS0).

  Upon completion of A/D conversion, the conversion result is saved to A/D conversion result register 0 (ADCR0). At the same time, an interrupt request signal (INTAD0) is generated. Once A/D conversion is activated, and completed, another session of A/D conversion is started. A/D conversion is repeated until new data is written to ADM0.

  If data where ADCS0 is 1 is written to ADM0 again during A/D conversion, the ongoing session of A/D conversion is discontinued, and a new session of A/D conversion begins for the new data.

  If data where ADCS0 is 0 is written to ADM0 again during A/D conversion, A/D conversion is stopped immediately.

Figure 11-6. Software-Started A/D Conversion



Remarks 1. n = 0 to 5

2. m = 0 to 5

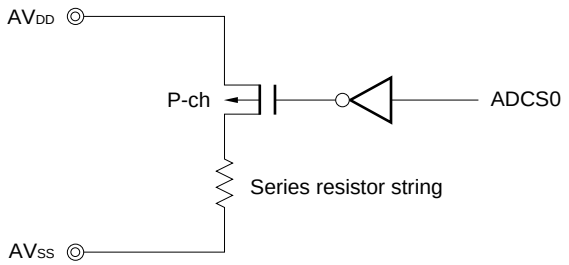
11.5 Cautions Related to 10-Bit A/D Converter

(1) Current consumption in standby mode

In standby mode, the A/D converter stops operation. Stopping conversion (bit 7 (ADCS0) of A/D converter mode register 0 (ADM0) = 0) can reduce the current consumption.

Figure 11-7 shows how to reduce the current consumption in standby mode.

Figure 11-7. How to Reduce Current Consumption in Standby Mode



(2) Input range for pins ANI0 to ANI5

Be sure to keep the input voltage at ANI0 to ANI5 within the rating. If a voltage not lower than AV_{DD} or not higher than AV_{SS} (even within the absolute maximum rating) is input into a conversion channel, the conversion output of the channel becomes undefined, which may affect the conversion output of the other channels.

(3) Conflict

<1> Conflict between writing to A/D conversion result register 0 (ADCR0) at the end of conversion and reading from ADCR0 using instruction

Reading from ADCR0 takes precedence. After reading, the new conversion result is written to ADCR0.

<2> Conflict between writing to ADCR0 at the end of conversion and writing to A/D converter mode register 0 (ADM0) or analog input channel specification register 0 (ADS0)

Writing to ADM0 or ADS0 takes precedence. ADCR0 is not written to. No A/D conversion end interrupt request signal (INTAD0) is generated.

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

The first A/D conversion value immediately after A/D conversion has been started is undefined. Poll the A/D conversion end interrupt request (INTAD0) and drop the first conversion result.

(5) Timing of undefined A/D conversion result

The A/D conversion value may become undefined if the timing of the completion of A/D conversion and that to stop the A/D conversion operation conflict. Therefore, read the A/D conversion result while the A/D conversion operation is in progress. To read the A/D conversion result after the A/D conversion operation has been stopped, stop the A/D conversion operation before the next conversion operation is completed. Figures 11-8 and 11-9 show the timing at which the conversion result is read.

Figure 11-8. Conversion Result Read Timing (If Conversion Result Is Undefined)

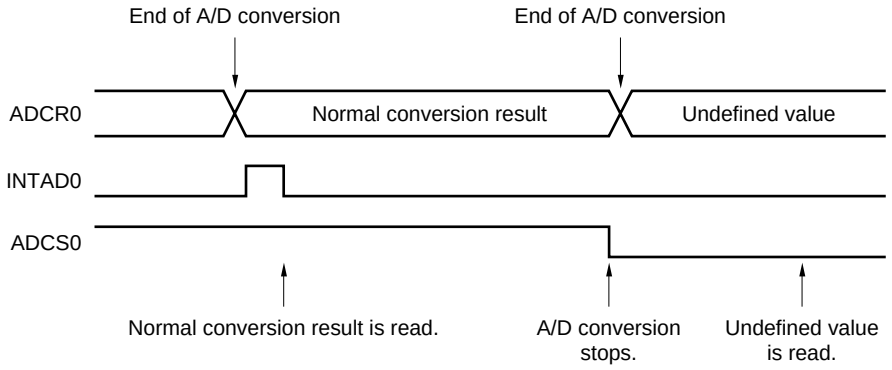
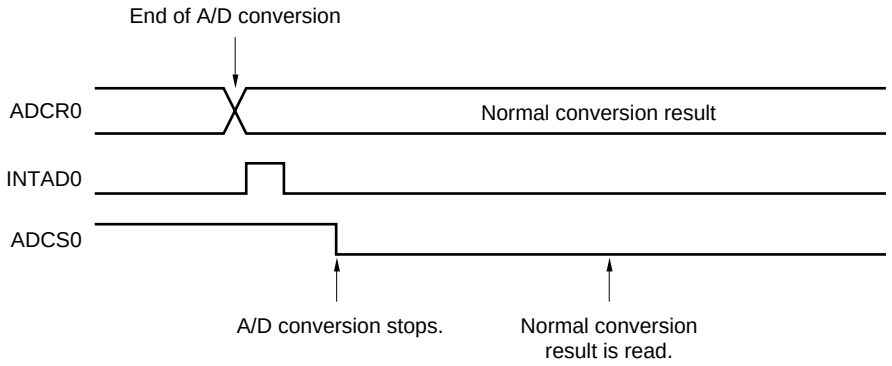


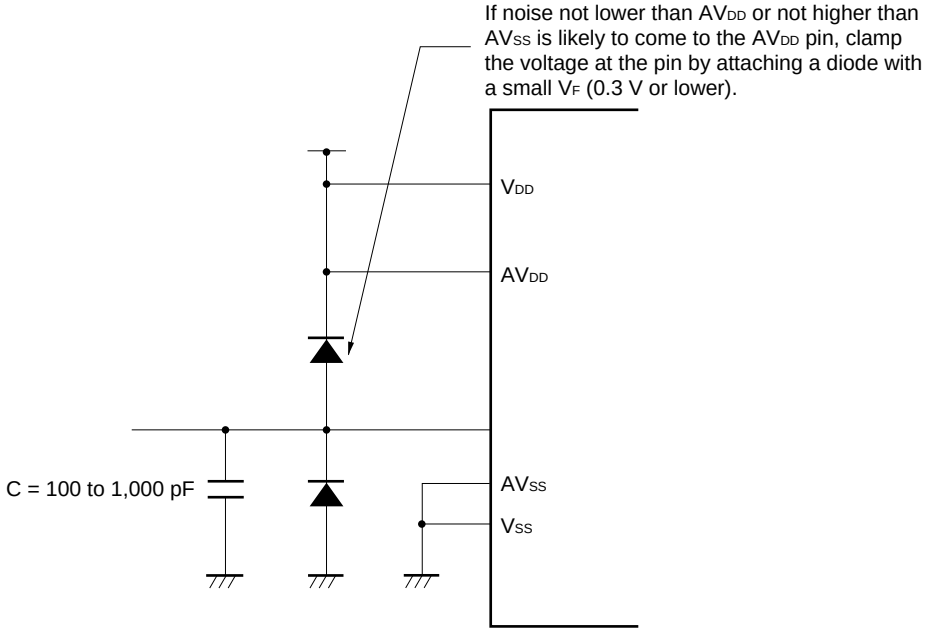
Figure 11-9. Conversion Result Read Timing (If Conversion Result Is Normal)



(6) Noise prevention

To maintain a resolution of 10 bits, watch for noise to the AV_{DD} and ANI0 to ANI5 pins. The higher the output impedance of the analog input source, the larger the effect by noise. To reduce noise, attach an external capacitor to the relevant pins as shown in Figure 11-10.

Figure 11-10. Analog Input Pin Treatment



(7) ANI0 to ANI5

The analog input pins (ANI0 to ANI5) are alternate-function pins. They are also used as port pins (P60 to P65).

If any of ANI0 to ANI5 has been selected for A/D conversion, do not execute input instructions for the ports; otherwise the conversion resolution may be reduced.

If a digital pulse is applied to a pin adjacent to the analog input pins during A/D conversion, coupling noise may occur that prevents an A/D conversion result from being obtained as expected. Avoid applying a digital pulse to pins adjacent to the analog input pins during A/D conversion.

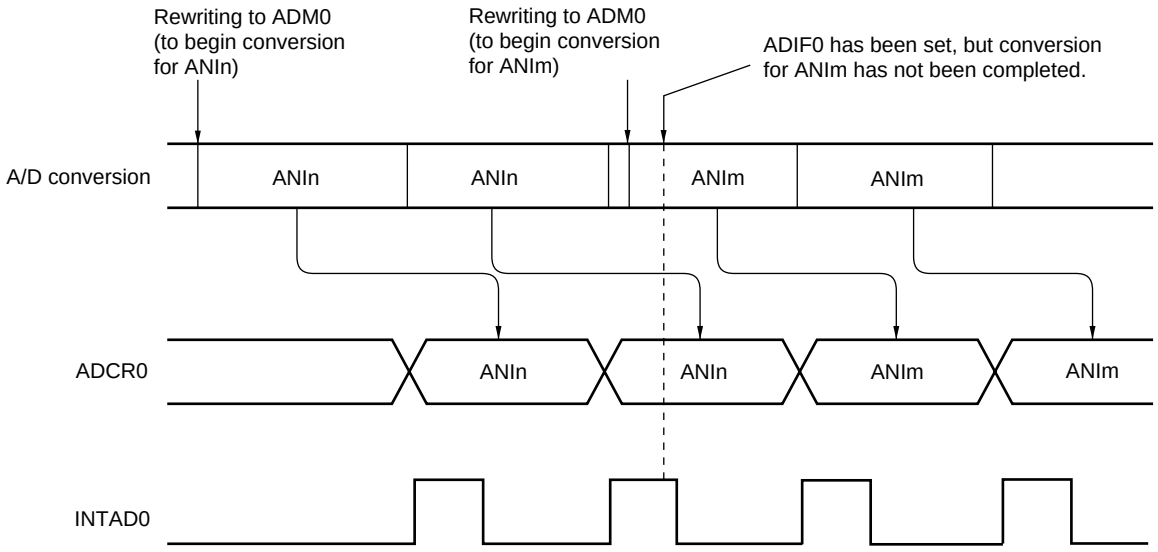
(8) Interrupt request flag (ADIF0)

Changing the contents of A/D converter mode register 0 (ADM0) does not clear the interrupt request flag (ADIF0).

If the analog input pins are changed during A/D conversion, therefore, the A/D conversion result and the conversion end interrupt request flag may reflect the previous analog input immediately before writing to ADM0 occurs. In this case, ADIF0 may already be set if it is read-accessed immediately after ADM0 is write-accessed, even when A/D conversion has not been completed for the new analog input.

In addition, when A/D conversion is restarted, ADIF0 must be cleared beforehand.

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



Remarks 1. n = 0 to 5

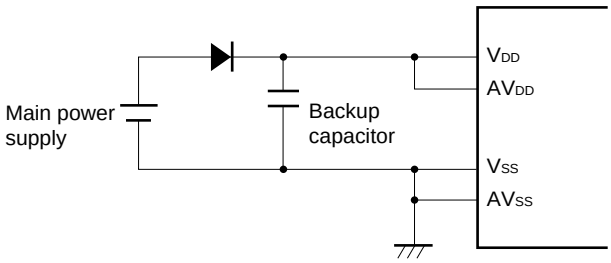
2. m = 0 to 5

(9) AV_{DD} pin

The AV_{DD} pin is used to supply power to the analog circuit. It is also used to supply power to the ANI0 to ANI5 input circuit.

If your application is designed to be changed to backup power, the AV_{DD} pin must be supplied with the same voltage level as the V_{DD} pin, as shown in Figure 11-12.

Figure 11-12. AV_{DD} Pin Handling



(10) AV_{DD} pin input impedance

A series resistor string of several ten of kΩ is connected between the AV_{DD} and AV_{SS} pins. Consequently, if the output impedance of the reference voltage supply is high, the reference voltage supply will form a parallel connection with the series resistor string, creating a large reference voltage differential.

[MEMO]

CHAPTER 12 SERIAL INTERFACE 20

12.1 Serial Interface 20 Functions

Serial interface 20 has the following three modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode
- 3-wire serial I/O mode

(1) Operation stop mode

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

(2) Asynchronous serial interface (UART) mode

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

Serial interface 20 contains an UART-dedicated baud rate generator, enabling communication over a wide range of baud rates. It is also possible to define baud rates by dividing the frequency of the clock input to the ASCK20 pin.

(3) 3-wire serial I/O mode (switchable between MSB-first and LSB-first transmission)

This mode is used to transmit 8-bit data, using three lines: a serial clock ($\overline{\text{SCK20}}$) line and two serial data lines (SI20 and SO20).

As it supports simultaneous transmission and reception, 3-wire serial I/O mode requires less processing time for data transmission than asynchronous serial interface mode.

Because, in 3-wire serial I/O mode, it is possible to select whether 8-bit data transmission begins with the MSB or LSB, serial interface 20 can be connected to any device regardless of whether that device is designed for MSB-first or LSB-first transmission.

3-wire serial I/O mode is useful for connecting peripheral I/O circuits and display controllers having conventional synchronous serial interfaces, such as those of the 75XL, 78K, and 17K Series devices.

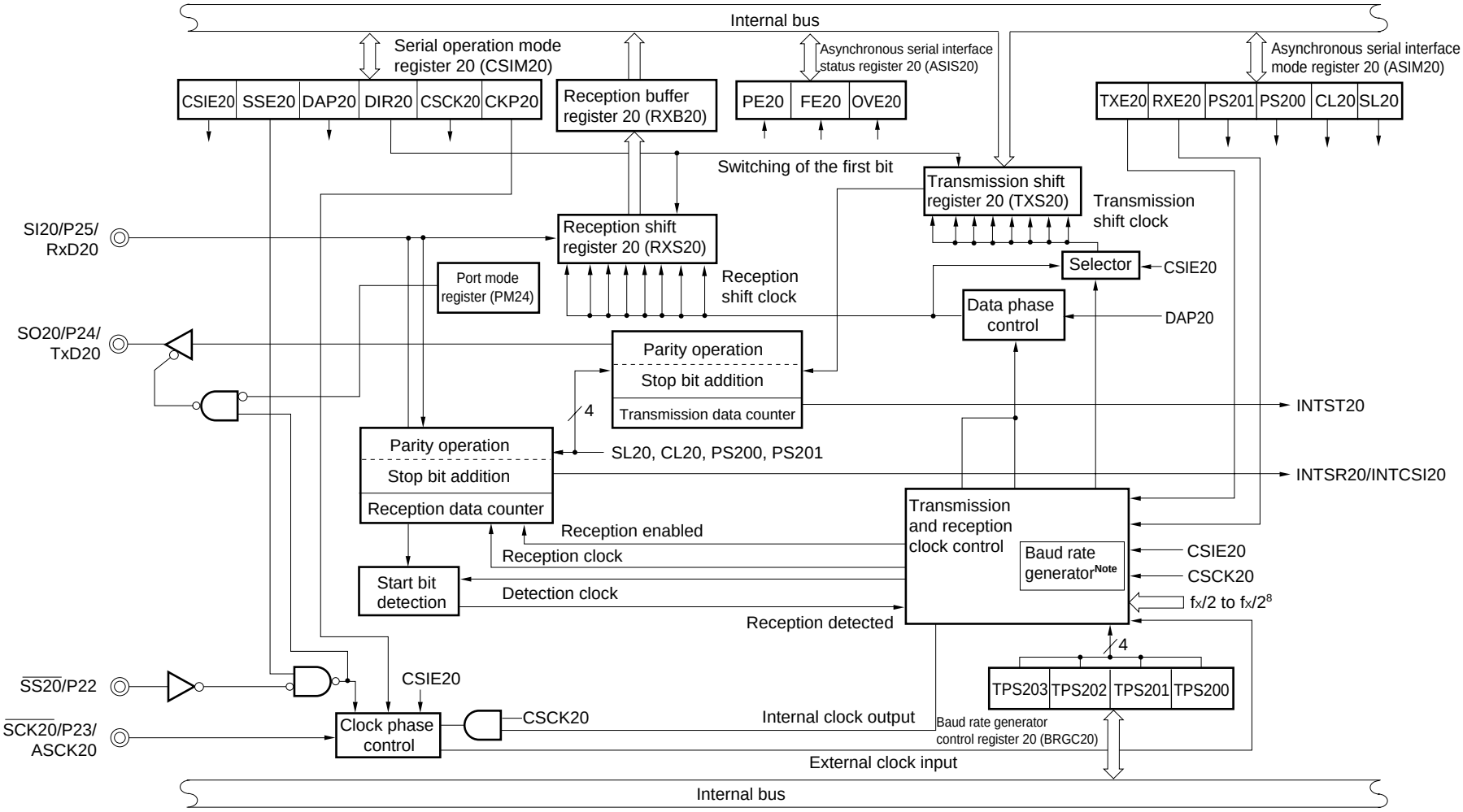
12.2 Serial Interface 20 Configuration

Serial interface 20 includes the following hardware.

Table 12-1. Configuration of Serial Interface 20

Item	Configuration
Registers	Transmission shift register 20 (TXS20) Reception shift register 20 (RXS20) Reception buffer register 20 (RXB20)
Control registers	Serial operation mode register 20 (CSIM20) Asynchronous serial interface mode register 20 (ASIM20) Asynchronous serial interface status register 20 (ASIS20) Baud rate generator control register 20 (BRGC20)

Figure 12-1. Block Diagram of Serial Interface 20



Note See **Figure 12-2** for the configuration of the baud rate generator.

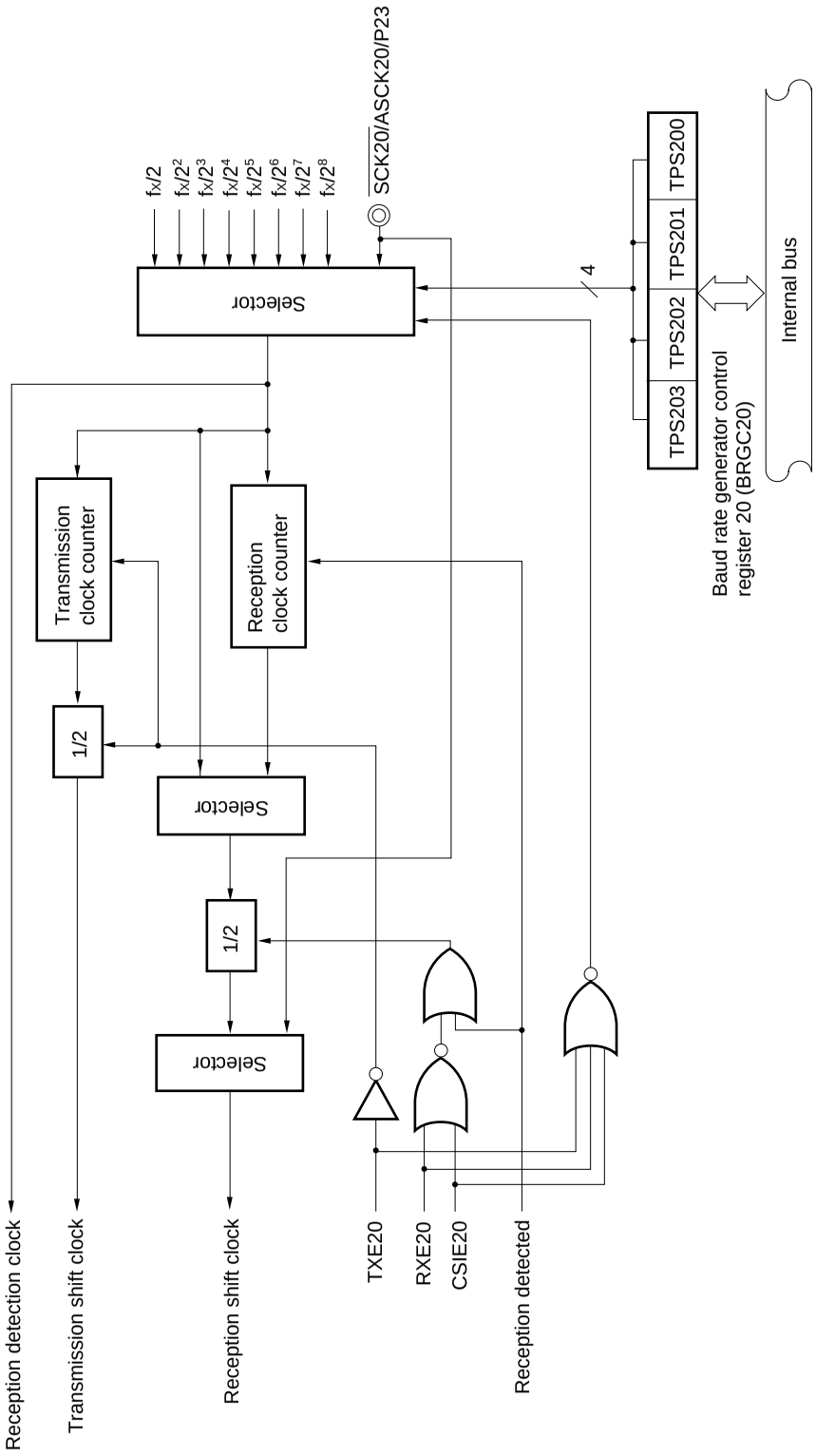


Figure 12-2. Block Diagram of Baud Rate Generator 20

(1) Transmission shift register 20 (TXS20)

TXS20 is a register in which transmission data is prepared. The transmission data is output from TXS20 bit-serially.

When the data length is seven bits, bits 0 to 6 of the data in TXS20 will be transmission data. Writing data to TXS20 triggers transmission.

TXS20 can be written with an 8-bit memory manipulation instruction, but cannot be read.

$\overline{\text{RESET}}$ input sets TXS20 to FFH.

Caution Do not write to TXS20 during transmission.

TXS20 and reception buffer register 20 (RXB20) are mapped at the same address, such that any attempt to read from TXS20 results in a value being read from RXB20.

(2) Reception shift register 20 (RXS20)

RXS20 is a register in which serial data, received at the RxD20 pin, is converted to parallel data. Once one entire byte has been received, RXS20 feeds the reception data to reception buffer register 20 (RXB20).

RXS20 cannot be manipulated directly by a program.

(3) Reception buffer register 20 (RXB20)

RXB20 holds a reception data. A new reception data is transferred from reception shift register 20 (RXS20) every 1-byte data reception.

When the data length is seven bits, the reception data is sent to bits 0 to 6 of RXB20, in which the MSB is always fixed to 0.

RXB20 can be read with an 8-bit memory manipulation instruction, but cannot be written.

$\overline{\text{RESET}}$ input makes RXB20 undefined.

Caution RXB20 and transmission shift register 20 (TXS20) are mapped at the same address, such that any attempt to write to RXB20 results in a value being written to TXS20.

(4) Transmission controller

The transmission controller controls transmission. For example, it adds start, parity, and stop bits to the data in transmission shift register 20 (TXS20), according to the setting of asynchronous serial interface mode register 20 (ASIM20).

(5) Reception controller

The reception controller controls reception according to the setting of asynchronous serial interface mode register 20 (ASIM20). It also checks for errors, such as parity errors, during reception. If an error is detected, asynchronous serial interface status register 20 (ASIS20) is set according to the status of the error.

12.3 Serial Interface 20 Control Registers

Serial interface 20 is controlled by the following registers.

- Serial operation mode register 20 (CSIM20)
- Asynchronous serial interface mode register 20 (ASIM20)
- Asynchronous serial interface status register 20 (ASIS20)
- Baud rate generator control register 20 (BRGC20)

(1) Serial operation mode register 20 (CSIM20)

CSIM20 is used to make the settings related to 3-wire serial I/O mode.

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CSIM20 to 00H.

Figure 12-3. Format of Serial Operation Mode Register 20

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	SSE20	0	0	DAP20	DIR20	CSCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control
0	Operation disabled
1	Operation enabled

SSE20	$\overline{\text{SS20}}$ pin selection	Function of $\overline{\text{SS20}}$/P22 pin	Communication status
0	Not used	Port function	Communication enabled
1	Used	0	Communication enabled
		1	Communication disabled

DAP20	3-wire serial I/O mode data phase selection
0	Outputs at the falling edge of $\overline{\text{SCK20}}$
1	Outputs at the rising edge of $\overline{\text{SCK20}}$

DIR20	First-bit specification
0	MSB
1	LSB

CSCK20	3-wire serial I/O mode clock selection
0	External clock input to the $\overline{\text{SCK20}}$ pin
1	Output of the dedicated baud rate generator

CKP20	3-wire serial I/O mode clock phase selection
0	Clock is low active, and $\overline{\text{SCK20}}$ is at high level in the idle state
1	Clock is high active, and $\overline{\text{SCK20}}$ is at low level in the idle state

Cautions 1. Bits 4 and 5 must be set to 0.

2. CSIM20 must be cleared to 00H if UART mode is selected.

(2) Asynchronous serial interface mode register 20 (ASIM20)

ASIM20 is used to make the settings related to asynchronous serial interface mode.

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIM20 to 00H.

Figure 12-4. Format of Asynchronous Serial Interface Mode Register 20

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stop
1	Transmit operation enable

RXE20	Receive operation control
0	Receive operation stop
1	Receive operation enable

PS201	PS200	Parity bit specification
0	0	No parity
0	1	Always add 0 parity at transmission. Parity check is not performed at reception (No parity error is generated).
1	0	Odd parity
1	1	Even parity

CL20	Transmit data character length specification
0	7 bits
1	8 bits

SL20	Transmit data stop bit length
0	1 bit
1	2 bits

Cautions 1. Bits 0 and 1 must be set to 0.

2. If 3-wire serial I/O mode is selected, ASIM20 must be set to 00H.

3. Switch operating modes after halting the serial transmit/receive operation.

Table 12-2. Serial Interface 20 Operating Mode Settings

(1) Operation stop mode

ASIM20		CSIM20			PM25	P25	PM24	P24	PM23	P23	First Bit	Shift Clock	P25/SI20/ RxD20 Pin Function	P24/SO20/ TxD20 Pin Function	P23/$\overline{\text{SCK20}}$/ ASCK20 Pin Function
TXE20	RXE20	CSIE20	DIR20	CSCK20											
0	0	0	×	×	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	–	–	P25	P24	P23
Other than above											Setting prohibited				

(2) 3-wire serial I/O mode

ASIM20		CSIM20			PM25	P25	PM24	P24	PM23	P23	First Bit	Shift Clock	P25/SI20/ RxD20 Pin Function	P24/SO20/ TxD20 Pin Function	P23/$\overline{\text{SCK20}}$/ ASCK20 Pin Function
TXE20	RXE20	CSIE20	DIR20	CSCK20											
0	0	1	0	0	×Note 2	×Note 2	0	1	1	×	MSB	External clock	SI20Note 2	SO20 (CMOS output)	$\overline{\text{SCK20}}$ input
				1					0	1		Internal clock			$\overline{\text{SCK20}}$ output
		1	1	0					1	×	LSB	External clock			$\overline{\text{SCK20}}$ input
				1					0	1		Internal clock			$\overline{\text{SCK20}}$ output
Other than above											Setting prohibited				

(3) Asynchronous serial interface mode

ASIM20		CSIM20			PM25	P25	PM24	P24	PM23	P23	First Bit	Shift Clock	P25/SI20/ RxD20 Pin Function	P24/SO20/ TxD20 Pin Function	P23/$\overline{\text{SCK20}}$/ ASCK20 Pin Function
TXE20	RXE20	CSIE20	DIR20	CSCK20											
1	0	0	0	0	×Note 1	×Note 1	0	1	1	×	LSB	External clock	P22	TxD20 (CMOS output)	ASCK20 input
									×Note 1	×Note 1		Internal clock			P23
0	1	0	0	0	1	×	×Note 1	×Note 1	1	×		External clock	RxD20	P24	ASCK20 input
									×Note 1	×Note 1		Internal clock			P23
1	1	0	0	0	1	×	0	1	1	×		External clock		TxD20 (CMOS output)	ASCK20 input
									×Note 1	×Note 1		Internal clock			P23
Other than above											Setting prohibited				

Notes 1. These pins can be used for port functions.

2. When only transmission is used, this pin can be used as P25 (CMOS I/O).

Remark ×: don't care.

(3) Asynchronous serial interface status register 20 (ASIS20)

ASIS20 indicates the type of a reception error, if it occurs while asynchronous serial interface mode is set.

ASIS20 is set with a 1-bit or 8-bit memory manipulation instruction.

The contents of ASIS20 are undefined in 3-wire serial I/O mode.

$\overline{\text{RESET}}$ input sets ASIS20 to 00H.

Figure 12-5. Format of Asynchronous Serial Interface Status Register 20

Symbol	7	6	5	4	3	<2>	<1>	<0>	Address	After reset	R/W
ASIS20	0	0	0	0	0	PE20	FE20	OVE20	FF71H	00H	R

PE20	Parity error flag
0	No parity error has occurred.
1	A parity error has occurred (when the parity of transmit data does not match).

FE20	Flaming error flag
0	No framing error has occurred.
1	A framing error has occurred (when stop bit is not detected).Note 1

OVE20	Overrun error flag
0	No overrun error has occurred.
1	An overrun error has occurred.Note 2 (when the next receive operation is completed before the data is read from reception buffer register 20)

Notes 1. Even when the stop bit length is set to 2 bits by setting bit 2 (SL20) of asynchronous serial interface mode register 20 (ASIM20), the stop bit detection at reception is performed with 1 bit.

2. Be sure to read reception buffer register 20 (RXB20) when an overrun error occurs. If not, every time the data is received an overrun error will occur.

(4) Baud rate generator control register 20 (BRGC20)

BRGC20 is used to specify the serial clock for serial interface 20.

BRGC20 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGC20 to 00H.

Figure 12-6. Format of Baud Rate Generator Control Register 20

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC20	TPS203	TPS202	TPS201	TPS200	0	0	0	0	FF73H	00H	R/W

TPS203	TPS202	TPS201	TPS200	3-bit counter source clock selection	n
0	0	0	0	$f_X/2$ (2.5 MHz)	1
0	0	0	1	$f_X/2^2$ (1.25 MHz)	2
0	0	1	0	$f_X/2^3$ (625 kHz)	3
0	0	1	1	$f_X/2^4$ (313 kHz)	4
0	1	0	0	$f_X/2^5$ (156 kHz)	5
0	1	0	1	$f_X/2^6$ (78.1 kHz)	6
0	1	1	0	$f_X/2^7$ (39.1 kHz)	7
0	1	1	1	$f_X/2^8$ (19.5 kHz)	8
1	0	0	0	External clock input to the ASCK20 pin[Note]	–
Other than above				Setting prohibited	

Note An external clock can be used only in UART mode.

Cautions 1. When writing to BRGC00 during a communication operation, the output of the baud rate generator is disrupted and communications cannot be performed normally. Be sure not to write to BRGC00 during a communication operation.

2. Be sure not to select n = 1 during operation at f_X = 5.0 MHz because the resulting baud rate exceeds the rated range.

3. When the external input clock is selected, set port mode register 2 (PM2) to input mode.

Remarks 1. f_X: Main system clock oscillation frequency

2. n: Values determined by the settings of TPS200 to TPS203 ($1 \leq n \leq 8$)

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

The baud rate transmit/receive clock to be generated is either a signal scaled from the system clock, or a signal scaled from the clock input to the ASCK20 pin.

(a) Generation of baud rate transmit/receive clock form system clock

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

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1} \times 8} \text{ [Hz]}$$

f_X: Main system clock oscillation frequency

n: Values in Figure 12-6, determined by the values of TPS200 to TPS203 ($2 \leq n \leq 8$)

Table 12-3. Example of Relationships Between System Clock and Baud Rate

Baud Rate (bps)	n	BRGC20 Set Value	Error (%)	
			f_X = 5.0 MHz	f_X = 4.9152 MHz
1,200	8	70H	1.73	0
2,400	7	60H		
4,800	6	50H		
9,600	5	40H		
19,200	4	30H		
38,400	3	20H		
76,800	2	10H		

Caution Do not select n = 1 during operation at f_X = 5.0 MHz because the resulting baud rate exceeds the rated range.

(b) Generation of baud rate transmit/receive clock from external clock input to ASCK20 pin

The transmit/receive clock is generated by scaling the clock input from the ASCK20 pin. The baud rate of a clock generated from the clock input to the ASCK20 pin is estimated by using the following expression.

$$[\text{Baud rate}] = \frac{f_{ASCK}}{16} \text{ [Hz]}$$

f_{ASCK}: Frequency of clock input to the ASCK20 pin

Table 12-4. Relationship Between ASCK20 Pin Input Frequency and Baud Rate (When BRGC20 Is Set to 80H)

Baud Rate (bps)	ASCK20 Pin Input Frequency (kHz)
75	1.2
150	2.4
300	4.8
600	9.6
1,200	19.2
2,400	38.4
4,800	76.8
9,600	153.6
19,200	307.2
31,250	500.0
38,400	614.4

12.4 Serial Interface 20 Operation

Serial interface 20 provides the following three modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode
- 3-wire serial I/O mode

12.4.1 Operation stop mode

In operation stop mode, serial transfer is not executed, thereby reducing the power consumption. The P23/$\overline{\text{SCK20}}$/ASCK20, P24/SO20/TxD20, and P25/SI20/RxD20 pins can be used as normal I/O ports.

(1) Register setting

Operation stop mode is set by serial operation mode register 20 (CSIM20) and asynchronous serial interface mode register 20 (ASIM20).

(a) Serial operation mode register 20 (CSIM20)

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears CSIM20 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	SSE20	0	0	DAP20	DIR20	CSCK20	CKP20	FF72H	00H	R/W

CSIE20	Operation control in 3-wire serial I/O mode
0	Operation disabled
1	Operation enabled

Caution Bits 4 and 5 must be set to 0.

(b) Asynchronous serial interface mode register 20 (ASIM20)

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIM20 to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stopped
1	Transmit operation enabled

RXE20	Receive operation control
0	Receive operation stopped
1	Receive operation enabled

Caution Bits 0 and 1 must be set to 0.

12.4.2 Asynchronous serial interface (UART) mode

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

This device incorporates UART-dedicated baud rate generator that enables communications at the desired baud rate. In addition, the baud rate can also be defined by dividing the clock input to the ASCK20 pin.

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

(1) Register setting

UART mode is set by serial operation mode register 20 (CSIM20), asynchronous serial interface mode register 20 (ASIM20), asynchronous serial interface status register 20 (ASIS20), and baud rate generator control register 20 (BRGC20).

(a) Serial operation mode register 20 (CSIM20)

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CSIM20 to 00H.

Set CSIM20 to 00H when UART mode is selected.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	SSE20	0	0	DAP20	DIR20	CSCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control
0	Operation disabled
1	Operation enabled

SSE20	$\overline{\text{SS20}}$ pin selection	Function of $\overline{\text{SS20}}$/P22 pin	Communication status
0	Not used	Port function	Communication enabled
1	Used	0	Communication enabled
		1	Communication disabled

DAP20	3-wire serial I/O mode data phase selection
0	Outputs at the falling edge of $\overline{\text{SCK20}}$
1	Outputs at the rising edge of $\overline{\text{SCK20}}$

DIR20	First-bit specification
0	MSB
1	LSB

CSCK20	3-wire serial I/O mode clock selection
0	External clock input to the $\overline{\text{SCK20}}$ pin
1	Output of the dedicated baud rate generator

CKP20	3-wire serial I/O mode clock phase selection
0	Clock is low active, and $\overline{\text{SCK20}}$ is high level in the idle state
1	Clock is high active, and $\overline{\text{SCK20}}$ is low level in the idle state

Caution Bits 4 and 5 must be set to 0.

(b) Asynchronous serial interface mode register 20 (ASIM20)

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIM20 to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stopped
1	Transmit operation enabled

RXE20	Receive operation control
0	Receive operation stopped
1	Receive operation enabled

PS201	PS200	Parity bit specification
0	0	No parity
0	1	Always add 0 parity at transmission. Parity check is not performed at reception (No parity error is generated).
1	0	Odd parity
1	1	Even parity

CL20	Character length specification
0	7 bits
1	8 bits

SL20	Transmit data stop bit length specification
0	1 bit
1	2 bits

Cautions 1. Bits 0 and 1 must be set to 0.

2. Switch operating modes after halting the serial transmit/receive operation.

(c) Asynchronous serial interface status register 20 (ASIS20)

ASIS20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIS20 to 00H.

Symbol	7	6	5	4	3	<2>	<1>	<0>	Address	After reset	R/W
ASIS20	0	0	0	0	0	PE20	FE20	OVE20	FF71H	00H	R

PE20	Parity error flag
0	No parity error has occured
1	A parity error has occured (when the parity of transmit data does not match)

FE20	Framing error flag
0	No framing error has occured
1	A framing error has occured (when stop bit is not detected)Note 1

OVE20	Overrun error flag
0	No overrun error has occured
1	An overrun error has occuredNote 2 (when the next receive operation is completed before data is read from reception buffer register 20)

Notes 1. Even when the stop bit length is set to 2 bits by setting bit 2 (SL20) of asynchronous serial interface mode register 20 (ASIM20), the stop bit detection at reception is performed with 1 bit.

2. Be sure to read reception buffer register 20 (RXB20) when an overrun error occurs. If not, every time the data is received an overrun error will occur.

(d) Baud rate generator control register 20 (BRGC20)

BRGC20 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGC20 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC20	TPS203	TPS202	TPS201	TPS200	0	0	0	0	FF73H	00H	R/W

TPS203	TPS202	TPS201	TPS200	3-bit counter source clock selection	n
0	0	0	0	$f_X/2$ (2.5 MHz)	1
0	0	0	1	$f_X/2^2$ (1.25 MHz)	2
0	0	1	0	$f_X/2^3$ (625 kHz)	3
0	0	1	1	$f_X/2^4$ (313 kHz)	4
0	1	0	0	$f_X/2^5$ (156 kHz)	5
0	1	0	1	$f_X/2^6$ (78.1 kHz)	6
0	1	1	0	$f_X/2^7$ (39.1 kHz)	7
0	1	1	1	$f_X/2^8$ (19.5 kHz)	8
1	0	0	0	External clock input to ASCK20 pin[Note]	–
Other than above				Setting prohibited	

Note Can only be used in the UART mode.

Cautions 1. When writing to BRGC20 during a communication operation, the output of the baud rate generator is disrupted and communications cannot be performed normally. Be sure not to write to BRGC20 during a communication operation.

2. Be sure not to select n = 1 during operation at f_X = 5.0 MHz because the resulting baud rate exceeds the rated range.

3. When the external input clock is selected, set port mode register 2 (PM2) to input mode.

Remarks 1. f_X: Main system clock oscillation frequency

2. n: Values determined by the settings of TPS200 to TPS203 ($1 \leq n \leq 8$)

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

The baud rate transmit/receive clock to be generated is either a signal scaled from the system clock, or a signal scaled from the clock input to the ASCK20 pin.

(i) Generation of baud rate transmit/receive clock from system clock

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

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1} \times 8} \text{ [Hz]}$$

f_X: Main system clock oscillation frequency

n: Values in the above table determined by the settings of TPS200 to TPS203 ($2 \leq n \leq 8$)

Table 12-5. Example of Relationships Between System Clock and Baud Rate

Baud Rate (bps)	n	BRGC20 Set Value	Error (%)	
			f_X = 5.0 MHz	f_X = 4.9152 MHz
1,200	8	70H	1.73	0
2,400	7	60H		
4,800	6	50H		
9,600	5	40H		
19,200	4	30H		
38,400	3	20H		
76,800	2	10H		

Caution Do not select n = 1 during operation at f_X = 5.0 MHz because the resulting baud rate exceeds the rated range.

(ii) Generation of baud rate transmit/receive clock from external clock input to ASCK20 pin

The transmit/receive clock is generated by scaling the clock input from the ASCK20 pin. The baud rate of a clock generated from the clock input to the ASCK20 pin is estimated by using the following expression.

$$[\text{Baud rate}] = \frac{f_{ASCK}}{16} \text{ [Hz]}$$

f_{ASCK}: Frequency of clock input to ASCK20 pin

Table 12-6. Relationship Between ASCK20 Pin Input Frequency and Baud Rate (When BRGC20 Is Set to 80H)

Baud Rate (bps)	ASCK20 Pin Input Frequency (kHz)
75	1.2
150	2.4
300	4.8
600	9.6
1,200	19.2
2,400	38.4
4,800	76.8
9,600	153.6
19,200	307.2
31,250	500.0
38,400	614.4

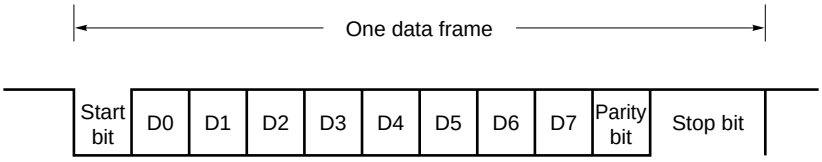
(2) Communication operation

(a) Data format

The transmit/receive data format is as shown in Figure 12-7. One data frame consists of a start bit, character bits, parity bit, and stop bit(s).

The specification of character bit length in one data frame, parity selection, and specification of stop bit length is carried out with asynchronous serial interface mode register 20 (ASIM20).

Figure 12-7. Format of Asynchronous Serial Interface Transmit/Receive Data



- Start bits ..... 1 bit
- Character bits ..... 7 bits/8 bits
- Parity bits..... Even parity/odd parity/0 parity/no parity
- Stop bits..... 1 bit/2 bits

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

The serial transfer rate is selected by ASIM20 and baud rate generator control register 20 (BRGC20).

If a serial data receive error occurs, the receive error contents can be determined by reading the status of asynchronous serial interface status register 20 (ASIS20).

(b) Parity types and operation

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

(i) Even parity

- **At transmission**

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

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

- **At reception**

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

(ii) Odd parity

- **At transmission**

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

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

- **At reception**

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

(iii) 0 parity

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

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

(iv) No parity

A parity bit is not added to the transmit data. At reception, data is received assuming that there is no parity bit. Since there is no parity bit, a parity error does not occur.

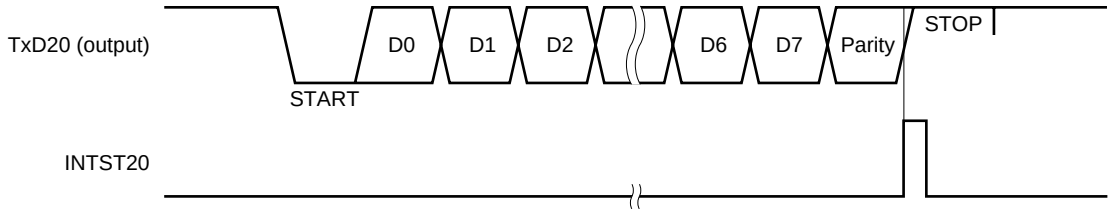
(c) Transmission

A transmit operation is started by writing transmit data to transmission shift register 20 (TXS20). The start bit, parity bit, and stop bit(s) are added automatically.

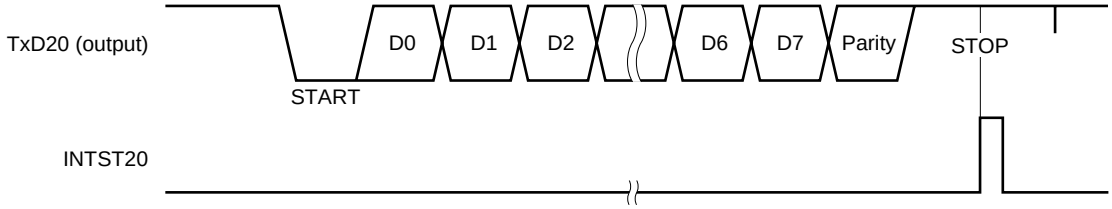
When the transmit operation starts, the data in TXS20 is shifted out, and when TXS20 is empty, a transmission completion interrupt (INTST20) is generated.

Figure 12-8. Asynchronous Serial Interface Transmission Completion Interrupt Timing

(a) Stop bit length: 1



(b) Stop bit length: 2



Caution Do not rewrite asynchronous serial interface mode register 20 (ASIM20) during a transmit operation. If the ASIM20 register is rewritten during transmission, subsequent transmission may not be able to be performed (the normal state is restored by $\overline{\text{RESET}}$ input).

It is possible to determine whether transmission is in progress by software by using a transmission completion interrupt (INTST20) or the interrupt request flag (STIF20) set by INTST20.

(d) Reception

When bit 6 (RXE20) of asynchronous serial interface mode register 20 (ASIM20) is set (1), a receive operation is enabled and sampling of the RxD20 pin input is performed.

RxD20 pin input sampling is performed using the serial clock specified by ASIM20.

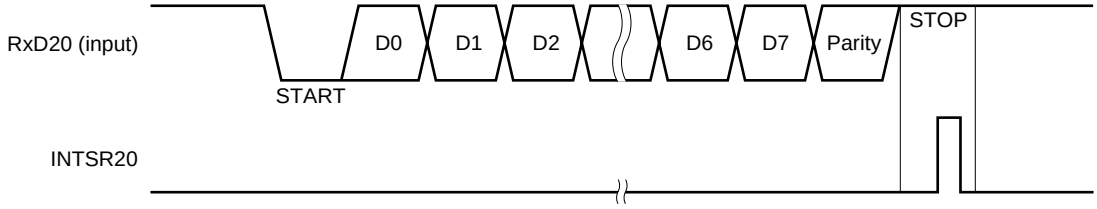
When the RxD20 pin input becomes low, the 3-bit counter starts counting, and when half the time determined by the specified baud rate has passed, the data sampling start timing signal is output. If the RxD20 pin input sampled again as a result of this start timing signal is low, it is identified as a start bit, the 3-bit counter is initialized and starts counting, and data sampling is performed. When character data, a parity bit, and one stop bit are detected after the start bit, reception of one frame of data ends.

When one frame of data has been received, the receive data in the shift register is transferred to reception buffer register 20 (RXB20), and a reception completion interrupt (INTSR20) is generated.

If an error occurs, the receive data in which the error occurred is still transferred to RXB20, and INTSR20 is generated.

If the RXE20 bit is reset (0) during the receive operation, the receive operation is stopped immediately. In this case, the contents of RXB20 and asynchronous serial interface status register 20 (ASIS20) are not changed, and INTSR20 is not generated.

Figure 12-9. Asynchronous Serial Interface Reception Completion Interrupt Timing



Caution Be sure to read reception buffer register 20 (RXB20) even if a receive error occurs. If RXB20 is not read, an overrun error will occur when the next data is received, and the receive error state will continue indefinitely.

(e) Receive errors

The following three errors may occur during a receive operation: a parity error, framing error, and overrun error. After data reception, an error flag is set in asynchronous serial interface status register 20 (ASIS20). Receive error causes are shown in Table 12-7.

It is possible to determine what kind of error occurred during reception by reading the contents of ASIS20 in the reception error interrupt servicing (see **Figures 12-9** and **12-10**).

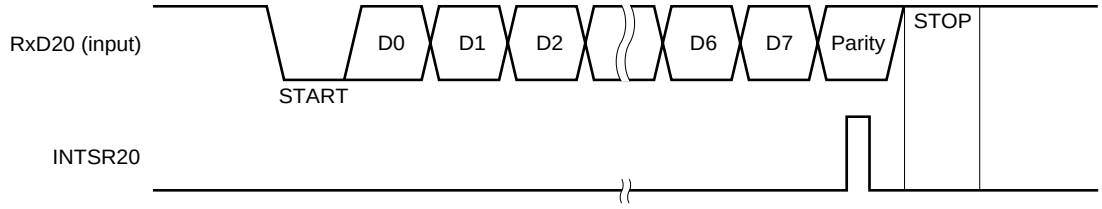
The contents of ASIS20 are reset (0) by reading reception buffer register 20 (RXB20) or receiving the next data (if there is an error in the next data, the corresponding error flag is set).

Table 12-7. Receive Error Causes

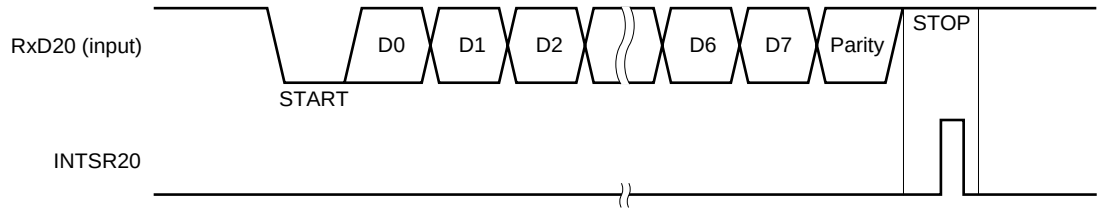
Receive Errors	Cause
Parity error	Transmission-time parity and reception data parity do not match
Framing error	Stop bit not detected
Overrun error	Reception of next data is completed before data is read from reception buffer register

Figure 12-10. Receive Error Timing

(a) Parity error occurrence



(b) Framing error or overrun error occurrence



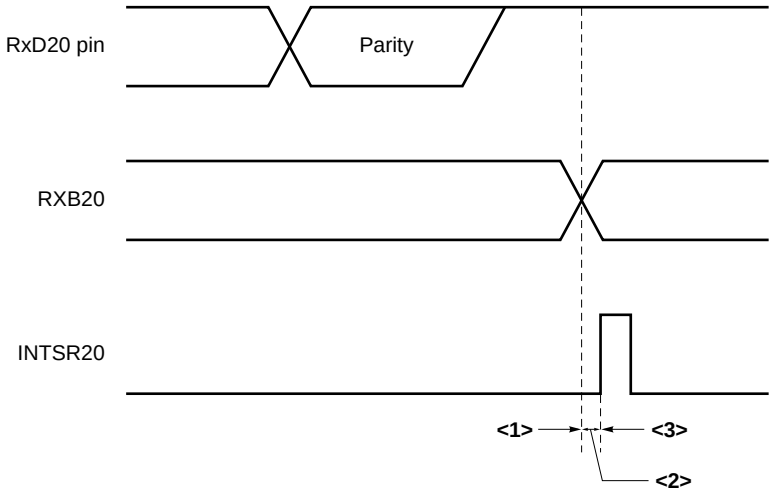
Cautions 1. The contents of the ASIS20 register are reset (0) by reading reception buffer register 20 (RXB20) or receiving the next data. To ascertain the error contents, read ASIS20 before reading RXB20.

2. Be sure to read reception buffer register 20 (RXB20) even if a receive error occurs. If RXB20 is not read, an overrun error will occur when the next data is received, and the receive error state will continue indefinitely.

(3) Cautions related to UART mode

(a) When bit 7 (TXE20) of asynchronous serial interface mode register 20 (ASIM20) is cleared during transmission, be sure to set transmission shift register 20 (TXS20) to FFH, then set TXE20 to 1 before executing the next transmission.

(b) When bit 6 (RXE20) of asynchronous serial interface mode register 20 (ASIM20) is cleared during reception, reception buffer register 20 (RXB20) and the receive completion interrupt (INTSR20) are as follows.



When RXE20 is set to 0 at a time indicated by **<1>**, RXB20 holds the previous data and INTSR20 is not generated.
When RXE20 is set to 0 at a time indicated by **<2>**, RXB20 renews the data and INTSR20 is not generated.
When RXE20 is set to 0 at a time indicated by **<3>**, RXB20 renews the data and INTSR20 is generated.

12.4.3 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, and 17K Series.

Communication is performed using three lines: a serial clock ($\overline{SCK20}$), serial output (SO20), and serial input (SI20).

(1) Register setting

3-wire serial I/O mode settings are performed using serial operation mode register 20 (CSIM20), asynchronous serial interface mode register 20 (ASIM20), and baud rate generator control register 20 (BRGC20).

(a) Serial operation mode register 20 (CSIM20)

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{RESET}$ input sets CSIM20 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	SSE20	0	0	DAP20	DIR20	CSCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control
0	Operation disabled
1	Operation enabled

SSE20	$\overline{SS20}$ pin selection	Function of $\overline{SS20}$/P22 pin	Communication status
0	Not used	Port function	Communication enabled
1	Used	0	Communication enabled
		1	Communication disabled

DAP20	3-wire serial I/O mode data phase selection
0	Outputs at the falling edge of $\overline{SCK20}$
1	Outputs at the rising edge of $\overline{SCK20}$

DIR20	First-bit specification
0	MSB
1	LSB

CSCK20	3-wire serial I/O mode clock selection
0	External clock input to the $\overline{SCK20}$ pin
1	Output of the dedicated baud rate generator

CKP20	3-wire serial I/O mode clock phase selection
0	Clock is low active, and $\overline{SCK20}$ is at high level in the idle state
1	Clock is high active, and $\overline{SCK20}$ is at low level in the idle state

Caution Bits 4 and 5 must be set to 0.

(b) Asynchronous serial interface mode register 20 (ASIM20)

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ASIM20 to 00H.

When 3-wire serial I/O mode is selected, ASIM20 must be set to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stopped
1	Transmit operation enabled

RXE20	Receive operation control
0	Receive operation stopped
1	Receive operation enabled

PS201	PS200	Parity bit specification
0	0	No parity
0	1	Always add 0 parity at transmission. Parity check is not performed at reception (No parity error occurs).
1	0	Odd parity
1	1	Even parity

CL20	Transmit data character length specification
0	7 bits
1	8 bits

SL20	Transmit data stop bit length specification
0	1 bit
1	2 bits

Cautions 1. Bits 0 and 1 must be set to 0.

2. Switch operating modes after halting the serial transmit/receive operation.

(c) Baud rate generator control register 20 (BRGC20)

BRGC20 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGC20 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC20	TPS203	TPS202	TPS201	TPS200	0	0	0	0	FF73H	00H	R/W

TPS203	TPS202	TPS201	TPS200	3-bit counter source clock selection	n
0	0	0	0	$f_X/2$ (2.5 MHz)	1
0	0	0	1	$f_X/2^2$ (1.25 MHz)	2
0	0	1	0	$f_X/2^3$ (625 kHz)	3
0	0	1	1	$f_X/2^4$ (313 kHz)	4
0	1	0	0	$f_X/2^5$ (156 kHz)	5
0	1	0	1	$f_X/2^6$ (78.1 kHz)	6
0	1	1	0	$f_X/2^7$ (39.1 kHz)	7
0	1	1	1	$f_X/2^8$ (19.5 kHz)	8
Other than above				Setting prohibited	

Cautions 1. When writing to BRGC20 during a communication operation, the baud rate generator output is disrupted and communications cannot be performed normally. Be sure not to write to BRGC20 during a communication operation.

2. Be sure not to select n = 1 during operation at f_X = 5.0 MHz because the resulting baud rate exceeds the rated range.

3. When the external input clock is selected, set port mode register 2 (PM2) to input mode.

Remarks 1. f_X: Main system clock oscillation frequency

2. n: Values determined by the settings of TPS200 to TPS203 ($1 \leq n \leq 8$)

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

If the internal clock is used as the serial clock for 3-wire serial I/O mode, set bits TPS200 to TPS203 to set the frequency of the serial clock. To obtain the frequency to be set, use the following expression. When an external clock is used, setting BRGC20 is not necessary.

$$\text{Serial clock frequency} = \frac{f_X}{2^{n+1}} \text{ [Hz]}$$

f_X: Main system clock oscillation frequency

n: Values in the above table determined by the settings of TPS200 to TPS203 ($1 \leq n \leq 8$)

(2) Communication operation

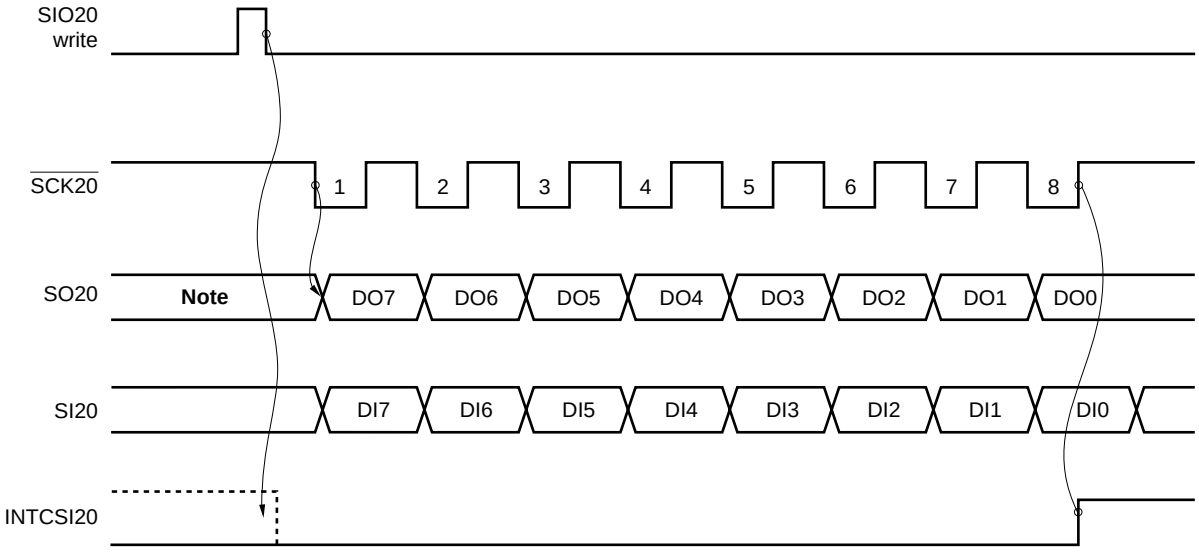
In 3-wire serial I/O mode, data transmission/reception is performed in 8-bit units. Data is transmitted/received bit by bit in synchronization with the serial clock.

Transmission shift register (TXS20/SIO20) and reception shift register (RXS20) shift operations are performed in synchronization with the fall of the serial clock ($\overline{SCK20}$). Then transmit data is held in the SO20 latch and output from the SO20 pin. Also, receive data input to the SI20 pin is latched in the reception buffer register (RXB20/SIO20) on the rise of $\overline{SCK20}$.

At the end of an 8-bit transfer, the operation of TXS20/SIO20 and RXS20 stops automatically, and the interrupt request signal (INTCSI20) is generated.

Figure 12-11. 3-Wire Serial I/O Mode Timing (1/7)

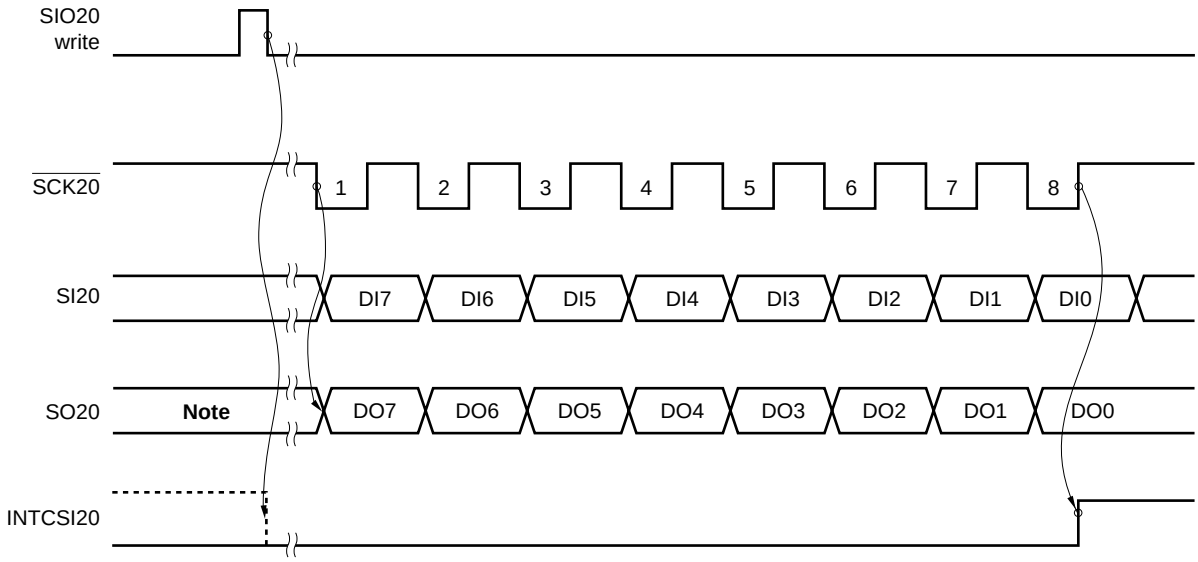
(i) Master operation timing (when DAP20 = 0, CKP20 = 0, SSE20 = 0)



Note The value of the last bit previously output is output.

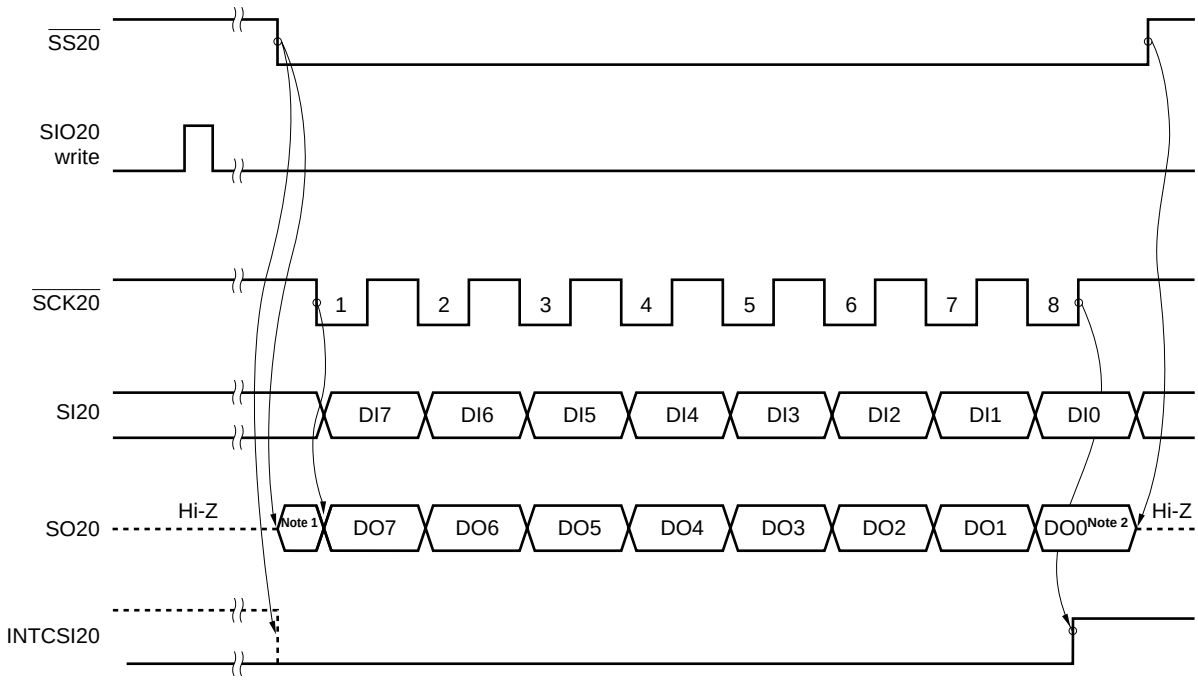
Figure 12-11. 3-Wire Serial I/O Mode Timing (2/7)

(ii) Slave operation timing (when DAP20 = 0, CKP20 = 0, SSE20 = 0)



Note The value of the last bit previously output is output.

(iii) Slave operation (when DAP20 = 0, CKP20 = 0, SSE20 = 1)

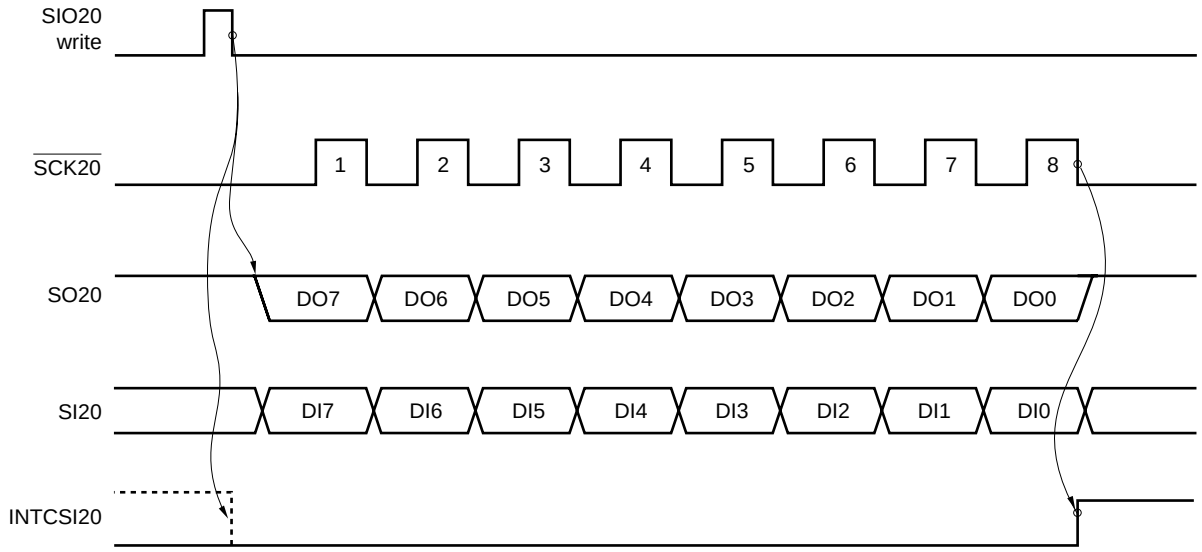


Notes 1. The value of the last bit previously output is output.

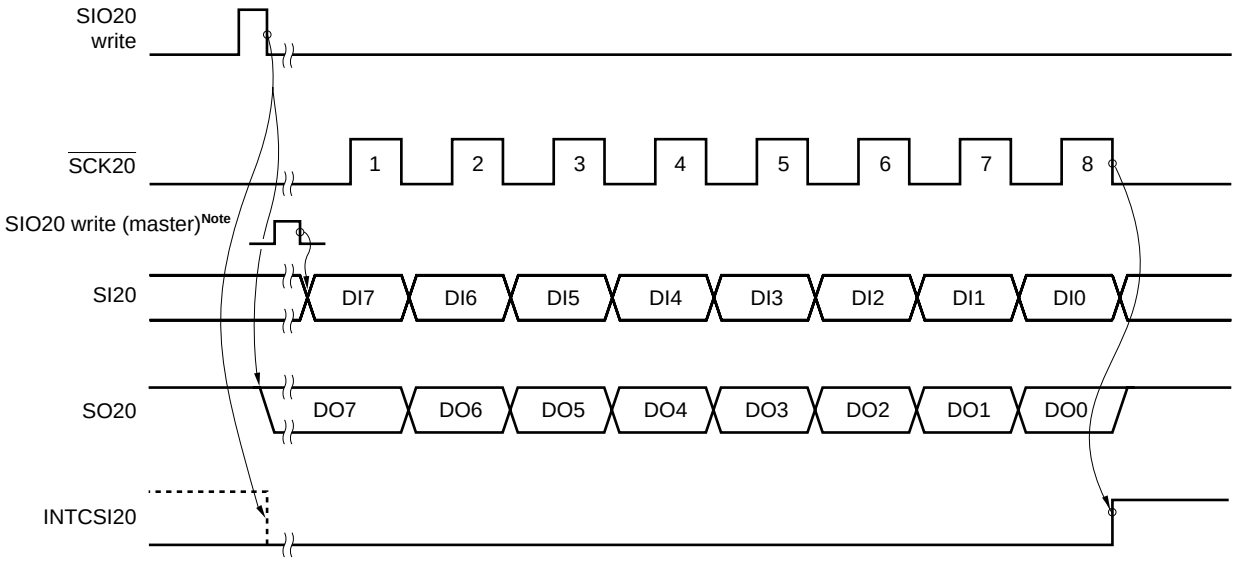
2. DO0 is output until $\overline{SS20}$ rises.
When $\overline{SS20}$ is high, SO20 is in a high-impedance state.

Figure 12-11. 3-Wire Serial I/O Mode Timing (3/7)

(iv) Master operation (when DAP20 = 0, CKP20 = 1, SSE20 = 0)



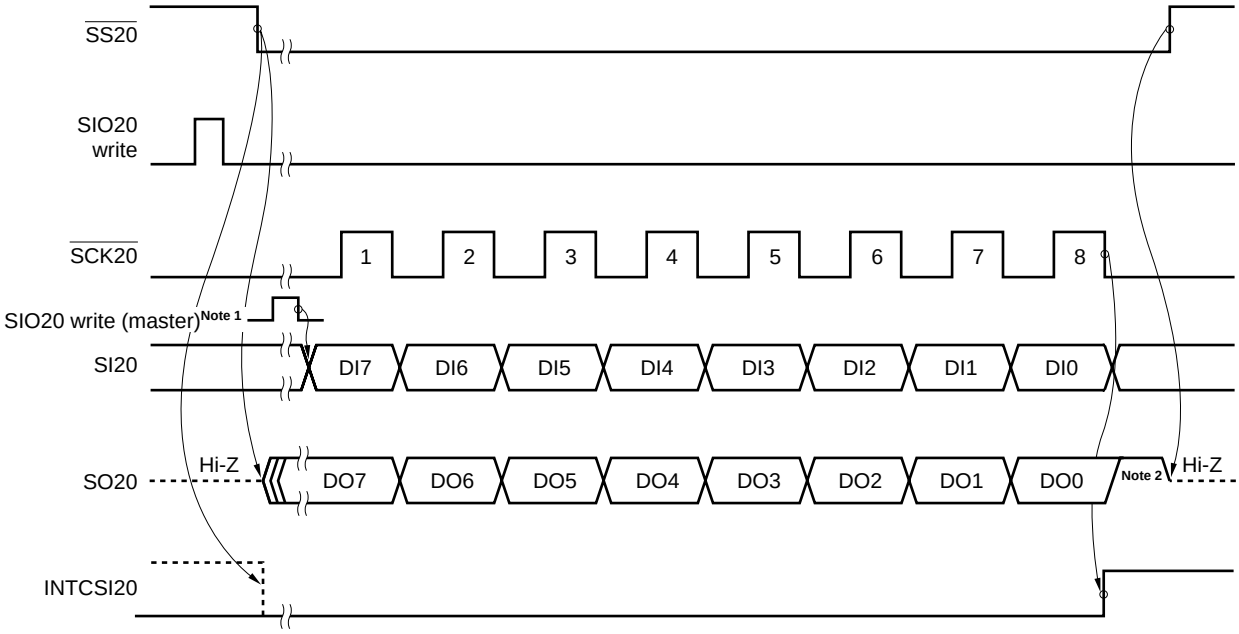
(v) Slave operation (when DAP20 = 0, CKP20 = 1, SSE20 = 0)



Note The data of SI20 is loaded at the first rising edge of $\overline{\text{SCK20}}$. Make sure that the master outputs the first bit before the first rising of $\overline{\text{SCK20}}$.

Figure 12-11. 3-Wire Serial I/O Mode Timing (4/7)

(vi) Slave operation (when DAP20 = 0, CKP20 = 1, SSE20 = 1)



Notes 1. The data of SI20 is loaded at the first rising edge of $\overline{\text{SCK20}}$. Make sure that the master outputs the first bit before the first rising of $\overline{\text{SCK20}}$.

2. SO20 is high until $\overline{\text{SS20}}$ rises after completion of DO0 output. When $\overline{\text{SS20}}$ is high, SO20 is in a high-impedance state.

(vii) Master operation (when DAP20 = 1, CKP20 = 0, SSE20 = 0)

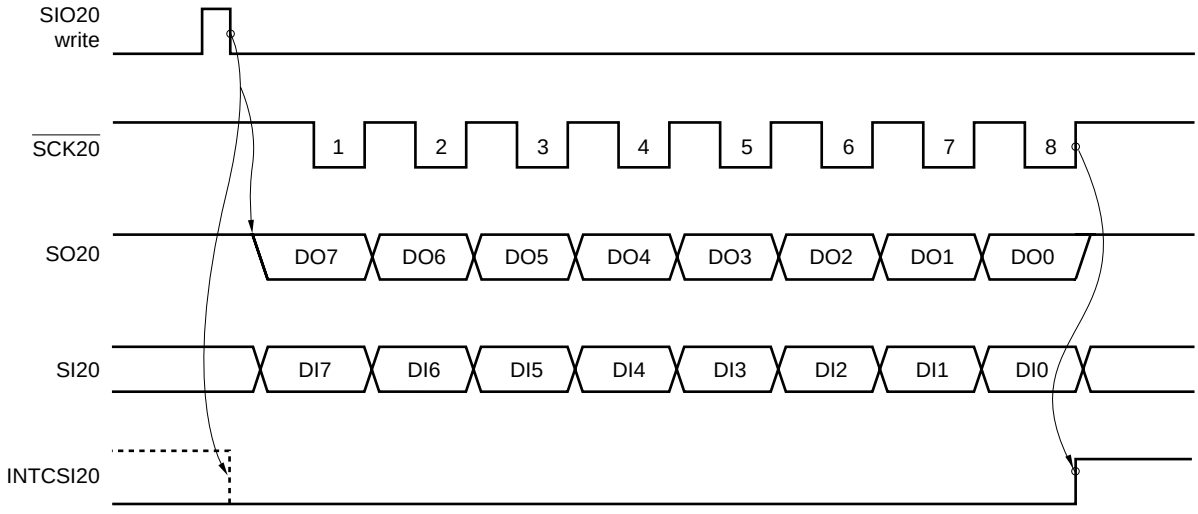
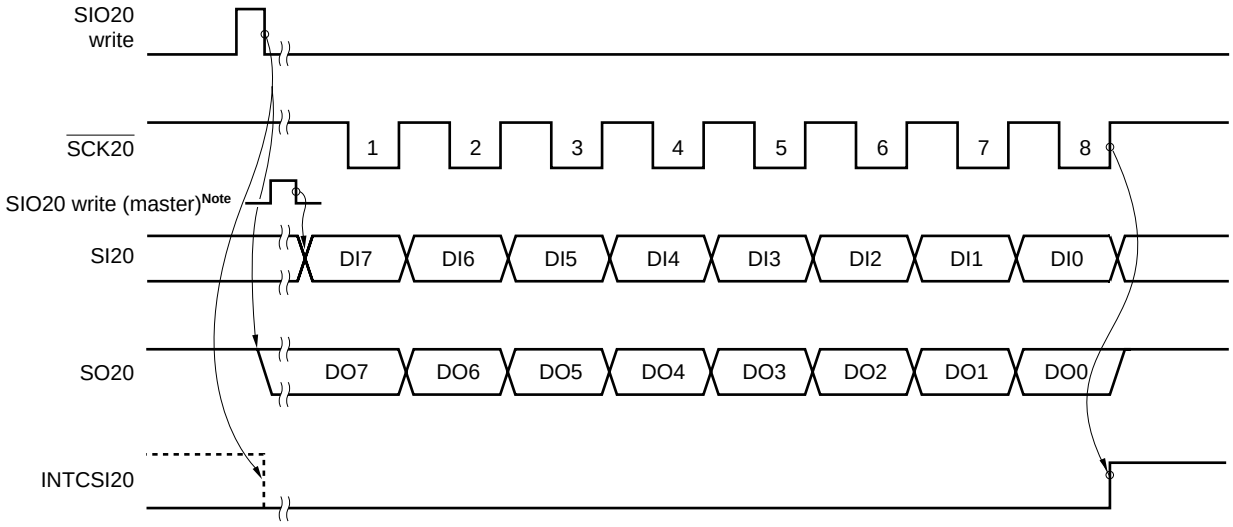


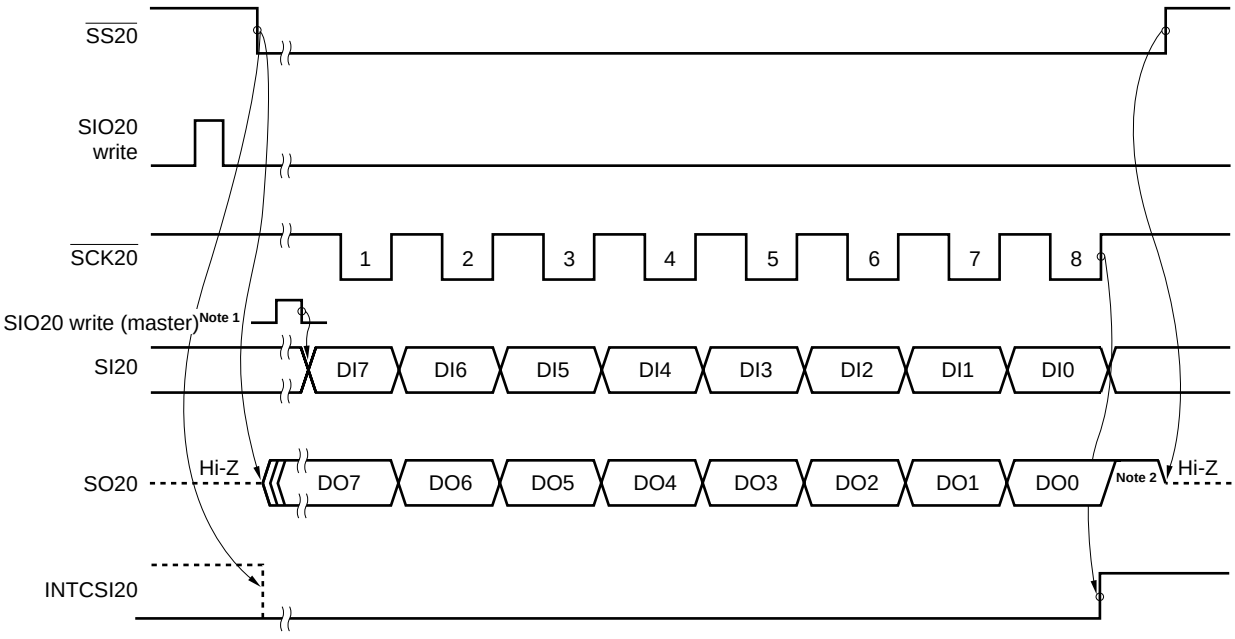
Figure 12-11. 3-Wire Serial I/O Mode Timing (5/7)

(viii) Slave operation (when DAP20 = 1, CKP20 = 0, SSE20 = 0)



Note The data of SI20 is loaded at the first falling edge of $\overline{\text{SCK20}}$. Make sure that the master outputs the first bit before the first falling of $\overline{\text{SCK20}}$.

(ix) Slave operation (when DAP20 = 1, CKP20 = 0, SSE20 = 1)

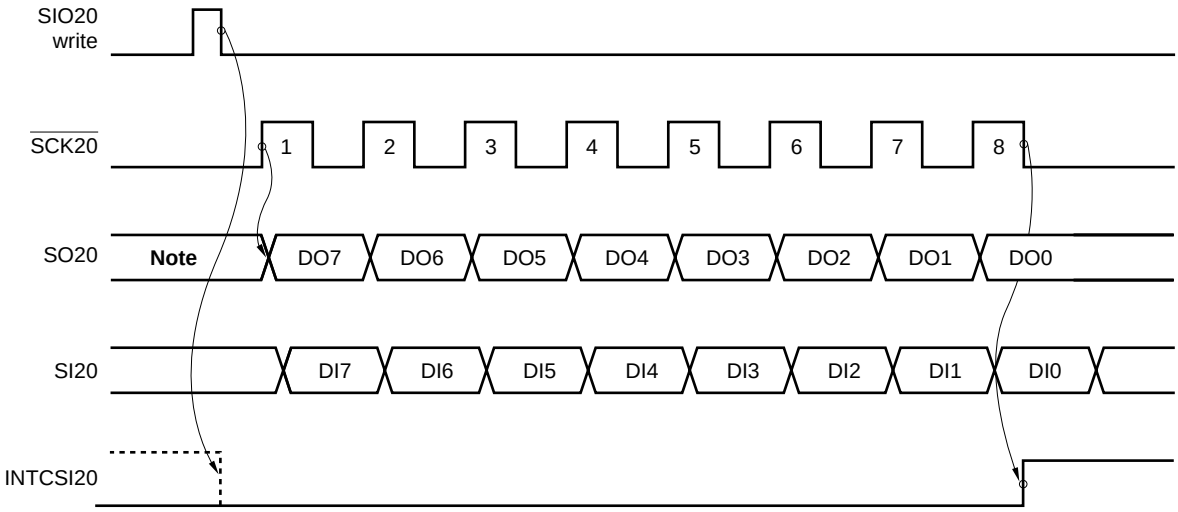


Notes 1. The data of SI20 is loaded at the first falling edge of $\overline{\text{SCK20}}$. Make sure that the master outputs the first bit before the first falling of $\overline{\text{SCK20}}$.

2. SO20 is high until $\overline{\text{SS20}}$ rises after completion of DO0 output. When $\overline{\text{SS20}}$ is high, SO20 is in a high-impedance state.

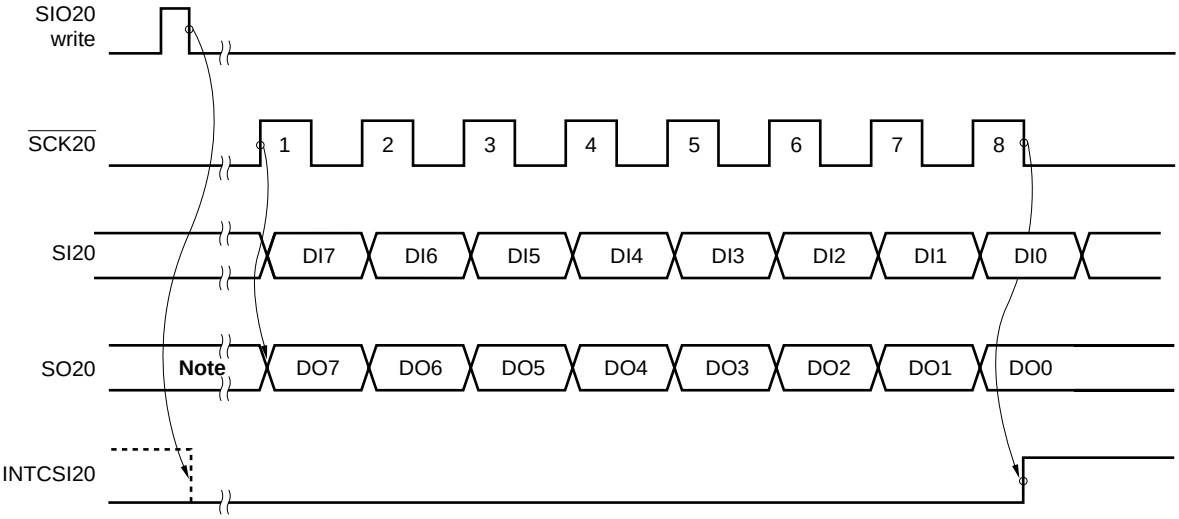
Figure 12-11. 3-Wire Serial I/O Mode Timing (6/7)

(x) Master operation (when DAP20 = 1, CKP20 = 1, SSE20 = 0)



Note The value of the last bit previously output is output.

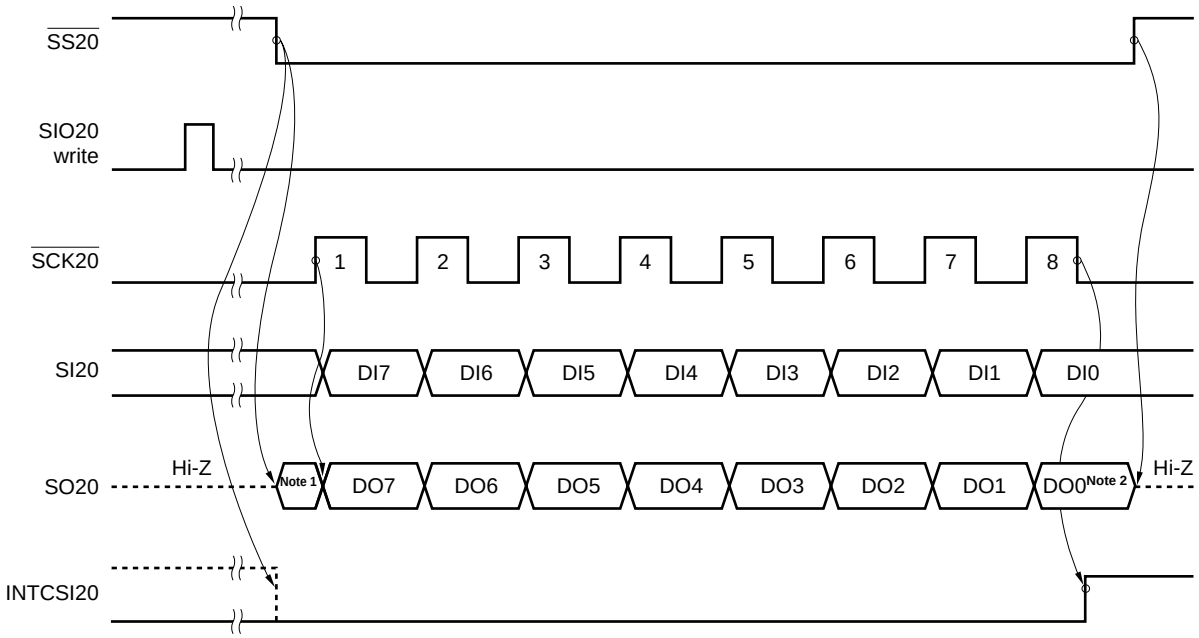
(xi) Slave operation (when DAP20 = 1, CKP20 = 1, SSE20 = 0)



Note The value of the last bit previously output is output.

Figure 12-11. 3-Wire Serial I/O Mode Timing (7/7)

(xii) Slave operation (when DAP20 = 1, CKP20 = 1, SSE20 = 1)



Notes 1. The value of the last bit previously output is output.

2. DO0 is output until $\overline{SS20}$ rises.
When $\overline{SS20}$ is high, SO20 is in a high-impedance state.

(3) Transfer start

Serial transfer is started by setting transfer data to the transmission shift register (TXS20/SIO20) when the following two conditions are satisfied.

- Bit 7 (CSIE20) of serial operation mode register 20 (CSIM20) = 1
- Internal serial clock is stopped or $\overline{SCK20}$ is high after 8-bit serial transfer.

Caution If CSIE20 is set to "1" after data is written to TXS20/SIO20, transfer does not start.

Termination of 8-bit transfer stops the serial transfer automatically and generates the interrupt request signal (INTCSI20).

CHAPTER 13 LCD CONTROLLER/DRIVER

13.1 LCD Controller/Driver Functions

The functions of the LCD controller/driver of the μPD789426, 789436, 789446, and 789456 Subseries are as follows.

(1) Automatic output of segment and common signals based on automatic display data memory read
(2) Two different display modes:
  • 1/3 duty (1/3 bias)
  • 1/4 duty (1/3 bias)
(3) Four different frame frequencies, selectable in each display mode
(4) Operation with a subsystem clock

Table 13-1 lists the maximum number of pixels that can be displayed in each display mode.

Table 13-1. Number of Segment Outputs and Maximum Number of Pixels

	Bias Method	Time Slots	Common Signals Used	Maximum Number of Segments	Maximum Number of Pixels
μPD789426, 789436 Subseries	1/3	3	COM0 to COM2	5	15 (5 segments × 3 commons)
		4	COM0 to COM3		20 (5 segments × 4 commons)
μPD789446, 789456 Subseries		3	COM0 to COM2	15	45 (15 segments × 3 commons)
		4	COM0 to COM3		60 (15 segments × 4 commons)

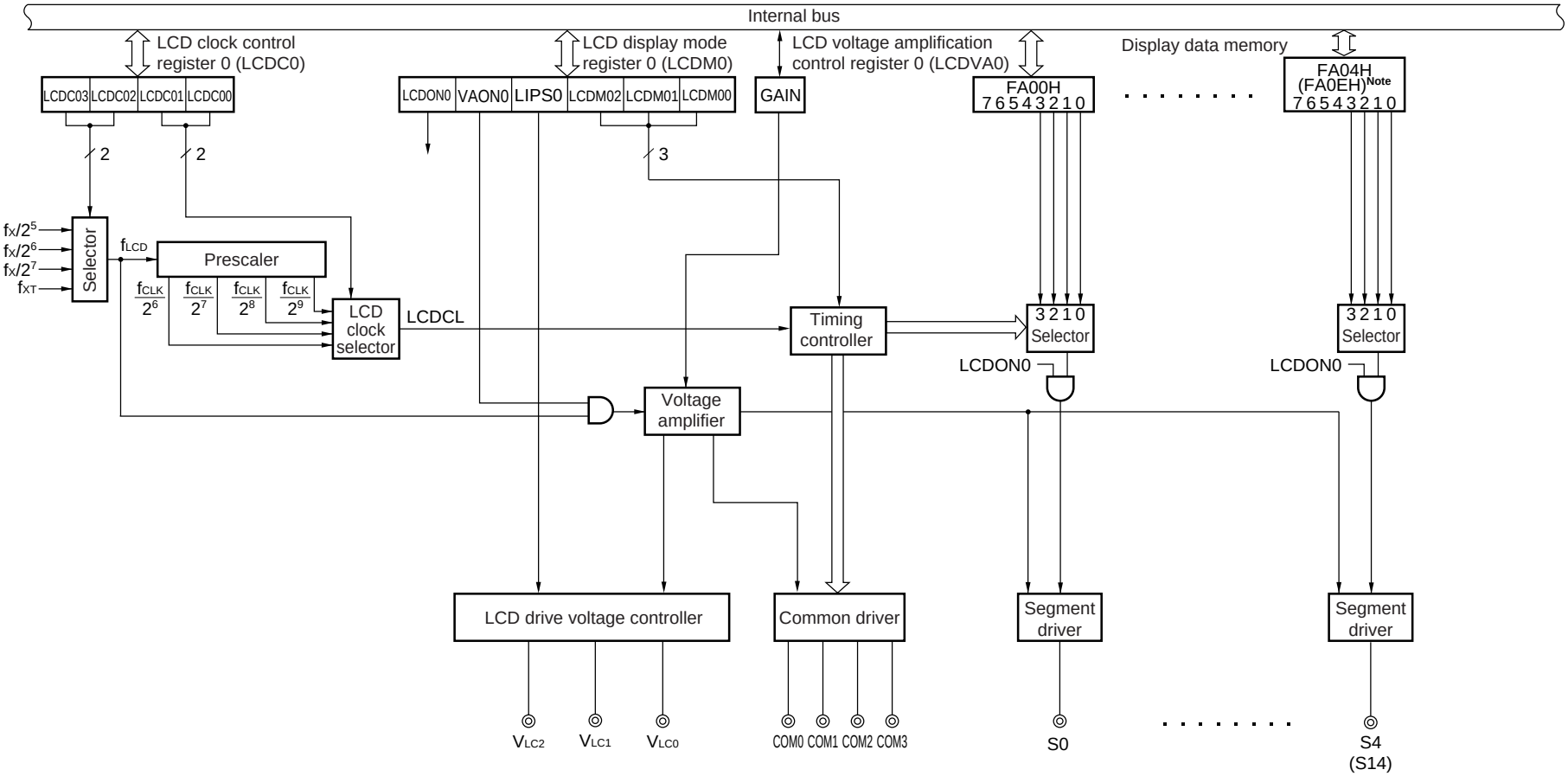
13.2 LCD Controller/Driver Configuration

The LCD controller/driver includes the following hardware.

Table 13-2. Configuration of LCD Controller/Driver

Item	Configuration
Display outputs	Segment signals: 5 (μPD789426 and 789436 Subseries) 15 (μPD789446 and 789456 Subseries) Common signals: 4 (COM0 to COM3)
Control registers	LCD display mode register 0 (LCDM0) LCD clock control register 0 (LCDC0) LCD voltage amplification control register 0 (LCDVA0)

Figure 13-1. Block Diagram of LCD Controller/Driver



Note The parenthesized values apply to the μPD789446 and 789456 Subseries.

13.3 Registers Controlling LCD Controller/Driver

- LCD display mode register 0 (LCDM0)
- LCD clock control register 0 (LCDC0)
- LCD voltage amplification control register 0 (LCDVA0)

(1) LCD display mode register 0 (LCDM0)

LCDM0 specifies whether to enable display operation. It also specifies the operation mode, LCD drive power supply, and display mode.

LCDM0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets LCDM0 to 00H.

Figure 13-2. Format of LCD Display Mode Register 0

Symbol	<7>	<6>	5	<4>	3	2	1	0	Address	After reset	R/W
LCDM0	LCDON0	VAON0	0	LIPS0	0	0	0	LCDM00	FFB0H	00H	R/W

LCDON0	LCD display enable/disable
0	Display off
1	Display on

VAON0	LCD controller/driver operation mode[Note]
0	No internal voltage amplification (Normal operation)
1	Internal voltage amplification enabled (Low-voltage operation)

LIPS0	Segment pin/common pin output control bit[Note]
0	Output ground level to segment/common pin
1	Output deselect level to segment pin and LCD waveform to common pin

LCDM00	LCD controller/driver display mode selection	
	Number of time slices	Bias mode
0	4	1/3
1	3	1/3

Note When the LCD display panel is not used, the VAON0 and LIPS0 must be set to 0 to reduce power consumption.

Cautions 1. Bits 1 to 3 and 5 must be set to 0.

2. When operating VAON0, follow the procedure described below.

A. To stop voltage amplification after switching display status from on to off:

1) Set to display off status by setting LCDON0 = 0.

2) Disable outputs of all the segment buffers and common buffers by setting LIPS0 = 0.

3) Stop voltage amplification by setting VAON0= 0.

B. To stop voltage amplification during display on status:

Setting prohibited. Be sure to stop voltage amplification after setting display off.

C. To set display on from voltage amplification stop status:

1) Start voltage amplification by setting VAON0 = 1, then wait for about 500 ms.

2) Set all the segment buffers and common buffers to non-display output status by setting LIPS0 = 1.

3) Set display on by setting LCDON0 = 1.

(2) LCD clock control register 0 (LCDC0)

LCDC0 specifies the LCD clock and frame frequency.

LCDC0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets LCDC0 to 00H.

Figure 13-3. Format of LCD Clock Control Register 0

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
LCDC0	0	0	0	0	LCDC03	LCDC02	LCDC01	LCDC00	FFB2H	00H	R/W

LCDC03	LCDC02	Internal clock (f_{LCD}) selection[Note]
0	0	f_{XT} (32.768 kHz)
0	1	$f_X/2^5$ (156.3 kHz)
1	0	$f_X/2^6$ (78.1 kHz)
1	1	$f_X/2^7$ (39.1 kHz)

LCDC01	LCDC00	LCD clock (LCDCL) selection
0	0	$f_{LCD}/2^6$
0	1	$f_{LCD}/2^7$
1	0	$f_{LCD}/2^8$
1	1	$f_{LCD}/2^9$

Note Specify an internal clock (f_{LCD}) frequency of at least 32 kHz.

Cautions 1. Bits 4 to 7 must be set to 0.

2. Before changing the LCDC0 setting, be sure to stop voltage amplification (VAON0 = 0).

3. Set the frame frequency to 128 Hz or lower.

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.

Table 13-3 lists the frame frequencies used when f_{XT} (32.768 kHz) is supplied to the internal clock (f_{CLK1}).

Table 13-3. Frame Frequencies (Hz)

LCD Clock (f_{LCD}) / Display Duty Ratio	$f_{XT}/2^9$ (64 Hz)	$f_{XT}/2^8$ (128 Hz)	$f_{XT}/2^7$ (256 Hz)	$f_{XT}/2^6$ (512 Hz)
1/3	21	43	85	171[Note]
1/4	16	32	64	128

Note This setting is prohibited because it causes the frame frequency to exceed 128 Hz.

(3) LCD voltage amplification control register 0 (LCDVA0)

LCDVA0 controls the voltage amplification level during the voltage amplifier operation.

Figure 13-4. Format of LCD Voltage Amplification Control Register 0

Symbol	7	6	5	4	3	2	1	<0>	Address	After reset	R/W
LCDVA0	0	0	0	0	0	0	0	GAIN	FFB3H	00H	R/W

GAIN	Reference voltage (V_{LC2}) level selection[Note]
0	1.5 times (specification of the LCD panel used is 4.5 V.)
1	1.0 times (specification of the LCD panel used is 3 V.)

Note Select the settings according to the specifications of the LCD panel that is used.

Caution Before changing the LCDVA0 setting, be sure to stop voltage amplification (VAON0 = 0).

Remark The TYP. value is indicated as the reference voltage (V_{LC2}) value.

13.4 Setting LCD Controller/Driver

Set the LCD controller/driver using the following procedure.

<1> Set the frame frequency using LCD clock control register 0 (LCDC0).
<2> Set the voltage amplification level using LCD voltage amplification control register 0 (LCDVA0).
GAIN = 0: V_{LC0} = 4.5 V, V_{LC1} = 3 V, V_{LC2} = 1. 5 V
GAIN = 1: V_{LC0} = 3 V, V_{LC1} = 2 V, V_{LC2} = 1 V
<3> Set the time division using LCDM00 (bit 0 of LCD display mode register 0 (LCDM0)).
<4> Enable voltage amplification by setting VAON0 (bit 6 of LCDM0) (VAON0 = 1).
<5> Wait for 500 ms or more after setting VAON0.
<6> Set LIPS0 (bit 4 of LCDM0) (LIPS0 = 1) and output the deselect potential.
<7> Start output corresponding to each data memory by setting LCDON0 (bit 7 of LCDM0) (LCDON0 =1).

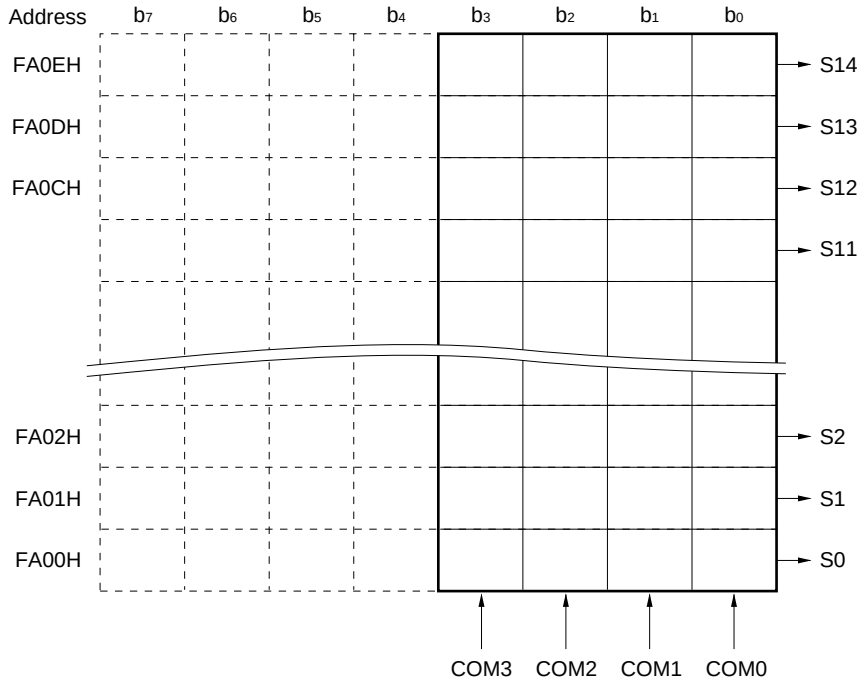
13.5 LCD Display Data Memory

The LCD display data memory is mapped at addresses FA00H to FA0EH. Data in the LCD display data memory can be displayed on the LCD panel using the LCD controller/driver.

Figure 13-5 shows the relationship between the contents of the LCD display data memory and the segment/common outputs.

That part of the display data memory which is not used for display can be used as ordinary RAM.

Figure 13-5. Relationship Between LCD Display Data Memory Contents and Segment/Common Outputs (μPD789446, 789456 Subseries)



Caution No memory has been installed as the higher 4 bits of the LCD display data memory. Be sure to set 0 to them.

13.6 Common and Segment Signals

Each pixel of the LCD panel turns on when the potential difference between the corresponding common and segment signals becomes higher than a specific voltage (LCD drive voltage, V_{LCD}). It turns off when the potential difference becomes lower than V_{LCD}.

Applying DC voltage to the common and segment signals for an LCD panel would deteriorate it. To avoid this problem, this LCD panel is driven with AC voltage.

(1) Common signals

Each common signal is selected sequentially according to a specified number of time slots at the timing listed in Table 13-4. In the static display mode, the same signal is output to COM0 to COM3 in common.

In the three-time slot mode, keep the COM3 pin open.

Table 13-4. COM Signals

Number of Time Slots \ COM Signal	COM0	COM1	COM2	COM3
Three-time slot mode				Open
Four-time slot mode				

(2) Segment signals

The segment signals correspond to LCD display data memory. Bits 0, 1, 2, and 3 of each byte are read in synchronization with COM0, COM1, COM2, and COM3, respectively. If the contents of each bit are 1, it is converted to the select voltage, and if 0, it is converted to the deselect voltage. The conversion results are output to the segment pins.

Check, with the information given above, what combination of the front-surface electrodes (corresponding to the segment signals) and the rear-surface electrodes (corresponding to the common signals) forms display patterns in the LCD display data memory, and write the bit data that corresponds to the desired display pattern on a one-to-one basis.

Bit 3 of the LCD display data memory is not used for LCD display in the three-time slot mode. So this bit can be used for purposes other than display.

LCD display data memory bits 4 to 7 are fixed to 0.

(3) Output waveforms of common and segment signals

Voltages listed in Table 13-5 are output as common and segment signals.

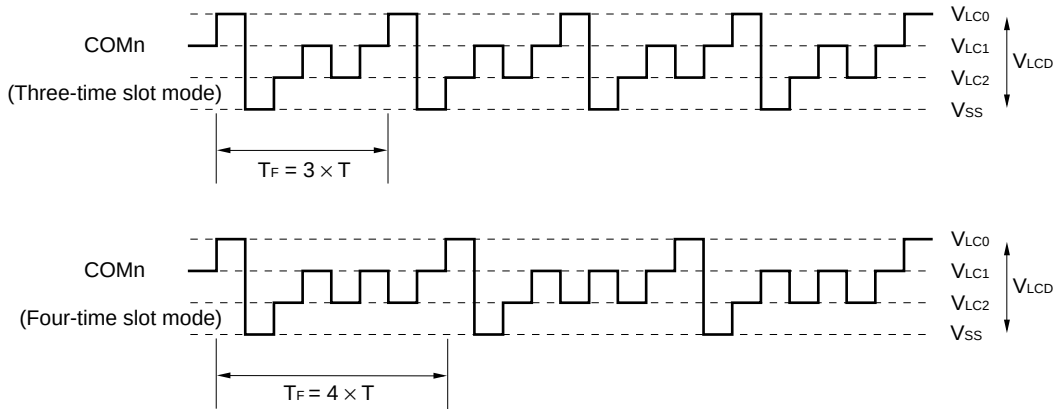
When both common and segment signals are at the select voltage, a display-on voltage of $\pm V_{LCD}$ is obtained. The other combinations of the signals correspond to the display-off voltage.

Table 13-5. LCD Drive Voltage

Common Signal \ Segment Signal		Select Signal Level	Deselect Signal Level
		V_{SS0}/V_{LC0}	V_{LC1}/V_{LC2}
Select signal level	V_{LC0}/V_{SS0}	$-V_{LCD}/+V_{LCD}$	$-\frac{1}{3}V_{LCD}/+\frac{1}{3}V_{LCD}$
Deselect signal level	V_{LC2}/V_{LC1}	$-\frac{1}{3}V_{LCD}/+\frac{1}{3}V_{LCD}$	$-\frac{1}{3}V_{LCD}/+\frac{1}{3}V_{LCD}$

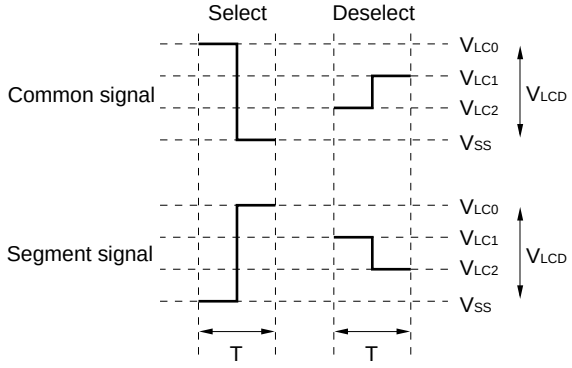
Figure 13-6 shows the common signal waveforms, and Figure 13-7 shows the voltages and phases of the common and segment signals.

Figure 13-6. Common Signal Waveforms



T: One LCD clock period T_F: Frame frequency

Figure 13-7. Voltages and Phases of Common and Segment Signals



T: One LCD clock period

13.7 Display Modes

13.7.1 Three-time slot display example

Figure 13-9 shows how the 5-digit LCD panel having the display pattern shown in Figure 13-8 is connected to the segment signals (S0 to S14) and the common signals (COM0 to COM2) of the μPD789446 or μPD789456 Subseries chip. This example displays data “123.45” in the LCD panel. The contents of the display data memory (addresses FA00H to FA0EH) correspond to this display.

The following description focuses on numeral “3.” (*3*.) displayed in the third digit. To display “3.” in the LCD panel, it is necessary to apply the select or deselect voltage to the S6 to S8 pins according to Table 13-6 at the timing of the common signals COM0 to COM2.

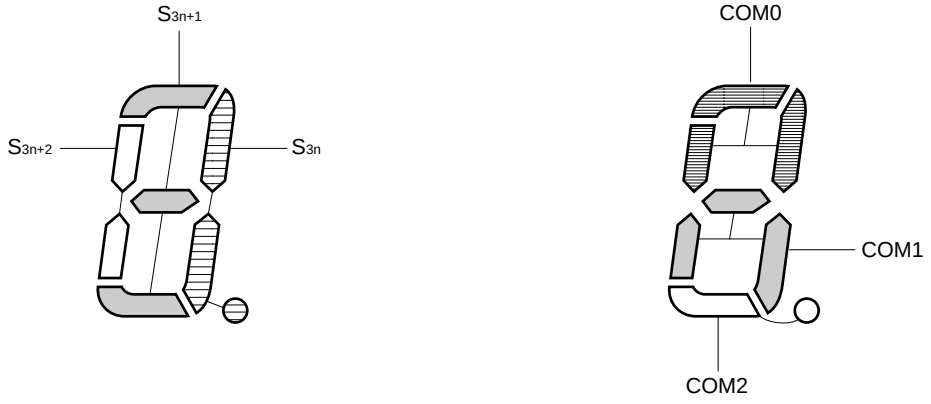
Table 13-6. Select and Deselect Voltages (COM0 to COM2)

Segment / Common	S6	S7	S8
COM0	Select	Select	Deselect
COM1	Select	Select	Deselect
COM2	Select	Select	–

According to Table 13-6, it is determined that the display data memory location (FA06H) that corresponds to S6 must contain x111.

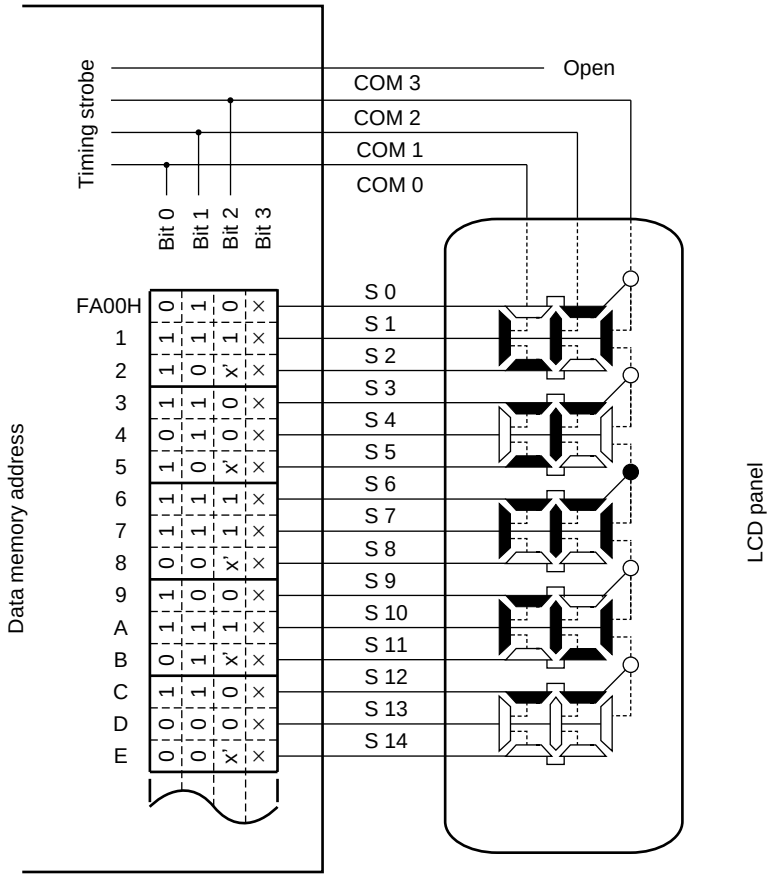
Figure 13-10 shows examples of LCD drive waveforms between the S6 signal and each common signal. When the select voltage is applied to S6 at the timing of COM1 or COM2, an alternate rectangle waveform, $+V_{LCD}/-V_{LCD}$, is generated to turn on the corresponding LCD segment.

Figure 13-8. Three-Time Slot LCD Display Pattern and Electrode Connections



Remark n = 0 to 4

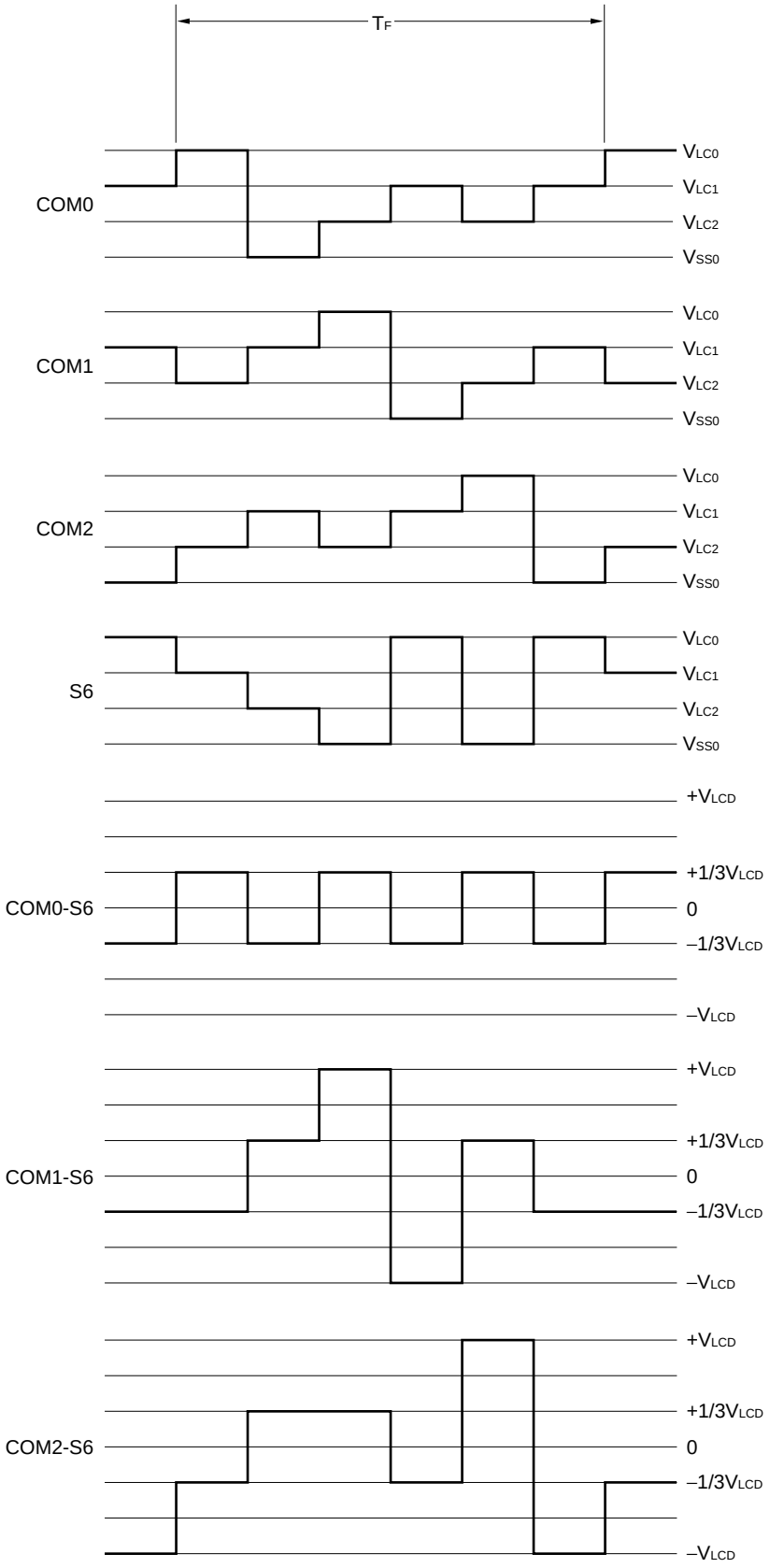
Figure 13-9. Example of Connecting Three-Time Slot LCD Panel



x': Can be used to store any data because there is no corresponding segment in the LCD panel.

×: Can always be used to store any data because of the three-time slot mode being used.

Figure 13-10. Three-Time Slot LCD Drive Waveform Examples



13.7.2 Four-time slot display example

Figure 13-12 shows how the 7-digit LCD panel having the display pattern shown in Figure 13-11 is connected to the segment signals (S0 to S14) and the common signals (COM0 to COM3) of the μPD789446 or μPD789456 Subseries chip. This example displays data “123456.7” in the LCD panel. The contents of the display data memory (addresses FA00H to FA0EH) correspond to this display.

The following description focuses on numeral “6.” (6.) displayed in the seventh digit. To display “6.” in the LCD panel, it is necessary to apply the select or deselect voltage to the S2 and S3 pins according to Table 13-7 at the timing of the common signals COM0 to COM3.

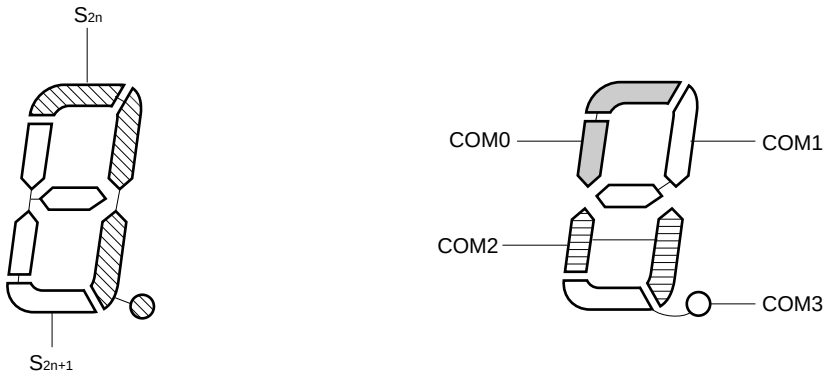
Table 13-7. Select and Deselect Voltages (COM0 to COM3)

Segment / Common	S2	S3
COM0	Select	Select
COM1	Deselect	Select
COM2	Select	Select
COM3	Select	Select

According to Table 13-7, it is determined that the display data memory location (FA02H) that corresponds to S2 must contain 1101.

Figure 13-13 shows examples of LCD drive waveforms between the S2 signal and the COM0 or COM1 signal (the waveforms for COM2 and COM3 have been left out from the drawing). When the select voltage is applied to S2 at the timing of COM0, an alternate rectangle waveform, $+V_{LCD}/-V_{LCD}$, is generated to turn on the corresponding LCD segment.

Figure 13-11. Four-Time Slot LCD Display Pattern and Electrode Connections



Remark n = 0 to 7

Figure 13-12. Example of Connecting Four-Time Slot LCD Panel

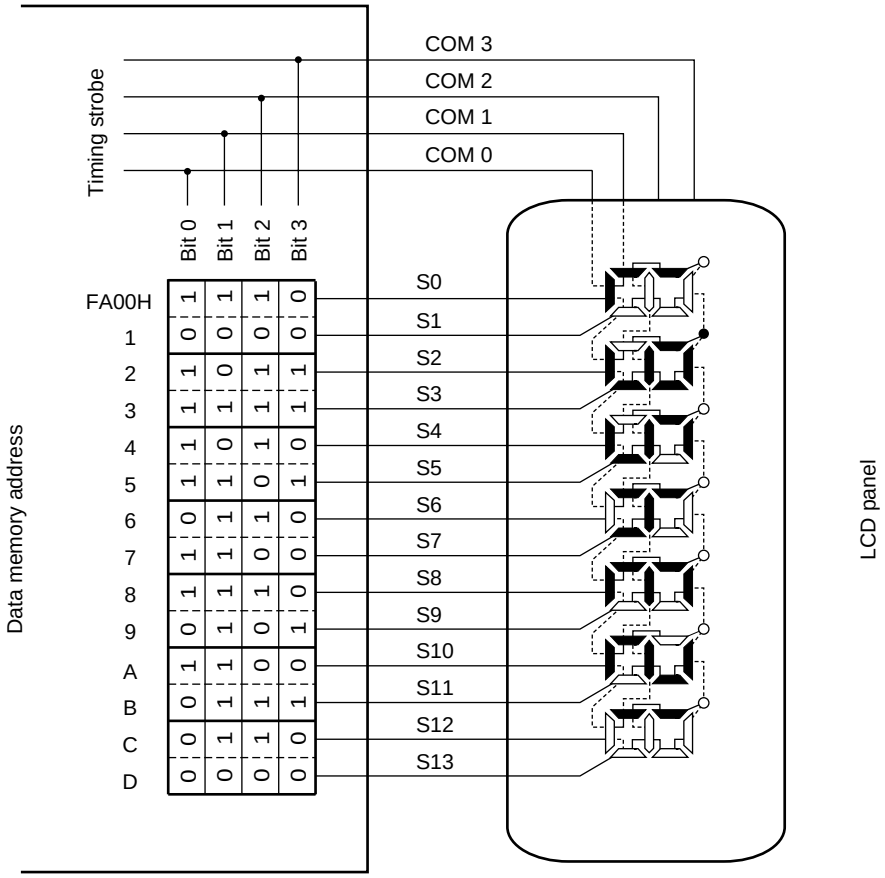
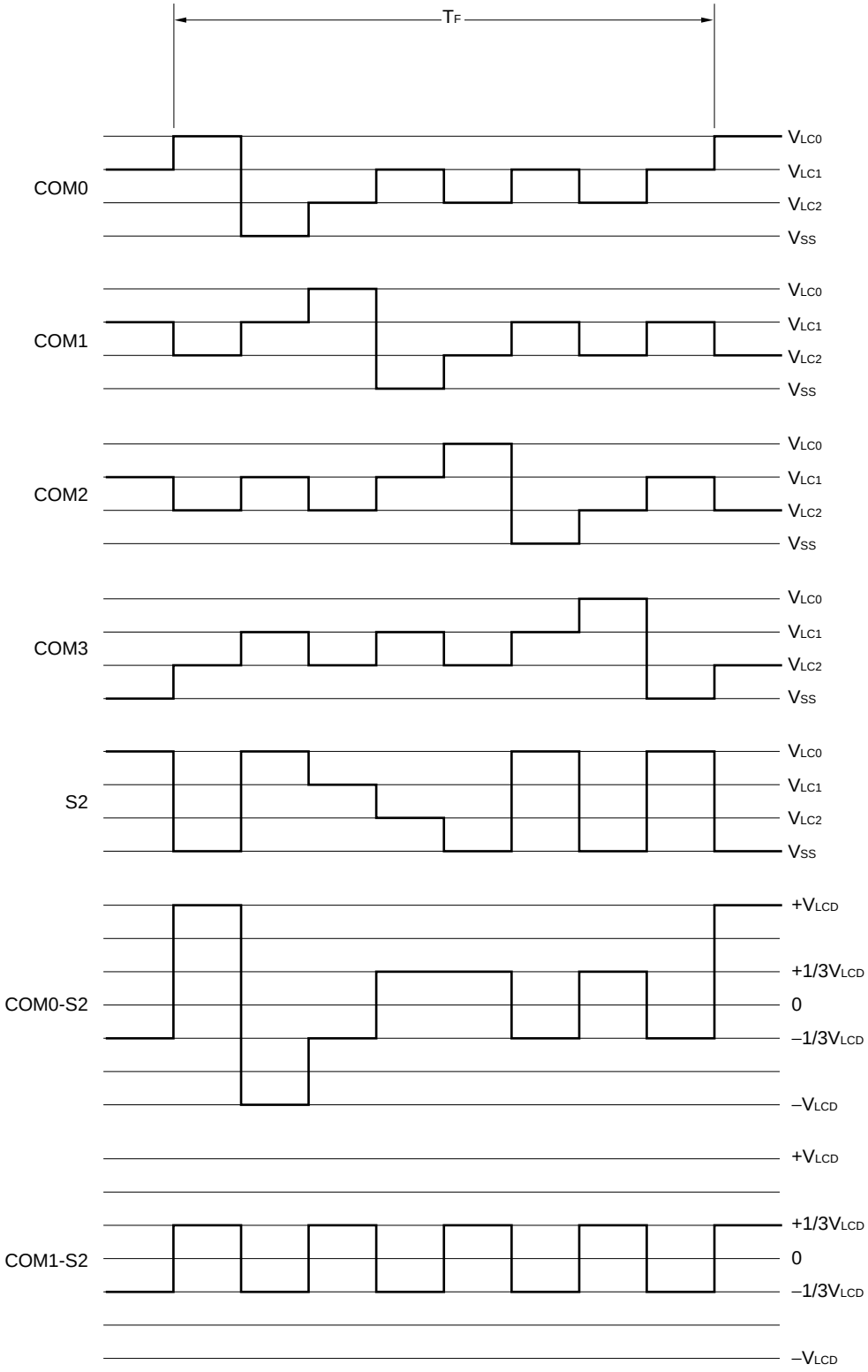


Figure 13-13. Four-Time Slot LCD Drive Waveform Examples



[MEMO]

CHAPTER 14 INTERRUPT FUNCTIONS

14.1 Interrupt Function Types

The following two types of interrupt functions are used.

(1) Non-maskable interrupt

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

A standby release signal is generated.

One interrupt source from the watchdog timer is incorporated as a non-maskable interrupt.

(2) Maskable interrupt

This interrupt undergoes mask control. If two or more interrupts with the same priority are simultaneously generated, each interrupt has a predetermined priority as shown in Table 14-1.

A standby release signal is generated.

5 external and 9 internal interrupt sources are incorporated as maskable interrupts.

14.2 Interrupt Sources and Configuration

A total of 15 non-maskable and maskable interrupts are incorporated as interrupt sources (see **Table 14-1**).

Table 14-1. 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 (with watchdog timer mode 1 selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (with interval timer mode selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTSR20	End of serial interface 20 UART reception	Internal	000EH	(B)
		INTCSI20	End of serial interface 20 3-wire SIO transfer reception			
	6	INTST20	End of serial interface 20 UART transmission		0012H	
	7	INTWTI	Interval timer interrupt		0014H	
	8	INTTM90	Generation of match signal of 16-bit timer 90		0016H	
	9	INTTM50	Generation of match signal of 8-bit timer 50		0018H	
	10	INTTM60	Generation of match signal of 8-bit timer 60		001AH	
	11	INTAD0	End of A/D conversion signal		001CH	
	12	INTWT	Watch timer interrupt		001EH	
	13	INTKR00	Key return signal detection	External	0020H	(C)

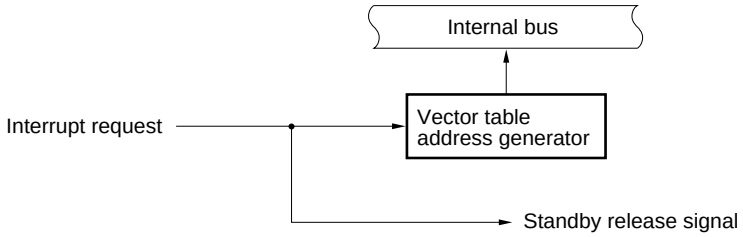
Notes 1. Priority is the priority order when several maskable interrupts are generated at the same time. 0 is the highest order and 13 is the lowest order.

2. Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 14-1.

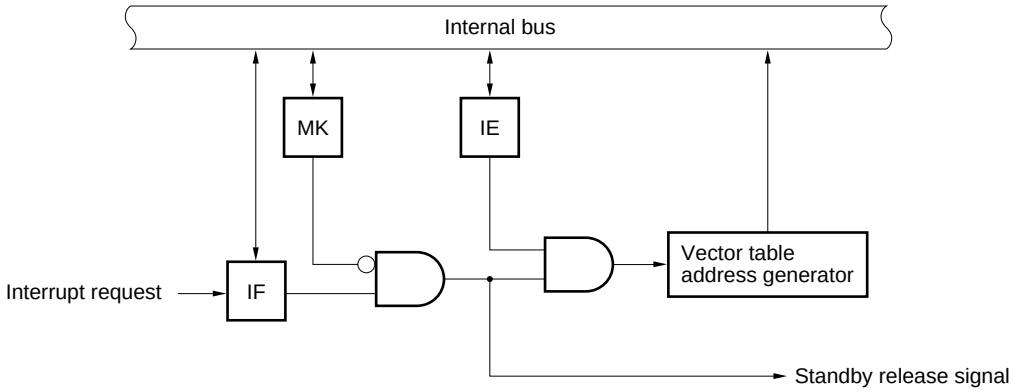
Remark There are two interrupt sources for the watchdog timer (INTWDT): non-maskable and maskable interrupts (internal). Either one (but not both) should be selected for actual use.

Figure 14-1. Basic Configuration of Interrupt Function

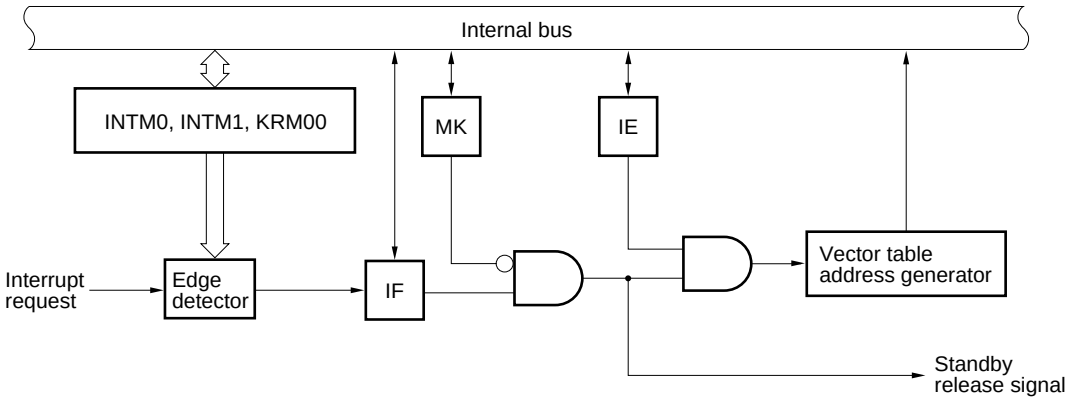
(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt



INTM0: External interrupt mode register 0
INTM1: External interrupt mode register 1
KRM00: Key return mode register 00
IF: Interrupt request flag
IE: Interrupt enable flag
MK: Interrupt mask flag

14.3 Registers Controlling Interrupt Function

The following five types of registers are used to control the interrupt functions.

- Interrupt request flag registers 0, 1 (IF0 and IF1)
- Interrupt mask flag registers 0, 1 (MK0 and MK1)
- External interrupt mode registers 0, 1 (INTM0 and INTM1)
- Program status word (PSW)
- Key return mode register 00 (KRM00)

Table 14-2 gives a listing of interrupt request flag and interrupt mask flag names corresponding to interrupt requests.

Table 14-2. Flags Corresponding to Interrupt Request Signal Name

Interrupt Request Signal Name	Interrupt Request Flag	Interrupt Mask Flag
INTWDT	WDTIF	WDTMK
INTP0	PIF0	PMK0
INTP1	PIF1	PMK1
INTP2	PIF2	PMK2
INTP3	PIF3	PMK3
INTSR20/INTCSI20	SRIF20	SRMK20
INTST20	STIF20	STMK20
INTWTI	WTIIF	WTIMK
INTTM90	TMIF90	TMMK90
INTTM50	TMIF50	TMMK50
INTTM60	TMIF60	TMMK60
INTAD0	ADIF0	ADMK0
INTWT	WTIF	WTMK
INTKR00	KRIF00	KRMK00

(1) Interrupt request flag registers 0, 1 (IF0 and IF1)

The interrupt request flag is set (1) when the corresponding interrupt request is generated or an instruction is executed. It is cleared (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 sets IF0 and IF1 to 00H.

Figure 14-2. Format of Interrupt Request Flag Registers

Symbol	<7>	6	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
IF0	STIF20	0	SRIF20	PIF3	PIF2	PIF1	PIF0	WDTIF	FFE0H	00H	R/W

	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
IF1	0	KRIF00	WTIF	ADIF0	TMIF40	TMIF30	TMIF20	WTIIF	FFE1H	00H	R/W

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

Cautions 1. Bit 7 of IF1 and bit 6 of IF0 must be set to 0.

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

3. Because port 3 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 0, 1 (MK0 and 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 MK0 and MK1 to FFH.

Figure 14-3. Format of Interrupt Mask Flag Registers

Symbol	<7>	6	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
MK0	STMK20	1	SRMK20	PMK3	PMK2	PMK1	PMK0	WDTMK	FFE4H	FFH	R/W

	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
MK1	1	KRMK00	WTMK	ADMK0	TMMK40	TMMK30	TMMK20	WTIMK	FFE5H	FFH	R/W

XXMK	Interrupt servicing control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Cautions 1. Bits 7 of MK1 and bit 6 of MK0 must be set to 1.

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

3. Because port 3 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 specify a valid edge for INTP0 to INTP2.

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

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

Figure 14-4. Format of External Interrupt Mode Register 0

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
INTM0	ES21	ES20	ES11	ES10	ES01	ES00	0	0	FFECH	00H	R/W

ES21	ES20	INTP2 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

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. Bits 0 and 1 must be set to 0.

2. Before setting the INTM0 register, be sure to set the relevant interrupt mask flag to 1 to disable interrupts.

After that, clear (0) the interrupt request flag, then set the interrupt mask flag to 0 to enable interrupts.

(4) External interrupt mode register 1 (INTM1)

INTM1 is used to specify a valid edge for INTP3.

INTM1 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets INTM1 to 00H.

Figure 14-5. Format of External Interrupt Mode Register 1

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
INTM1	0	0	0	0	0	0	ES31	ES30	FFEDH	00H	R/W

ES31	ES30	INTP3 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

Cautions 1. Bits 2 to 7 must be set to 0.

2. Before setting INTM1, set PMK3 to 1 to disable interrupts.
After that, clear (0) PIF3, then set PMK3 to 0 to enable interrupts.

(5) 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.

Besides 8-bit unit read/write, this register can carry out operations with a bit manipulation instruction and dedicated instructions (EI, DI). When a vectored interrupt is acknowledged, the PSW is automatically saved into a stack, and the IE flag is reset to 0.

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

Figure 14-6. Configuration of Program Status Word

Symbol	7	6	5	4	3	2	1	0	After reset
PSW	IE	Z	0	AC	0	0	1	CY	02H

Used when normal instruction is executed

IE	Interrupt acknowledgement enabled/disabled
0	Disabled
1	Enabled

(6) Key return mode register 00 (KRM00)

This register sets the pin that detects a key return signal (falling edge of port 0).

KRM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets KRM00 to 00H.

Figure 14-7. Format of Key Return Mode Register 00

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
KRM00	0	0	0	0	0	0	0	KRM000	FFF5H	00H	R/W

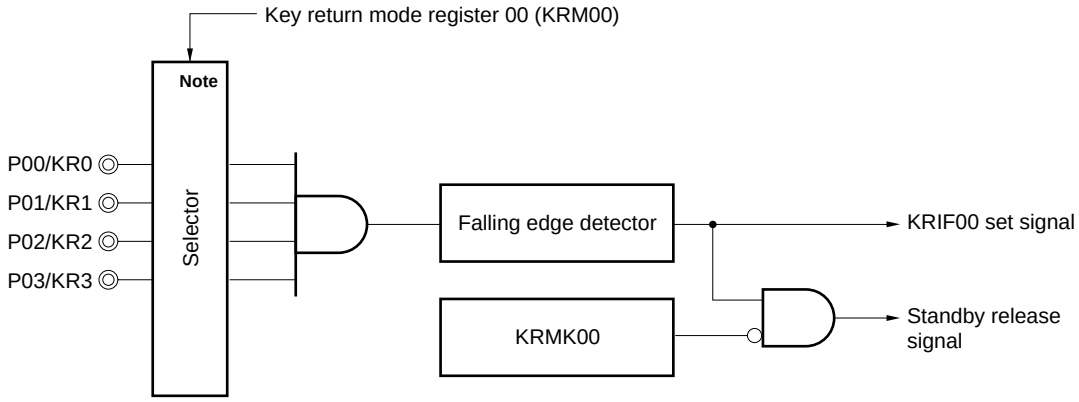
KRM000	Key return signal detection control
0	No detection
1	Detection (detecting falling edge of port 0)

Cautions 1. Bits 1 to 7 must be set to 0.

2. Before setting KRM00, always set bit 6 of MK1 (KRMK00 = 1) to disable interrupts. After setting KRM00, clear KRMK00 after clearing bit 6 of IF1 (KRIF00 = 0) to enable interrupts.

3. When P00 to P03 are in input mode, on-chip pull-up resistors are connected to P00 to P03 by the setting of KRM000. After switching to output mode, the on-chip pull-up resistors are cut off. However, key return signal detection continues.

Figure 14-8. Block Diagram of Falling Edge Detector



Note Selector that selects the pin used for falling edge input

14.4 Interrupt Servicing Operation

14.4.1 Non-maskable interrupt request acknowledgment operation

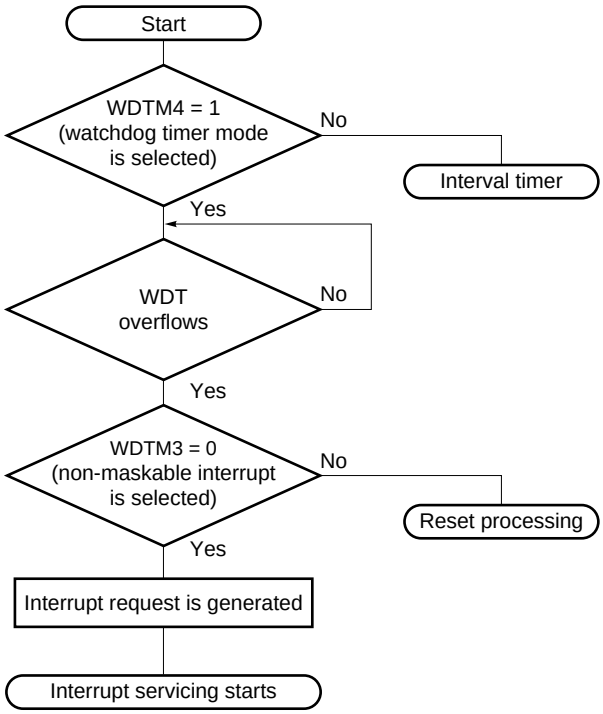
The non-maskable interrupt request is unconditionally acknowledged even when interrupts are disabled. It is not subject to interrupt priority control and takes precedence over all other interrupts.

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 14-9 shows the flow from non-maskable interrupt request generation to acknowledgement, Figure 14-10 shows the timing of non-maskable interrupt acknowledgement, and Figure 14-11 shows the acknowledgement operation when a number of non-maskable interrupts are generated.

Caution During 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 non-maskable interrupt request will be acknowledged.

Figure 14-9. Flow from Generation of Non-Maskable Interrupt Request to Acknowledgment



WDTM: Watchdog timer mode register

WDT: Watchdog timer

Figure 14-10. Timing of Non-Maskable Interrupt Request Acknowledgment

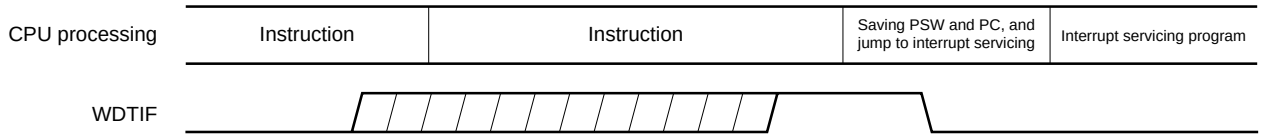
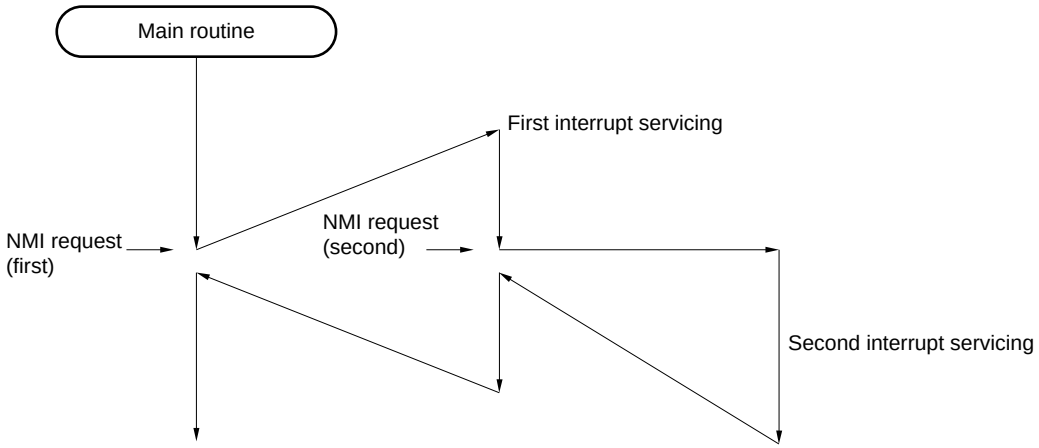


Figure 14-11. Non-Maskable Interrupt Request Acknowledgment



14.4.2 Maskable interrupt request acknowledgment 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 is acknowledged in the interrupt enabled status (when the IE flag is set to 1).

The time required to start the interrupt servicing after a maskable interrupt request has been generated is shown in Table 14-3.

Refer to Figures 14-13 and 14-14 for the timing of interrupt request acknowledgement.

Table 14-3. Time from Generation of Maskable Interrupt Request to Servicing

Minimum Time	Maximum Time[Note]
9 clocks	19 clocks

Note The wait time is maximum when an interrupt request is generated immediately before BT or 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 one assigned the highest priority by the priority specification flag.

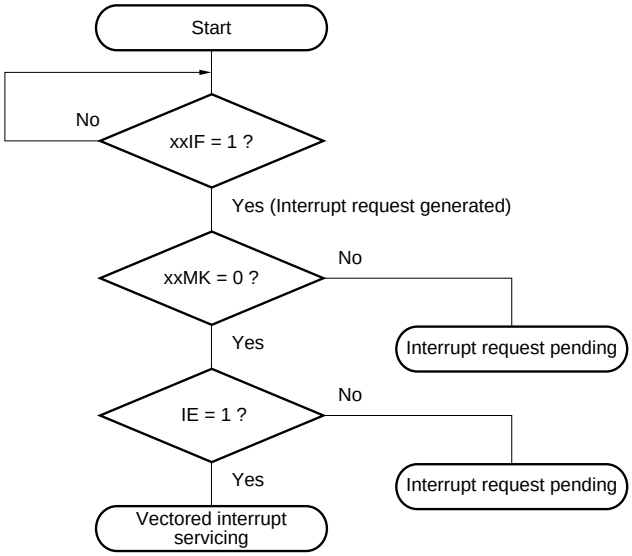
A pending interrupt is acknowledged when the status where it can be acknowledged is set.

Figure 14-12 shows the algorithm of interrupt request acknowledgement.

When a maskable interrupt request is acknowledged, the 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 servicing, use the RETI instruction.

Figure 14-12. Interrupt Request Acknowledgment Program Algorithm

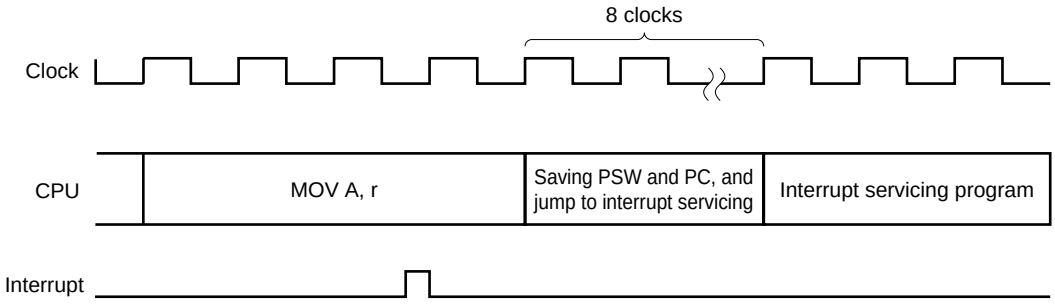


xxIF: Interrupt request flag

xxMK: Interrupt mask flag

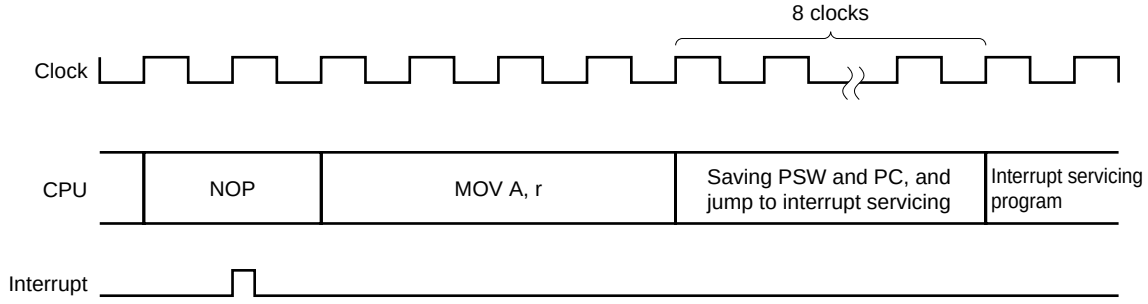
IE: Flag to control maskable interrupt request acknowledgement (1 = enable, 0 = disable)

Figure 14-13. Interrupt Request Acknowledgment Timing (Example: MOV A, r)



If the interrupt request has generated an interrupt request flag (XXIF) by the time the instruction clocks under execution, n clocks (n = 4 to 10), are n – 1, interrupt request acknowledgment processing will start following the completion of the instruction under execution. Figure 14-13 shows an example using the 8-bit data transfer instruction MOV A, r. Because this instruction is executed in 4 clocks, if an interrupt request is generated between the start of execution and the 3rd clock, interrupt request acknowledgment processing will take place following the completion of MOV A, r.

Figure 14-14. Interrupt Request Acknowledgment Timing
(When Interrupt Request Flag Is Generated in Final Clock Under Execution)



If the interrupt request flag (XXIF) is generated in the final clock of the instruction, interrupt request acknowledgment processing will begin after execution of the next instruction is complete.

Figure 14-14 shows an example whereby an interrupt request was generated in the 2nd clock of NOP (a 2-clock instruction). In this case, the interrupt request will be processed after execution of MOV A, r, which follows NOP, is complete.

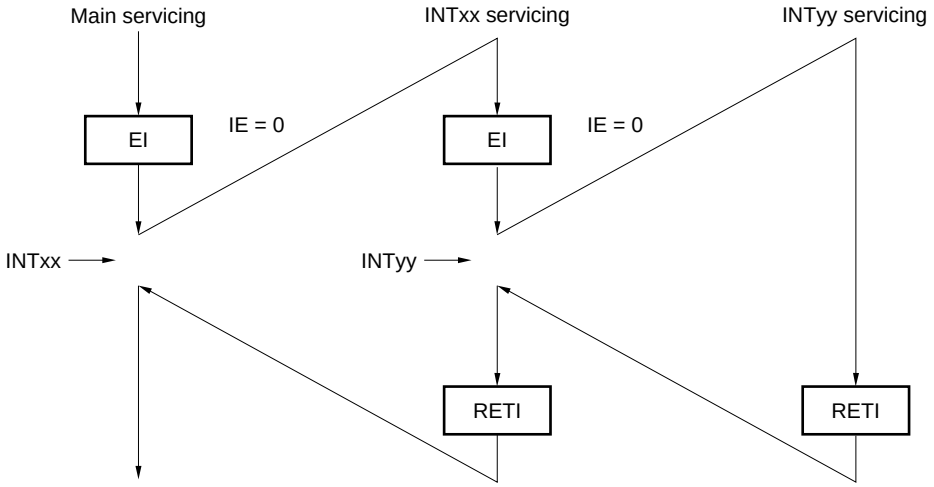
Caution When interrupt request flag registers 0 and 1 (IF0 and IF1), or interrupt mask flag registers 0 and 1 (MK0 and MK1) are being accessed, interrupt requests will be held pending.

14.4.3 Multiple interrupt servicing

Multiple interrupts, in which another interrupt request is acknowledged while an interrupt request being serviced, can be serviced using the priority order. If multiple interrupts are generated at the same time, they are serviced in the order according to the priority assigned to each interrupt request in advance (refer to **Table 14-1**).

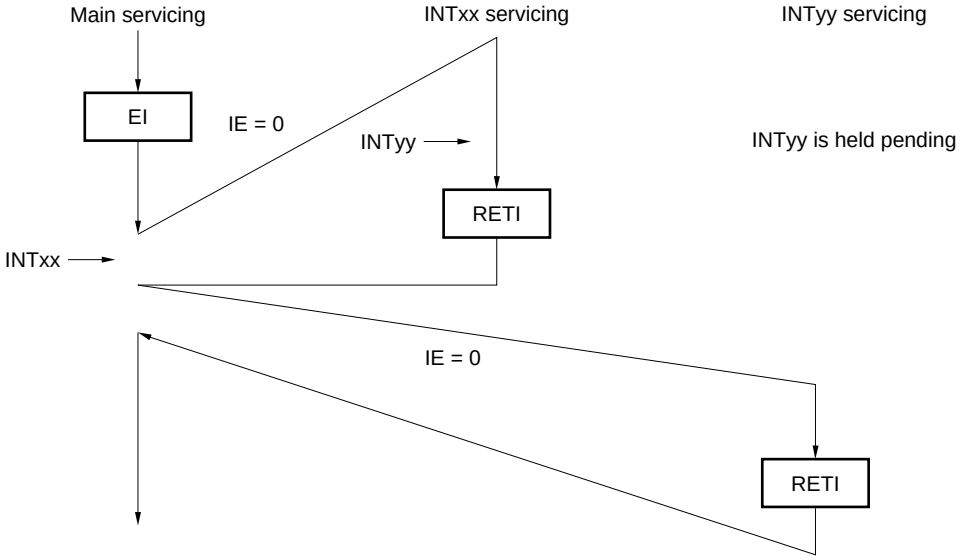
Figure 14-15. Example of Multiple Interrupts

Example 1. Acknowledging multiple interrupts



The interrupt request INTyy is acknowledged during the servicing of interrupt INTxx and multiple interrupts are performed. Before each interrupt request is acknowledged, the EI instruction is issued and the interrupt request is enabled.

Example 2. Multiple interrupts are not performed because interrupts are disabled



Because interrupt requests are disabled (the EI instruction has not been issued) in the interrupt INTxx servicing, the interrupt request INTyy is not acknowledged and multiple interrupts are not performed. INTyy is held pending and is acknowledged after INTxx servicing is completed.

IE = 0: Interrupt requests disabled

14.4.4 Putting interrupt requests on hold

If an interrupt request (such as a maskable, non-maskable, or external interrupt) is generated when a certain type of instruction is being executed, the interrupt request will not be acknowledged until the instruction is completed. Such instructions (interrupt request pending instructions) are as follows.

- Instructions that manipulate interrupt request flag registers 0, 1 (IF0 and IF1)
- Instructions that manipulate interrupt mask flag registers 0, 1 (MK0 and MK1)

[MEMO]

CHAPTER 15 STANDBY FUNCTION

15.1 Standby Function and Configuration

15.1.1 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 oscillator 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 oscillator and stops the entire system. The power consumption of the CPU can be substantially reduced in this mode.

The data memory can be retained at the low voltage (V_{DD} = 1.8 V). 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 oscillator 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.

15.1.2 Register controlling standby function

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

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

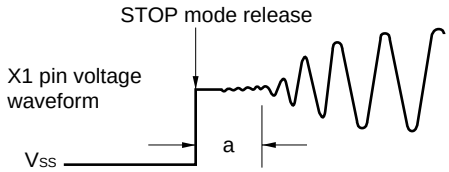
$\overline{\text{RESET}}$ input sets OSTS to 04H. However, it takes $2^{15}/f_X$, not $2^{17}/f_X$, after $\overline{\text{RESET}}$ input.

Figure 15-1. Format of Oscillation Stabilization Time Select Register

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

OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection
0	0	0	$2^{12}/f_X$ (819 μs)
0	1	0	$2^{15}/f_X$ (6.55 ms)
1	0	0	$2^{17}/f_X$ (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 whether STOP mode is released by $\overline{\text{RESET}}$ input or by interrupt generation.



Remarks 1. f_X: Main system clock oscillation frequency

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

15.2 Standby Function Operation

15.2.1 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 15-1. HALT Mode Operating Status

Item		HALT Mode Operation Status While The Main System Clock Is Running		HALT Mode Operation Status While The Subsystem Clock Is Running	
		While the subsystem clock is running	While the subsystem clock is not running	While the main system clock is running	While the main system clock is not running
Main system clock		Oscillation enabled			Oscillation stopped
CPU		Operation stopped			
Port (output latch)		Remains in the state existing before the selection of HALT mode.			
16-bit timer		Operation enabled			Operation stopped
8-bit timer	TM50	Operation enabled			Operation enabled[Note 1]
	TM60				Operation enabled[Note 2]
Watch timer		Operation enabled	Operation enabled[Note 3]	Operation enabled	Operation enabled[Note 4]
Watchdog timer		Operation enabled		Operation stopped	
Serial interface		Operation enabled			Operation stopped[Note 5]
A/D converter		Operation stopped			
LCD controller/driver		Operation enabled	Operation enabled[Note 3]	Operation enabled	Operation enabled[Note 4]
External interrupt		Operation enabled[Note 6]			

Notes 1. Operation is enabled only when input signal from timer 60 (timer 60 operation is enabled) is selected as the count clock.

2. Operation is enabled when TMI60 is selected as the count clock.

3. Operation is enabled while the main system clock is selected.

4. Operation is enabled while the subsystem clock is selected.

5. Operation is enabled only when external clock is selected.

6. Maskable interrupt that is not masked

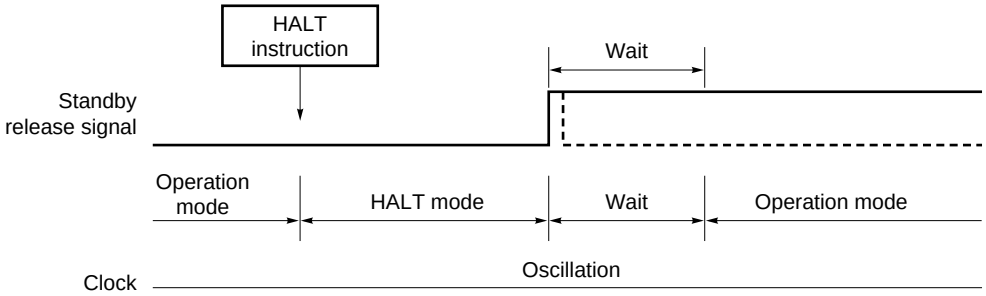
(2) Releasing HALT mode

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 is enabled to be acknowledged, vectored interrupt processing is performed. If the interrupt is disabled, the instruction at the next address is executed.

Figure 15-2. Releasing HALT Mode by Interrupt



Remarks 1. The broken line indicates 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

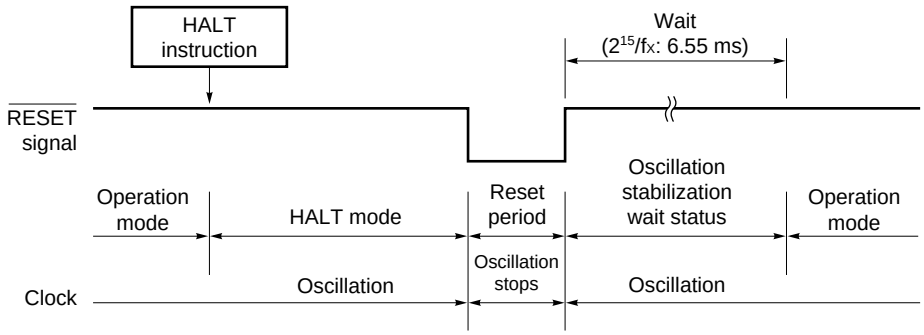
(b) Releasing by non-maskable interrupt request

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 15-3. Releasing HALT Mode by $\overline{\text{RESET}}$ Input



Remark fx: Main system clock oscillation frequency

Table 15-2. Operation After Releasing HALT Mode

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

x: don't care

15.2.2 STOP mode

(1) Setting and operation status of STOP mode

The STOP mode is set by executing the STOP instruction.

Caution 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 15-3. STOP Mode Operating Status

Item		STOP Mode Operation Status While The Main System Clock Is Running	
		While the subsystem clock is running	While the subsystem clock is not running
Main system clock		Oscillation stopped	
CPU		Operation stopped	
Port (output latch)		Remains in the state existing before the selection of STOP mode.	
16-bit timer		Operation stopped	
8-bit timer	TM50	Operation enabled[Note 1]	
	TM60	Operation enabled[Note 2]	
Watch timer		Operation enabled[Note 3]	Operation stopped
Watchdog timer		Operation enabled	Operation stopped
Serial interface		Operation enabled[Note 4]	
A/D converter		Operation stopped	
LCD controller/driver		Operation enabled[Note 3]	Operation stopped
External interrupt		Operation enabled[Note 5]	

Notes 1. Operation is enabled only when input signal from timer 60 (timer 60 operation is enabled) is selected as the count clock.

2. Operation is enabled when TMI60 is selected as the count clock.

3. Operation is enabled while the subsystem clock is selected.

4. Operation is enabled only when external clock is selected.

5. Maskable interrupt that is not masked

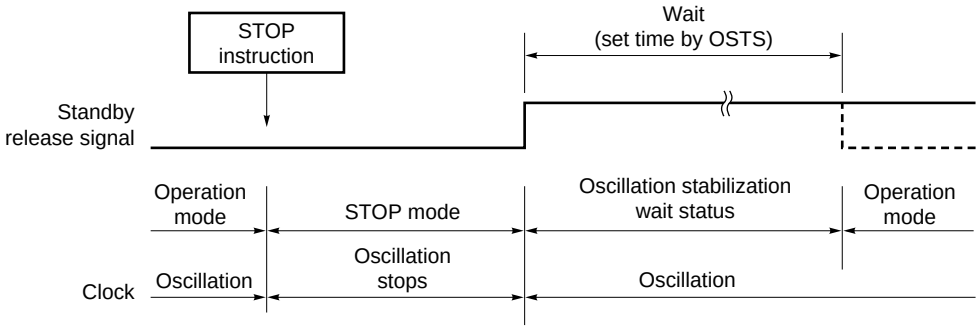
(2) Releasing STOP mode

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 enabled 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 15-4. Releasing STOP Mode by Interrupt

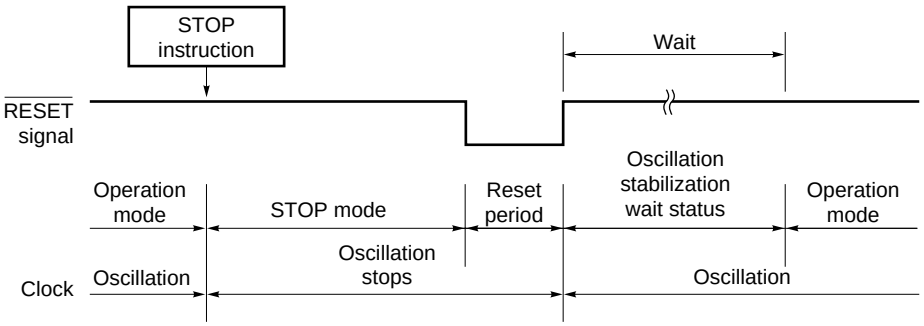


Remark The broken line indicates the case where the interrupt request that has released the standby mode is acknowledged.

(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 15-5. Releasing STOP Mode by $\overline{\text{RESET}}$ Input



Remark fx: Main system clock oscillation frequency

Table 15-4. Operation After Releasing STOP Mode

Releasing Source	MKxx	IE	Operation
Maskable interrupt request	0	0	Executes next address instruction
	0	1	Executes interrupt servicing
	1	x	Retains STOP mode
$\overline{\text{RESET}}$ input	–	–	Reset processing

x: don't care

CHAPTER 16 RESET FUNCTION

The following two operations are available to generate reset signals.

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

External and internal reset have no functional differences. In both cases, program execution starts at the address at 0000H and 0001H by $\overline{\text{RESET}}$ 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 is set to the status shown in Table 16-1. Each pin has a high impedance during reset input or during 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 16-2** to **16-4**.)

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 16-1. Block Diagram of Reset Function

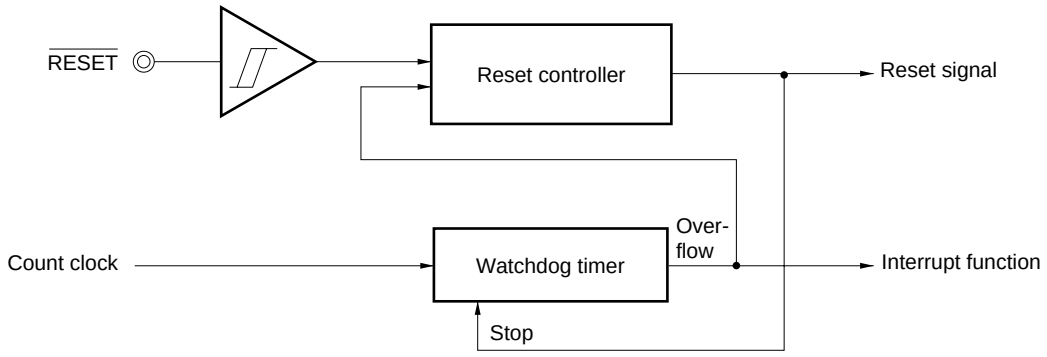


Figure 16-2. Reset Timing by $\overline{\text{RESET}}$ Input

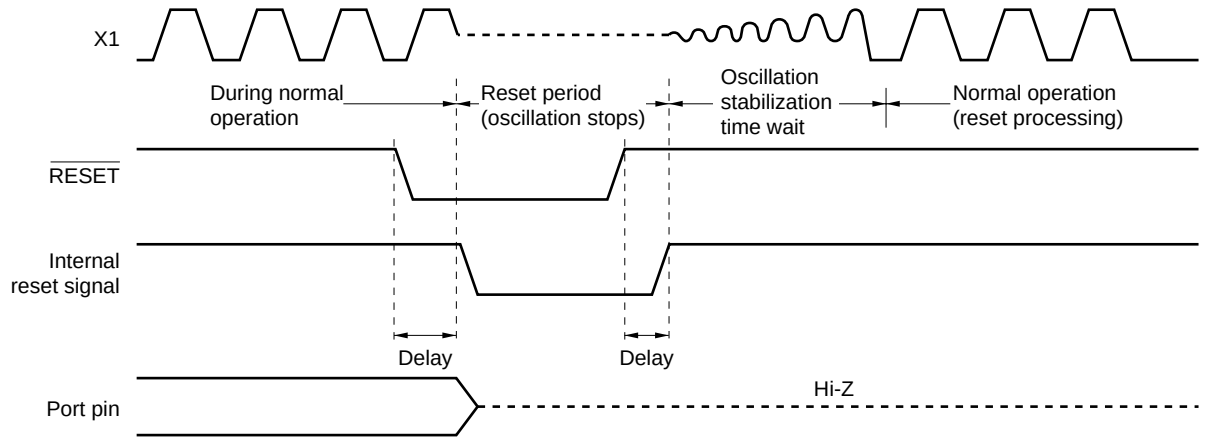


Figure 16-3. Reset Timing by Overflow in Watchdog Timer

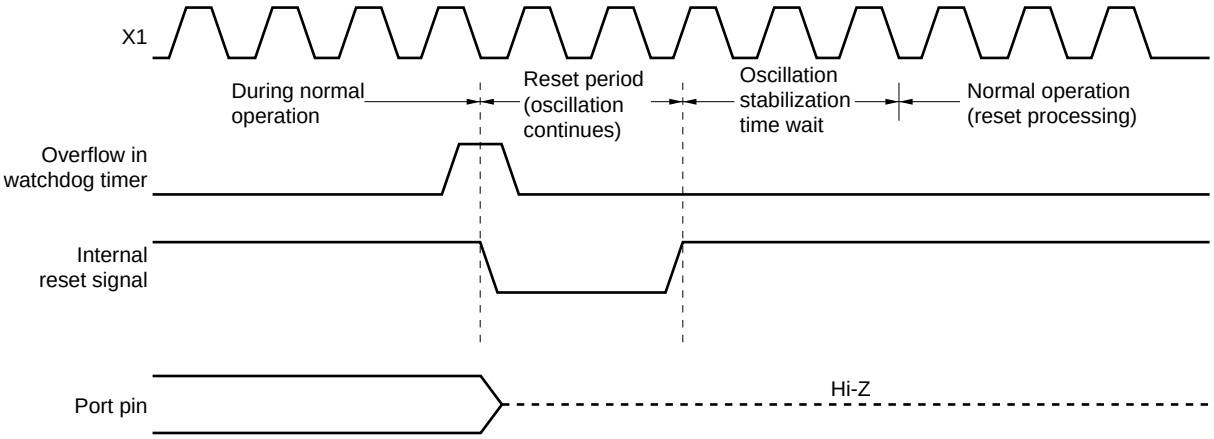


Figure 16-4. Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode

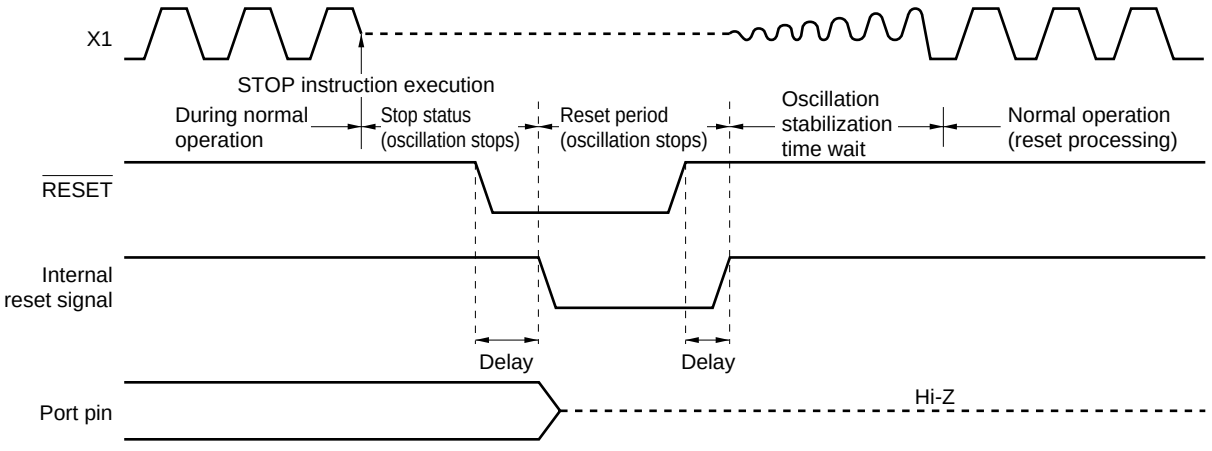


Table 16-1. Hardware Status After Reset (1/2)

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 P3, P5, P7) (Output latch)		00H
Port (P8, P9) (Output latch)[Note 3]		00H
Port mode register (PM0 to PM3, PM5, PM7)		FFH
Port mode register (PM8, PM9)[Note 3]		FFH
Pull-up resistor option register (PU0, PUB2, PUB3, PUB7)		00H
Pull-up resistor option register (PUB8[Note 3], PUB9[Note 3])		00H
Processor clock control register (PCC)		02H
Suboscillation mode register (SCKM)		00H
Subclock control register (CSS)		00H
Oscillation stabilization time select register (OSTS)		04H
16-bit timer	Timer counter (TM90)	0000H
	Compare register (CR90)	FFFFH
	Control register (TMC90)	00H
	Capture register (TCP90)	Undefined
8-bit timer	Timer counter (TM50, TM60)	00H
	Compare register (CR50, CR60, CRH60)	Undefined
	Mode control register (TMC50, TMC60)	00H
Watch timer	Mode control register (WTM)	00H
Watchdog timer	Clock select register (WDCS)	00H
	Mode register (WDTM)	00H
Serial interface	Serial operation mode register (CSIM20)	00H
	Asynchronous serial interface mode register (ASIM20)	00H
	Asynchronous serial interface status register (ASIS20)	00H
	Baud rate generator control register (BRGC20)	00H
	Transmit shift register (TXS20)	FFH
	Receive buffer register (RXB20)	Undefined
A/D converter	A/D conversion result register (ADCR0)	0000H
	Mode register (ADM0)	00H
	Analog input channel specification register (ADS0)	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. The post-reset values are retained in the standby mode.

3. μPD789426, 789436 Subseries only

Table 16-1. Hardware Status After Reset (2/2)

Hardware		Status After Reset
LCD controller/driver	Display mode register (LCDM0)	00H
	Clock control register (LCDC0)	00H
	Voltage amplification control register (LCDVA0)	00H
Interrupt	Request flag register (IF0, IF1)	00H
	Mask flag register (MK0, MK1)	FFH
	External interrupt mode register (INTM0, INTM1)	00H
	Key return mode register (KRM00)	00H

CHAPTER 17 μPD78F9436, 78F9456

The μPD78F9436 and 78F9456 are available as the flash memory versions of the μPD789426, 789436, 789446, and 789456 Subseries.

The μPD78F9436 is a version with the internal ROM of the μPD789426 and 789436 Subseries replaced with flash memory and the μPD78F9456 is a version with the internal ROM of the μPD789446 and 789456 Subseries replaced with flash memory. The differences between the μPD78F9436, 78F9456 and the mask ROM versions are shown in Table 17-1.

Table 17-1. Differences Between μPD78F9436, 78F9456 and Mask ROM Versions

Item \ Part Number		Flash Memory Version		Mask ROM Version			
		μPD78F9436	μPD78F9456	μPD789425, 789435	μPD789426, 789436	μPD789445, 789455	μPD789446, 789456
Internal memory	ROM	12 KB	16 KB	12 KB	16 KB	12 KB	16 KB
	High-speed RAM	512 bytes					
	LCD display RAM	5 bytes	15 bytes	5 bytes		15 bytes	
IC pin		Not provided		Provided			
V_{PP} pin		Provided		Not provided			
Electrical specifications		Varies depending on flash memory or mask ROM versions.					

Caution There are differences in noise immunity and noise radiation between the flash memory 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 version.

17.1 Flash Memory Programming

The on-chip program memory in the μPD78F9436 and 78F9456 is a flash memory.

The flash memory can be written with the μPD78F9436 and 78F9456 mounted on the target system (on-board). Connect the dedicated flash writer (Flashpro III (part no. 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.

17.1.1 Selecting communication mode

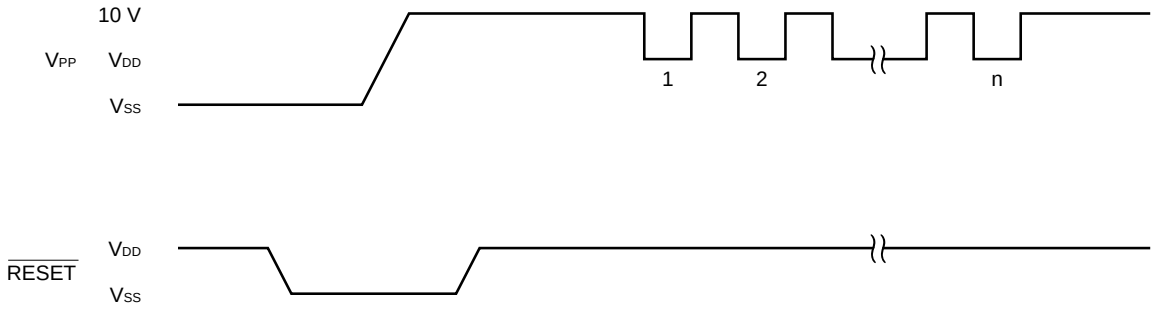
The flash memory is written by using Flashpro III and by means of serial communication. Select a communication mode from those listed in Table 17-2. To select a communication mode, the format shown in Figure 17-1 is used. Each communication mode is selected by the number of V_{PP} pulses shown in Table 17-2.

Table 17-2. Communication Mode

Communication Mode	Pins Used	Number of V_{PP} Pulses
3-wire serial I/O	$\overline{SCK20}$/ASCK20/P23 SO20/TxD20/P24 SI20/RxD20/P25	0
UART	TxD20/SO20/P24 RxD20/SI20/P25	8

Caution Be sure to select a communication mode depending on the number of V_{PP} pulses shown in Table 17-2.

Figure 17-1. Communication Mode Selection Format



17.1.2 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 17-3 shows the major functions of flash memory programming.

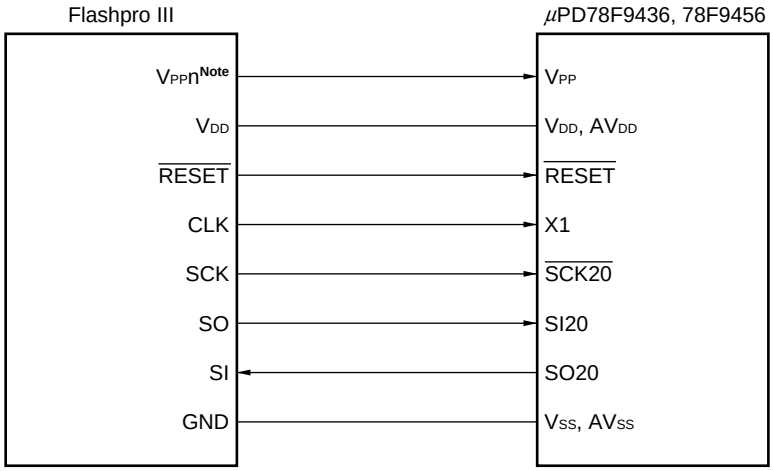
Table 17-3. Functions of Flash Memory Programming

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

17.1.3 Flashpro III connection example

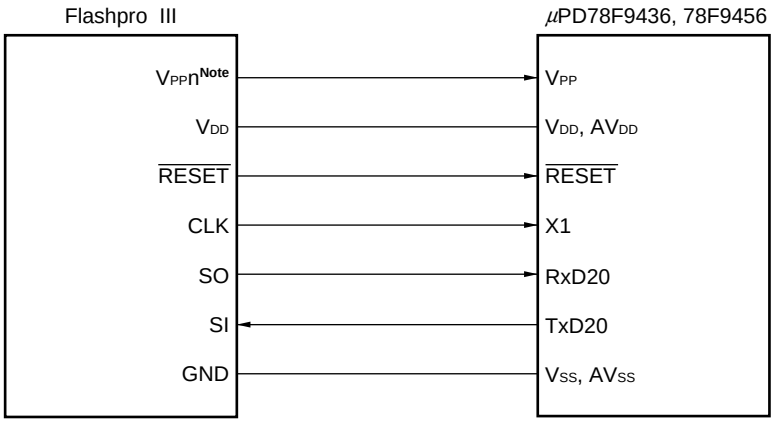
How the Flashpro III is connected to the μPD78F9436 or 78F9456 differs depending on the communication mode (3-wire serial I/O or UART). Figures 17-2 and 17-3 show the connection in the respective modes.

Figure 17-2. Flashpro III Connection Example in 3-Wire Serial I/O Mode



Note n = 1, 2

Figure 17-3. Flashpro III Connection Example in UART Mode



Note n = 1, 2

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

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

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

Table 17-4. Example of Settings for PG-FP3

Communication Mode	Example of Settings for PG-FP3		Number of V_{PP} Pulses[Note 1]
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	
UART	COMM PORT	UART-ch0	8
	CPU CLK	On Target Board	
	On Target Board	4.1943 MHz	
	UART BPS	9600 bps[Note 2]	

Notes 1. 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.

2. Select one of 9600 bps, 19200 bps, 38400 bps, or 76800 bps.

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

[MEMO]

CHAPTER 18 MASK OPTIONS

Table 18-1. Selection of Mask Option for Pins

Pin	Mask Option
P50 to P53	Whether a pull-up resistor is to be incorporated can be specified in 1-bit units.

For P50 to P53 (port 5), a mask option is used to specify whether a pull-up resistor is to be incorporated. The mask option is selectable in 1-bit units.

Caution Flash memory versions do not have a mask option-based on-chip pull-up resistor function.

[MEMO]

CHAPTER 19 INSTRUCTION SET

This chapter lists the instruction set of the μPD789426, 789436, 789446, and 789456 Subseries. For the details of the operation and machine language (instruction code) of each instruction, refer to **78K/0S Series Instructions User's Manual (U11047E)**.

19.1 Operation

19.1.1 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 functional 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 19-1. Operand Identifiers and Description Methods

Identifier	Description Method
r rp sfr	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7) AX (RP0), BC (RP1), DE (RP2), HL (RP3) Special-function register symbol
saddr saddrp	FE20H to FF1FH Immediate data or labels FE20H to FF1FH Immediate data or labels (even addresses only)
addr16 addr5	0000H to FFFFH Immediate data or labels (only even addresses for 16-bit data transfer instructions) 0040H to 007FH Immediate data or labels (even addresses only)
word byte bit	16-bit immediate data or label 8-bit immediate data or label 3-bit immediate data or label

Remark See **Table 3-4 Special Function Register List** for symbols of special function registers.

19.1.2 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
X_H, X_L:	Higher 8 bits and lower 8 bits of 16-bit register
∧:	Logical product (AND)
∨:	Logical sum (OR)
∀:	Exclusive logical sum (exclusive OR)
–:	Inverted data
addr16:	16-bit immediate data or label
jdisp8:	Signed 8-bit data (displacement value)

19.1.3 Description of “Flag” column

(Blank):	Unchanged
0:	Cleared to 0
1:	Set to 1
x:	Set/cleared according to the result
R:	Previously saved value is restored

19.2 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	x x x
	A, PSW	2	4	A ← PSW	
	PSW, A	2	4	PSW ← A	x x x
	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	x x x
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte	x x x
	A, r	2	4	A, CY ← A + r	x x x
	A, saddr	2	4	A, CY ← A + (saddr)	x x x
	A, !addr16	3	8	A, CY ← A + (addr16)	x x x
	A, [HL]	1	6	A, CY ← A + (HL)	x x x
	A, [HL+byte]	2	6	A, CY ← A + (HL + byte)	x x x
ADDC	A, #byte	2	4	A, CY ← A + byte + CY	x x x
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte + CY	x x x
	A, r	2	4	A, CY ← A + r + CY	x x x
	A, saddr	2	4	A, CY ← A + (saddr) + CY	x x x
	A, !addr16	3	8	A, CY ← A + (addr16) + CY	x x x
	A, [HL]	1	6	A, CY ← A + (HL) + CY	x x x
	A, [HL+byte]	2	6	A, CY ← A + (HL + byte) + CY	x x x
SUB	A, #byte	2	4	A, CY ← A – byte	x x x
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte	x x x
	A, r	2	4	A, CY ← A – r	x x x
	A, saddr	2	4	A, CY ← A – (saddr)	x x x
	A, !addr16	3	8	A, CY ← A – (addr16)	x x x
	A, [HL]	1	6	A, CY ← A – (HL)	x x x
	A, [HL+byte]	2	6	A, CY ← A – (HL + byte)	x x x

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	x	x	x
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte – CY	x	x	x
	A, r	2	4	A, CY ← A – r – CY	x	x	x
	A, saddr	2	4	A, CY ← A – (saddr) – CY	x	x	x
	A, !addr16	3	8	A, CY ← A – (addr16) – CY	x	x	x
	A, [HL]	1	6	A, CY ← A – (HL) – CY	x	x	x
	A, [HL+byte]	2	6	A, CY ← A – (HL + byte) – CY	x	x	x
AND	A, #byte	2	4	A ← A ∧ byte	x		
	saddr, #byte	3	6	(saddr) ← (saddr) ∧ byte	x		
	A, r	2	4	A ← A ∧ r	x		
	A, saddr	2	4	A ← A ∧ (saddr)	x		
	A, !addr16	3	8	A ← A ∧ (addr16)	x		
	A, [HL]	1	6	A ← A ∧ (HL)	x		
	A, [HL+byte]	2	6	A ← A ∧ (HL + byte)	x		
OR	A, #byte	2	4	A ← A ∨ byte	x		
	saddr, #byte	3	6	(saddr) ← (saddr) ∨ byte	x		
	A, r	2	4	A ← A ∨ r	x		
	A, saddr	2	4	A ← A ∨ (saddr)	x		
	A, !addr16	3	8	A ← A ∨ (addr16)	x		
	A, [HL]	1	6	A ← A ∨ (HL)	x		
	A, [HL+byte]	2	6	A ← A ∨ (HL + byte)	x		
XOR	A, #byte	2	4	A ← A ∀ byte	x		
	saddr, #byte	3	6	(saddr) ← (saddr) ∀ byte	x		
	A, r	2	4	A ← A ∀ r	x		
	A, saddr	2	4	A ← A ∀ (saddr)	x		
	A, !addr16	3	8	A ← A ∀ (addr16)	x		
	A, [HL]	1	6	A ← A ∀ (HL)	x		
	A, [HL+byte]	2	6	A ← A ∀ (HL + byte)	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
CMP	A, #byte	2	4	A – byte	x	x	x
	saddr, #byte	3	6	(saddr) – byte	x	x	x
	A, r	2	4	A – r	x	x	x
	A, saddr	2	4	A – (saddr)	x	x	x
	A, !addr16	3	8	A – (addr16)	x	x	x
	A, [HL]	1	6	A – (HL)	x	x	x
	A, [HL+byte]	2	6	A – (HL + byte)	x	x	x
ADDW	AX, #word	3	6	AX, CY ← AX + word	x	x	x
SUBW	AX, #word	3	6	AX, CY ← AX – word	x	x	x
CMPW	AX, #word	3	6	AX – word	x	x	x
INC	r	2	4	r ← r + 1	x	x	
	saddr	2	4	(saddr) ← (saddr) + 1	x	x	
DEC	r	2	4	r ← r + 1	x	x	
	saddr	2	4	(saddr) ← (saddr) – 1	x	x	
INCW	rp	1	4	rp ← rp + 1			
DECW	rp	1	4	rp ← rp – 1			
ROR	A, 1	1	2	(CY, $A_7 \leftarrow A_0$, $A_{m-1} \leftarrow A_m$) × 1			x
ROL	A, 1	1	2	(CY, $A_0 \leftarrow A_7$, $A_{m+1} \leftarrow A_m$) × 1			x
RORC	A, 1	1	2	(CY ← A_0, $A_7 \leftarrow$ CY, $A_{m-1} \leftarrow A_m$) × 1			x
ROLC	A, 1	1	2	(CY ← A_7, $A_0 \leftarrow$ CY, $A_{m+1} \leftarrow A_m$) × 1			x
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	x	x	x
	[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	x	x	x
	[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 ← $\overline{CY}$			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
CALL	!addr16	3	6	$(SP-1) \leftarrow (PC+3)_H$, $(SP-2) \leftarrow (PC+3)_L$, PC ← addr16, SP ← SP – 2	
CALLT	[addr5]	1	8	$(SP-1) \leftarrow (PC+1)_H$, $(SP-2) \leftarrow (PC+1)_L$, $PC_H \leftarrow$ (00000000, addr5 + 1), $PC_L \leftarrow$ (00000000, addr5), SP ← SP – 2	
RET		1	6	$PC_H \leftarrow (SP+1)$, $PC_L \leftarrow (SP)$, SP ← SP + 2	
RETI		1	8	$PC_H \leftarrow (SP+1)$, $PC_L \leftarrow (SP)$, PSW ← (SP + 2), SP ← SP + 3, NMIS ← 0	R R R
PUSH	PSW	1	2	(SP – 1) ← PSW, SP ← SP – 1	
	rp	1	4	$(SP-1) \leftarrow rp_H$, $(SP-2) \leftarrow rp_L$, SP ← SP – 2	
POP	PSW	1	4	PSW ← (SP), SP ← SP + 1	R R R
	rp	1	6	$rp_H \leftarrow (SP+1)$, $rp_L \leftarrow (SP)$, 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	$PC_H \leftarrow A$, $PC_L \leftarrow X$	
BC	$saddr16	2	6	PC ← PC + 2 + jdisp8 if CY = 1	
BNC	$saddr16	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).

19.3 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

2nd Operand / 1st 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 development of systems using the μPD789426, 789436, 789446, and 789456 Subseries.

Figure A-1 shows development tools.

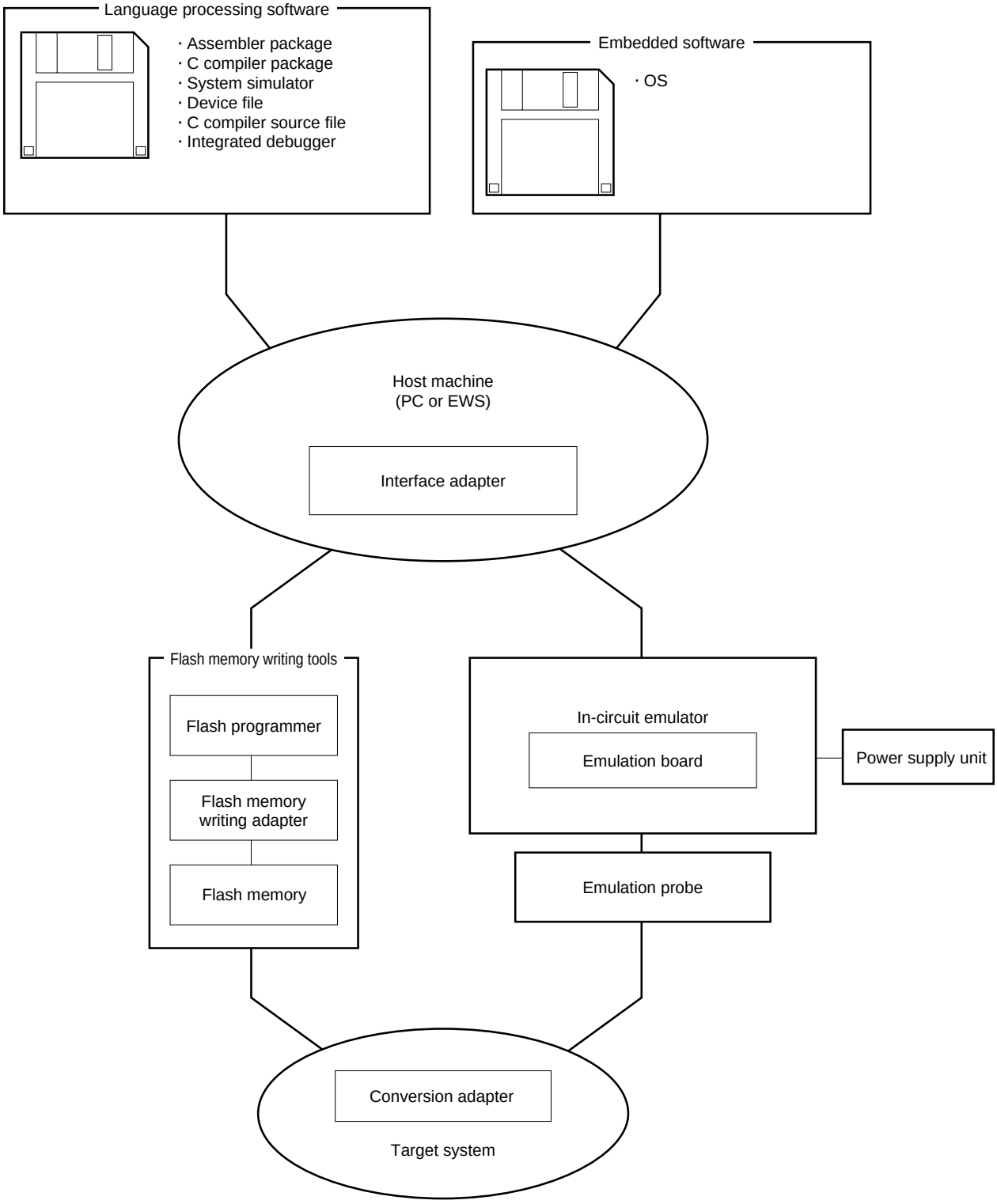
- Support to PC98-NX Series

  Unless specified otherwise, the products supported by IBM PC/AT™ compatibles can be used in PC98-NX Series. When using the PC98-NX Series, refer to the explanation of IBM PC/AT compatibles.

- Windows

  Unless specified otherwise, “Windows” indicates the following operating systems.
  - Windows 3.1
  - Windows 95
  - Windows NT™ Ver.4.0

Figure A-1. Development Tools



A.1 Language Processing Software

RA78K0S Assembler package	Program that converts program written in mnemonic into object codes that can be executed by microcontroller. In addition, automatic functions to generate symbol table and optimize branch instructions are also provided. Used in combination with optional device file (DF789456). **<Caution when used under PC environment>** The assembler package is a DOS-based application but may be used under the Windows environment by using Project Manager of Windows (included in the assembler package).
	Part number: μS××××RA78K0S
CC78K0S C compiler package	Program that converts program written in C language into object codes that can be executed by microcontroller. Used in combination with optional assembler package (RA78K0S) and device file (DF789316). **<Caution when used under PC environment>** The C compiler package is a DOS-based application but may be used under the Windows environment by using Project Manager of Windows (included in the assembler package).
	Part number: μS××××CC78K0S
DF789456[Note] Device file	File containing the information inherent to the device. Used in combination with optional RA78K0S, CC78K0S, and SM78K0S.
	Part number: μS××××DF789456
CC78K0S-L C compiler source file	Source file of functions for generating object library included in C compiler package. Necessary for changing object library included in 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 DF789456 is a common file that can be used with RA78K0S, CC78K0S, and SM78K0S.

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

μS××××RA78K0S
μS××××CC78K0S
μS××××DF789456
μS××××CC78K0S-L

××××	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
BB13		English Windows[Note]	
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 under the DOS environment.

A.2 Flash Memory Writing Tools

Flashpro III (Part No. FL-PR3, PG-FP3) Flash programmer	Dedicated flash programmer for microcomputers incorporating flash memory
FA-64GK Flash memory writing adapter	Adapter for writing to flash memory and connected to Flashpro III. • FA-64GK: for 64-pin plastic TQFP (fine pitch) (GK-9ET type)

Remark The FL-PR3 and FA-64GK 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 hardware and software of application system using 78K/0S Series. Supports integrated debugger (ID78K0S-NS). Used in combination with AC adapter, emulation probe, and interface adapter for connecting the host machine.
IE-70000-MC-PS-B AC adapter		Adapter for supplying power from AC 100 to 240 V outlet.
IE-70000-98-IF-C Interface adapter		Adapter necessary when using PC-9800 series PC (except notebook type) as host machine of IE-78K0S-NS (C bus supported)
IE-70000-CD-IF-A PC card interface		PC card and interface cable necessary when using notebook PC as host machine of IE-78K0S-NS (PCMCIA socket supported)
IE-70000-PC-IF-C Interface adapter		Interface adapter necessary when using IBM PC/AT compatible as host machine of IE-78K0S-NS (ISA bus supported)
IE-70000-PCI-IF Interface adapter		Adapter necessary when using personal computer incorporating PCI bus as host machine of IE-78K0S-NS
IE-789456-NS-EM1 Emulation board		Board for emulating the peripheral hardware inherent to the device. Used in combination with in-circuit emulator.
NP-64GK Emulation probe		Probe to connect the in-circuit emulator and target system. Used in combination with the TGK-064SBW and TGK-064SBP.
	TGK-064SBW, TGK-064SBP Conversion adapter	Conversion socket to connect the NP-64GK and a target system board on which a 64-pin plastic TQFP (fine pitch) (GK-9ET type) can be mounted

Remarks 1. The NP-64GK is a product made by Naito Densei Machida Mfg. Co., Ltd. (TEL +81-44-822-3813).

2. The TGK-064SBW and TGK-064SBP are products made by TOKYO ELETECH CORPORATION.
For further information, contact: Daimaru Kogyo, Ltd.
Tokyo Electronics Department (TEL +81-3-3820-7112)
Osaka Electronics Department (TEL +81-6-6244-6672)

A.3.2 Software

ID78K0S-NS Integrated debugger (Supports in-circuit emulator IE-78K0S-NS)	Control program for debugging 78K/0S Series. This program provides a graphical use 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 source program, disassembled display, and memory display, to the trace result. This software also allows users to add other function extension modules such as task debugger and system performance analyzer to improve the debug efficiency for programs using a real-time operating system. Used in combination with optional device file (DF789456).
	Part number: μS××××ID78K0S-NS

Remark ×××× in the part number differs depending on the host machines and operating systems 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
BB13		English Windows[Note]	

Note Also operates under the DOS environment.

SM78K0S System simulator	Debugs program at C source level or assembler level while simulating operation of target system on host machine. SM78K0S runs under 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 optional device file (DF789456).
	Part number: μS××××SM78K0S
DF789456[Note] Device file	File containing the information inherent to the device. Used in combination with the optional RA78K0S, CC78K0S, and SM78K0S.
	Part number: μS××××DF789456

Note DF789456 is a common file that can be used with RA78K0S, CC78K0S, and SM78K0S.

Remark ×××× in the part number differs depending on the host machines and operating systems 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
BB13		English Windows[Note]	

Note Also operates under the DOS environment.

APPENDIX B EMBEDDED SOFTWARE

The following embedded software is provided to perform the program development and maintenance of the μPD789426, 789436, 789446, and 789456 Subseries effectively.

MX78K0S OS	Subset OS conformed to μITRON. Includes the nucleus of the MX78K0S. Task control, event control, and time control are performed. In the task control, task execution sequences are controlled to switch the task to be executed next. **<Caution when used under PC environment>** The MX78K0S is a DOS-based application. Use this software in the DOS prompt when running it on Windows.
	Part number: μS××××MX78K0S

Remark ×××× in the part number differs depending on the host machines and operating systems to be 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" 2HC FD
BB13		English Windows[Note]	

Note Also operates under the DOS environment.

[MEMO]

APPENDIX C REGISTER INDEX

C.1 Register Index (Alphabetic Order of Register Name)

C.2 Register Index (Alphabetic Order of Register Symbol)

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	❑	❑	❑	❑