NEC

User's Manual

μPD789167, 789177, 789167Y, 789177Y Subseries

8-Bit Single-Chip Microcontrollers

μPD789166 μPD789166Y
μPD789167 μPD789167Y
μPD789176 μPD789176Y
μPD789177 μPD789177Y
μPD78F9177 μPD78F9177Y

Document No. U14186EJ2V0UM00 (2nd edition)
Date Published January 2000 J CP(K)

[MEMO]

SUMMARY OF CONTENTS

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

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

License not needed:	μPD78F9177, μPD78F9177Y
The customer must judge the need for license:	μPD789166, μPD789167, μPD789176, μPD789177 μPD789166Y, μPD789167Y, μPD789176Y, μPD789177Y

Caution This product contains an I^2C bus interface circuit.
When using the I^2C bus interface, notify its use to NEC when ordering custom code. NEC can guarantee the following only when the customer informs NEC of the use of the interface:
Purchase of NEC I^2C components conveys a license under the Philips I^2C Patent Rights to use these components in an I^2C system, provided that the system conforms to the I^2C Standard Specification as defined by Philips.

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

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

M7D 98. 12

Regional Information

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

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

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

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

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

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

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

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

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

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 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 1130
Tel: 65-253-8311
Fax: 65-250-3583

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

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

J99.1

MAJOR REVISIONS IN THIS EDITION

Page	Description
Throughout	Addition of description of μPD789166Y, μPD789167Y, μPD789176Y, and μPD789177Y
	Change of status of μPD789166, μPD789167, μPD789176, and μPD789177 from "under development" to "developed"
p.78	Addition of description of SMB0 special function registers in Table 5-3 Special Function Registers
p.94	Modification of Figure 6-5 Block Diagram of P21
p.135	Addition of 8.5 Notes on Using 16-Bit Timer
p.231	Addition of 15 SMB0 (μPD789167Y AND 789177Y SUBSERIES)
p.295	Addition of description of SMB0 interrupt to 17 INTERRUPT FUNCTIONS
p.326	Addition of Figure 20-3 Flashpro III Connection in SMB Mode
p.327	Addition of setting with SMB mode to Table 20-4 Setting with PG-FB3
p.344	Addition of development tools for μPD789166Y, μPD789167Y, μPD789176Y, and μPD789177Y to A DEVELOPMENT TOOLS

The mark ★ shows major revised points.

[MEMO]

INTRODUCTION

Readers

This manual is intended for user engineers who understand the functions of the μPD789167, 789177, 789167Y, and 789177Y Subseries to design and develop its application systems and programs.

Target products:

- μPD789167 Subseries: μPD789166 and μPD789167
- μPD789177 Subseries: μPD789176, μPD789177, and μPD78F9177
- μPD789167Y Subseries: μPD789166Y and μPD789167Y
- μPD789177Y Subseries: μPD789176Y, μPD789177Y, and μPD78F9177Y

Purpose

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

Organization

The μPD789167, 789177, 789167Y, 789177Y Subseries manual is divided into two parts: this manual and the instruction (common to the 78K/0S Series).

μPD789167, 789177, 789167Y, 789177Y Subseries User's Manual (This manual)	78K/0S Series User's Manual Instruction
• Pin functions • Internal block functions • Interrupt • Other internal peripheral functions	• CPU function • Instruction set • Instruction description

How to Read This Manual

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

◊ To understand the overall functions of the μPD789167, 789177, 789167Y, and 789177Y Subseries
→ Read this manual in the order of the **CONTENTS**.

◊ How to read register formats
→ The name of a bit whose number is enclosed with < > 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**.

◊ To learn the details of the instruction functions of the 78K/0S Series
→ Refer to **78K/0S Series User's Manual — Instructions (U11047E)** separately available.

Differences between μPD789167, 789177, 789167Y, and 789177Y Subseries

The μPD789167, 789177, 789167Y, and 789177Y Subseries differ in their package type, A/D converter resolution, and serial interface configuration.

Item \ Subseries		μPD789167	μPD789177	μPD789167Y	μPD789177Y
Package		• 44-pin plastic LQFP		• 44-pin plastic LQFP • 48-pin plastic TQFP	
IC2 pin		Not provided		Provided	
A/D converter resolution		8 bits	10 bits	8 bits	10 bits
Serial interface configuration	3-wire serial I/O mode	1 channel			
	SMB0	Not provided		1 channel	

Configuration of This Manual This manual uses separate chapters to describe those functions that vary with the subseries. The chapters related to each subseries are listed below.

For information about a certain subseries, see only the chapters indicated by checkmarks in its column.

Chapter	μPD789167 Subseries	μPD789177 Subseries	μPD789167Y Subseries	μPD789177Y Subseries
CHAPTER 1 GENERAL (μPD789167 AND 789177 SUBSERIES)	√	√	–	–
CHAPTER 2 GENERAL (μPD789167Y AND 789177Y SUBSERIES)	–	–	√	√
CHAPTER 3 PIN FUNCTIONS (μPD789167 AND 789177 SUBSEREIS)	√	√	–	–
CHAPTER 4 PIN FUNCTIONS (μPD789167Y AND 789177Y SUBSEREIS)	–	–	√	√
CHAPTER 5 CPU ARCHITECTURE	√	√	√	√
CHAPTER 6 PORT FUNCTIONS	√	√	√	√
CHAPTER 7 CLOCK GENERATION CIRCUIT	√	√	√	√
CHAPTER 8 16-BIT TIMER	√	√	√	√
CHAPTER 9 8-BIT TIMER/EVENT COUNTERS	√	√	√	√
CHAPTER 10 WATCH TIMER	√	√	√	√
CHAPTER 11 WATCHDOG TIMER	√	√	√	√
CHAPTER 12 8-BIT A/D CONVERTER (μPD789167 AND 789167Y SUBSERIES)	√	–	√	–
CHAPTER 13 10-BIT A/D CONVERTER (μPD789177 AND 789177Y SUBSERIES)	–	√	–	√
CHAPTER 14 SERIAL INTERFACE 20	√	√	√	√
CHAPTER 15 SMB0 (μPD789167Y AND 789177Y SUBSERIES)	–	–	√	√
CHAPTER 16 MULTIPLIER	√	√	√	√
CHAPTER 17 INTERRUPT FUNCTIONS	√	√	√	√
CHAPTER 18 STANDBY FUNCTION	√	√	√	√
CHAPTER 19 RESET FUNCTION	√	√	√	√
CHAPTER 20 μPD78F9177 AND μPD78F9177Y	√	√	√	√
CHAPTER 21 MASK OPTION	√	√	√	√
CHAPTER 22 INSTRUCTION SET	√	√	√	√

Conventions

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

Related Documents

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

Documents Related to Devices

Document Name	Document No.	
	English	Japanese
μPD789166, 789167, 789176, 789177 Data Sheet	To be prepared	U14017J
μPD78F9177 Preliminary Product Information	U14022E	U14022J
μPD789166Y, 789167Y, 789176Y, 789177Y Data Sheet	To be prepared	To be prepared
μPD78F9177Y Data Sheet	To be prepared	To be prepared
μPD789167, 789177, 789167Y, 789177Y Subseries User's Manual	This manual	U14186J
78K/0S Series User's Manual — Instructions	U11047E	U11047J
Preliminary Application Note 78K/0, 78K/0S Series Flash Memory Writing	To be prepared	U14458J

Documents Related to Development Tools (User's Manuals)

Document Name		Document No.	
		English	Japanese
RA78K0S Assembler Package	Operation	U11622E	U11622J
	Assembly Language	U11599E	U11599J
	Structured Assembly Language	U11623E	U11623J
CC78K/0S C Compiler	Operation	U11816E	U11816J
	Language	U11817E	U11817J
SM78K0S System Simulator Windows™ Based	Reference	U11489E	U11489J
SM78K Series System Simulator	External Parts User Open Interface Specifications	U10092E	U10092J
ID78K0S-NS Integrated Debugger Windows Based	Reference	U12901E	U12901J
IE-78K0S-NS		U13549E	U13549J
IE-789177-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 documents for designing, etc.

Documents Related to Embedded Software (User's Manuals)

Document Name		Document No.	
		English	Japanese
78K/0S Series OS MX78K0S	Basics	U12938E	U12938J

Other Related Documents

Document Name	Document No.	
	English	Japanese
SEMICONDUCTORS SELECTION GUIDE Product & Packages (CD-ROM)	X13769X	
Semiconductor Device Mounting Technology Manual	C10535E	C10535J
Quality Grades on NEC Semiconductor Device	C11531E	C11531J
NEC Semiconductor Device Reliability/Quality Control System	C10983E	C10983J
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892E	C11892J
Semiconductor Device Quality Control/Reliability Handbook	–	C12769J
Guide for Products Related to Micro-Computer: Other Companies	–	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.

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 (1/3)

LIST OF TABLES (2/3)

LIST OF TABLES (3/3)

[MEMO]

CHAPTER 1 GENERAL (μPD789167 AND 789177 SUBSERIES)

1.1 Features

• ROM and RAM capacity

Item / Product Name	Program Memory (ROM)		Data Memory (Internal High-Speed RAM)
μPD789166, μPD789176	Mask ROM	16 Kbytes	512 bytes
μPD789167, μPD789177		24 Kbytes	
μPD78F9177	Flash memory	24 Kbytes	

- Minimum instruction execution time changeable from high-speed (0.4 μs: Main system clock 5.0-MHz operation) to ultra-low speed (122 μs: Subsystem clock 32.768-kHz operation)
- I/O port: 31
- Serial interface
  - 3-wire serial I/O mode/UART mode: 1 channel
- 8-bit resolution A/D converter: 8 channels (μPD789167 Subseries)
- 10-bit resolution A/D converter: 8 channels (μPD789177 Subseries)
- Timer: 6 channels
  - 16-bit timer: 1 channel
  - 8-bit timer/event counter: 2 channels
  - 8-bit timer: 1 channel
  - Watch timer: 1 channel
  - Watchdog timer: 1 channel
- Vectored interrupt source: 15
- Supply voltage: V_{DD} = 1.8 to 5.5 V
- Operating ambient temperature: T_A = –40°C to +85°C

1.2 Applications

Power windows, keyless entry, battery management units, side air bags, etc.

1.3 Ordering Information

Part Number	Package	Internal ROM
μPD789166GB-×××-8ES	44-pin plastic LQFP (10 × 10 mm)	Mask ROM
μPD789167GB-×××-8ES	44-pin plastic LQFP (10 × 10 mm)	Mask ROM
μPD789176GB-×××-8ES	44-pin plastic LQFP (10 × 10 mm)	Mask ROM
μPD789177GB-×××-8ES	44-pin plastic LQFP (10 × 10 mm)	Mask ROM
μPD78F9177GB-8ES	44-pin plastic LQFP (10 × 10 mm)	Flash memory

Remark ××× indicates ROM code suffix.

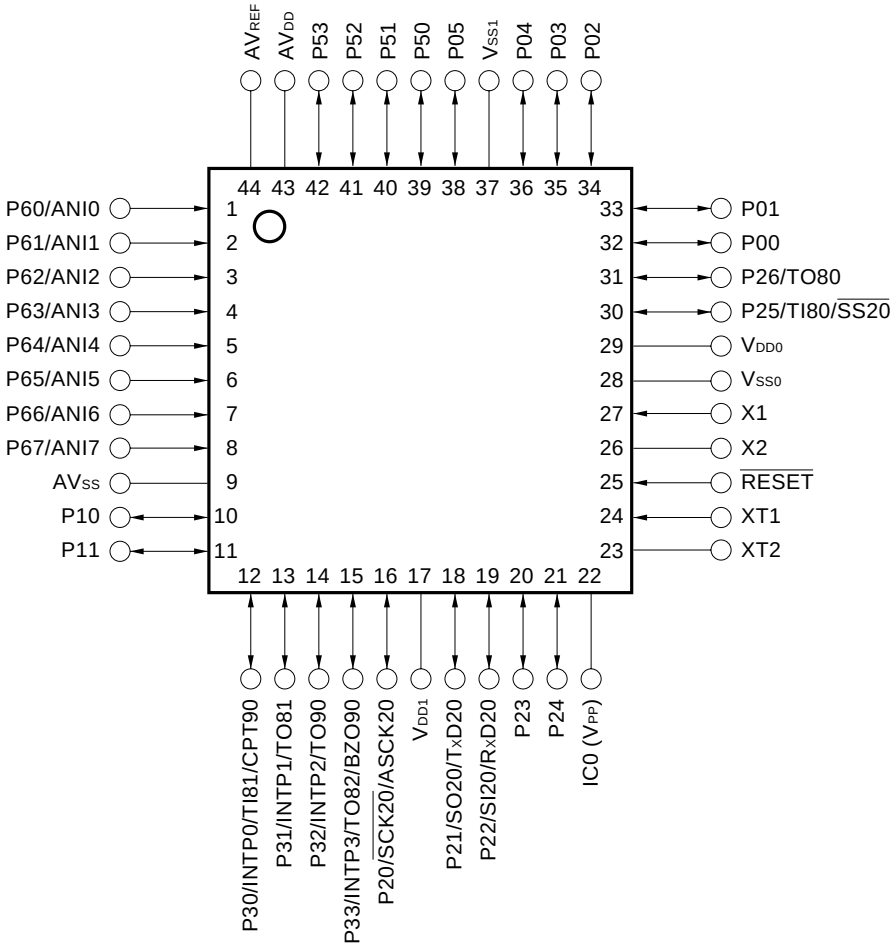
1.4 Pin Configuration (Top View)

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

  μPD789166GB-×××-8ES μPD789176GB-×××-8ES

  μPD789167GB-×××-8ES μPD789177GB-×××-8ES

  μPD78F9177GB-8ES



Cautions 1. Connect the IC0 (internally connected) pin directly to the V_{SS0} or V_{SS1} pin.

2. Connect the AV_{DD} pin to the V_{DD0} pin.

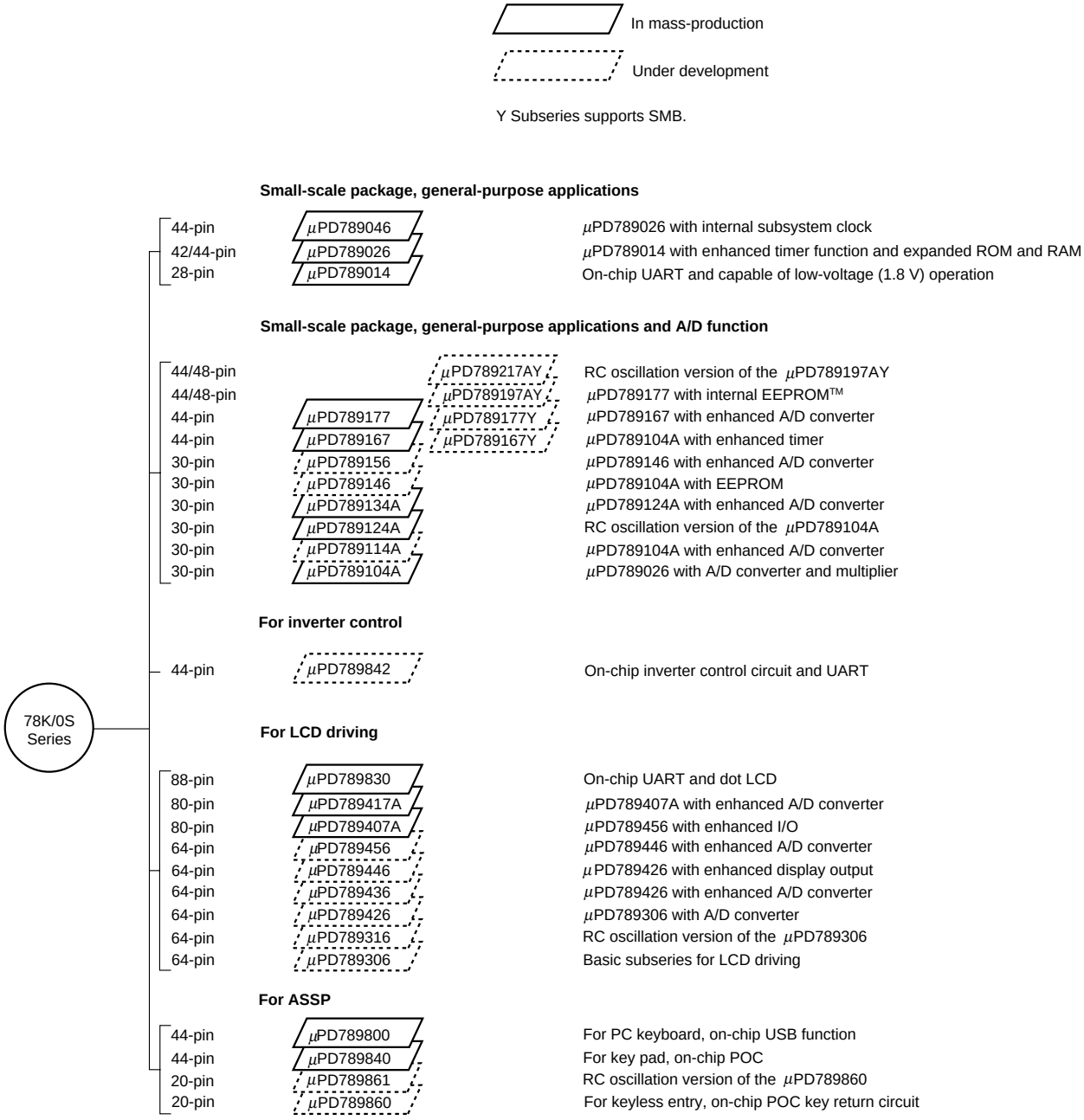
3. Connect the AV_{SS} pin to the V_{SS0} pin.

Remark Pin connections in parentheses are intended for the μPD78F9177.

ANI0 to ANI7:	Analog Input	$\overline{\text{RESET}}$:	Reset
ASCK20:	Asynchronous Serial Input	RxD20:	Receive Data
AV_{DD}:	Analog Power Supply	$\overline{\text{SCK20}}$:	Serial Clock
AV_{REF}:	Analog Reference Voltage	SI20:	Serial Input
AV_{SS}:	Analog Ground	SO20:	Serial Output
BZO90:	Buzzer Output	$\overline{\text{SS20}}$:	Chip Select Input
CPT90:	Capture Trigger Input	TI80, TI81:	Timer Input
IC0:	Internally Connected	TO80 to TO82, TO90:	Timer Output
INTP0 to INTP3:	Interrupt from Peripherals	TxD20:	Transmit Data
P00 to P05:	Port 0	V_{DD0}, V_{DD1}:	Power Supply
P10, P11:	Port 1	V_{PP}:	Programming Power Supply
P20 to P26:	Port 2	V_{SS0}, V_{SS1}:	Ground
P30 to P33:	Port 3	X1, X2:	Crystal (Main System Clock)
P50 to P53:	Port 5	XT1, XT2:	Crystal (Subsystem Clock)
P60 to P67:	Port 6		

1.5 78K/0S Series Development

The 78K/0S Series products are shown below. The subseries names are indicated in frames.

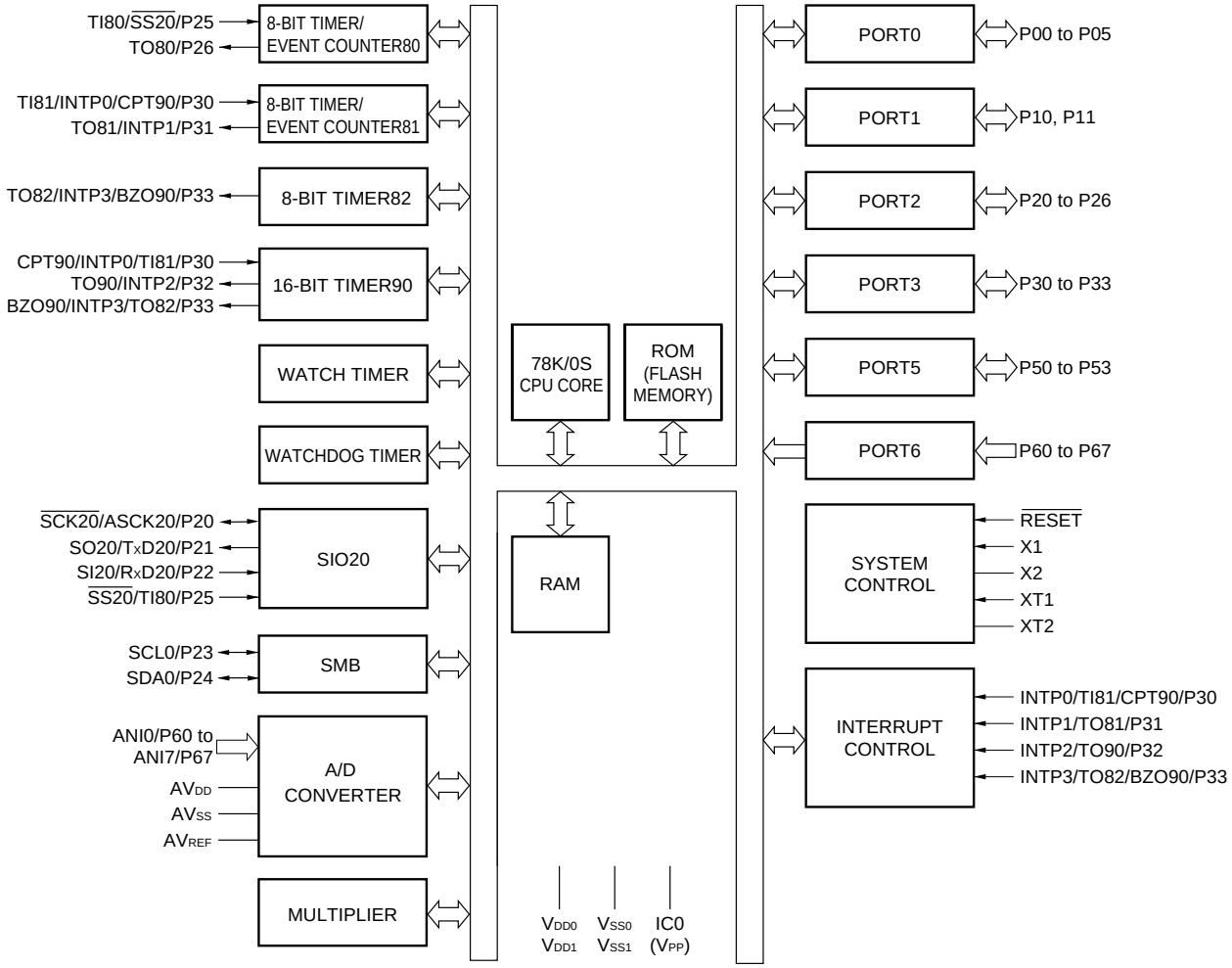


The major functional differences among the subseries are listed below.

Subseries Name \ Function		ROM Capacity	Timer				8-Bit A/D	10-Bit A/D	Serial Interface	I/O (pins)	V_{DD} MIN. Value	Remark
			8-Bit	16-Bit	Watch	WDT						
Small-scale package, general-purpose application	μPD789046	16K	1 ch	1 ch	1 ch	1 ch	–	–	1 ch (UART: 1 ch)	34	1.8 V	–
	μPD789026	4K to 16K			–							
	μPD789014	2K to 4K	2 ch	–						22		
Small-scale package, general-purpose application + 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	8K to 16K	1 ch		–		–	4 ch		20		On-chip EEPROM
	μPD789146						4 ch	–				
	μPD789134A	2K to 8K					–	4 ch				RC oscillation version
	μPD789124A						4 ch	–				
	μPD789114A						–	4 ch				–
	μPD789104A						4 ch	–				
Inverter control	μPD789842	8K to 16K	3 ch	**Note**	1 ch	1 ch	8 ch	–	1 ch (UART: 1 ch)	30	4.0 V	–
LCD driving	μPD789830	24 K	1 ch	1 ch	1 ch	1 ch	–	–	1 ch (UART: 1 ch)	30	2.7 V	–
	μPD789417A	12 K to 24 K	3 ch					7 ch		43	1.8 V	
	μPD789407A						7 ch	–				
	μPD789456	12 K to 16 K	2 ch				–	6 ch	1 ch (UART: 1 ch)	30		
	μPD789446						6 ch	–				
	μPD789436						–	6 ch		40		
	μPD789426						6 ch	–				
	μPD789316	8 K to 16 K					–		2 ch (USB: 1ch)	23		RC oscillation version
	μPD789306											–
ASSP	μPD789800	8 K	2 ch	1 ch	–	1 ch	–	–	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
	μPD789860											–

Note 10-bit timer: 1 channel

1.6 Block Diagram



Remarks 1. The size of the internal ROM varies depending on the model.

2. Pin connections in parentheses are intended for the μPD78F9177.

1.7 Outline of Function

Item \ Part Number		μPD789166 μPD789176	μPD789167 μPD789177	μPD78F9177
Internal memory	ROM	Mask ROM		Flash Memory
		16 Kbytes	24 Kbytes	24 Kbytes
	High-speed RAM	512 bytes		
Minimum instruction execution time		• 0.4/1.6 μs (operation with main system clock running at 5.0 MHz) • 122 μs (operation with subsystem clock running at 32.768 kHz)		
General-purpose registers		8 bits × 8 registers		
Instruction set		• 16-bit operations • Bit manipulations (such as set, reset, and test)		
Multiplier		8 bits × 8 bits = 16 bits		
I/O ports		Total: 31 • CMOS input: 8 • CMOS I/O: 17 • N-ch open-drain: 6		
A/D converter		• 8-bit resolution × 8 channels (μPD789167 Subseries) • 10-bit resolution × 8 channels (μPD789177 Subseries)		
Serial interface		• Switchable between 3-wire serial I/O and UART modes: 1 channel		
Timers		• 16-bit timer: 1 channel • 8-bit timer/event counter: 2 channels • 8-bit timer: 1 channel • Watch timer: 1 channel • Watchdog timer: 1 channel		
Timer output		Four outputs		
Buzzer output		One output		
Vectored interrupt sources	Maskable	Internal: 10, external: 4		
	Nonmaskable	Internal: 1		
Power supply voltage		V_{DD} = 1.8 to 5.5 V		
Operating ambient temperature		T_A = –40°C to +85°C		
Package		44-pin plastic LQFP (10 × 10 mm)		

The timers are outlined below.

		16-Bit Timer 90	8-Bit Timer/Event Counter 80	8-Bit Timer/Event Counter 81	8-Bit Timer 82	Watch Timer	Watchdog Timer
Operating mode	Interval timer	–	1 channel	1 channel	1 channel	1 channel[Note 1]	1 channel[Note 2]
	External event counter	–	1 channel	1 channel	–	–	–
Function	Timer output	1 output	1 output	1 output	1 output	–	–
	PWM output	–	1 output	1 output	1 output	–	–
	Square-wave output	–	1 output	1 output	1 output	–	–
	Buzzer output	1 output	–	–	–	–	–
	Capture	1 input	–	–	–	–	–
	Interrupt source	1	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 provides the watchdog timer function and interval timer function. Use either of the functions.

[MEMO]

★ CHAPTER 2 GENERAL (μPD789167Y AND 789177Y SUBSERIES)

2.1 Features

• ROM and RAM capacity

Item / Product Name	Program Memory (ROM)		Data Memory (Internal High-Speed RAM)
μPD789166Y, μPD789176Y	Mask ROM	16 Kbytes	512 bytes
μPD789167Y, μPD789177Y		24 Kbytes	
μPD78F9177Y	Flash memory	24 Kbytes	

- Minimum instruction execution time changeable from high-speed (0.4 μs: Main system clock 5.0-MHz operation) to ultra-low speed (122 μs: Subsystem clock 32.768-kHz operation)
- I/O port: 31
- Serial interface: 2 channels
  - 3-wire serial I/O mode/UART mode: 1 channel
  - SMB: 1 channel
- 8-bit resolution A/D converter: 8 channels (μPD789167Y Subseries)
- 10-bit resolution A/D converter: 8 channels (μPD789177Y Subseries)
- Timer: 6 channels
  - 16-bit timer: 1 channel
  - 8-bit timer/event counter: 2 channels
  - 8-bit timer: 1 channel
  - Watch timer: 1 channel
  - Watchdog timer: 1 channel
- Vectored interrupt source: 17
- Supply voltage: V_{DD} = 1.8 to 5.5 V
- Operating ambient temperature: T_A = –40°C to +85°C

2.2 Applications

Power windows, keyless entry, battery management units, side air bags, etc.

2.3 Ordering Information

Part Number	Package	Internal ROM
μPD789166YGB-×××-8ES[Note]	44-pin plastic LQFP (10 × 10 mm)	Mask ROM
μPD789166YGA-×××-9EU[Note]	48-pin plastic TQFP (fine pitch) (7 × 7 mm)	Mask ROM
μPD789167YGB-×××-8ES[Note]	44-pin plastic LQFP (10 × 10 mm)	Mask ROM
μPD789167YGA-×××-9EU[Note]	48-pin plastic TQFP (fine pitch) (7 × 7 mm)	Mask ROM
μPD789176YGB-×××-8ES[Note]	44-pin plastic LQFP (10 × 10 mm)	Mask ROM
μPD789176YGA-×××-9EU[Note]	48-pin plastic TQFP (fine pitch) (7 × 7 mm)	Mask ROM
μPD789177YGB-×××-8ES[Note]	44-pin plastic LQFP (10 × 10 mm)	Mask ROM
μPD789177YGA-×××-9EU[Note]	48-pin plastic TQFP (fine pitch) (7 × 7 mm)	Mask ROM
μPD78F9177YGB-8ES[Note]	44-pin plastic LQFP (10 × 10 mm)	Flash memory
μPD78F9177YGA-9EU[Note]	48-pin plastic TQFP (fine pitch) (7 × 7 mm)	Flash memory

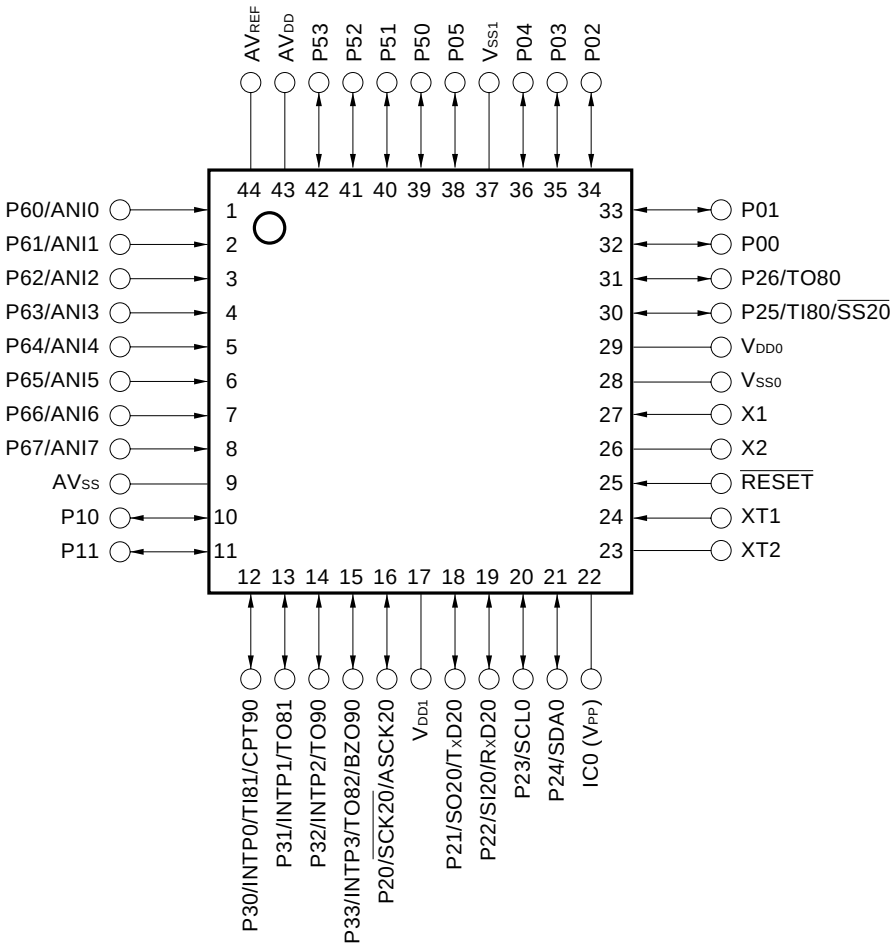
Note Under development

Remark ××× indicates ROM code suffix.

2.4 Pin Configuration (Top View)

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

  μPD789166YGB-×××-8ES[Note] μPD789176YGB-×××-8ES[Note]
  μPD789167YGB-×××-8ES[Note] μPD789177YGB-×××-8ES[Note]
  μPD78F9177YGB-8ES[Note]



Note Under development

Cautions 1. Connect the IC0 (internally connected) pin directly to the V_{SS0} or V_{SS1} pin.

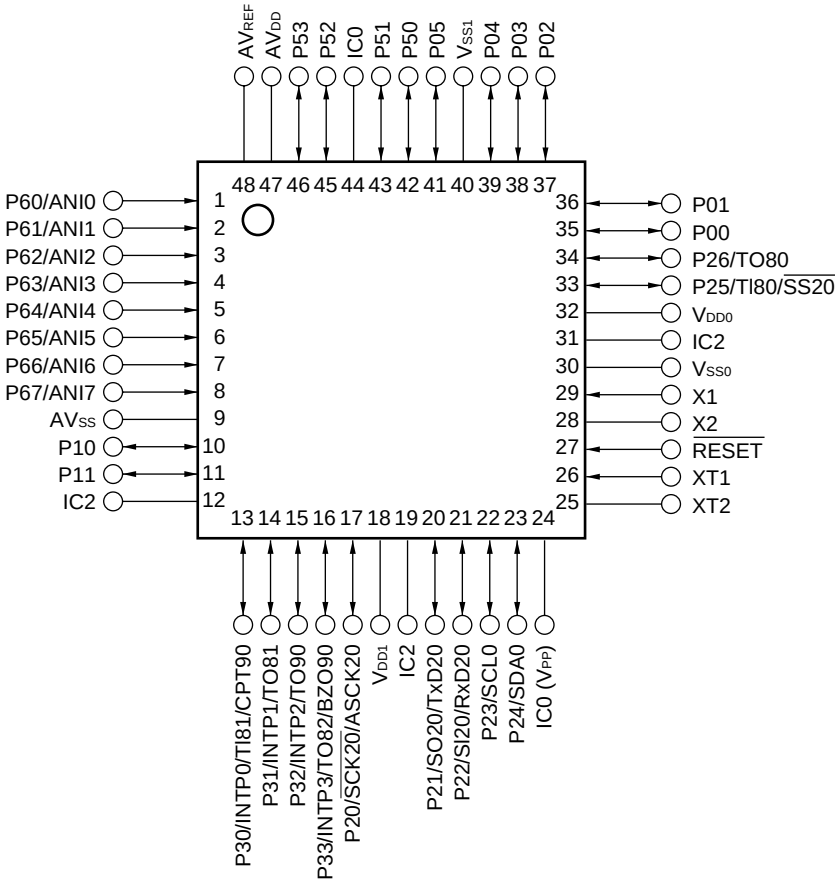
2. Connect the AV_{DD} pin to the V_{DD0} pin.

3. Connect the AV_{SS} pin to the V_{SS0} pin.

Remark Pin connections in parentheses are intended for the μPD78F9177Y.

- **48-pin plastic TQFP (fine pitch) (7 × 7 mm)**

  μPD789166YGA-×××-9EU[Note] μPD789176YGA-×××-9EU[Note]
  μPD789167YGA-×××-9EU[Note] μPD789177YGA-×××-9EU[Note]
  μPD78F9177YGA-9EU[Note]



Note Under development

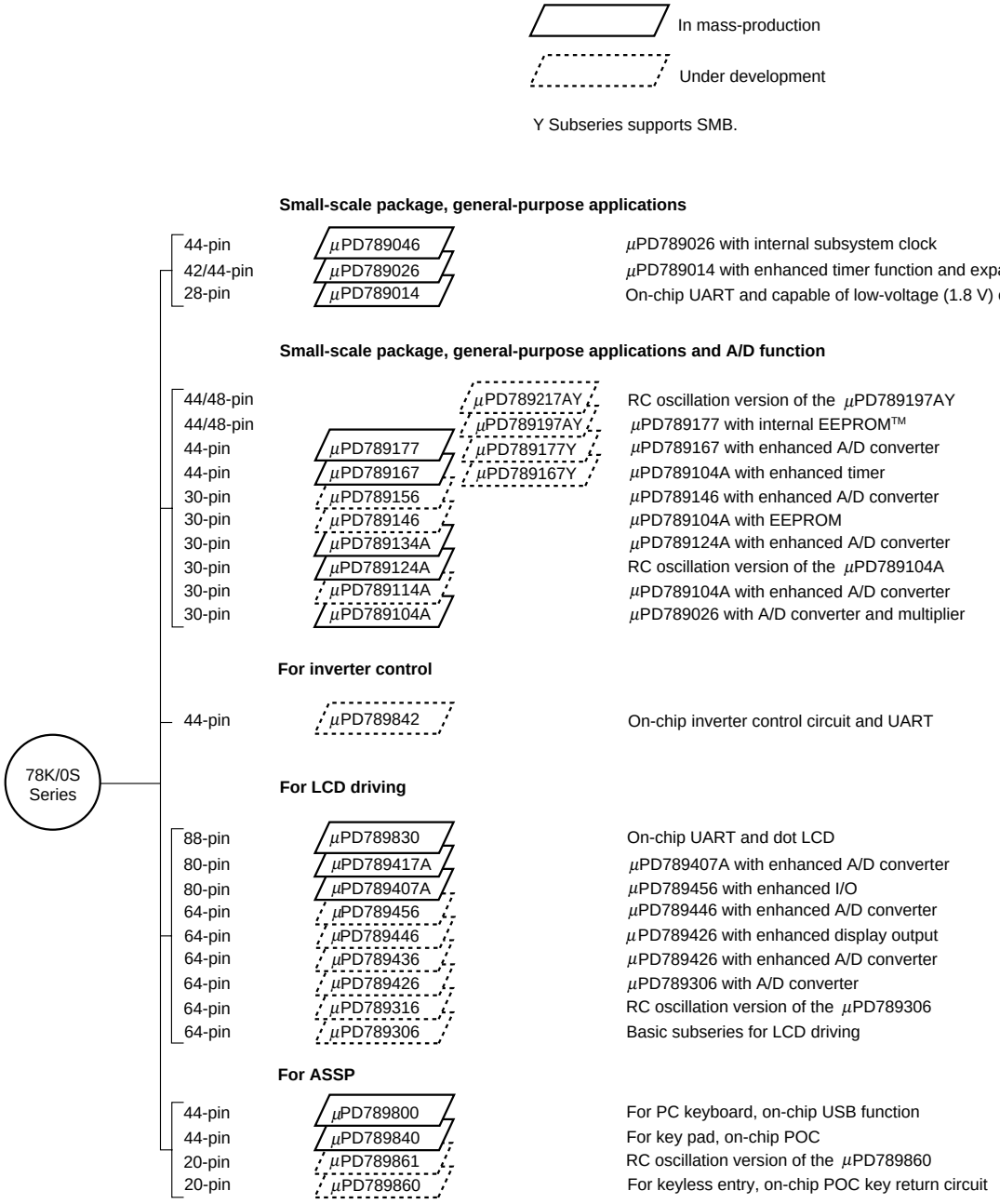
Cautions 1. Connect the IC0 (internally connected) pin directly to the V_{SS0} or V_{SS1} pin.
2. Leave the IC2 pin open.
3. Connect the AV_{DD} pin to the V_{DD0} pin.
4. Connect the AV_{SS} pin to the V_{SS0} pin.

Remark Pin connections in parentheses are intended for the μPD78F9177Y.

ANI0 to ANI7:	Analog Input
ASCK20:	Asynchronous Serial Input
AV_{DD}:	Analog Power Supply
AV_{REF}:	Analog Reference Voltage
AV_{SS}:	Analog Ground
BZO90:	Buzzer Output
CPT90:	Capture Trigger Input
IC0, IC2:	Internally Connected
INTP0 to INTP3:	Interrupt from Peripherals
P00 to P05:	Port 0
P10, P11:	Port 1
P20 to P26:	Port 2
P30 to P33:	Port 3
P50 to P53:	Port 5
P60 to P67:	Port 6
$\overline{\text{RESET}}$:	Reset
RxD20:	Receive Data
$\overline{\text{SCK20}}$:	Serial Clock (for SIO20)
SCL0:	Serial Clock (for SMB0)
SDA0:	Serial Data
SI20:	Serial Input
SO20:	Serial Output
$\overline{\text{SS20}}$:	Chip Select Input
TI80, TI81:	Timer Input
TO80 to TO82, TO90:	Timer Output
TxD20:	Transmit Data
V_{DD0}, V_{DD1}:	Power Supply
V_{PP}:	Programming Power Supply
V_{SS0}, V_{SS1}:	Ground
X1, X2:	Crystal (Main System Clock)
XT1, XT2:	Crystal (Subsystem Clock)

2.5 78K/0S Series Development

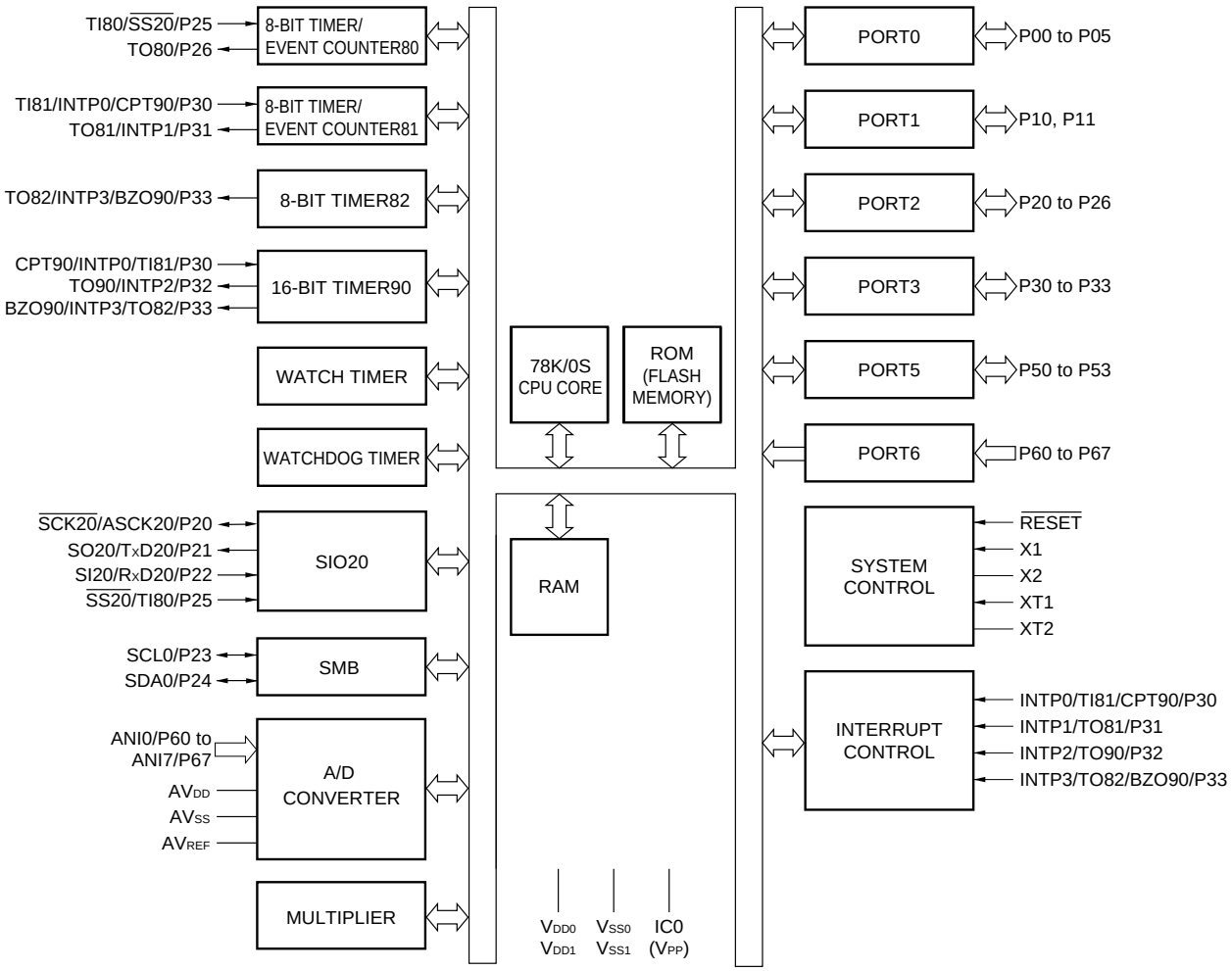
The 78K/0S Series products are shown below. The subseries names are indicated in frames.



The functions of the Y Subseries are listed below.

Subseries Name \ Function		ROM Capacity	Serial Interface Configuration	I/O (pins)	V_{DD} MIN. Value	Remark
Small-scale package, general-purpose application + A/D converter	μPD789217AY	16 K to 24 K	3-wire/UART: 1 ch SMB: 1 ch	31	1.8 V	On-chip EEPROM
	μPD789197AY					
	μPD789177Y					–
	μPD789167Y					

2.6 Block Diagram



Remarks 1. The size of the internal ROM varies depending on the model.

2. Pin connections in parentheses are intended for the μPD78F9177Y.

2.7 Outline of Function

Item \ Part Number		μPD789166Y μPD789176Y	μPD789167Y μPD789177Y	μPD78F9177Y
Internal memory	ROM	Mask ROM		Flash Memory
		16 Kbytes	24 Kbytes	24 Kbytes
	High-speed RAM	512 bytes		
Minimum instruction execution time		• 0.4/1.6 μs (operation with main system clock running at 5.0 MHz) • 122 μs (operation with subsystem clock running at 32.768 kHz)		
General-purpose registers		8 bits × 8 registers		
Instruction set		• 16-bit operations • Bit manipulations (such as set, reset, and test)		
Multiplier		8 bits × 8 bits = 16 bits		
I/O ports		Total: 31 • CMOS input: 8 • CMOS I/O: 17 • N-ch open-drain: 6		
A/D converter		• 8-bit resolution × 8 channels (μPD789167Y Subseries) • 10-bit resolution × 8 channels (μPD789177Y Subseries)		
Serial interface		• Switchable between 3-wire serial I/O and UART modes: 1 channel • SMB (System Management Bus): 1 channel		
Timers		• 16-bit timer: 1 channel • 8-bit timer/event counter: 2 channels • 8-bit timer: 1 channel • Watch timer: 1 channel • Watchdog timer: 1 channel		
Timer output		Four outputs		
Buzzer output		One output		
Vectored interrupt sources	Maskable	Internal: 12, external: 4		
	Nonmaskable	Internal: 1		
Power supply voltage		V_{DD} = 1.8 to 5.5 V		
Operating ambient temperature		T_A = –40°C to +85°C		
Package		• 44-pin plastic LQFP (10 × 10 mm) • 48-pin plastic TQFP (fine pitch) (7 × 7 mm)		

The timers are outlined below.

		16-Bit Timer 90	8-Bit Timer/Event Counter 80	8-Bit Timer/Event Counter 81	8-Bit Timer 82	Watch Timer	Watchdog Timer
Operating mode	Interval timer	–	1 channel	1 channel	1 channel	1 channel[Note 1]	1 channel[Note 2]
	External event counter	–	1 channel	1 channel	–	–	–
Function	Timer output	1 output	1 output	1 output	1 output	–	–
	PWM output	–	1 output	1 output	1 output	–	–
	Square-wave output	–	1 output	1 output	1 output	–	–
	Buzzer output	1 output	–	–	–	–	–
	Capture	1 input	–	–	–	–	–
	Interrupt source	1	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 provides the watchdog timer function and interval timer function. Use either of the functions.

CHAPTER 3 PIN FUNCTIONS (μPD789167 AND 789177 SUBSEREIS)

3.1 Pin Function List

(1) Port pins

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P05	I/O	Port 0 6-bit input/output port Input/output mode can be specified in 1-bit units When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).	Input	–
P10, P11	I/O	Port 1 2-bit input/output port Input/output mode 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 input/output port Input/output mode can be specified in 1-bit units For P20 to P22, P25, and P26, an on-chip pull-up resistor can be specified by means of pull-up resistor option register B2 (PUB2). Only P23 and P24 can be used as N-ch open-drain input/output port pins.	Input	$\overline{\text{SCK20}}$/ASCK20
P21				SO20/TxD20
P22				SI20/RxD20
P23				–
P24				–
P25				TI80/$\overline{\text{SS20}}$
P26				TO80
P30	I/O	Port 3 4-bit input/output port Input/output mode can be specified in 1-bit units An on-chip pull-up resistor can be specified by means of pull-up resistor option register B3 (PUB3).	Input	INTP0/TI81/CPT90
P31				INTP1/TO81
P32				INTP2/TO90
P33				INTP3/TO82/BZO90
P50 to P53	I/O	Port 5 4-bit N-ch open-drain input/output port Input/output mode 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 P67	Input	Port 6 8-bit input-only port	Input	ANI0 to ANI7

(2) Non-port pins

Pin Name	I/O	Function	After Reset	Alternate Function
INTP0	Input	External interrupt input for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified	Input	P30/TI81/CPT90
INTP1				P31/TO81
INTP2				P32/TO90
INTP3				P33/TO82/BZO90
SI20	Input	Serial data input to serial interface	Input	P22/RxD20
SO20	Output	Serial data output from serial interface	Input	P21/TxD20
$\overline{SCK20}$	I/O	Serial clock input/output for serial interface	Input	P20/ASCK20
$\overline{SS20}$	Input	Chip select input to serial interface	Input	P25/TI80
ASCK20	Input	Serial clock input for asynchronous serial interface	Input	P20/$\overline{SCK20}$
RxD20	Input	Serial data input for asynchronous serial interface	Input	P22/SI20
TxD20	Output	Serial data output for asynchronous serial interface	Input	P21/SO20
TI80	Input	External count clock input to 8-bit timer/event counter (TM80)	Input	P25/$\overline{SS20}$
TI81	Input	External count clock input to 8-bit timer/event counter (TM81)	Input	P30/INTP0/CPT90
TO80	Output	8-bit timer/event counter (TM80) output	Input	P26
TO81	Output	8-bit timer/event counter (TM81) output	Input	P31/INTP1
TO82	Output	8-bit timer (TM82) output	Input	P33/INTP3/BZO90
TO90	Output	16-bit timer (TM90) output	Input	P32/INTP2
CPT90	Input	Capture edge input	Input	P30/INTP0/TI81
BZO90	Output	Buzzer output	Input	P33/INTP3/TO82
ANI0 to ANI7	Input	A/D converter analog input	Input	P60 to P67
AV_{REF}	–	A/D converter reference voltage	–	–
AV_{SS}	–	A/D converter ground potential	–	–
AV_{DD}	–	A/D converter analog power supply	–	–
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_{DD0}	–	Positive power supply	–	–
V_{DD1}	–	Positive power supply (other than ports)	–	–
V_{SS0}	–	Ground potential	–	–
V_{SS1}	–	Ground potential (other than ports)	–	–
IC0	–	Internally connected. Connect this pin directly to the V_{SS0} or V_{SS1} pin.	–	–
V_{PP}	–	This pin is used to set flash memory programming mode and applies a high voltage when a program is written or verified. In normal operation mode, connect this pin directly to the V_{SS0} or V_{SS1} pin.	–	–

3.2 Description of Pin Functions

3.2.1 P00 to P05 (Port 0)

These pins constitute a 6-bit I/O port and can be set to input or output port mode in 1-bit units by using port mode register 0 (PM0). When these pins are used as an input port, an on-chip pull-up resistor can be used by setting pull-up resistor option register 0 (PU0).

3.2.2 P10, P11 (Port 1)

These pins constitute a 2-bit I/O port and can be set to input or output port mode in 1-bit units by using port mode register 1 (PM1). When these pins are used as an input port, an on-chip pull-up resistor can be used by setting pull-up resistor option register 0 (PU0).

3.2.3 P20 to P26 (Port 2)

These pins constitute a 7-bit I/O port. In addition, these pins provide a function to perform input/output to/from the timer and to input/output the data and clock of the serial interface.

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

(1) Port mode

In port mode, P20 to P26 function as a 7-bit I/O port. Port 2 can be set to input or output mode in 1-bit units by using port mode register 2 (PM2). For P20 to P22, P25, and P26, whether to use on-chip pull-up resistors can be specified in 1-bit units by using pull-up resistor option register B2 (PUB2), regardless of the setting of port mode register 2 (PM2). P23 and P24 are N-ch open-drain I/O ports.

(2) Control mode

In this mode, P20 to P26 function as the timer input/output, the data input/output and the clock input/output of the serial interface.

(a) TI80

This is the external clock input pin for 8-bit timer/event counter 80.

(b) TO80

This is the timer output pin of 8-bit timer/event counter 80.

(c) SI20, SO20

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

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

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

(e) $\overline{\text{SS20}}$

This is the chip select input pin of the serial interface.

(f) RxD20, TxD20

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

(g) ASCK20

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

Caution When using P20 to P26 as serial interface pins, the input/output mode and output latch must be set according to the functions to be used. For details of the setting, see Table 14-2 Serial Interface 20 Operating Mode Settings.

3.2.4 P30 to P33 (Port 3)

These pins constitute a 4-bit I/O port. In addition, these pins function as the timer input/output and the external interrupt input.

Port 3 can be set to the following operation modes in 1-bit units.

(1) Port mode

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

(2) Control mode

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

(a) TI81

This is the external clock input pin for 8-bit timer/event counter 81.

(b) TO90, TO81, TO82

These are the output pins of 16-bit timer 90, 8-bit timer/event counter 81, and 8-bit timer 82.

(c) CPT90

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

(d) BZO90

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

(e) INTP0 to INTP3

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

3.2.5 P50 to P53 (Port 5)

These pins constitute a 4-bit N-ch open-drain I/O port. Port 5 can be set to input or output mode in 1-bit units by using port mode register 5 (PM5). For a mask ROM version, whether a pull-up resistor is to be incorporated can be specified by a mask option.

3.2.6 P60 to P67 (Port 6)

These pins constitute an 8-bit input-only port. They can function as A/D converter input pins as well as a general-purpose input port.

(1) Port mode

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

(2) Control mode

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

3.2.7 $\overline{\text{RESET}}$

A low-level active system reset signal is input to this pin.

3.2.8 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.

3.2.9 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.

3.2.10 AV_{DD}

Analog power supply pin of the A/D converter. Always use the same potential as that of the V_{DD0} pin even when A/D converter is not used.

3.2.11 AV_{SS}

This is a ground potential pin of the A/D converter. Always use the same potential as that of the V_{SS0} pin even when the A/D converter is not used.

3.2.12 AV_{REF}

This is an A/D converter reference voltage input pin. When an A/D converter is not used, connect this pin to V_{SS0}.

3.2.13 V_{DD0}, V_{DD1}

V_{DD0} is a positive power supply port pin.

V_{DD1} is a positive power supply pin other than port pin.

3.2.14 V_{SS0}, V_{SS1}

V_{SS0} is a ground potential port pin.

V_{SS1} is a ground potential pin other than port pin.

3.2.15 V_{PP} (μPD78F9177 only)

High voltage apply pin for flash memory programming mode setting and program write/verify.

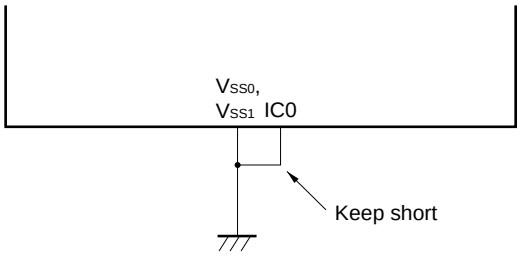
Directly connect this pin to V_{SS0} or V_{SS1} in normal operation mode.

3.2.16 IC0 (mask ROM version only)

The IC0 (Internally Connected) pin is used to set the μPD789167 and 789177 Subseries to test mode before shipment. In normal operation mode, directly connect this pin to the V_{SS0} or V_{SS1} pin with as short a wiring length as possible.

If a potential difference is generated between the IC0 pin and V_{SS0} or V_{SS1} pin due to a long wiring length between the IC0 pin and V_{SS0} or V_{SS1} pin or an external noise superimposed on the IC0 pin, a user program may not run correctly.

- Directly connect the IC0 pin to the V_{SS0} or V_{SS1} pin.



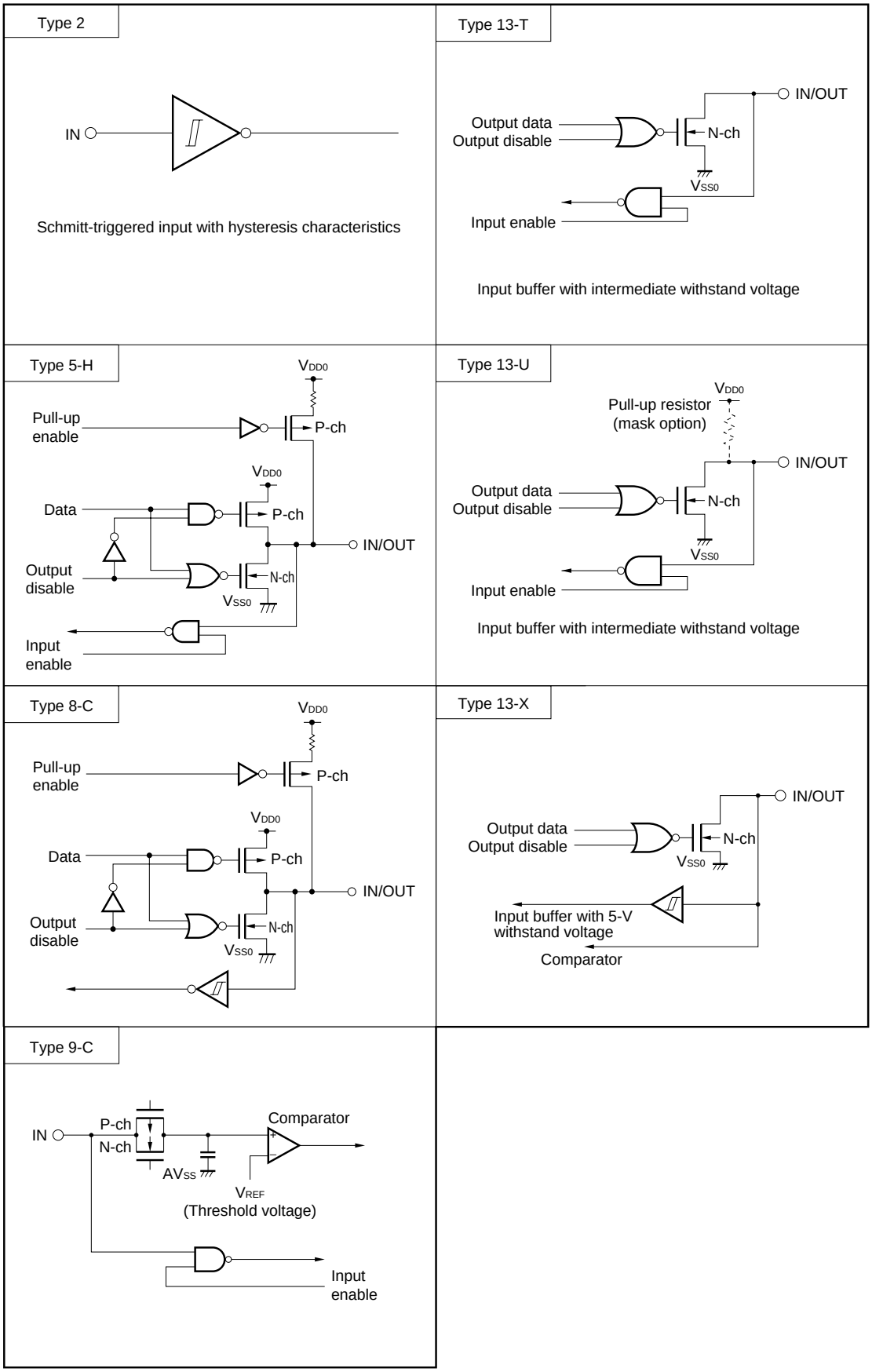
3.3 Pin Input/Output Circuits and Recommended Connection of Unused Pins

The input/output circuit type of each pin and recommended connection of unused pins are shown in Table 3-1. For the input/output circuit configuration of each type, refer to Figure 3-1.

Table 3-1. Types of Input/Output Circuits for Each Pin and Recommended Connection of Unused Pins

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
P00 to P05	5-H	I/O	Input: Independently connect to V_{DD0}, V_{DD1} or V_{SS0}, V_{SS1} via a resistor. Output: Leave open.
P10, P11			
P20/$\overline{SCK20}$/ASCK20	8-C		
P21/SO20/TxD20			
P22/SI20/RxD20			
P23	13-X		Input: Independently connect to V_{DD0} or V_{DD1} via a resistor. Output: Leave open.
P24			
P25/TI80/$\overline{SS20}$	8-C		Input: Independently connect to V_{DD0}, V_{DD1} or V_{SS0}, V_{SS1} via a resistor. Output: Leave open.
P26/TO80			
★ P30/INTP0/TI81/CPT90			Input: Independently connect to V_{SS0} or V_{SS1} via a resistor. Output: Leave open.
★ P31/INTP1/TO81			
★ P32/INTP2/TO90			
★ P33/INTP3/TO82/BZO90			
P50 to P53 (mask ROM version)	13-U		Input: Independently connect to V_{DD0} or V_{DD1} via a resistor. Output: Leave open.
P50 to P53 (flash memory version)	13-T		
P60/ANI0 to P67/ANI7	9-C	Input	Connect directly to V_{DD0}, V_{DD1} or V_{SS0}, V_{SS1}.
XT1	–	Input	Connect to V_{SS0} or V_{SS1}.
XT2		–	Leave open.
$\overline{RESET}$	2	Input	–
IC0 (mask ROM version)	–	–	Connect directly to V_{SS0} or V_{SS1}.
V_{PP} (flash memory version)			Connect directly to V_{SS0} or V_{SS1}.

Figure 3-1. Pin Input/Output Circuits



★ CHAPTER 4 PIN FUNCTIONS (μPD789167Y AND 789177Y SUBSEREIS)

4.1 Pin Function List

(1) Port pins

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P05	I/O	Port 0 6-bit input/output port Input/output mode can be specified in 1-bit units When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).	Input	–
P10, P11	I/O	Port 1 2-bit input/output port Input/output mode 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 input/output port Input/output mode can be specified in 1-bit units For P20 to P22, P25, and P26, an on-chip pull-up resistor can be specified by means of pull-up resistor option register B2 (PUB2). Only P23 and P24 can be used as N-ch open-drain input/output port pins.	Input	$\overline{\text{SCK20}}$/ASCK20
P21				SO20/TxD20
P22				SI20/RxD20
P23				SCL0
P24				SDA0
P25				TI80/$\overline{\text{SS20}}$
P26				TO80
P30	I/O	Port 3 4-bit input/output port Input/output mode can be specified in 1-bit units An on-chip pull-up resistor can be specified by means of pull-up resistor option register B3 (PUB3).	Input	INTP0/TI81/CPT90
P31				INTP1/TO81
P32				INTP2/TO90
P33				INTP3/TO82/BZO90
P50 to P53	I/O	Port 5 4-bit N-ch open-drain input/output port Input/output mode 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 P67	Input	Port 6 8-bit input-only port	Input	ANI0 to ANI7

(2) Non-port pins

Pin Name	I/O	Function	After Reset	Alternate Function
INTP0	Input	External interrupt input for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified	Input	P30/TI81/CPT90
INTP1				P31/TO81
INTP2				P32/TO90
INTP3				P33/TO82/BZO90
SI20	Input	Serial data input to serial interface	Input	P22/RxD20
SO20	Output	Serial data output from serial interface	Input	P21/TxD20
$\overline{SCK20}$	I/O	Serial clock input/output for serial interface	Input	P20/ASCK20
$\overline{SS20}$	Input	Chip select input to serial interface	Input	P25/TI80
ASCK20	Input	Serial clock input for asynchronous serial interface	Input	P20/$\overline{SCK20}$
RxD20	Input	Serial data input for asynchronous serial interface	Input	P22/SI20
TxD20	Output	Serial data output for asynchronous serial interface	Input	P21/SO20
SCL0	I/O	SMB0 clock input/output	Input	P23
SDA0	I/O	SMB0 data input/output	Input	P24
TI80	Input	External count clock input to 8-bit timer/event counter (TM80)	Input	P25/$\overline{SS20}$
TI81	Input	External count clock input to 8-bit timer/event counter (TM81)	Input	P30/INTP0/CPT90
TO80	Output	8-bit timer/event counter (TM80) output	Input	P26
TO81	Output	8-bit timer/event counter (TM81) output	Input	P31/INTP1
TO82	Output	8-bit timer (TM82) output	Input	P33/INTP3/BZO90
TO90	Output	16-bit timer (TM90) output	Input	P32/INTP2
CPT90	Input	Capture edge input	Input	P30/INTP0/TI81
BZO90	Output	Buzzer output	Input	P33/INTP3/TO82
ANI0 to ANI7	Input	A/D converter analog input	Input	P60 to P67
AV_{REF}	–	A/D converter reference voltage	–	–
AV_{SS}	–	A/D converter ground potential	–	–
AV_{DD}	–	A/D converter analog power supply	–	–
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_{DD0}	–	Positive power supply	–	–
V_{DD1}	–	Positive power supply (other than ports)	–	–
V_{SS0}	–	Ground potential	–	–
V_{SS1}	–	Ground potential (other than ports)	–	–
IC0	–	Internally connected. Connect this pin directly to the V_{SS0} or V_{SS1} pin.	–	–
IC2	–	Internally connected. Leave this pin open.	–	–
V_{PP}	–	This pin is used to set flash memory programming mode and applies a high voltage when a program is written or verified. In normal operation mode, connect this pin directly to the V_{SS0} or V_{SS1} pin.	–	–

4.2 Description of Pin Functions

4.2.1 P00 to P05 (Port 0)

These pins constitute a 6-bit I/O port and can be set to input or output port mode in 1-bit units by using port mode register 0 (PM0). When these pins are used as an input port, an on-chip pull-up resistor can be used by setting pull-up resistor option register 0 (PU0).

4.2.2 P10, P11 (Port 1)

These pins constitute a 2-bit I/O port and can be set to input or output port mode in 1-bit units by using port mode register 1 (PM1). When these pins are used as an input port, an on-chip pull-up resistor can be used by setting pull-up resistor option register 0 (PU0).

4.2.3 P20 to P26 (Port 2)

These pins constitute a 7-bit I/O port. In addition, these pins provide a function to perform input/output to/from the timer and to input/output the data and clock of the serial interface.

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

(1) Port mode

In port mode, P20 to P26 function as a 7-bit I/O port. Port 2 can be set to input or output mode in 1-bit units by using port mode register 2 (PM2). For P20 to P22, P25, and P26, whether to use on-chip pull-up resistors can be specified in 1-bit units by using pull-up resistor option register B2 (PUB2), regardless of the setting of port mode register 2 (PM2). P23 and P24 are N-ch open-drain I/O ports.

(2) Control mode

In this mode, P20 to P26 function as the timer input/output, the data input/output and the clock input/output of the serial interface.

(a) TI80

This is the external clock input pin for 8-bit timer/event counter 80.

(b) TO80

This is the timer output pin of 8-bit timer/event counter 80.

(c) SI20, SO20

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

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

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

(e) $\overline{\text{SS20}}$

This is the chip select input pin of the serial interface.

(f) RxD20, TxD20

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

(g) ASCK20

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

(h) SCL0

This is the clock I/O pin of SMB0.

(i) SDA0

This is the data I/O pin of SMB0.

Caution When using P20 to P26 as serial interface pins, the input/output mode and output latch must be set according to the functions to be used. For details of the setting, see Table 14-2 Serial Interface 20 Operating Mode Settings.

4.2.4 P30 to P33 (Port 3)

These pins constitute a 4-bit I/O port. In addition, these pins function as the timer input/output and the external interrupt input.

Port 3 can be set to the following operation modes in 1-bit units.

(1) Port mode

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

(2) Control mode

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

(a) TI81

This is the external clock input pin for 8-bit timer/event counter 81.

(b) TO90, TO81, TO82

These are the output pins of 16-bit timer 90, 8-bit timer/event counter 81, and 8-bit timer 82.

(c) CPT90

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

(d) BZO90

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

(e) INTP0 to INTP3

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

4.2.5 P50 to P53 (Port 5)

These pins constitute a 4-bit N-ch open-drain I/O port. Port 5 can be set to input or output mode in 1-bit units by using port mode register 5 (PM5). For a mask ROM version, whether a pull-up resistor is to be incorporated can be specified by a mask option.

4.2.6 P60 to P67 (Port 6)

These pins constitute an 8-bit input-only port. They can function as A/D converter input pins as well as a general-purpose input port.

(1) Port mode

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

(2) Control mode

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

4.2.7 $\overline{\text{RESET}}$

A low-level active system reset signal is input to this pin.

4.2.8 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.

4.2.9 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.

4.2.10 AV_{DD}

Analog power supply pin of the A/D converter. Always use the same potential as that of the V_{DD0} pin even when A/D converter is not used.

4.2.11 AV_{SS}

This is a ground potential pin of the A/D converter. Always use the same potential as that of the V_{SS0} pin even when the A/D converter is not used.

4.2.12 AV_{REF}

This is an A/D converter reference voltage input pin. When an A/D converter is not used, connect this pin to V_{SS0}.

4.2.13 V_{DD0}, V_{DD1}

V_{DD0} is a positive power supply port pin.

V_{DD1} is a positive power supply pin other than port pin.

4.2.14 V_{SS0}, V_{SS1}

V_{SS0} is a ground potential port pin.

V_{SS1} is a ground potential pin other than port pin.

4.2.15 V_{PP} (μPD78F9177Y only)

High voltage apply pin for flash memory programming mode setting and program write/verify.

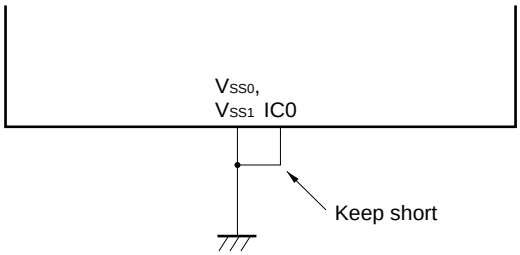
Directly connect this pin to V_{SS0} or V_{SS1} in normal operation mode.

4.2.16 IC0 (mask ROM version only)

The IC0 (Internally Connected) pin is used to set the μPD789167Y and 789177Y Subseries to test mode before shipment. In normal operation mode, directly connect this pin to the V_{SS0} or V_{SS1} pin with as short a wiring length as possible.

If a potential difference is generated between the IC0 pin and V_{SS0} or V_{SS1} pin due to a long wiring length between the IC0 pin and V_{SS0} or V_{SS1} pin or an external noise superimposed on the IC0 pin, a user program may not run correctly.

- Directly connect the IC0 pin to the V_{SS0} or V_{SS1} pin.



4.2.17 IC2

The IC2 pin is internally connected. Leave this pin open.

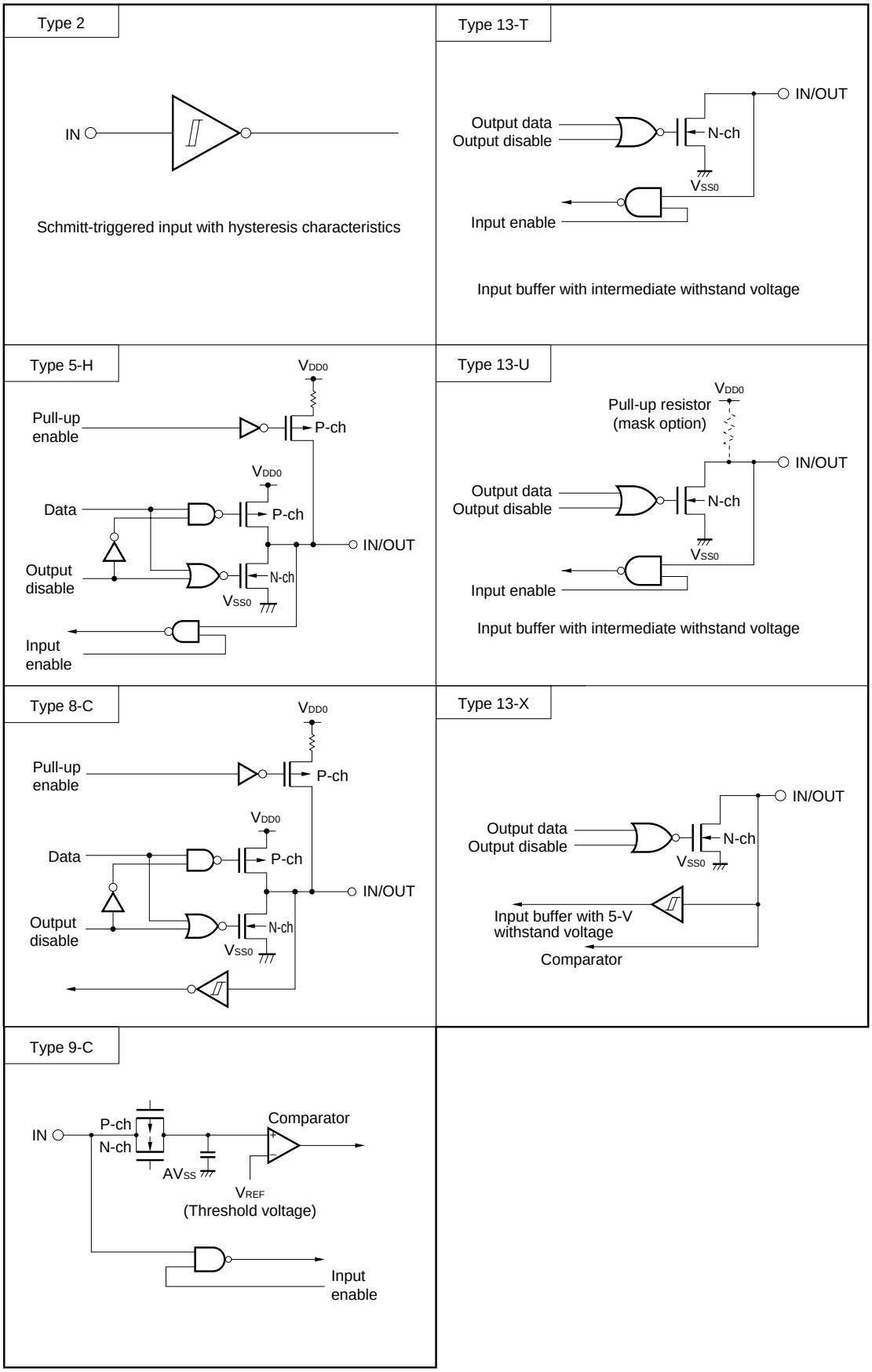
4.3 Pin Input/Output Circuits and Recommended Connection of Unused Pins

The input/output circuit type of each pin and recommended connection of unused pins are shown in Table 4-1. For the input/output circuit configuration of each type, refer to Figure 4-1.

Table 4-1. Types of Input/Output Circuits for Each Pin and Recommended Connection of Unused Pins

	Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
	P00 to P05	5-H	I/O	Input: Independently connect to V_{DD0}, V_{DD1} or V_{SS0}, V_{SS1} via a resistor. Output: Leave open.
	P10, P11			
	P20/$\overline{SCK20}$/ASCK20	8-C		
	P21/SO20/TxD20			
	P22/SI20/RxD20			
	P23/SCL0	13-X		Input: Independently connect to V_{DD0} or V_{DD1} via a resistor. Output: Leave open.
	P24/SDA0			
	P25/TI80/$\overline{SS20}$	8-C		Input: Independently connect to V_{DD0}, V_{DD1} or V_{SS0}, V_{SS1} via a resistor. Output: Leave open.
	P26/TO80			
★	P30/INTP0/TI81/CPT90			Input: Independently connect to V_{SS0} or V_{SS1} via a resistor. Output: Leave open.
★	P31/INTP1/TO81			
★	P32/INTP2/TO90			
★	P33/INTP3/TO82/BZO90			
	P50 to P53 (mask ROM version)	13-U		Input: Independently connect to V_{DD0} or V_{DD1} via a resistor. Output: Leave open.
	P50 to P53 (flash memory version)	13-T		
	P60/ANI0 to P67/ANI7	9-C	Input	Connect directly to V_{DD0}, V_{DD1} or V_{SS0}, V_{SS1}.
	XT1	–	Input	Connect to V_{SS0} or V_{SS1}.
	XT2		–	Leave open.
	$\overline{RESET}$	2	Input	–
	IC0 (mask ROM version)	–	–	Connect directly to V_{SS0} or V_{SS1}.
	IC2			Leave open.
	V_{PP} (flash memory version)			Connect directly to V_{SS0} or V_{SS1}.

Figure 4-1. Pin Input/Output Circuits



CHAPTER 5 CPU ARCHITECTURE

5.1 Memory Space

Products in the μPD789167, 789177, 789167Y, and 789177Y Subseries can each access up to 64 Kbytes of memory space. Figures 5-1 through 5-3 show the memory maps.

Figure 5-1. Memory Map (μPD789166, μPD789176, μPD789166Y, and μPD789176Y)

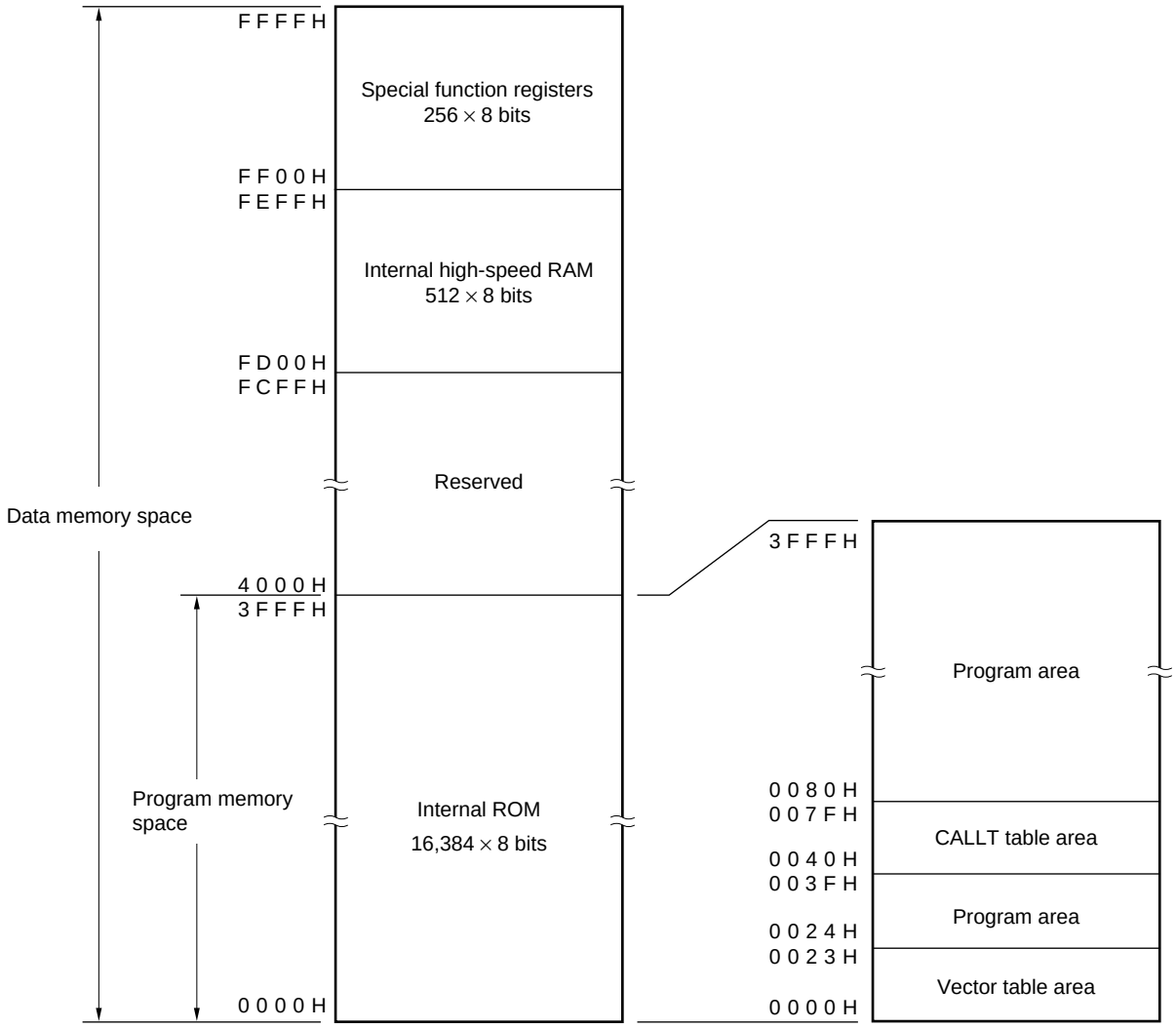


Figure 5-2. Memory Map (μPD789167, μPD789177, μPD789167Y, and μPD789177Y)

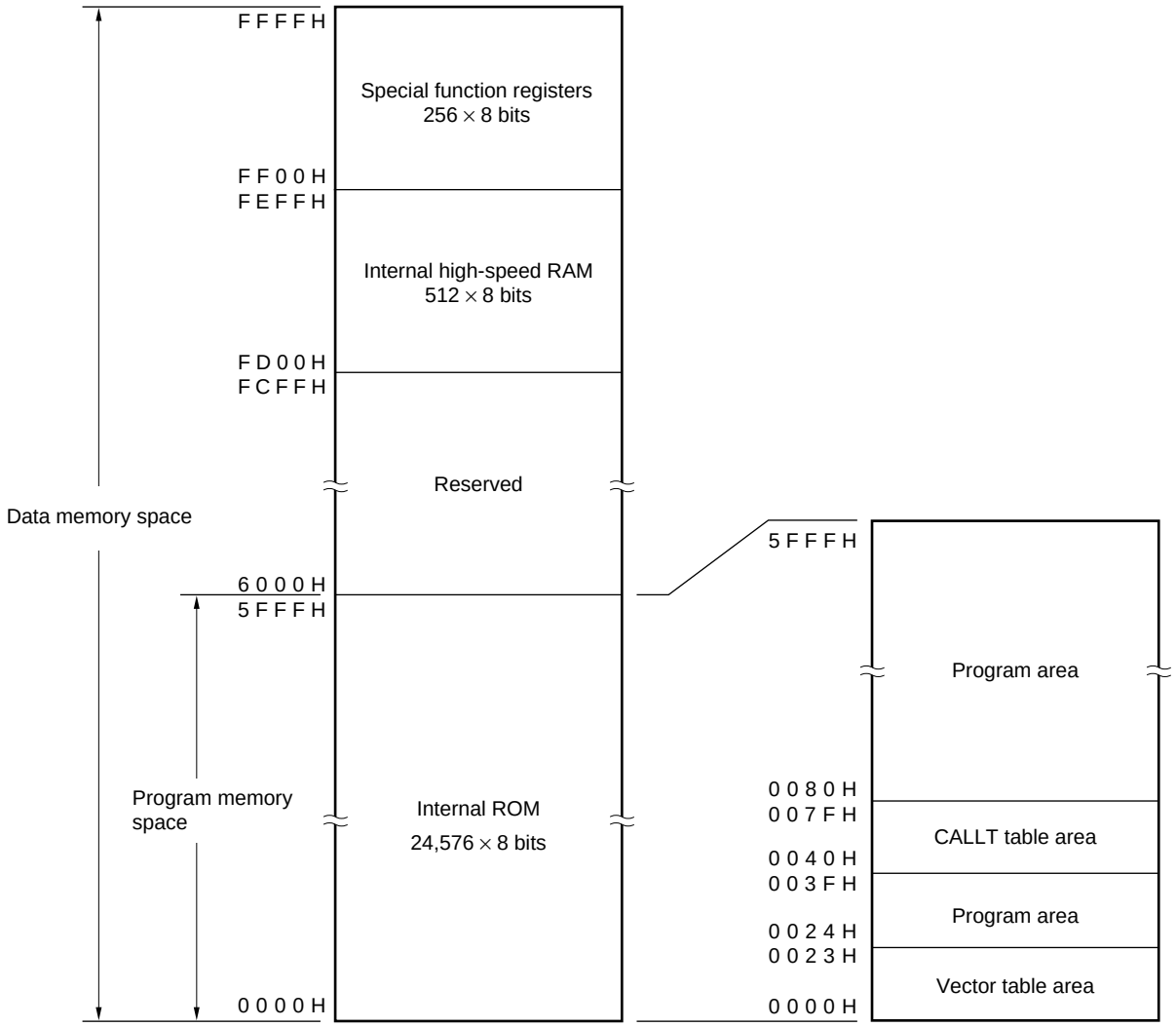
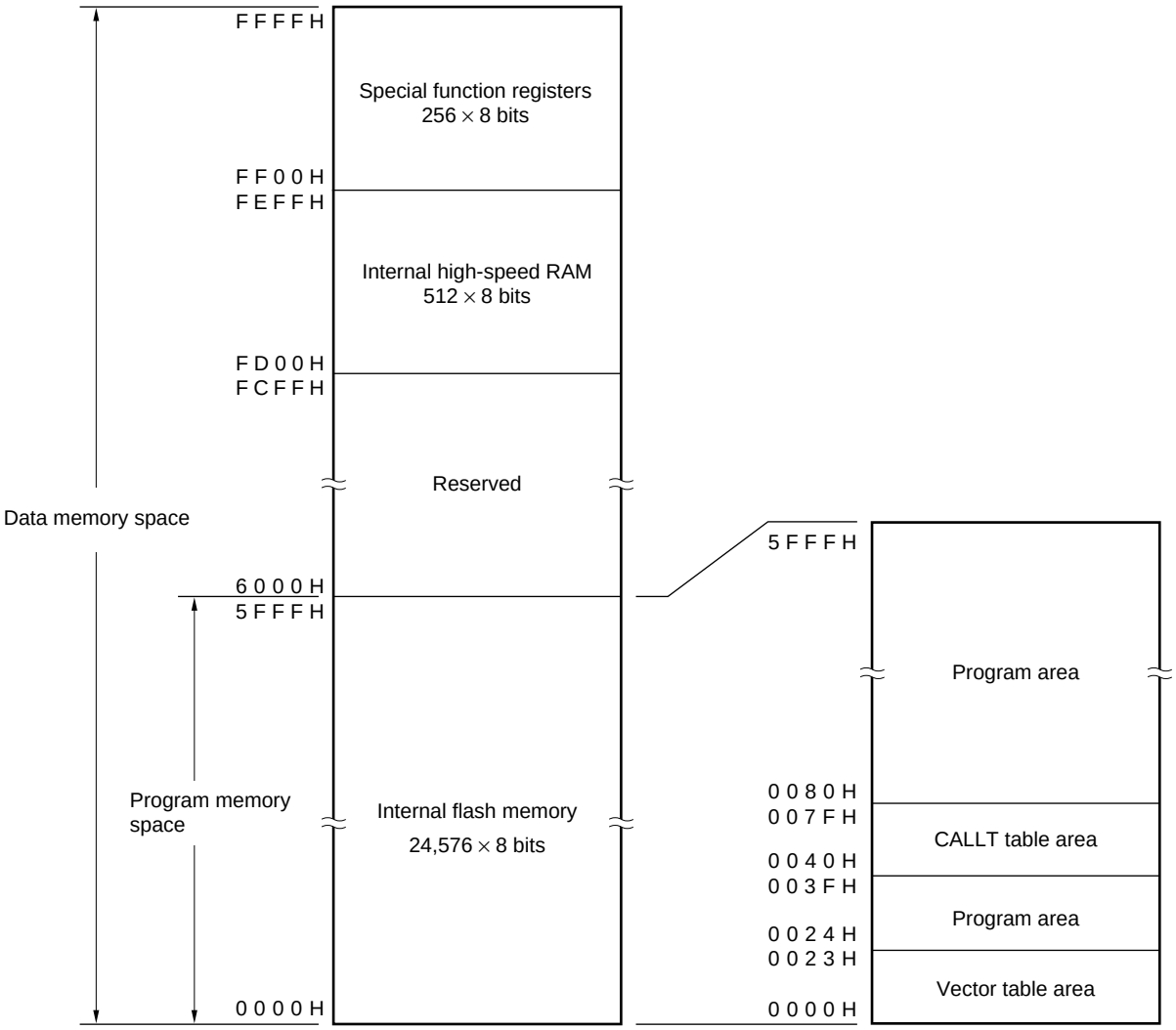


Figure 5-3. Memory Map (μPD78F9177 and μPD78F9177Y)



5.1.1 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 μPD789167, 789177, 789167Y, and 789177Y Subseries provide the following internal ROMs (or flash memory) containing the following capacities.

Table 5-1. Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
μPD789166, μPD789176, μPD789166Y, μPD789176Y	Mask ROM	16,384 × 8 bits
μPD789167, μPD789177, μPD789167Y, μPD789177Y		24,576 × 8 bits
μPD78F9177, μPD78F9177Y	Flash memory	24,576 × 8 bits

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

(1) Vector table area

A 36-byte area of addresses 0000H to 0023H 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 5-2. Vector Table

Vector Table Address	Interrupt Request	Vector Table Address	Interrupt Request
0000H	$\overline{\text{RESET}}$ input	0014H	INTWTI
0004H	INTWDT	0016H	INTTM80
0006H	INTP0	0018H	INTTM81
0008H	INTP1	001AH	INTTM82
000AH	INTP2	001CH	INTTM90
★ 000CH	INTP3	001EH	INTSMB0 Note
★ 000EH	INTSR20/INTCSI20	0020H	INTSMBOV0 Note
0010H	INTST20	0022H	INTAD0
0012H	INTWT		

Note For the μPD789167Y and 789177Y Subseries only

(2) CALLT instruction table area

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

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

The μPD789167, 789177, 789167Y, and 789177Y Subseries provide 512-byte internal high-speed RAM.

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

5.1.3 Special function register (SFR) area

Special function registers (SFRs) of on-chip peripheral hardware are allocated to an area of FF00H to FFFFH (see **Table 5-3**).

5.1.4 Data memory addressing

Each of the μPD789167, 789177, 789167Y, 789177Y Subseries is provided with a wide range of addressing modes to make memory manipulation as efficient as possible. A data memory area (FD00H to FFFFH) can be accessed using a unique addressing mode according to its use, such as a special function register (SFR). Figures 5-4 through 5-6 illustrate the data memory addressing modes.

Figure 5-4. Data Memory Addressing Modes (μPD789166, μPD789176, μPD789166Y, and μPD789176Y)

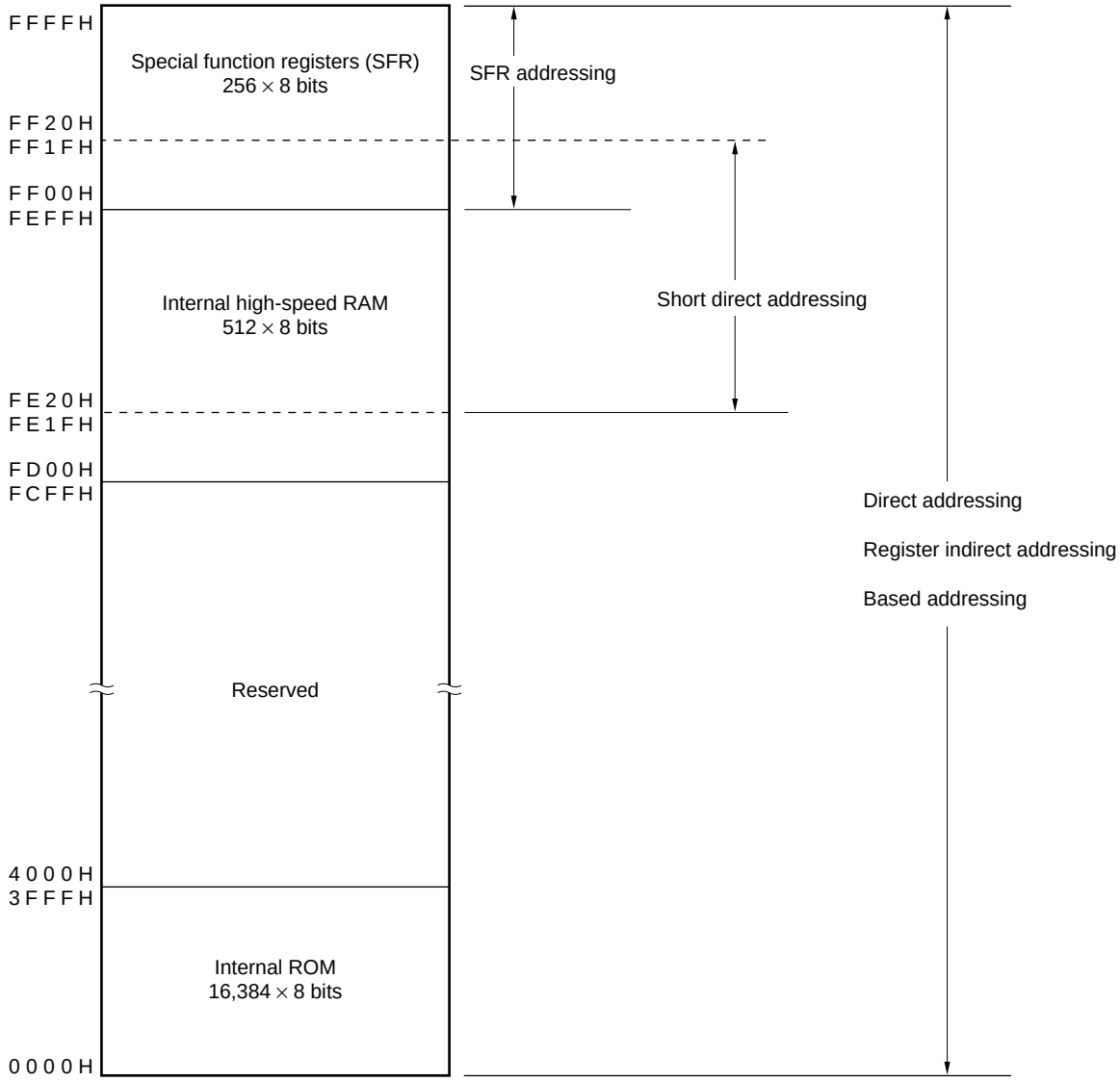


Figure 5-5. Data Memory Addressing Modes (μPD789167, μPD789177, μPD789167Y, and μPD789177Y)

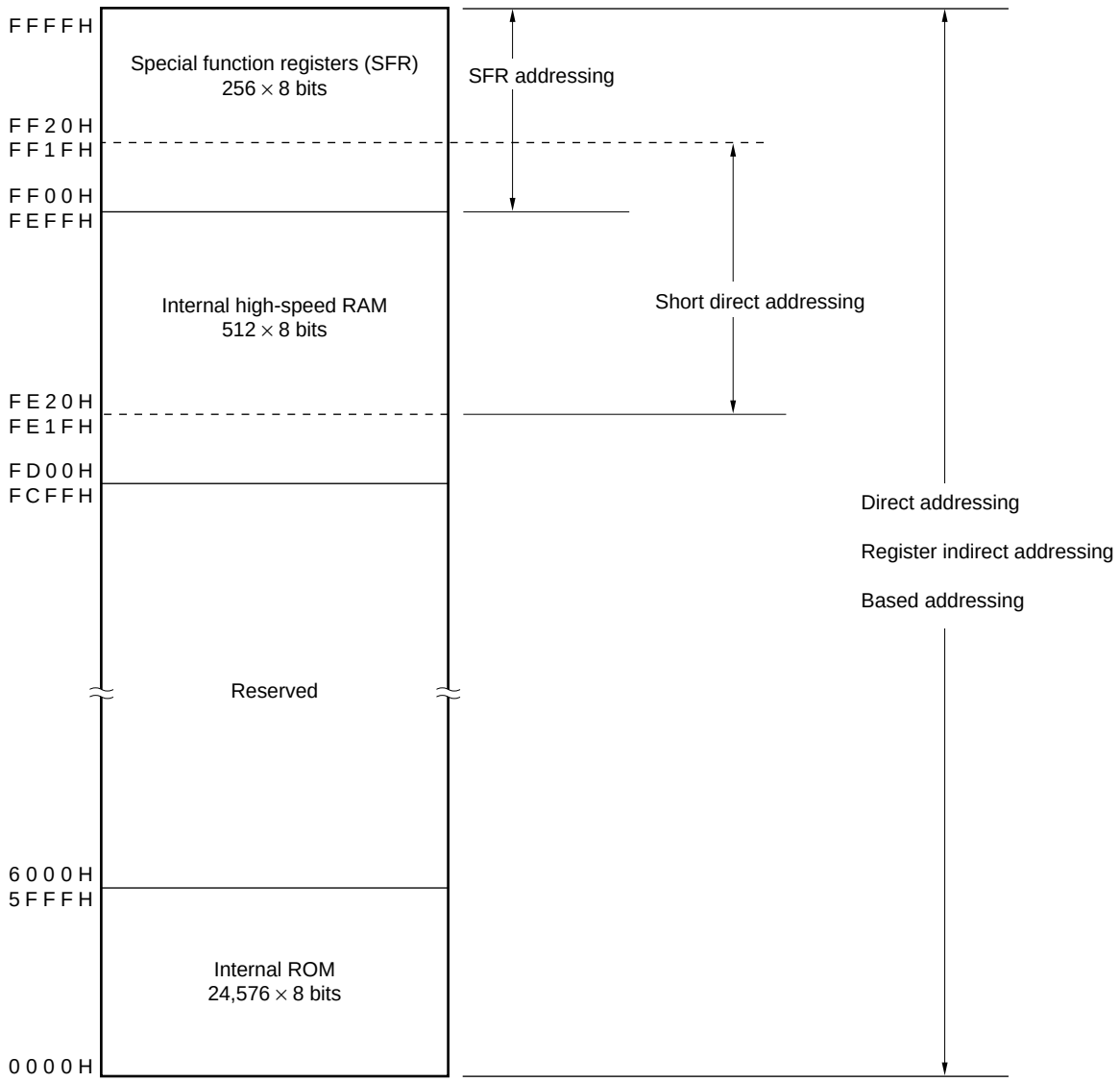
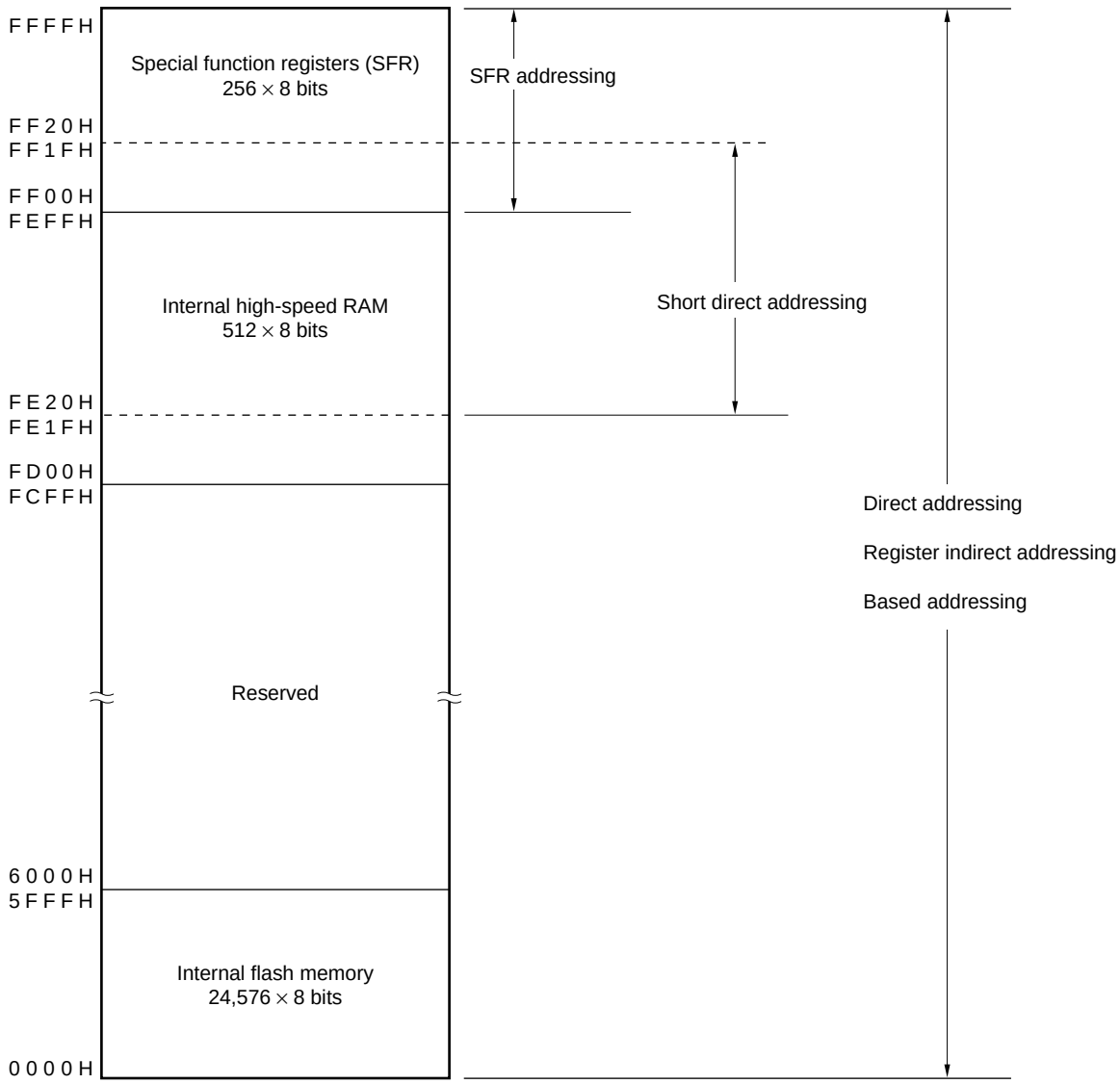


Figure 5-6. Data Memory Addressing Modes (μPD78F9177 and μPD78F9177Y)



5.2 Processor Registers

The μPD789167, 789177, 789167Y, and 789177Y Subseries provide the following on-chip processor registers:

5.2.1 Control registers

The control registers have special functions to control the program sequence statuses and stack memory. The control registers include a program counter, a program status word, and a stack pointer.

(1) Program counter (PC)

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

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

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

Figure 5-7. Program Counter Configuration

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

(2) Program status word (PSW)

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

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

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

Figure 5-8. Program Status Word Configuration

	7							0
PSW	IE	Z	0	AC	0	0	1	CY

(a) Interrupt enable flag (IE)

This flag controls interrupt request acknowledge operations of CPU.

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

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

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

(b) Zero flag (Z)

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

(c) Auxiliary carry flag (AC)

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

(d) Carry flag (CY)

This flag stores overflow and underflow that have occurred upon add/subtract instruction execution. It stores the shift-out value upon rotate instruction execution and functions as a bit accumulator during bit operation 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 5-9. Stack Pointer Configuration

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

The SP is decremented ahead of writing (saving) to the stack memory and is incremented after reading (restoring) from the stack memory.

Each stack operation saves/restores data as shown in Figures 5-10 and 5-11.

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

Figure 5-10. Data to be Saved to Stack Memory

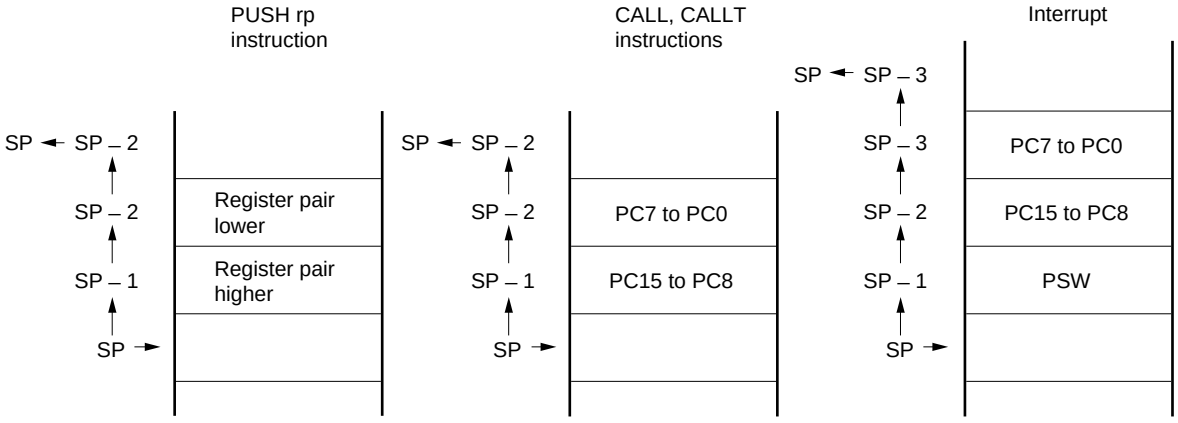
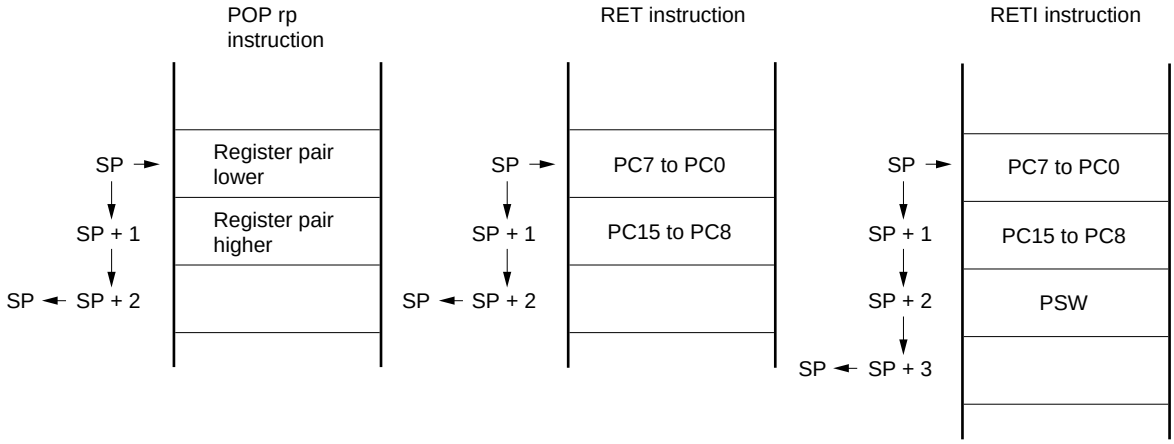


Figure 5-11. Data to be Restored from Stack Memory



5.2.2 General-purpose registers

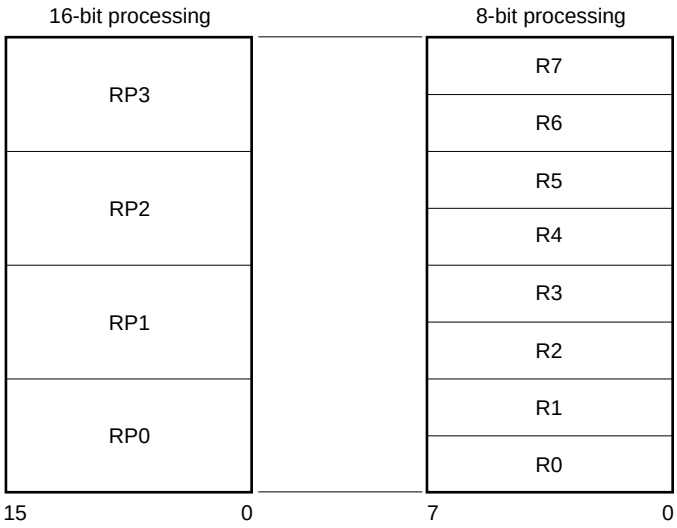
The general-purpose registers consist of eight 8-bit registers (X, A, C, B, E, D, L, and H).

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

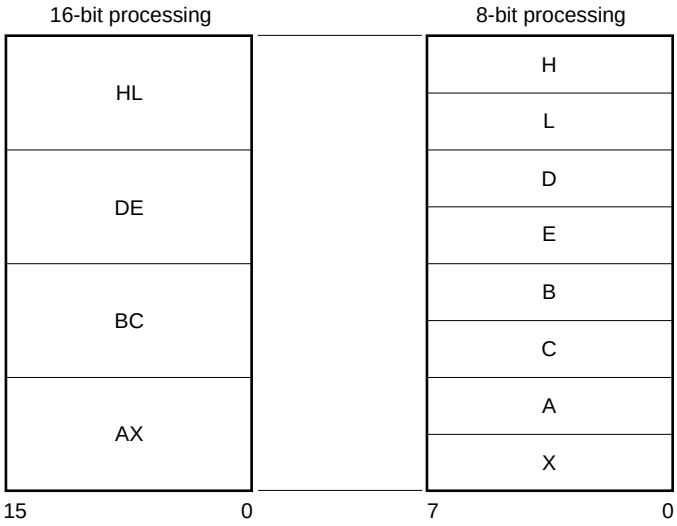
They can be described in terms of functional names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

Figure 5-12. General-Purpose Register Configuration

(a) Absolute Names



(b) Functional Names



5.2.3 Special function registers (SFR)

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

They are allocated to the 256-byte area FF00H to FFFFH.

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

Each manipulation bit unit can be specified as follows.

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

Table 5-3 lists the special function registers. The meanings of the symbols in this table are as follows:

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

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

- Bit units for manipulation
  Indicates the bit units (1, 8, and 16) in which the special function register can be manipulated.

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

Table 5-3. Special Function Registers (1/2)

Address	Special Function Register (SFR) Name	Symbol		R/W	Bit Units for Manipulation			After Reset
					1 Bit	8 Bits	16 Bits	
FF00H	Port 0	P0		R/W	√	√	–	00H
FF01H	Port 1	P1			√	√	–	
FF02H	Port 2	P2			√	√	–	
FF03H	Port 3	P3			√	√	–	
FF05H	Port 5	P5			√	√	–	
FF06H	Port 6	P6		R	√	√	–	
FF10H	16-bit multiplication result storage register 0	MUL0L	MUL0		–	–	√ Notes 2, 3	Undefined
FF11H		MUL0H						
FF14H	A/D conversion result register 0	ADCR0			–	√ Note 1	√ Note 2	
FF15H								
FF16H	16-bit compare register 90	CR90L	CR90	W	–	–	√ Notes 2, 3	FFFFH
FF17H		CR90H						
FF18H	16-bit timer counter 90	TM90L	TM90	R	–	–	√ Notes 2, 3	0000H
FF19H		TM90H						
FF1AH	16-bit capture register 90	TCP90L	TCP90		–	–	√ Notes 2, 3	Undefined
FF1BH		TCP90H						
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			√	√	–	
FF32H	Pull-up resistor option register B2	PUB2			√	√	–	00H
FF33H	Pull-up resistor option register B3	PUB3			√	√	–	
FF42H	Timer clock selection register 2	TCL2			–	√	–	
FF48H	16-bit timer mode control register 90	TMC90			√	√	–	
FF49H	Buzzer output control register 90	BZC90			√	√	–	
FF4AH	Watch timer mode control register	WTM			√	√	–	
FF50H	8-bit compare register 80	CR80		W	–	√	–	Undefined
FF51H	8-bit timer counter 80	TM80		R	–	√	–	00H
FF53H	8-bit timer mode control register 80	TMC80		R/W	√	√	–	

Notes 1. When using this register with an 8-bit A/D converter (μPD789167 or 789167Y Subseries), the register can be accessed in 8-bit units. At this time, the address is FF15H.
When using this register with a 10-bit A/D converter (μPD789177 or 789177Y Subseries), the register can be accessed only in 16-bit units. When the μPD78F9177, a flash memory counterpart of the μPD789166 or μPD789167, is used, the register can be accessed in 8-bit units. However, only an object file assembled with the μPD789166 or μPD789167 can be used. The same is also true for the μPD78F9177Y, a flash memory counterpart of the μPD789166Y or μPD789167Y. When the μPD78F9177Y is used, the register can be accessed in 8-bit units. However, only an object file assembled with the μPD789166Y and μPD789167Y can be used.

2. 16-bit access is allowed only with short direct addressing.

3. MUL0, CR90, TM90, and TCP90 are designed only for 16-bit access. With direct addressing, however, they can also be accessed in 8-bit mode.

Table 5-3. Special Function Registers (2/2)

	Address	Special Function Register (SFR) Name	Symbol	R/W	Bit Units for Manipulation			After Reset
					1 Bit	8 Bits	16 Bits	
	FF54H	8-bit compare register 81	CR81	W	–	√	–	Undefined
	FF55H	8-bit timer counter 81	TM81	R	–	√	–	00H
	FF57H	8-bit timer mode control register 81	TMC81	R/W	√	√	–	
	FF58H	8-bit compare register 82	CR82	W	–	√	–	Undefined
	FF59H	8-bit timer counter 82	TM82	R	–	√	–	00H
	FF5BH	8-bit timer mode control register 82	TMC82	R/W	√	√	–	
	FF70H	Asynchronous serial interface mode register 20	ASIM20		√	√	–	
	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		–	√	–	
	FF74H	Transmission shift register 20	TXS20, SIO20	W	–	√	–	FFH
		Reception buffer register 20	RXB20	R	–	√	–	Undefined
★	FF78H	SMB control register 0[Note]	SMBC0	R/W	√	√	–	00H
★	FF79H	SMB status register 0[Note]	SMBS0	R	√	√	–	
★	FF7AH	SMB clock selection register 0[Note]	SMBCL0	R/W	√	√	–	
★	FF7BH	SMB slave address register 0[Note]	SMBSVA0		√	√	–	
★	FF7CH	SMB mode register 0[Note]	SMBM0		√	√	–	20H
★	FF7DH	SMB input level setting register 0[Note]	SMBVI0		√	√	–	00H
★	FF7EH	SMB shift register 0[Note]	SMB0		√	√	–	
	FF80H	A/D converter mode register 0	ADM0		√	√	–	
	FF84H	A/D input selection register 0	ADS0		√	√	–	
	FFD0H	Multiplication data register A0	MRA0	W	√	√	–	Undefined
	FFD1H	Multiplication data register B0	MRB0		√	√	–	
	FFD2H	Multiplier control register 0	MULC0	R/W	√	√	–	00H
	FFE0H	Interrupt request flag register 0	IF0		√	√	–	
	FFE1H	Interrupt request flag register 1	IF1		√	√	–	
	FFE4H	Interrupt mask flag register 0	MK0		√	√	–	FFH
	FFE5H	Interrupt mask flag register 1	MK1		√	√	–	
	FFECH	External interrupt mode register 0	INTM0		–	√	–	00H
	FFEDH	External interrupt mode register 1	INTM1		–	√	–	
	FFF0H	Suboscillation mode register	SCKM		√	√	–	
	FFF2H	Subclock control register	CSS		√	√	–	
	FFF7H	Pull-up resistor option register 0	PU0		√	√	–	
	FFF9H	Watchdog timer mode register	WDTM		√	√	–	
	FFFAH	Oscillation stabilization time selection register	OSTS		–	√	–	04H
	FFFBH	Processor clock control register	PCC		√	√	–	02H

Note For the μPD789167Y and 789177Y Subseries only

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

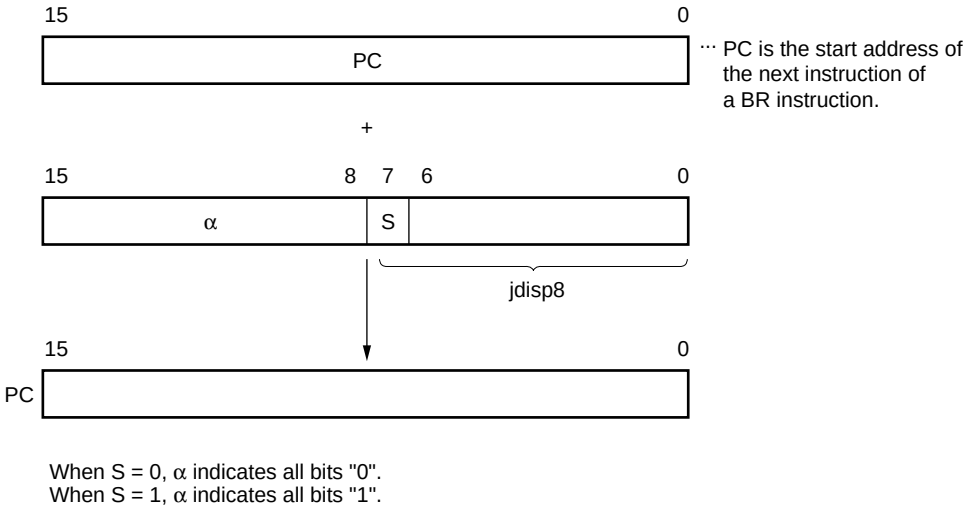
5.3.1 Relative addressing

[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) to branch. The displacement value is treated as signed two's complement data (–128 to +127) and bit 7 becomes a sign bit. In other words, the range of branch in relative addressing is between –128 and +127 of the start address of the following instruction.

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

[Illustration]



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

5.3.2 Immediate addressing

[Function]

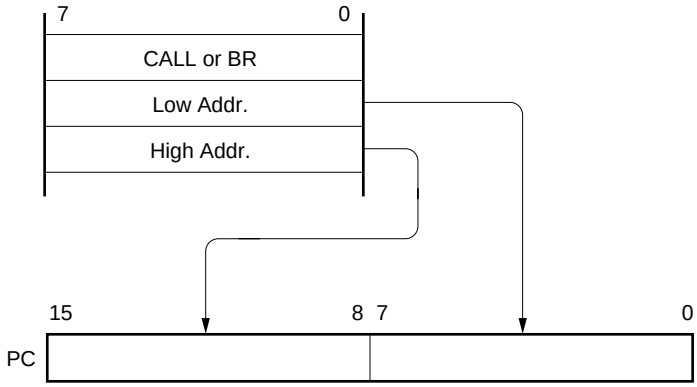
Immediate data in the instruction word is transferred to the program counter (PC) to branch.

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

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

[Illustration]

In case of CALL !addr16 and BR !addr16 instructions



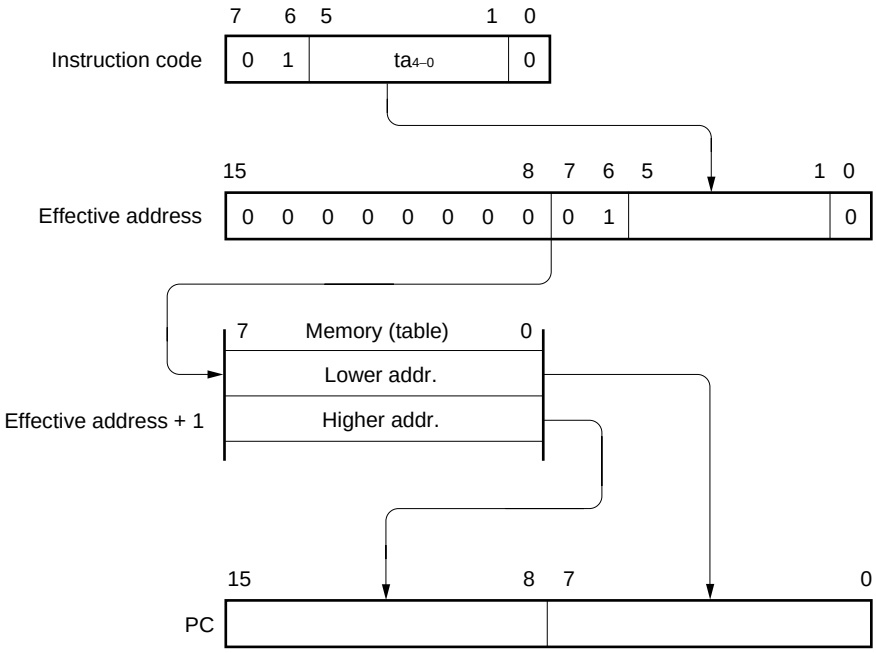
5.3.3 Table indirect addressing

[Function]

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

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

[Illustration]



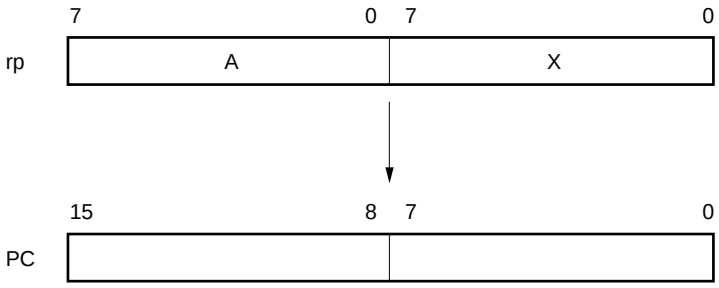
5.3.4 Register addressing

[Function]

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

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

[Illustration]



5.4 Operand Address Addressing

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

5.4.1 Direct addressing

[Function]

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

[Operand format]

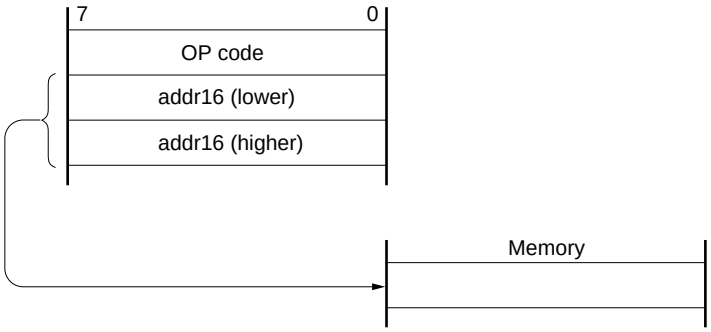
Identifier	Description
addr16	Label or 16-bit immediate data

[Description example]

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

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

[Illustration]



5.4.2 Short direct addressing

[Function]

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

The fixed space where this addressing is applied to is the 256-byte space FE20H to FF1FH. An internal high-speed RAM and a special function register (SFR) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively.

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

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

[Operand format]

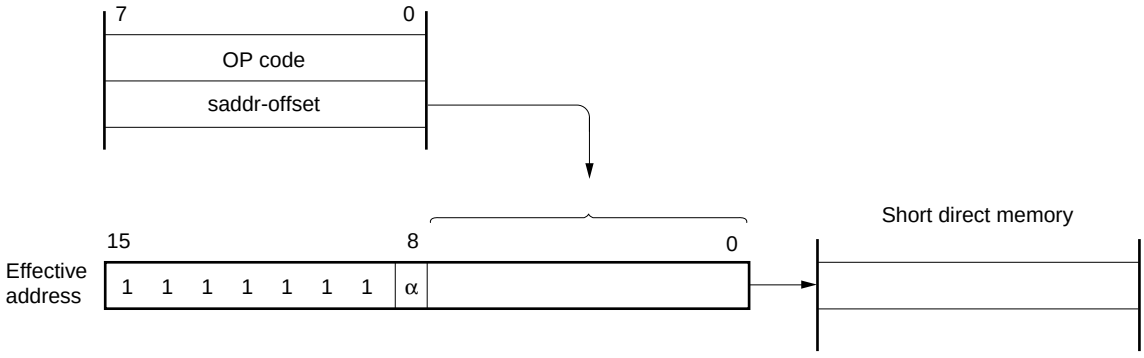
Identifier	Description
saddr	Label or FE20H to FF1FH immediate data
saddrp	Label or FE20H to FF1FH immediate data (even address only)

[Description example]

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

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

[Illustration]



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

5.4.3 Special function register (SFR) addressing

[Function]

The memory-mapped special function registers (SFR) 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 SFR mapped at FF00H to FF1FH can also be accessed with short direct addressing.

[Operand format]

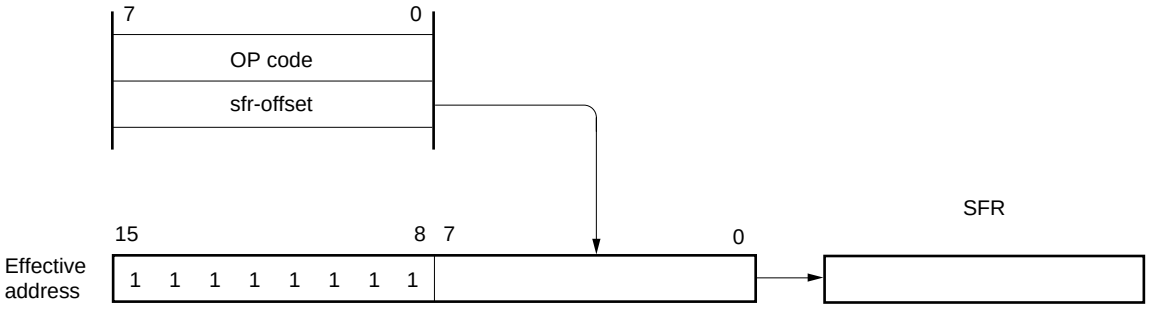
Identifier	Description
sfr	Special function register name

[Description example]

MOV PM0, A; When selecting PM0 for sfr

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

[Illustration]



5.4.4 Register addressing

[Function]

The general-purpose registers are accessed as operands. The general-purpose register to be accessed is specified with register specify code and functional name in the instruction code.

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

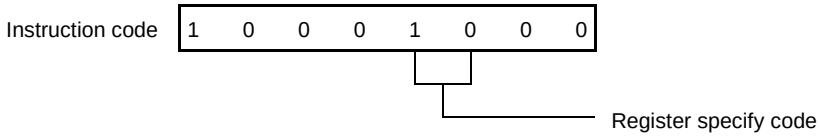
[Operand format]

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

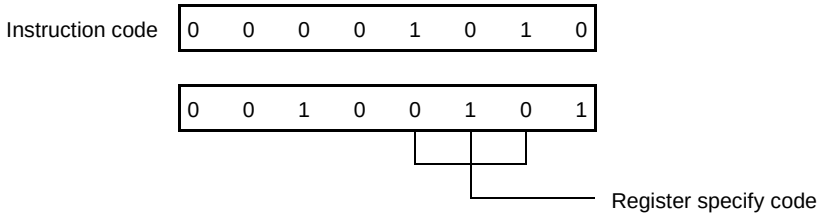
'r' and 'rp' can be described with absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

[Description example]

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



INCW DE; When selecting the DE register pair for rp



5.4.5 Register indirect addressing

[Function]

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

[Operand format]

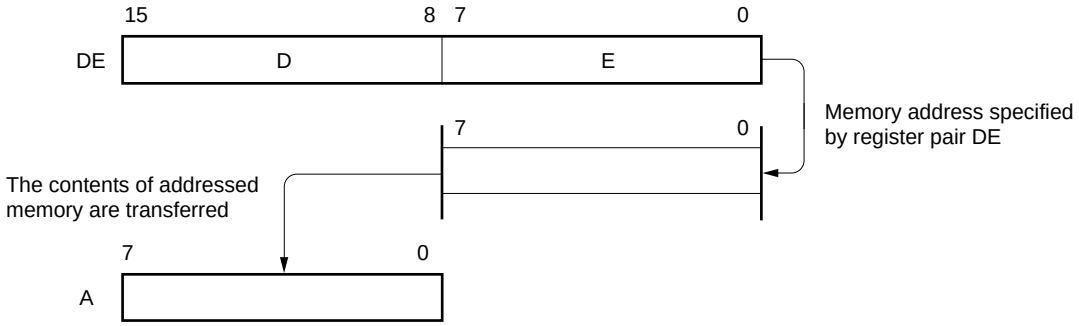
Identifier	Description
–	[DE], [HL]

[Description example]

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

Instruction code	0	0	1	0	1	0	1	1

[Illustration]



5.4.6 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

5.4.7 Stack addressing

[Function]

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

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

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

[Description example]

In the case of PUSH DE

Instruction code	1	0	1	0	1	0	1	0

[MEMO]

CHAPTER 6 PORT FUNCTIONS

6.1 Port Functions

The μPD789167, 789177, 789167Y, and 789177Y Subseries are provided with the ports shown in Figure 6-1. These ports are used to enable several types of control. Table 6-1 lists the functions of each port.

These ports, while originally designed as digital input/output ports, have alternate functions, as summarized in **3.1 Pin Function List** (μPD789167 and 789177 Subseries) and **4.1 Pin Function List** (μPD789167Y and 789177Y Subseries).

Figure 6-1. Port Types

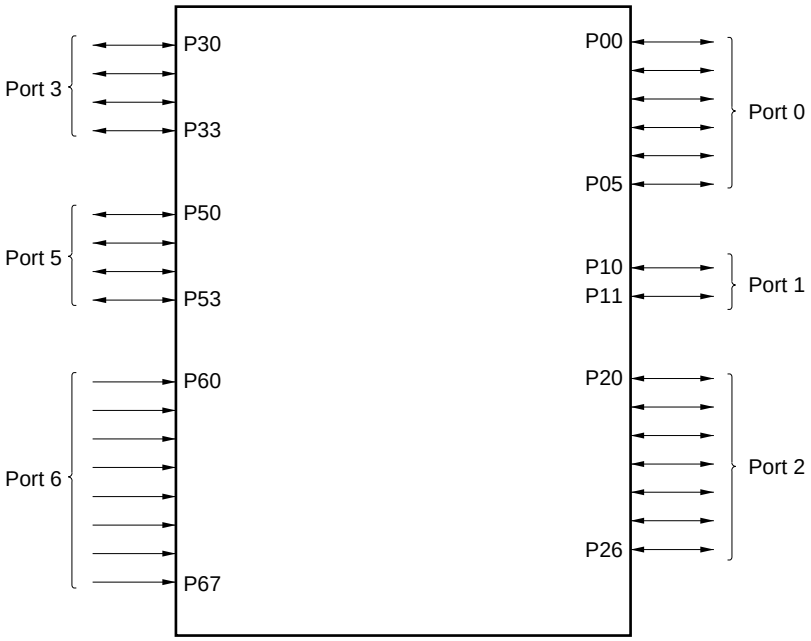


Table 6-1. Port Functions

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P05	I/O	Port 0 6-bit input/output port Input/output mode can be specified in 1-bit units When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register 0 (PU0).	Input	–
P10, P11	I/O	Port 1 2-bit input/output port Input/output mode 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 input/output port Input/output mode can be specified in 1-bit units For P20 to P22, P25, and P26, an on-chip pull-up resistor can be specified by means of pull-up resistor option register B2 (PUB2). Only P23 and P24 can be used as N-ch open-drain input/output port pins.	Input	$\overline{\text{SCK20}}$/ASCK20
P21				SO20/TxD20
P22				SI20/RxD20
P23				SCL0[Note]
P24				SDA0[Note]
P25				TI80/$\overline{\text{SS20}}$
P26				TO80
P30	I/O	Port 3 4-bit input/output port Input/output mode can be specified in 1-bit units An on-chip pull-up resistor can be specified by means of pull-up resistor option register B3 (PUB3).	Input	INTP0/TI81/CPT90
P31				INTP1/TO81
P32				INTP2/TO90
P33				INTP3/TO82/BZO90
P50 to P53	I/O	Port 5 4-bit N-ch open-drain input/output port Input/output mode 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 P67	Input	Port 6 8-bit input-only port	Input	ANI0 to ANI7

Note For the μPD789167Y and 789177Y Subseries only

6.2 Port Configuration

Ports have the following hardware configuration.

Table 6-2. Configuration of Port

Parameter	Configuration
Control register	Port mode registers (PMm: m = 0 to 3, 5) Pull-up resistor option register 0 (PU0) Pull-up resistor option registers B2, B3 (PUB2, PUB3)
Port	Total: 31 (CMOS input/output: 17, CMOS input: 8, N-ch open-drain input/output: 6)
Pull-up resistor	• Mask ROM versions Total: 21 (software control: 17, mask option control: 4) • Flash memory versions Total: 17 (software control only)

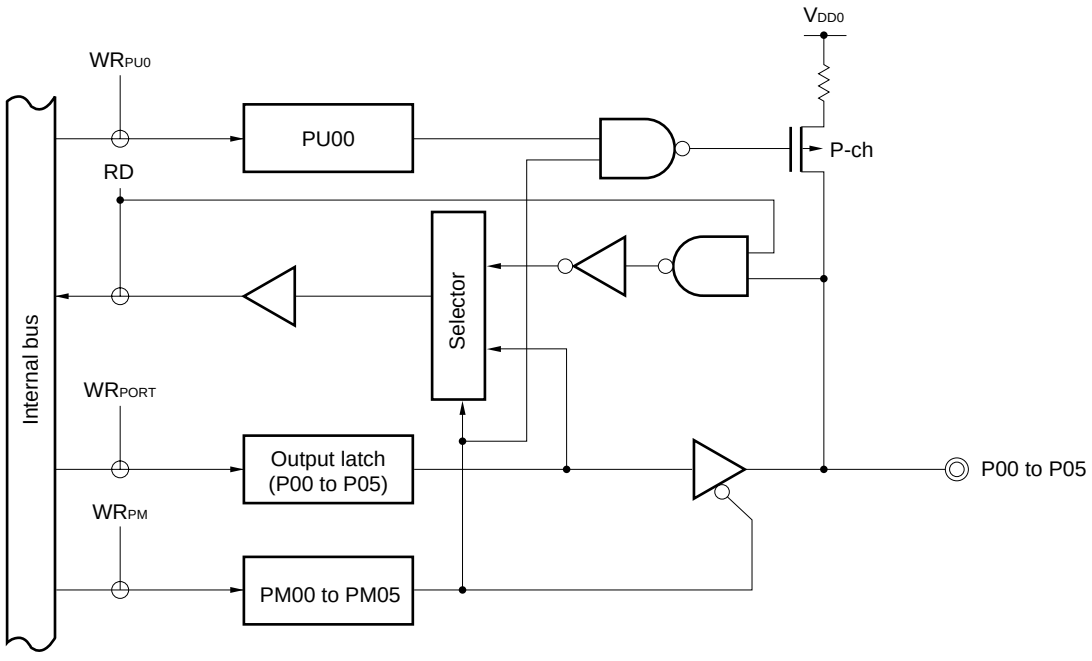
6.2.1 Port 0

This is a 6-bit I/O port with output latches. Port 0 can be set to input or output mode in 1-bit units by using port mode register 0 (PM0). When pins P00 to P05 are used as input port pins, on-chip pull-up resistors can be connected in 6-bit units by using pull-up resistor option register 0 (PU0).

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

Figure 6-2 shows a block diagram of port 0.

Figure 6-2. Block Diagram of P00 to P05



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

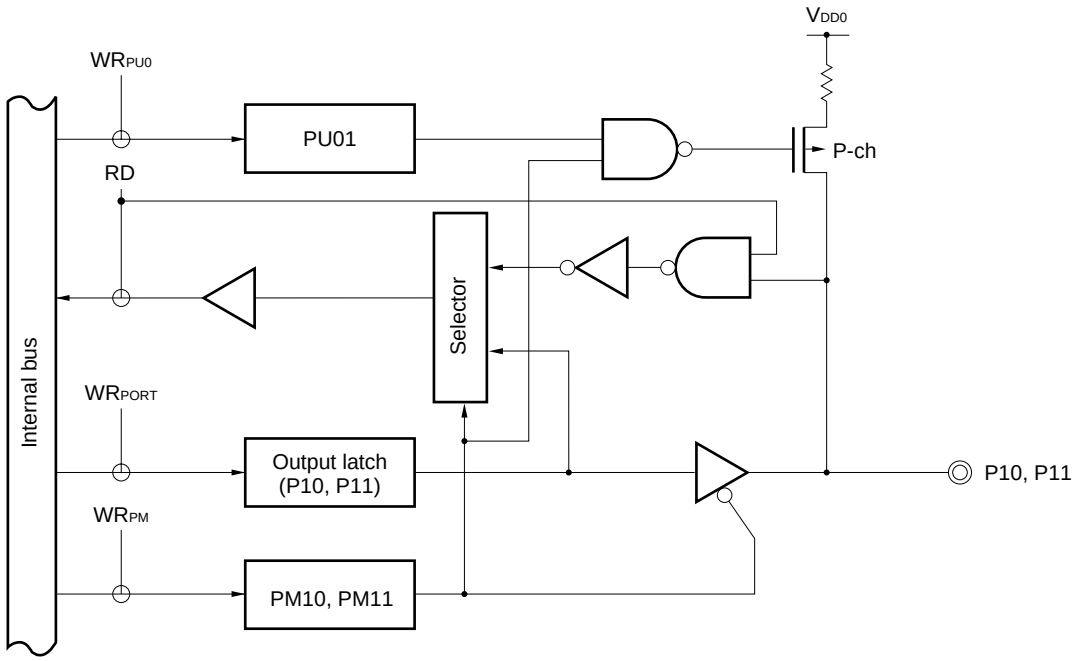
6.2.2 Port 1

This is a 2-bit I/O port with output latches. Port 1 can be set to input or output mode in 1-bit units by using the port mode register 1 (PM1). When the P10 and P11 pins are used 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).

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

Figure 6-3 shows a block diagram of port 1.

Figure 6-3. 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

6.2.3 Port 2

This is a 7-bit I/O port with output latches. Port 2 can be set to input or output mode in 1-bit units by using port mode register 2 (PM2). For pins P20 to P22, P25, and P26, on-chip pull-up resistors can be connected in 1-bit units by using pull-up resistor option register B2 (PUB2).

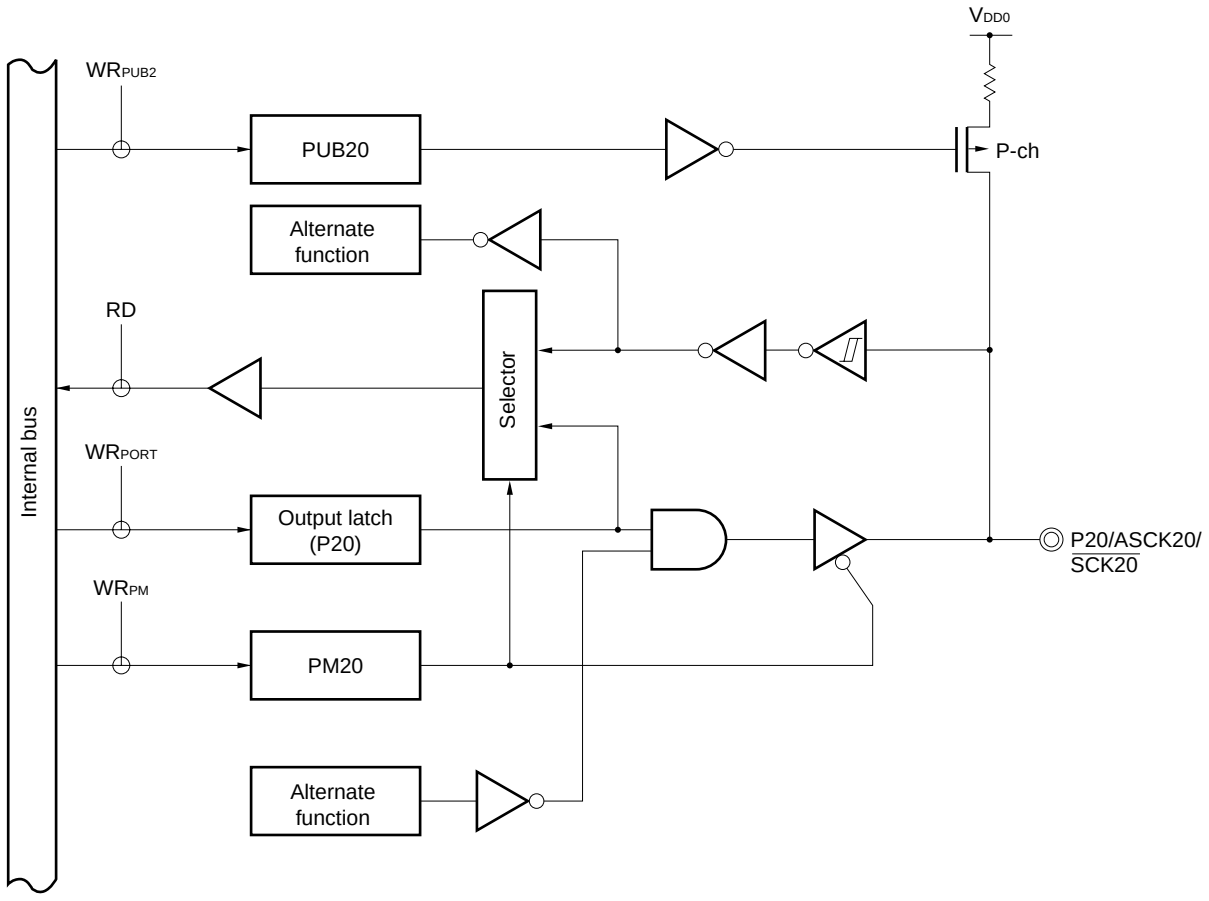
The port is also used as a data I/O and clock I/O to and from the serial interface, and timer I/O.

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

Figures 6-4 through 6-8 show block diagrams of port 2.

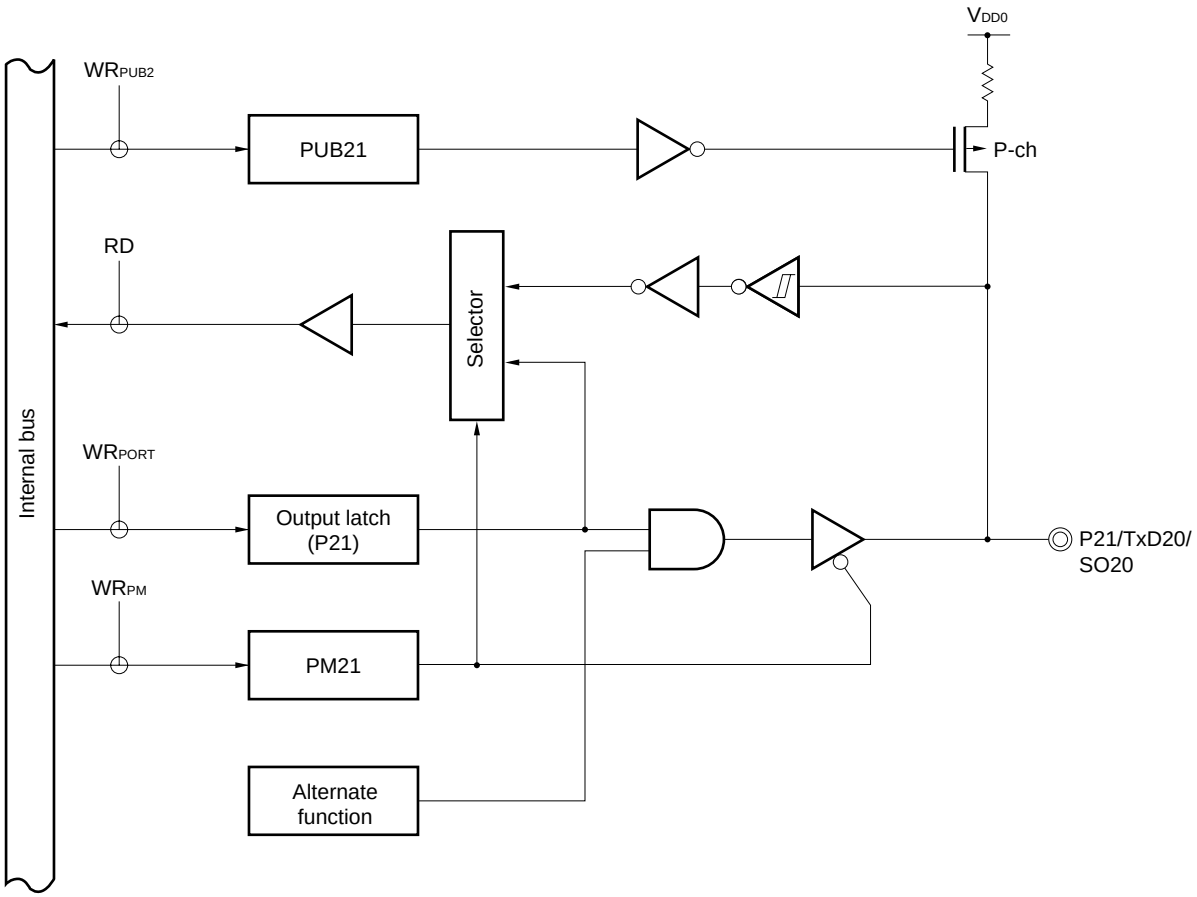
Caution When using the pins of port 2 as the serial interface, the I/O and output latches must be set according to the function to be used. For details of the settings, see Table 14-2 Serial Interface 20 Operating Mode Settings.

Figure 6-4. Block Diagram of P20



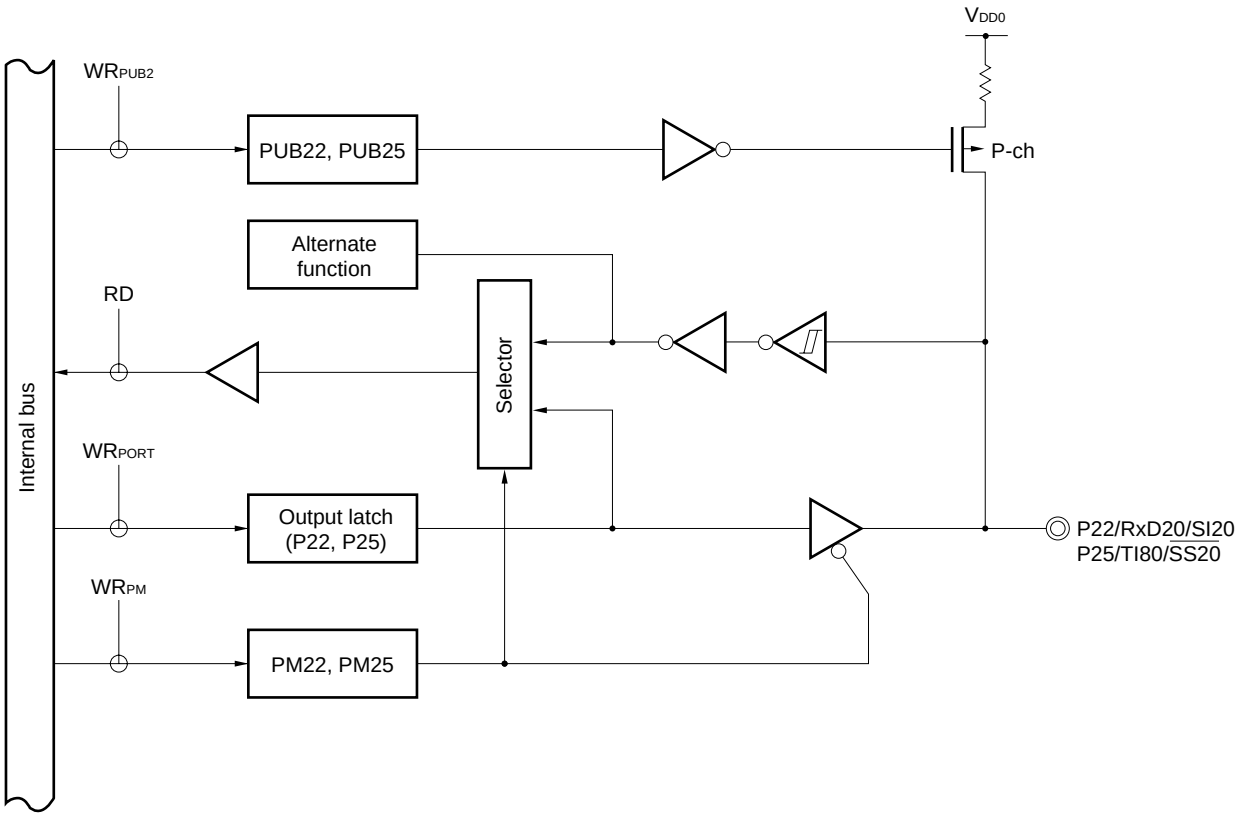
PUB2: Pull-up resistor option register B2
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

★ **Figure 6-5. Block Diagram of P21**



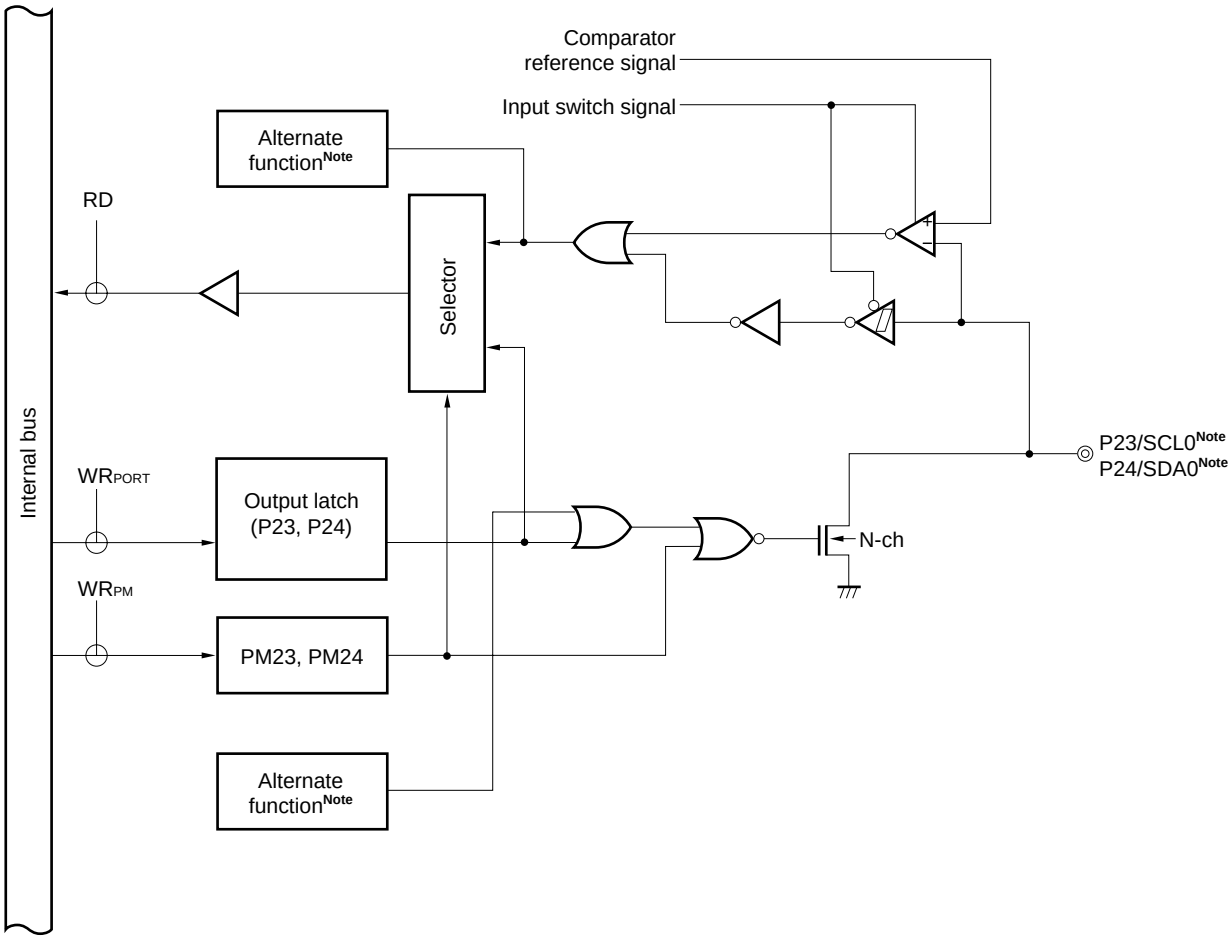
PUB2: Pull-up resistor option register B2
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

Figure 6-6. Block Diagram of P22 and P25



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

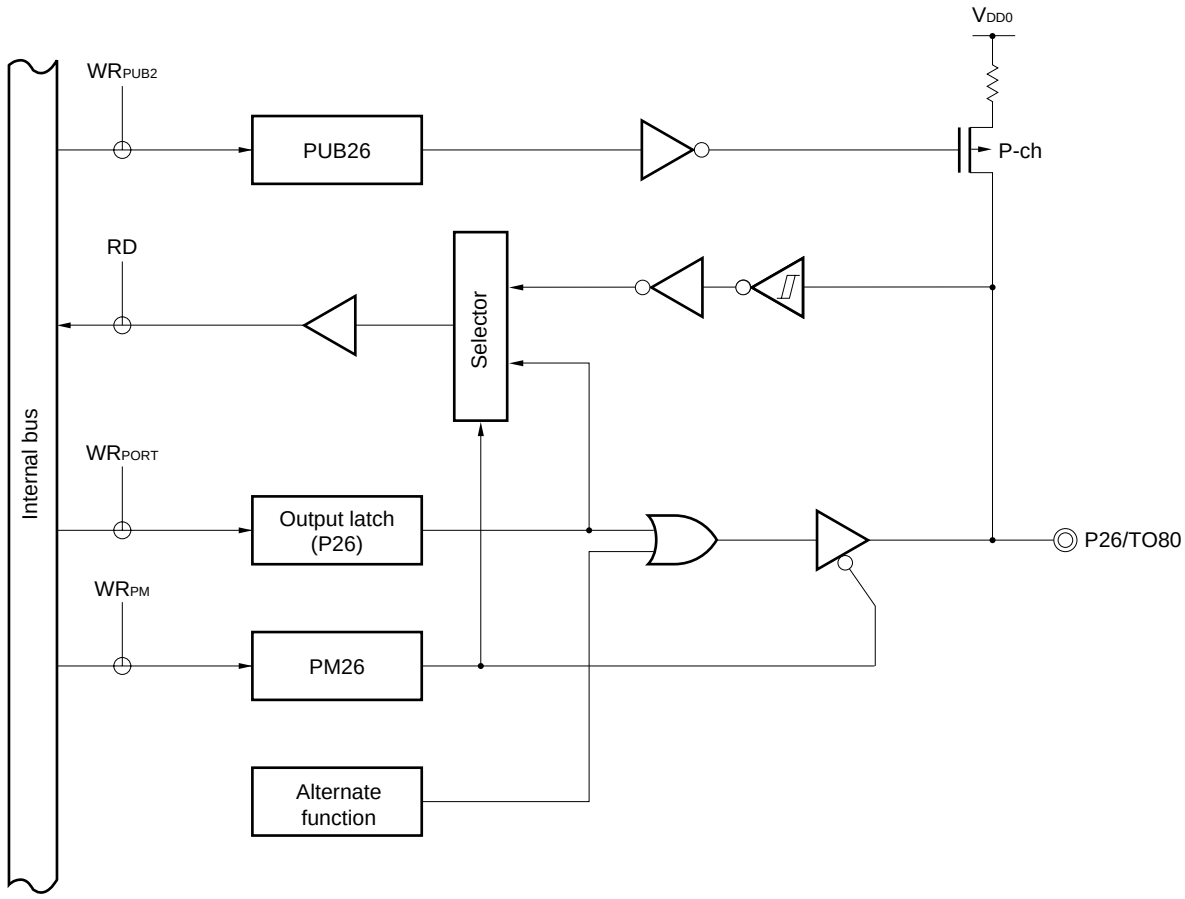
Figure 6-7. Block Diagram of P23 and P24



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

Note This function is provided for the μPD789167Y and 789177Y Subseries only. For the μPD789167 and 789177 Subseries, P23 and P24 cannot be used as alternate pins.

Figure 6-8. Block Diagram of P26



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

6.2.4 Port 3

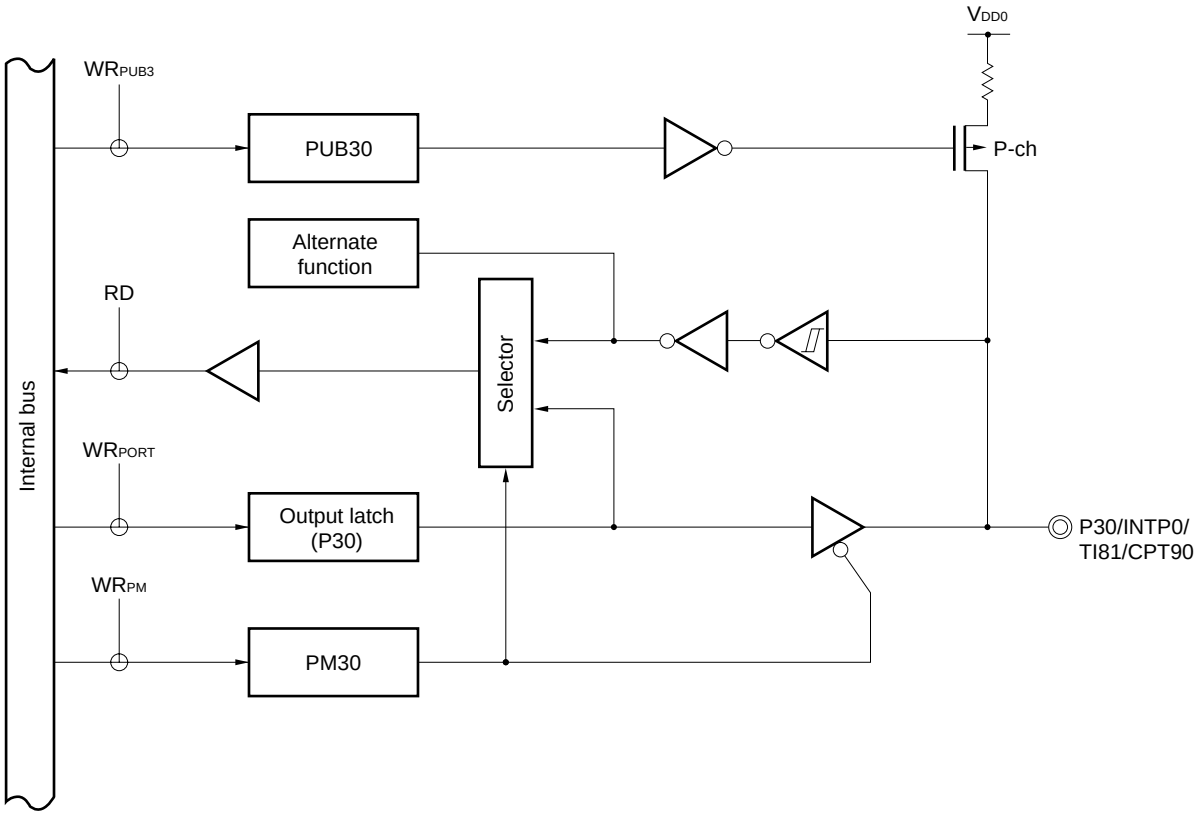
This is a 4-bit I/O port with output latches. Port 3 can be set to input or output mode in 1-bit units by using port mode register 3 (PM3). For pins P30 to P33, on-chip pull-up resistors can be connected in 1-bit units by using pull-up resistor option register B3 (PUB3).

The port is also used as an external interrupt input, capture input, timer output, and buzzer output.

$\overline{\text{RESET}}$ input sets port 3 to input mode.

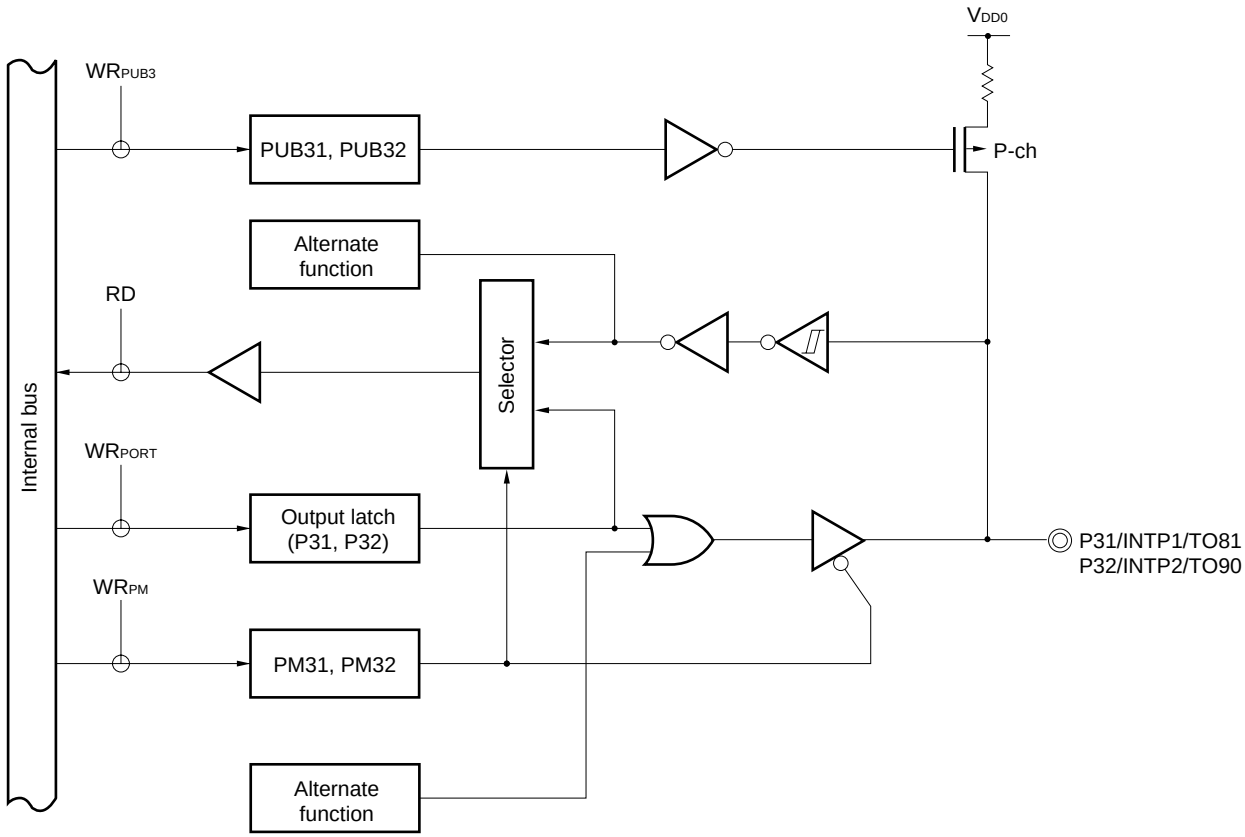
Figures 6-9 through 6-11 show block diagrams of port 3.

Figure 6-9. Block Diagram of P30



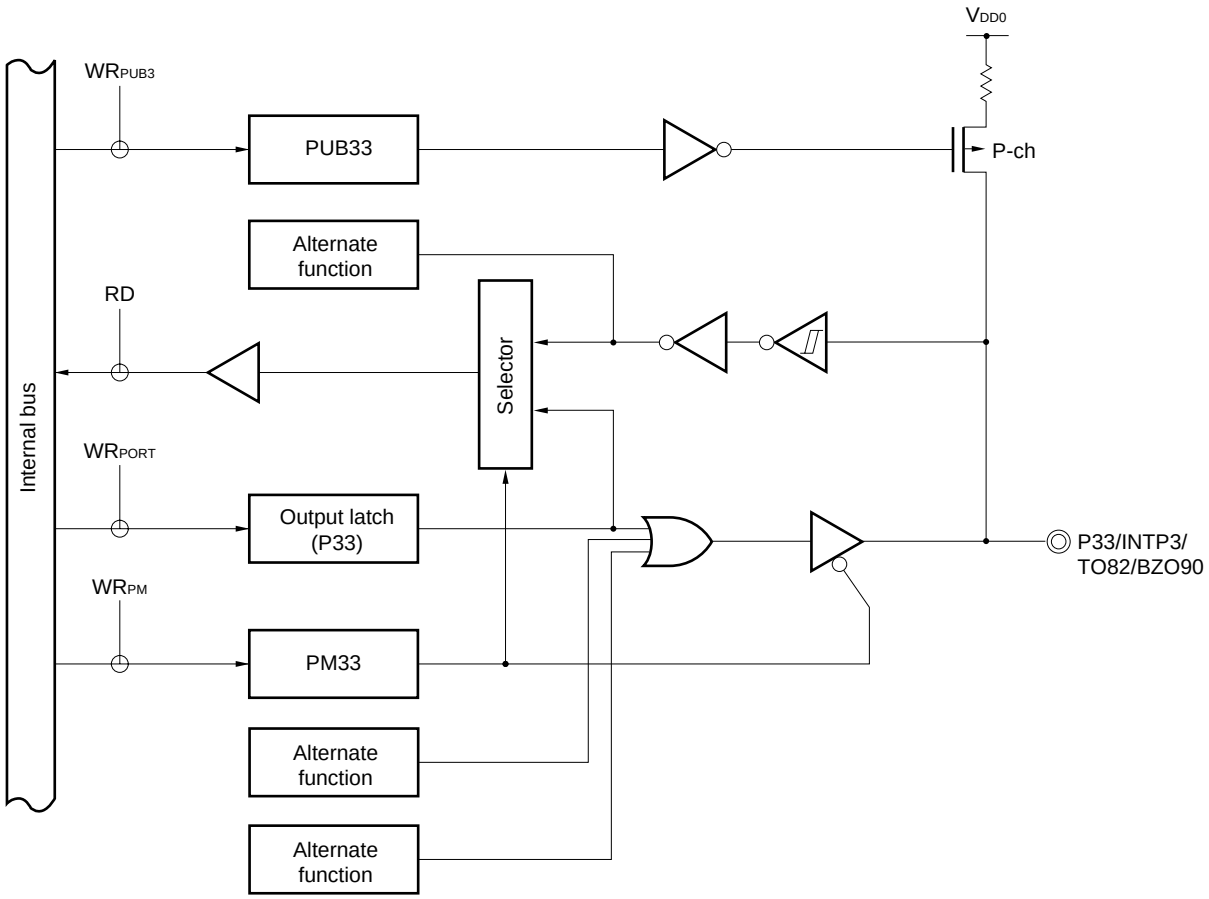
PUB3: Pull-up resistor option register B3
PM: Port mode register
RD: Port 3 read signal
WR: Port 3 write signal

Figure 6-10. Block Diagram of P31 and P32



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

Figure 6-11. Block Diagram of P33



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

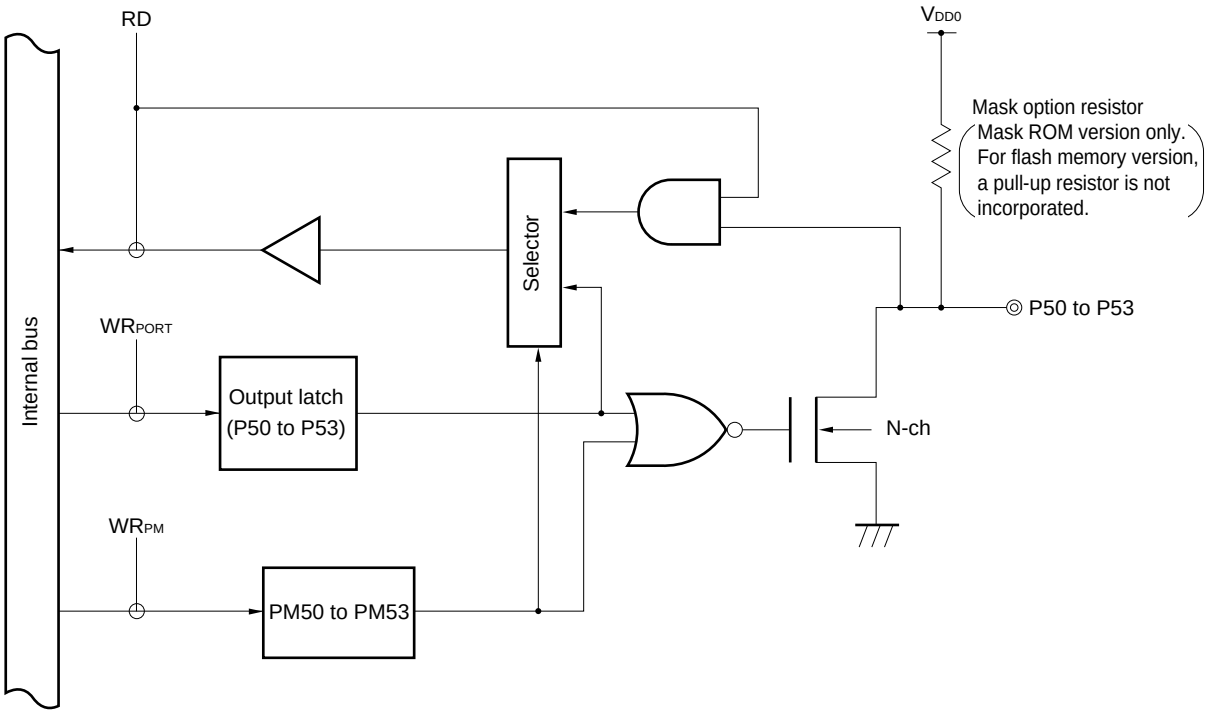
6.2.5 Port 5

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

$\overline{\text{RESET}}$ input sets port 5 to input mode.

Figure 6-12 shows a block diagram of port 5.

Figure 6-12. Block Diagram of P50 to P53



PM: Port mode register

RD: Port 5 read signal

WR: Port 5 write signal

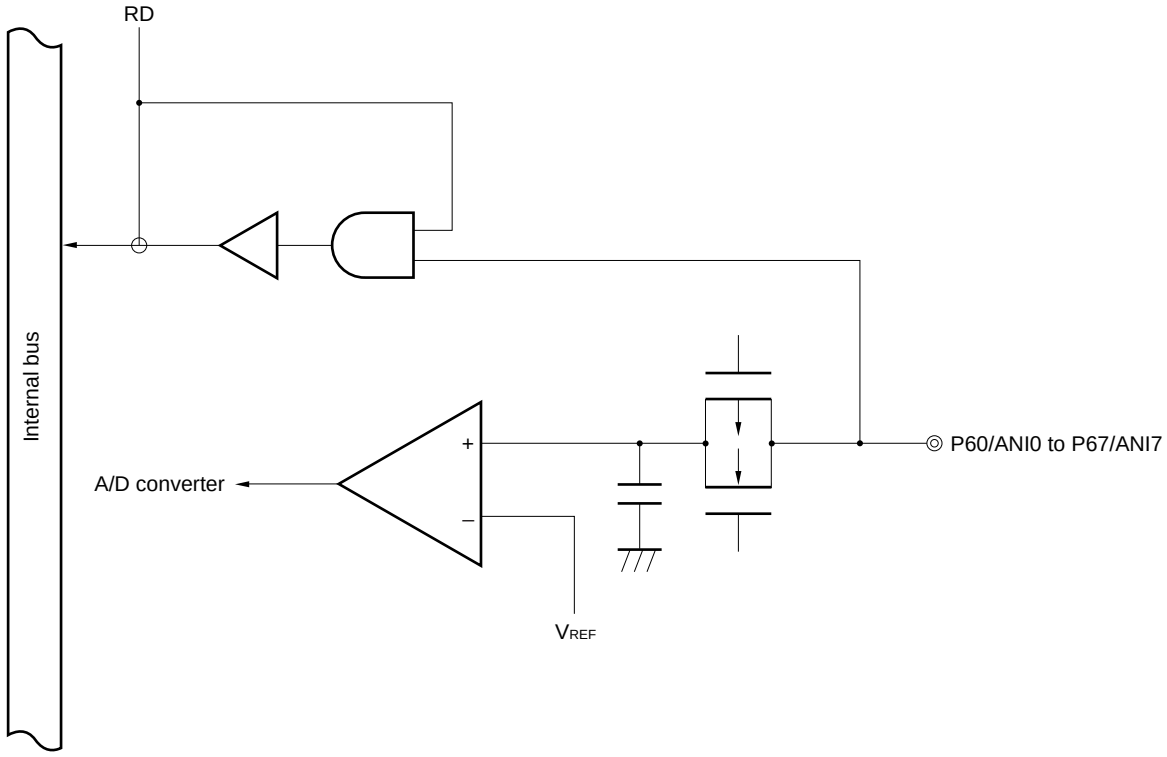
6.2.6 Port 6

This is an 8-bit input port.

The port is also used as an analog input to the A/D converter.

Figure 6-13 shows a block diagram of port 6.

Figure 6-13. Block Diagram of P60 to P67



6.3 Port Function Control Registers

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

- Port mode registers (PM0 to PM3, and PM5)
- Pull-up resistor option registers (PU0, PUB2, and PUB3)

(1) Port mode registers (PM0 to PM3, and PM5)

The port mode registers separately set each port bit to either input or output.

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

$\overline{\text{RESET}}$ input writes FFH into the port mode registers.

When port pins are used for alternate functions, the corresponding port mode register and output latch must be set or reset as described in Table 6-3.

Caution When port 3 is acting as an output port, and its output level is changed, an interrupt request flag is set, because this port is also used as the input for an external interrupt. To use port 3 in output mode, therefore, the interrupt mask flag must be set to 1 in advance.

Figure 6-14. Format of Port Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	1	1	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM1	1	1	1	1	1	1	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

PMmn	Pmn pin input/output mode selection (m = 0 : n = 0 to 5, m = 1 : n = 0, 1 m = 2 : n = 0 to 6, m = 3 : n = 0 to 3 m = 5 : n = 0 to 3)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

Table 6-3. Port Mode Register and Output Latch Settings for Using Alternate Functions

Pin Name	Alternate Function		PM××	P××
	Name	Input/Output		
P25	TI80	Input	1	×
P26	TO80	Output	0	0
P30	INTP0	Input	1	×
	TI81	Input	1	×
	CPT90	Input	1	×
P31	INTP1	Input	1	×
	TO81	Output	0	0
P32	INTP2	Input	1	×
	TO90	Output	0	0
P33	INTP3	Input	1	×
	TO82	Output	0	0
	BZO90	Output	0	0
P60 to P67	ANI0 to ANI7	Input	1	×

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 details of the settings, see Table 14-2 Serial Interface 20 Operating Mode Settings.

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

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

The pull-up resistor option register (PU0) sets whether an on-chip pull-up resistor on each port is used. On the port which is specified to use the on-chip pull-up resistor in PU0, the pull-up resistor can be internally used only for the bits set to input mode. No on-chip pull-up resistors can be used for the bits set to output mode regardless of the setting of PU0. On-chip pull-up resistors cannot be used even when the pins are used as the alternate-function output pins.

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

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

Figure 6-15. 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 all be set to 0.

(3) Pull-up resistor option registers B2 and B3 (PUB2 and PUB3)

These registers specify whether an on-chip pull-up resistor is connected to each pin of ports 2 and 3. The pin specified by PUB2 or PUB3 is connected to on-chip pull-up resistor regardless of the setting of the port mode register.

PUB2 and PUB3 are set with a 1-bit or 8-bit manipulation instruction.

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

Figure 6-16. 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	0	0	PUB22	PUB21	PUB20	FF32H	00H	R/W

PUB2n	P2n on-chip pull-up resistor selection (n = 0 to 2, 5, 6)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

Caution Bits 3, 4, and 7 must all be set to 0.

Figure 6-17. 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

Caution Bits 4 to 7 must all be set to 0.

6.4 Operation of Port Functions

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

6.4.1 Writing to I/O port

(1) In output mode

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

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

(2) In input mode

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

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

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

6.4.2 Reading from I/O port

(1) In output mode

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

(2) In input mode

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

6.4.3 Arithmetic operation of I/O port

(1) In output mode

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

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

(2) In input mode

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

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

[MEMO]

CHAPTER 7 CLOCK GENERATION CIRCUIT

7.1 Clock Generation Circuit Functions

The clock generation circuit 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 setting the suboscillation mode register (SCKM).

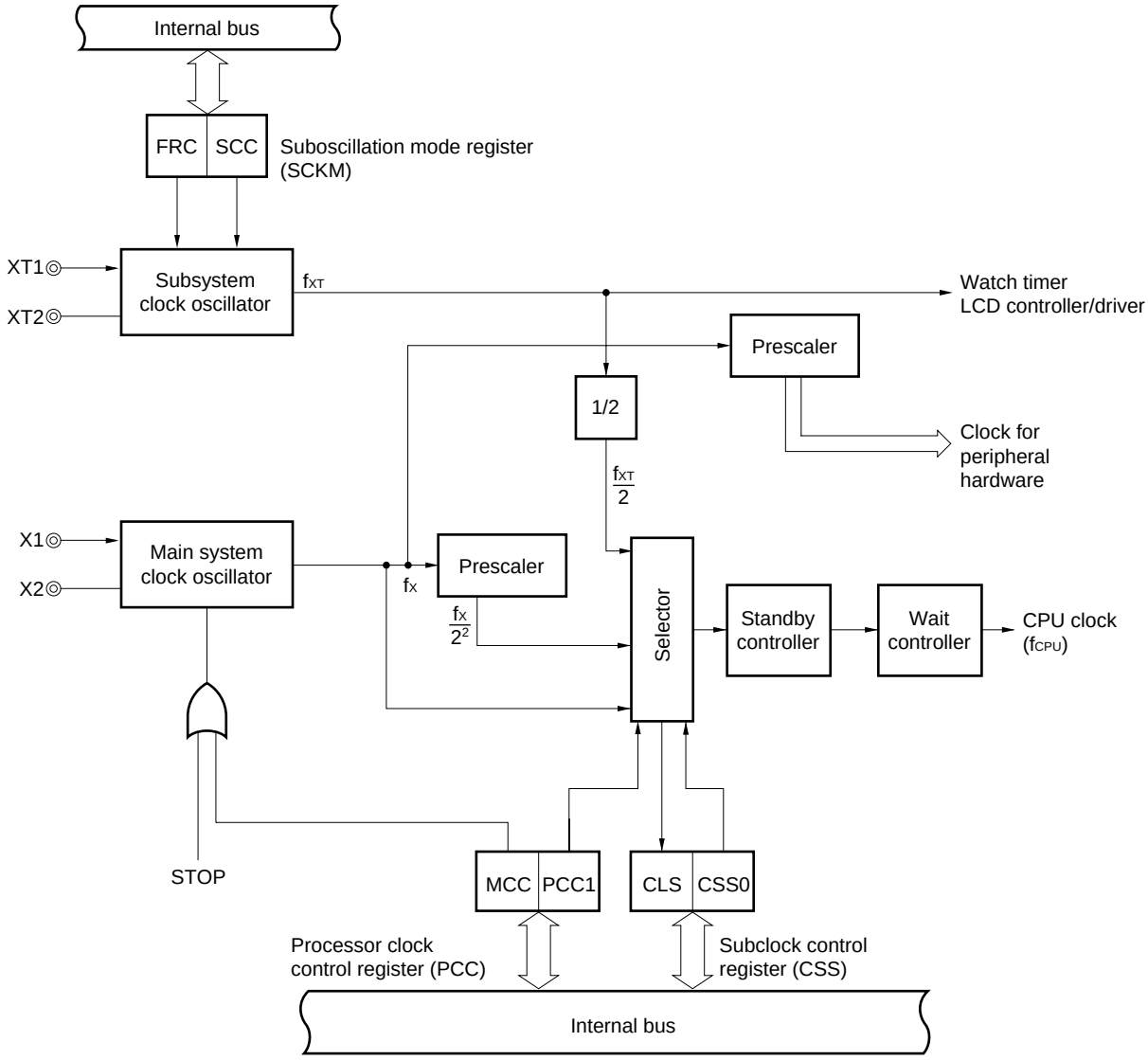
7.2 Clock Generation Circuit Configuration

The clock generation circuit consists of the following items of hardware.

Table 7-1. Configuration of Clock Generation Circuit

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

Figure 7-1. Block Diagram of Clock Generation Circuit



7.3 Registers Controlling Clock Generation Circuit

The clock generation circuit 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 selects the CPU clock and the ratio of division.

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

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

Figure 7-2. 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). See **7.3 (3) Subclock control register (CSS)**.

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

2. 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: 2 f_{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 specifies whether to use 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 clears SCKM to 00H.

Figure 7-3. 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	Use of feedback resistor
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 all be set to 0.

(3) Subclock control register (CSS)

CSS specifies whether the main system or subsystem clock oscillator is to be used. It also specifies how the CPU clock operates.

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

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

Figure 7-4. 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 (divided) main system clock
1	Operation based on the subsystem clock

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

Note Bit 5 is read-only.

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

7.4 System Clock Oscillators

7.4.1 Main system clock oscillator

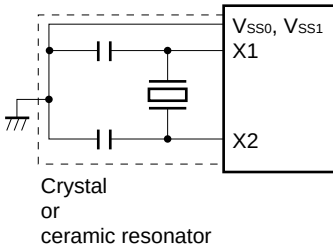
The main system clock oscillator is oscillated by the crystal or ceramic resonator (5.0 MHz TYP.) connected across the X1 and X2 pins.

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

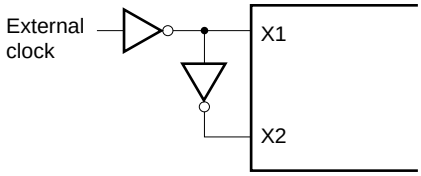
Figure 7-5 shows the external circuit of the main system clock oscillator.

Figure 7-5. External Circuit of Main System Clock Oscillator

(a) Crystal or ceramic oscillation



(b) External clock



7.4.2 Subsystem clock oscillator

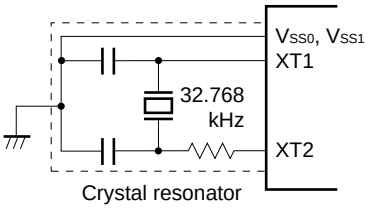
The subsystem clock oscillator is oscillated by the crystal resonator (32.768 kHz TYP.) connected across the XT1 and XT2 pins.

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

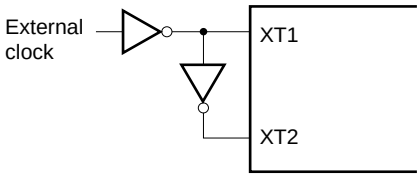
Figure 7-6 shows the external circuit of the subsystem clock oscillator.

Figure 7-6. External Circuit of Subsystem Clock Oscillator

(a) Crystal oscillation



(b) External clock



Caution When using the main system or subsystem clock oscillator, wire in the area enclosed by the broken line in Figures 7-5 and 7-6 as follows to avoid an adverse effect from wiring capacitance.

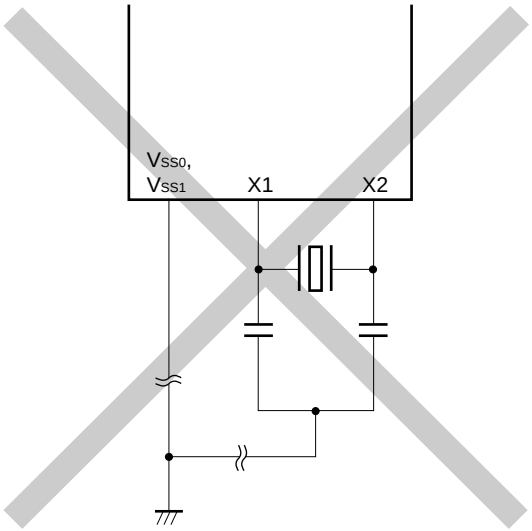
- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS0} and V_{SS1}. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

The subsystem clock oscillator is designed as low-amplitude circuit for reducing current consumption. Particular care is therefore required with the wiring method when the subsystem clock is used.

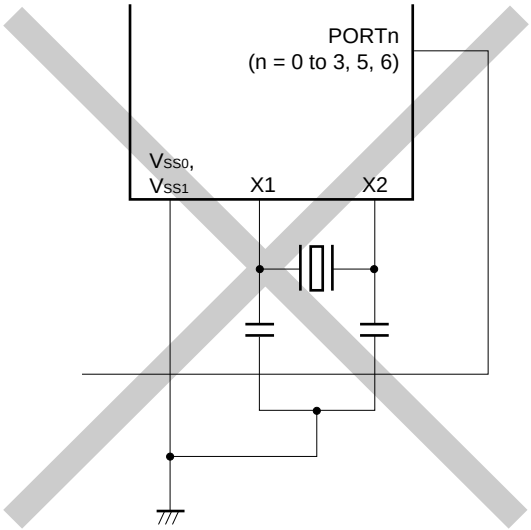
Figure 7-7 shows examples of incorrect oscillator connections.

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

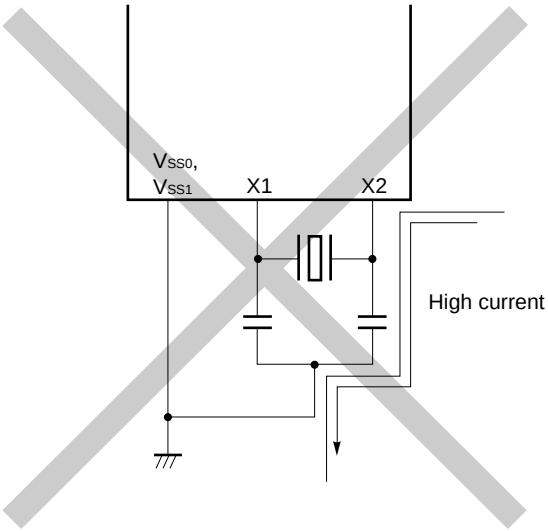
(a) Wiring too long



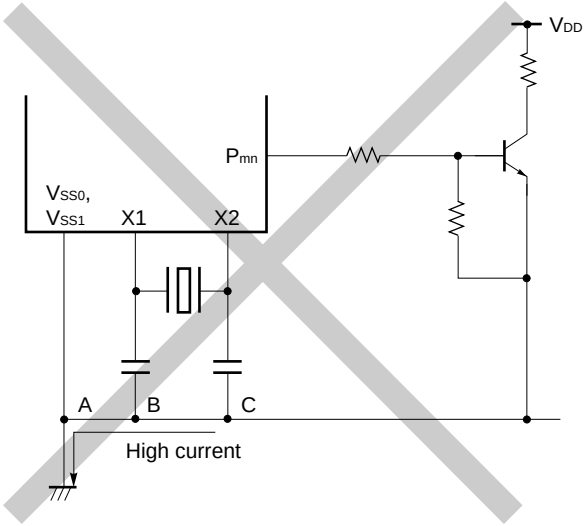
(b) Crossed signal line



(c) Wiring near high alternating current



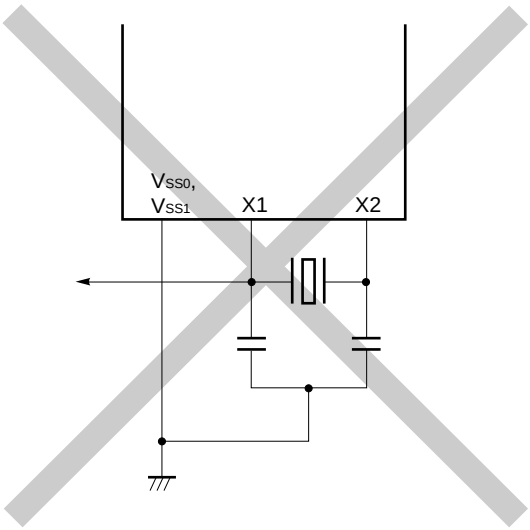
(d) Current flowing through ground line of oscillator (potential at points A, B, and C fluctuates)



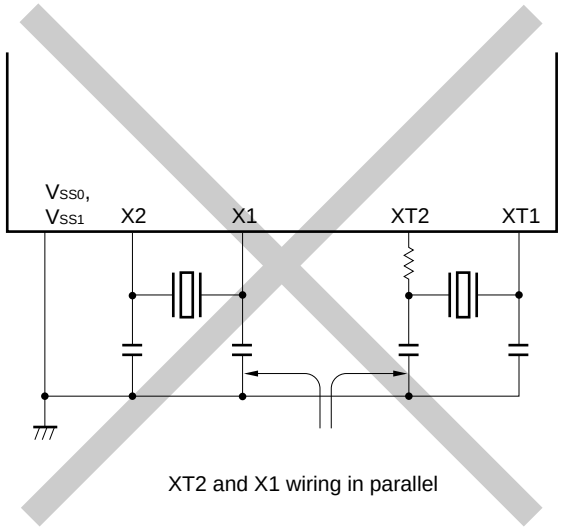
Remark When using the subsystem clock, read X1 and X2 as XT1 and XT2, respectively, and connect a resistor to the XT2 pin in series.

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

(e) Signals are fetched



(f) Signal conductors of the main and subsystem clocks are parallel and near to each other



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

7.4.3 Scaler

The scaler divides the main system clock oscillator output (f_X) and generates clocks.

7.4.4 When no subsystem clocks are used

If it is not necessary to use subsystem clocks for low power consumption operations and watch operations, connect the XT1 and XT2 pins as follows.

XT1: Connect to V_{SS0} or V_{SS1}
XT2: Open

In this state, however, some current may leak via the internal feedback resistor of the subsystem clock oscillator when the main system clock stops. To minimize the leakage current, the internal feedback resistor can be removed by setting bit 1 (FRC) of the suboscillation mode register (SCKM). In this case, also connect the XT1 and XT2 pins as described above.

7.5 Clock Generation Circuit Operation

The clock generation circuit generates the following clocks and controls operation modes of the CPU, such as standby mode:

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

The operation of the clock generation circuit is determined by the processor clock control register (PCC), suboscillation mode register (SCKM), and subclock control register (CSS), as follows:

(a) The slow mode 2 f_{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 SCKM so that the built-in feedback resistor cannot be used reduces current drain during 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 drain 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 by using bit 7 (MCC) of PCC. HALT mode can be used, but STOP mode cannot.

(f) The clock for the peripheral hardware is generated by dividing the frequency of the main system clock. The subsystem clock is supplied to 16-bit timer 90, 8-bit timer 82, and the watch timer only. So, even in standby mode, 16-bit timer 90, 8-bit timer 82, and watch function can keep running. The other hardware stops when the main system clock stops, because it runs based on the main system clock (except for an external clock).

7.6 Changing Setting of System Clock and CPU Clock

7.6.1 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 7-2).

Table 7-2. Maximum Time Required for Switching CPU Clock

Set Value before Switching		Set Value after Switching					
CSS0	PCC1	CSS0	PCC1	CSS0	PCC1	CSS0	PCC1
		0	0	0	1	1	×
0	0			4 clocks		$2f_X/f_{XT}$ clocks (306 clocks)	
	1	2 clocks				$f_X/2f_{XT}$ clocks (76 clocks)	
1	×	2 clocks		2 clocks			

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

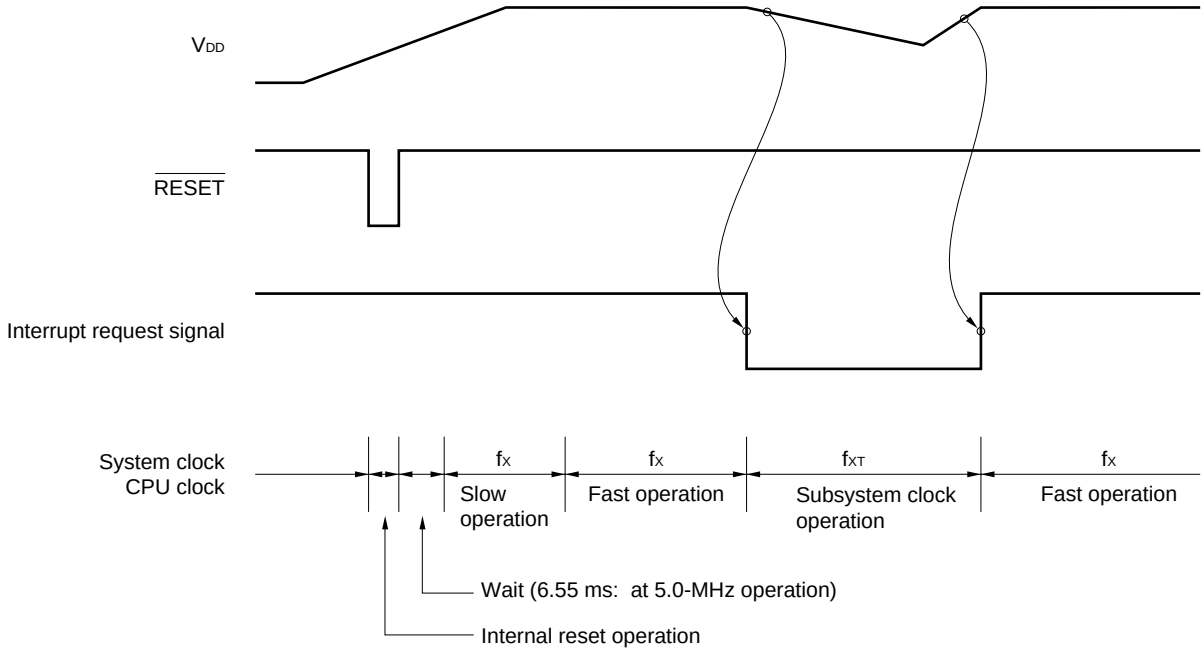
2. The parenthesized values apply to operation at f_X = 5.0 MHz or f_{XT} = 32.768 kHz.

3. ×: don't care

7.6.2 Switching between system clock and CPU clock

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

Figure 7-8. 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 time during which oscillation stabilizes ($2^{15}/f_X$) 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 the high speed has elapsed, bit 1 (PCC1) of the processor clock control register (PCC) and bit 4 (CSS0) of the subclock control register (CSS0) are rewritten so that the high speed operation can be selected.

<3> When 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 stabilized status.)

<4> When a recover of the V_{DD} voltage is detected with an interrupt request signal, bit 7 (MCC) of PCC is set to 0 to make the main system clock start 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 subsystem clock is operating, allow sufficient time for the oscillation to stabilize by coding the program before switching again from the subsystem clock to the main system clock.

CHAPTER 8 16-BIT TIMER

8.1 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.

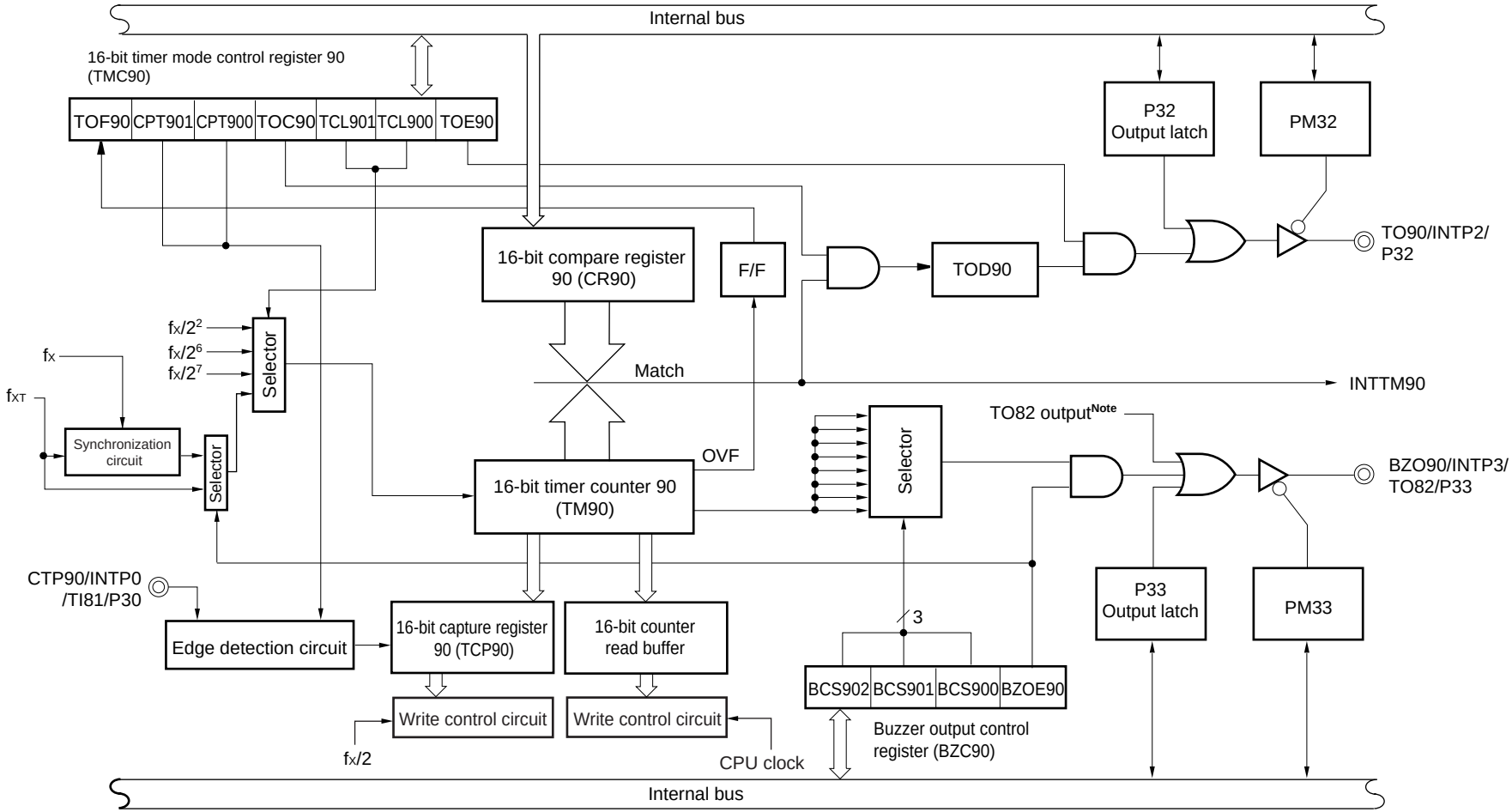
8.2 16-Bit Timer Configuration

The 16-bit timer consists of the following hardware.

Table 8-1. Configuration of 16-Bit Timer

Item	Configuration
Timer counter	16 bits × 1 (TM90)
Register	Compare register: 16 bits × 1 (CR90) Capture register: 16 bits × 1 (TCP90)
Timer output	1 (TO90)
Control register	16-bit timer mode control register 90 (TMC90) Buzzer output control register 90 (BZC90) Port mode register 3 (PM3)

Figure 8-1. Block Diagram of 16-Bit Timer



Note See **Figure 9-3 Block Diagram of 8-Bit Timer 82**.

(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. It can also be manipulated with 8-bit memory manipulation instructions, however. When an 8-bit memory manipulation instruction is used to set CR90, it must be accessed in direct addressing.

2. To re-set CR90 during 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 clears TM90 to 0000H.

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

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

3. When an 8-bit memory manipulation instruction is used to manipulate TM90, the lower and upper 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. It can also be manipulated with 8-bit memory manipulation instructions, however. When an 8-bit memory manipulation instruction is used to manipulate TCP90, it must be accessed in direct addressing.

(4) 16-bit counter read buffer 90

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

8.3 Registers Controlling 16-Bit Timer

The following three types of registers control the 16-bit timer.

- 16-bit timer mode control register 90 (TMC90)
- Buzzer output control register 90 (BZC90)
- Port mode register 3 (PM3)

(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 clears TMC90 to 00H.

Figure 8-2. 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 of 0
1	Timer output of 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 at the CPT90 pin
1	0	Captured at the falling edge at the CPT90 pin
1	1	Captured at both the rising and falling edges at 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 the interrupt 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 a square wave.

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

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

Figure 8-3. 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 1

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 2

Notes 1. Bits 4 to 7 must all be set to 0.

2. When setting BZOE90 to 1, TOE82 must be set to 0. (See **Figure 9-6 Format of 8-Bit Timer Mode Control Register 82**.)

Caution If the subclock is selected as the count clock (TCL901 = 1, TCL900 = 1: see Figure 8-2 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 3 (PM3)

PM3 is used to set each bit of port 3 to input or output.

When pin P32/INTP2/TO90 is used for timer output, reset the output latch of P32 and PM32 to 0; when pin P33/INTP3/TO82/BZO90 is used for buzzer output,[Note] reset the output latch of P33 and PM33 to 0.

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

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

Note Never output the TO82 and BZO90 signals at the same time.

Figure 8-4. 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	P3n pin I/O mode (n = 2 or 3)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

8.4 16-Bit Timer Operation

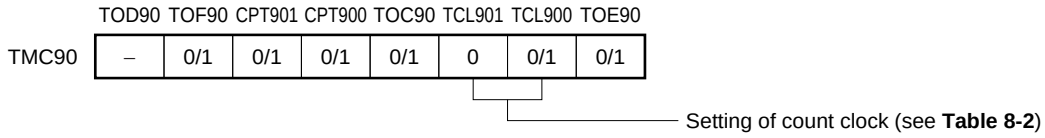
8.4.1 Operation as timer interrupt

In the timer interrupt function, interrupts are repeatedly generated at the count value set in 16-bit compare register 90 (CR90) in advance 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 8-5.

Figure 8-5. Settings of 16-Bit Timer Mode Control Register 90 for Timer Interrupt Operation



Caution If 0 is set both for CPT901 and CPT900 flags, capture operation becomes 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 8-2 shows interval time, and Figure 8-6 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 8-2. Interval Time of 16-Bit Timer

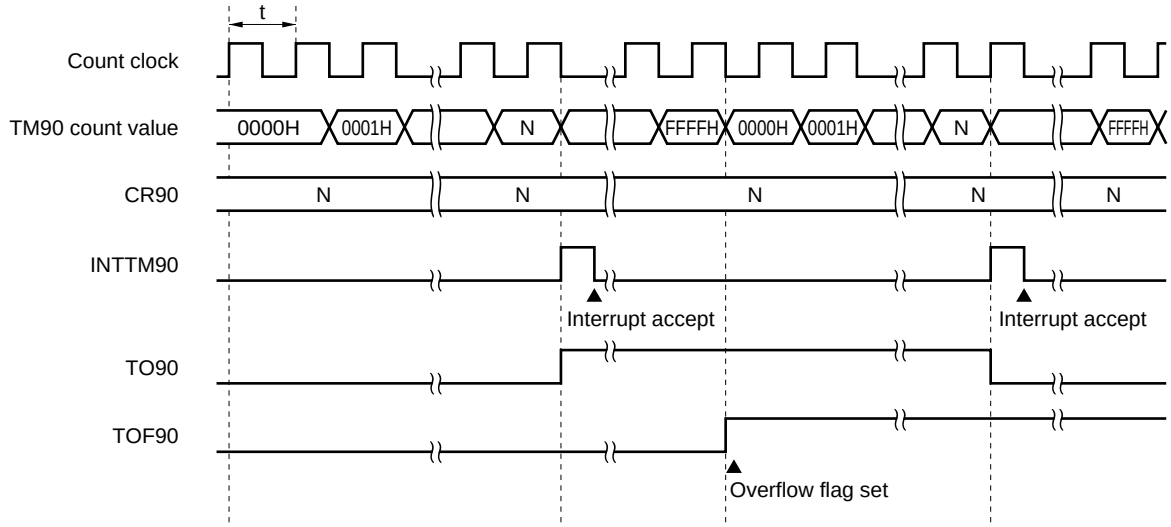
TCL901	TCL900	Count Clock	Interval Time
0	0	$2^2/f_X$ (0.8 μs)	$2^{18}/f_X$ (52.4 ms)
0	1	$2^6/f_X$ (12.8 μs)	$2^{22}/f_X$ (838.9 ms)
1	0	$2^7/f_X$ (25.6 μs)	$2^{23}/f_X$ (1.68 s)
1	1	$1/f_{XT}$ (30.5 μs)	$2^{16}/f_{XT}$ (2.0 s)

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 8-6. Timing of Timer Interrupt Operation



Remark N = 0000H to FFFFH

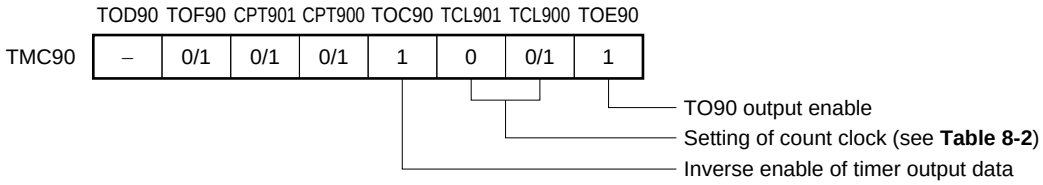
8.4.2 Operation as timer output

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

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

- Set P32 to output mode (PM32 = 0).
- Reset output latch of P32 to 0.
- Set the count value in CR90.
- Set 16-bit timer mode control register 90 (TMC90) as shown in Figure 8-7.

Figure 8-7. Settings of 16-Bit Timer Mode Control Register 90 for Timer Output Operation

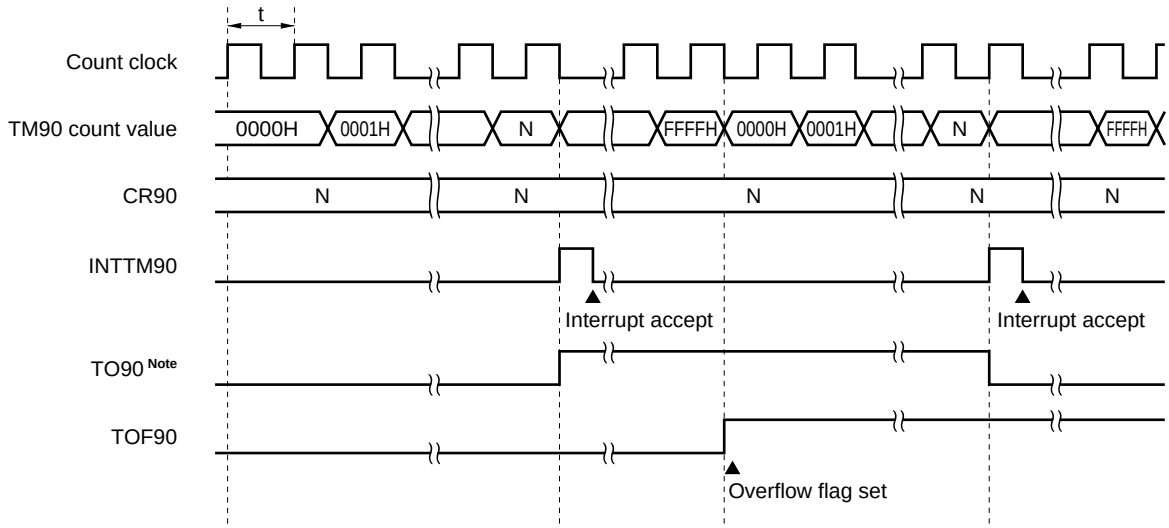


Caution If both the CPT901 flag and CPT900 flag are set to 0, the capture operation becomes prohibited.

When the count value of 16-bit timer counter 90 (TM90) matches the value set in CR90, the output status of the TO90/P32/INTP2 pin is inverted. This enables timer output. At that time, TM90 count is continued and an interrupt request signal (INTTM90) is generated.

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

Figure 8-8. Timer Output Timing



Note The TO90 initial value becomes low level during output enable (TOE90 = 1).

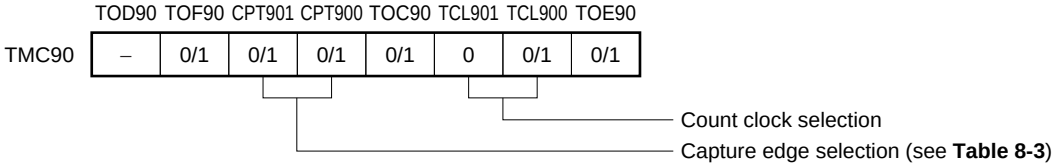
Remark N = 0000H to FFFFH

8.4.3 Capture operation

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

Set TMC90 as shown in Figure 8-9 to allow the 16-bit timer to start the capture operation.

Figure 8-9. Settings of 16-Bit Timer Mode Control Register 90 for Capture Operation



16-bit capture register 90 (TCP90) starts 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 count value within 2 clocks and retains the count value until the next capture edge detection.

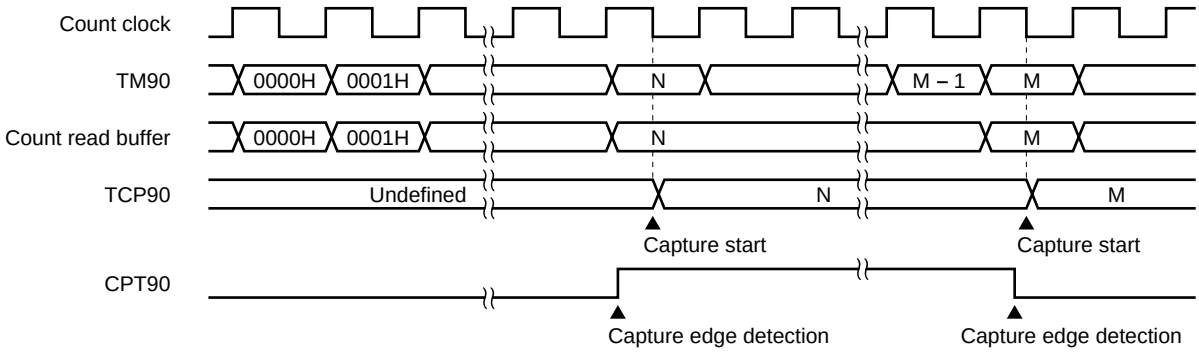
Table 8-3 and Figure 8-10 shows the settings of capture edge and capture operation timing, respectively.

Table 8-3. 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 8-10. Capture Operation Timing (Both Edges of CPT90 Pin are Specified)



8.4.4 16-bit timer counter 90 readout

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

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

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

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

Figure 8-11 shows the timing of 16-bit timer counter 90 readout.

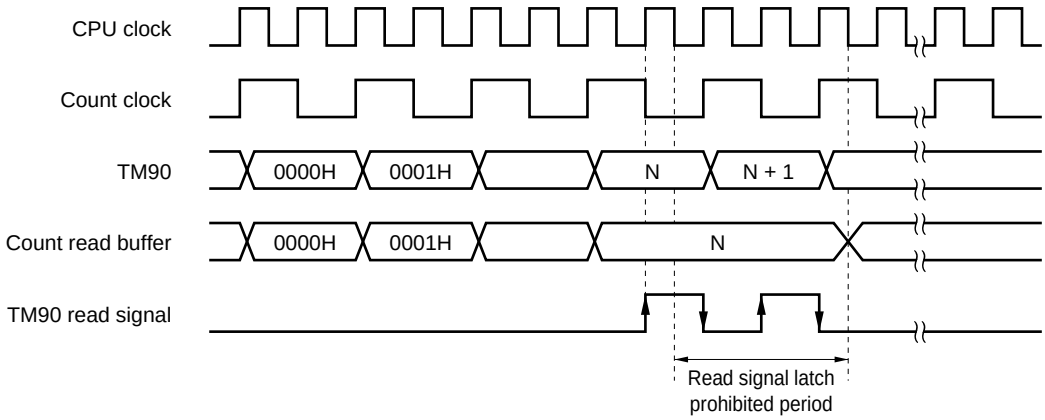
Cautions 1. The count value after releasing stop becomes undefined because the count operation is executed during oscillation stabilization time.

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

When using the 8-bit transfer instruction, execute it in direct addressing.

3. When using the 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 8-11. 16-Bit Timer Counter 90 Readout Timing



8.4.5 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 8-4 shows the buzzer frequency.

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

- Set P33 to output mode (PM33 = 0).
- Reset output latch of P33 to 0.
- Set a count clock by using TCL901 and TCL900.
- Set BZC90 as shown in Figure 8-12.
- Clear TOE82 of 8-bit timer mode control register 82 (TMC82) to 0 to disable the output of 8-bit timer 82.

Figure 8-12. Settings of Buzzer Output Control Register 90 for Buzzer Output Operation

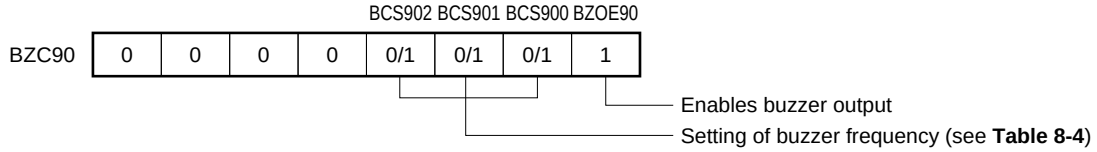


Table 8-4. Buzzer Frequency of 16-Bit Timer

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

★ 8.5 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 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 8-1 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 8-1), 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 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 8-1). 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, and the 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 enable)

At this time, when the setting of P33, the buzzer output alternate function pin is “PM33 = 0, P33 = 0”, a square wave of the buzzer frequency is output from P33. When making the above settings, perform either of the following.

- Set P33 to input mode (PM33 = 1)
- If P33 cannot be set input mode, set the port latch value of P33 to 1 (P33 = 1) (In this case a high level is output from P33)

CHAPTER 9 8-BIT TIMER/EVENT COUNTERS

9.1 Functions of 8-Bit Timer/Event Counters

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

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

(1) 8-bit interval timer

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

Table 9-1. Interval Time of 8-Bit Timer/Event Counter 80

Minimum Interval Time	Maximum Interval Time	Resolution
$1/f_X$ (200 ns)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (200 ns)
$2^3/f_X$ (1.6 μs)	$2^{11}/f_X$ (409.6 μs)	$2^3/f_X$ (1.6 μs)

Remarks 1. f_X: Main system clock oscillation frequency

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

Table 9-2. Interval Time of 8-Bit Timer/Event Counter 81

Minimum Interval Time	Maximum Interval Time	Resolution
$2^4/f_X$ (3.2 μs)	$2^{12}/f_X$ (819.2 μs)	$2^4/f_X$ (3.2 μs)
$2^8/f_X$ (51.2 μs)	$2^{16}/f_X$ (13.1 ms)	$2^8/f_X$ (51.2 μs)

Remarks 1. f_X: Main system clock oscillation frequency

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

Table 9-3. Interval Time of 8-Bit Timer 82

Minimum Interval Time	Maximum Interval Time	Resolution
$2^5/f_X$ (6.4 μs)	$2^{13}/f_X$ (1.64 ms)	$2^5/f_X$ (6.4 μs)
$2^7/f_X$ (25.6 μs)	$2^{15}/f_X$ (6.55 ms)	$2^7/f_X$ (25.6 μs)
$1/f_{XT}$ (30.5 μs)	$2^8/f_{XT}$ (7.81 ms)	$1/f_{XT}$ (30.5 μs)

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) External event counter

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

(3) Square wave output

A square wave of arbitrary frequency can be output.

Table 9-4. Square Wave Output Range of 8-Bit Timer/Event Counter 80

Minimum Pulse Width	Maximum Pulse Width	Resolution
$1/f_X$ (200 ns)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (200 ns)
$2^3/f_X$ (1.6 μs)	$2^{11}/f_X$ (409.6 μs)	$2^3/f_X$ (1.6 μs)

Remark f_X: Main system clock oscillation frequency. The parenthesized values apply to operation at f_X = 5.0 MHz.

Table 9-5. Square Wave Output Range of 8-Bit Timer/Event Counter 81

Minimum Pulse Width	Maximum Pulse Width	Resolution
$2^4/f_X$ (3.2 μs)	$2^{12}/f_X$ (819.2 μs)	$2^4/f_X$ (3.2 μs)
$2^8/f_X$ (51.2 μs)	$2^{16}/f_X$ (13.1 ms)	$2^8/f_X$ (51.2 μs)

Remark f_X: Main system clock oscillation frequency. The parenthesized values apply to operation at f_X = 5.0 MHz.

Table 9-6. Square Wave Output Range of 8-Bit Timer 82

Minimum Pulse Width	Maximum Pulse Width	Resolution
$2^5/f_X$ (6.4 μs)	$2^{13}/f_X$ (1.64 ms)	$2^5/f_X$ (6.4 μs)
$2^7/f_X$ (25.6 μs)	$2^{15}/f_X$ (6.55 ms)	$2^7/f_X$ (25.6 μs)
$1/f_{XT}$ (30.5 μs)	$2^8/f_{XT}$ (7.81 ms)	$1/f_{XT}$ (30.5 μs)

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) PWM output

8-bit resolution PWM output can be produced.

9.2 8-Bit Timer/Event Counter Configuration

The 8-bit timer/event counter consists of the following hardware.

Table 9-7. 8-Bit Timer/Event Counter Configuration

Item	Configuration
Timer counter	8 bits × 3 (TM80 to TM82)
Register	Compare register: 8 bits × 3 (CR80 to CR82)
Timer output	3 (TO80 to TO82)
Control register	8-bit timer mode control register 80 to 82 (TMC80 to TMC82) Port mode register 2, 3 (PM2, PM3)

Figure 9-1. Block Diagram of 8-Bit Timer/Event Counter 80

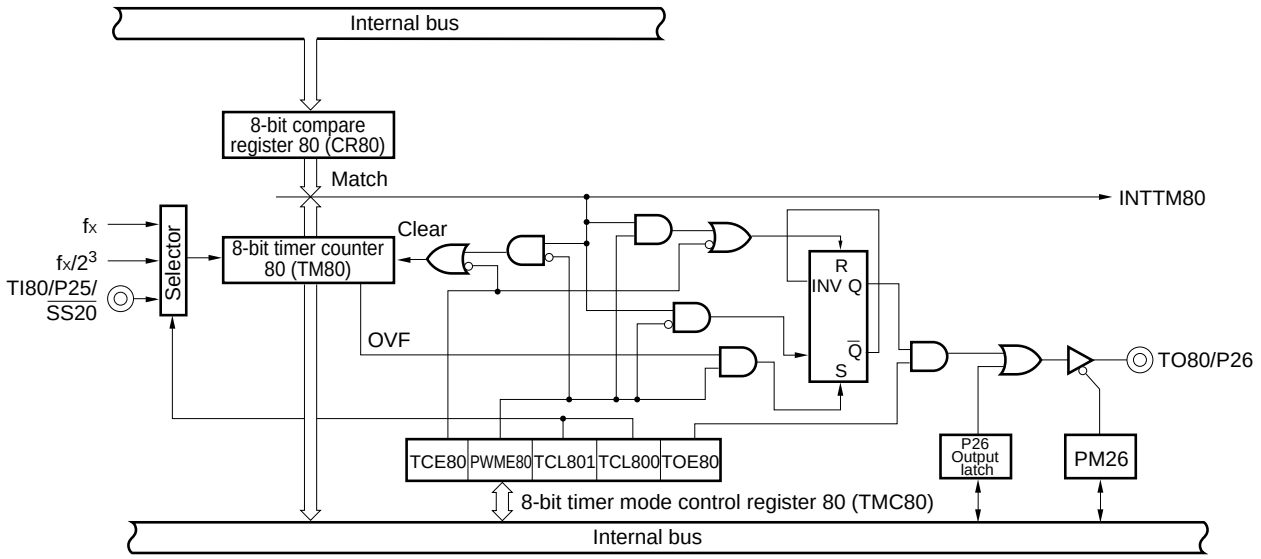


Figure 9-2. Block Diagram of 8-Bit Timer/Event Counter 81

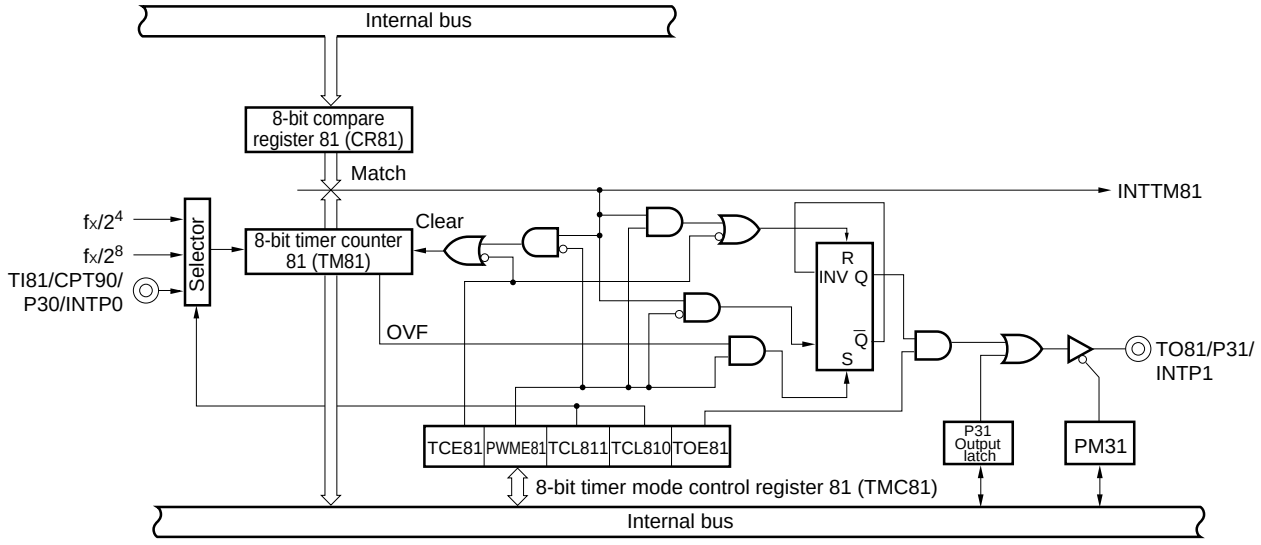
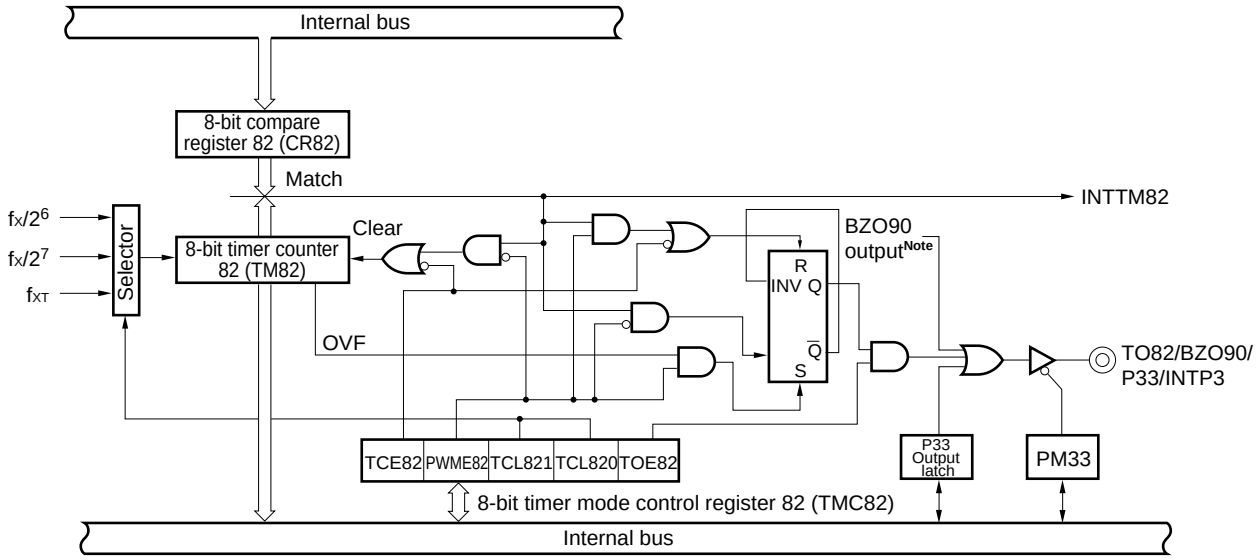


Figure 9-3. Block Diagram of 8-Bit Timer 82



Note See **Figure 8-1 Block Diagram of 16-Bit Timer**.

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

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

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

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

Cautions 1. Before rewriting CR8n, stop the timer operation once. If CR8n is rewritten in the timer operation-enabled state, a match interrupt request signal may occur at the moment of rewrite.

2. Do not clear CR8n to 00H in PWM output mode (when PWME8n = 1: bit 6 of 8-bit timer mode control register 8n (TMC8n)); otherwise, PWM output may not be produced normally.

Remark n = 0 to 2

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

TM8n is used to count the number of pulses.

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

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

Remark n = 0 to 2

9.3 8-Bit Timer/Event Counter Control Registers

The following two types of registers are used to control the 8-bit timer/event counter.

- 8-bit timer mode control registers 80, 81, and 82 (TMC80, TMC81, and TMC82)
- Port mode registers 2 and 3 (PM2 and PM3)

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

TMC80 determines whether to enable or disable 8-bit timer counter 80 (TM80), specifies the count clock for TM80, and controls the operation of the output control circuit of 8-bit timer/event counter 80.

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

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

Figure 9-4. Format of 8-Bit Timer Mode Control Register 80

Symbol	<7>	<6>	5	4	3	2	1	<0>	Address	After reset	R/W
TMC80	TCE80	PWME80	0	0	0	TCL801	TCL800	TOE80	FF53H	00H	R/W

TCE80	TM80 operation control
0	Operation disabled (TM80 is cleared to 00H.)
1	Operation enabled

PWME80	PWM output selection
0	Timer counter operation mode
1	PWM output operation mode

TCL801	TCL800	8-bit timer counter 80 count clock selection
0	0	f_X (5.0 MHz)
0	1	$f_X/2^3$ (625 kHz)
1	0	Rising edge of TI80
1	1	Falling edge of TI80

TOE80	8-bit timer/event counter 80 output control
0	Output disabled (port mode)
1	Output enabled

Cautions 1. Always stop the timer before setting TMC80.

2. For PWM mode operation, the interrupt mask flag (TMMK80) must be set.

Remarks 1. f_X: Main system clock oscillation frequency

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

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

TMC81 determines whether to enable or disable 8-bit timer counter 81 (TM81), specifies the count clock for TM81, and controls the operation of the output control circuit of 8-bit timer/event counter 81.

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

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

Figure 9-5. Format of 8-Bit Timer Mode Control Register 81

Symbol	<7>	<6>	5	4	3	2	1	<0>	Address	After reset	R/W
TMC81	TCE81	PWME81	0	0	0	TCL811	TCL810	TOE81	FF57H	00H	R/W

TCE81	TM81 operation control
0	Operation disabled (TM81 is cleared to 00H.)
1	Operation enabled

PWME81	PWM output selection
0	Timer counter operation mode
1	PWM output operation mode

TCL811	TCL810	8-bit timer counter 81 count clock selection
0	0	$f_X/2^4$ (312.5 kHz)
0	1	$f_X/2^8$ (19.5 kHz)
1	0	Rising edge of TI81
1	1	Falling edge of TI81

TOE81	8-bit timer/event counter 81 output control
0	Output disabled (port mode)
1	Output enabled

Cautions 1. Always stop the timer before setting TMC81.

2. For PWM mode operation, the interrupt mask flag (TMMK81) must be set.

Remarks 1. f_X: Main system clock oscillation frequency

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

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

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

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

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

Figure 9-6. Format of 8-Bit Timer Mode Control Register 82

Symbol	<7>	<6>	5	4	3	2	1	<0>	Address	After reset	R/W
TMC82	TCE82	PWME82	0	0	0	TCL821	TCL820	TOE82	FF5BH	00H	R/W

TCE82	TM82 operation control
0	Operation disabled (TM82 is cleared to 00H.)
1	Operation enabled

PWME82	PWM output selection
0	Timer counter operation mode
1	PWM output operation mode

TCL821	TCL820	8-bit timer counter 82 count clock selection
0	0	$f_X/2^5$ (156.3 kHz)
0	1	$f_X/2^7$ (39.1 kHz)
1	0	f_{XT} (32.768 kHz)
1	1	Setting prohibited

TOE82	8-bit timer 82 output control
0	Output disabled (port mode)
1	Output enabled Note

Note When TOE82 is set to 1, BZOE90 must be set to 0 (see **Figure 8-3 Format of Buzzer Output Control Register 90**).

Cautions 1. Always stop the timer before setting TMC82.

2. For PWM mode operation, the interrupt mask flag (TMMK82) must be set.

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) Port mode registers 2 and 3 (PM2 and PM3)

PM2 and PM3 specify whether each bit of port 2 and port 3 is used for input or output.

To use the P26/TO80 pin for timer output, the PM26 and P26 output latch must be reset to 0.

To use the P31/TO81/INTP1 pin for timer output, the PM31 and P31 output latch must be reset to 0.

To use the P33/INTP3/TO82/BZO90 pin for timer output, the PM33 and P33 output latch must be reset to 0.

PM2 and PM3 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM2 and PM3 to FFH.

Figure 9-7. 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

PM26	P26 pin input/output mode selection
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

Figure 9-8. 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

PM31	P31 pin input/output mode selection
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

PM33	P33 pin input/output mode selection
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

9.4 Operation of 8-Bit Timer/Event Counter

9.4.1 Operation as interval timer

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

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

<1> Set 8-bit timer counter 8n (TM8n) to operation disable (by setting TCE8n (bit 7 of 8-bit timer mode control register 8n (TMC8n)) to 0).
<2> Set the count clock of the 8-bit timer/event counter (see Tables 9-8 to 9-10).
<3> Set a count value in CR8n.
<4> Set TM8n to operation enabled (TCE8n = 1).

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

Tables 9-8 to 9-10 show interval time, and Figure 9-10 shows the timing of interval timer operation.

Cautions 1. Before rewriting CR8n, stop the timer operation once. If CR8n is rewritten in the timer operation-enabled state, a match interrupt request signal may occur at the moment of rewrite.

2. If the count clock setting and TM8n operation-enabled are set in TCM8n simultaneously using an 8-bit memory manipulation instruction, an error of more than a clock in one cycle may occur after the timer start. Therefore, always follow the above procedure when operating the 8-bit timer/event counter as an interval timer.

Remark n = 0 to 2

Table 9-8. Interval Time of 8-Bit Timer/Event Counter 80

TCL801	TCL800	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	$1/f_X$ (200 ns)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (200 ns)
0	1	$2^3/f_X$ (1.6 μs)	$2^{11}/f_X$ (409.6 μs)	$2^3/f_X$ (1.6 μs)
1	0	TI80 input cycle	$2^8 \times$ TI80 input cycle	TI80 input edge cycle
1	1	TI80 input cycle	$2^8 \times$ TI80 input cycle	TI80 input edge cycle

Remarks 1. f_X: Main system clock oscillation frequency

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

Table 9-9. Interval Time of 8-Bit Timer/Event Counter 81

TCL811	TCL810	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	$2^4/f_X$ (3.2 μs)	$2^{12}/f_X$ (819.2 μs)	$2^4/f_X$ (3.2 μs)
0	1	$2^8/f_X$ (51.2 μs)	$2^{16}/f_X$ (13.1 ms)	$2^8/f_X$ (51.2 μs)
1	0	TI81 input cycle	$2^8 \times$ TI81 input cycle	TI81 input edge cycle
1	1	TI81 input cycle	$2^8 \times$ TI81 input cycle	TI81 input edge cycle

Remarks 1. f_X: Main system clock oscillation frequency

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

Table 9-10. Interval Time of 8-Bit Timer 82

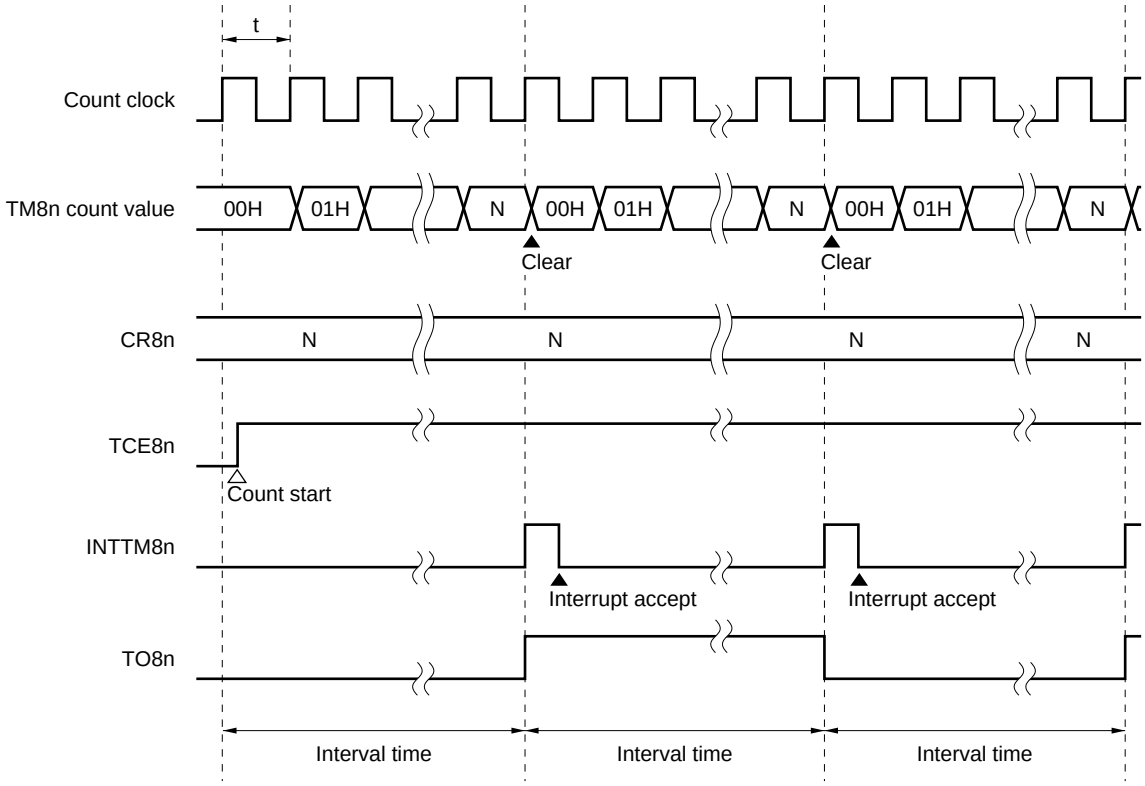
TCL821	TCL820	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	$2^5/f_X$ (6.4 μs)	$2^{13}/f_X$ (1.64 ms)	$2^5/f_X$ (6.4 μs)
0	1	$2^7/f_X$ (25.6 μs)	$2^{15}/f_X$ (6.55 ms)	$2^7/f_X$ (25.6 μs)
1	0	$1/f_{XT}$ (30.5 μs)	$2^8/f_{XT}$ (7.81 ms)	$1/f_{XT}$ (30.5 μs)
1	1	Setting prohibited		

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 9-9. Interval Timer Operation Timing



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

2. n = 0 to 2

9.4.2 Operation as external event counter[Note]

The external event counter counts the number of external clock pulses input to the TI80/P25/$\overline{SS20}$ or TI81/P30/INTP0/CPT90 pin by using 8-bit timer counters 80 and 81 (TM80 and TM81).

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

<1> Set P25 and P30 to input mode (PM25 = 1, PM30 = 1).

<2> Set 8-bit timer register 8n (TM8n) to operation disable (by setting TCE8n (bit 7 of 8-bit timer mode control register 8n (TMC8n)) to 0).

<3> Specify the rising/falling edges of TI8n (see Tables 9-8 and 9-9).

<4> Set a count value in CR8n.

<5> Set TM8n to operation enabled (TCE8n = 1).

Note Only TM80 and TM81 have this function.

Each time the valid edge specified by bit 1 (TCL8n0) of TMC8n is input, the value TM8n is incremented.

When the count value of TM8n matches the value set in CR8n, TM8n is cleared to 0 and continues counting. At the same time, an interrupt request signal (INTTM8n) is generated.

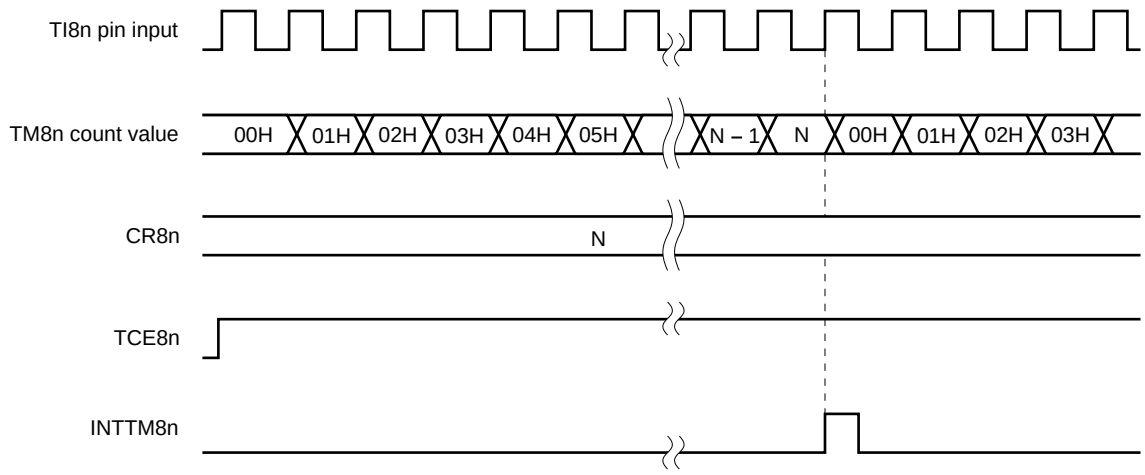
Figure 9-10 shows the timing of the external event counter operation (with rising edge specified).

Cautions 1. Before rewriting CR8n, stop the timer operation once. If CR8n is rewritten in the timer operation-enabled state, a match interrupt request signal may occur at the moment of rewrite.

2. If the count clock setting and TM8n operation-enabled are set in TCM8n simultaneously using an 8-bit memory manipulation instruction, an error of more than a clock in one cycle may occur after the timer start. Therefore, always follow the above procedure when operating the 8-bit timer/event counter as an external event counter.

Remark n = 0, 1

Figure 9-10. External Event Counter Operation Timing (with Rising Edge Specified)



Remarks 1. N = 00H to FFH

2. n = 0, 1

9.4.3 Operation as square wave output

The 8-bit timer/event counter can generate output square waves of an arbitrary frequency at intervals specified by the count value set in 8-bit compare registers 8n (CR8n) in advance.

To operate 8-bit timer/event counters 8n for square wave output, the following settings are required.

<1> Set P26, P31, and P33 to output mode (PM26 = 0, PM31 = 0, PM33 = 0).

<2> Reset the output latches of P26, P31, and P33 to 0.

<3> Set 8-bit timer counter 8n (TM8n) to operation disable (by setting TCE8n (bit 7 of 8-bit timer mode control register 8n (TMC8n)) to 1).

<4> Set the count clock of 8-bit timer/event counter 8n and set TO8n to output enable (TOE8n (bit 0 of TMC8n) = 1).

<5> Set count value in CR8n.

<6> Set TM8n to operation enable (TCE8n = 1).

When the count value of TM8n matches the value set in CR8n, the TO8n pin output will be inverted. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurs, TM8n will be cleared to 00H and resumes to count, generating an interrupt request signal (INTTM8n).

Setting 0 for bit 7 (TCE8n) of TMC8n clears the square-wave output to 0.

Tables 9-11 through 9-13 show square wave output range, and Figure 9-11 shows timing of square wave output.

Cautions 1. Before rewriting CR8n, stop the timer operation once. If CR8n is rewritten in the timer operation-enabled state, a match interrupt request signal may occur at the moment of rewrite.

2. If the count clock setting and TM8n operation-enabled are set in TCM8n simultaneously using an 8-bit memory manipulation instruction, an error of more than a clock in one cycle may occur after the timer start. Therefore, always follow the above procedure when operating the 8-bit timer/event counter for square wave output.

Remark n = 0 to 2

Table 9-11. Square Wave Output Range of 8-Bit Timer/Event Counter 80

TCL801	TCL800	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	$1/f_X$ (200 ns)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (200 ns)
0	1	$2^3/f_X$ (1.6 μs)	$2^{11}/f_X$ (409.6 μs)	$2^3/f_X$ (1.6 μs)

Remarks 1. f_X: Main system clock oscillation frequency

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

Table 9-12. Square Wave Output Range of 8-Bit Timer/Event Counter 81

TCL811	TCL810	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	$2^4/f_X$ (3.2 μs)	$2^{12}/f_X$ (819.2 μs)	$2^4/f_X$ (3.2 μs)
0	1	$2^8/f_X$ (51.2 μs)	$2^{16}/f_X$ (13.1 ms)	$2^8/f_X$ (51.2 μs)

Remarks 1. f_X: Main system clock oscillation frequency

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

Table 9-13. Square Wave Output Range of 8-Bit Timer 82

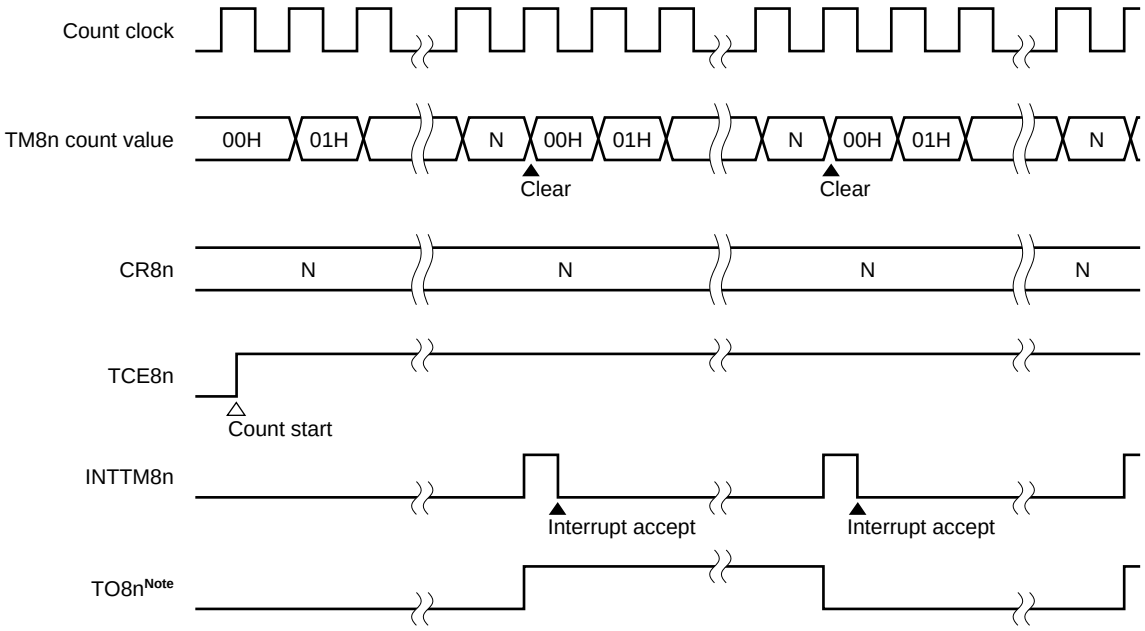
TCL821	TCL820	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	$2^5/f_X$ (6.4 μs)	$2^{13}/f_X$ (1.64 ms)	$2^5/f_X$ (6.4 μs)
0	1	$2^7/f_X$ (25.6 μs)	$2^{15}/f_X$ (6.55 ms)	$2^7/f_X$ (25.6 μs)
1	0	$1/f_{XT}$ (30.5 μs)	$2^8/f_{XT}$ (7.81 ms)	$1/f_{XT}$ (30.5 μs)

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 9-11. Square Wave Output Timing



Note The initial value of TO8n is low for output enable (TOE8n = 1).

Remark n = 0 to 2

9.4.4 PWM output operation

PWM output enables an interruption repeatedly at intervals specified by the count value set in 8-bit compare register 8n (CR8n) in advance.

To use 8-bit timer/event counter 8n for PWM output, the following settings are required.

<1> Set P26, P31, and P33 to output mode (PM26 = 0, PM31 = 0, PM33 = 0).

<2> Reset the output latches of P26, P31, and P33 to 0.

<3> Set 8-bit timer counter 8n (TM8n) to operation disable (by setting TCE8n (bit 7 of 8-bit timer mode control register 8n (TMC8n)) to 0).

<4> Set the count clock of 8-bit timer/event counter 8n, and set TO8n to output enable (TOE8n (bit 0 of TMC8n) = 1), and to PWM output mode (PWME8n = 1).

<5> Set a count value in CR8n.

<6> Set TM8n to operation enable (TCE8n = 1).

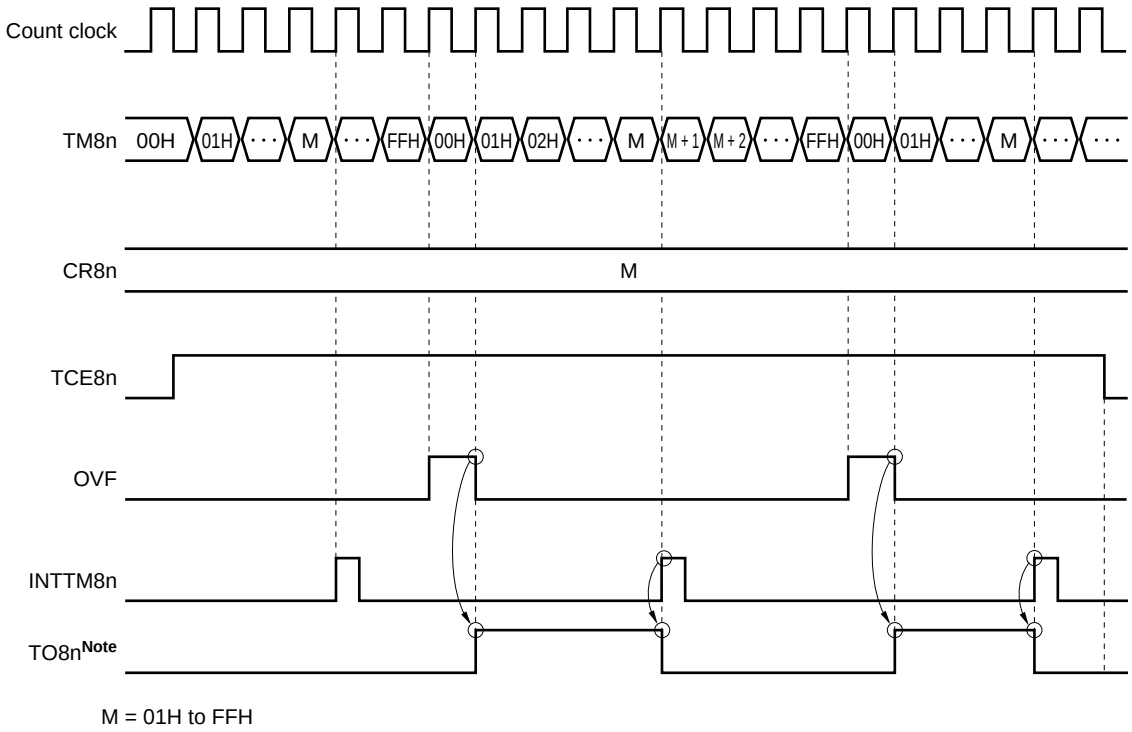
When the count value of TM8n matches the value set in CR8n, TM8n continues counting, and an interrupt request signal (INTTM8n) is generated.

★ **Cautions 1. Before rewriting CR8n, stop the timer. If CR8n is rewritten in the timer operation-enabled state, a high-level signal may be output for the next cycle (256 count pulses) (for details, see 9.5 (3) Timer operation after compare register is rewritten during PWM output).**

2. If the count clock setting and TM8n operation-enabled are set in TCM8n simultaneously using an 8-bit memory manipulation instruction, an error of more than a clock in one cycle may occur after the timer start. Therefore, always follow the above procedure when operating the 8-bit timer/event counter for PWM output.

Remark n = 0 to 2

Figure 9-12. PWM Output Timing



Note The initial value of TO8n is low for output enable (TOE8n = 1).

Caution Do not set CR8n to 00H in PWM output mode; otherwise, PWM may not be output normally.

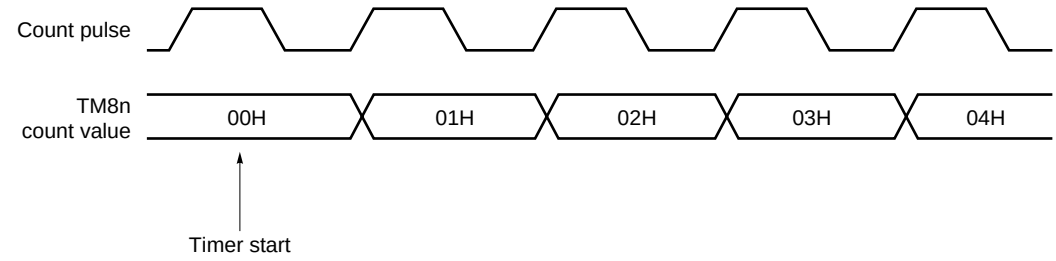
Remark n = 0 to 2

9.5 Notes on Using 8-Bit Timer/Event Counters

(1) Error on starting timer

An error of up to 1 clock is included in the time between the timer being started and a coincidence signal being generated. This is because 8-bit timer counter 8n (TM8n) is started in asynchronization with the count pulse.

Figure 9-13. Start Timing of 8-Bit Timer Counter



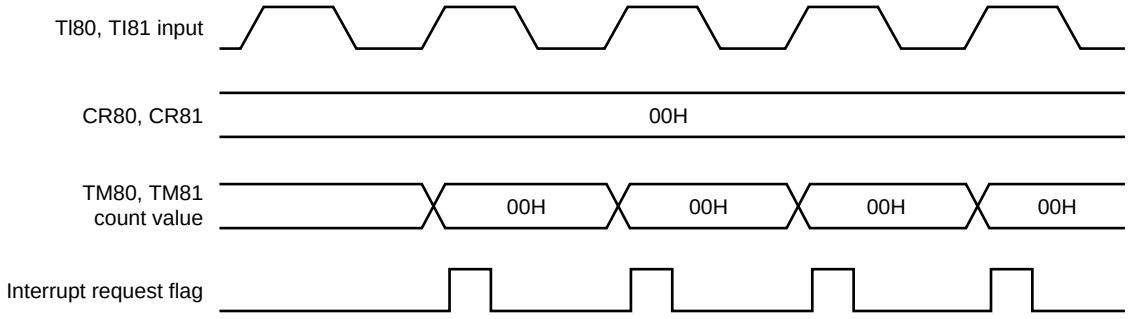
Remark n = 0 to 2

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

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

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

Figure 9-14. External Event Counter Operation Timing



Cautions 1. When CR8n is rewritten to timer counter operation mode (PWME8n (8-bit timer mode control register 8n (TMC8n)) = 0), be sure to stop the timer operation beforehand. If CR8n is rewritten in the timer operation-enabled state, a match interrupt request signal may occur at the moment of rewrite.

2. If CR8n is rewritten during timer operation in the PWM operation mode (PWME8n = 1), pulses may not be generated for one cycle after the rewrite.

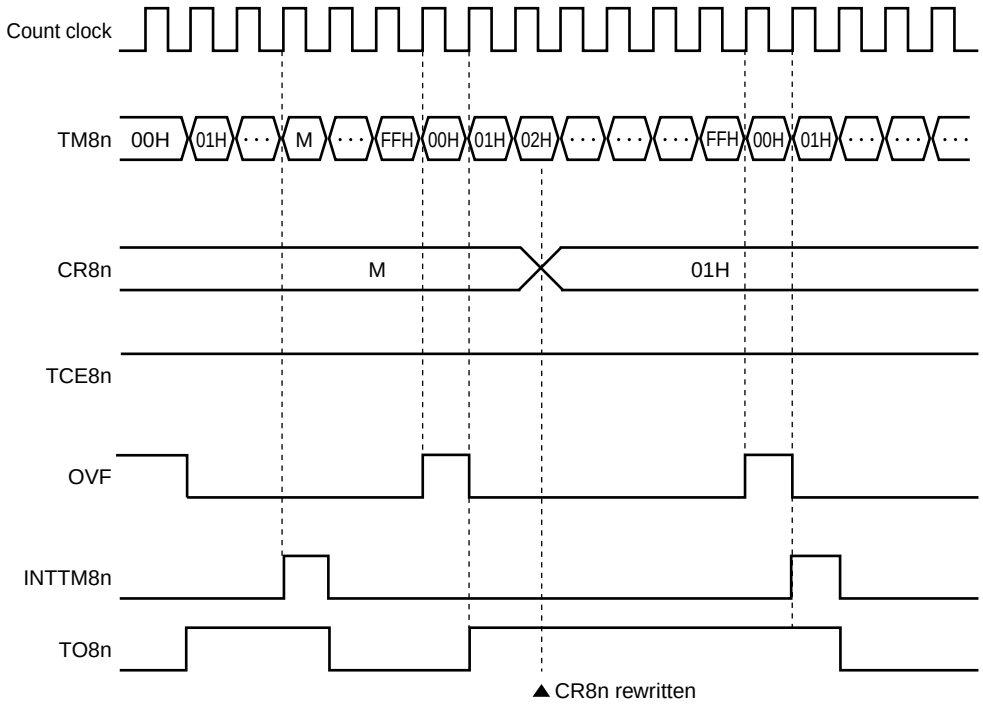
3. Do not set CR8n to 00H in PWM operation mode; otherwise, PWM may not be output normally.

Remark n = 0 to 2

★ **(3) Timer operation after compare register is rewritten during PWM output**

When 8-bit compare register 8n (CR8n) is rewritten during PWM output, if the new value is smaller than that of 8-bit timer/counter 8n (TM8n), a high-level signal may be output for the next cycle (256 count pulses) after the CR8n value is rewritten. Figure 9-15 shows the timing at which the high-level signal is output.

Figure 9-15. Operation Timing after Compare Register is Rewritten during PWM Output



M = 01H to FFH

Remark n = 0 to 2

CHAPTER 10 WATCH TIMER

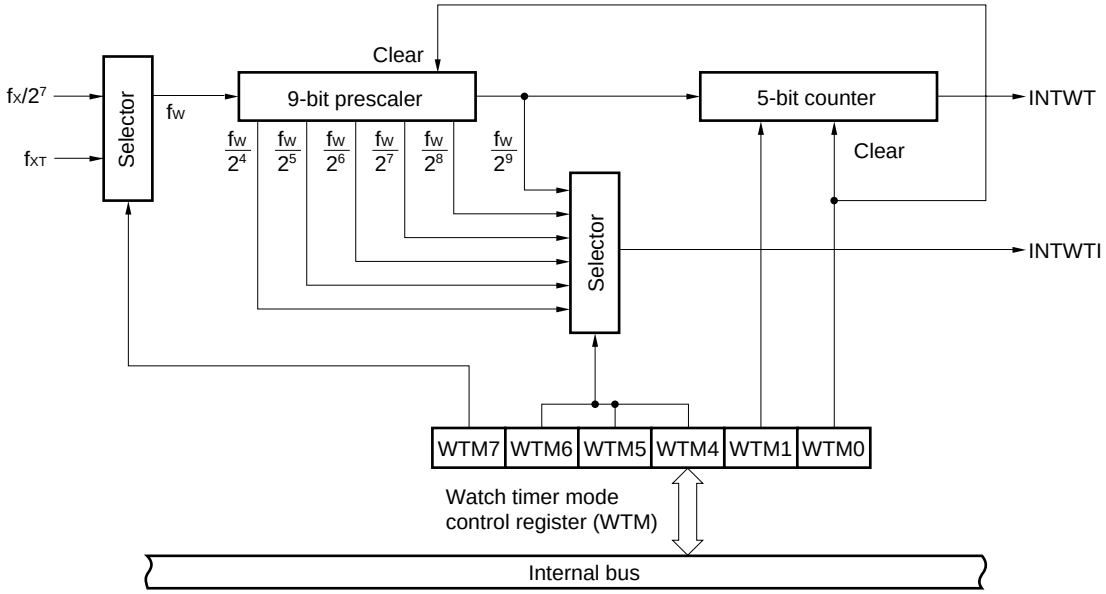
10.1 Watch Timer Functions

The watch timer has the following functions:

- Watch timer
- Interval timer

The watch and interval timers can be used at the same time.
Figure 10-1 is a block diagram of the watch timer.

Figure 10-1. 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 10-1. 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

10.2 Watch Timer Configuration

The watch timer consists of the following hardware.

Table 10-2. Watch Timer Configuration

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

10.3 Watch Timer Control Register

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

- Watch timer mode control register (WTM)
  WTM selects a count clock for the watch timer and specifies whether to enable 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 10-2. 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.

10.4 Watch Timer Operation

10.4.1 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.

Only the watch timer can be started form 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.

10.4.2 Operation as interval timer

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

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

Table 10-3. Interval Generated Using the Interval Timer

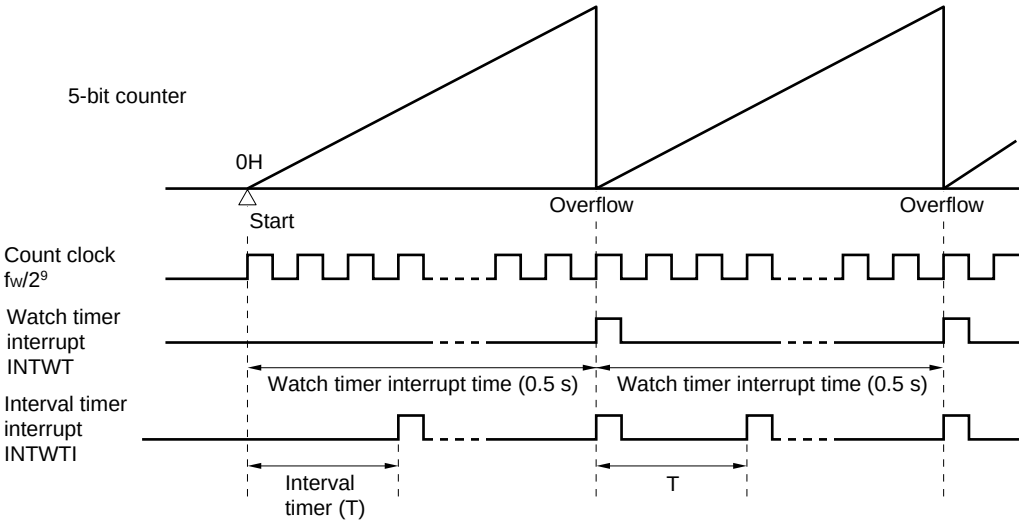
WTM6	WTM5	WTM4	Interval	At f_X = 5.0 MHz	At f_X = 4.19 MHz	At f_{XT} = 32.768 kHz
0	0	0	$2^4 \times 1/f_W$	409.6 μs	489 μs	488 μs
0	0	1	$2^5 \times 1/f_W$	819.2 μs	978 μs	977 μs
0	1	0	$2^6 \times 1/f_W$	1.64 ms	1.96 ms	1.95 ms
0	1	1	$2^7 \times 1/f_W$	3.28 ms	3.91 ms	3.91 ms
1	0	0	$2^8 \times 1/f_W$	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 10-3. 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 11 WATCHDOG TIMER

11.1 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 inadvertent program loops. When an inadvertent loop is detected, a non-maskable interrupt or a $\overline{\text{RESET}}$ signal can be generated.

Table 11-1. Inadvertent Loop Detection Time of Watchdog Timer

Inadvertent Loop 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 interval set in advance.

Table 11-2. 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

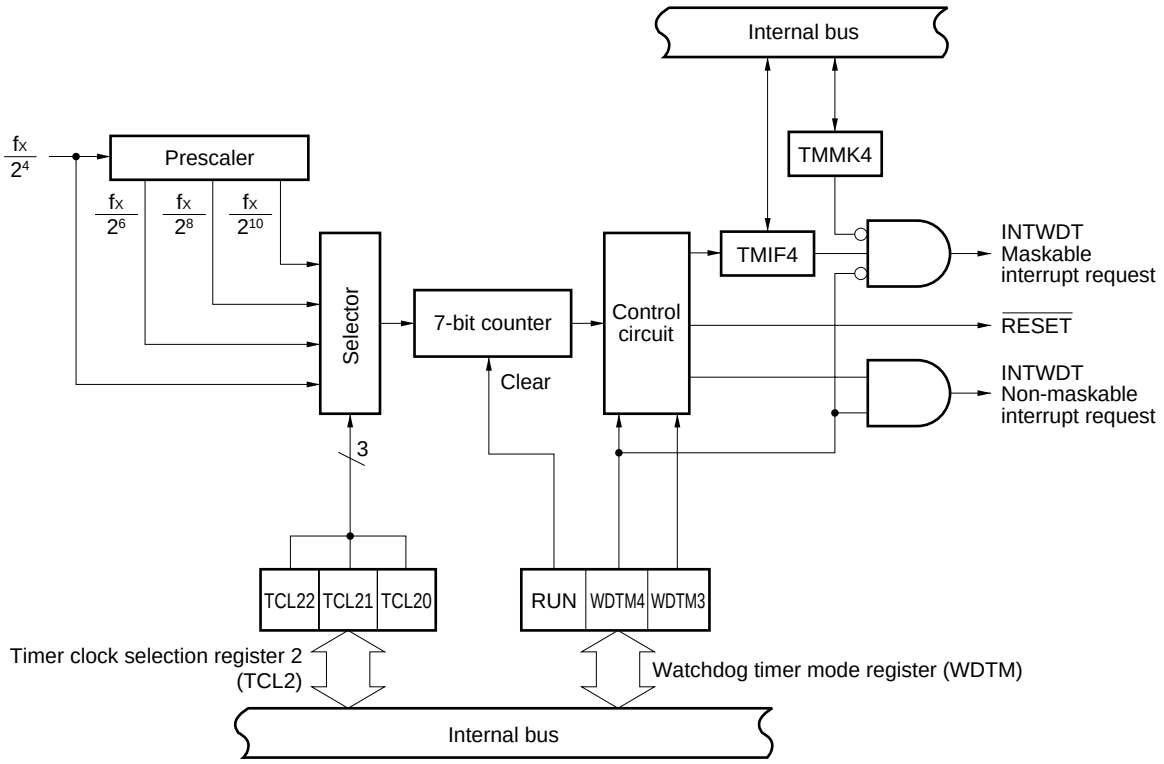
11.2 Watchdog Timer Configuration

The watchdog timer consists of the following hardware.

Table 11-3. Configuration of Watchdog Timer

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

Figure 11-1. Block Diagram of Watchdog Timer



11.3 Watchdog Timer Control Registers

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

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

(1) Timer clock selection register 2 (TCL2)

This register sets the watchdog timer count clock.

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

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

Figure 11-2. Format of Timer Clock Selection Register 2

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
TCL2	0	0	0	0	0	TCL22	TCL21	TCL20	FF42H	00H	R/W

TCL22	TCL21	TCL20	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 an operation mode of the watchdog timer, and enables/disables counting of the watchdog timer.

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

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

Figure 11-3. 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 to 1, it cannot be cleared to 0 by software. Therefore, when counting is started, it cannot be stopped by any means other than $\overline{\text{RESET}}$ input.

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

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

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

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)) being 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.

11.4 Watchdog Timer Operation

11.4.1 Operation as watchdog timer

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

The count clock (inadvertent loop detection time interval) of the watchdog timer can be selected by bits 0 to 2 (TCL20 to TCL22) of timer clock selection register 2 (TCL2). By setting bit 7 (RUN) of WDTM to 1, the watchdog timer is started. Set RUN to 1 within the set inadvertent loop detection time interval after the watchdog timer has been started. By setting RUN to 1, the watchdog timer can be cleared and start counting. If RUN is not set to 1, and the inadvertent loop detection time is exceeded, 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 inadvertent loop detection time may be up to 0.8% shorter than the set time.

2. When the subsystem clock is selected as the CPU clock, watchdog timer count operation is stopped. Even when the main system clock continues oscillating in this case, watchdog timer count operation is stopped.

Table 11-4. Inadvertent Loop Detection Time of Watchdog Timer

TCL22	TCL21	TCL20	Inadvertent Loop 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

11.4.2 Operation as interval timer

When bits 4 and 3 (WDTM4, WDTM3) of 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 count value set in advance.

Select a count clock (or interval) by setting bits 0 to 2 (TCL20 to TCL22) of timer clock selection register 2 (TCL2). The watchdog timer starts operation as an interval timer when the RUN bit (bit 7 of WDTM) is set to 1.

In interval timer mode, the interrupt mask flag (TMMK4) 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 a $\overline{\text{RESET}}$ 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 11-5. Interval Time of Interval Timer

TCL22	TCL21	TCL20	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 12 8-BIT A/D CONVERTER (μPD789167 AND 789167Y SUBSERIES)

12.1 8-Bit A/D Converter Functions

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

A/D conversion can be started only by software.

One of analog inputs ANI0 to ANI7 is selected for A/D conversion. A/D conversion is performed repeatedly, with an interrupt request (INTAD0) being issued each time A/D conversion is completed.

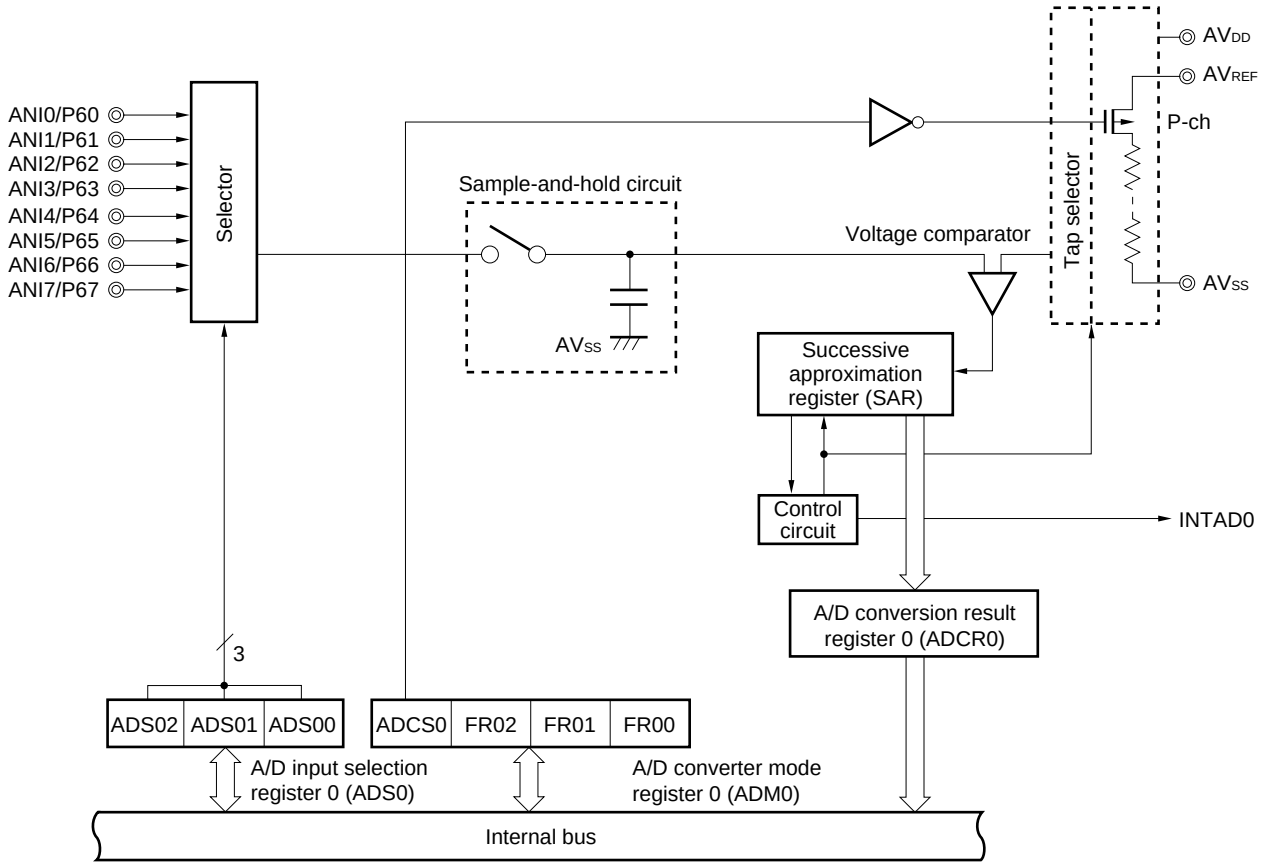
12.2 8-Bit A/D Converter Configuration

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

Table 12-1. Configuration of 8-Bit A/D Converter

Item	Configuration
Analog input	8 channels (ANI0 to ANI7)
Register	Successive approximation register (SAR) A/D conversion result register 0 (ADCR0)
Control register	A/D converter mode register 0 (ADM0) A/D input selection register 0 (ADS0)

Figure 12-1. Block Diagram of 8-Bit A/D Converter



(1) Successive approximation register (SAR)

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

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

(3) Sample-and-hold circuit

The sample-and-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 serial resistor string.

(5) Serial resistor string

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

(6) ANI0 to ANI7

Pins ANI0 to ANI7 are the 8-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 ANI7 with voltages that fall outside the rated range. If a voltage greater than AV_{REF} or less than AV_{SS} (even if within the absolute maximum rating) is supplied to any of these pins, the conversion value for the corresponding channel will be undefined. Furthermore, the conversion values for the other channels may also be affected.

(7) AV_{REF}

This pin inputs the A/D converter reference voltage.

It converts signals input to ANI0 to ANI7 into digital signals according to the voltage applied between AV_{REF} and AV_{SS}.

(8) 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_{SS0} pin, even while the A/D converter is not being used.

(9) 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_{DD0} pin, even while the A/D converter is not being used.

12.3 8-Bit A/D Converter Control Registers

The following two registers are used to control the 8-bit A/D converter.

- A/D converter mode register 0 (ADM0)
- A/D input selection 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 clears ADM0 to 00H.

Figure 12-2. 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. The result of conversion performed immediately after bit 7 (ADCS0) is set is undefined.

2. The conversion result may be undefined after ADCS0 has been cleared to 0 (For details, see 12.5 (5) Timing of undefined A/D conversion result).

Remarks 1. fx: Main system clock oscillation frequency

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

(2) A/D input selection 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 12-3. Format of A/D Input Selection 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
1	1	0	ANI6
1	1	1	ANI7

Caution Bits 3 to 7 must all be set to 0.

12.4 8-Bit A/D Converter Operation

12.4.1 Basic operation of 8-bit A/D converter

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

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

<3> After sampling continues for a certain period of time, the sample and 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_{REF}.

<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_{REF}, the MSB of SAR is left set. If it is lower than half of AV_{REF}, 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_{REF}
- Bit 7 = 0: One quarter of AV_{REF}

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

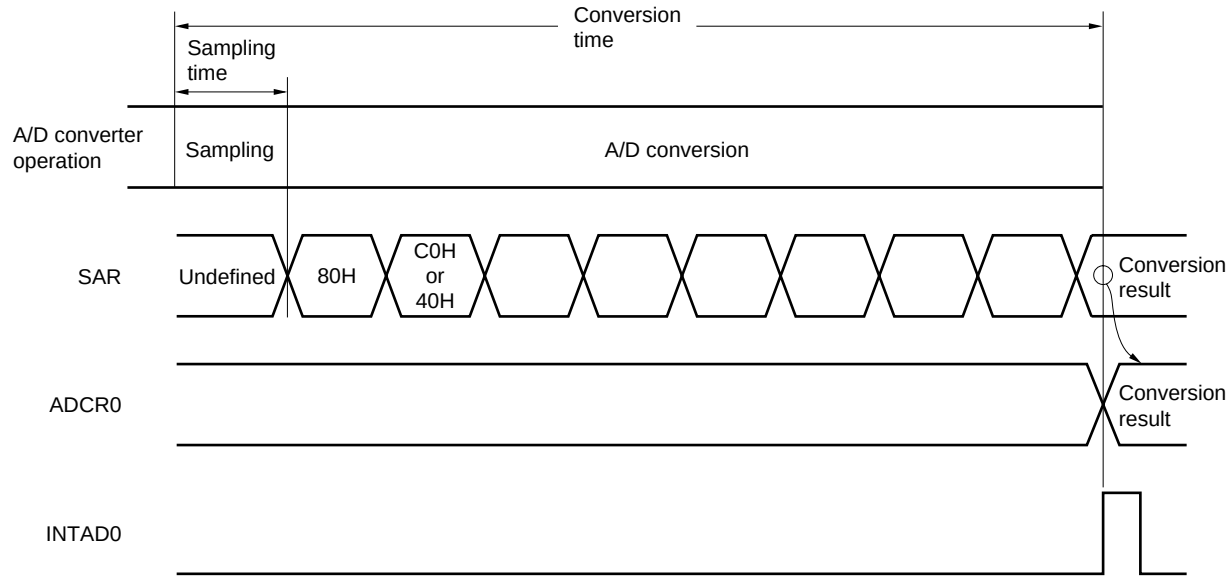
<7> Comparison is repeated until bit 0 of SAR is reached.

<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 12-4. 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 to 0 by software.

If an attempt is made to write to ADM0 or A/D input selection 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 to 1.

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

12.4.2 Input voltage and conversion result

The relationships between the analog input voltage at the analog input pins (ANI0 to ANI7) and the A/D conversion result (A/D conversion result register 0 (ADCR0)) are represented by:

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

or

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

INT( ): Function that returns the integer part of a parenthesized value

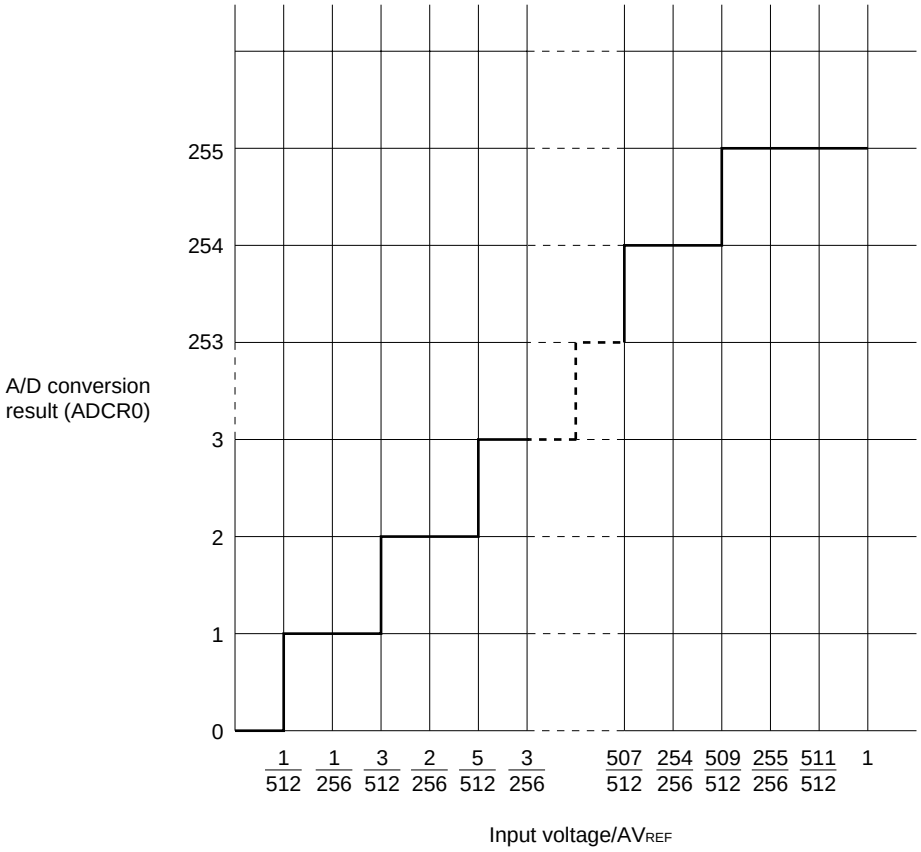
V_{IN}: Analog input voltage

AV_{REF}: Voltage of AV_{REF} pin

ADCR0: Value in A/D conversion result register 0 (ADCR0)

Figure 12-5 shows the relationship between the analog input voltage and the A/D conversion result.

Figure 12-5. Relationship between Analog Input Voltage and A/D Conversion Result



12.4.3 Operation mode of 8-bit A/D converter

The A/D converter is initially in select mode. In this mode, A/D input selection register 0 (ADS0) is used to select an analog input channel from ANI0 to ANI7 for A/D conversion.

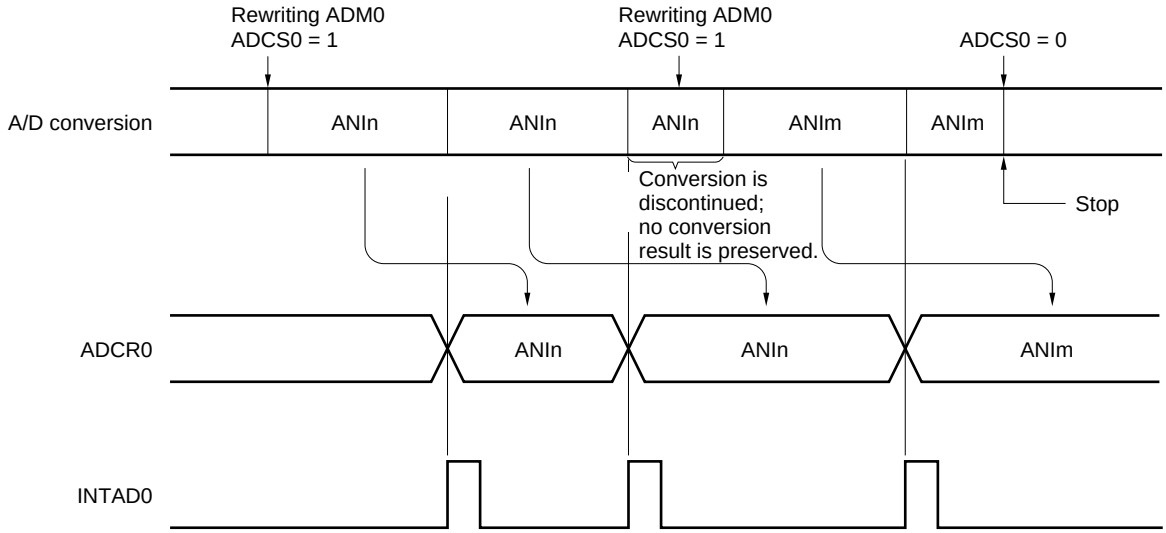
A/D conversion can be started only by software, that is, by setting A/D converter mode register 0 (ADM0).

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 12-6. Software-Started A/D Conversion



Remarks 1. n = 0 to 7

2. m = 0 to 7

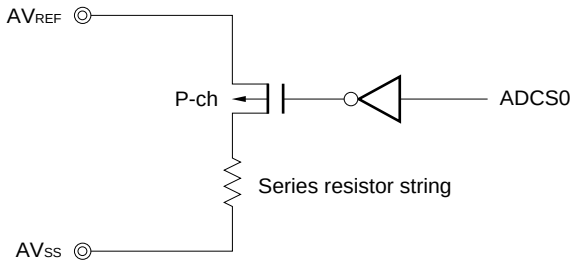
12.5 Cautions Related to 8-Bit A/D Converter

(1) Current drain in standby mode

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

Figure 12-7 shows how to reduce the current drain in standby mode.

Figure 12-7. How to Reduce Current Drain in Standby Mode



(2) Input range for pins ANI0 to ANI7

Be sure to keep the input voltage at ANI0 to ANI7 within its rating. If a voltage not lower than AV_{REF} 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. It 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 A/D input selection 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 12-8 and 12-9 show the timing at which the conversion result is read.

Figure 12-8. Conversion Result Read Timing (If Conversion Result is Undefined)

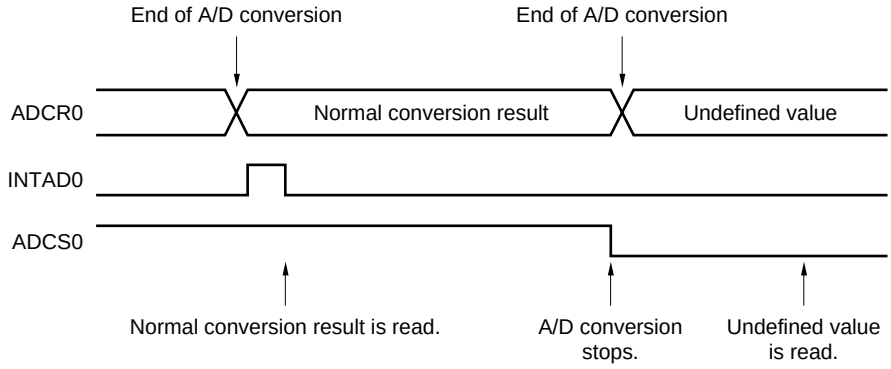
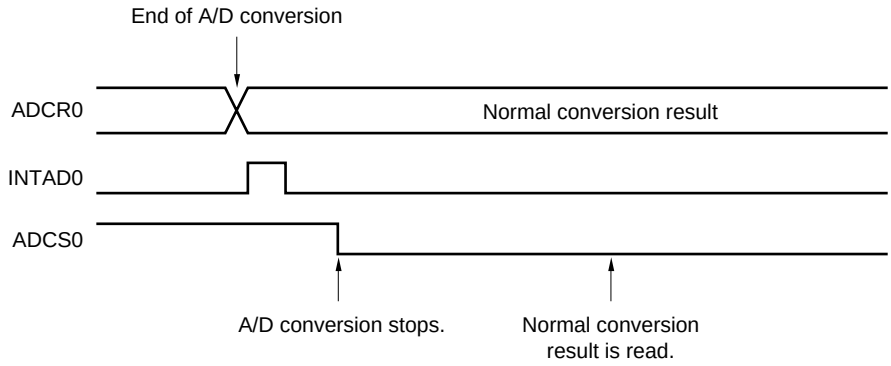


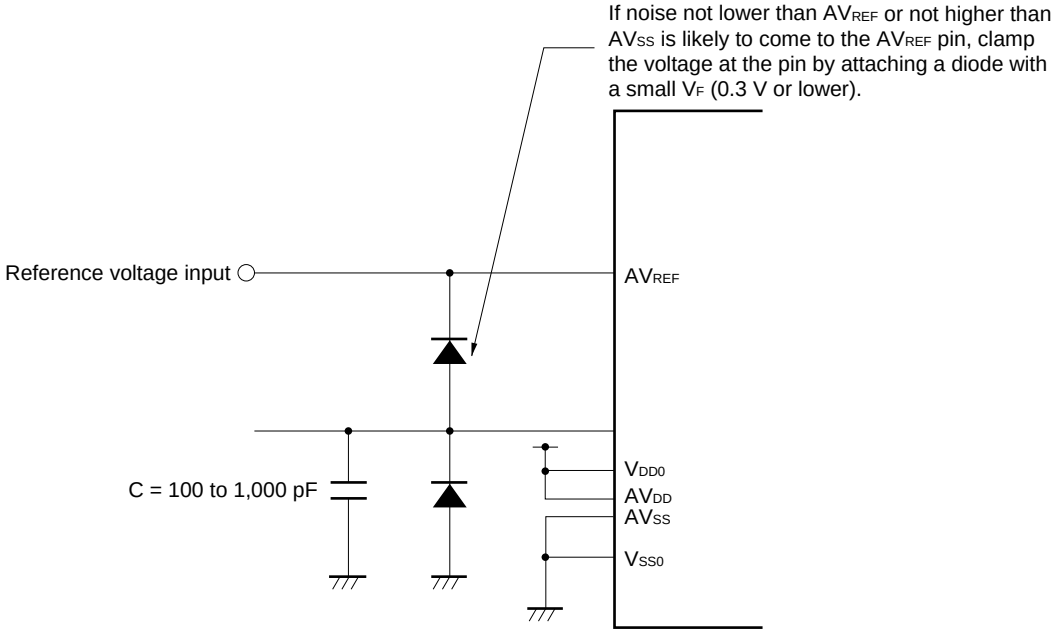
Figure 12-9. Conversion Result Read Timing (If Conversion Result is Normal)



(6) Noise prevention

To maintain a resolution of 8 bits, watch for noise to pins AV_{REF} and ANI0 to ANI7. The higher the output impedance of the analog input source is, the larger the effect by noise is. To reduce noise, attach an external capacitor to the relevant pins as shown in Figure 12-10.

Figure 12-10. Analog Input Pin Treatment



(7) ANI0 to ANI7

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

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

If a digital pulse is applied to a pin adjacent to the analog input pins during A/D conversion, coupling noise may occur which 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) Input impedance of the AV_{REF} pin

A series resistor string of several 10 kΩ is connected across the AV_{REF} and AV_{SS} pins.

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

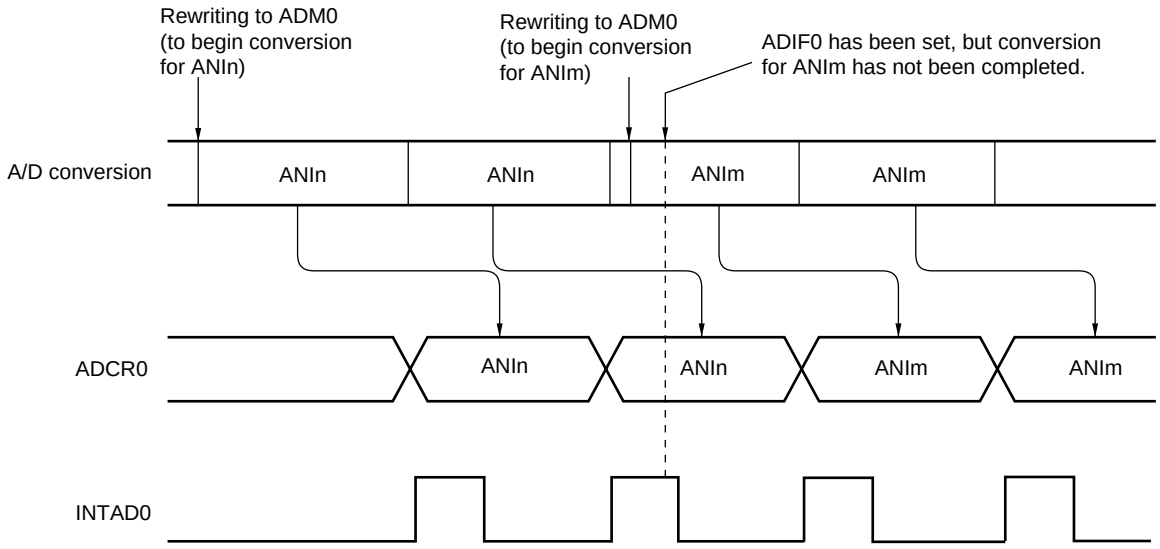
(9) Interrupt request flag (ADIF0)

Changing the content 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 12-11. A/D Conversion End Interrupt Request Generation Timing



Remarks 1. n = 0 to 7

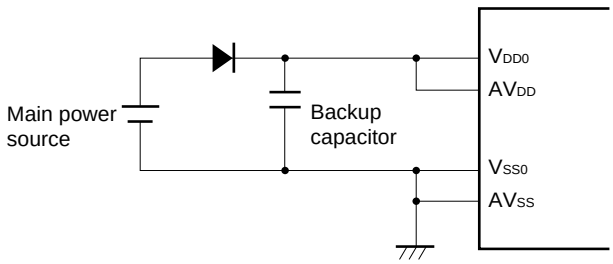
2. m = 0 to 7

(10) AV_{DD} pin

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

If your application is designed to be changed to backup power, the AV_{DD} pin must be supplied with the same voltage level as for the V_{DD0} pin, as shown in Figure 12-12.

Figure 12-12. AV_{DD} Pin Treatment



[MEMO]

CHAPTER 13 10-BIT A/D CONVERTER (μPD789177 AND 789177Y SUBSERIES)

13.1 10-Bit A/D Converter Functions

The 10-bit A/D converter is a 10-bit resolution converter to convert an analog input to digital signals. This converter can control eight channels (ANI0 to ANI7) of analog inputs.

A/D conversion can be started only by software.

One of analog inputs ANI0 to ANI7 is selected for A/D conversion. A/D conversion is performed repeatedly, with an interrupt request (INTAD0) being issued each time A/D conversion is completed.

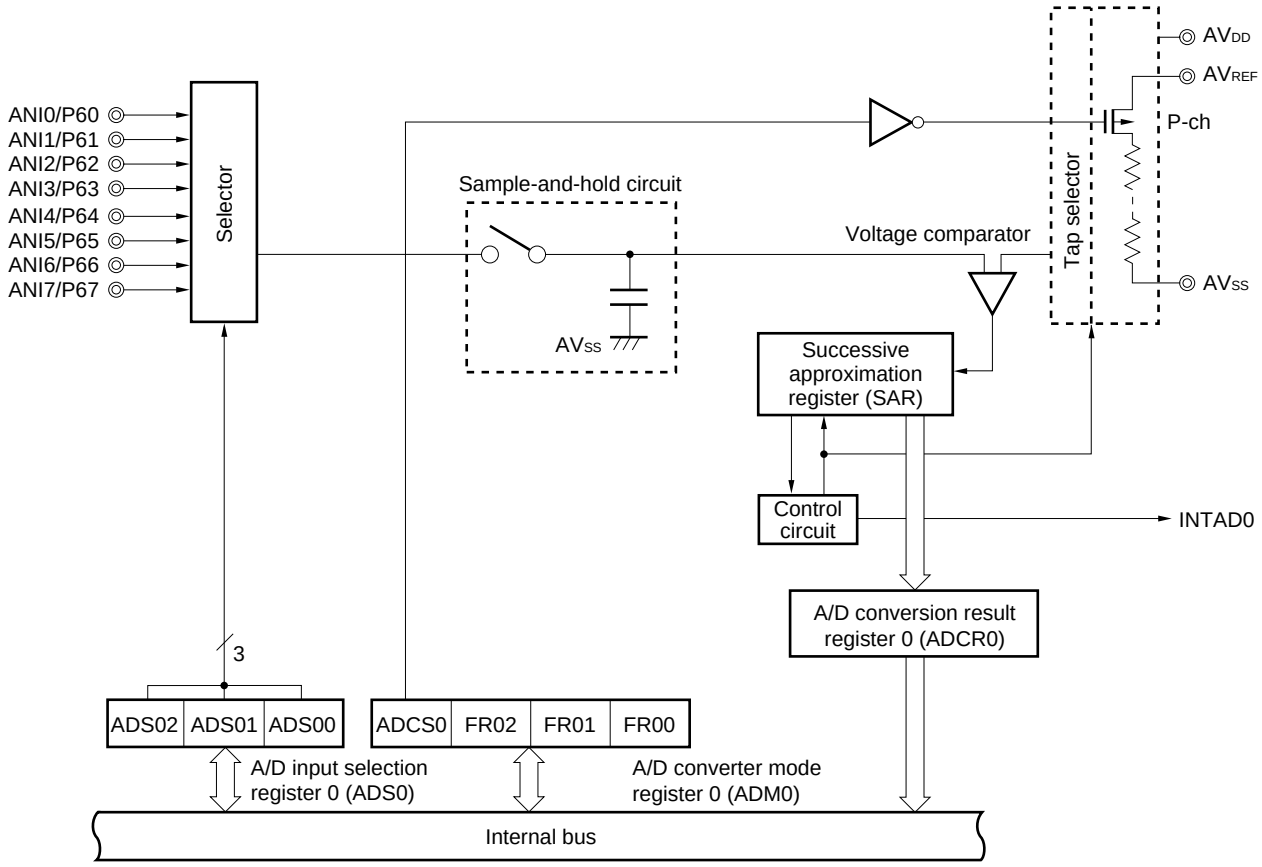
13.2 10-Bit A/D Converter Configuration

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

Table 13-1. Configuration of 10-Bit A/D Converter

Item	Configuration
Analog input	8 channels (ANI0 to ANI7)
Register	Successive approximation register (SAR) A/D conversion result register 0 (ADCR0)
Control register	A/D converter mode register 0 (ADM0) A/D input selection register 0 (ADS0)

Figure 13-1. Block Diagram of 10-Bit A/D Converter



(1) Successive approximation register (SAR)

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

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

Caution When the μPD78F9177, a flash memory counterpart of the μPD789166 or μPD789167, is used, the register can be accessed in 8-bit units. However, only an object file assembled with the μPD789166 or μPD789167 can be used. The same is also true for the μPD78F9177Y, a flash memory counterpart of the μPD789166Y or μPD789167Y. When the μPD78F9177Y is used, the register can be accessed in 8-bit units. However, only an object file assembled with the μPD789166Y or μPD789167Y can be used.

(3) Sample-and-hold circuit

The sample-and-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 serial resistor string.

(5) Serial resistor string

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

(6) ANI0 to ANI7

Pins ANI0 to ANI7 are the 8-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 ANI7 with voltages that fall outside the rated range. If a voltage greater than AV_{REF} or less than AV_{SS} (even if within the absolute maximum rating) is supplied to any of these pins, the conversion value for the corresponding channel will be undefined. Furthermore, the conversion values for the other channels may also be affected.

(7) AV_{REF} pin

This pin inputs the A/D converter reference voltage.

It converts signals input to ANI0 to ANI7 into digital signals according to the voltage applied between AV_{REF} and AV_{SS}.

(8) 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_{SS0} pin, even while the A/D converter is not being used.

(9) 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_{DD0} pin, even while the A/D converter is not being used.

13.3 10-Bit A/D Converter Control Registers

The following two registers are used to control the 10-bit A/D converter.

- A/D converter mode register 0 (ADM0)
- A/D input selection 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 clears ADM0 to 00H.

Figure 13-2. 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. The result of conversion performed immediately after bit 7 (ADCS0) is set is undefined.

2. The conversion result may be undefined after ADCS0 has been cleared to 0 (For details, see 13.5 (5) Timing of undefined A/D conversion result).

Remarks 1. fx: Main system clock oscillation frequency

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

(2) A/D input selection 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 13-3. Format of A/D Input Selection 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
1	1	0	ANI6
1	1	1	ANI7

Caution Bits 3 to 7 must all be set to 0.

13.4 10-Bit A/D Converter Operation

13.4.1 Basic operation of 10-bit A/D converter

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

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

<3> After sampling continues for a certain period of time, the sample and 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_{REF}.

<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_{REF}, the MSB of SAR is left set. If it is lower than half of AV_{REF}, 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_{REF}
- Bit 9 = 0: One quarter of AV_{REF}

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

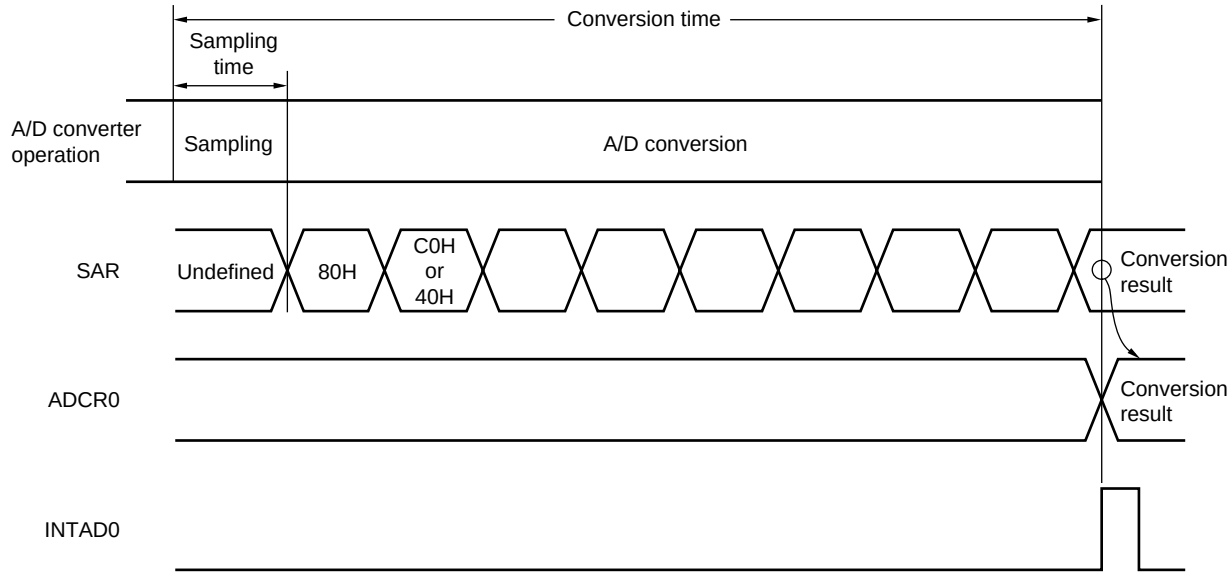
<7> Comparison is repeated until bit 0 of SAR is reached.

<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 13-4. 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 to 0 by software.

If an attempt is made to write to ADM0 or A/D input selection 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 to 1.

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

13.4.2 Input voltage and conversion result

The relationships between the analog input voltage at the analog input pins (ANI0 to ANI7) and the A/D conversion result (A/D conversion result register 0 (ADCR0)) are represented by:

$$ADCR0 = INT\left(\frac{V_{IN}}{AV_{REF}} \times 1{,}024 + 0.5\right)$$

or

$$(ADCR0 - 0.5) \times \frac{AV_{REF}}{1{,}024} \leq V_{IN} < (ADCR0 + 0.5) \times \frac{AV_{REF}}{1{,}024}$$

INT( ): Function that returns the integer part of a parenthesized value

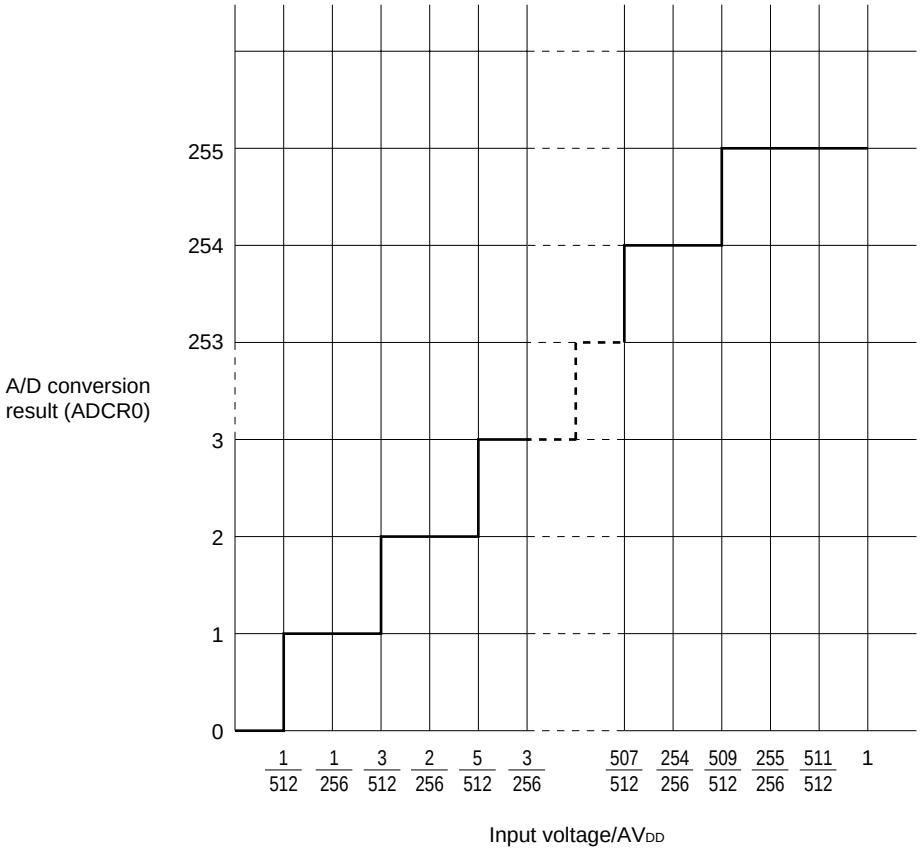
V_{IN}: Analog input voltage

AV_{REF}: Voltage of AV_{REF} pin

ADCR0: Value in A/D conversion result register 0 (ADCR0)

Figure 13-5 shows the relationship between the analog input voltage and the A/D conversion result.

Figure 13-5. Relationship between Analog Input Voltage and A/D Conversion Result



13.4.3 Operation mode of 10-bit A/D converter

The A/D converter is initially in select mode. In this mode, A/D input selection register 0 (ADS0) is used to select an analog input channel from ANI0 to ANI7 for A/D conversion.

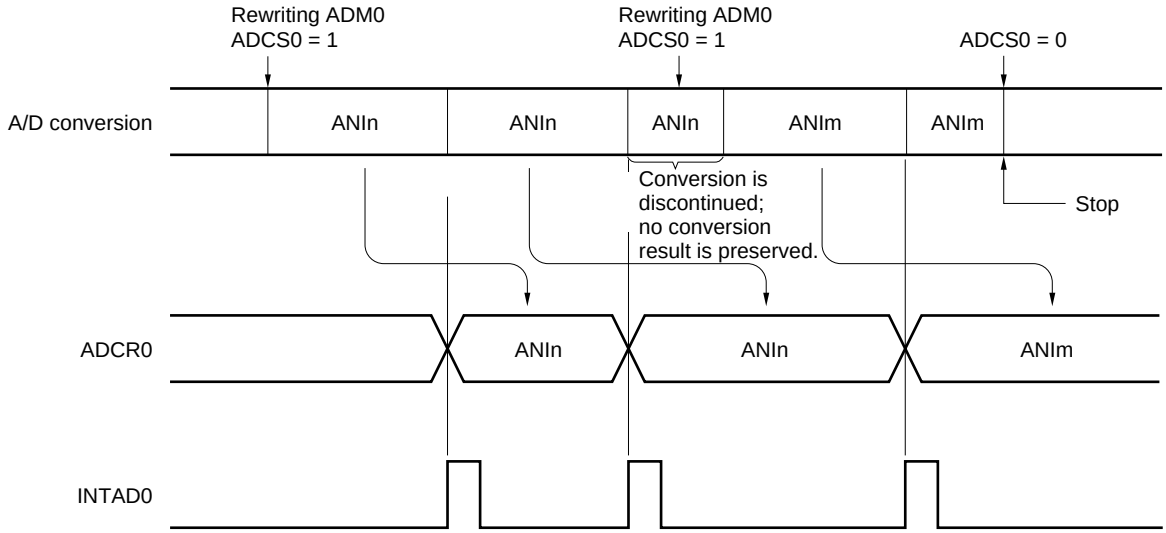
A/D conversion can be started only by software, that is, by setting A/D converter mode register 0 (ADM0).

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 13-6. Software-Started A/D Conversion



Remarks 1. n = 0 to 7

2. m = 0 to 7

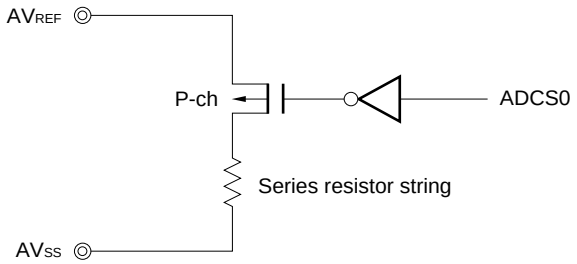
13.5 Cautions Related to 10-Bit A/D Converter

(1) Current drain in standby mode

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

Figure 13-7 shows how to reduce the current drain in standby mode.

Figure 13-7. How to Reduce Current Drain in Standby Mode



(2) Input range for pins ANI0 to ANI7

Be sure to keep the input voltage at ANI0 to ANI7 within its rating. If a voltage not lower than AV_{REF} 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. It 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 A/D input selection 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 13-8 and 13-9 show the timing at which the conversion result is read.

Figure 13-8. Conversion Result Read Timing (If Conversion Result is Undefined)

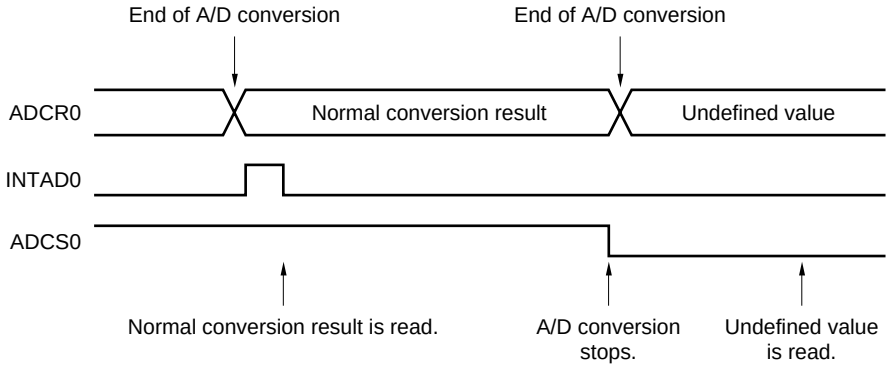
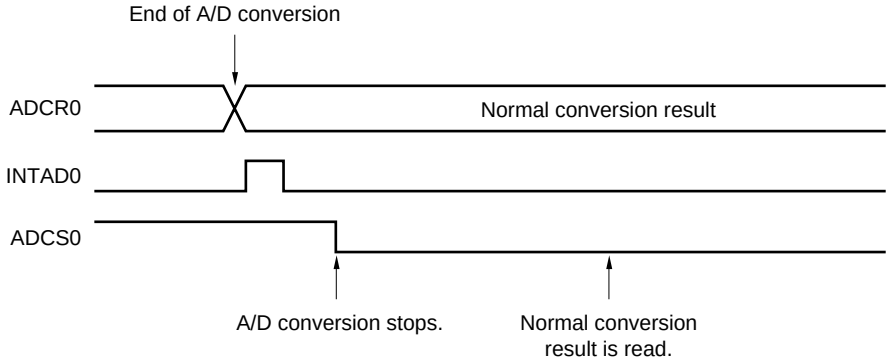


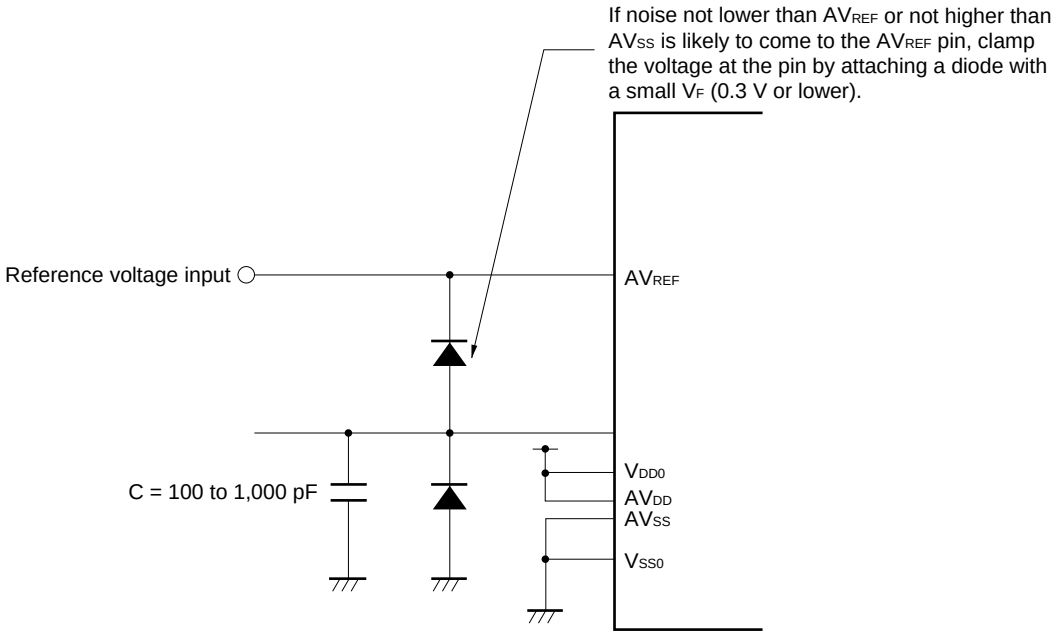
Figure 13-9. Conversion Result Read Timing (If Conversion Result is Normal)



(6) Noise prevention

To maintain a resolution of 10 bits, watch for noise to pins AV_{REF} and ANI0 to ANI7. The higher the output impedance of the analog input source is, the larger the effect by noise is. To reduce noise, attach an external capacitor to the relevant pins as shown in Figure 13-10.

Figure 13-10. Analog Input Pin Treatment



(7) ANI0 to ANI7

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

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

If a digital pulse is applied to a pin adjacent to the analog input pins during A/D conversion, coupling noise may occur which 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) Input impedance of the AV_{REF} pin

A series resistor string of several 10 kΩ is connected across the AV_{REF} and AV_{SS} pins.

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

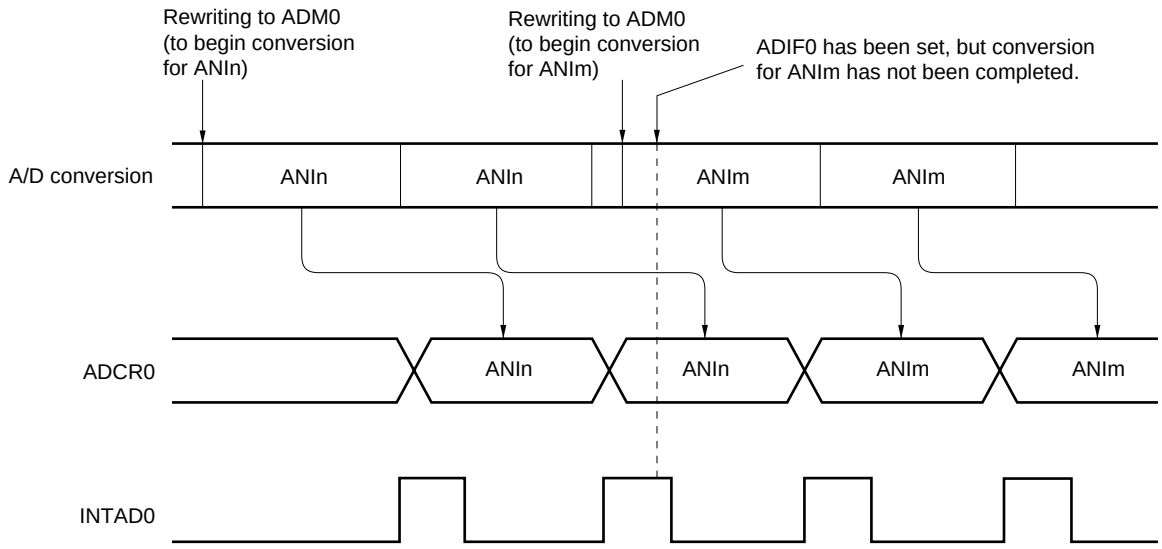
(9) Interrupt request flag (ADIF0)

Changing the content 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 13-11. A/D Conversion End Interrupt Request Generation Timing



Remarks 1. n = 0 to 7

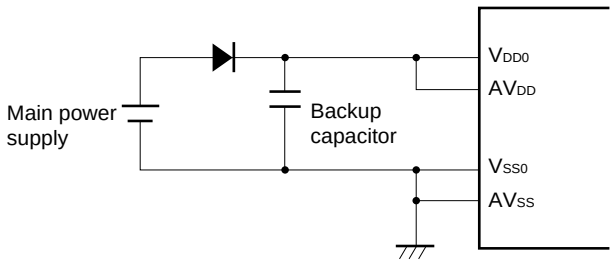
2. m = 0 to 7

(10) AV_{DD} pin

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

If your application is designed to be changed to backup power, the AV_{DD} pin must be supplied with the same voltage level as for the V_{DD0} pin, as shown in Figure 13-12.

Figure 13-12. AV_{DD} Pin Treatment



[MEMO]

CHAPTER 14 SERIAL INTERFACE 20

14.1 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 ($\overline{SCK20}$) 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.

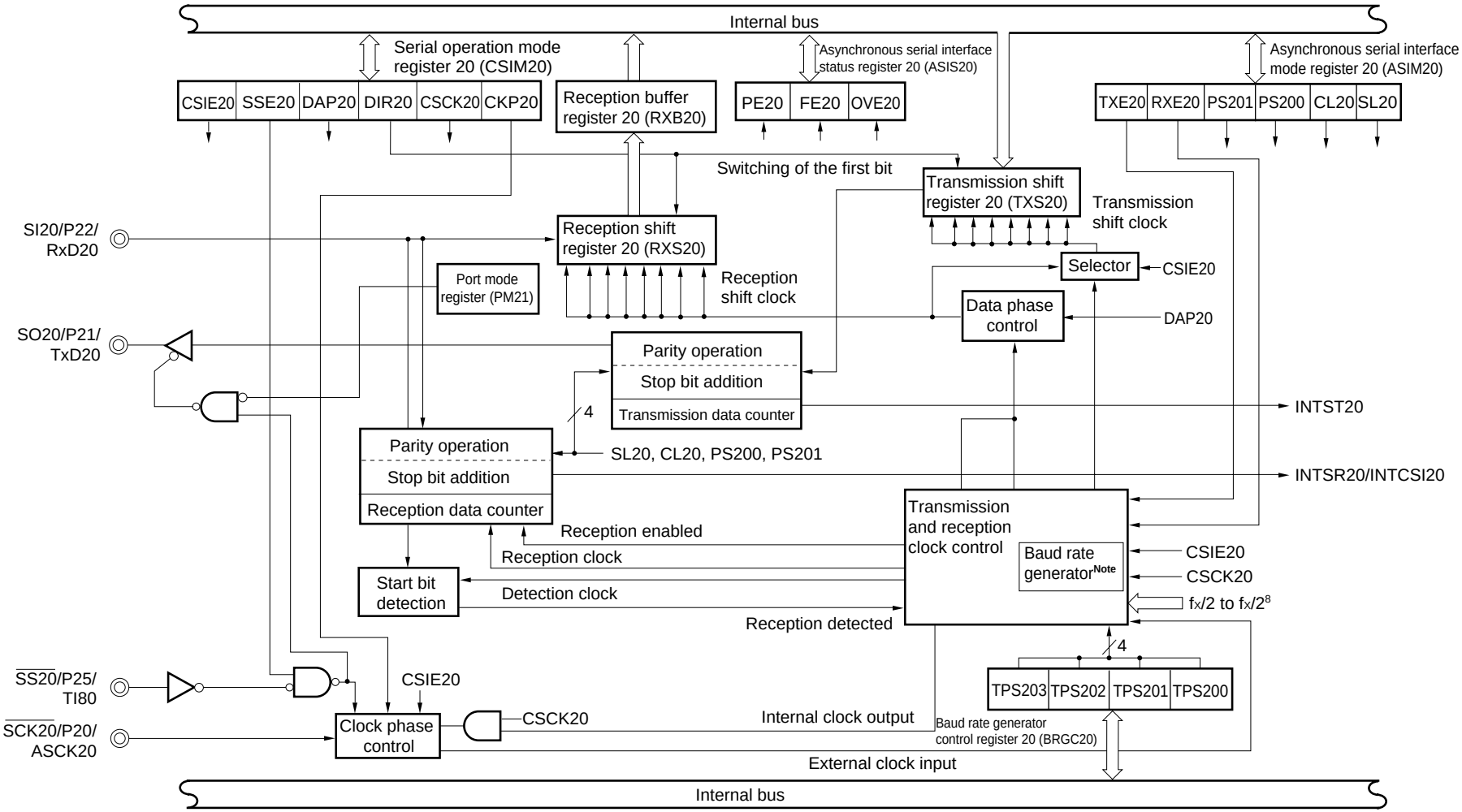
14.2 Serial Interface 20 Configuration

Serial interface 20 consists of the following hardware.

Table 14-1. Configuration of Serial Interface 20

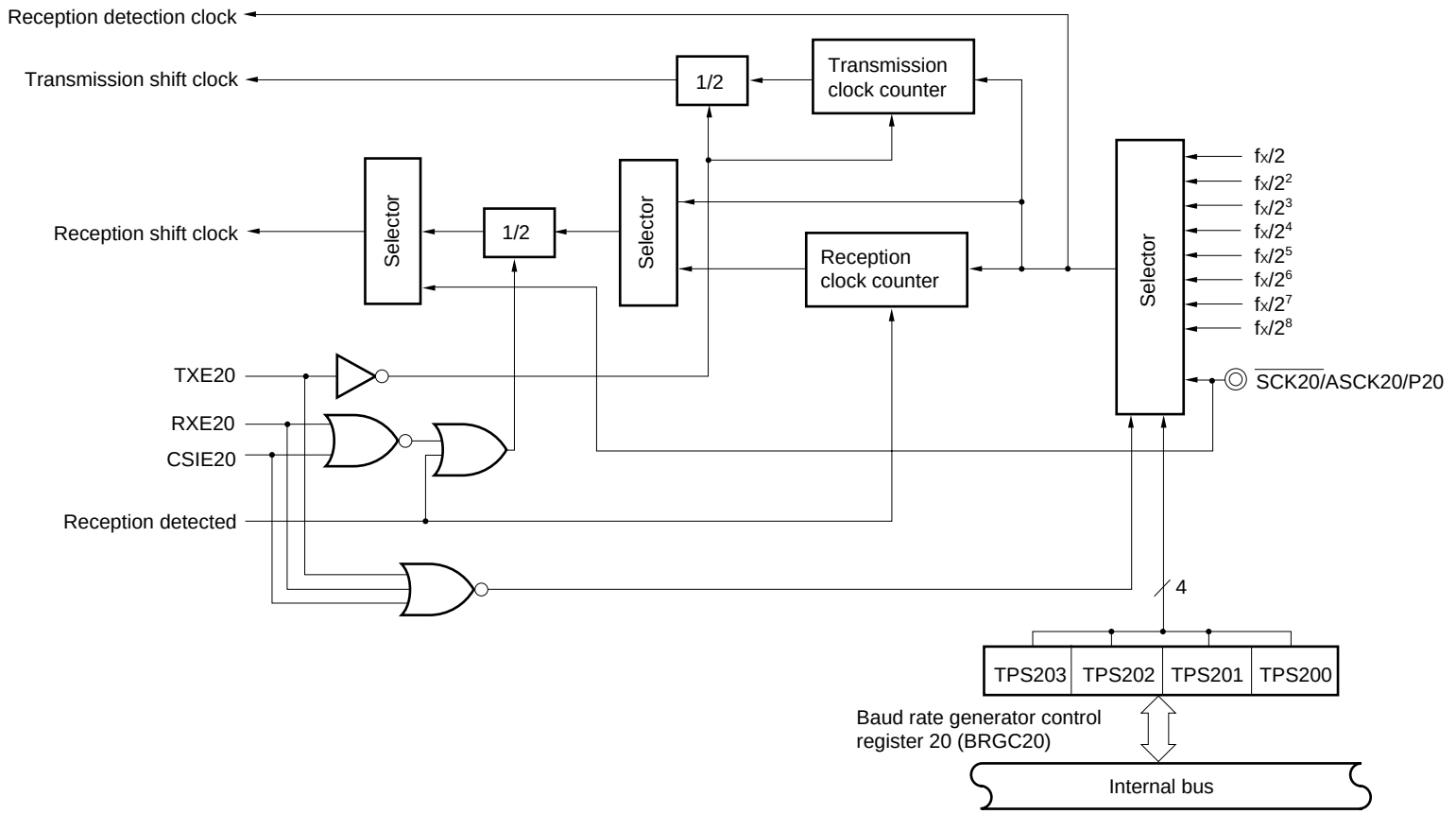
Item	Configuration
Register	Transmission shift register 20 (TXS20) Reception shift register 20 (RXS20) Reception buffer register 20 (RXB20)
Control register	Serial operation mode register 20 (CSIM20) Asynchronous serial interface mode register 20 (ASIM20) Asynchronous serial interface status register 20 (ASIS20) Baud rate generator control register 20 (BRGC20)

Figure 14-1. Block Diagram of Serial Interface 20



Note See **Figure 14-2** for the configuration of the baud rate generator.

Figure 14-2. 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 control circuit

The transmission control circuit 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 control circuit

The reception control circuit 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.

14.3 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 clears CSIM20 to 00H.

Figure 14-3. 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 $\overline{\text{SS20}}$/P23 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 all 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 clears ASIM20 to 00H.

Figure 14-4. 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 all be set to 0.

2. If 3-wire serial I/O mode is selected, ASIM20 must be cleared to 00H.

3. Switch operating modes after halting serial transmit/receive operation.

Table 14-2. Serial Interface 20 Operating Mode Settings

(1) Operation stop mode

ASIM20		CSIM20			PM22	P22	PM21	P21	PM20	P20	First Bit	Shift Clock	P22/SI20/ RxD20 Pin Function	P21/SO20/ TxD20 Pin Function	P20/$\overline{\text{SCK20}}$/ ASCK20 Pin Function
TXE20	RXE20	CSIE20	DIR20	CSCK20											
0	0	0	×	×	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	×Note 1	–	–	P22	P21	P20
Other than above											Setting prohibited				

(2) 3-wire serial I/O mode

ASIM20		CSIM20			PM22	P22	PM21	P21	PM20	P20	First Bit	Shift Clock	P22/SI20/ RxD20 Pin Function	P21/SO20/ TxD20 Pin Function	P20/$\overline{\text{SCK20}}$/ 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			PM22	P22	PM21	P21	PM20	P20	First Bit	Shift Clock	P22/SI20/ RxD20 Pin Function	P21/SO20/ TxD20 Pin Function	P20/$\overline{\text{SCK20}}$/ 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			P20
0	1	0	0	0	1	×	×Note 1	×Note 1	1	×		External clock	RxD20	P21	ASCK20 input
									×Note 1	×Note 1		Internal clock			P20
1	1	0	0	0	1	×	0	1	1	×		External clock		TxD20 (CMOS output)	ASCK20 input
									×Note 1	×Note 1		Internal clock			P20
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 P22 (CMOS input/output).

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 clears ASIS20 to 00H.

Figure 14-5. 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 is generated.

(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 clears BRGC20 to 00H.

Figure 14-6. 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 is performed during a communication operation, the output of baud rate generator is disrupted and communications cannot be performed normally. Be sure not to write to BRGC00 during communication operations.

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 14-6, determined by the values of TPS200 to TPS203 ($2 \leq n \leq 8$)

Table 14-3. 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 14-4. 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

14.4 Serial Interface 20 Operation

Serial interface 20 provides the following three types of modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode
- 3-wire serial I/O mode

14.4.1 Operation stop mode

In operation stop mode, serial transfer is not executed; therefore, the power consumption can be reduced. The P20/$\overline{\text{SCK20}}$/ASCK20, P21/SO20/TxD20, and P22/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 disable
1	Operation enable

Caution Bits 4 and 5 must all 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 clears 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 stop
1	Transmit operation enable

RXE20	Receive operation control
0	Receive operation stop
1	Receive operation enable

Caution Bits 0 and 1 must all be set to 0.

14.4.2 Asynchronous serial interface (UART) mode

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

This device incorporates an UART-dedicated baud rate generator that enables communications at a desired baud rate from many options. 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 clears 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 $\overline{\text{SS20}}$/P25 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 all 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 clears 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 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	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 all be set to 0.

2. Switch operating modes after halting 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 clears 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	Parity error not generated
1	Parity error generated (when the parity of transmit data does not match)

FE20	Framing error flag
0	Framing error not generated
1	Framing error generated (when stop bit is not detected)Note 1

OVE20	Overrun error flag
0	Overrun error not generated
1	Overrun error generatedNote 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 is generated.

(d) Baud rate generator control register 20 (BRGC20)

BRGC20 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears 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	–
Other than above				Setting prohibited	

Cautions 1. When writing to BRGC20 is performed during a communication operation, the output of baud rate generator is disrupted and communications cannot be performed normally. Be sure not to write to BRGC20 during communication operation.

2. Be sure not to select n = 1 during an 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 14-5. 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 14-6. 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

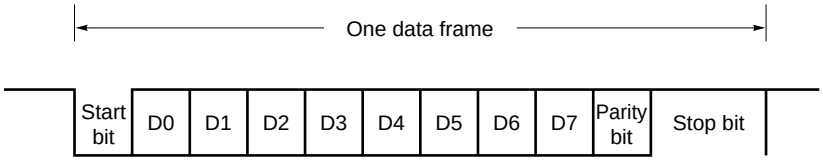
(2) Communication operation

(a) Data format

The transmit/receive data format is as shown in Figure 14-7. 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 14-7. Asynchronous Serial Interface Transmit/Receive Data Format



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

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 the baud rate generator control register 20 (BRGC20).

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

(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 is generated.

(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 is not generated, 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 is not generated.

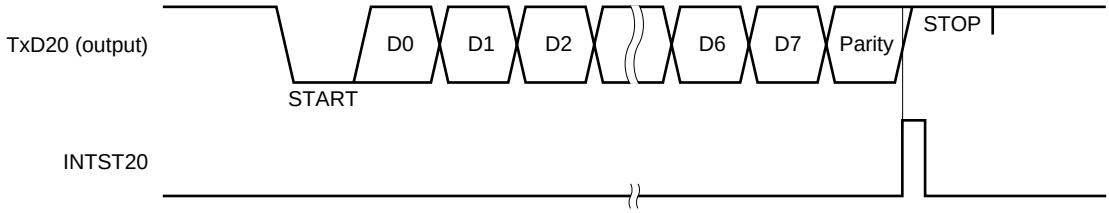
(c) Transmission

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.

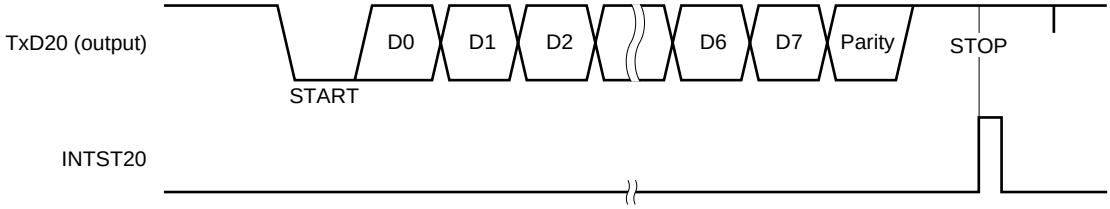
When the transmit operation starts, the data in TXS20 is shifted out, and when TXS20 is empty, a transmission completion interrupt (INTST20) is generated.

Figure 14-8. Asynchronous Serial Interface Transmission Completion Interrupt Timing

(a) Stop bit length: 1



(b) Stop bit length: 2



Caution Do not rewrite to asynchronous serial interface mode register 20 (ASIM20) during a transmit operation. If the ASIM20 register is rewritten to during transmission, subsequent transmission may not 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 to 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.

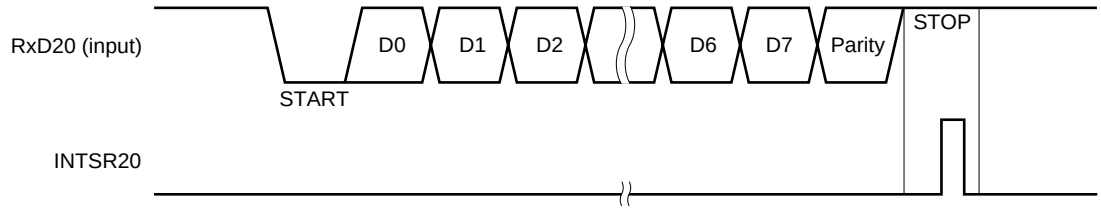
When the RxD20 pin input becomes low, the 3-bit counter starts counting, and at the time when half the time determined by the specified baud rate has passed, the data sampling start timing signal is output. If the RxD20 pin input sampled again as a result of this start timing signal is low, it is identified 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 is generated, the receive data in which the error was generated is still transferred to RXB20, and INTSR20 is generated.

If the RXE20 bit is reset to 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 14-9. 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 be generated 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 14-7.

It is possible to determine what kind of error was generated during reception by reading the contents of ASIS20 in the reception error interrupt servicing (see **Figures 14-9** and **14-10**).

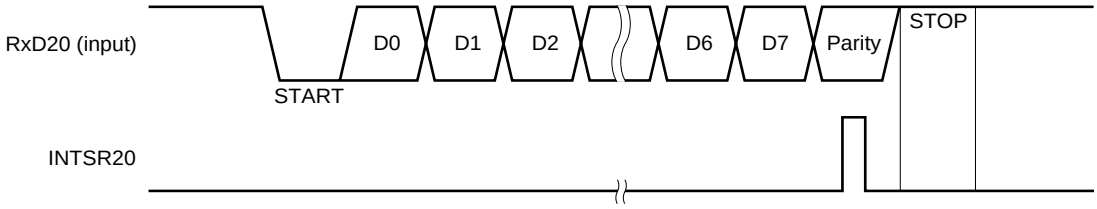
The contents of ASIS20 are reset to 0 by reading reception buffer register 20 (RXB20) or receiving the next data (if there is an error in the next data, the corresponding error flag is set).

Table 14-7. Receive Error Causes

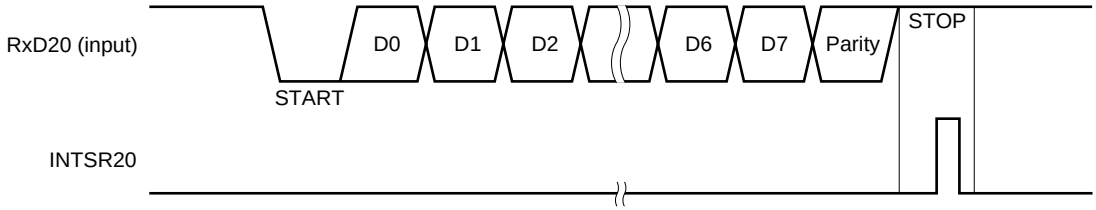
Receive Errors	Cause
Parity error	Transmission-time parity and reception data parity do not match
Framing error	Stop bit not detected
Overrun error	Reception of next data is completed before data is read from reception buffer register

Figure 14-10. Receive Error Timing

(a) Parity error generated



(b) Framing error or overrun error generated



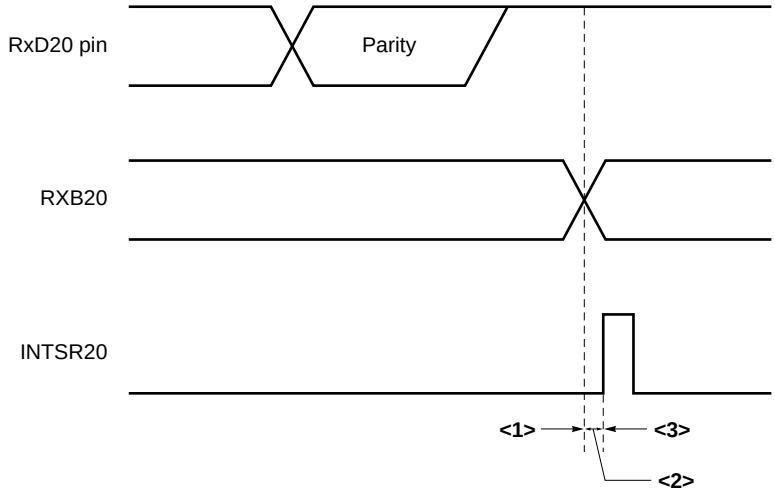
Cautions 1. The contents of the ASIS20 register are reset to 0 by reading reception buffer register 20 (RXB20) or receiving the next data. To ascertain the error contents, read ASIS20 before reading RXB20.

2. Be sure to read reception buffer register 20 (RXB20) even if a receive error is generated. If RXB20 is not read, an overrun error will be generated 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.

14.4.3 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 synchronous serial interface, such as the 75XL Series, 78K Series, 17K Series, etc.

Communication is performed using three lines: the 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 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	3-wire serial I/O mode operation control
0	Operation disabled
1	Operation enabled

SSE20	$\overline{SS20}$-pin selection	Function of $\overline{SS20}$/P25 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 all 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 clears 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 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	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 all be set to 0.

2. Switch operating modes after halting 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 clears 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 is performed 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 communication operation.

2. Be sure not to select n = 1 during an 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

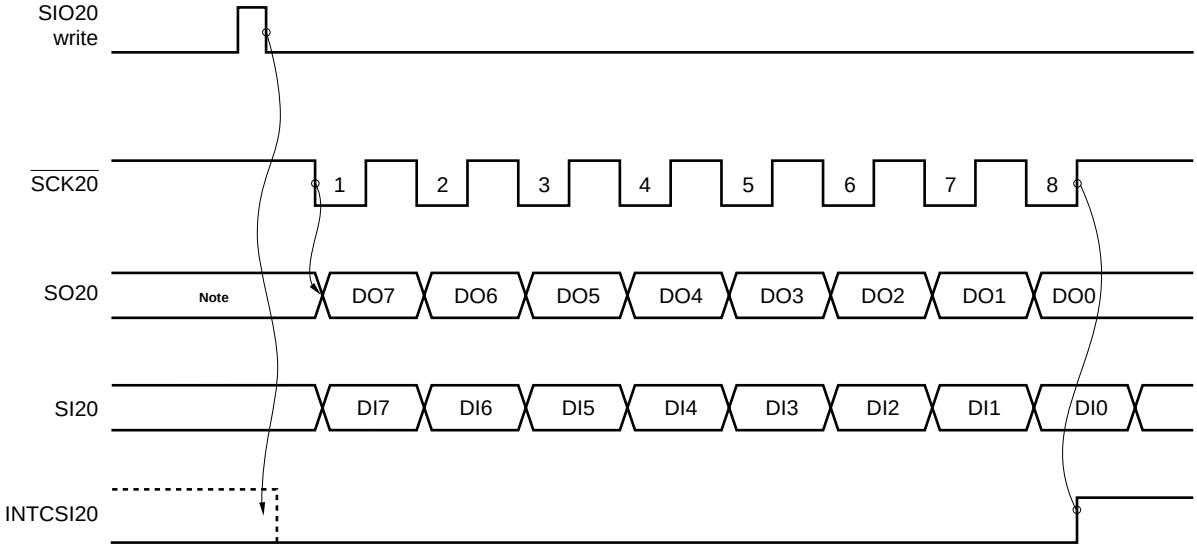
In 3-wire serial I/O mode, data transmission/reception is performed in 8-bit units. Data is transmitted/received bit by bit in synchronization with the serial clock.

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 14-11. 3-Wire Serial I/O Mode Timing (1/7)

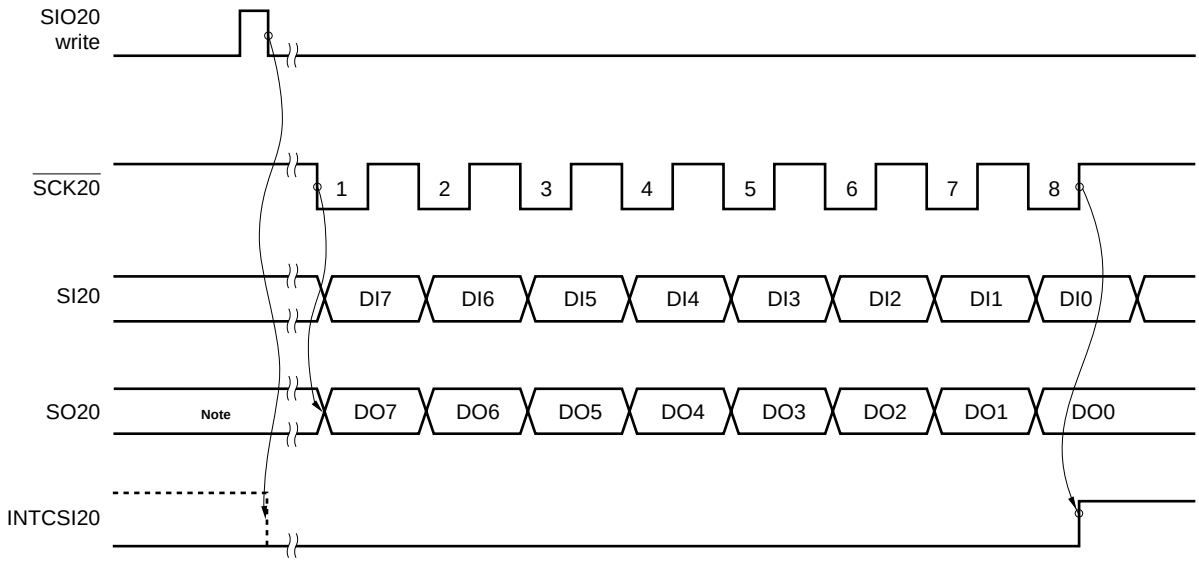
(i) Master operation timing (when DAP20 = 0, CKP20 = 0, SSE20 = 0)



Note The value of the last bit previously output is output.

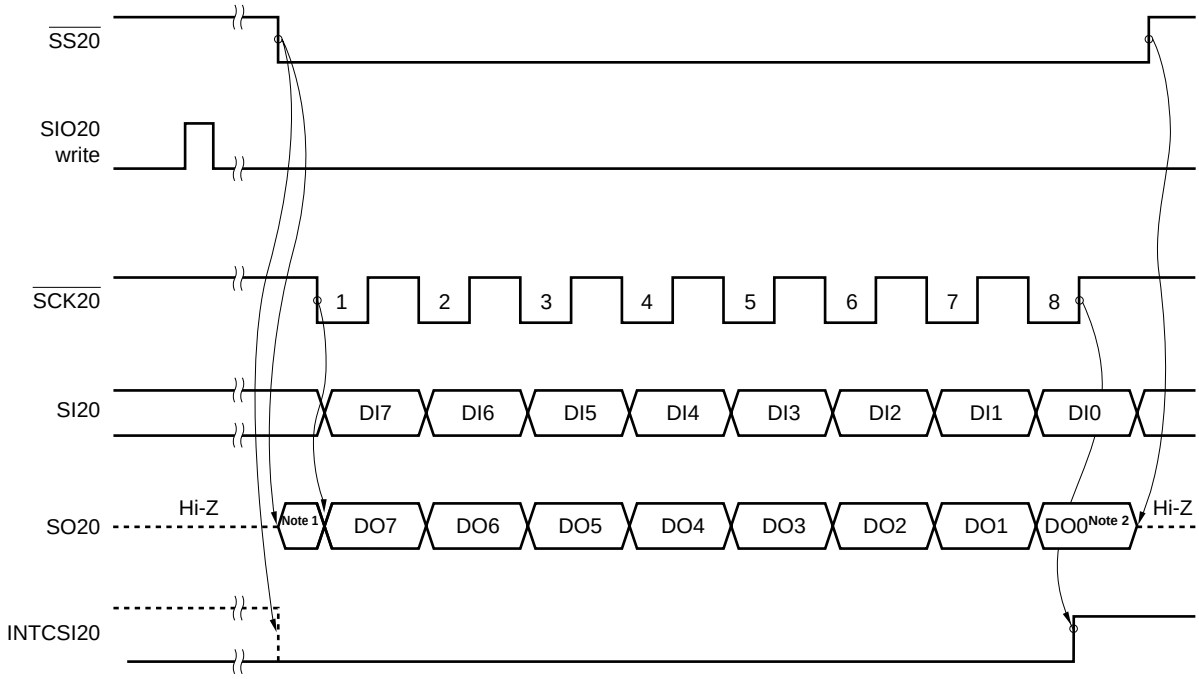
Figure 14-11. 3-Wire Serial I/O Mode Timing (2/7)

(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)

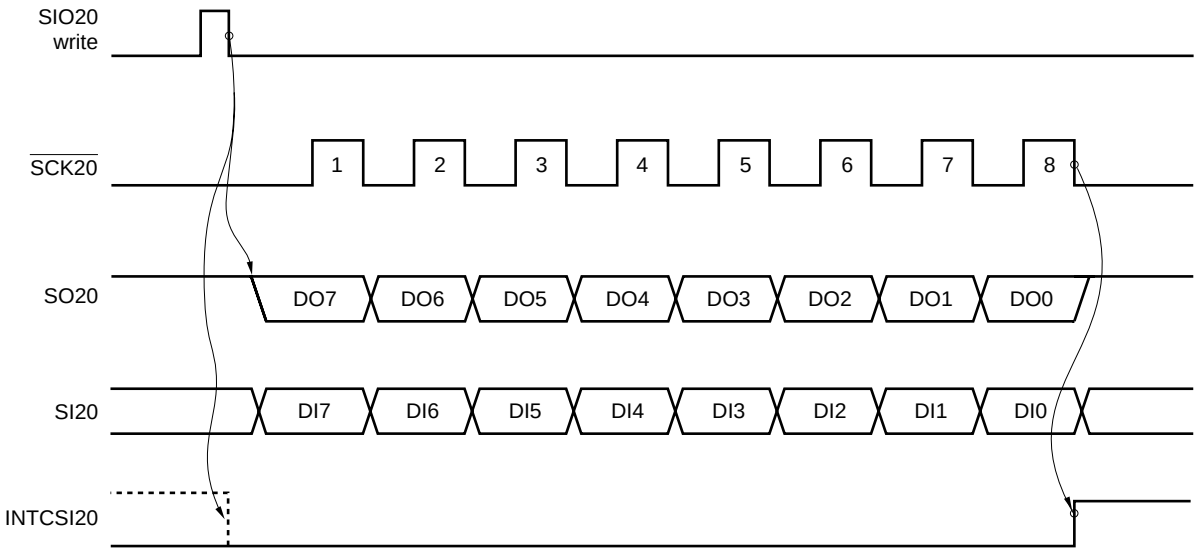


Notes 1. The value of the last bit previously output is output.

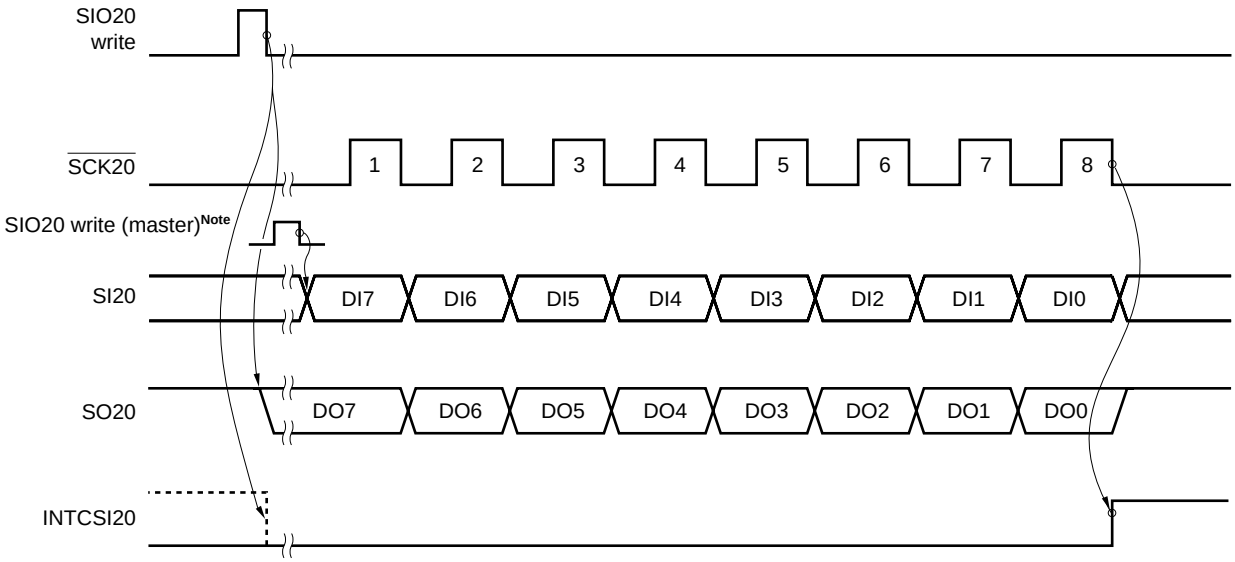
2. DO0 is output until $\overline{SS20}$ rises.
When $\overline{SS20}$ is high, SO20 is in a high-impedance state.

Figure 14-11. 3-Wire Serial I/O Mode Timing (3/7)

(iv) Master operation (when DAP20 = 0, CKP20 = 1, SSE20 = 0)



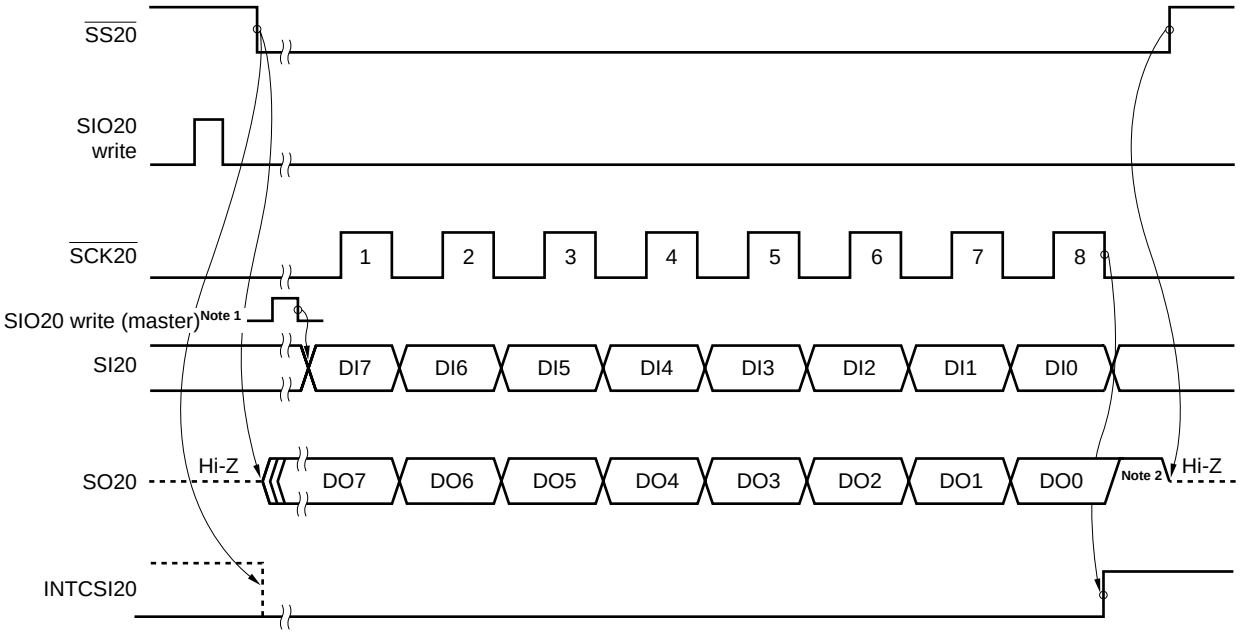
(v) Slave operation (when DAP20 = 0, CKP20 = 1, SSE20 = 0)



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 14-11. 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)

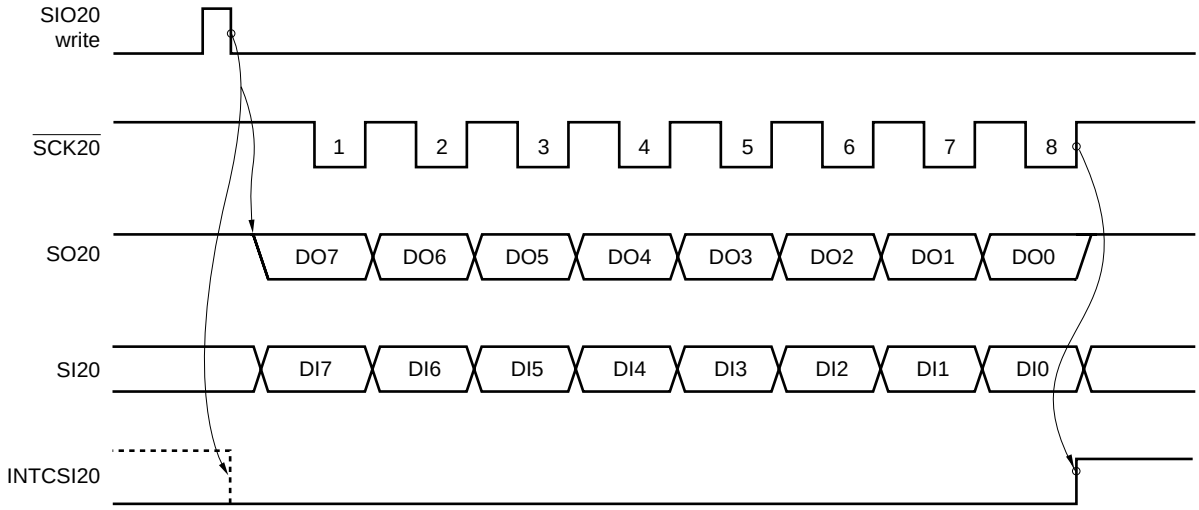
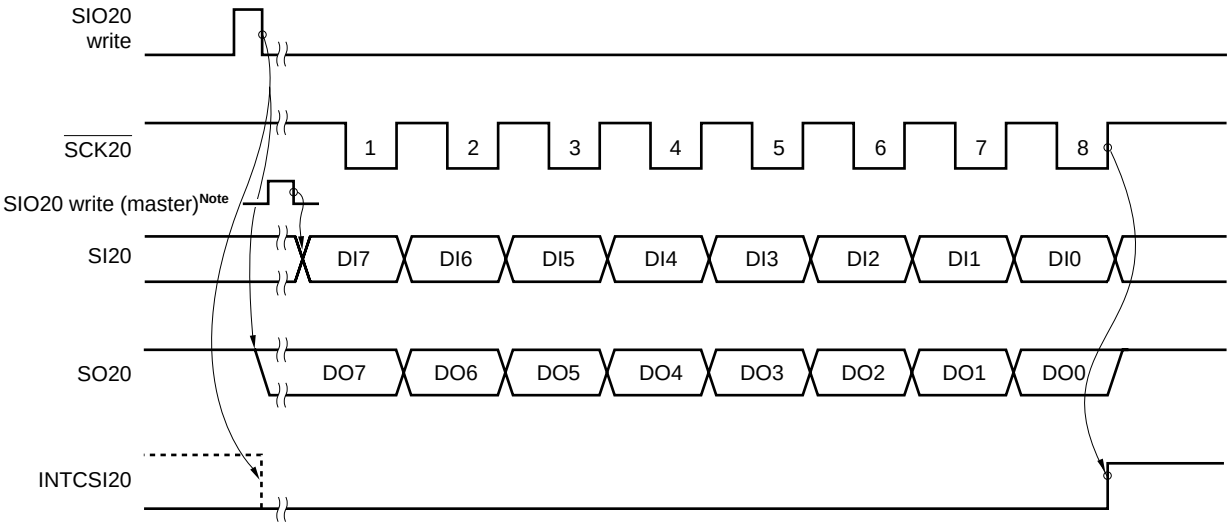


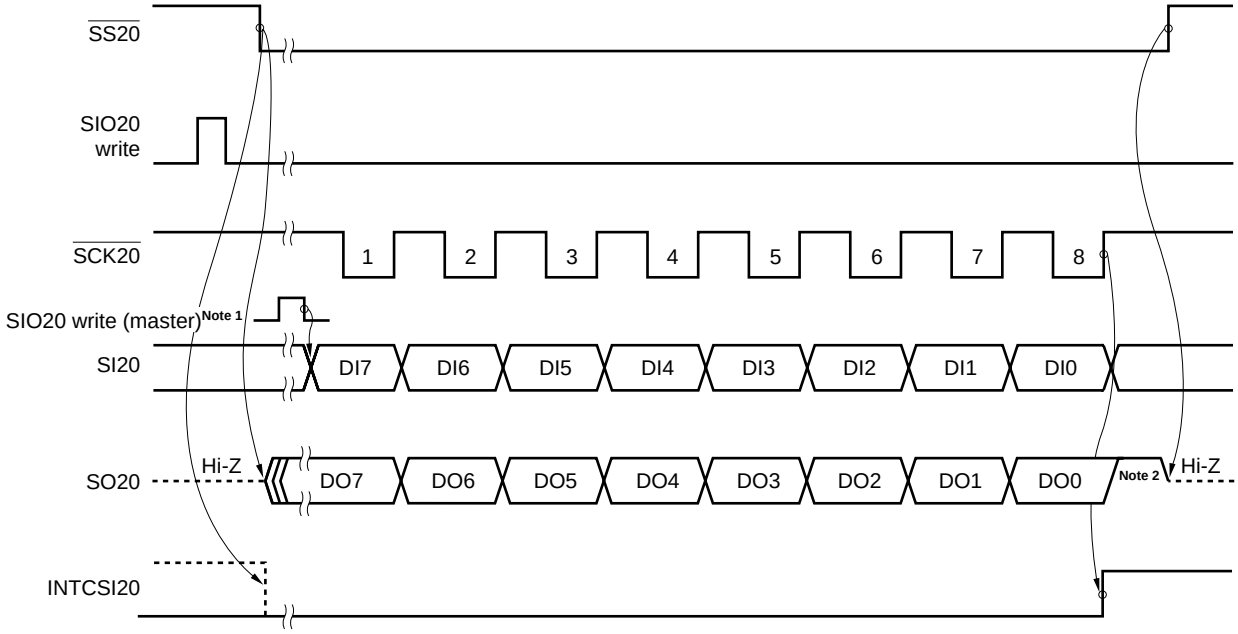
Figure 14-11. 3-Wire Serial I/O Mode Timing (5/7)

(viii) Slave operation (when DAP20 = 1, CKP20 = 0, SSE20 = 0)



Note The data of SI20 is loaded at the first falling edge of $\overline{SCK20}$. Make sure that the master outputs the first bit before the first falling of $\overline{SCK20}$.

(ix) Slave operation (when DAP20 = 1, CKP20 = 0, SSE20 = 1)

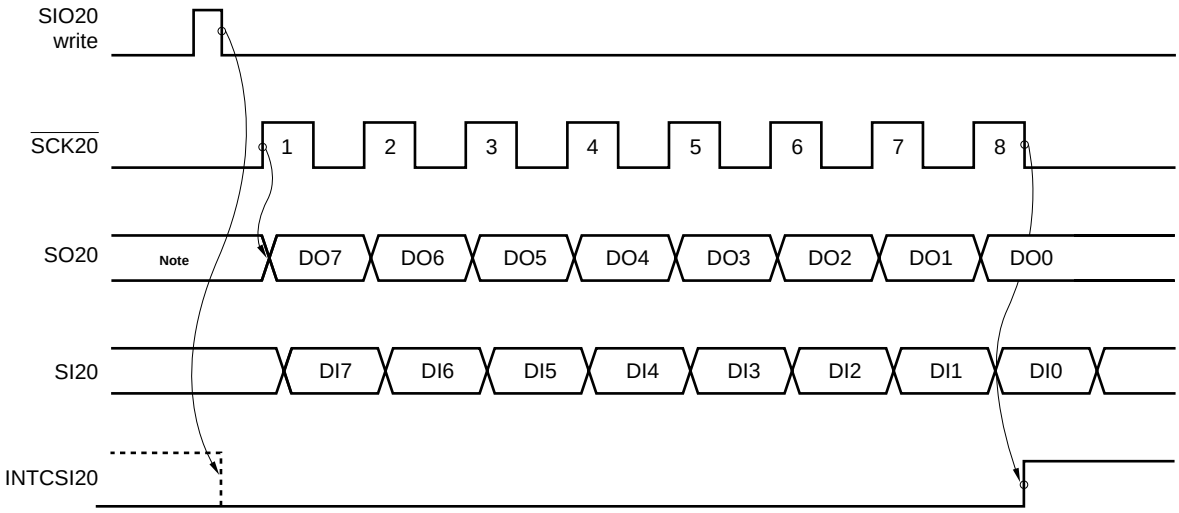


Notes 1. The data of SI20 is loaded at the first falling edge of $\overline{SCK20}$. Make sure that the master outputs the first bit before the first falling of $\overline{SCK20}$.

2. SO20 is high until $\overline{SS20}$ rises after completion of DO0 output. When $\overline{SS20}$ is high, SO20 is in a high-impedance state.

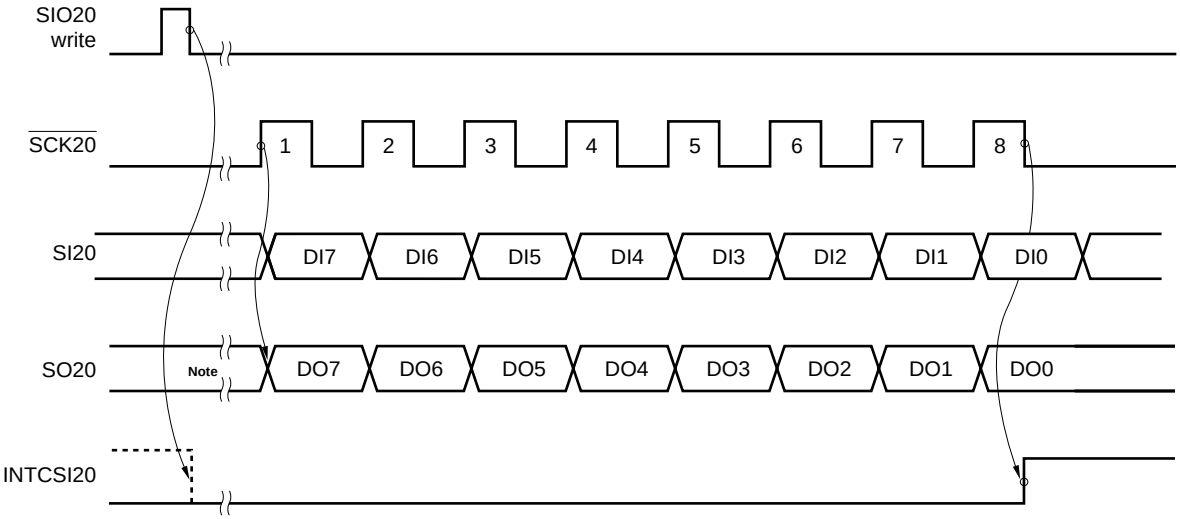
Figure 14-11. 3-Wire Serial I/O Mode Timing (6/7)

(x) Master operation (when DAP20 = 1, CKP20 = 1, SSE20 = 0)



Note The value of the last bit previously output is output.

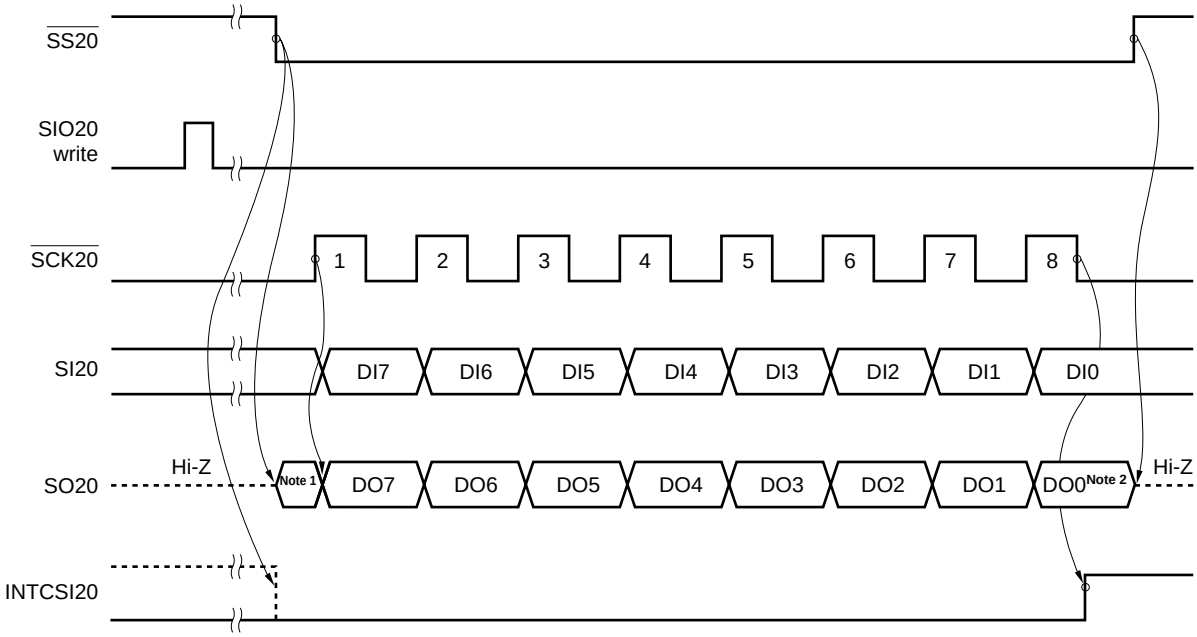
(xi) Slave operation (when DAP20 = 1, CKP20 = 1, SSE20 = 0)



Note The value of the last bit previously output is output.

Figure 14-11. 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.

- Serial operation mode register 20 (CSIM20) bit 7 (CSIE20) = 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.

A termination of 8-bit transfer stops the serial transfer automatically and generates the interrupt request signal (INTCSI20).

★ CHAPTER 15 SMB0 (μPD789167Y AND 789177Y SUBSERIES)

15.1 SMB0 Functions

SMB0 (system management bus) has the following two types of modes.

- Operation stop mode
- SMB mode (supporting multiple masters)

(a) Operation stop mode

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

(b) SMB mode (supporting multiple masters)

This mode is used for performing 8-bit data transmission to several devices, using a serial clock (SCL0) line and a serial data bus (SDA0) line.

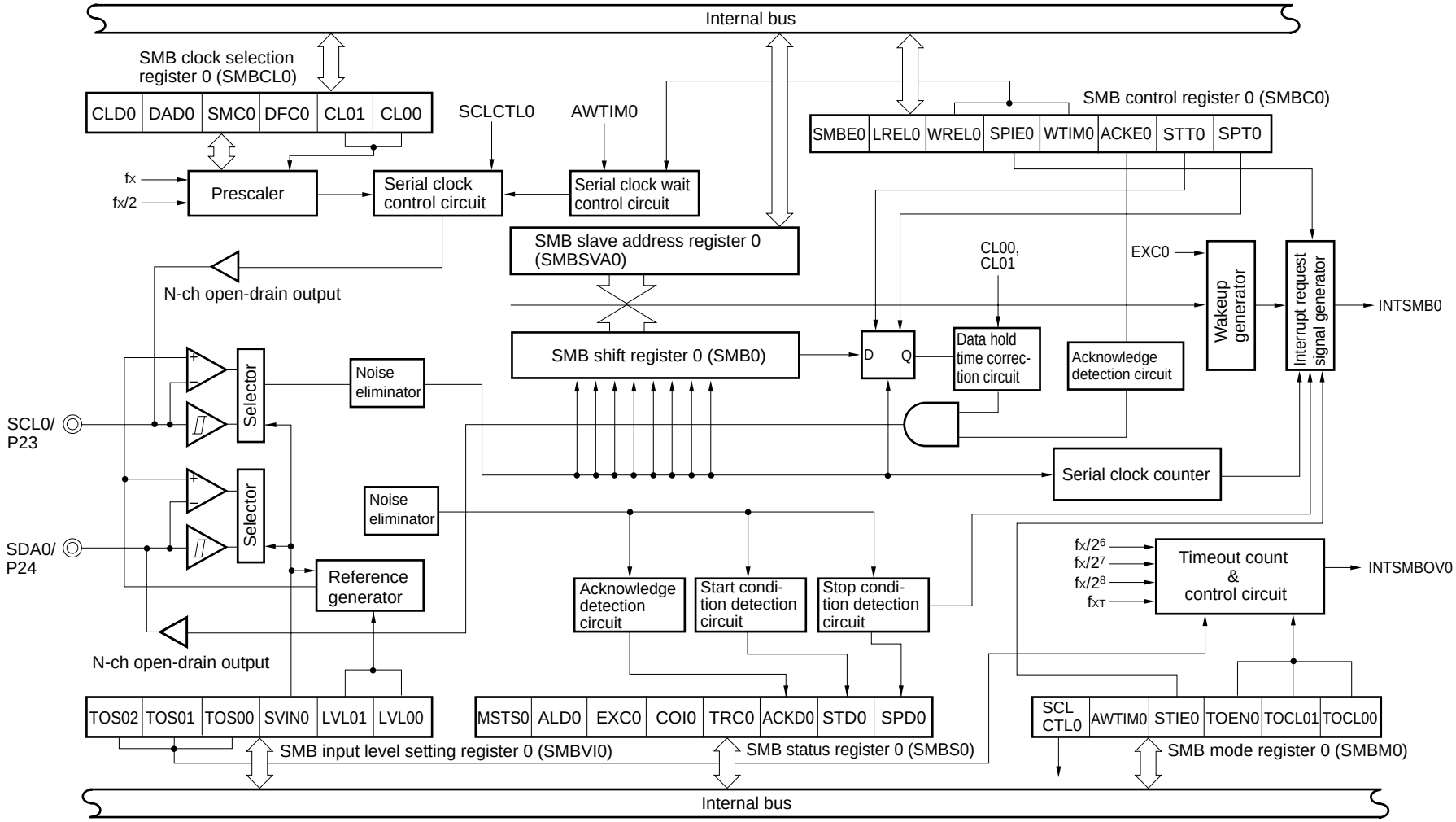
In this mode, which conforms to the SMB format, start conditions, data, and stop conditions can be output on the serial data bus during transmission. Moreover, these data can be automatically detected by hardware during reception.

In SMB0, SCL0 and SDA0 are open-drain outputs, and therefore a pull-up resistor is required for the serial clock line and serial data bus line.

I^2C (Inter IC) bus standard mode or high-speed mode can be specified by software in SMB mode.

Figure 15-1 shows the block diagram of SMB0.

Figure 15-1. Block Diagram of SMB0



15.2 SMB0 Configuration

SMB0 consists of the following hardware.

Table 15-1. Configuration of SMB0

Item	Configuration
Register	SMB shift register 0 (SMB0) SMB slave address register 0 (SMBSVA0)
Control register	SMB control register 0 (SMBC0) SMB status register 0 (SMBS0) SMB clock selection register 0 (SMBCL0) SMB mode register 0 (SMBM0) SMB input level setting register 0 (SMBVI0)

(1) SMB shift register 0 (SMB0)

SMB0 is a register that converts 8-bit serial data to 8-bit parallel data, and vice-versa. SMB0 is used both for sending and receiving data.

Write and read operations for SMB0 control actual send and receive operations.

SMB0 is manipulated with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears SMB0 to 00H.

(2) SMB slave address register 0 (SMBSVA0)

This register is used to set a local address when used as a slave.

SMBSVA0 is manipulated with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears SMBSVA0 to 00H.

(3) SO latch

The SO latch is a latch that holds the SDA0 pin output level.

(4) Wakeup control circuit

This circuit generates an interrupt request when the address value set in SMB slave address register 0 (SMBSVA0) and the received address match, or when an extension code is received.

(5) Clock selector

Selects the sampling clock to be used.

(6) Serial clock counter

Counts the serial clock output/input during send/receive operations, to check if 8-bit data has been sent or received.

(7) Interrupt request signal generation circuit

Controls the generation of interrupt request signals.

SMB interrupts are generated with the following two triggers.

- 8th clock or 9th clock of serial clock (set with WTIM0 bit[Note])
- Generation of interrupt request at detection of stop condition (set with bit SPIE0[Note])

Note WTIM0 bit: SMB control register 0 (SMBC0) bit 3
SPIE0 bit: SMB control register 0 (SMBC0) bit 4

(8) Serial clock control circuit

In master mode, generates the clock to be output to the SCL0 pin from the sampling clock.

(9) Serial clock wait control circuit

Controls the wait timing.

(10) Acknowledge output circuit, stop condition detection circuit, start condition detection circuit, acknowledge detection circuit

Perform output and detection of control signals.

(11) Data hold time correction circuit

Generates the data hold time from the falling edge of the serial clock.

15.3 SMB0 Control Registers

The following five types of registers are used to control SMB0.

- SMB control register 0 (SMBC0)
- SMB status register 0 (SMBS0)
- SMB clock selection register 0 (SMBCL0)
- SMB mode register 0 (SMBM0)
- SMB input level setting register 0 (SMBVI0)

The following additional registers are also used.

- SMB shift register 0 (SMB0)
- SMB slave address register 0 (SMBSVA0)

(1) SMB control register 0 (SMBC0)

This register sets SMB operation enable/disable, the wait timing, and other SMB operations.

SMBC0 is manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears SMBC0 to 00H.

Caution Set port mode register 2 (PM2×) as follows in SMB mode.
Reset the output latch to 0.

- **Set P23 (SCL0) in output mode (PM23 = 0).**
- **Set P24 (SDA0) in output mode (PM24 = 0).**

Figure 15-2. Format of SMB Control Register 0 (1/4)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SMBC0	SMBE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0	FF78H	00H	R/W

SMBE0	SMB operation
0	Operation disabled. Presets extension register (SMBS0). Internal operation also disabled.
1	Operation enabled

Clear conditions (SMBE0 = 0)	Set conditions (SMBE0 = 1)
• Cleared with instruction • Cleared by $\overline{\text{RESET}}$ input	• Set with instruction

LREL0	Escape from transmission
0	Normal operation
1	Escape from the current transmission and enters the standby status. Automatically cleared after execution. This bit is used when extension codes not relevant to the local station are received. The SCL0 and SDA0 lines enter the high impedance status. The following flags are cleared. • STD0 • STT0 • SPT0 • ACKD0 • TRC0 • COI0 • EXC0 • MSTS0

The standby status continues until the following communication participation conditions are met.
- Startup as master after detection of stop condition
- Matching addresses or extension code reception after start condition

Clear conditions (LREL0 = 0)[Note]	Set conditions (LREL0 = 1)
• Automatically cleared after execution • Cleared by $\overline{\text{RESET}}$ input	• Set with instruction

WREL0	Wait cancel
0	Does not cancel wait.
1	Cancels wait. Automatically cleared after wait cancellation.

Clear conditions (WREL0 = 0)[Note]	Set conditions (WREL0 = 1)
• Automatically cleared after execution • Cleared by $\overline{\text{RESET}}$ input	• Set with instruction

SPIE0	Interrupt request generation at stop condition detection
0	Disabled
1	Enabled

Clear conditions (SPIE0 = 0)[Note]	Set conditions (SPIE0 = 1)
• Cleared with instruction • Cleared by $\overline{\text{RESET}}$ input	• Set with instruction

Note This flag's signals are made invalid by setting SMBE0 = 0.

Figure 15-2. Format of SMB Control Register 0 (2/4)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SMBC0	SMBE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0	FF78H	00H	R/W

WTIM0	Wait and interrupt request generation control
0	Generates interrupt request at falling edge of 8th clock. In case of master: Waits with clock output at low level after 8 clocks have been output. In case of slave: Waits master with clock set to low level after 8 clocks have been input.
1	Generates interrupt request at falling edge of 9th clock. In case of master: Waits with clock at low level after 9 clocks have been output. In case of slave: Waits master with clock set at low level after 9 clocks have been input.
The setting of this bit becomes invalid during address transmission, and becomes effective at the end of transmission. During operation as master, a wait is inserted at the falling edge of the 9th clock during address transmission. A slave that receives a local address enters the wait status at the falling edge of the 9th clock after generation of an acknowledge. A slave that receives an extension code enters the wait status at the falling edge of the 8th clock.	
Clear conditions (WTIM0 = 0)[Note]	Set conditions (WTIM0 = 1)
• Cleared with instruction • Cleared by $\overline{\text{RESET}}$ input	• Set with instruction

ACKE0	Acknowledge control
0	Acknowledge disabled
1	Acknowledge enabled. SDA0 line set to low level during 9 clocks. However, invalid during address transmission, and valid when EXC0 = 1.
Clear conditions (ACKE0 = 0)[Note]	Set conditions (ACKE0 = 1)
• Cleared with instruction • Cleared by $\overline{\text{RESET}}$ input	• Set with instruction

Note This flag's signals are made invalid by setting SMBE0 = 0.

Figure 15-2. Format of SMB Control Register 0 (3/4)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SMBC0	SMBE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0	FF78H	00H	R/W

STT0	Start condition trigger
0	Doesn't generate start condition.
1	When bus is released (stop status): Generates start conditions (activation as master). Changes SDA0 line from high level to low level and generates start condition. Then secures rated time and sets SCL0 to low level. When not participating on bus: Functions as start condition reservation flag. When set, automatically generates start condition after bus is released.

Cautions regarding set timing
- Master receive operation: Setting during transmission is prohibited.
  Set ACKE0 = 0; Can be set only after end of receive operation has been notified to slave.
- Master send operation: Note that start condition may not be generated normally during $\overline{\text{ACK}}$ period.
- Setting at the same time as SPT0 is prohibited.

Clear conditions (STT0 = 0)[Note]	Set conditions (STT0 = 1)
• Cleared with instruction • Cleared upon defeat in arbitration • Cleared after generation of start condition by master • Cleared by $\overline{\text{RESET}}$ input	• Set with instruction

Note This flag's signals are made invalid by setting SMBE0 = 0.

Figure 15-2. Format of SMB Control Register 0 (4/4)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SMBC0	SMBE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0	FF78H	00H	R/W

SPT0	Stop condition trigger
0	Does not generate stop condition.
1	Generates stop condition (end transmission as master). After setting SDA0 line to low level, sets SCL0 line to high level, or maintains SCL0 line at high level. Then, secures rated time, changes SDA0 line from low level to high level, and generates stop condition.

Cautions regarding set timing

- Master receive operation: Setting during transmission is prohibited.
  Set ACKE0 = 0; Can be set only after end of receive operation has been notified to slave.
- Master send operation: Note that stop condition may not be generated normally during $\overline{\text{ACK}}$ period.
- Setting at the same time as STT0 is prohibited.
- Set SPT0 only during operation as master.[Note 1]
- Note that when WTIM0 = 0, if SPT0 is set during the wait period after 8-clock output, a stop condition is generated during the high-level period of the 9th clock following wait release.
  If it is necessary to output a 9th clock, change the setting of WTIM0 from 0 to 1 during the wait period following 8-clock output, and set SPT0 during the wait period following the 9th clock output.

Clear conditions (SPT0 = 0)[Note 2]	Set conditions (SPT0 = 1)
• Cleared with instruction • Cleared upon defeat in arbitration • Cleared automatically after detection of stop condition • Cleared by $\overline{\text{RESET}}$ input	• Set with instruction

Notes 1. Set SPT0 only during operation as master. However, for master operation by the time a stop condition is detected for the first time following operation enable, SPT0 must be set once to generate a stop condition.

2. This flag's signals are made invalid by setting SMBE0 = 0.

Caution While SMB status register 0 (SMBS0) bit 3 (TRC0) = 1, when WREL0 is set at the 9th clock and wait is released, TRC0 is cleared and the SDA0 line is placed in high impedance.

Remark STD0 : SMB status register 0 (SMBS0) bit 1
ACKD0 : SMB status register 0 (SMBS0) bit 2
TRC0 : SMB status register 0 (SMBS0) bit 3
COI0 : SMB status register 0 (SMBS0) bit 4
EXC0 : SMB status register 0 (SMBS0) bit 5
MSTS0 : SMB status register 0 (SMBS0) bit 7

(2) SMB status register 0 (SMBS0)

This register indicates the SMB status.

SMBS0 is manipulated with a 1-bit or 8-bit memory manipulation instruction. SMBS0 is a read-only register.

$\overline{\text{RESET}}$ input clears SMBS0 to 00H.

Figure 15-3. Format of SMB Status Register 0 (1/3)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SMBS0	MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0	FF79H	00H	R

MSTS0	Master status
0	Slave status or communication wait status
1	Master transmission status

Clear conditions (MSTS0 = 0)	Set conditions (MSTS0 = 1)
• Cleared upon detection of stop condition • Cleared when ALD0 = 1 • Cleared when LREL0 = 1 • Cleared when SMBE0 changes from 1 to 0 • Cleared by $\overline{\text{RESET}}$ input	• Set during generation of start condition

ALD0	Arbitration defeat detection
0	No arbitration, or won in arbitration.
1	Defeated in arbitration. MSTS0 cleared.

Clear conditions (ALD0 = 0)	Set conditions (ALD0 = 1)
• Automatically cleared after reading SMBS0[Note] • Cleared when SMBE0 changes from 1 to 0 • Cleared by $\overline{\text{RESET}}$ input	• Set upon defeat in arbitration

EXC0	Extension code receive detection
0	Does not receive extension code.
1	Receives extension code.

Clear conditions (EXC0 = 0)	Set conditions (EXC0 = 1)
• Cleared upon detection of start condition • Cleared upon detection of stop condition • Cleared when LREL0 = 1 • Cleared when SMBE0 changes from 1 to 0 • Cleared by $\overline{\text{RESET}}$ input	• Set when high-order 4 bits of received address are 0000 or 1111 (set at rising edge of 8th clock)

Note The bit is also cleared when a bit manipulation instruction is executed for any of other bit of SMBS0.

Figure 15-3. Format of SMB Status Register 0 (2/3)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SMBS0	MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0	FF79H	00H	R

COI0	Matching address detection
0	Address does not match.
1	Address matches.

Clear conditions (COI0 = 0)	Set conditions (COI0 = 1)
• Cleared upon detection of start condition • Cleared upon detection of stop condition • Cleared when LREL0 = 1 • Cleared when SMBE0 changes from 1 to 0 • Cleared by $\overline{\text{RESET}}$ input	• Set when received address matches local address (SVA0) (set at rising edge of 8th clock)

TRC0	Receive/send status detection
0	Receive status (when not in send status). Sets SDA0 line to high impedance.
1	Send status. Sets so that SO latch value can be output to SDA0 line (valid from falling edge of 9th clock of 1st byte).

Clear conditions (TRC0 = 0)	Set conditions (TRC0 = 1)
• Cleared upon detection of stop condition • Cleared when LREL0 = 1 • Cleared SMBE0 changes from 1 to 0 • Cleared when WREL0 = 1 • Cleared when ALD0 changes from 0 to 1 • Cleared by $\overline{\text{RESET}}$ input In case of master: • When "1" is output to 1st byte LSB (transmission direction specification bit) In case of slave: • Upon detection of start condition In case of non-participation in communication	In case of master: • Upon generation of start condition In case of slave: • When "1" is input to 1st byte LSB (transmission direction specification bit)

ACKD0	Acknowledge output
0	Does not detect acknowledge.
1	Detects acknowledge.

Clear conditions (ACKD0 = 0)	Set conditions (ACKD0 = 1)
• Cleared upon detection of stop condition • Cleared at rising edge of 1st clock of following byte • Cleared when LREL0 = 1 • Cleared when SMBE0 changes from 1 to 0 • Cleared by $\overline{\text{RESET}}$ input	• Set when SDA0 line is low level at rising edge of 9th clock of SCL0

Figure 15-3. Format of SMB Status Register 0 (3/3)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
SMBS0	MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0	FF79H	00H	R

STD0	Start condition detection
0	Does not detect start condition.
1	Detects start condition. Indicates that address transmission is in progress.

Clear conditions (STD0 = 0)	Set conditions (STD0 = 1)
• Cleared upon detection of stop condition • Cleared at rising edge of 1st clock of byte following address transmission • Cleared when LREL0 = 1 • Cleared when SMBE0 changes from 1 to 0 • Cleared by $\overline{\text{RESET}}$ input	• Set upon detection of start condition

SPD0	Stop condition detection
0	Does not detect stop condition.
1	Detects stop condition. Transmission by master is completed and bus is released.

Clear conditions (SPD0 = 0)	Set conditions (SPD0 = 1)
• Cleared at rising edge of 1st clock of address transfer byte following detection of start condition after this bit has been set • Cleared when SMBE0 changes from 1 to 0 • Cleared by $\overline{\text{RESET}}$ input	• Set upon detection of stop condition

Remark LREL0 : SMB control register 0 (SMBC0) bit 6
SMBE0 : SMB control register 0 (SMBC0) bit 7

(3) SMB clock selection register 0 (SMBCL0)

This register sets the SMB transmission clock.

SMBCL0 is manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears SMBCL0 to 00H. Table 15-2 shows the SMB communication clock.

Figure 15-4. Format of SMB Clock Selection Register 0 (1/2)

Symbol	7	6	<5>	<4>	3	2	1	0	Address	After reset	R/W
SMBCL0	0	0	CLD0	DAD0	SMC0	DFC0	CL01	CL00	FF7AH	00H	R/W[Note 1]

CLD0	SCL0 line level detection (valid only when SMBE0 = 1)	
0	Detects that SCL0 line is low level.	
1	Detects that SCL0 line is high level.	
Clear conditions (CLD0 = 0)		Set conditions (CLD0 = 1)
• Cleared when SCL0 line is low level • Cleared when SMBE0 = 0 • Cleared by $\overline{\text{RESET}}$ input		• Set when SCL0 line is high level

DAD0	SDA0 line level detection (valid only when SMBE0 = 1)	
0	Detects that SDA0 line is low level.	
1	Detects that SDA0 line is high level.	
Clear conditions (DAD0 = 0)		Set conditions (DAD0 = 1)
• Cleared when SDA0 line is low level • Cleared when SMBE0 = 0 • Cleared by $\overline{\text{RESET}}$ input		• Set when SDA0 line is high level

SMC0	Operating mode switching	
0	IIC standard mode or SMB mode operation	
1	IIC high-speed mode	
Clear conditions (SMC0 = 0)		Set conditions (SMC0 = 1)
• Cleared with instruction • Cleared by $\overline{\text{RESET}}$ input		• Set with instruction

DFC0	Digital filter operation control[Note 2]
0	Digital filter OFF
1	Digital filter ON

Figure 15-4. Format of SMB Clock Selection Register 0 (2/2)

Symbol	7	6	<5>	<4>	3	2	1	0	Address	After reset	R/W
SMBCL0	0	0	CLD0	DAD0	SMC0	DFC0	CL01	CL00	FF7AH	00H	R/W Note 1

Communication clock			
CL01	CL00	SMB/standard mode (SMC0 = 0)	High-speed mode (SMC0 = 1)
0	0	f_X/44	f_X/24
0	1	f_X/86	
1	0	f_X/172	f_X/48
1	1	Setting prohibited	

Notes 1. Bits 4 and 5 are read-only.

2. The digital filter can be used in high-speed mode. When used in high-speed mode, the digital filter provides a slower response.

Caution Bits 6 and 7 must all be set to 0.

Remark f_X: Main system clock oscillation frequency

Table 15-2. SMB0 Communication Clock

SMC0	CL01	CL00	Communication Clock	Digital Filter Input Delay
			f_X = 5.0 MHz	
0	0	0	113.6 kHz Note	250 ns
0	0	1	58.1 kHz	250 ns
0	1	0	29.06 kHz	500 ns
1	0	0	208.3 kHz Note	250 ns
1	0	1	208.3 kHz Note	250 ns
1	1	0	104.1 kHz Note	500 ns
Other than above			Setting prohibited	

Note Since the SMB0 standards specify a range of 10 to 100 kHz, this communication clock falls outside the specifications.

(4) SMB mode register 0 (SMBM0)

SMBM0 is used to specify SCL0 level control and interrupt control.

SMBM0 is manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets SMBM0 to 20H.

Figure 15-5. Format of SMB Mode Register 0 (1/2)

Symbol	7	6	<5>	<4>	<3>	<2>	1	0	Address	After reset	R/W
SMBM0	0	0	SCLCTL0	AWTIM0	STIE0	TOEN0	TOCL01	TOCL00	FF7CH	20H	R/W

SCLCTL0	SCL level control[Note 1]
0	SCL0 is held low. When SCL0 is high, SCL0 is held low after waiting until SCL0 is made low.
1	Normal operation

STIE0	Start condition interrupt enable
0	Start condition interrupt generation is disabled.
1	Normal operation

AWTIM0	Wait and interrupt control when an address match is found[Notes 2, 3]
0	At the slave, an interrupt request is generated on the falling edge of the 9th clock period when an address match (COI0 = 1) is found during address data reception. The clock is pulled low to cause the master to wait.
1	At the slave, an interrupt request is generated on the falling edge of the 8th clock period when an address match (COI0 = 1) is found during address data reception. The clock is pulled low to cause the master to wait.

Figure 15-5. Format of SMB Mode Register 0 (2/2)

Symbol	7	6	<5>	<4>	<3>	<2>	1	0	Address	After reset	R/W
SMBM0	0	0	SCLCTL0	AWTIM0	STIE0	TOEN0	TOCL01	TOCL00	FF7CH	20H	R/W

TOEN0	Time-out count enable bit[Note 4]
0	The time-out count is cleared to 0, then count operation is disabled.
1	Time-out count operation is enabled.

TOCL01	TOCL00	Time-out clock selection bits
0	0	$f_X/2^6$ (78.1 kHz)
0	1	$f_X/2^7$ (39.1 kHz)
1	0	$f_X/2^8$ (19.5 kHz)
1	1	f_{XT} (32.768 kHz)

Notes 1. If SCL0 is made low with SCLCTL0, wait state cannot be released with WREL0.

2. When an extension code is received (EXC0 = 1), wait state is forcibly set in the 8th clock period.

3. During address transfer, the master waits in the 9th clock period.

4. An interrupt (INTSMBOV0) is generated when the time-out counter overflows. The hardware does not reset SMB operation. Ensure that SMB operation is reset by software after INTSMBOV0 generation.

Caution Bits 6 and 7 must all be set to 0.

Remarks 1. f_X : Main system clock oscillation frequency

2. f_{XT} : Subsystem clock oscillation frequency

(5) SMB input level setting register 0 (SMBVI0)

SMBVI0 is manipulated with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears SMBVI0 to 00H.

Figure 15-6. Format of SMB Input Level Setting Register 0

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
SMBVI0	0	TOS02	TOS01	TOS00	SVIN0	0	LVL01	LVL00	FF7DH	00H	R/W

TOS02	TOS01	TOS00	Time-out time selection bits			
			$f_{TO} = f_X/2^6$	$f_{TO} = f_X/2^7$	$f_{TO} = f_X/2^8$	$f_{TO} = f_{XT}$
0	0	0	$1{,}024/f_{TO}$ (13.1 ms)	$1{,}024/f_{TO}$ (26.2 ms)	$1{,}024/f_{TO}$ (52.4 ms)	$1{,}024/f_{TO}$ (31.3 ms)
0	0	1	$896/f_{TO}$ (11.5 ms)	$896/f_{TO}$ (22.9 ms)	$896/f_{TO}$ (45.9 ms)	$896/f_{TO}$ (27.3 ms)
0	1	0	$768/f_{TO}$ (9.8 ms)	$768/f_{TO}$ (19.7 ms)	$768/f_{TO}$ (39.3 ms)	$768/f_{TO}$ (23.4 ms)
0	1	1	$640/f_{TO}$ (8.2 ms)	$640/f_{TO}$ (16.4 ms)	$640/f_{TO}$ (32.8 ms)	$640/f_{TO}$ (19.5 ms)
1	0	0	$512/f_{TO}$ (6.6 ms)	$512/f_{TO}$ (13.1 ms)	$512/f_{TO}$ (26.2 ms)	$512/f_{TO}$ (15.6 ms)
1	0	1	$384/f_{TO}$ (4.9 ms)	$384/f_{TO}$ (9.8 ms)	$384/f_{TO}$ (19.7 ms)	$384/f_{TO}$ (11.7 ms)
1	1	0	$256/f_{TO}$ (3.3 ms)	$256/f_{TO}$ (6.6 ms)	$256/f_{TO}$ (13.1 ms)	$256/f_{TO}$ (7.8 ms)
1	1	1	$128/f_{TO}$ (1.6 ms)	$128/f_{TO}$ (3.2 ms)	$128/f_{TO}$ (6.6 ms)	$128/f_{TO}$ (3.9 ms)

SVIN0	Input level selection bit
0	Same input level as the ordinary hysteresis
1	The voltage set with LVL01 and LVL00 is used as the SCL0 and SDA0 input level threshold.

LVL01	LVL00	Input level selection bits[Note]
0	0	The input level is $0.1875 \times V_{DD}$.
0	1	The input level is $0.25 \times V_{DD}$.
1	0	The input level is $0.375 \times V_{DD}$.
1	1	The input level is $0.5 \times V_{DD}$.

Note Set an input level from 0.75 to 1.25 V.

Caution Bits 2 and 7 must all be set to 0.

Remarks 1. f_X : Main system clock oscillation frequency

2. f_{XT} : Subsystem clock oscillation frequency

3. f_{TO} : Clock selected using bits 0 and 1 (TOCL00, TOCL01) of SMB mode register 0 (SMBM0)

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

(6) SMB shift register 0 (SMB0)

This register is used to perform serial send/receive (shift operation) in synchronization with the serial clock. Read/write operations can be performed in 8-bit units, but do not write data to the SMB0 during transmission.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
SMB0									FF7EH	00H	R/W

(7) SMB slave address register 0 (SMBSVA0)

This register stores the SMB slave address.

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

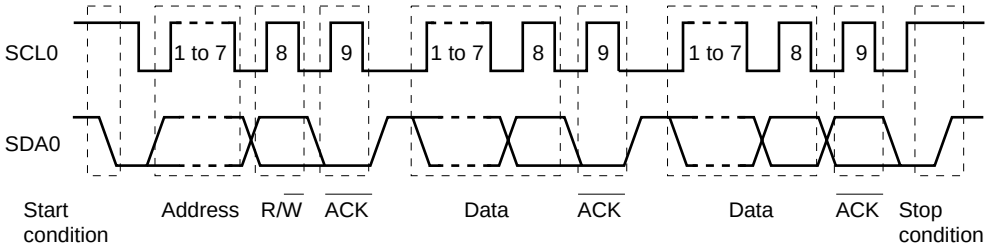
Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
SMBSVA0								0	FF7BH	00H	R/W

15.4 SMB0 Definition and Control Methods

The SMB0 serial data transmission format and the meaning of the signals used are described below.

The transmission timings of the start condition, data, and stop condition output to the serial data bus of the SMB0 are shown in Figure 15-7.

Figure 15-7. SMB0 Serial Data Transmission Timing



The start condition, slave address, and stop condition are output by the master.

SDA0 of only the start condition and stop condition can be changed when SCL0 is high.

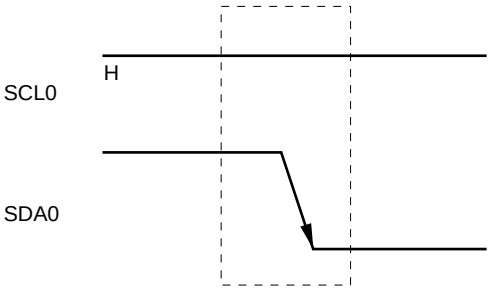
The acknowledge signal ($\overline{ACK}$) can be output by either the master or slave (The slave output $\overline{ACK}$ when an address is transferred. The receiver of the data outputs $\overline{ACK}$ when 8-bit data is transferred).

The master continuously outputs the serial clock (SCL0). However, it is possible to prolong the low-level period of the SCL0 and insert a wait in the case of the slave.

15.4.1 Start condition

A start condition is generated when the SDA0 pin changes from high level to low level while the SCL0 pin is high level (serial clock is not output). The start condition of the SCL0 and SDA0 pins is output at the start of serial transmission from the master to the slave. The slave incorporates hardware that detects the start condition.

Figure 15-8. Start Condition



The start condition is output when SMB control register 0 (SMBC0) bit 1 (STT0) is set to 1 in the stop condition detection status (STD0: SMB status register 0 (SMBS0) bit 1 = 1). Moreover, when the start condition is detected, SMBS0 bit 1 (STD0) is set to 1.

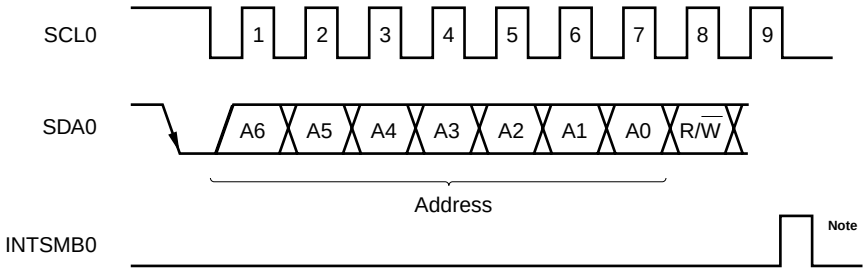
15.4.2 Address

The 7-bit data following the start condition is defined as an address.

An address consists of 7 bits of data output to select a particular slave among several slaves connected to the master through a bus line. Therefore, slaves on the bus line must each have a different address.

Slaves detect start conditions through hardware and check if the 7-bit data matches the value of the SMB slave address register 0 (SMBSVA0). If the 7-bit data and the SMBSVA0 value match, that slave is selected, and communication between the master and that slave is performed until the master issues a start condition or a stop condition.

Figure 15-9. Address



Note If other than a local address or extension code is received during slave operation, INTSMB0 is not issued.

Addresses are written and output to SMB shift register 0 (SMB0) as 8 bits consisting of the slave address and the transmission direction (see **15.4.3**). Moreover, received addresses are written to SMB0.

Slave addresses are allocated to the high-order 7 bits of SMB0.

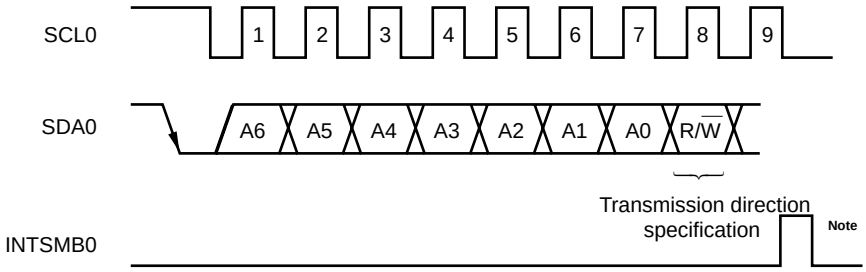
15.4.3 Specification of transmission direction

The master sends a 1-bit data following the 7-bit address to specify the transmission direction.

When this transmission direction bit is 0, the master sends data to the slave.

When this bit is 1, the slave sends data to the master.

Figure 15-10. Specification of Transmission Direction



Note If other than a local address or extension code is received during slave operation, INTSMB0 is not issued.

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

The acknowledge signal ($\overline{\text{ACK}}$) is used to confirm reception of serial data at the sending and receiving sides.

At the receiving side, an acknowledge signal is returned each time 8 bits of data are received. At the sending side, an acknowledge signal is normally received following transmission of 8 bits of data. However, when the master is receiving, no acknowledge signal is output after the final data has been received. The sending side detects whether an acknowledge signal is returned following transmission of 8 bits of data. If an acknowledge signal is returned, processing is continued assuming that the data was successfully received. If no acknowledge signal is returned by the slave, the master outputs a stop condition or a restart condition, and stops transmission. An acknowledge signal is not returned for the following two reasons.

<1> Reception was not performed normally.

<2> The final data was received.

If the receiving side sets the SDA0 line to low level at the 9th clock, the acknowledge signal becomes active (normal reception response).

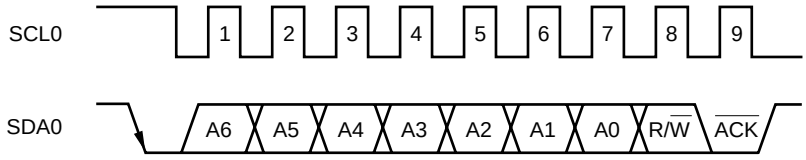
When SMB control register 0 (SMBC0) bit 2 (ACKE0) = 1, the acknowledge signal automatic generation enable state is entered.

SMB status register 0 (SMBS0) bit 3 (TRC0) is set by the 8th bit following the 7-bit address. However, when the TRC0 bit value is 0, receive status is selected, therefore set ACKE0 to 1.

During slave receive operation (TRC0 = 0), if the slave side receives several bytes and does not require subsequent data, ACKE0 can be set to 0 so that the master does not start the next transmission.

In the same way, if, during master receive operation (TRC0 = 0), subsequent data is not required and you want to output a restart condition or a stop condition, set ACKE0 to 0 so that no $\overline{\text{ACK}}$ signal is output. This must be done so that the data's MSB is not output to the SDA0 line during the slave send operation (transmission stop).

Figure 15-11. Acknowledge Signal



When a slave receives a local address, it automatically outputs an acknowledge signal in synchronization with the falling edge of the 8th clock of SCL0, regardless of the value of ACKE0. If a slave receives other than a local address, no acknowledge signal is output.

The acknowledge signal output method during data reception depends on the wait timing setting, as follows.

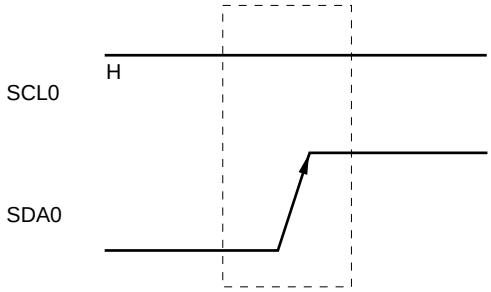
- 8-clock wait: Acknowledge signal is output when the value of ACKE0 becomes 1 before wait cancellation is performed.
- 9-clock wait: Acknowledge signal is automatically output in synchronization with the falling edge of the 8th clock of SCL0 by setting ACKE0 to 1 beforehand.

15.4.5 Stop condition

When the SDA0 pin changes from low level to high level while the SCL0 pin is at high level, a stop condition is generated.

A stop condition is the signal that is output when serial transfer from the master to a slave is completed. Slaves incorporate hardware for the detection of stop conditions.

Figure 15-12. Stop Condition



A stop condition is generated when bit 0 (SPT0) of SMB control register 0 (SMBC0) is set to 1. If, when a stop condition is detected, bit 0 (SPD0) of SMB status register 0 (SMBS0) and bit 4 (SPIE0) of SMBC0 are set to 1, INTSMB0 is generated.

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

A wait signal ($\overline{\text{WAIT}}$) indicates to the other party that the master or slave is getting ready (wait status) to send or receive data.

A wait status is notified by making the SCL0 pin low level. When the wait status of both the master and slave is canceled, the next transmission starts.

Figure 15-13. Wait Signal (1/2)

(1) When master = 9-clock wait, slave = 8-clock wait
(Master: send, Slave: receive, ACKE0 = 1)

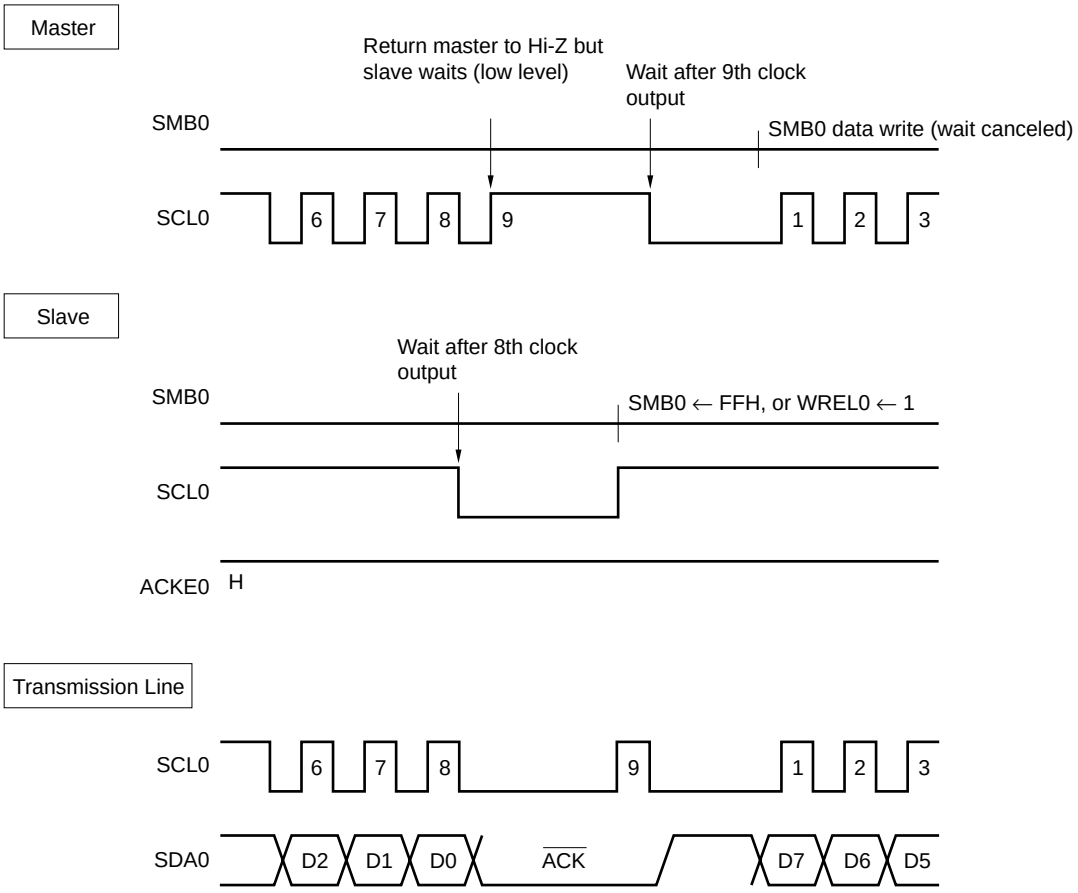
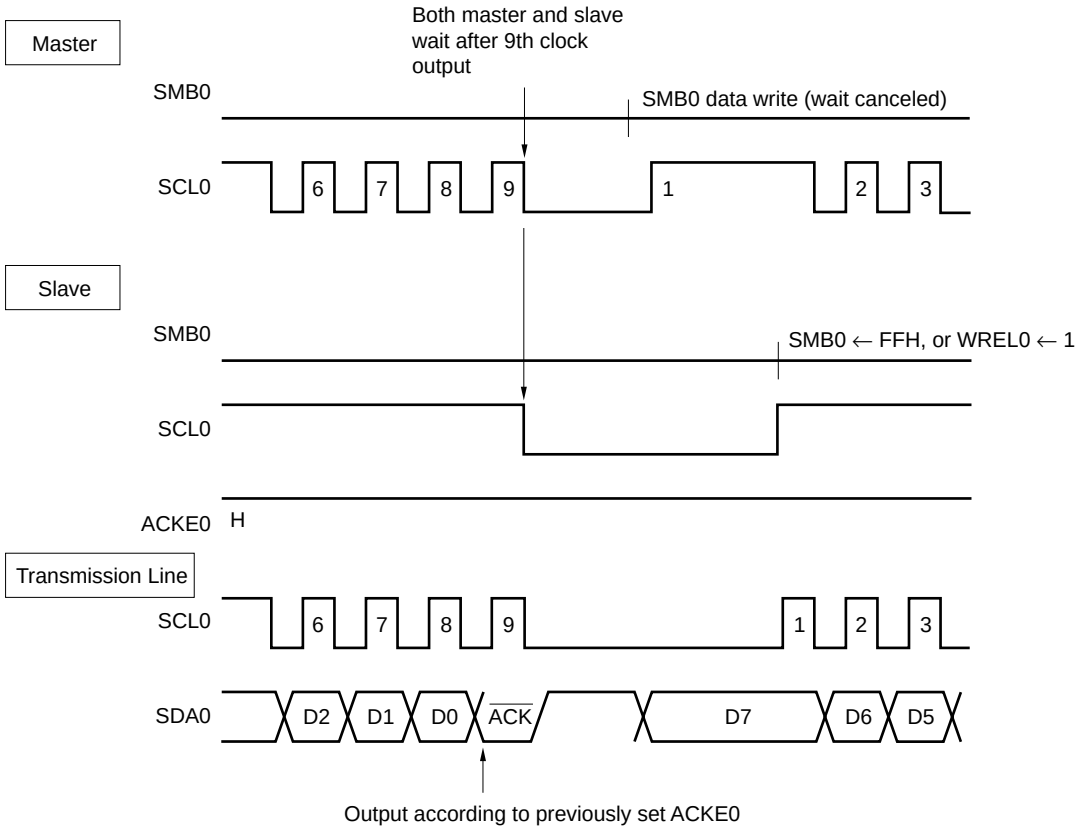


Figure 15-13. Wait Signal (2/2)

(2) Master, slave = 9-clock wait
(Master: send, Slave: receive, ACKE0 = 1)



Remark ACKE0 : SMB control register 0 (SMBC0) bit 2
WREL0 : SMB control register 0 (SMBC0) bit 5

Waits are automatically generated by setting bit 3 (WTIM0) of SMB control register 0 (SMBC0).
Normally, the receive side cancels the wait status when SMBC0 bit 5 (WREL0) = 1 or SMB shift register (SMB0) ← FFH write, and the send side cancels the wait status when data is written to SMB0.
In the case of the master, wait status can be canceled with the following methods.

- Setting SMBC0 bit 1 (STT0) to 1
- Setting SMBC0 bit 0 (SPT0) to 1

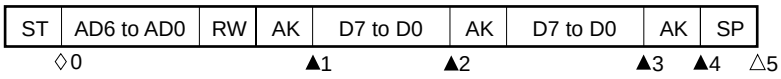
15.4.7 SMB0 interrupt (INTSMB0)

The following section shows the values of SMB status register 0 (SMBS0) using the INTSMB0 interrupt request generation timing and INTSMB0 interrupt timing.

(1) Master operation

(a) Start — Address — Data — Data — Stop (normal send/receive)

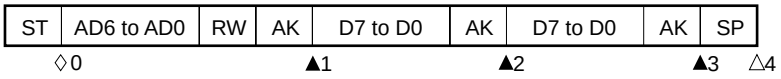
<1> When WTIM0 = 0



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 1000×000B
▲3: SMBS0 = 1000×000B
▲4: SMBS0 = 1000××00B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1

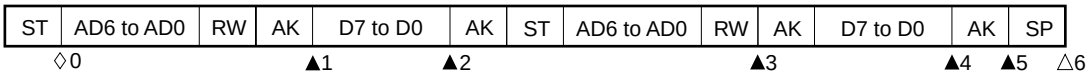


◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 1000×100B
▲3: SMBS0 = 1000××00B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

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

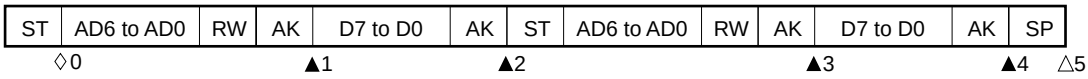
<1> When WTIM0 = 0



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 1000×000B
▲3: SMBS0 = 1000×110B
▲4: SMBS0 = 1000×000B
▲5: SMBS0 = 1000××00B
△6: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1

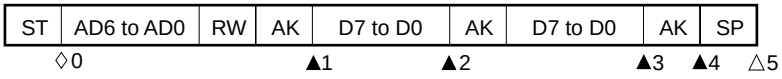


◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 1000××00B
▲3: SMBS0 = 1000×110B
▲4: SMBS0 = 1000××00B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

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

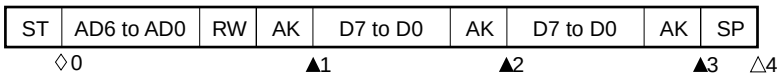
<1> When WTIM0 = 0



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1010××10B
▲2: SMBS0 = 1010×000B
▲3: SMBS0 = 1010×000B
▲4: SMBS0 = 1010××00B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1



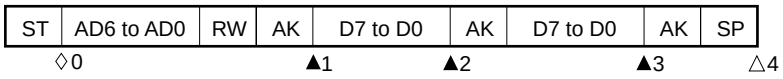
◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 0010×110B
▲2: SMBS0 = 0010×100B
▲3: SMBS0 = 0010××00B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(2) Slave operation (during slave address data reception (matching SVA0))

(a) Start — Address — Data — Data — Stop

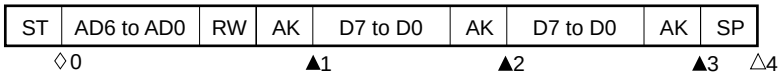
<1> When WTIM0 = 0



◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0001×110B
▲2: SMBS0 = 0001×000B
▲3: SMBS0 = 0001×000B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1

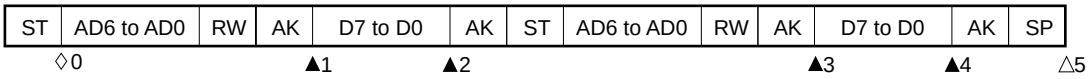


◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0001×110B
▲2: SMBS0 = 0001×100B
▲3: SMBS0 = 0001××00B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(b) Start — Address — Data — Start — Address — Data — Stop

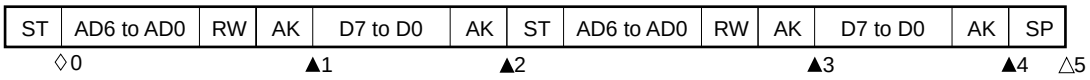
<1> When WTIM0 = 0 (matching SVA0 after restart)



◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0001×110B
▲2: SMBS0 = 0001×000B
▲3: SMBS0 = 0001×110B
▲4: SMBS0 = 0001×000B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1 (matching SVA0 after restart)

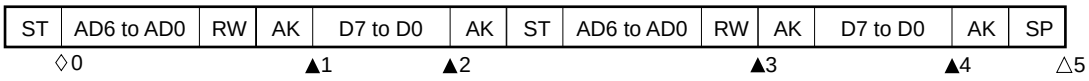


◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0001×110B
▲2: SMBS0 = 0001××00B
▲3: SMBS0 = 0001×110B
▲4: SMBS0 = 0001××00B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

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

<1> When WTIM0 = 0 (extension code reception after restart)



◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0001×110B
▲2: SMBS0 = 0001×000B
▲3: SMBS0 = 0010×010B
▲4: SMBS0 = 0010×000B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

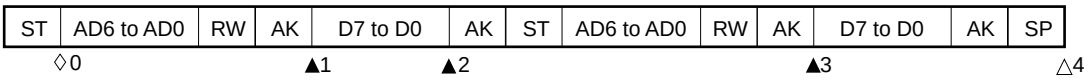
<2> When WTIM0 = 1 (extension code reception after restart)

◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0001×110B
▲2: SMBS0 = 0001××00B
▲3: SMBS0 = 0010×010B
▲4: SMBS0 = 00100110B
▲5: SMBS0 = 0010××00B
△6: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(d) Start — Address — Data — Start — Address — Data — Stop

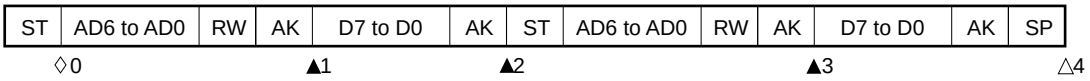
<1> When WTIM0 = 0 (unmatching address (except extension code) after restart)



◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0001×110B
▲2: SMBS0 = 0001×000B
▲3: SMBS0 = 0000××10B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1 (unmatching address (except extension code) after restart)



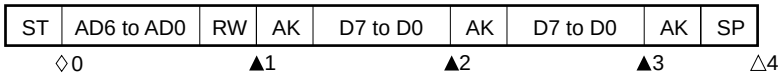
◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0001×110B
▲2: SMBS0 = 0001××00B
▲3: SMBS0 = 0000××10B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(3) Slave operation (during extension code reception)

(a) Start — Code — Data — Data — Stop

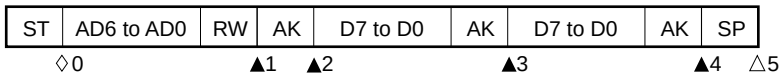
<1> When WTIM0 = 0



◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0010×010B
▲2: SMBS0 = 0010×000B
▲3: SMBS0 = 0010×000B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1

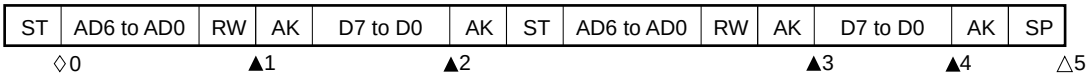


◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0010×010B
▲2: SMBS0 = 0010×110B
▲3: SMBS0 = 0010××00B
▲4: SMBS0 = 0010××00B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

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

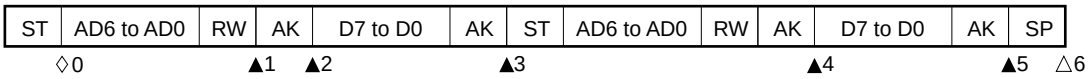
<1> When WTIM0 = 0 (matching SVA0 after restart)



◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0010×010B
▲2: SMBS0 = 0010×000B
▲3: SMBS0 = 0001×110B
▲4: SMBS0 = 0001×000B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1 (matching SVA0 after restart)

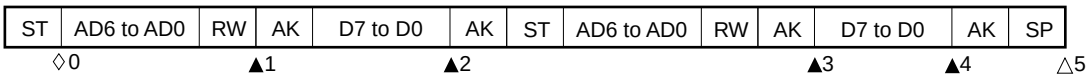


◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0010×010B
▲2: SMBS0 = 0010×110B
▲3: SMBS0 = 0010××00B
▲4: SMBS0 = 0001×110B
▲5: SMBS0 = 0001××00B
△6: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(c) Start — Code — Data — Start — Code — Data — Stop

<1> When WTIM0 = 0 (extension code reception after restart)



◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0010×010B
▲2: SMBS0 = 0010×000B
▲3: SMBS0 = 0010×010B
▲4: SMBS0 = 0010×000B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

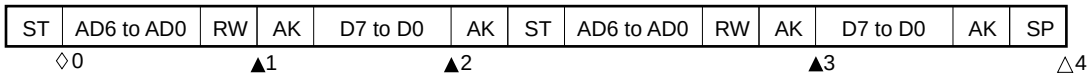
<2> When WTIM0 = 1 (extension code reception after restart)

◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0010×010B
▲2: SMBS0 = 0010×110B
▲3: SMBS0 = 0010××00B
▲4: SMBS0 = 0010×010B
▲5: SMBS0 = 0010×110B
▲6: SMBS0 = 0010××00B
△7: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

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

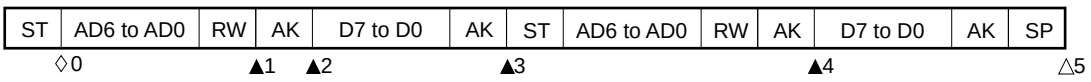
<1> When WTIM0 = 0 (unmatching address (except extension code) after restart)



◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0010×010B
▲2: SMBS0 = 0010×000B
▲3: SMBS0 = 00000×10B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1 (unmatching address (except extension code) after restart)

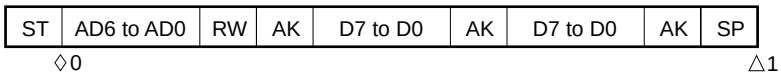


◊ 0: SMBS0 = 00000010B
▲1: SMBS0 = 0010×010B
▲2: SMBS0 = 0010×110B
▲3: SMBS0 = 0010××00B
▲4: SMBS0 = 00000×10B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(4) Non-participation in communication

(a) Start — Code — Data — Data — Stop



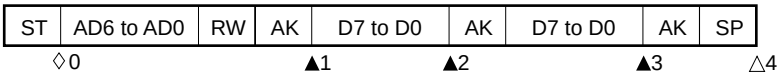
◊ 0: SMBS0 = 00000010B
△1: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
△ Generate only when SPIE0 = 1

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

(a) In case of arbitration defeat during slave address data transmission

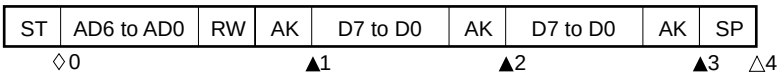
<1> When WTIM0 = 0



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 0101×110B (Example: Read ALD0 during interrupt processing)
▲2: SMBS0 = 0001×000B
▲3: SMBS0 = 0001×000B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1

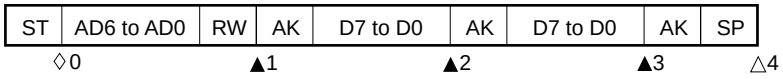


◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 0101×110B (Example: Read ALD0 during interrupt processing)
▲2: SMBS0 = 0001×100B
▲3: SMBS0 = 0001××00B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(b) In case of arbitration defeat during extension code transmission

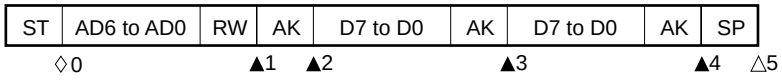
<1> When WTIM0 = 0



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 0110×010B (Example: Read ALD0 during interrupt processing)
▲2: SMBS0 = 0010×000B
▲3: SMBS0 = 0010×000B
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1

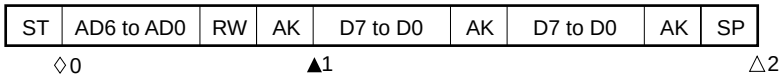


◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 0110×010B (Example: Read ALD0 during interrupt processing)
▲2: SMBS0 = 0010×110B
▲3: SMBS0 = 0010×100B
▲4: SMBS0 = 0010××00B
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(6) Arbitration defeat operation (non-participation after arbitration defeat)

(a) In case of arbitration defeat during slave address data transmission



◊ 0: SMBS0 = 10001010B

▲1: SMBS0 = 01000110B (Example: Read ALD0 during interrupt processing)

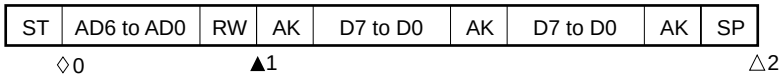
△2: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1

▲ Always generate

△ Generate only when SPIE0 = 1

(b) In case of arbitration defeat during extension code transmission



◊ 0: SMBS0 = 10001010B

▲1: SMBS0 = 0110×010B (Example: Read ALD0 during interrupt processing, LREL0 = 1 set by software)

△2: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1

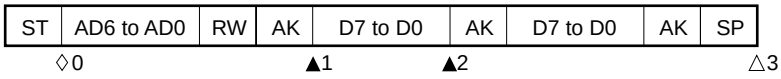
▲ Always generate

△ Generate only when SPIE0 = 1

× Don't care

(c) In case of arbitration defeat during data transmission

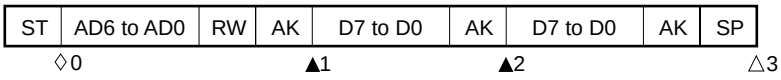
<1> When WTIM0 = 0



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 10001110B
▲2: SMBS0 = 01000000B (Example: Read ALD0 during interrupt processing)
△3: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1

<2> When WTIM0 = 1

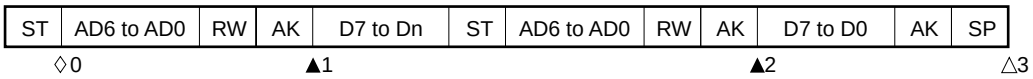


◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 10001110B
▲2: SMBS0 = 01000100B (Example: Read ALD0 during interrupt processing)
△3: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1

(d) In case of defeat by restart condition during data transmission

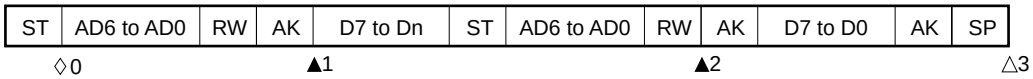
<1> Other than extension code (Example: Matching SVA0)



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 01000110B (Example: Read ALD0 during interrupt processing)
△3: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care
Dn = D6 to D0

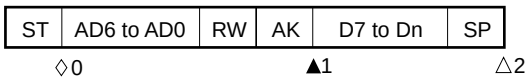
<2> Extension code



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 0110×010B (Example: Read ALD0 during interrupt processing,
SMBC0: LREL0 = 1 set by software)
△3: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care
Dn = D6 to D0

(e) In case of defeat by stop condition during data transmission



◊ 0: SMBS0 = 10001010B

▲ 1: SMBS0 = 1000×110B

△ 2: SMBS0 = 01000001B

Remark ◊ Generate only when STIE0 = 1

▲ Always generate

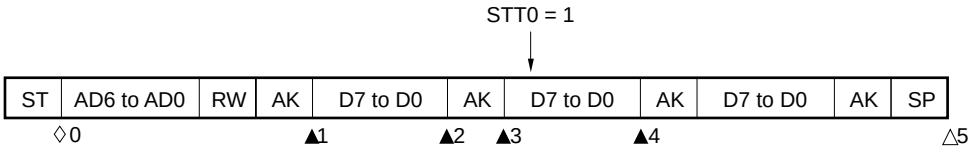
△ Generate only when SPIE0 = 1

× Don't care

Dn = D6 to D0

(f) In case of arbitration defeat by data low level while attempting to generate restart condition

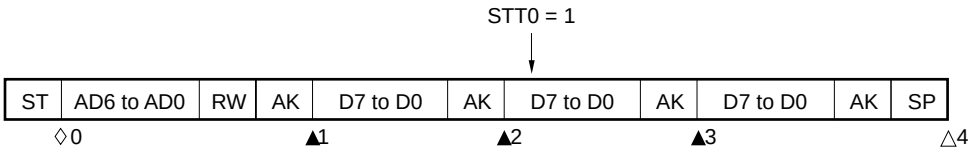
<1> When WTIM0 = 0



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 1000×000B
▲3: SMBS0 = 1000××00B
▲4: SMBS0 = 10000000B (Example: Read ALD0 during interrupt processing)
△5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1

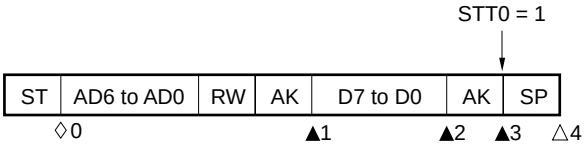


◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 1000××00B
▲3: SMBS0 = 01000100B (Example: Read ALD0 during interrupt processing)
△4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(g) In case of arbitration defeat by stop condition while attempting to generate restart condition

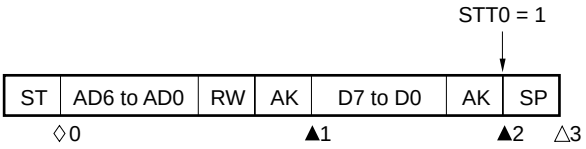
<1> When WTIM0 = 0



◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 1000×000B
▲3: SMBS0 = 1000××00B
△4: SMBS0 = 01000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1

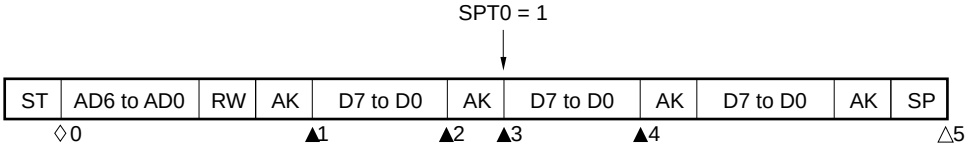


◊ 0: SMBS0 = 10001010B
▲1: SMBS0 = 1000×110B
▲2: SMBS0 = 1000××00B
△3: SMBS0 = 01000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

(h) In case of arbitration defeat by data low level while attempting to generate a stop condition

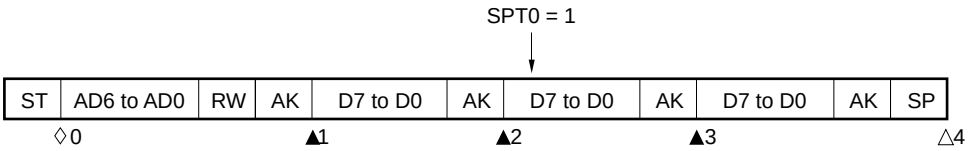
<1> When WTIM0 = 0



◊ 0: SMBS0 = 10001010B
▲ 1: SMBS0 = 1000×110B
▲ 2: SMBS0 = 1000×000B
▲ 3: SMBS0 = 1000××00B
▲ 4: SMBS0 = 01000000B (Example: Read ALD0 during interrupt processing)
△ 5: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

<2> When WTIM0 = 1



◊ 0: SMBS0 = 10001010B
▲ 1: SMBS0 = 1000×110B
▲ 2: SMBS0 = 1000××00B
▲ 3: SMBS0 = 01000000B (Example: Read ALD0 during interrupt processing)
△ 4: SMBS0 = 00000001B

Remark ◊ Generate only when STIE0 = 1
▲ Always generate
△ Generate only when SPIE0 = 1
× Don't care

15.4.8 Interrupt request (INTSMB0) generation timing and wait control

INTSMB0 generation and wait control can be done with the timings indicated in Table 15-3 by setting bit 3 (WTIM0) of the SMB control register 0 (SMBC0).

Table 15-3. INTSMB0 Generation Timing and Wait Control

WTIM0	During Slave Operation			During Master Operation		
	Address	Data Receive	Data Send	Address	Data Receive	Data Send
0	9 Notes 1, 2	8 Note 2	8 Note 2	9	8	8
1	9 Notes 1, 2	9 Note 2	9 Note 2	9	9	9

Notes 1. INTSMB0 and wait signals are generated by a slave at the falling edge of the 9th clock only when matching with the address set in the SMB slave address register (SMBSVA0) occurs.
Moreover, at this time, an $\overline{ACK}$ signal is output regardless of the setting of bit 2 (ACKE0) of SMBC0. A slave that receives an extension code generates INTSMB0 at the falling edge of the 8th clock.

2. If the address received does not match the address set in the SMB slave address register 0 (SMBSVA0), the slave does not generate INTSMB0 and wait signals.

Remark Figures listed in Table 15-3 above indicate the number of serial clocks. Interrupt requests and wait control are synchronized with the falling edge of the serial clock.

(1) During address transmission/reception

- During slave operation: Interrupt and wait timings are set regardless of the WTIM0 bit setting.
- During master operation: Interrupt and wait signals are generated at the falling edge of the 9th clock regardless of the WTIM0 bit setting.

(2) During data reception

- During master/slave operation: Interrupt and wait timings are set with the WTIM0 bit.

(3) During data transmission

- During master/slave operation: Interrupt and wait timings are set with the WTIM0 bit.

(4) Wait cancellation method

Waits can be canceled with one of the following four methods.

- Setting SMB control register 0 (SMBC0) bit 5 (WREL0) to 1
- Performing SMB shift register 0 (SMB0) write operation
- Setting a start condition (by setting SMBC0 bit 1 (STT0) to 1)
- Setting a stop condition (by setting SMBC0 bit 0 (SPT0) to 1)

When 8-clock wait is selected (WTIM0 = 0), the $\overline{ACK}$ output level must be determined before the wait status is released.

(5) Stop condition detection

An INTSMB0 signal is output when a stop condition is detected.

15.4.9 Matching address detection method

In SMB mode, a particular slave device can be selected by sending that slave address to the master.

The detection of matching addresses is performed automatically by hardware. If a local address has been set in SMB slave address register 0 (SMBSVA0), and the slave address sent from the master matches the address set in SMBSVA0, or if the extension code is received, an INTSMB0 interrupt request is generated.

15.4.10 Error detection

In SMB mode, because the status of the serial data bus (SDA0) during transmission is also input to SMB shift register 0 (SMB0), transmission errors can be detected by comparing the SMB0 data before transmission start and at transmission end. If two data do not match, a transmission error is considered to have occurred.

15.4.11 Extension code

(1) An extension code is considered to have been received when the high-order four bits of the receive address are 0000 or 1111, and in this case the extension code receive flag (EXC0) is set and interrupt request (INTSMB0) is generated at the falling edge of the 8th clock.
The local address stored in SMB slave address register 0 (SMBSVA0) is not affected.

(2) When 111110×× is set to SMBSVA0 and 111110××0 is transferred from the master during transfer of a 10-bit address, the following occurs. However, INTSMB0 is generated at the falling edge of the 8th clock.

- Matching high-order 4 bits: EXC0 = 1[Note]
- Matching 7-bit data: COI0 = 1[Note]

Note EXC0 : SMB status register 0 (SMBS0) bit 5
COI0 : SMB status register 0 (SMBS0) bit 4

(3) Because the processing after an interrupt request is generated differs depending on the data that follows the extension code, it is performed by software. For instance, if operation as a slave is not desired following the reception of an extension code, set LREL0 to 1, in which case the following communication standby status is entered.

Table 15-4. Extension Code Bit Definition

Slave Address	R/W Bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	×	CBUS address
0000 010	×	Address reserved for different bus format
1111 0××	×	10-bit slave address specification

Addresses reserved for system management bus are described below.

Slave Address	Description
0001 000	SMB host
0001 100	Response address for SMB alert
1010 001	Default address of SMB device
1001 0××	Free address

15.4.12 Arbitration

If several masters output a start condition simultaneously (when STT0 is set to 1 before STD0 is set to 1[Note]), master communication is performed while adjusting the clock until data differs. This operation is referred to as arbitration.

A master defeated in arbitration sets the arbitration defeat flag (ALD0) of SMB status register 0 (SMBS0), and sets the SCL0 and SDA0 lines to Hi-Z to release the bus.

Arbitration defeat is detected by software when ALD0 = 1 at the next interrupt request generation timing (8th or 9th clock, stop condition detection, etc.).

For the interrupt generation timing, see **15.4.7 SMB0 interrupt (INTSMB0)**.

Note STD0 : SMB status register 0 (SMBS0) bit 1
STT0 : SMB control register 0 (SMBC0) bit 1

Figure 15-14. Arbitration Timing Examples

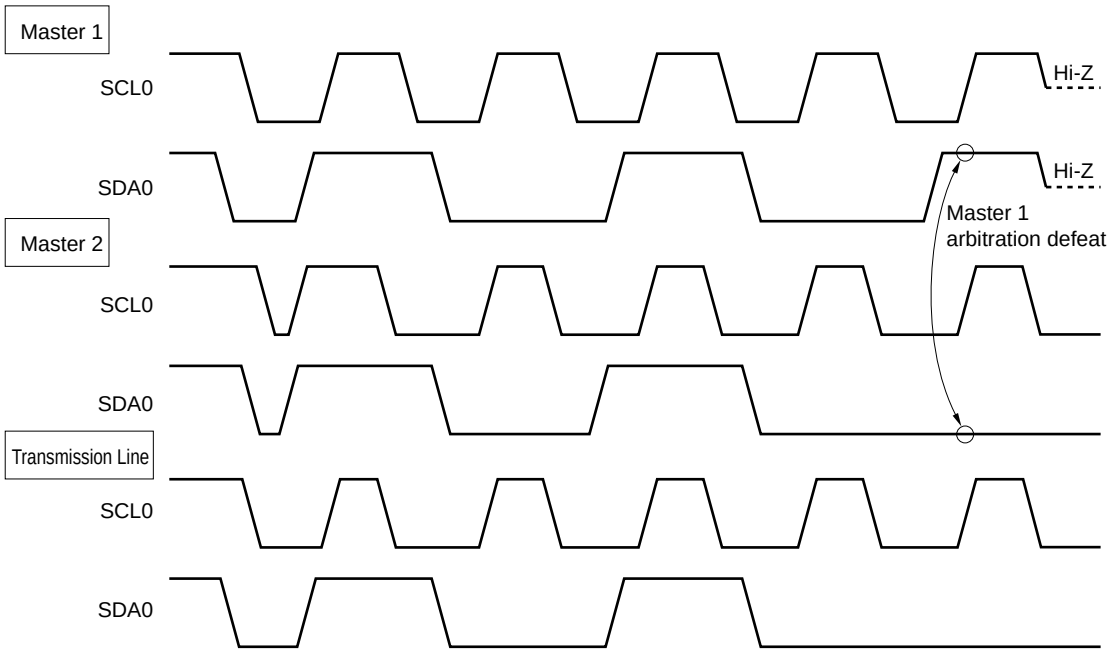


Table 15-5. Status at Arbitration and Interrupt Request Generation Timing

Status at Arbitration	Interrupt Request Generation Timing
Address transmission in progress	Falling edge of 8th or 9th clock following byte transmission[Note 1]
Read/write information following address transmission	
Extension code transmission in progress	
Read/write information following extension code transmission	
Data transmission in progress	
$\overline{\text{ACK}}$ transmission in progress following data transmission	
Data transmission in progress, restart condition detection	
Data transmission in progress, stop condition detection	During stop condition output (SPIE0 = 1)[Note 2]
Attempt to output restart condition was made, but data was low level	Falling edge of 8th or 9th clock following byte transfer[Note 1]
Attempt to output restart condition was made, but stop condition was detected	During stop condition output (SPIE0 = 1)[Note 2]
Attempt to output stop condition was made, but data was low level	Falling edge of 8th or 9th clock following byte transfer[Note 1]
Attempt to output restart condition was made, but SCL0 was low level	

Notes 1. If WTIM0 (bit 3 of SMB control register 0 (SMBC0) = 1, an interrupt request is generated at the falling edge of the 9th clock. During reception of an extension code slave address when WTIM0 = 0, an interrupt request is generated at the falling edge of the 8th clock.

2. If there is a possibility of arbitration occurring, set SPIE0 to 1 at master operation.

Remark SPIE0: SMB control register 0 (SMBC0) bit 4

15.4.13 Wakeup function

The SMB0 slave function generates an interrupt request (INTSMB0) when a local address and extension code are received.

When the address does not match, no unnecessary interrupt request is generated, allowing greater processing efficiency.

When a start condition is detected, the wakeup standby status is entered. Because even a master (when a start condition is output) may become a slave if defeated in arbitration, the wakeup standby status is entered while address transmission is performed.

However, when a stop condition is detected, interrupt request enable/disable is determined by setting bit 4 (SPIE0) of SMB control register 0 (SMBC0) regardless of the wakeup function.

15.4.14 Communication reservation

If, during non-participation on the bus, the next master communication is desired, a start condition can be made to be sent at bus release by performing communication reservation. Non-participation on the bus includes the following two statuses.

- When could become neither master nor slave during bus arbitration
- When extension code is received and does not operate as slave (released bus with SMB control register 0 (SMBC0) bit 6 (LREL0) = 1, without returning $\overline{\text{ACK}}$).

When bit 1 (STT0) of SMBC0 is set during non-participation on the bus, a start condition is generated automatically after the bus is released (following detection of stop condition), and the wait status is entered.

When bus release is detected (detection of stop condition), address transmission as master is started through SMB shift register 0 (SMB0) write operation. At this time, set SMBC0 bit 4 (SPIE0).

When STT0 is set, whether operation as a start condition or operation as communication reservation is selected depends on the bus status.

- If bus is released ..... Start condition generation
- If bus is not released (standby status) ... Communication reservation

The method to detect which operation is selected by STT0 is to set STT0, and after the wait time elapses, reconfirming the STT0 bit.

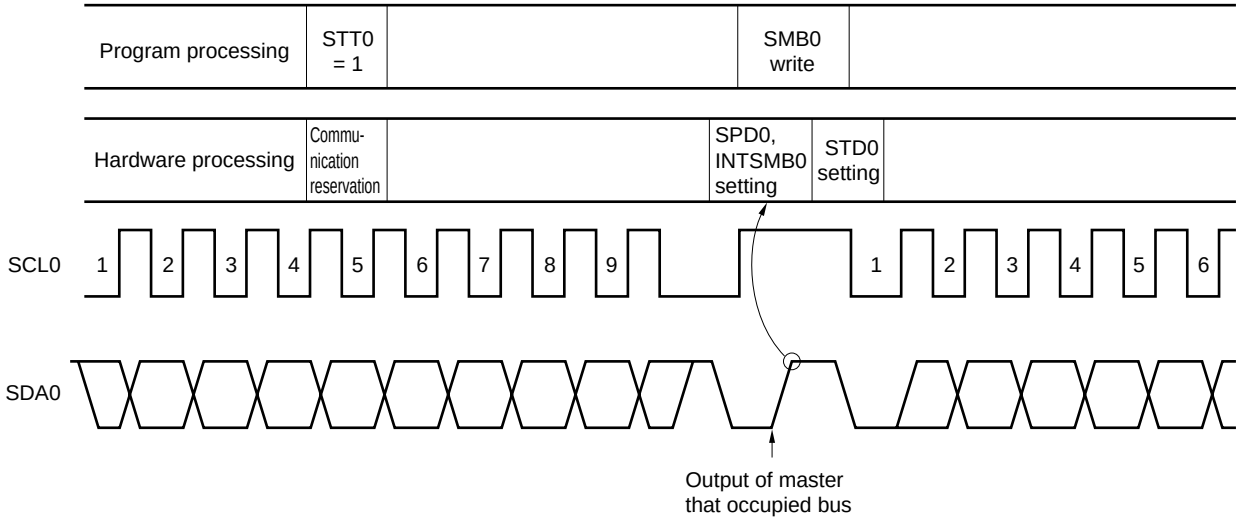
Secure the wait time by software as shown in Table 15-6. The wait time is set with bit 3 (SMC0) of SMB clock selection register 0 (SMBCL0).

Table 15-6. Wait Time

SMC0	Wait Time
0	46 clocks
1	16 clocks

The communication reservation timing is shown in Figure 15-15.

Figure 15-15. Communication Reservation Timing



SMB0 : SMB shift register 0
STT0 : SMB control register 0 (SMBC0) bit 1
STD0 : SMB status register 0 (SMBS0) bit 1
SPD0 : SMB status register 0 (SMBS0) bit 0

Communication reservations are received in the following timing. After bit 1 (STD0) of SMB status register 0 (SMBS0) becomes 1, communication reservation is done by setting bit 1 (STT0) of SMB control register 0 (SMBC0) to "1" before detection of a stop condition.

Figure 15-16. Communication Reservation Received Timing

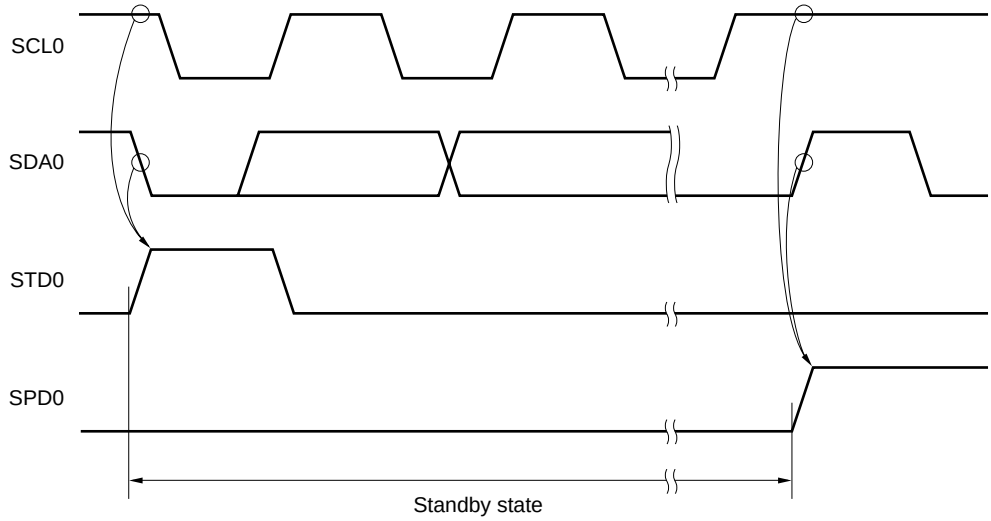
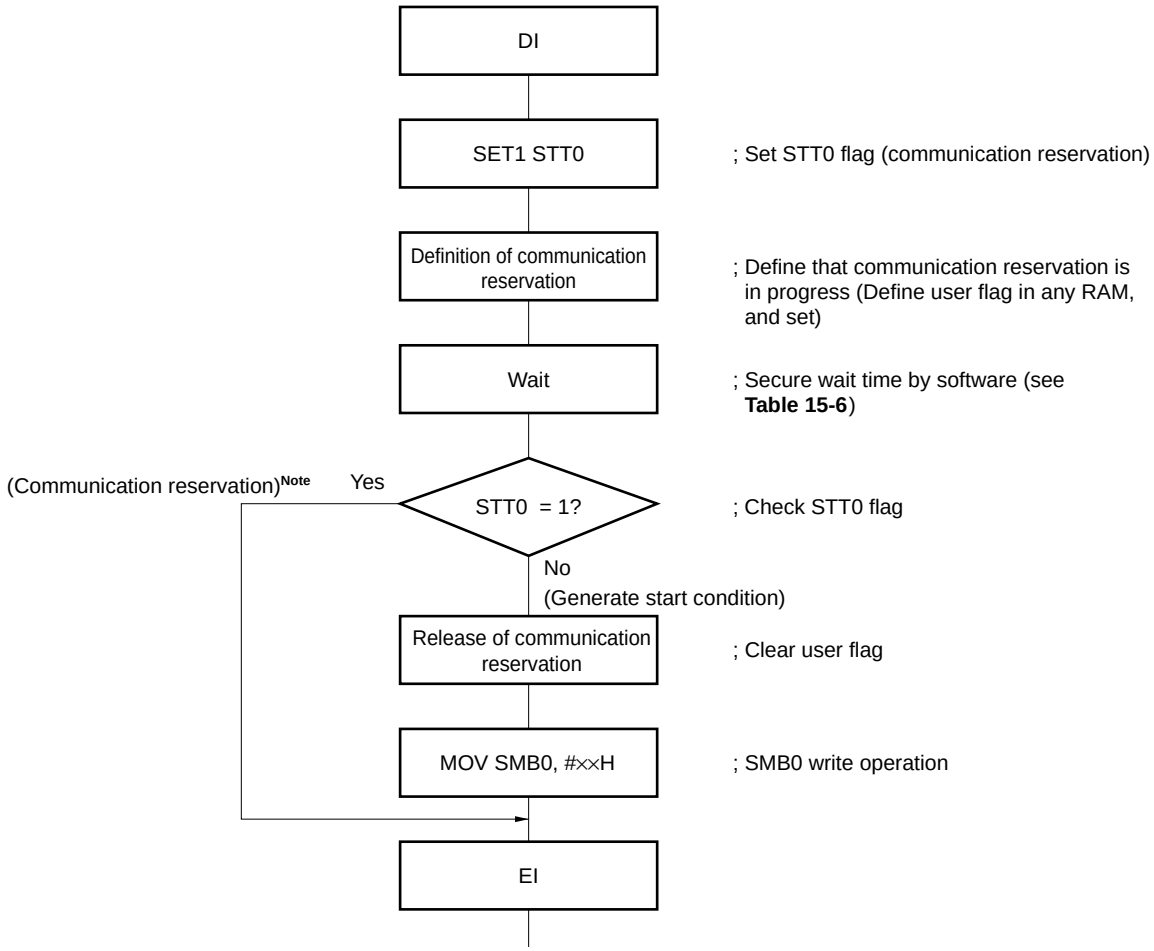


Figure 15-17 shows the communication reservation procedure.

Figure 15-17. Communication Reservation Procedure



Note During the communication reservation operation, execute writing to SMB shift register 0 (SMB0) with a stop condition interrupt.

15.4.15 Additional cautions

If, after reset, master communication is attempted from a status where no stop condition is detected (bus is not released), a stop condition must be generated and the bus released before performing master communication.

In the case of multiple masters, master communication cannot be performed while the bus is not released (stop condition not detected).

A stop condition is generated in the following sequence.

<1> Setting of SMB clock selection register 0 (SMBCL0)

<2> Setting of SMB control register 0 (SMBC0) bit 7 (SMBE0)

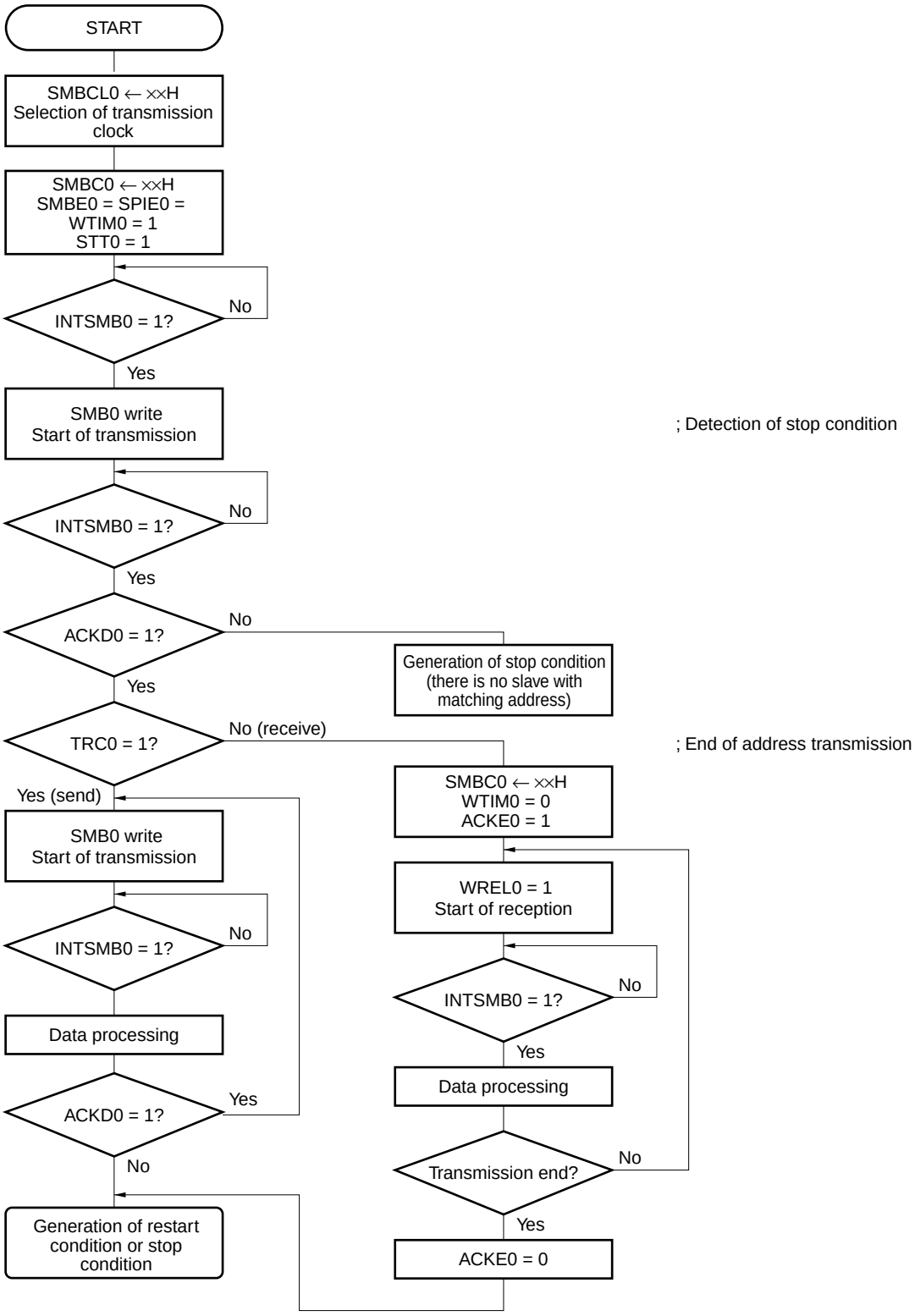
<3> Setting of SMBC0 bit 0

15.4.16 Communication operation

(1) Master operation

The master operation sequence is illustrated below.

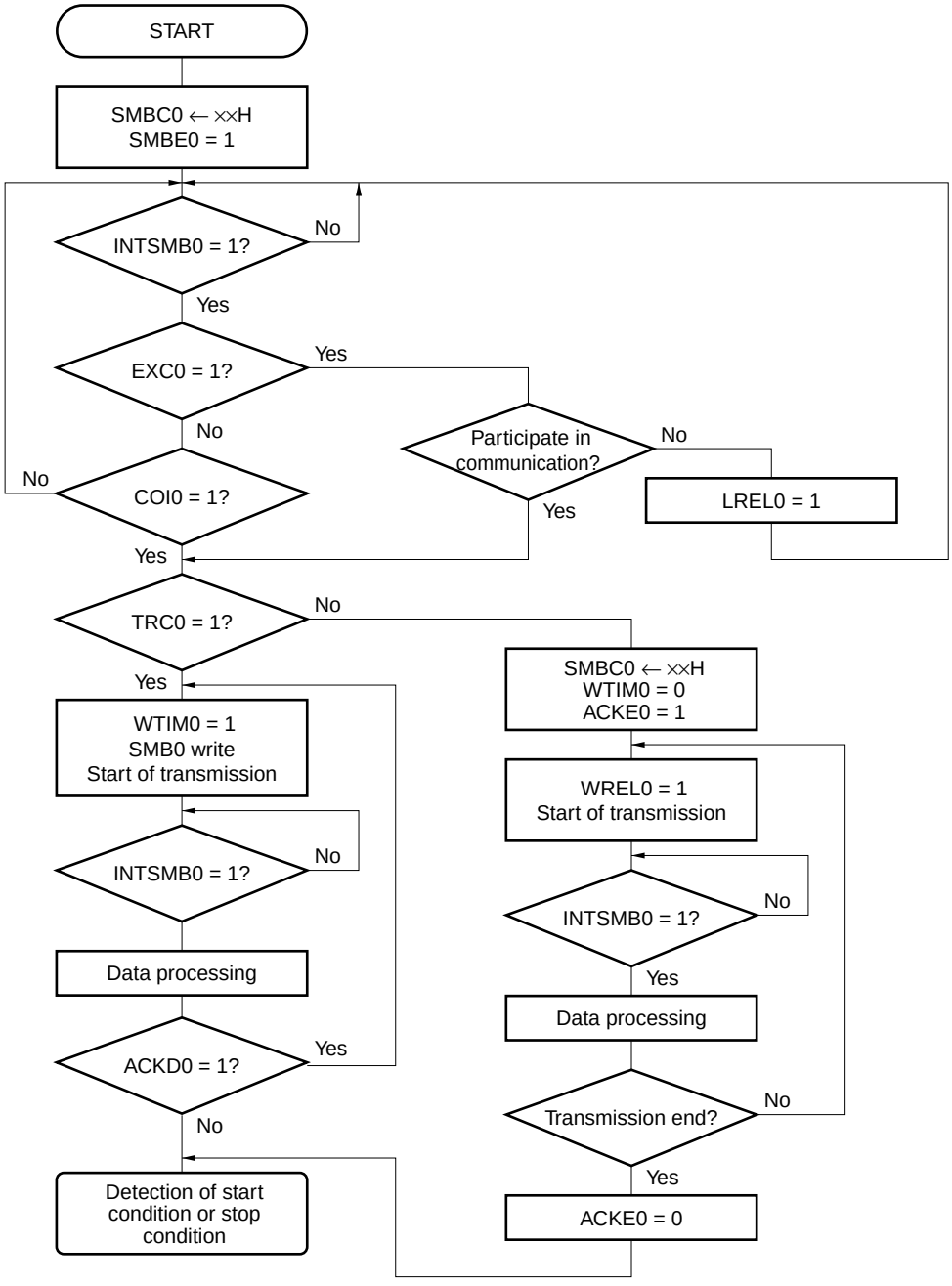
Figure 15-18. Master Operation Sequence



(2) Slave operation

The slave operation sequence is illustrated below.

Figure 15-19. Slave Operation Sequence



15.5 Timing Chart

In SMB mode, a master can select for communication a slave device from among many such devices by outputting an address to the serial bus.

After the slave address, the master sends the TRC0 bit (bit 3 of the SMB status register 0 (SMBS0)) indicating the data transmission direction and starts the serial communication with the slave.

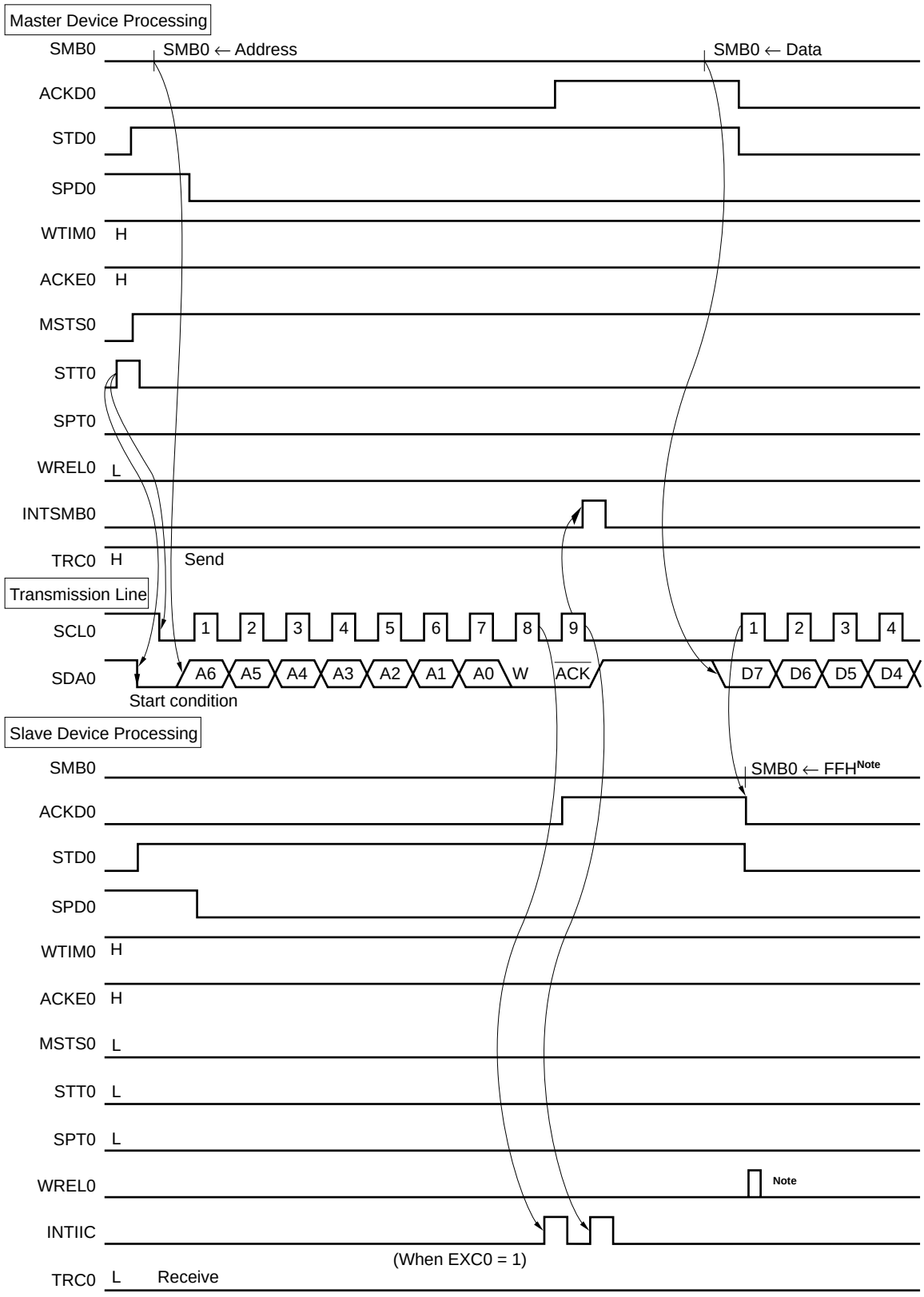
The timing charts for data transmission are shown in Figures 15-20 and 15-21.

The shift operation of SMB shift register 0 (SMB0) is performed in synchronization with the falling edge of the serial clock (SCL0), send data is transmitted to the SO0 latch and output MSB first from the SDA0 pin.

Data input to the SDA0 pin at the rising edge of SCL0 is read by SMB0.

Figure 15-20. Master → Slave Communication Example (When 9-Clock Wait is Selected for Both Master and Slave) (1/3)

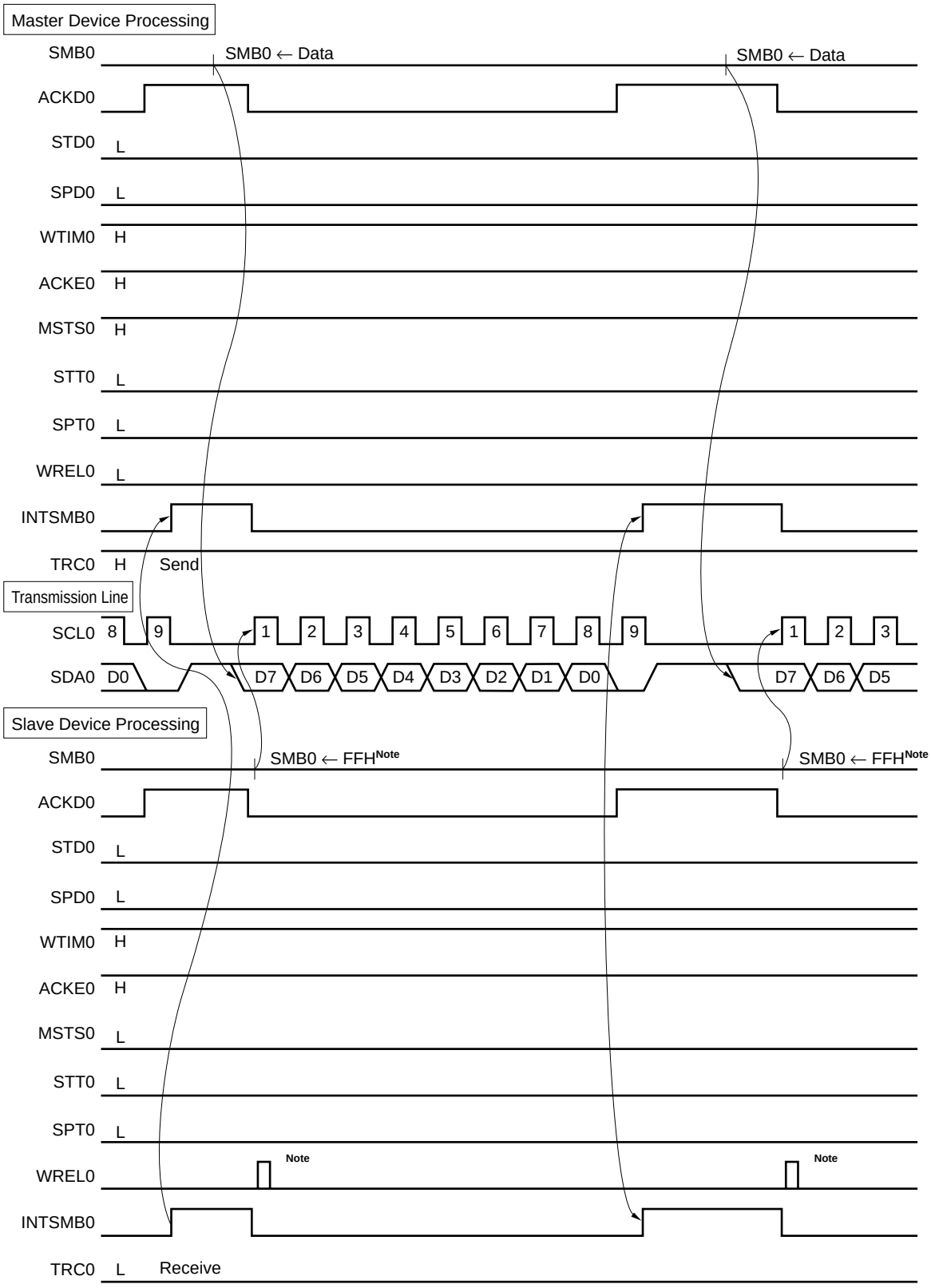
(1) Start condition — Address



Note Perform slave wait cancellation by either changing SMBC0 ← FFH, or setting WREL0.

Figure 15-20. Master → Slave Communication Example (When 9-Clock Wait is Selected for Both Master and Slave) (2/3)

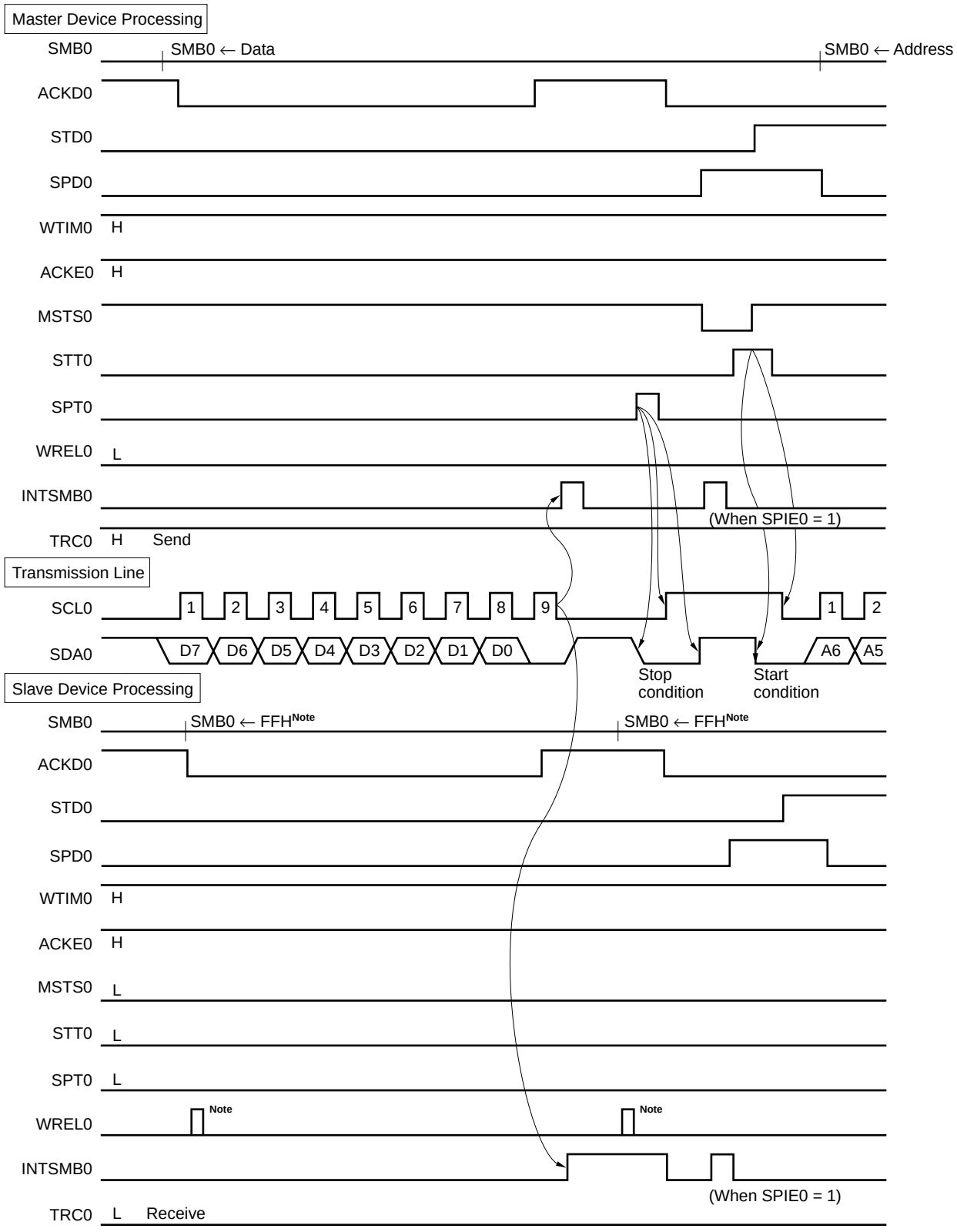
(2) Data



Note Perform slave wait cancellation by either changing SMBC0 ← FFH, or setting WREL0.

Figure 15-20. Master → Slave Communication Example (When 9-Clock Wait is Selected for Both Master and Slave) (3/3)

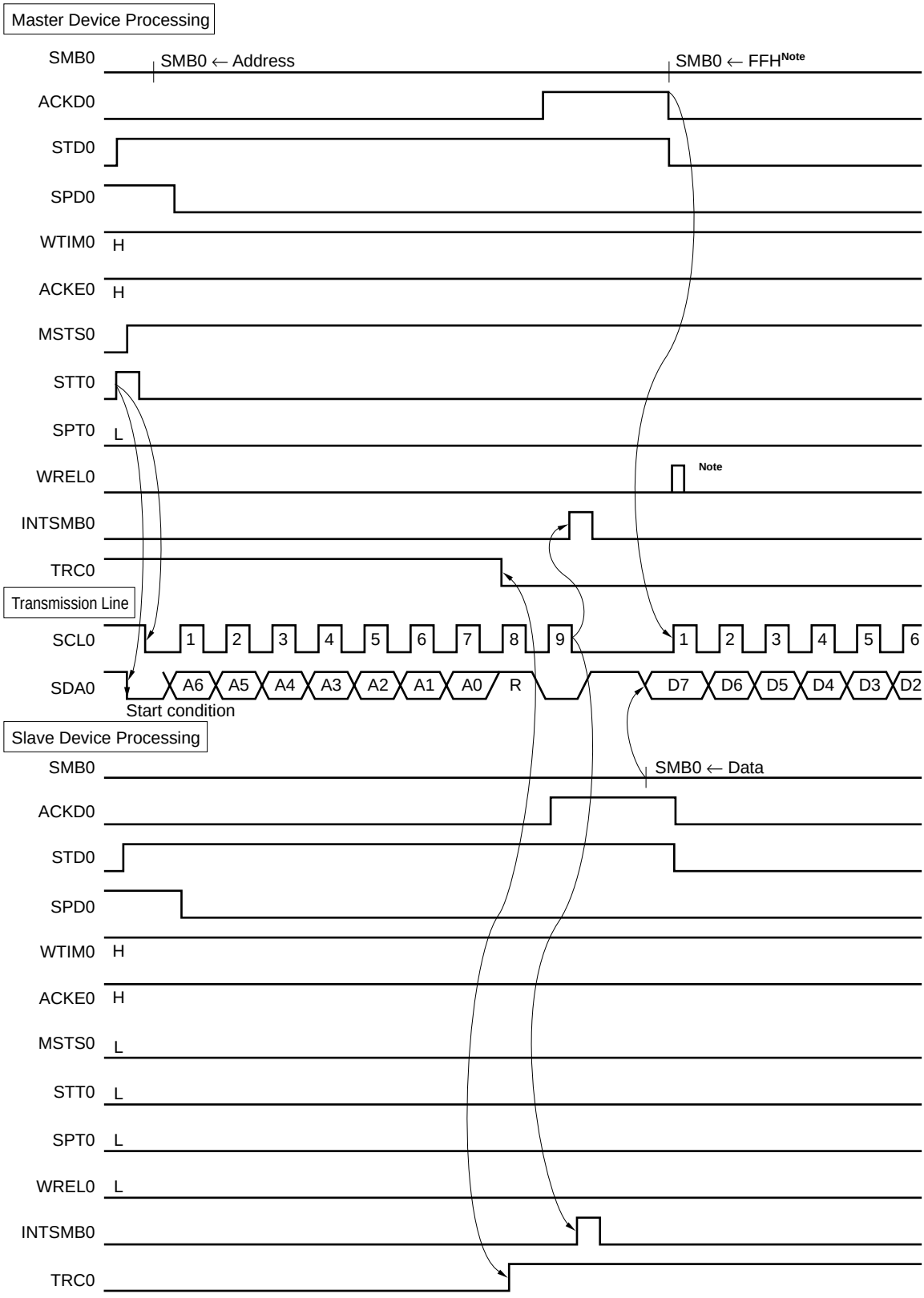
(3) Stop condition



Note Perform slave wait cancellation by either changing SMBC0 ← FFH, or setting WREL0.

Figure 15-21. Slave → Master Communication Example (When 9-Clock Wait is Selected for Both Master and Slave) (1/3)

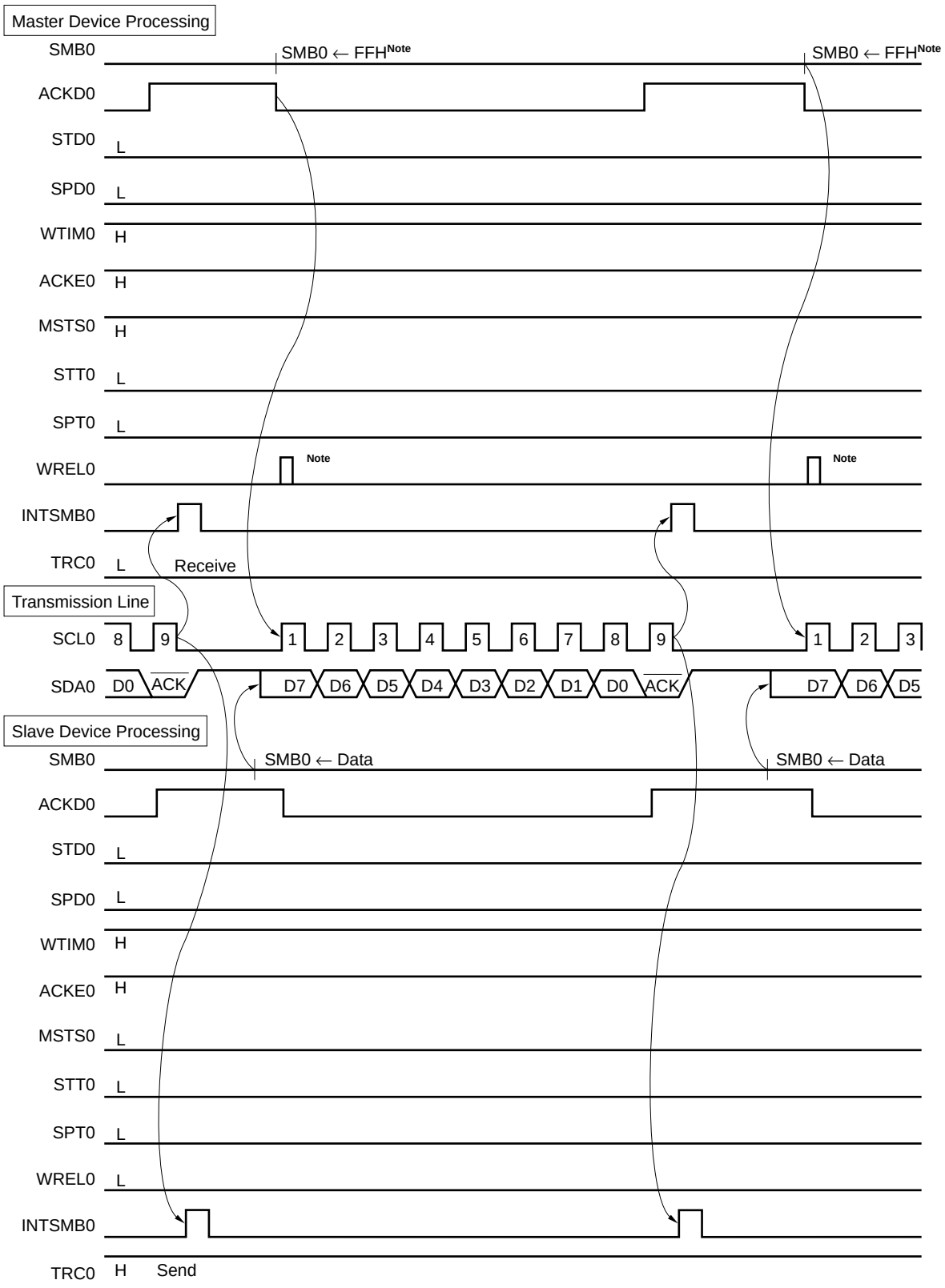
(1) Start condition — Address



Note Perform slave wait cancellation by either changing SMBC0 ← FFH, or setting WREL0.

Figure 15-21. Slave → Master Communication Example (When 9-Clock Wait is Selected for Both Master and Slave) (2/3)

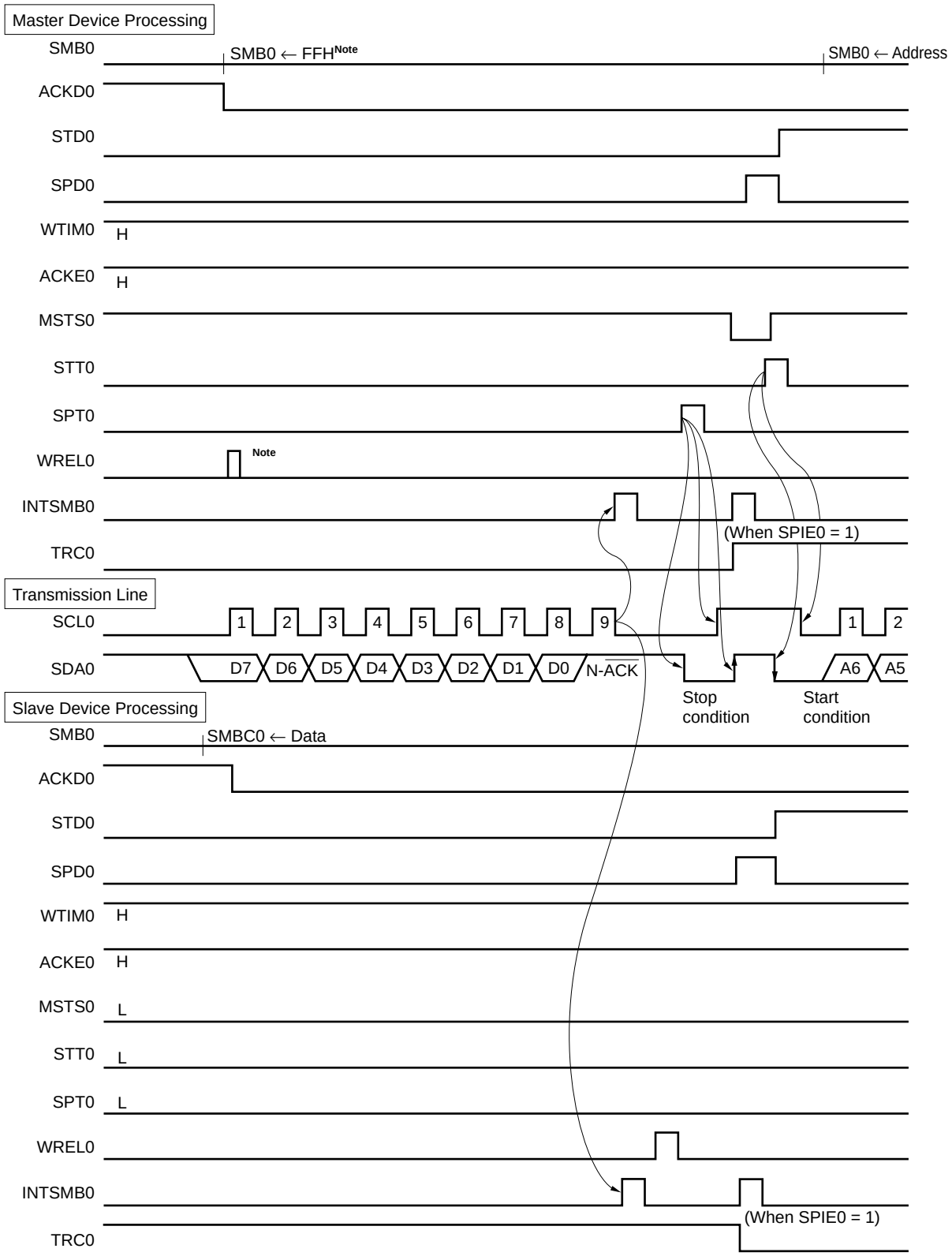
(2) Data



Note Perform slave wait cancellation by either changing SMBC0 ← FFH, or setting WREL0.

Figure 15-21. Slave → Master Communication Example (When 9-Clock Wait is Selected for Both Master and Slave) (3/3)

(3) Stop condition



Note Perform slave wait cancellation by either changing SMBC0 ← FFH, or setting WREL0.

CHAPTER 16 MULTIPLIER

16.1 Multiplier Function

The multiplier has the following function:

- Calculation of 8 bits × 8 bits = 16 bits

16.2 Multiplier Configuration

(1) 16-bit multiplication result storage register 0 (MUL0)

This register stores the 16-bit result of multiplication.

This register holds the result of multiplication after the 16 CPU clocks have elapsed.

MUL0 is set with a 16-bit memory manipulation instruction.

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

Caution MUL0 is designed to be manipulated with a 16-bit memory manipulation instruction. It can also be manipulated with 8-bit memory manipulation instructions, however. When an 8-bit memory manipulation instruction is used to manipulate MUL0, it must be accessed in direct addressing.

(2) Multiplication data registers A and B (MRA0 and MRB0)

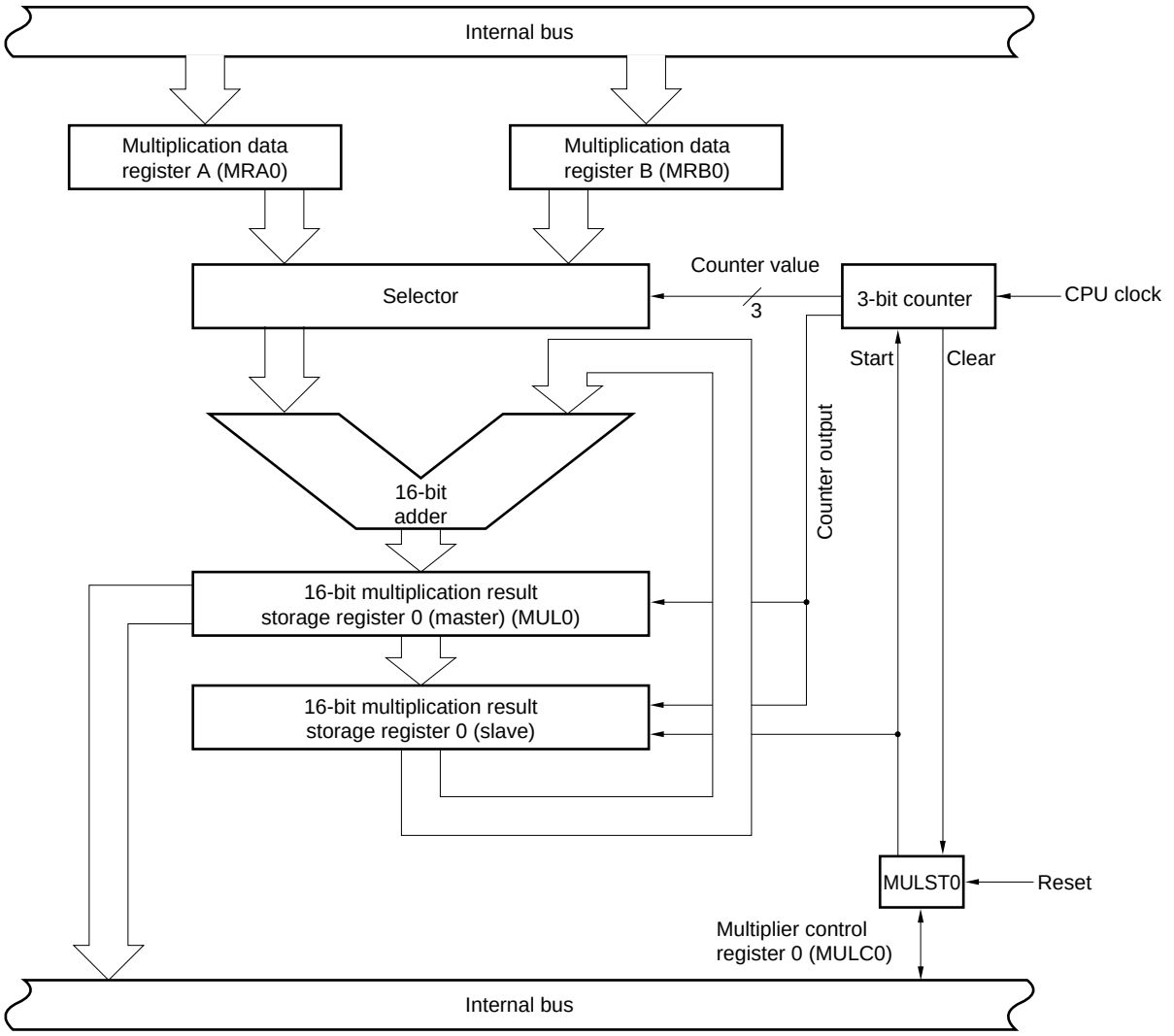
These are 8-bit multiplication data storage registers. The multiplier multiplies the values of MRA0 and MRB0.

MRA0 and MRB0 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input makes these registers undefined.

Figure 16-1 shows a block diagram of the multiplier.

Figure 16-1. Block Diagram of Multiplier



16.3 Multiplier Control Register

The multiplier is controlled by the following register:

- Multiplier control register 0 (MULC0)

MULC0 indicates the operating status of the multiplier, as well as controls the multiplier.
MULC0 is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears this register to 00H.

Figure 16-2. Format of Multiplier Control Register 0

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
MULC0	0	0	0	0	0	0	0	MULST0	FFD2H	00H	R/W

MULST0	Multiplier operation start control bit	Operating status of multiplier
0	Stops operation after resetting counter to 0.	Operation stops
1	Enables operation	Operation in progress

Caution Bits 1 to 7 must all be set to 0.

16.4 Multiplier Operation

The multiplier of the μPD789167, 789177, 789167Y, and 789177Y Subseries can execute calculation of 8 bits × 8 bits = 16 bits.

Figure 16-3 shows the operation timing of the multiplier where MRA0 is set to AAH and MRB0 is set to D3H.

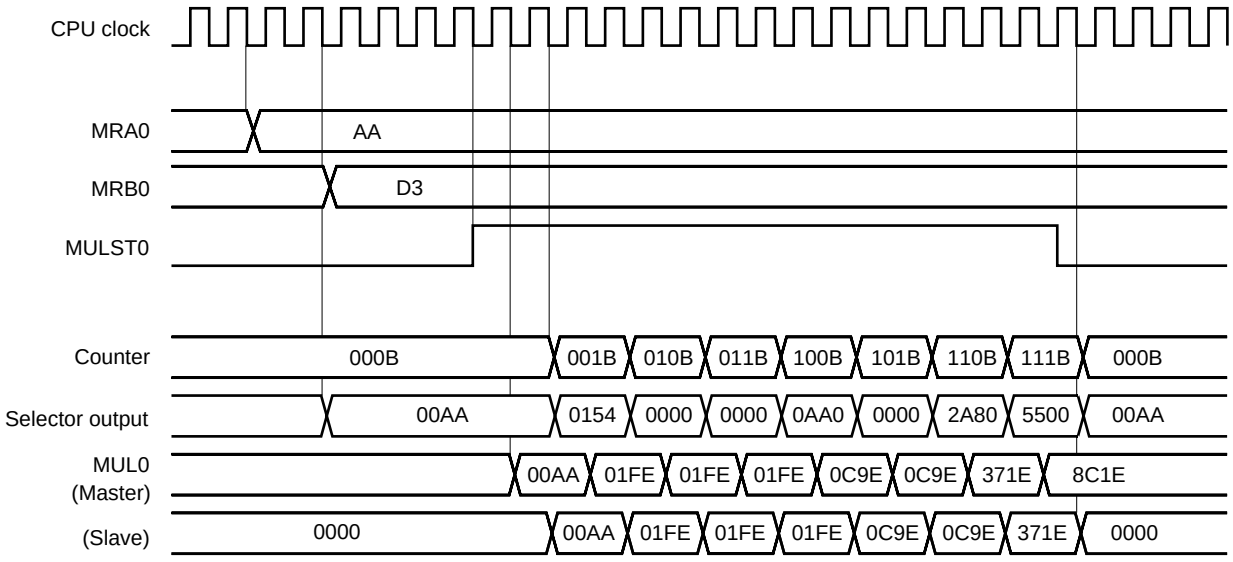
<1> Counting is started by setting MULST0.

<2> The data generated by the selector is added to the data of MUL0 at each CPU clock, and the counter value is incremented by one.

<3> If MULST0 is cleared when the counter value is 111B, the operation is stopped. At this time, MUL0 holds the data.

<4> While MULST0 is low, the counter and slave are cleared.

Figure 16-3. Multiplier Operation Timing (Example of AAH × D3H)



CHAPTER 17 INTERRUPT FUNCTIONS

17.1 Interrupt Function Types

The following two types of interrupt functions are used.

(1) Non-maskable interrupt

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

A standby release signal is generated.

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

(2) Maskable interrupt

These interrupts undergo mask control. If two or more interrupts are simultaneously generated, each interrupt has a predetermined priority (priority) as shown in Table 17-1.

A standby release signal is generated.

★ For the μPD789167 and 789177 Subseries, the maskable interrupt has four sources of external interrupts and ten sources of internal interrupts. For the μPD789167Y and 789177Y Subseries, the maskable interrupt has four sources of external interrupts and 12 sources of internal interrupts.

17.2 Interrupt Sources and Configuration

★ There are a total of 15 non-maskable and maskable interrupts in the interrupt sources for the μPD789167 and 789177 Subseries, and a total of 17 non-maskable and maskable interrupts in the interrupt sources for the μPD789167Y and 789177Y Subseries (see **Table 17-1**).

Table 17-1. Interrupt Sources

Interrupt Type	Priority[Note 1]	Interrupt Source		Internal/External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable interrupt	–	INTWDT	Watchdog timer overflow (when watchdog timer mode 1 is selected)	Internal	0004H	(A)
Maskable interrupt	0	INTWDT	Watchdog timer overflow (when interval timer mode is selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTSR20	End of UART reception on serial interface 20	Internal	000EH	(B)
		INTCSI20	End of three-wire SIO transfer reception on serial interface 20			
	6	INTST20	End of UART transmission on serial interface 20		0010H	
	7	INTWT	Watch timer interrupt		0012H	
	8	INTWTI	Interval timer interrupt		0014H	
	9	INTTM80	Generation of match signal for 8-bit timer/event counter 80		0016H	
	10	INTTM81	Generation of match signal for 8-bit timer/event counter 81		0018H	
	11	INTTM82	Generation of match signal for 8-bit timer 82		001AH	
	12	INTTM90	Generation of match signal for 16-bit timer 90		001CH	
★	13	INTSM0[Note 3]	SMB interrupt		001EH	
★	14	INTSMBOV0[Note 3]	SMB timeout interrupt		0020H	
	15	INTAD0	A/D conversion completion signal		0022H	

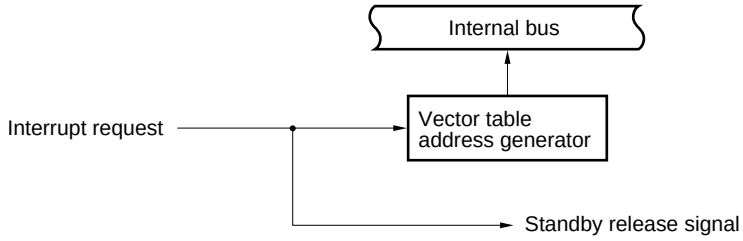
Notes 1. The priority regulates which maskable interrupt is higher, when two or more maskable interrupts are requested simultaneously. Zero signifies the highest priority, while 15 is the lowest.

2. Basic configuration types (A), (B), and (C) correspond to (A), (B), and (C) in Figure 17-1, respectively.

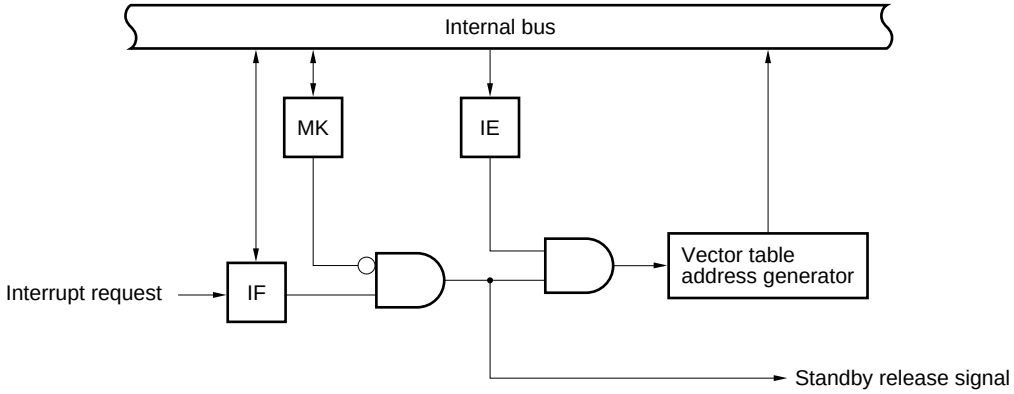
3. For the μPD789167Y and 789177Y Subseries only

Figure 17-1. Basic Configuration of Interrupt Function

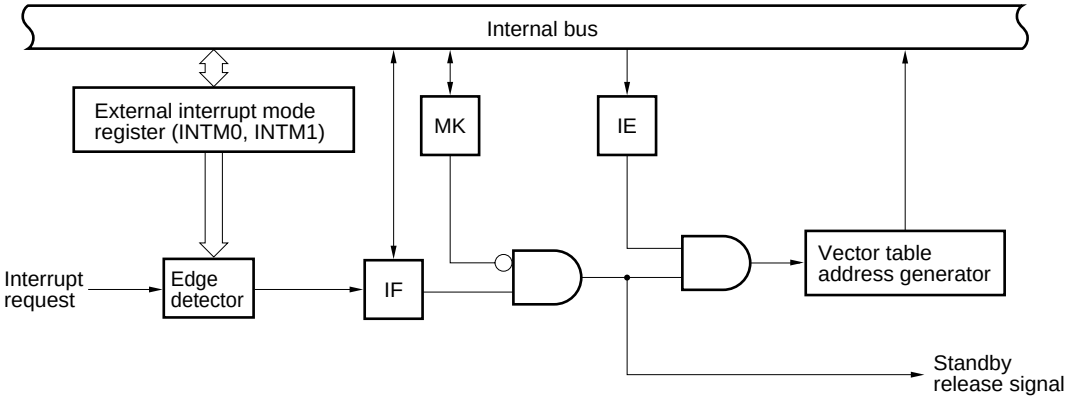
(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt



IF: Interrupt request flag

IE: Interrupt enable flag

MK: Interrupt mask flag

17.3 Interrupt Function Control Registers

The interrupt functions are controlled by the following registers:

- Interrupt request flag registers 0 and 1 (IF0 and IF1)
- Interrupt mask flag registers 0 and 1 (MK0 and MK1)
- External interrupt mode registers 0 and 1 (INTM0 and INTM1)
- Program status word (PSW)

Table 17-2 lists interrupt requests, the corresponding interrupt request flags, and interrupt mask flags.

Table 17-2. Interrupt Request Signals and Corresponding Flags

Interrupt Request Signal	Interrupt Request Flag	Interrupt Mask Flag
INTWDT	TMIF4	TMMK4
INTP0	PIF0	PMK0
INTP1	PIF1	PMK1
INTP2	PIF2	PMK2
INTP3	PIF3	PMK3
INTSR20/INTCSI20	SRIF20	SRMK20
INTST20	STIF20	STMK20
INTWT	WTIF	WTMK
INTWTI	WTIIF	WTIMK
INTTM80	TMIF80	TMMK80
INTTM81	TMIF81	TMMK81
INTTM82	TMIF82	TMMK82
INTTM90	TMIF90	TMMK90
★ INTSMB0[Note]	SMBIF0[Note]	SMBMK0[Note]
★ INTSMBOV0[Note]	SMBOVIF0[Note]	SMBOVMK0[Note]
INTAD0	ADIF0	ADMK0

Note For the μPD789167Y and 789177Y Subseries only

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

An interrupt request flag is set to 1, when the corresponding interrupt request is issued, or when the related instruction is executed. It is cleared to 0, when the interrupt request is accepted, when a $\overline{\text{RESET}}$ signal is input, or when a related instruction is executed.

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

$\overline{\text{RESET}}$ input clears IF0 and IF1 to 00H.

Figure 17-2. Format of Interrupt Request Flag Register

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
IF0	WTIF	STIF20	SRIF20	PIF3	PIF2	PIF1	PIF0	TMIF4	FFE0H	00H	R/W
	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
★ IF1	ADIF0	SMBOVIF0 Note	SMBIF0 Note	TMIF90	TMIF82	TMIF81	TMIF80	WTIIF	FFE1H	00H	R/W

××IF×	Interrupt request flag
0	No interrupt request signal has been issued.
1	An interrupt request signal has been issued; an interrupt request has been made.

Note This flag is provided for the μPD789167Y and 789177Y Subseries only. For the μPD789167 and 789177 Subseries, the flag must be set to 0.

Cautions 1. The TMIF4 flag can be read- and write-accessed only when the watchdog timer is being used as an interval timer. It must be cleared to 0 if the watchdog timer is used in watchdog timer mode 1 or 2.

2. When port 3 is being used as an output port, and its output level is changed, an interrupt request flag is set, because this port is also used as an external interrupt input. To use port 3 in output mode, therefore, the interrupt mask flag must be set to 1 in advance.

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

The interrupt mask flags are used to enable and disable the corresponding maskable interrupts.

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 17-3. Format of Interrupt Mask Flag Register

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
MK0	WTMK	STMK20	SRMK20	PMK3	PMK2	PMK1	PMK0	TMMK4	FFE4H	FFH	R/W

	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>			
★ MK1	ADMK0	SMBOVMK0 Note	SMBMK0 Note	TMMK90	TMMK82	TMMK81	TMMK80	WTIMK	FFE5H	FFH	R/W

××MK×	Interrupt handling control
0	Enable interrupt handling.
1	Disable interrupt handling.

Note This flag is provided for the μPD789167Y and 789177Y Subseries only. For the μPD789167 and 789177 Subseries, the flag must be set to 1.

Cautions 1. When the watchdog timer is being used in watchdog timer mode 1 or 2, any attempt to read TMMK4 flag results in an undefined value being detected.

2. When port 3 is being used as an output port, and its output level is changed, an interrupt request flag is set, because this port is also used as an external interrupt input. To use port 3 in output mode, therefore, the interrupt mask flag must be set to 1 in advance.

(3) External interrupt mode register 0 (INTM0)

INTM0 is used to specify the valid edge for INTP0 to INTP2.

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

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

Figure 17-4. 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 all be set to 0.

2. Before setting INTM0, set the corresponding interrupt mask flag register (××MK×) to 1 to disable interrupts.

To enable interrupts, clear to 0 the corresponding interrupt request flag (××IF×), then the corresponding interrupt mask flag register (××MK×).

(4) External interrupt mode register 1 (INTM1)

INTM1 is used to specify the valid edge for INTP3 and INTLVI0.

INTM1 is set with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears INTM1 to 00H.

Figure 17-5. 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	ES41	ES40	ES31	ES30	FFEDH	00H	R/W

ES41	ES40	INTLVI0 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

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 4 to 7 must all be set to 0.

2. Before setting INTM1, set the corresponding interrupt mask flag register (××MK×) to 1 to disable interrupts.

To enable interrupts, clear to 0 the corresponding interrupt request flag (××IF×), then the corresponding interrupt mask flag register (××MK×).

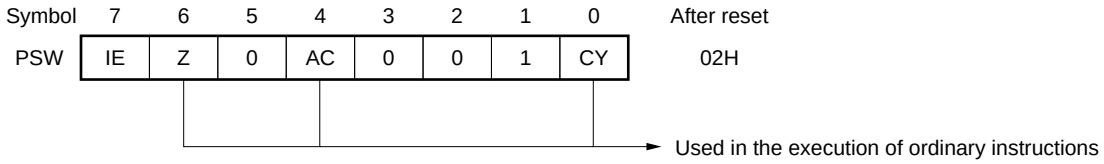
(5) Program status word (PSW)

The program status word is used to hold the instruction execution result and the current status of the interrupt requests. The IE flag, used to enable and disable maskable interrupts, is mapped to PSW.

PSW can be read- and write-accessed in 8-bit units, as well as using bit manipulation instructions and dedicated instructions (EI and DI). When a vector interrupt is accepted, PSW is automatically saved to a stack, and the IE flag is reset to 0.

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

Figure 17-6. Program Status Word Configuration



IE	Whether to enable/disable interrupt acceptance
0	Disable
1	Enable

17.4 Interrupt Processing Operation

17.4.1 Non-maskable interrupt request acceptance operation

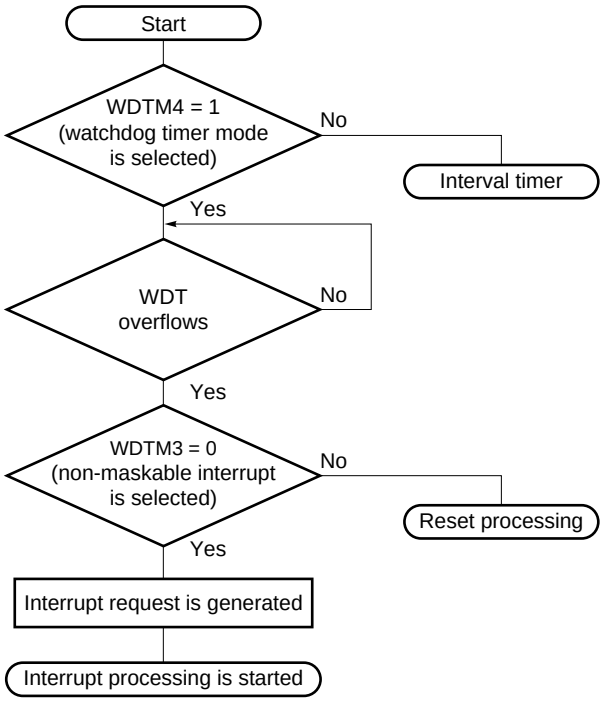
The non-maskable interrupt request is unconditionally accepted even when interrupts are disabled. It is not subject to interrupt priority control and takes precedence over all other interrupts.

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 17-7 shows the flowchart from non-maskable interrupt request generation to acceptance. Figure 17-8 shows the timing of non-maskable interrupt request acceptance. Figure 17-9 shows the acceptance operation if multiple non-maskable interrupts are generated.

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

Figure 17-7. Flowchart from Non-Maskable Interrupt Request Generation to Acceptance



WDTM: Watchdog timer mode register
WDT: Watchdog timer

Figure 17-8. Timing of Non-Maskable Interrupt Request Acceptance

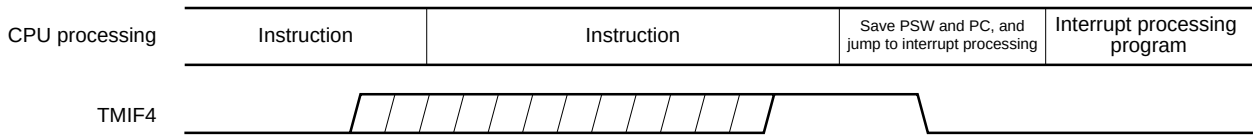
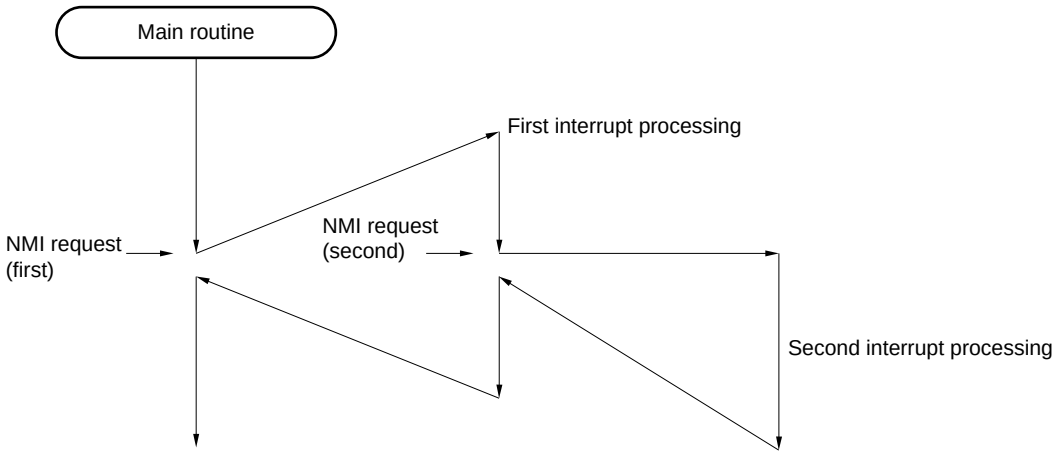


Figure 17-9. Accepting Non-Maskable Interrupt Request



17.4.2 Maskable interrupt request acceptance operation

A maskable interrupt request can be accepted when the interrupt request flag is set to 1 and the corresponding interrupt mask flag is cleared to 0. A vectored interrupt request is accepted in the interrupt enabled status (when the IE flag is set to 1).

The time required to start the interrupt processing after a maskable interrupt request has been generated is shown in Table 17-3.

See Figures 17-11 and 17-12 for the interrupt request acceptance timing.

Table 17-3. Time from Generation of Maskable Interrupt Request to Processing

Minimum Time	Maximum Time[Note]
9 clocks	19 clocks

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

Remark 1 clock: $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

When two or more maskable interrupt requests are generated at the same time, they are accepted starting from the interrupt request assigned the highest priority.

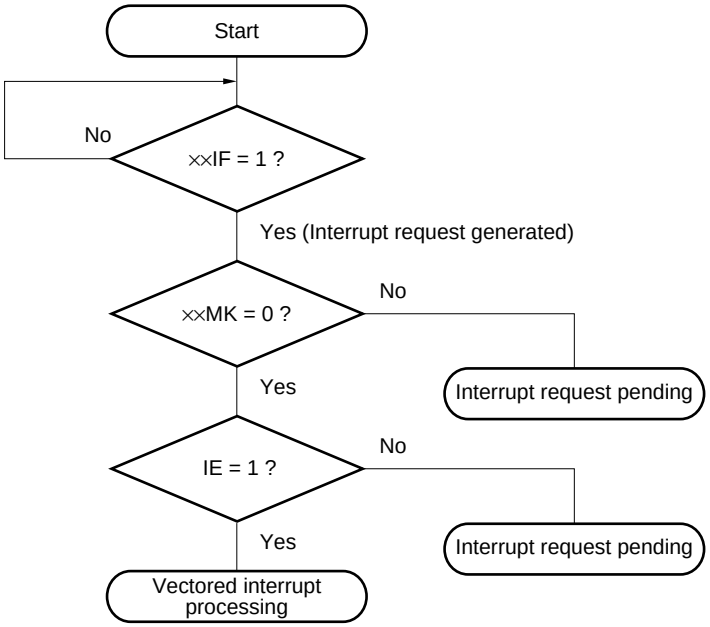
A pending interrupt is accepted when the status where it can be accepted is set.

Figure 17-10 shows the algorithm of accepting interrupt requests.

When a maskable interrupt request is accepted, the contents of PSW and PC are saved to the stack in that order, the IE flag is reset to 0, and the data in the vector table determined for each interrupt request is loaded to the PC, and execution branches.

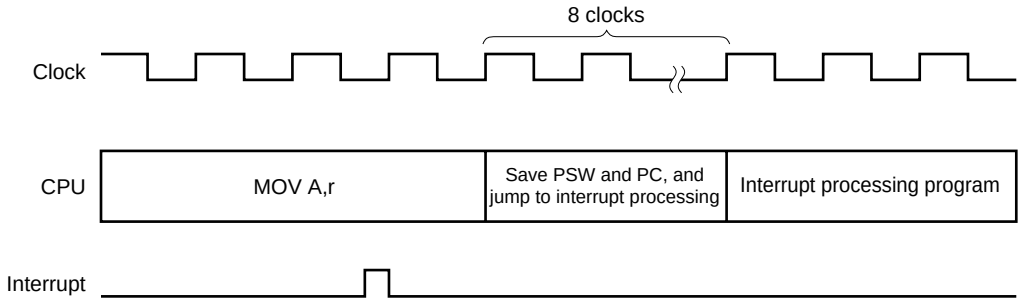
To return from interrupt processing, use the RETI instruction.

Figure 17-10. Interrupt Request Acceptance Processing Algorithm



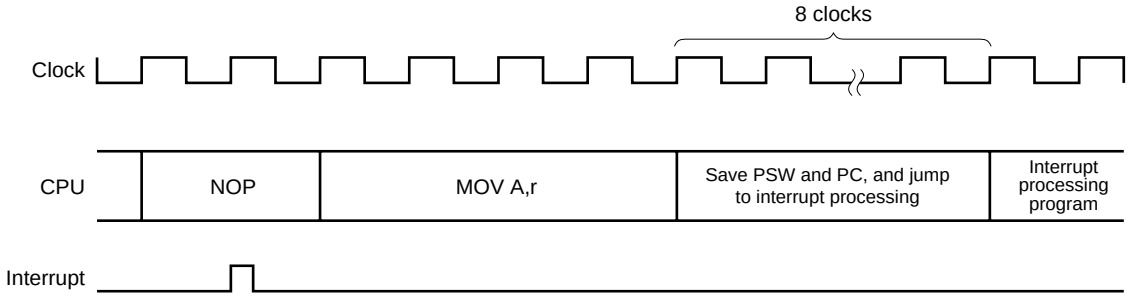
××IF: Interrupt request flag
××MK: Interrupt mask flag
IE: Flag to control maskable interrupt request acceptance (1 = enable, 0 = disable)

Figure 17-11. Interrupt Request Acceptance Timing (Example of MOV A,r)



If an interrupt request flag (××IF) is set before an instruction clock n (n = 4 to 10) under execution becomes n – 1, the interrupt is accepted after the instruction under execution completes. Figure 17-11 shows an example of the interrupt request acceptance timing for an 8-bit data transfer instruction MOV A,r. Since this instruction is executed for 4 clocks, if an interrupt occurs for 3 clocks after the execution starts, the interrupt acceptance processing is performed after the MOV A,r instruction is completed.

Figure 17-12. Interrupt Request Acceptance Timing (When Interrupt Request Flag Generates at the Last Clock During Instruction Execution)



If an interrupt request flag (××IF) is set at the last clock of the instruction, the interrupt acceptance processing starts after the next instruction is executed. Figure 17-12 shows an example of the interrupt acceptance timing for an interrupt request flag that is set at the second clock of NOP (2-clock instruction). In this case, the MOV A,r instruction after the NOP instruction is executed, and then the interrupt acceptance processing is performed.

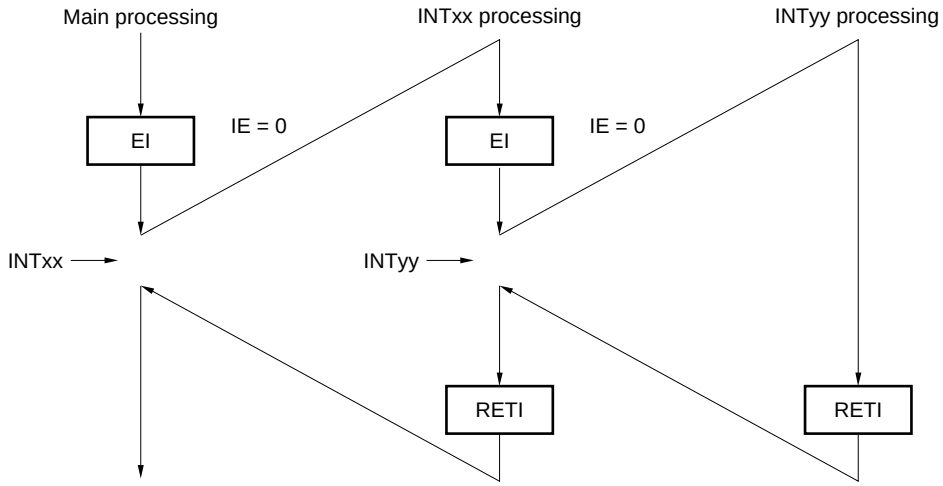
Caution Interrupt requests are reserved while interrupt request flag register 0 or 1 (IF0 or IF1) or the interrupt mask flag register 0 or 1 (MK0 or MK1) is being accessed.

17.4.3 Multiple interrupt processing

Multiple interrupt processing in which another interrupt is accepted while an interrupt is processed can be processed by priority. When two or more interrupts are generated at once, interrupt processing is performed according to the priority assigned to each interrupt request in advance (see **Table 17-1**).

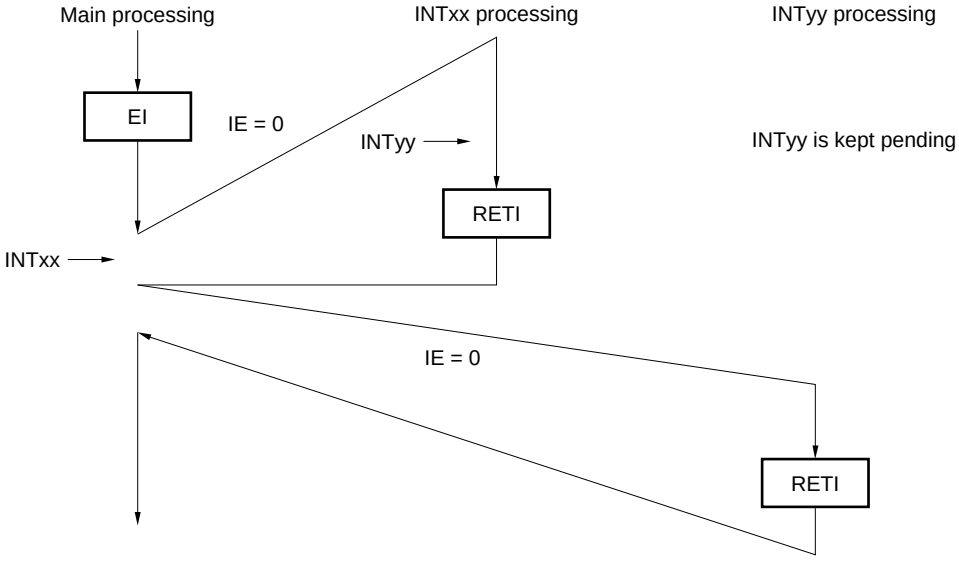
Figure 17-13. Example of Multiple Interrupt

Example 1. Multiple interrupt is accepted



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

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



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

IE = 0: Interrupt request acceptance disabled

17.4.4 Interrupt request reserve

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

- Manipulation instruction for the interrupt request flag registers 0 and 1 (IF0 and IF1)
- Manipulation instruction for the interrupt mask flag registers 0 and 1 (MK0 and MK1)

CHAPTER 18 STANDBY FUNCTION

18.1 Standby Function and Configuration

18.1.1 Standby function

The standby function is to reduce the power dissipation of the system and can be effected in the following two modes:

(1) HALT mode

This mode is set when the HALT instruction is executed. HALT mode stops the operation clock of the CPU. The system clock oscillation circuit continues oscillating. This mode does not reduce the current drain as much as STOP mode, but is useful for resuming processing immediately when an interrupt request is generated, or for intermittent operations.

(2) STOP mode

This mode is set when the STOP instruction is executed. STOP mode stops the main system clock oscillation circuit and stops the entire system. The current drain of the CPU can be substantially reduced in this mode.

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

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

In both modes, the previous contents of the registers, flags, and data memory before setting standby mode are all retained. In addition, the statuses of the output latch of the I/O ports and output buffer are also retained.

Caution To set STOP mode, be sure to stop the operations of the peripheral hardware, and then execute the STOP instruction.

18.1.2 Standby function control register

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

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

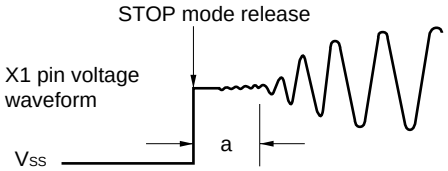
$\overline{\text{RESET}}$ input sets OSTS to 04H. However, the oscillation stabilization time after $\overline{\text{RESET}}$ input is $2^{15}/f_X$, instead of $2^{17}/f_X$.

Figure 18-1. Format of Oscillation Stabilization Time Selection Register

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

OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection
0	0	0	$2^{12}/f_X$ (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 STOP mode is released does not include the time from STOP mode release to clock oscillation start ("a" in the figure below), regardless of release by $\overline{\text{RESET}}$ input or by interrupt generation.



Remarks 1. f_X: System clock oscillation frequency

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

18.2 Operation of Standby Function

18.2.1 HALT mode

(1) HALT mode

HALT mode is set by executing the HALT instruction.

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

Table 18-1. Operation Statuses in HALT Mode

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 generation circuit	Main system clock oscillation enabled			Does not run
CPU	Operation disabled			
Port (output latch)	Remains in the state existing before the selection of HALT mode			
★ 16-bit timer (TM90)	Operation enabled	Operation enabled[Note 1]	Operation enabled	Operation enabled[Note 2]
8-bit timer/event counter (TM80)	Operation enabled			Operation enabled[Note 3]
8-bit timer/event counter (TM81)	Operation enabled			Operation enabled[Note 4]
8-bit timer (TM82)	Operation enabled	Operation enabled[Note 1]	Operation enabled	Operation enabled[Note 5]
Watch timer	Operation enabled	Operation enabled[Note 1]	Operation enabled	Operation enabled[Note 5]
★ Watchdog timer	Operation enabled		Operation disabled	
Serial interface 20	Operation enabled			Operation enabled[Note 6]
★ SMB0	Operation enabled			Operation enabled[Note 7]
A/D converter	Operation disabled			
Multiplier	Operation disabled			
External interrupt	Operation enabled[Note 8]			

Notes 1. Operation is enabled when the main system clock is selected.

2. Operation is enabled when the subsystem clock is selected and when buzzer output is allowed (For details, see **8.5 Notes on Using 16-Bit Timer**).

3. Operation is enabled only when TI80 is selected as the count clock.

4. Operation is enabled only when TI81 is selected as the count clock.

5. Operation is enabled when the subsystem clock is selected.

6. Operation is enabled in both 3-wire serial I/O and UART modes while an external clock is being used.

7. An interrupt can be generated when addresses match during the slave operation.

8. Maskable interrupt that is not masked

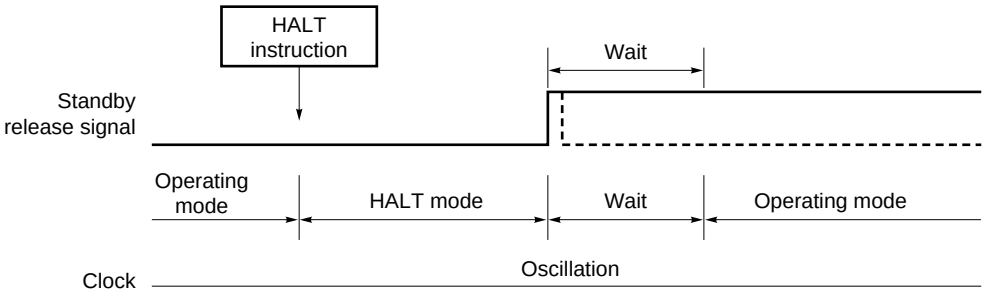
(2) Releasing HALT mode

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

(a) Releasing by unmasked interrupt request

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

Figure 18-2. Releasing HALT Mode by Interrupt



Remarks 1. The broken line indicates the case where the interrupt request that has released standby mode is accepted.

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

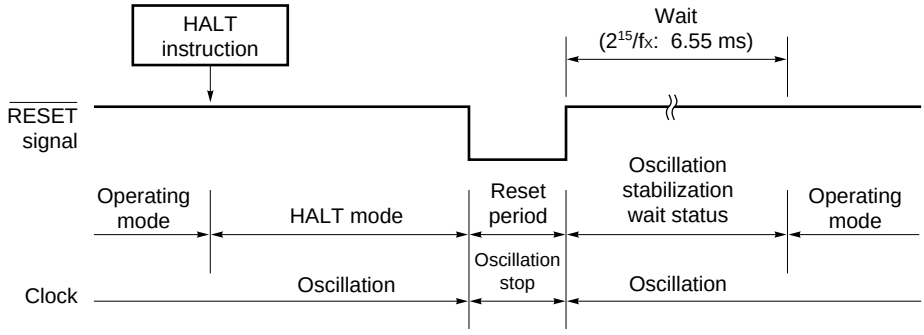
(b) Releasing by non-maskable interrupt request

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 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 18-3. Releasing HALT Mode by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

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

Table 18-2. Operation after Release of HALT Mode

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

×: Don't care

18.2.2 STOP mode

(1) Setting and operation status of STOP mode

STOP mode is set by executing the STOP instruction.

Caution Because standby mode can be released by an interrupt request signal, standby mode is released as soon as it is set if there is an interrupt source whose interrupt request flag is set and interrupt mask flag is reset. When STOP mode is set, therefore, HALT mode is set immediately after the STOP instruction has been executed, the wait time set by the oscillation stabilization time selection register (OSTS) elapses, and then operation mode is set.

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

Table 18-3. Operation Statuses in STOP Mode

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
Clock generation circuit	Main system clock oscillation stopped	
CPU	Operation disabled	
Port (output latch)	Remains in the state existing before the selection of STOP mode	
★ 16-bit timer (TM90)	Operation enabled[Note 1]	Operation disabled
8-bit timer/event counter (TM80)	Operation enabled[Note 2]	
8-bit timer/event counter (TM81)	Operation enabled[Note 3]	
8-bit timer (TM82)	Operation enabled[Note 4]	Operation disabled
Watch timer	Operation enabled[Note 4]	Operation disabled
Watchdog timer	Operation disabled	
Serial interface 20	Operation enabled[Note 5]	
★ SMB0	Operation enabled[Note 6]	
A/D converter	Operation disabled	
Multiplier	Operation disabled	
External interrupt	Operation enabled[Note 7]	

Notes 1. Operation is enabled when the subsystem clock is selected and when buzzer output is allowed.

2. Operation is enabled only when TI80 is selected as the count clock.

3. Operation is enabled only when TI81 is selected as the count clock.

4. Operation is enabled when the subsystem clock is selected.

5. Operation is enabled in both 3-wire serial I/O and UART modes while an external clock is being used.

6. An interrupt can be generated when addresses match during the slave operation.

7. Maskable interrupt that is not masked

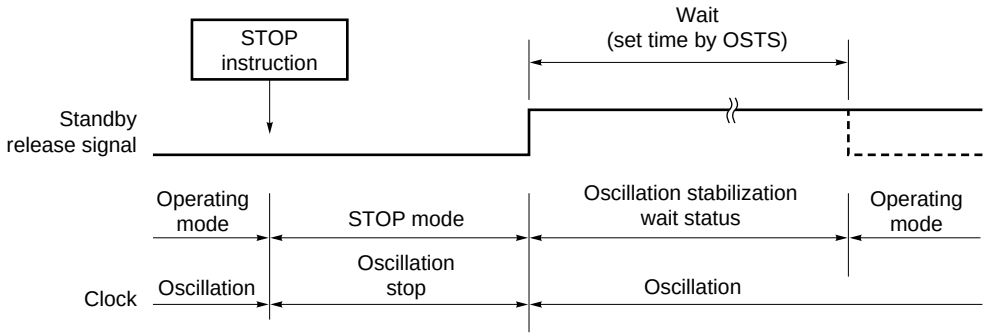
(2) Releasing STOP mode

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

(a) Releasing by unmasked interrupt request

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

Figure 18-4. Releasing STOP Mode by Interrupt

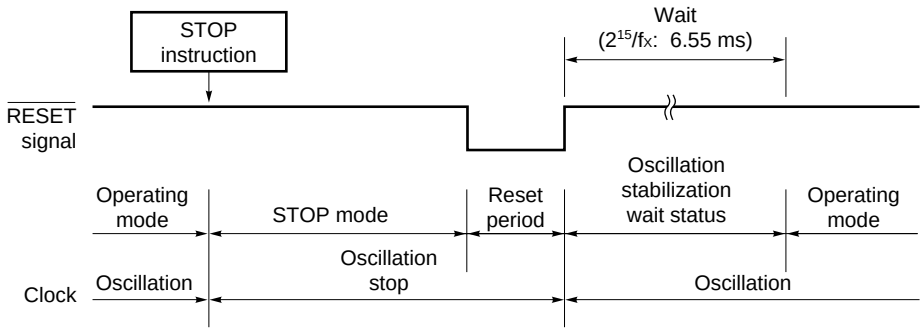


Remark The broken lines indicate the case where the interrupt request that has released standby mode is accepted.

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

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

Figure 18-5. Releasing STOP Mode by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

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

Table 18-4. Operation after Release of STOP Mode

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

×: Don't care

CHAPTER 19 RESET FUNCTION

The following two operations are available to generate reset signals.

(1) External reset input with $\overline{\text{RESET}}$ pin
(2) Internal reset by program run-away time detected with watchdog timer

External and internal reset have no functional differences. In both cases, program execution starts at the address at 0000H and 0001H by reset signal input.

When a low level is input to the $\overline{\text{RESET}}$ pin or the watchdog timer overflows, a reset is applied and each hardware is set to the status shown in Table 19-1. Each pin has a high impedance during reset input or during oscillation settling 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 settling time has elapsed. The reset applied by the watchdog timer overflow is automatically cleared after reset, and program execution is started after the oscillation settling time has elapsed (see **Figures 19-2** through **19-4**).

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

Figure 19-1. Block Diagram of Reset Function

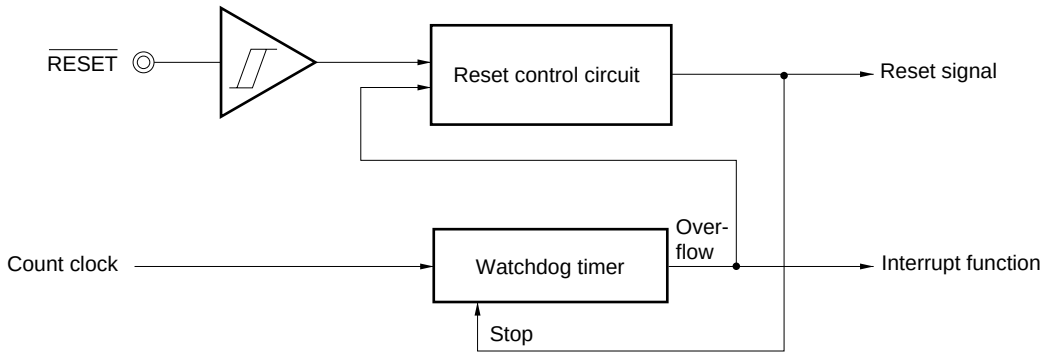


Figure 19-2. Reset Timing by $\overline{\text{RESET}}$ Input

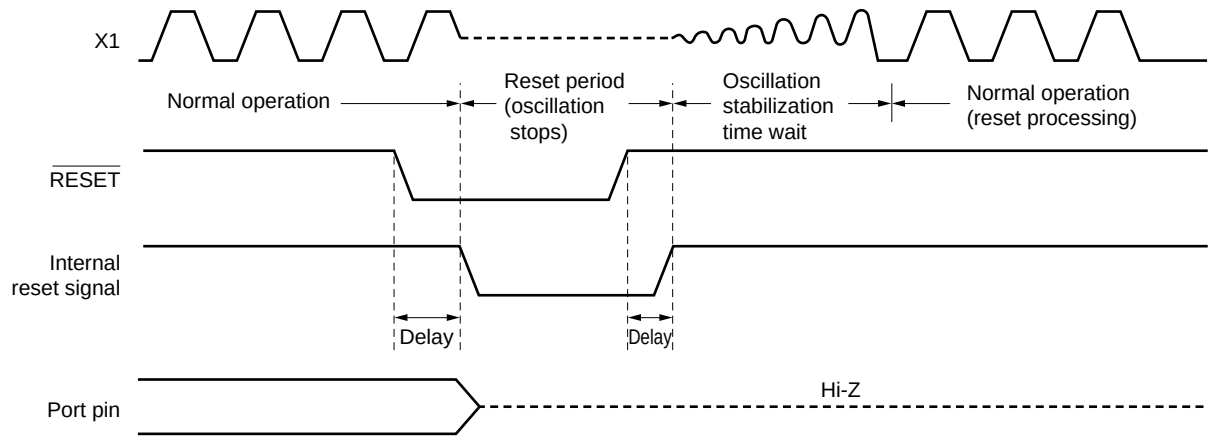


Figure 19-3. Reset Timing by Overflow in Watchdog Timer

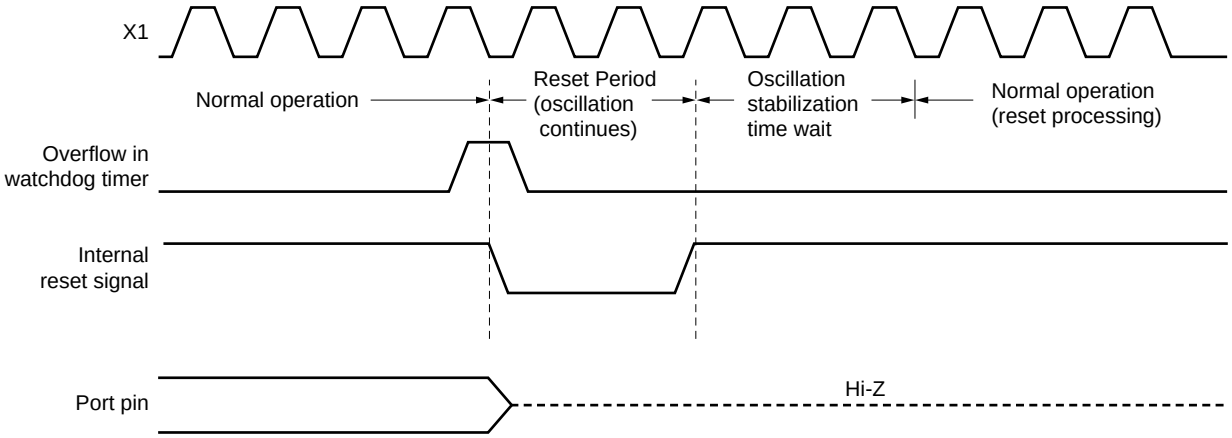


Figure 19-4. Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode

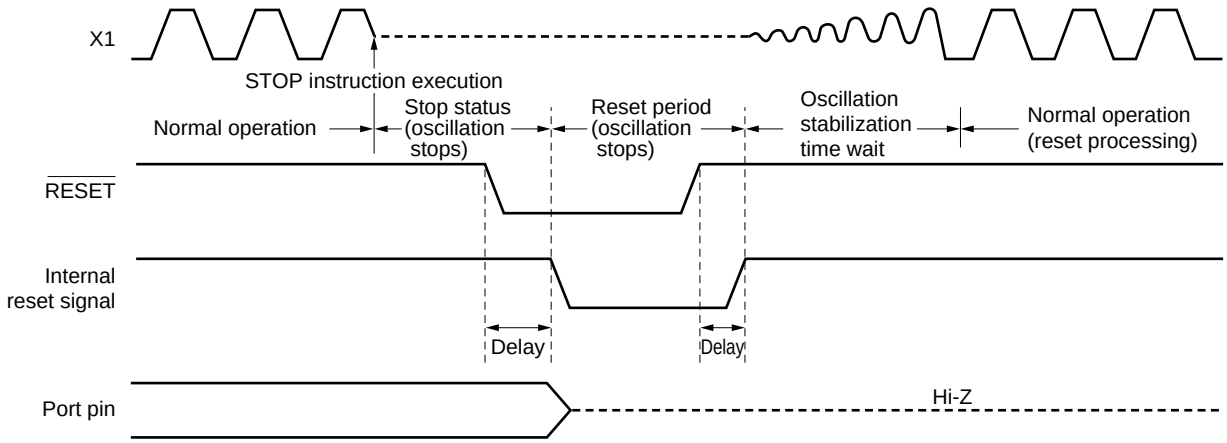


Table 19-1. State of the Hardware after a Reset (1/2)

Hardware		State after Reset
Program counter (PC)[Note 1]		Loaded with the contents of the reset vector table (0000H, 0001H)
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined[Note 2]
	General-purpose register	Undefined[Note 2]
Ports (P0 to P3, P5, P6) (output latch)		00H
Port mode registers (PM0 to PM3, PM5)		FFH
Pull-up resistor option registers (PU0, PUB2, PUB3)		00H
Processor clock control register (PCC)		02H
Suboscillation mode register (SCKM)		00H
Subclock control register (CSS)		00H
Oscillation stabilization time selection register (OSTS)		04H
16-bit timer 90	Timer counter (TM90)	0000H
	Compare register (CR90)	FFFFH
	Capture register (TCP90)	Undefined
	Mode control register (TMC90)	00H
	Buzzer output control register (BZC90)	00H
8-bit timer/event counters 80 to 82	Timer counters (TM80 to TM82)	00H
	Compare registers (CR80 to CR82)	Undefined
	Mode control registers (TMC80 to TMC82)	00H
Watch timer	Mode control register (WTM)	00H
Watchdog timer	Timer clock selection register (TCL2)	00H
	Mode register (WDTM)	00H
A/D converter	Mode register (ADM0)	00H
	A/D input selection register (ADS0)	00H
	A/D conversion result register (ADCR0)	Undefined
Serial interface 20	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
	Transmission shift register (TXS20)	FFH
	Reception buffer register (RXB20)	Undefined

Notes 1. While a reset signal is being input, and during the oscillation stabilization period, the contents of the PC will be undefined, while the remainder of the hardware will be the same as after the reset.

2. In standby mode, the RAM enters the hold state after a reset.

★ **Table 19-1. State of the Hardware after a Reset (2/2)**

Hardware		State after Reset
SMB0	Control register (SMBC0)	00H
	Status register (SMBS0)	00H
	Clock selection register (SMBCL0)	00H
	Slave address register (SMBSVA0)	00H
	Mode register (SMBM0)	20H
	Input level setting register (SMBVI0)	00H
	Shift register (SMB0)	00H
Multiplier	16-bit multiplication result storage register (MUL0)	Undefined
	Multiplication data registers (MRA0, MRB0)	Undefined
	Multiplier control register (MULC0)	00H
Interrupts	Request flag registers (IF0, IF1)	00H
	Mask flag registers (MK0, MK1)	FFH
	External interrupt mode registers (INTM0, INTM1)	00H

CHAPTER 20 μPD78F9177 AND μPD78F9177Y

The μPD78F9177 and μPD78F9177Y are flash memory versions of the μPD789177 and 789177Y Subseries. The μPD78F9177 replaces the internal ROM of the μPD789166, μPD789167, μPD789176, and μPD789177 with flash memory, while the μPD78F9177Y replaces the internal ROM of the μPD789166Y, μPD789167Y, μPD789176Y, and μPD789177Y with flash memory. The differences between the flash memory and the mask ROM versions are shown in Table 20-1.

Table 20-1. Differences between Flash Memory and Mask ROM Versions

Item		Flash Memory	Mask ROM	
		μPD78F9177 μPD78F9177Y	μPD789167 Subseries μPD789167Y Subseries	μPD789177 Subseries μPD789177Y Subseries
Internal memory	ROM structure	Flash memory	Mask ROM	
	ROM capacity	24 Kbytes	μPD789166: 16 Kbytes μPD789167: 24 Kbytes μPD789166Y: 16 Kbytes μPD789167Y: 24 Kbytes	μPD789176: 16 Kbytes μPD789177: 24 Kbytes μPD789176Y: 16 Kbytes μPD789177Y: 24 Kbytes
	High-speed RAM	512 bytes		
Pull-up resistor		17 (software control only)	21 (software control: 17, mask option specification: 4)	
A/D converter resolution		10 bits	8 bits	10 bits
Specification of built-in pull-up resistors for P50 to P53 by mask option		Disabled	Enabled	
V_{PP} pin		Provided	Not provided	
Electric characteristics		Varies depending on flash memory or mask ROM version.		

Cautions 1. There are differences in the amount of 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 on the commercial samples (CS) (not engineering sample, ES) of the mask ROM version.

2. When using A/D conversion result register 0 (ADCR0) with an 8-bit A/D converter (μPD789167 Subseries), manipulate with an 8-bit memory manipulation instruction; when using it with a 10-bit A/D converter (μPD789177 Subseries), use a 16-bit memory manipulation instruction.

When the μPD78F9177, a flash memory counterpart of the μPD789166 or μPD789167, is used, however, ADCR0 can be manipulated with an 8-bit memory manipulation instruction. In this case, use an object file assembled with the μPD789166 or μPD789167. The same is also true for the μPD78F9177Y, a flash memory counterpart of the μPD789166Y or μPD789167Y. When the μPD78F9177Y is used, ADCR0 can be manipulated with an 8-bit memory manipulation instruction. In this case, use an object file assembled with the μPD789166Y or μPD789167Y.

20.1 Flash Memory Programming

The on-chip program memory in the μPD78F9177 or μPD78F9177Y is a flash memory.

The flash memory can be written with the μPD78F9177 or μPD78F9177Y mounted on the target system (on-board). Connect the dedicated flash writer (Flashpro III (part number: FL-PR3, PG-FP3)) to the host machine and target system to write the flash memory.

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

20.1.1 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 20-2. To select a communication mode, the format shown in Figure 20-2 is used. Each communication mode is selected by the number of V_{PP} pulses shown in Table 20-2.

Table 20-2. Communication Mode

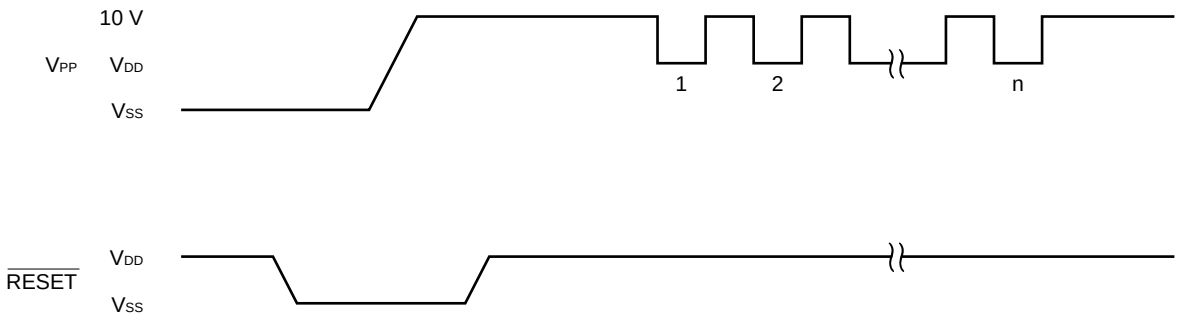
Communication Mode	Pins Used	Number of V_{PP} Pulses
3-wire serial I/O	$\overline{SCK20}$/ASCK20/P20 SO20/TxD20/P21 SI20/RxD20/P22	0
★ SMB[Note 1]	SCL0/P23 SDA0/P24	4
UART	TxD20/SO20/P21 RxD20/SI20/P22	8
★ Pseudo 3-wire mode[Note 2]	P00 (Serial clock input) P01 (Serial data input) P02 (Serial data output)	12

Notes 1. For the μPD789167Y and 789177Y Subseries only

2. Serial transfer is performed by controlling a port by software.

Caution Be sure to select a communication mode depending on the V_{PP} pulse number shown in Table 20-2.

Figure 20-1. Format of Communication Mode Selection



20.1.2 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 20-3 shows the major functions of flash memory programming.

Table 20-3. Major Functions of Flash Memory Programming

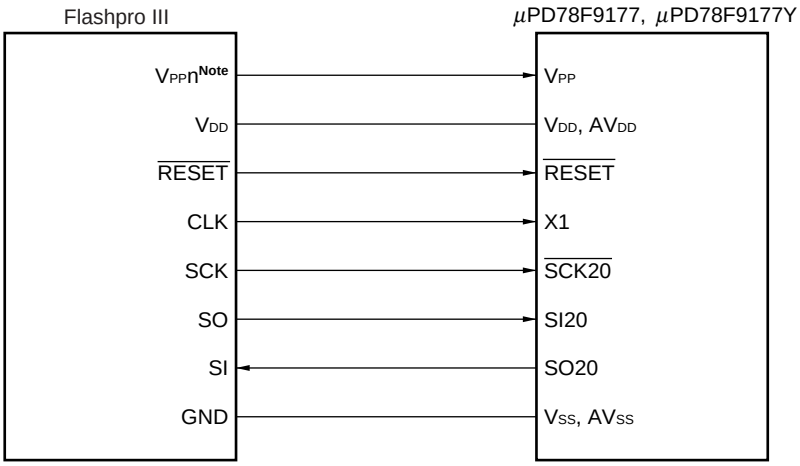
Function	Description
Batch erase	Erases all contents of memory
Batch blank check	Checks erased state of entire memory
Data write	Write to flash memory based on write start address and number of data written (number of bytes)
Batch verify	Compares all contents of memory with input data

20.1.3 Flashpro III connection

Connection between the Flashpro III and the μPD78F9177 or μPD78F9177Y differs depending on the communication mode (3-wire serial I/O, SMB[Note], UART, or pseudo 3-wire mode). Figures 20-2 to 20-5 show the connection in the respective modes.

Note For the μPD78F9177Y only

Figure 20-2. Flashpro III Connection in 3-Wire Serial I/O Mode



Note n = 1, 2

★ **Figure 20-3. Flashpro III Connection in SMB Mode**

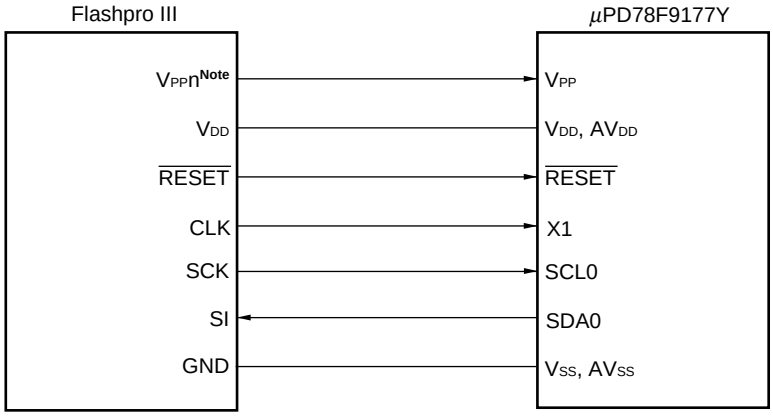
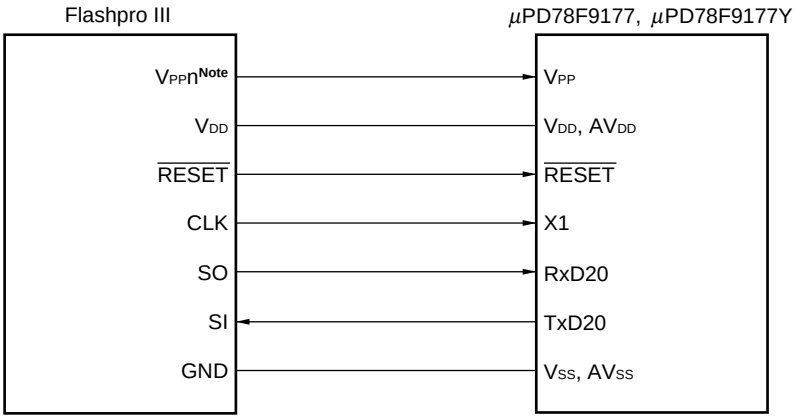
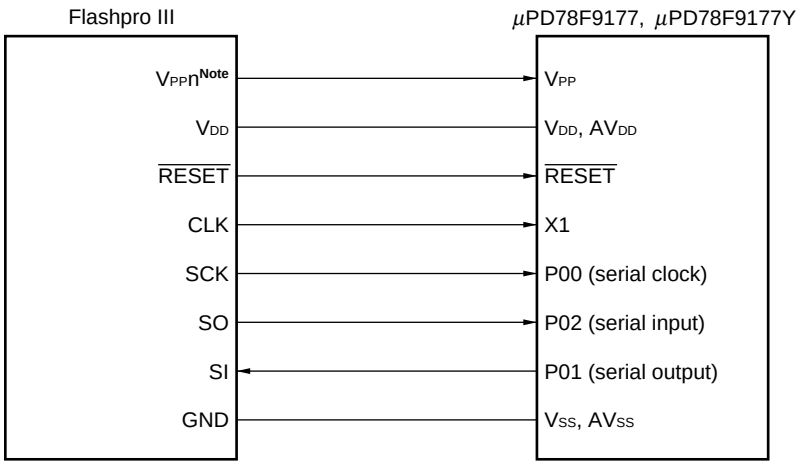


Figure 20-4. Flashpro III Connection in UART Mode



Note n = 1, 2

Figure 20-5. Flashpro III Connection in Pseudo 3-Wire Mode (When P0 is Used)



Note n = 1, 2

20.1.4 Setting with Flashpro III (PG-FP3)

When writing data to flash memory by using Flashpro III (PG-FP3), set the following.

<1> Load the parameter file.

<2> Use the type command to select a serial mode and a serial clock.

<3> An example of setting with PG-FP3 is shown below.

Table 20-4. Setting with PG-FP3

Communication Mode	Setting with PG-FP3		V_{PP} Pulse Count[Note 1]
3-wire serial I/O	COMM PORT	SIO ch-0	0
	CPU CLK	On Target Board	
		In Flashpro	
	On Target Board	4.1943 MHz	
	SIO CLK	1.0 MHz	
	In Flashpro	4.0 MHz	
	SIO CLK	1.0 MHz	
★ SMB[Note 2]	COMM PORT	IIC ch-0	4
	SLAVE ADDRESS	10H	
	IIC CLOCK	100 kHz	
	CPU CLOCK	In Flashpro	
	Flashpro Clock	4.0 MHz[Note 3]	
	Multiple Rate	01.00	
UART	COMM PORT	UART ch-0	8
	CPU CLK	On Target Board	
	On Target Board	4.1943 MHz	
	UART BPS	9,600 bps[Note 4]	
Pseudo 3-wire mode	COMM PORT	Port A	12
	CPU CLK	On Target Board	
		In Flashpro	
	On Target Board	4.1943 MHz	
★	SIO CLK	1.0 kHz	
	In Flashpro	4.0 MHz	
★	SIO CLK	1.0 kHz	

Notes 1. The number of V_{PP} pulses supplied from the Flashpro III when serial communication is initialized. These pulse counts determine the pins used for communication.

2. For the μPD789167Y and 789177Y Subseries only

3. Select 4.0 MHz or 3.125 MHz.

4. Select 9,600 bps, 19,200 bps, 38,400 bps, or 76,800 bps.

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

[MEMO]

CHAPTER 21 MASK OPTION

Table 21-1. 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), whether a pull-up resistor is to be incorporated can be specified by a mask option. The mask option is specified in 1-bit units.

Caution The flash memory versions do not provide the pull-up resistor incorporation function by a mask option.

[MEMO]

CHAPTER 22 INSTRUCTION SET

This chapter lists the instruction set of the μPD789167, 789177, 789167Y, and 789177Y Subseries. For details of the operation and machine language (instruction code) of each instruction, refer to **78K/0S Series User's Manual — Instruction (U11047E)**.

22.1 Operation

22.1.1 Operand identifiers and description methods

Operands are described in "Operand" column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for detail). When there are two or more description methods, select one of them. Alphabetic letters in capitals and symbols, #, !, $, and [ ] are key words and are described as they are. Each symbol has the following meaning.

- #: Immediate data specification
- !: Absolute address specification
- $: Relative address specification
- [ ]: Indirect address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, $ and [ ] symbols.

For operand register identifiers, r and rp, either function names (X, A, C, etc.) or absolute names (names in parenthesis in the table below, R0, R1, R2, etc.) can be used for description.

Table 22-1. 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 5-3 Special Function Registers** for symbols of special function registers.

22.1.2 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
×H, ×L:	Higher 8 bits and lower 8 bits of 16-bit register
∧:	Logical product (AND)
∨:	Logical sum (OR)
∀:	Exclusive logical sum (exclusive OR)
—:	Inverted data
addr16:	16-bit immediate data or label
jdisp8:	Signed 8-bit data (displacement value)

22.1.3 Description of "Flag" column

(Blank):	Unchanged
0:	Cleared to 0
1:	Set to 1
×:	Set/cleared according to the result
R:	Previously saved value is stored

22.2 Operation List

Mnemonic	Operands	Byte	Clock	Operation	Flag		
					Z	AC	CY
MOV	r, #byte	3	6	r ← byte			
	saddr, #byte	3	6	(saddr) ← byte			
	sfr, #byte	3	6	sfr ← byte			
	A, r Note 1	2	4	A ← r			
	r, A Note 1	2	4	r ← A			
	A, saddr	2	4	A ← (saddr)			
	saddr, A	2	4	(saddr) ← A			
	A, sfr	2	4	A ← sfr			
	sfr, A	2	4	sfr ← A			
	A, !addr16	3	8	A ← (addr16)			
	!addr16, A	3	8	(addr16) ← A			
	PSW, #byte	3	6	PSW ← byte	×	×	×
	A, PSW	2	4	A ← PSW			
	PSW, A	2	4	PSW ← A	×	×	×
	A, [DE]	1	6	A ← (DE)			
	[DE], A	1	6	(DE) ← A			
	A, [HL]	1	6	A ← (HL)			
	[HL], A	1	6	(HL) ← A			
	A, [HL + byte]	2	6	A ← (HL + byte)			
	[HL + byte], A	2	6	(HL + byte) ← A			
XCH	A, X	1	4	A ↔ X			
	A, r Note 2	2	6	A ↔ r			
	A, saddr	2	6	A ↔ (saddr)			
	A, sfr	2	6	A ↔ sfr			
	A, [DE]	1	8	A ↔ (DE)			
	A, [HL]	1	8	A ↔ (HL)			
	A, [HL + byte]	2	8	A ↔ (HL + byte)			

Notes 1. Except r = A.

2. Except r = A, X.

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag		
					Z	AC	CY
MOVW	rp, #word	3	6	rp ← word			
	AX, saddrp	2	6	AX ← (saddrp)			
	saddrp, AX	2	8	(saddrp) ← AX			
	AX, rp Note	1	4	AX ← rp			
	rp, AX Note	1	4	rp ← AX			
XCHW	AX, rp Note	1	8	AX ↔ rp			
ADD	A, #byte	2	4	A, CY ← A + byte	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte	×	×	×
	A, r	2	4	A, CY ← A + r	×	×	×
	A, saddr	2	4	A, CY ← A + (saddr)	×	×	×
	A, !addr16	3	8	A, CY ← A + (addr16)	×	×	×
	A, [HL]	1	6	A, CY ← A + (HL)	×	×	×
	A, [HL + byte]	2	6	A, CY ← A + (HL + byte)	×	×	×
ADDC	A, #byte	2	4	A, CY ← A + byte + CY	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte + CY	×	×	×
	A, r	2	4	A, CY ← A + r + CY	×	×	×
	A, saddr	2	4	A, CY ← A + (saddr) + CY	×	×	×
	A, !addr16	3	8	A, CY ← A + (addr16) + CY	×	×	×
	A, [HL]	1	6	A, CY ← A + (HL) + CY	×	×	×
	A, [HL + byte]	2	6	A, CY ← A + (HL + byte) + CY	×	×	×
SUB	A, #byte	2	4	A, CY ← A – byte	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte	×	×	×
	A, r	2	4	A, CY ← A – r	×	×	×
	A, saddr	2	4	A, CY ← A – (saddr)	×	×	×
	A, !addr16	3	8	A, CY ← A – (addr16)	×	×	×
	A, [HL]	1	6	A, CY ← A – (HL)	×	×	×
	A, [HL + byte]	2	6	A, CY ← A – (HL + byte)	×	×	×

Note Only when rp = BC, DE, or HL.

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag		
					Z	AC	CY
SUBC	A, #byte	2	4	A, CY ← A – byte – CY	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte – CY	×	×	×
	A, r	2	4	A, CY ← A – r – CY	×	×	×
	A, saddr	2	4	A, CY ← A – (saddr) – CY	×	×	×
	A, !addr16	3	8	A, CY ← A – (addr16) – CY	×	×	×
	A, [HL]	1	6	A, CY ← A – (HL) – CY	×	×	×
	A, [HL + byte]	2	6	A, CY ← A – (HL + byte) – CY	×	×	×
AND	A, #byte	2	4	A ← A ∧ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ∧ byte	×		
	A, r	2	4	A ← A ∧ r	×		
	A, saddr	2	4	A ← A ∧ (saddr)	×		
	A, !addr16	3	8	A ← A ∧ (addr16)	×		
	A, [HL]	1	6	A ← A ∧ (HL)	×		
	A, [HL + byte]	2	6	A ← A ∧ (HL + byte)	×		
OR	A, #byte	2	4	A ← A ∨ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ∨ byte	×		
	A, r	2	4	A ← A ∨ r	×		
	A, saddr	2	4	A ← A ∨ (saddr)	×		
	A, !addr16	3	8	A ← A ∨ (addr16)	×		
	A, [HL]	1	6	A ← A ∨ (HL)	×		
	A, [HL + byte]	2	6	A ← A ∨ (HL + byte)	×		
XOR	A, #byte	2	4	A ← A ▽ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ▽ byte	×		
	A, r	2	4	A ← A ▽ r	×		
	A, saddr	2	4	A ← A ▽ (saddr)	×		
	A, !addr16	3	8	A ← A ▽ (addr16)	×		
	A, [HL]	1	6	A ← A ▽ (HL)	×		
	A, [HL + byte]	2	6	A ← A ▽ (HL + byte)	×		

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag		
					Z	AC	CY
CMP	A, #byte	2	4	A – byte	×	×	×
	saddr, #byte	3	6	(saddr) – byte	×	×	×
	A, r	2	4	A – r	×	×	×
	A, saddr	2	4	A – (saddr)	×	×	×
	A, !addr16	3	8	A – (addr16)	×	×	×
	A, [HL]	1	6	A – (HL)	×	×	×
	A, [HL + byte]	2	6	A – (HL + byte)	×	×	×
ADDW	AX, #word	3	6	AX, CY ← AX + word	×	×	×
SUBW	AX, #word	3	6	AX, CY ← AX – word	×	×	×
CMPW	AX, #word	3	6	AX – word	×	×	×
INC	r	2	4	r ← r + 1	×	×	
	saddr	2	4	(saddr) ← (saddr) + 1	×	×	
DEC	r	2	4	r ← r + 1	×	×	
	saddr	2	4	(saddr) ← (saddr) – 1	×	×	
INCW	rp	1	4	rp ← rp + 1			
DECW	rp	1	4	rp ← rp – 1			
ROR	A, 1	1	2	(CY, $A_7 \leftarrow A_0$, $A_{m-1} \leftarrow A_m$) × 1			×
ROL	A, 1	1	2	(CY, $A_0 \leftarrow A_7$, $A_{m+1} \leftarrow A_m$) × 1			×
RORC	A, 1	1	2	(CY ← A_0, A_7 ← CY, $A_{m-1} \leftarrow A_m$) × 1			×
ROLC	A, 1	1	2	(CY ← A_7, A_0 ← CY, $A_{m+1} \leftarrow A_m$) × 1			×
SET1	saddr.bit	3	6	(saddr.bit) ← 1			
	sfr.bit	3	6	sfr.bit ← 1			
	A.bit	2	4	A.bit ← 1			
	PSW.bit	3	6	PSW.bit ← 1	×	×	×
	[HL].bit	2	10	(HL).bit ← 1			
CLR1	saddr.bit	3	6	(saddr.bit) ← 0			
	sfr.bit	3	6	sfr.bit ← 0			
	A.bit	2	4	A.bit ← 0			
	PSW.bit	3	6	PSW.bit ← 0	×	×	×
	[HL].bit	2	10	(HL).bit ← 0			
SET1	CY	1	2	CY ← 1			1
CLR1	CY	1	2	CY ← 0			0
NOT1	CY	1	2	CY ← $\overline{CY}$			×

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

Mnemonic	Operands	Byte	Clock	Operation	Flag		
					Z	AC	CY
CALL	!addr16	3	6	$(SP-1) \leftarrow (PC+3)_H$, $(SP-2) \leftarrow (PC+3)_L$, $PC \leftarrow addr16$, $SP \leftarrow SP-2$			
CALLT	[addr5]	1	8	$(SP-1) \leftarrow (PC+1)_H$, $(SP-2) \leftarrow (PC+1)_L$, $PC_H \leftarrow (00000000, addr5+1)$, $PC_L \leftarrow (00000000, addr5)$, $SP \leftarrow SP-2$			
RET		1	6	$PC_H \leftarrow (SP+1)$, $PC_L \leftarrow (SP)$, $SP \leftarrow SP+2$			
RETI		1	8	$PC_H \leftarrow (SP+1)$, $PC_L \leftarrow (SP)$, $PSW \leftarrow (SP+2)$, $SP \leftarrow SP+3$, $NMIS \leftarrow 0$	R	R	R
PUSH	PSW	1	2	$(SP-1) \leftarrow PSW$, $SP \leftarrow SP-1$			
	rp	1	4	$(SP-1) \leftarrow rp_H$, $(SP-2) \leftarrow rp_L$, $SP \leftarrow SP-2$			
POP	PSW	1	4	$PSW \leftarrow (SP)$, $SP \leftarrow SP+1$	R	R	R
	rp	1	6	$rp_H \leftarrow (SP+1)$, $rp_L \leftarrow (SP)$, $SP \leftarrow SP+2$			
MOVW	SP, AX	2	8	$SP \leftarrow AX$			
	AX, SP	2	6	$AX \leftarrow SP$			
BR	!addr16	3	6	$PC \leftarrow addr16$			
	$addr16	2	6	$PC \leftarrow PC+2+jdisp8$			
	AX	1	6	$PC_H \leftarrow A$, $PC_L \leftarrow X$			
BC	$saddr16	2	6	$PC \leftarrow PC+2+jdisp8$ if CY = 1			
BNC	$saddr16	2	6	$PC \leftarrow PC+2+jdisp8$ if CY = 0			
BZ	$saddr16	2	6	$PC \leftarrow PC+2+jdisp8$ if Z = 1			
BNZ	$saddr16	2	6	$PC \leftarrow PC+2+jdisp8$ if Z = 0			
BT	saddr.bit, $addr16	4	10	$PC \leftarrow PC+4+jdisp8$ if (saddr.bit) = 1			
	sfr.bit, $addr16	4	10	$PC \leftarrow PC+4+jdisp8$ if sfr.bit = 1			
	A.bit, $addr16	3	8	$PC \leftarrow PC+3+jdisp8$ if A.bit = 1			
	PSW.bit, $addr16	4	10	$PC \leftarrow PC+4+jdisp8$ if PSW.bit = 1			
BF	saddr.bit, $addr16	4	10	$PC \leftarrow PC+4+jdisp8$ if (saddr.bit) = 0			
	sfr.bit, $addr16	4	10	$PC \leftarrow PC+4+jdisp8$ if sfr.bit = 0			
	A.bit, $addr16	3	8	$PC \leftarrow PC+3+jdisp8$ if A.bit = 0			
	PSW.bit, $addr16	4	10	$PC \leftarrow PC+4+jdisp8$ if PSW.bit = 0			
DBNZ	B, $addr16	2	6	$B \leftarrow B-1$, then $PC \leftarrow PC+2+jdisp8$ if $B \neq 0$			
	C, $addr16	2	6	$C \leftarrow C-1$, then $PC \leftarrow PC+2+jdisp8$ if $C \neq 0$			
	saddr, $addr16	3	8	$(saddr) \leftarrow (saddr)-1$, then $PC \leftarrow PC+3+jdisp8$ if $(saddr) \neq 0$			
NOP		1	2	No Operation			
EI		3	6	IE ← 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 processor clock control register (PCC).

22.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, INC, DEC, ROR, ROL, RORC, ROLC, PUSH, POP, DBNZ

1st Operand \ 2nd Operand	#byte	A	r	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL + byte]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV[Note] XCH[Note] ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV											INC DEC
B, C											DBNZ		
sfr	MOV	MOV											
saddr	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV									DBNZ		INC DEC
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											
[HL + byte]		MOV											

Note Except r = A.

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

1st Operand \ 2nd Operand	#word	AX	rp[Note]	saddrp	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	
rp	MOVW	MOVW[Note]				INCW DECW PUSH POP
saddrp		MOVW				
sp		MOVW				

Note Only when rp = BC, DE, or HL.

(3) Bit manipulation instructions

SET1, CLR1, NOT1, BT, BF

1st Operand \ 2nd Operand	$addr16	None
A.bit	BT BF	SET1 CLR1
sfr.bit	BT BF	SET1 CLR1
saddr.bit	BT BF	SET1 CLR1
PSW.bit	BT BF	SET1 CLR1
[HL].bit		SET1 CLR1
CY		SET1 CLR1 NOT1

(4) Call instructions/branch instructions

CALL, CALLT, BR, BC, BNC, BZ, BNZ, DBNZ

2nd Operand / 1st 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

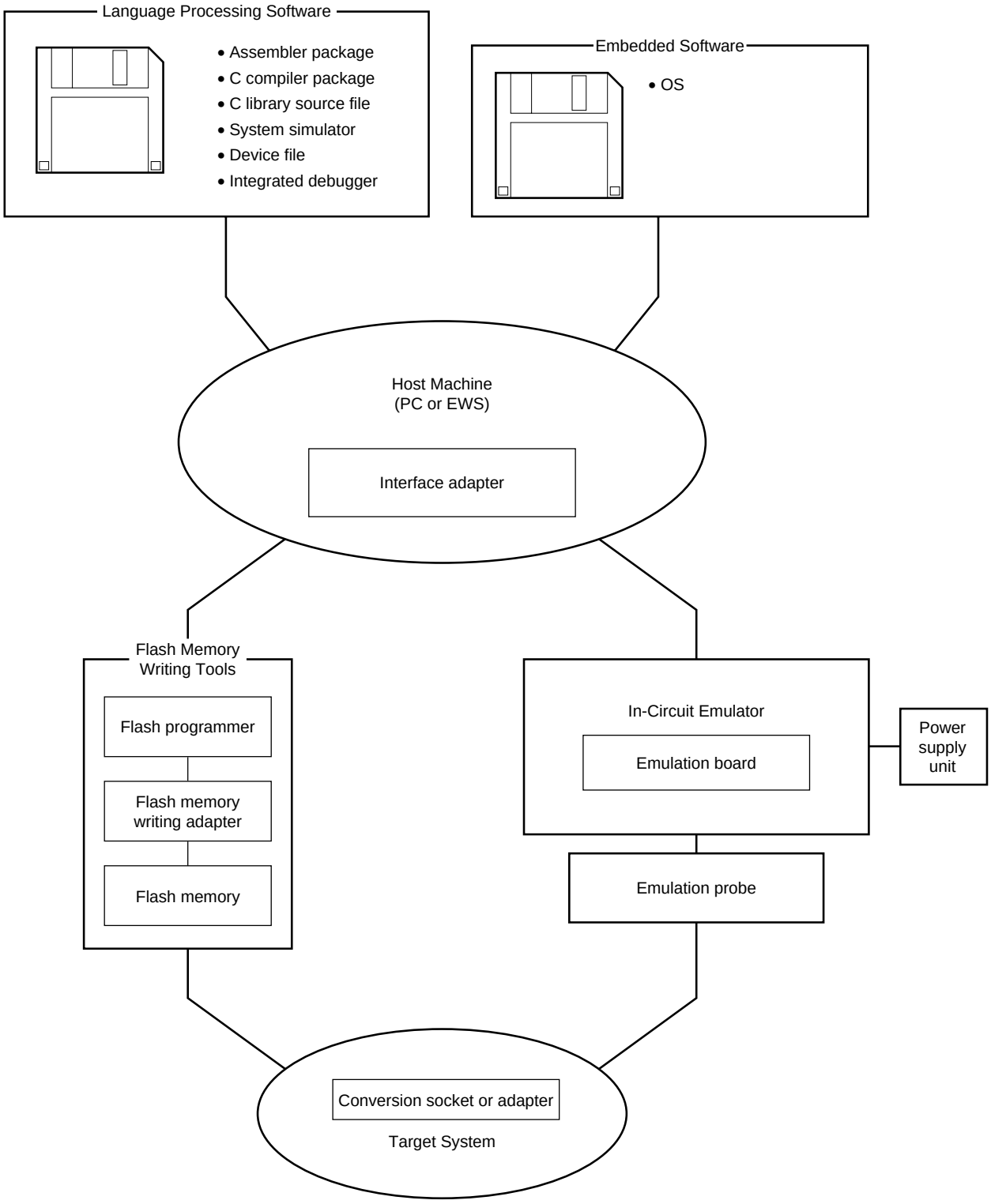
APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for development of systems using the μPD789167, 789177, 789167Y, and 789177Y Subseries. Figure A-1 shows development tools.

- Support of PC98-NX Series
  Unless otherwise specified, products that operate in IBM PC/AT™ or compatibles can operate in the PC98-NX Series. When using the PC98-NX Series, refer to the descriptions for IBM PC/AT™ or compatibles.

- Windows
  Unless otherwise specified, “Windows” refers to the following OSs.
  - Windows 3.1
  - Windows 95
  - Windows NT™ Version 4.0

Figure A-1. Development Tools



A.1 Language Processing Software

RA78K0S Assembler package	Program that converts program written in mnemonic into object code 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 (DF789177). **<Caution when used in PC environment>** The assembler package is a DOS-based application but may be used in the Windows environment by using Project Manager of Windows (included in the package).
	Part number: μS××××RA78K0S
CC78K0S C compiler package	Program that converts program written in C language into object codes that can be executed by microcontroller. Used in combination with optional assembler package (RA78K0S) and device file (DF789177). **<Caution when used in PC environment>** The C compiler package is a DOS-based application but may be used in the Windows environment by using Project Manager of Windows (included in the package).
	Part number: μS××××CC78K0S
DF789177[Note] Device file	File containing the information inherent to the device. Used in combination with other optional tools (RA78K0S, CC78K0S, SM78K0S). The corresponding operating system and host machine depend on the tools they are combined with.
	Part number: μS××××DF789177
CC78K0S-L C compiler source file	Source file of functions constituting object library included in C compiler package. Necessary for changing object library included in C compiler according to customer's specifications. Since this is the source file, its working environment does not depend on any particular operating system.
	Part number: μS××××CC78K0S-L

Note DF789177 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××××DF789177
μS××××CC78K0S-L

××××	Host Machine	OS	Supply Media
AA13	PC-9800 series	Japanese Windows[Note]	3.5” 2HD FD
AB13	IBM PC/AT™ and compatibles	Japanese Windows[Note]	3.5” 2HC FD
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 number: FL-PR3, PG-FP3) Flash programmer	Flash programmer dedicated to the microcontrollers incorporating a flash memory.
★ FA-44GB-8ES FA-48GA Flash memory writing adapter	These are flash memory writing adapter and connected to Flashpro III. FA-44GB-8ES: For 44-pin plastic LQFP (GB-8ES type) FA-48GA: For 48-pin plastic TQFP (GA-9EU type)

Remark FL-PR3, FA-44GB-8ES, and FA-48GA are products manufactured by Naito Densei Machida Mfg. Co., Ltd.
(+81-44-822-3813). Consult Naito Densei Machida Mfg. Co., Ltd. for purchase.

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	This is the adapter for supplying power from 100 to 240 VAC outlet.
IE-70000-98-IF-C Interface adapter	This adapter is needed when PC-9800 series (excluding notebook models) is used as a host machine of IE-78K0S-NS (supports C bus).
IE-70000-CD-IF-A PC card interface	This PC card and interface cable are needed when a notebook-type personal computer is used as a host machine of IE-78K0S-NS (supports PCMCIA socket).
IE-70000-PC-IF-C Interface adapter	This adapter is needed when IBM PC/AT and compatibles are used as a host machine of IE-78K0S-NS (supports ISA bus).
IE-70000-PC-IF Interface adapter	This adapter is needed when connecting a personal computer that includes a PCI bus as the host machine of the IE-78K0S-NS.
IE-789177-NS-EM1 Emulation board	Emulation board for emulating the peripheral hardware inherent to the device. Used in combination with in-circuit emulator.
NP-44GB Emulation probe	Board for connecting the in-circuit emulator and target system. Used in combination with the EV-9200G-44.
EV-9200G-44 Conversion socket	This conversion socket connects the NP-44GB to the target system board designed to mount a 44-pin plastic LQFP (GB-8ES type).
NP-44GB-TQ Emulation probe	Board for connecting the in-circuit emulator and target system. Used in combination with the TGB-044SAP.
TGB-044SAP Conversion adapter	This conversion socket connects the NP-44GB-TQ to the target system board designed to mount a 44-pin plastic LQFP (GB-8ES type).
NP-48GA Emulation probe	Board for connecting the in-circuit emulator and target system. Used in combination with the TGA-048SDP.
TGA-048SDP Conversion adapter	This conversion socket connects the NP-48GA to the target system board designed to mount a 48-pin plastic TQFP (fine pitch) (GA-9EU type).

Remarks 1. NP-44GB, NP-44GB-TQ, and NP-48GA are products manufactured by Naito Densei Machida Mfg. Co., Ltd. (+81-44-822-3813). Consult Naito Densei Machida Mfg. Co., Ltd. for purchase.

2. TGB-044SAP and TGA-048SDP are products manufactured by Tokyo Eletech Corporation. Consult Daimaru Kogyo, Ltd. for more information.

Daimaru Kogyo, Ltd. Tokyo Electronics Division (+81-3-3820-7112)
Osaka Electronics Division (+81-6-6244-6672)

A.3.2 Software

ID78K0S-NS Integrated debugger (Supports in-circuit emulator IE-78K0S-NS)	Control program for debugging the 78K/0S Series. This program provides a graphical user interface. It runs on Windows for personal computer users and on OSF/Motif™ for engineering work station users, and has visual designs and operationability that comply with these operating systems. In addition, it has a powerful debug function that supports C language. Therefore, trace results can be displayed at a C language level by the window integration function that links 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 a device file (DF789177) (sold separately).
	Part number: μS××××ID78K0S-NS

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

μS××××ID78K0S-NS

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

Note Also operates in 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 on Windows. By using SM78K0S, the logic and performance of an application can be verified independently of hardware development even when the in-circuit emulator is not used. This enhances development efficiency and improves software quality. Used in combination with a device file (DF789177) (sold separately).
	Part number: μS××××SM78K0S
DF789177[Note] Device file	Files including information peculiar to devices. Used in combination with RA78K0S, CC78K0S, or SM78K0S (sold separately).
	Part number: μS××××DF789177

Note DF789177 can be used commonly with RA78K0S, CC78K0S, and SM78K0S.

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

μS××××SM78K0S

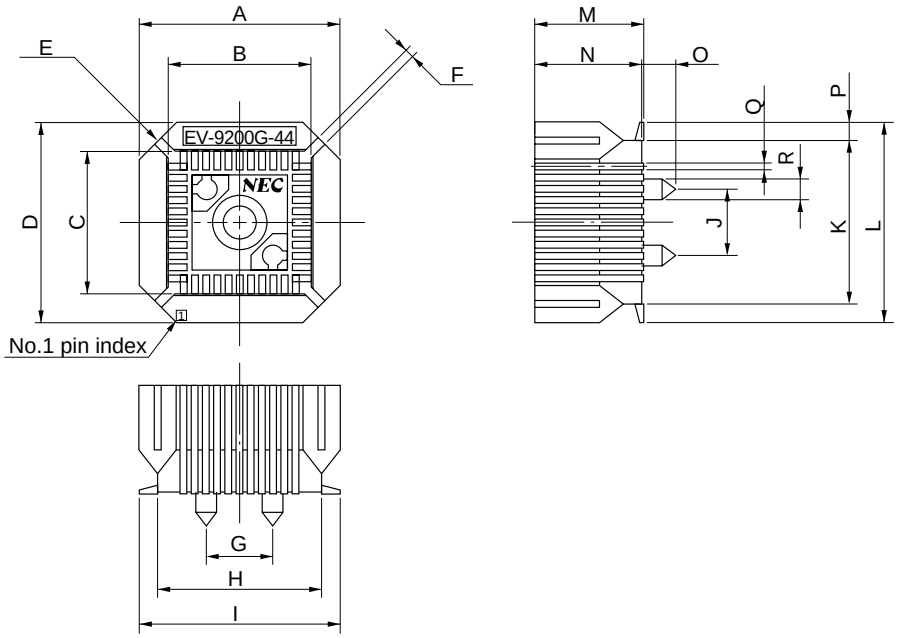
××××	Host Machine	OS	Supply Media
AA13	PC-9800 series	Japanese Windows[Note]	3.5" 2HD FD
AB13	IBM PC/AT and compatibles	Japanese Windows[Note]	3.5" 2HC FD
BB13		English Windows[Note]	

Note Also operates in the DOS environment.

A.4 Conversion Socket (EV-9200G-44) Drawing and Recommended Footprint

Figure A-2. EV-9200G-44 Drawing (for reference) (unit: mm)

Based on EV-9200G-44
(1) Package drawing (in mm)

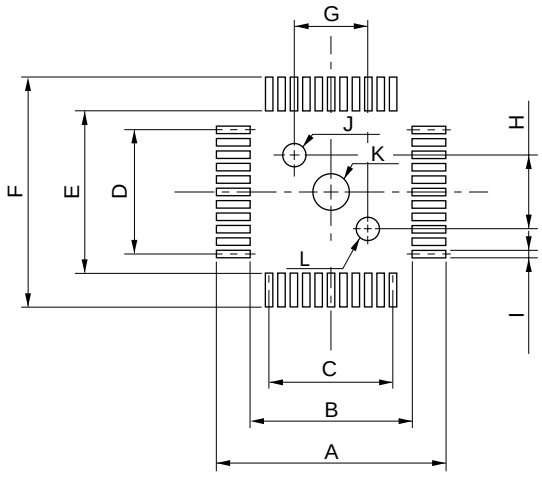


EV-9200G-44-G0E

ITEM	MILLIMETERS	INCHES
A	15.0	0.591
B	10.3	0.406
C	10.3	0.406
D	15.0	0.591
E	4-C 3.0	4-C 0.118
F	0.8	0.031
G	5.0	0.197
H	12.0	0.472
I	14.7	0.579
J	5.0	0.197
K	12.0	0.472
L	14.7	0.579
M	8.0	0.315
N	7.8	0.307
O	2.0	0.079
P	1.35	0.053
Q	0.35±0.1	$0.014^{+0.004}_{-0.005}$
R	ϕ1.5	ϕ0.059

Figure A-3. EV-9200G-44 Recommended Footprints (for reference) (unit: mm)

Based on EV-9200G-44
(2) Pad drawing (in mm)



EV-9200G-44-P1E

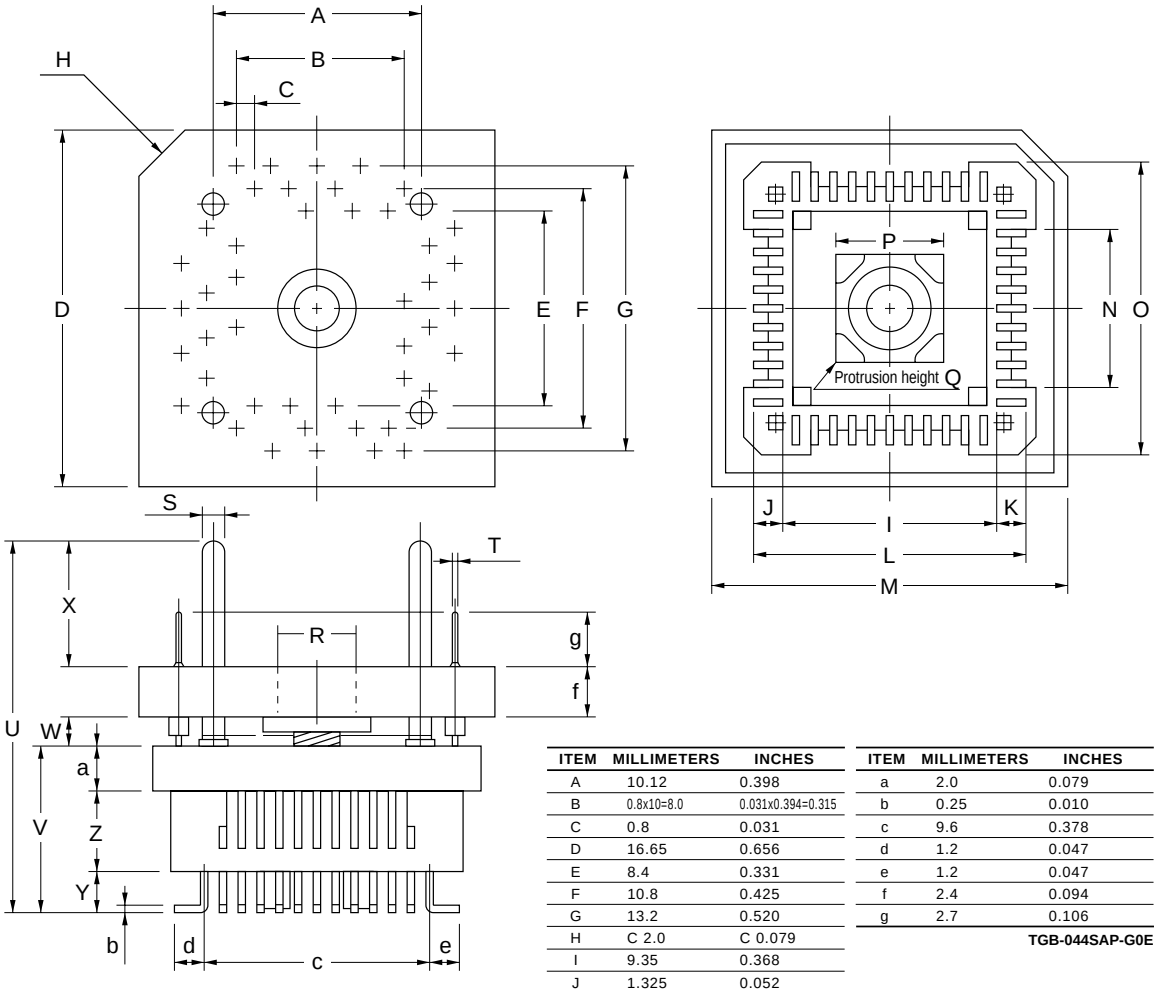
ITEM	MILLIMETERS	INCHES
A	15.7	0.618
B	11.0	0.433
C	$0.8\pm0.02\times10=8.0\pm0.05$	$0.031^{+0.002}_{-0.001}\times0.394=0.315^{+0.002}_{-0.002}$
D	$0.8\pm0.02\times10=8.0\pm0.05$	$0.031^{+0.002}_{-0.001}\times0.394=0.315^{+0.002}_{-0.002}$
E	11.0	0.433
F	15.7	0.618
G	5.00 ± 0.08	$0.197^{+0.003}_{-0.004}$
H	5.00 ± 0.08	$0.197^{+0.003}_{-0.004}$
I	0.5 ± 0.02	$0.02^{+0.001}_{-0.002}$
J	$\phi1.57\pm0.03$	$\phi0.062^{+0.001}_{-0.002}$
K	$\phi2.2\pm0.1$	$\phi0.087^{+0.004}_{-0.005}$
L	$\phi1.57\pm0.03$	$\phi0.062^{+0.001}_{-0.002}$

Caution Dimensions of mount pad for EV-9200 and that for target device (QFP) may be different in some parts. For the recommended mount pad dimensions for QFP, refer to "SEMICONDUCTOR DEVICE MOUNTING TECHNOLOGY MANUAL" (C10535E).

A.5 Conversion Adapter Drawing (TGB-044SAP)

Figure A-4. TGB-044SAP Drawing (for reference) (unit: mm)

Reference diagram: TGB-044SAP (TQPACK044SA+TQSOCKET044SAP) Package dimension (unit: mm)



ITEM	MILLIMETERS	INCHES
A	10.12	0.398
B	0.8x10=8.0	0.031x0.394=0.315
C	0.8	0.031
D	16.65	0.656
E	8.4	0.331
F	10.8	0.425
G	13.2	0.520
H	C 2.0	C 0.079
I	9.35	0.368
J	1.325	0.052
K	1.325	0.052
L	12.0	0.472
M	16.65	0.656
N	8.5	0.335
O	13.15	0.518
P	5.0	0.197
Q	1.8	0.071
R	ϕ3.55	ϕ0.140
S	ϕ0.9	ϕ0.035
T	ϕ0.3	ϕ0.012
U	(16.95)	(0.667)
V	7.35	0.289
W	1.2	0.047
X	6.0	0.236
Y	1.85	0.073
Z	3.5	0.138

ITEM	MILLIMETERS	INCHES
a	2.0	0.079
b	0.25	0.010
c	9.6	0.378
d	1.2	0.047
e	1.2	0.047
f	2.4	0.094
g	2.7	0.106

TGB-044SAP-G0E

note: Product by TOKYO ELETECH CORPORATION.

[MEMO]

APPENDIX B EMBEDDED SOFTWARE

The following embedded software products are available for efficient program development and maintenance of the μPD789167, 789177, 789167Y, and 789177Y Subseries.

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

Remark ×××× in the part number differs depending on the host machines and OS used.

μS××××MX78K0S

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

Note Also operates in the DOS environment.

[MEMO]

APPENDIX C REGISTER INDEX

C.1 Register Name Index

[A]

[B]

[E]

[I]

[M]

[O]

[P]

[R]

[S]

[T]

[W]

C.2 Register Symbol Index

[R]

[S]

[T]

[W]

★

APPENDIX D REVISION HISTORY

The revision history for this manual is detailed below. "Chapter" indicates the chapter of the edition.

Edition	Revision from Previous Edition	Chapter
Second edition	Addition of description of μPD789166Y, μPD789167Y, μPD789176Y, and μPD789177Y	Throughout
	Change of status of μPD789166, μPD789167, μPD789176, and μPD789177 from "under development" to "developed"	
	Addition of description of SMB0 special function registers to Table 5-3 Special Function Registers	CHAPTER 5 CPU ARCHITECTURE
	Modification of Figure 6-5 Block Diagram of P21	CHAPTER 6 PORT FUNCTIONS
	Addition of 8.5 Notes on Using 16-Bit Timer	CHAPTER 8 16-BIT TIMER
	Addition of 15 SMB0 (μPD789167Y AND 789177Y SUBSERIES)	CHAPTER 15 SMB0 (μPD789167Y AND 789177Y SUBSERIES)
	Addition of description of SMB0 interrupt to 17 INTERRUPT FUNCTIONS	CHAPTER 17 INTERRUPT FUNCTIONS
	Addition of Figure 20-3 Flashpro III Connection in SMB Mode	CHAPTER 20 μPD78F9177 AND μPD78F9177Y
	Addition of setting with SMB mode in Table 20-4 Setting with PG-FP3	
	Addition of development tools for μPD789166Y, μPD789167Y, μPD789176Y, and μPD789177Y	APPENDIX A DEVELOPMENT TOOLS

[MEMO]

NEC

Facsimile Message

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.

From:

Name

Company

Tel. FAX

Address

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: +55-11-6465-6829

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: 044-548-7900

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	❑	❑	❑	❑