NEC

User's Manual

μPD789026 Subseries

8-Bit Single-Chip Microcontrollers

μPD789022
μPD789024
μPD789025
μPD789026
μPD78F9026A

Document No. U11919EJ3V0UMJ1 (3rd edition)
Date Published October 2000 N 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: μPD78F9026A

The customer must judge the need for license: μPD789022, μPD789024, μPD789025, μPD789026

Major Revision in This Edition

Page	Description
Throughout	Completion of development of μPD789022 and μPD789024
	Change of part number from μPD78F9026 to μPD78F9026A
	Deletion of following products: μPD789022CU-×××, 789024CU-×××
	Addition of GB-8ES type package to all models
p.39	Change of circuit type and recommended connection of unused pins in **Table 2-1**
p.99	Addition of cautions on rewriting CR20 to **Section 6.4.1**
p.106	Addition of cautions on rewriting CR00 to **Section 7.2 (1)**
p.109	Addition of description of operation to **Section 7.4.1**
p.111	Addition of description of operation to **Section 7.4.2**
p.112	Addition of description of operation to **Section 7.4.3**
pp.180 to 183	Change of flash writer from Flashpro II to Flashpro III
p.183	Addition of setting example to **Section 13.1.4**
p.205	Addition of part number of MX78K0S to **Appendix B**

The mark ★ shows the major revised points.

Regional Information

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

J00.7

[MEMO]

INTRODUCTION

Readers	This manual is intended for user engineers who understand the functions of the μPD789026 Subseries to design and develop its application systems and programs. The target subseries is the μPD789026 Subseries, which consists of the μPD789022, μPD789024, μPD789025, μPD789026, and μPD78F9026A.
Purpose	This manual is designed to deepen your understanding of the following functions described in the following organization.
Organization	Two manuals are available for the μPD789026 Subseries: this manual and Instruction Manual (common to the 78K/0S Series).

μPD789026 Subseries User's 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 μPD789026 Subseries
→ Read this manual in the order of the **TABLE OF CONTENTS**.

◊ How to read register formats
→ The name of a bit whose number is encircled is reserved for the assembler and is defined for the C compiler by the header file sfrbit.h.

◊ To learn the detailed functions of a register whose register name is known
→ See **APPENDEX C**.

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

Legend

Data significance	:	Left: higher digit, right: lower digit
Active low	:	$\overline{\times\times\times}$ (top bar over pin or signal name)
Note	:	Description of text marked **Note**
Caution	:	Important information
Remark	:	Supplement
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.

Device Related Documents

Document Name	Document No.	
	English	Japanese
μPD789022, 789024, 789025, 789026 Data Sheet	U11715E	U11715J
μPD78F9026A Data Sheet	U14356E	U14356J
μPD789026 Subseries User's Manual	This manual	U11919J
78K/0S Series User's Manual — Instruction	U11047E	U11047J

Documents for Development Tool (User's Manual)

Document Name		Document No.	
		English	Japanese
RA78K0S Assembler Package	Operation	U11622E	U11622J
	Assembly Language	U11599E	U11599J
	Structured Assembly Language	U11623E	U11623J
CC78K0S C Compiler	Operation	U11816E	U11816J
	Language	U11817E	U11817J
SM78K0S System Simulator Windows™ Based	Reference	U11489E	U11489J
SM78K Series System Simulator	External Part User Open Interface Specifications	U10092E	U10092J
ID78K0S Integrated Debugger Windows Based	Reference	U12901E	U12901J
IE-78K0S-NS In-Circuit Emulator		U13549E	U13549J
IE-789026-NS-EM1 Emulation Board		To be created	To be created

Document for Embedded Software (User's Manual)

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

Caution The related documents listed above are subject to change without notice. Be sure to use the latest documents for designing, etc.

Other Related Documents

Document Name	Document No.	
	English	Japanese
SEMICONDUCTORS SELECTION GUIDE Products & 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 Microcomputer: Other Companies	–	U11416J

Caution The related documents listed above are subject to change without notice. Be sure to use the latest documents for designing, etc.

[MEMO]

TABLE OF CONTENTS

LIST OF FIGURES (1/3)

LIST OF FIGURES (2/3)

LIST OF FIGURES (3/3)

LIST OF TABLES (1/2)

LIST OF TABLES (2/2)

[MEMO]

CHAPTER 1 GENERAL

1.1 Features

• ROM and RAM capacity

Part Number \ Item	Program Memory		Data Memory
μPD789022	ROM	4 Kbytes	256 bytes
μPD789024		8 Kbytes	
μPD789025		12 Kbytes	512 bytes
μPD789026		16 Kbytes	
★ μPD78F9026A	Flash memory	16 Kbytes	

- Variable minimum instruction execution time - from high speed (0.4 μs: with 5.0-MHz system clock) to slow (1.6 μs: with 5.0-MHz system clock)
- I/O port: 34 lines
- Serial interface: 1 channel
  3-wire serial I/O mode/UART mode selection
- Timer: 3 channels
  - 16-bit timer : 1 channel
  - 8-bit timer/event counter : 1 channel
  - Watchdog timer : 1 channel
- Vectored interrupt: 10
- Supply voltage: V_{DD} = 1.8 to 5.5 V
- Operating ambient temperature: T_A = -40°C to +85°C

1.2 Applications

Home appliances, car accessories, air conditioners, game machines, etc.

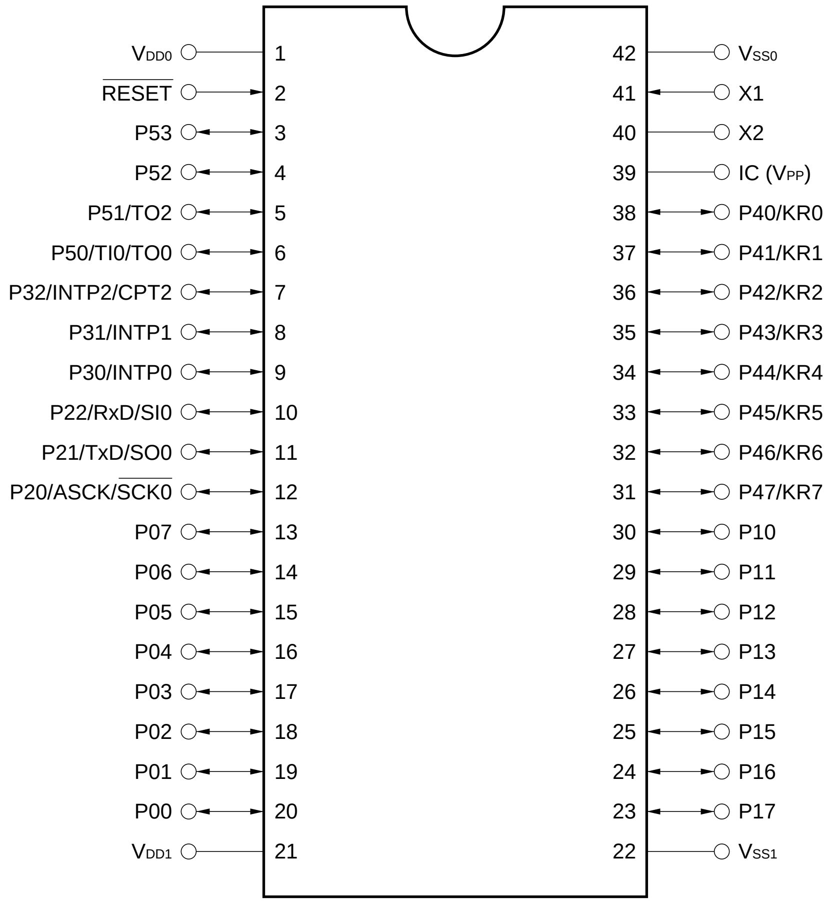
★ 1.3 Ordering Information

Part Number	Package	Internal ROM
μPD789022GB-×××-3BS-MTX	44-pin plastic QFP (10 × 10 mm, resin thickness 2.7 mm)	Mask ROM
μPD789022GB-×××-8ES	44-pin plastic LQFP (10 × 10 mm, resin thickness 1.4 mm)	Mask ROM
μPD789024GB-×××-3BS-MTX	44-pin plastic QFP (10 × 10 mm, resin thickness 2.7 mm)	Mask ROM
μPD789024GB-×××-8ES	44-pin plastic LQFP (10 × 10 mm, resin thickness 1.4 mm)	Mask ROM
μPD789025CU-×××	42-pin plastic shrink DIP (600 mil)	Mask ROM
μPD789025GB-×××-3BS-MTX	44-pin plastic QFP (10 × 10 mm, resin thickness 2.7 mm)	Mask ROM
μPD789025GB-×××-8ES	44-pin plastic LQFP (10 × 10 mm, resin thickness 1.4 mm)	Mask ROM
μPD789026CU-×××	42-pin plastic shrink DIP (600 mil)	Mask ROM
μPD789026GB-×××-3BS-MTX	44-pin plastic QFP (10 × 10 mm, resin thickness 2.7 mm)	Mask ROM
μPD789026GB-×××-8ES	44-pin plastic LQFP (10 × 10 mm, resin thickness 1.4 mm)	Mask ROM
μPD78F9026ACU	42-pin plastic shrink DIP (600 mil)	Flash memory
μPD78F9026AGB-3BS-MTX	44-pin plastic QFP (10 × 10 mm, resin thickness 2.7 mm)	Flash memory
μPD78F9026AGB-8ES	44-pin plastic LQFP (10 × 10 mm, resin thickness 1.4 mm)	Flash memory

Remark ××× indicates ROM code suffix.

1.4 Pin Configuration (Top View)

- **42-pin plastic shrink DIP (600 mil)**
  μPD789025CU-×××
  μPD789026CU-×××
  μPD78F9026ACU



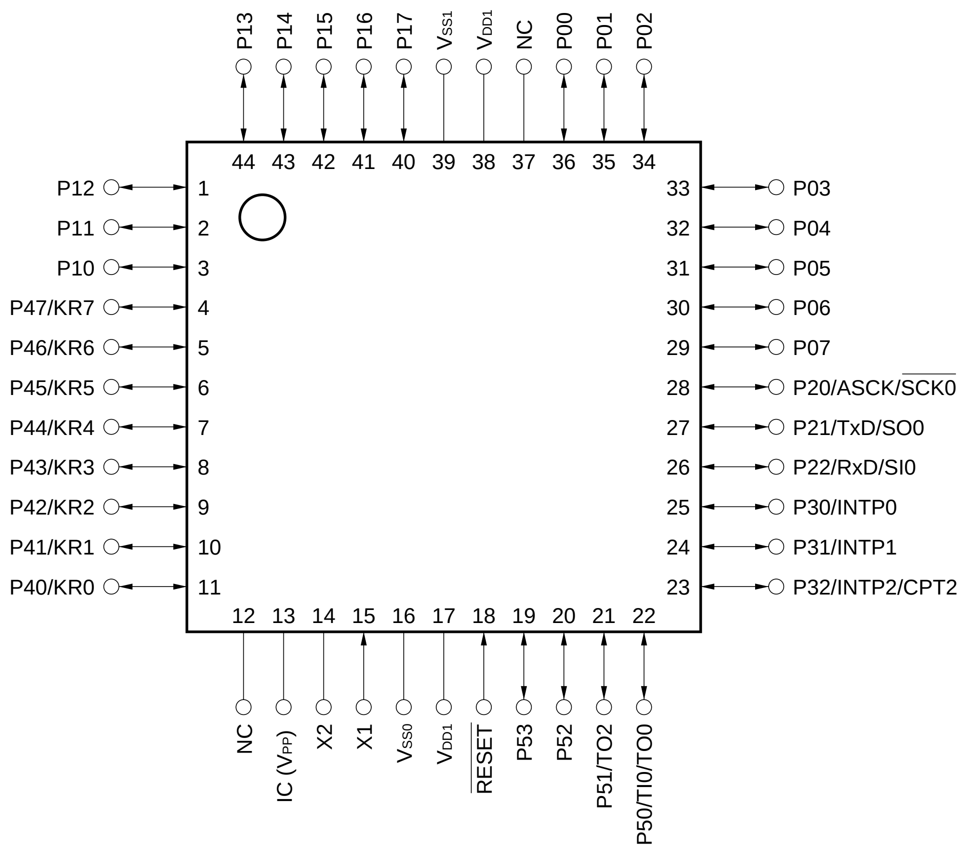
Caution Connect the IC pin directly to V_{SS0} or V_{SS1}.

Remark An item in parentheses applies to the μPD78F9026A only.

★ • **44-pin plastic QFP (10 × 10 mm, resin thickness 2.7 mm)**
μPD789022GB-×××-3BS-MTX
μPD789024GB-×××-3BS-MTX
μPD789025GB-×××-3BS-MTX
μPD789026GB-×××-3BS-MTX
μPD78F9026AGB-3BS-MTX

★ • **44-pin plastic LQFP (10 × 10 mm, resin thickness 1.4 mm)**
μPD789022GB-×××-8ES
μPD789024GB-×××-8ES
μPD789025GB-×××-8ES
μPD789026GB-×××-8ES
μPD78F9026AGB-8ES

★



★

Caution Connect the IC pin directly to V_{SS0} or V_{SS1}.

Remark An item in parentheses applies to the μPD78F9026A only.

ASCK	: Asynchronous Serial Clock	$\overline{\text{RESET}}$	: Reset
CPT2	: Capture Trigger Input	RxD	: Receive Data
IC	: Internally Connected	$\overline{\text{SCK0}}$	: Serial Clock
INTP0 to INTP2	: Interrupt from Peripherals	SI0	: Serial Input
KR0 to KR7	: Key Return	SO0	: Serial Output
NC	: Non-connection	TI0	: Timer Input
P00 to P07	: Port 0	TO0, TO2	: Timer Output
P10 to P17	: Port 1	TxD	: Transmit Data
P20 to P22	: Port 2	V_{DD0}, V_{DD1}	: Power Supply
P30 to P32	: Port 3	V_{PP}	: Programming Power Supply
P40 to P47	: Port 4	V_{SS0}, V_{SS1}	: Ground
P50 to P53	: Port 5	X1, X2	: Crystal

★ 1.5 Development of 78K/0S Series

The following shows the history of 78K/0S Series product development. Subseries names are shown inside frames.

In production

Under development

78K/0S Series

For small-scale, general-purpose applicationns

44-pin	μPD789046	Device developed by adding the subsystem clock to the μPD789026
42/44-pin	μPD789026	Device developed by enhancing the timers of the μPD789014 and expanding ROM and RAM
28-pin	μPD789014	With built-in UART bus and capable of low-voltage (1.8 V) operation

For small-scale, general-purpose applications and A/D function

44/48-pin	μPD789217AY	RC oscillation of the μPD789197AY
44/48-pin	μPD789197AY	With built-in EEPROM™ and SMB in the μPD789177
44-pin	μPD789177	Device developed by enhancing the A/D function of the μPD789167
44-pin	μPD789167	Device developed by enhancing the timers of the μPD789104A
30-pin	μPD789156	Device developed by enhancing the A/D function of the μPD789146
30-pin	μPD789146	With built-in EEPROM in the μPD789104A
30-pin	μPD789134A	Device developed by enhancing the A/D function of the μPD789124A
30-pin	μPD789124A	RC oscillation of the μPD789104A
30-pin	μPD789114A	Device developed by enhancing the A/D function of the μPD789104A
30-pin	μPD789104A	Device developed by adding the A/D function and multiplier to the μPD789026

For inverter control

44-pin	μPD789842	With built-in inverter control circuit and UART bus

For LCD driving

88-pin	μPD789830	With built-in UART bus and dot LCD
80-pin	μPD789417A	Device developed by enhancing the A/D function of the μPD789407A
80-pin	μPD789407A	Device developed by enhancing the I/O of the μPD789457
64-pin	μPD789457	Device developed by enhancing the A/D function of the μPD789447
64-pin	μPD789447	RC oscillation of the μPD789427
64-pin	μPD789437	Device developed by enhancing the A/D funciton of the μPD789427
64-pin	μPD789427	Device developed by adding the A/D function of the μPD789306
64-pin	μPD789316	RC oscillation of the μPD789306
64-pin	μPD789306	Basic subseries for LCD driving

For ASSP

44-pin	μPD789800	Device for a PC keyboard, with a built-in USB function
44-pin	μPD789840	Device for a keypad, with a built-in POC
20-pin	μPD789861	RC oscillation of the μPD789860
20-pin	μPD789860	Device for a keyless entry, with built-in POC and key return circuit

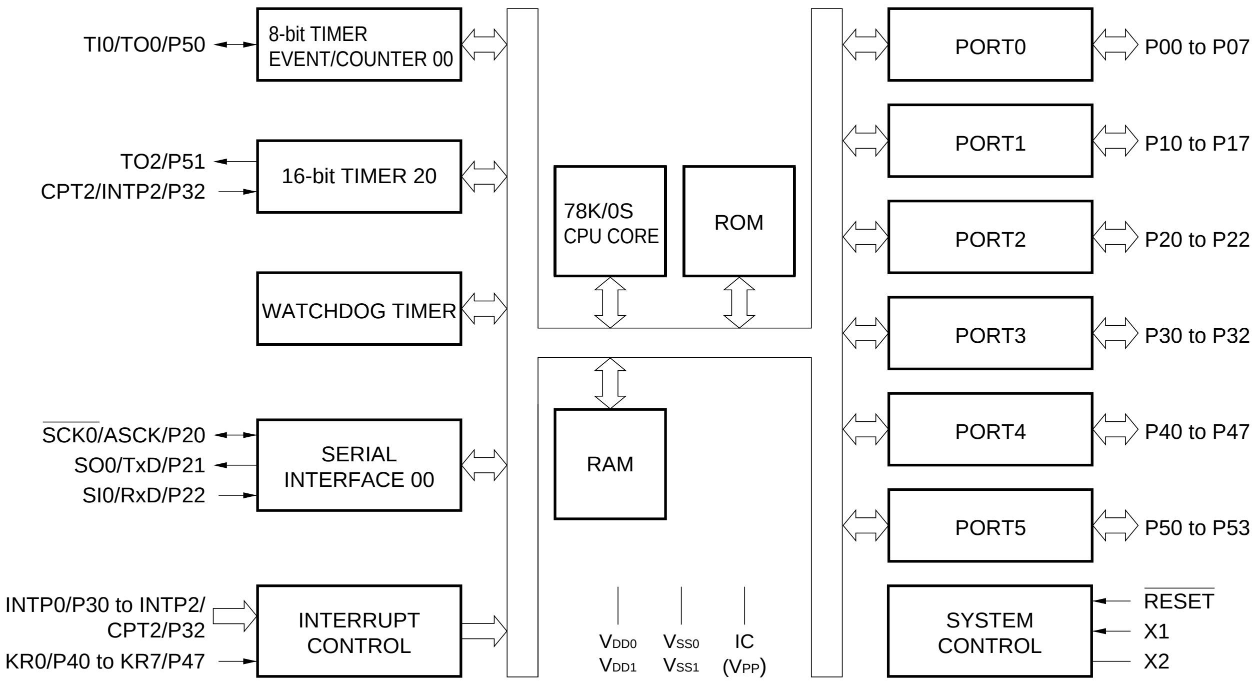
For IC card

5-pin	μPD789810	With built-in EEPROM and security circuit

The following lists the main functional differences between subseries products.

Subseries Name (Function)		ROM Capacity	Timer 8-Bit	Timer 16-Bit	Timer Watch	Timer WDT	8-Bit A/D	10-Bit A/D	Serial Interface	I/O	V_{DD} MIN. Value	Remark
General compact	μPD789046	16 K	1 ch	1 ch	1 ch	1 ch	–	–	1 ch (UART: 1 ch)	34	1.8 V	–
	μPD789026	4 K to 16 K			–							
	μPD789014	2 K to 4 K	2 ch	–						22		
General compact + A/D	μPD789217AY	16 K to 24 K	3 ch	1 ch	1 ch	1 ch	–	8 ch	2 ch [UART: 1 ch, SMB : 1 ch]	31	1.8 V	RC oscillation, EEPROM on chip
	μPD789197AY											EEPROM on chip
	μPD789177								1 ch (UART: 1 ch)			–
	μPD789167						8 ch	–				
	μPD789156	8 K to 16 K	1 ch		–		–	4 ch		20		EEPROM on chip
	μPD789146						4 ch	–				
	μPD789134A	2 K to 8 K					–	4 ch				RC oscillation
	μPD789124A						4 ch	–				
	μPD789114A						–	4 ch				–
	μPD789104A						4 ch	–				
Inverter control	μPD789842	8 K to 16 K	3 ch	**Note**	1 ch	1 ch	8 ch	–	1 ch (UART: 1 ch)	30	4.0 V	–
LCD drive	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	–		25		
	μPD789457	16 K to 24 K	2 ch				–	4 ch	2 ch (UART: 1 ch)			RC oscillation
	μPD789447						4 ch	–				
	μPD789437						–	4 ch				–
	μPD789427						4 ch	–				
	μPD789316	8 K to 16 K					–			23		RC oscillation
	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
	μPD789860											–
IC card	μPD789810	6 K	–	–	–	1 ch	–	–	–	1	2.7 V	EEPROM on chip

1.6 Block Diagram



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

2. An item in parentheses applies to the μPD78F9026A only.

1.7 Outline of Functions

Item		μPD789022	μPD789024	μPD789025	μPD789026	μPD78F9026A
Internal memory	ROM	Mask ROM				Flash Memory
		4 Kbytes	8 Kbytes	12 Kbytes	16 Kbytes	16 Kbytes
	High-speed RAM	256 bytes		512 bytes		
Minimum instruction execution time		0.4/1.6 μs (when operated at 5.0 MHz with system clock)				
Instruction set		• 16-bit operations • Bit manipulations (set, reset, test), etc.				
I/O port		Total : 34 • CMOS input/output : 34				
Serial interface		• 3-wire serial I/O mode/UART mode selectable: 1 channel				
Timer		• 16-bit timer : 1 channel • 8-bit timer/event counter : 1 channel • Watchdog timer : 1 channel				
Timer output		2				
Vectored interrupt source	Maskable	Internal: 5, External: 4				
	Non-maskable	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 QFP (10 × 10 mm, resin thickness 2.7 mm) • 44-pin plastic LQFP (10 × 10 mm, resin thickness 1.4 mm)		• 42-pin plastic shrink DIP (600 mil) • 44-pin plastic QFP (10 × 10 mm, resin thickness 2.7 mm) • 44-pin plastic LQFP (10 × 10 mm, resin thickness 1.4 mm)		

The outline of the timer is as follows.

		16-Bit Timer 20	8-Bit Timer/Event Counter 00	Watchdog Timer
Operating mode	Interval timer	–	1 channel	1 channel[Note]
	External event counter	–	1 channel	–
Function	Timer output	1 output	1 output	–
	Capture	1 input	–	–
	Interrupt source	1	1	1

Note Watchdog timer has a watchdog timer and interval timer functions. Select one of them.

[MEMO]

CHAPTER 2 PIN FUNCTIONS

2.1 List of Pin Functions

(1) Port pins

Pin Name	Input/Output	Function	After Reset	Alternate Function
P00 to P07	Input/output	Port 0 8-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	–
P10 to P17	Input/output	Port 1 8-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	–
P20 P21 P22	Input/output	Port 2 3-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	$\overline{\text{SCK0}}$/ASCK SO0/TxD SI0/RxD
P30 P31 P32	Input/output	Port 3 3-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	INTP0 INTP1 INTP2/CPT2
P40 to P47	Input/output	Port 4 8-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	KR0 to KR7
P50 P51 P52, P53	Input/output	Port 5 4-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	TI0/TO0 TO2 –

(2) Non-port pins

Pin Name	Input/Output	Function	After Reset	Alternate Function
INTP0	Input	External interrupt request input for which the active edge (rising edge, falling edge, or both) can be specified	Input	P30
INTP1				P31
INTP2				P32/CPT2
KR0 to KR7	Input	Key return signal detection	Input	P40 to P47
SI0	Input	3-wire serial interface serial data input	Input	P22/RxD
SO0	Output	3-wire serial interface serial data output	Input	P21/TxD
$\overline{\text{SCK0}}$	Input/output	3-wire serial interface serial clock input/output	Input	P20/ASCK
ASCK	Input	Asynchronous serial interface serial clock input	Input	P20/$\overline{\text{SCK0}}$
RxD	Input	Asynchronous serial interface serial data input	Input	P22/SI0
TxD	Output	Asynchronous serial interface serial data output	Input	P21/SO0
TO2	Output	16-bit timer (TM20) output	Input	P51
CPT2	Input	16-bit timer capture edge input	Input	P32/INTP2
TI0	Input	External count clock input to 8-bit timer (TM00)	Input	P50/TO0
TO0	Output	8-bit timer (TM00) output	Input	P50/TI0
X1	Input	System clock oscillation crystal connection	–	–
X2	–		–	–
$\overline{\text{RESET}}$	Input	System reset input	Input	–
NC	–	Not connected internally. Connect this pin directly to the V_{SS} pin (it can also be left open).	–	–
V_{DD0}	–	Positive power supply for ports	–	–
V_{DD1}	–	Positive power supply (except for ports)	–	–
V_{SS0}	–	Ground potential for ports	–	–
V_{SS1}	–	Ground potential (except for ports)	–	–
IC	–	Internally connected. Connect this pin directly to the V_{SS0} or V_{SS1} pin.	–	–
V_{PP}	–	Flash memory programming mode setting. Apply high voltage during program write/verify. Connect this pin directly to the V_{SS} pin in normal operating mode.	–	–

2.2 Description of Pin Functions

2.2.1 P00 to P07 (Port 0)

These pins constitute an 8-bit I/O port and can be set in the 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 in the pull-up resistor option register (PUO).

This port can drive LEDs directly.

2.2.2 P10 to P17 (Port 1)

These pins constitute an 8-bit I/O port. Can be set in the 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 in the pull-up resistor option register (PUO).

This port can drive LEDs directly.

2.2.3 P20 to P22 (Port 2)

These pins constitute a 3-bit I/O port. In addition, these pins provide the function to input/output the data and clock of the serial interface.

This port can drive LEDs directly.

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

(1) Port mode

In this mode, port 2 functions as a 3-bit I/O port. Port 2 can be set in the input or output mode in 1-bit units by using the port mode register 2 (PM2). When the port is used as an input port, an on-chip pull-up resistor can be used in the pull-up resistor option register (PUO).

(2) Control mode

In this mode, port 2 functions as the data input/output and the clock input/output of the serial interface.

(a) SI0, SO0

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

(b) $\overline{\text{SCK0}}$

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

(c) RxD, TxD

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

(d) ASCK

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

Caution When using port 2 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 9-2.

2.2.4 P30 to P32 (Port 3)

These pins constitute a 3-bit I/O port. In addition, they also function as external interrupt input and capture edge input.

This port can drive LEDs directly.

Port 3 can be specified in the following operation modes in bit-wise.

(1) Port mode

In this mode, port 3 functions as a 3-bit I/O port. Port 3 can be set in the input or output mode in 1-bit units by using the port mode register 3 (PM3). When the port is used as an input port, an on-chip pull-up resistor can be used in the pull-up resistor option register (PUO).

(2) Control mode

In this mode, port 3 functions as the external interrupt input.

(a) INTP0 to INTP2

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

(b) CPT2

This is a capture edge input pin.

2.2.5 P40 to P47 (Port 4)

These pins constitute an 8-bit I/O port. In addition, they also function as key return signal detection.

This port can drive LEDs directly.

Port 4 can be set in the following operation modes in bit-wise.

(1) Port mode

In this mode, port 4 functions as an 8-bit I/O port which can be set in the input or output mode in 1-bit units by using the port mode register 4 (PM4). When used as an input port, an on-chip pull-up resistor can be used in the pull-up resistor option register (PUO).

(2) Control mode

In this mode, the pins of port 4 can be used as key return signal detection pin (KR0 to KR7).

2.2.6 P50 to P53 (Port 5)

These pins constitute a 4-bit I/O port. In addition, these pins provide the function for performing input/output to/from the timer.

This port can drive LEDs directly.

Port 5 can be specified in the following operation modes in bit-wise.

(1) Port mode

In this mode, port 5 functions as a 4-bit I/O port. Port 5 can be set in the input or output mode in 1-bit units by using the port mode register 5 (PM5). When the port is used as an input port, an on-chip pull-up resistor can be used in the pull-up resistor option register (PUO).

(2) Control mode

In this mode, port 5 functions as the timer input/output.

(a) TI0

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

(b) TO0

This is an 8-bit timer output pin.

(C) TO2

This is a 16-bit timer output pin.

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

This pin inputs an active-low system reset signal.

2.2.8 X1, X2

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

To supply an external clock, input the clock to X1 and input the inverted signal to X2.

2.2.9 NC

The NC (Non-connection) pin is not connected internally. Connect this pin directly to the V_{SS} pin (it can also be left open).

2.2.10 V_{DD}

Positive power supply pins

2.2.11 V_{SS}

Ground potential pins

2.2.12 V_{PP} (μPD78F9026A only)

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

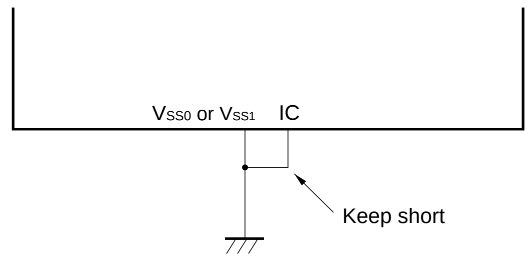
★ Directly connect this pin to the V_{SS0} or V_{SS1} pin in the normal operating mode.

★ 2.2.13 IC (mask ROM model only)

The IC (Internally Connected) pin is used to set the μPD789026 Subseries in the test mode in testing before shipment. In the normal operating mode, directly connect the IC pin to the V_{SS0} or V_{SS1} pin with as short a wire as possible.

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

- Connect the IC pin directly to the V_{SS0} or V_{SS1} pin.



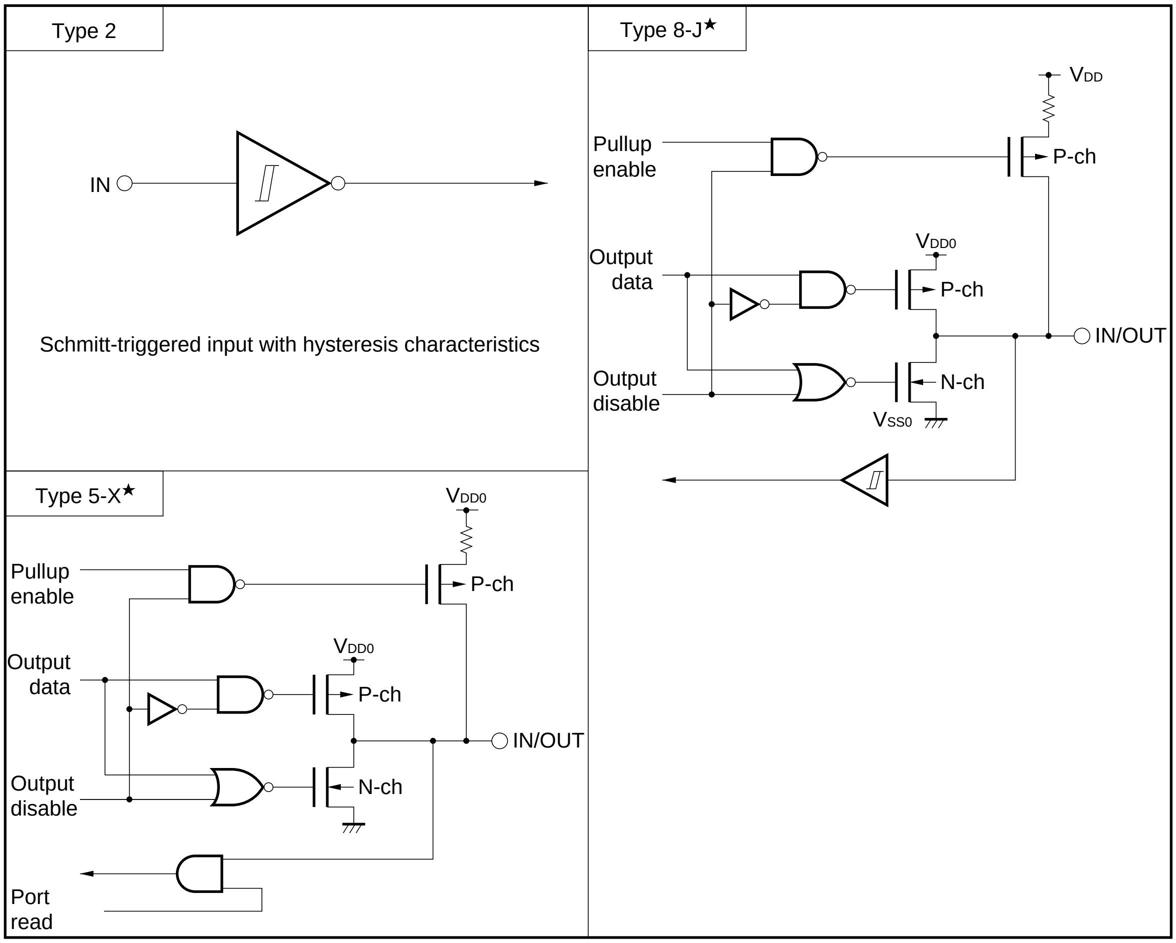
2.3 Pin Input/Output Circuits and Connection of Unused Pins

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

★ **Table 2-1. Type of Input/Output Circuit of Each Pin and Handling of Unused Pins**

Pin Name	I/O Circuit Type	Input/Output	Recommended Connection for Unused Pins
P00 to P07	5-X	Input/output	Input: Connect these pins to the V_{DD0}, V_{DD1}, V_{SS0}, or V_{SS1} pin via respective resistors. Output: Leave these pins open.
P10 to P17			
P20/ASCK/$\overline{SCK0}$	8-J		
P21/TxD/SO0	5-X		
P22/RxD/SI0	8-J		
P30/INTP0			
P31/INTP1			
P32/INTP2/CPT2			
P40/KR0 to P47/KR7			
P50/TI0/TO0			
P51/TO2	5-X		
P52, P53			
$\overline{RESET}$	2	Input	–
NC	–	–	Connect this pin directly to the V_{SS0} or V_{SS1} pin (possible to leave open).
IC (mask ROM model)			Connect these pins directly to the V_{SS0} or V_{SS1} pin.
V_{PP} (μPD78F9026A)			

Figure 2-1. List of Pin Input/Output Circuits



3.1 Memory Space

The μPD789026 Subseries can access 64 Kbytes of memory space. Figures 3-1 through 3-5 show the memory maps.

Figure 3-1. Memory Map (μPD789022)

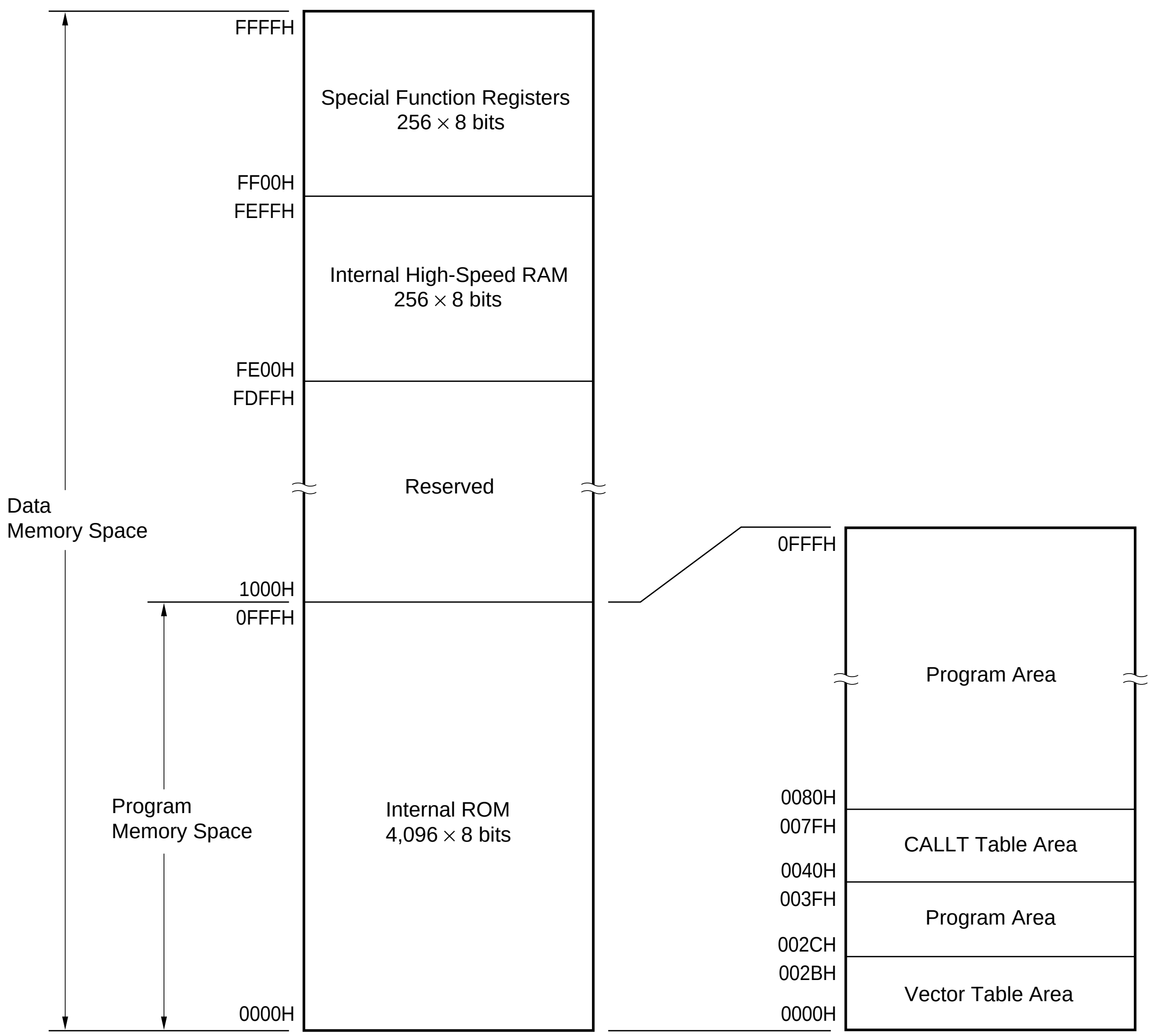


Figure 3-2. Memory Map (μPD789024)

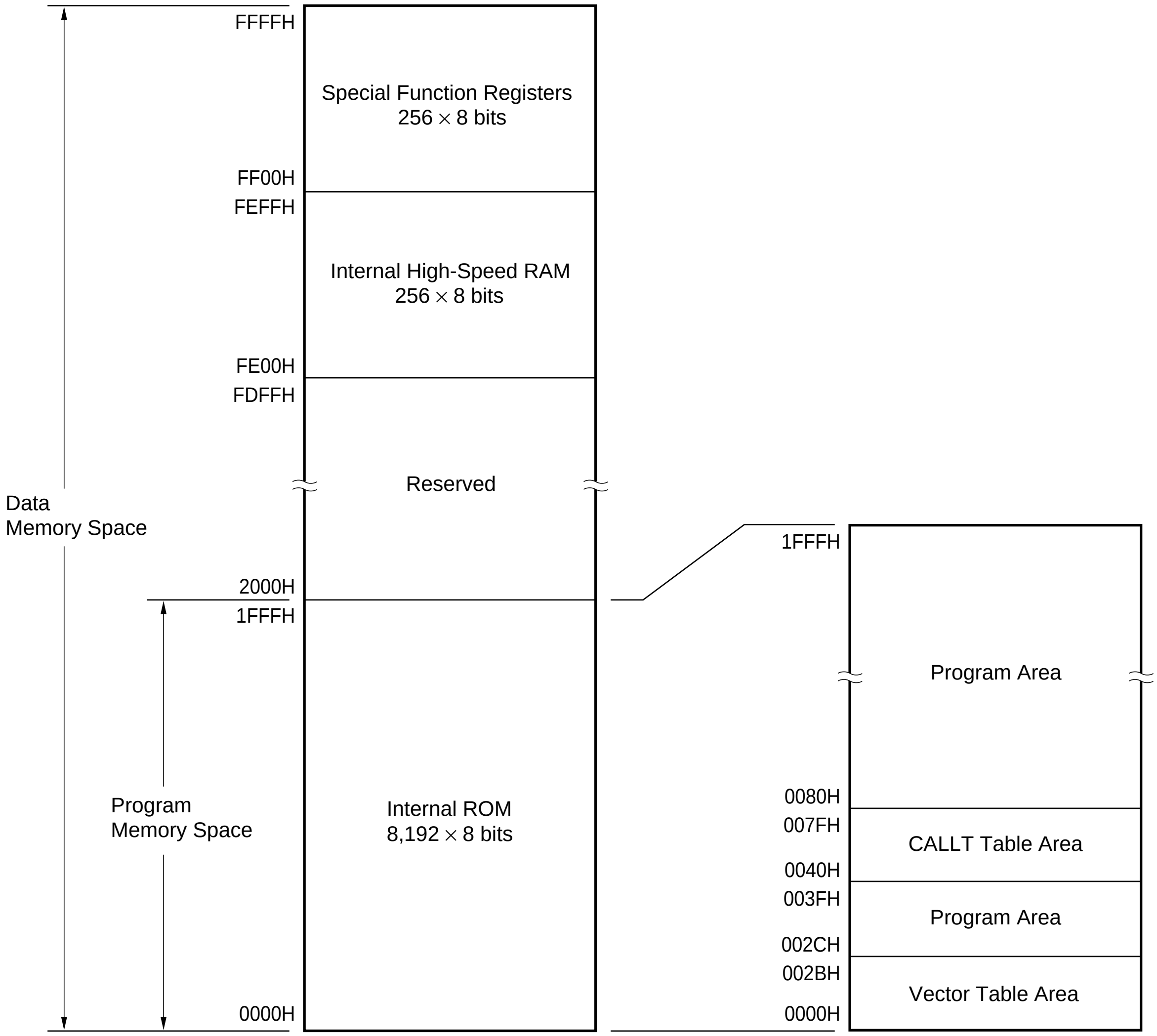


Figure 3-3. Memory Map (μPD789025)

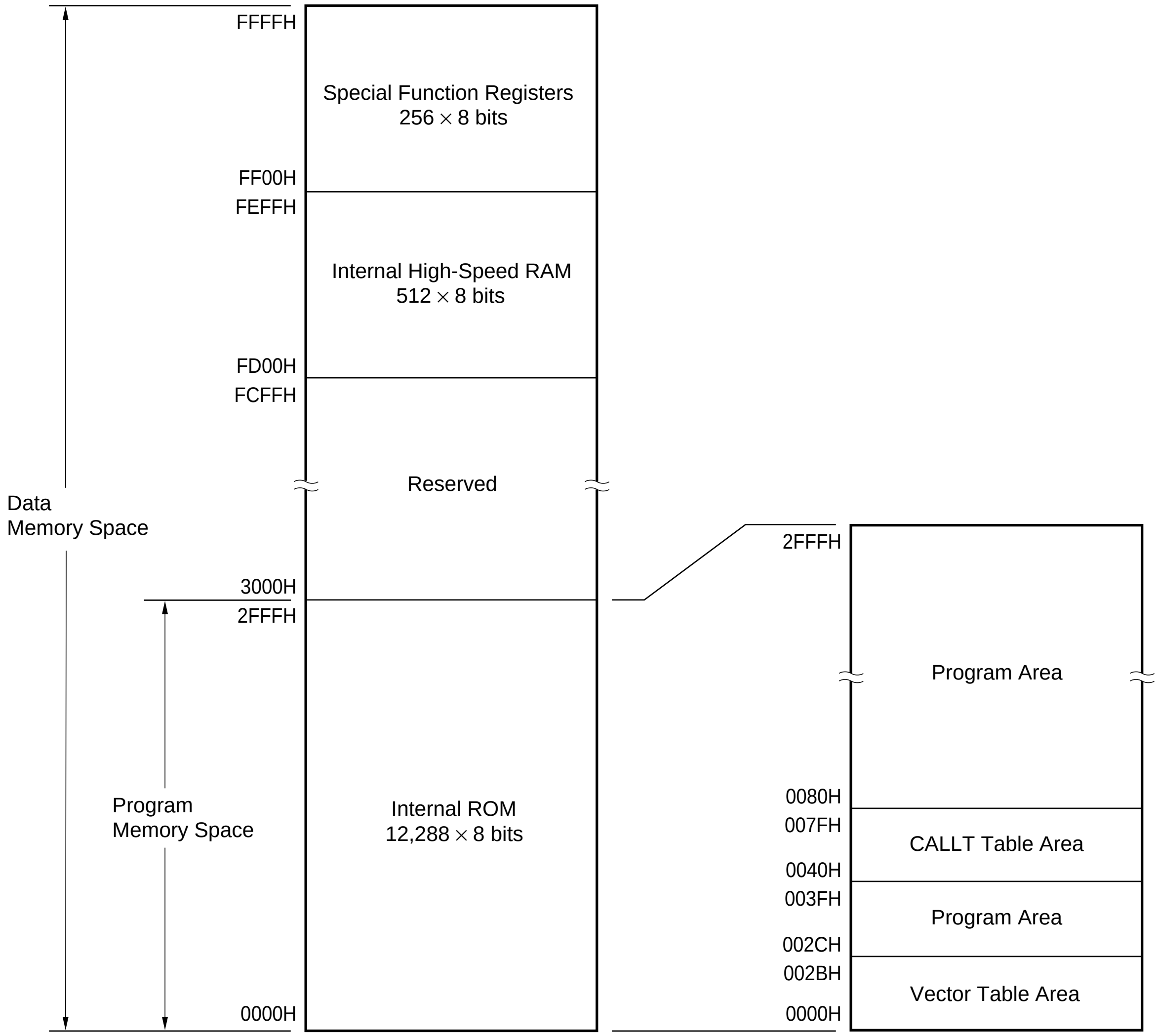


Figure 3-4. Memory Map (μPD789026)

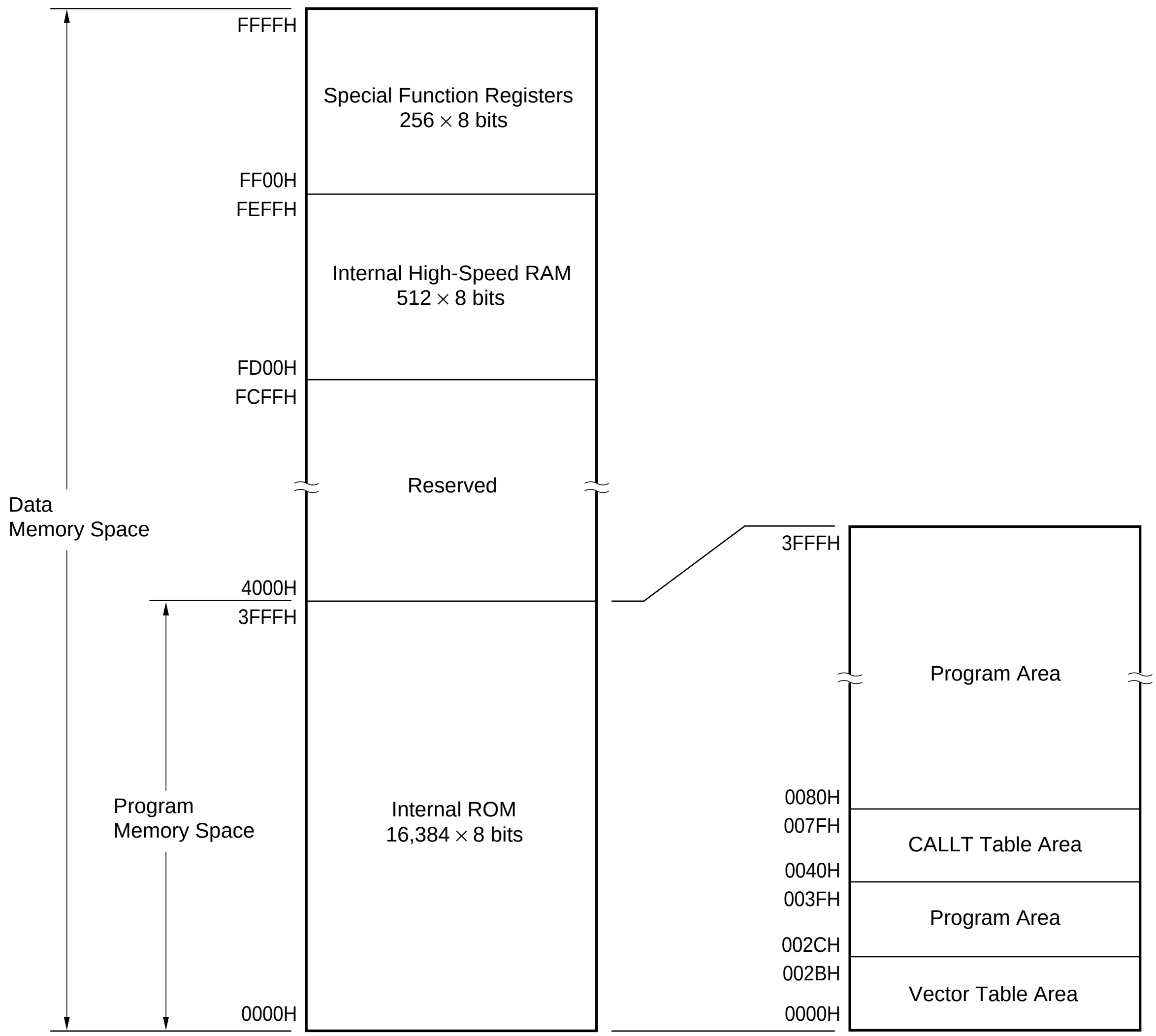
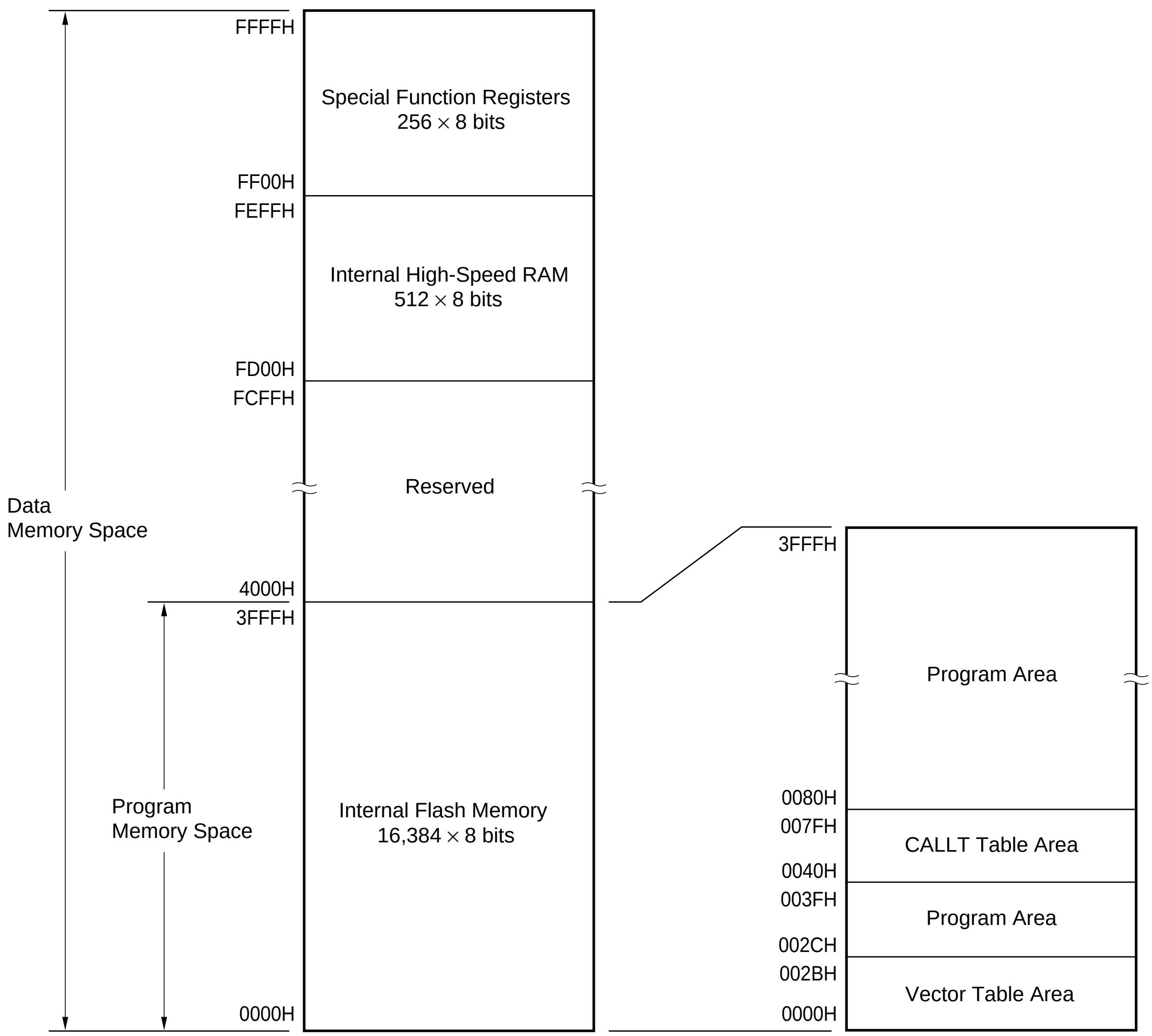


Figure 3-5. Memory Map (μPD78F9026A)



3.1.1 Internal program memory space

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

The μPD789026 Subseries provides the internal ROMs (or flash memory) containing the following capacities on each product.

Table 3-1. Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
μPD789022	Mask ROM	4,096 × 8 bits
μPD789024		8,192 × 8 bits
μPD789025		12,288 × 8 bits
μPD789026		16,384 × 8 bits
μPD78F9026A	Flash memory	16,384 × 8 bits

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

(1) Vector table area

A 44-byte area of addresses 0000H to 002BH 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 low-order 8 bits are stored in an even address, and the high-order 8 bits are stored in an odd address.

Table 3-2. Vector Table

Vector Table Address	Interrupt Request	Vector Table Address	Interrupt Request
0000H	$\overline{\text{RESET}}$ input	000CH	INTSR/INTCSI0
0004H	INTWDT	000EH	INTST
0006H	INTP0	0010H	INTTM0
0008H	INTP1	0014H	INTTM2
000AH	INTP2	002AH	INTKR

(2) CALLT instruction table area

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

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

The μPD789026 Subseries provides internal high-speed RAM containing the following capacity on each product.
The internal high-speed RAM can also be used as a stack memory.

Table 3-3. Internal High-Speed RAM Capacity

Part Number	Capacity
μPD789022	256 × 8 bits
μPD789024	
μPD789025	512 × 8 bits
μPD789026	
μPD78F9026A	

3.1.3 Special function register (SFR) area

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

3.1.4 Data memory addressing

The μPD789026 Subseries provides a variety of addressing modes which take account of memory manipulability, etc. Especially at address corresponding to data memory area (FE00H to FFFFH[Note 1], FD00H to FFFFH[Note 2]), particular addressing modes are possible to meet the functions of the special function registers (SFR) and other registers. Figures 3-6 through 3-10 show the data memory addressing modes.

Notes 1. With μPD789022 or μPD789024

2. With μPD789025, μPD789026, or μPD78F9026A

Figure 3-6. Data Memory Addressing (μPD789022)

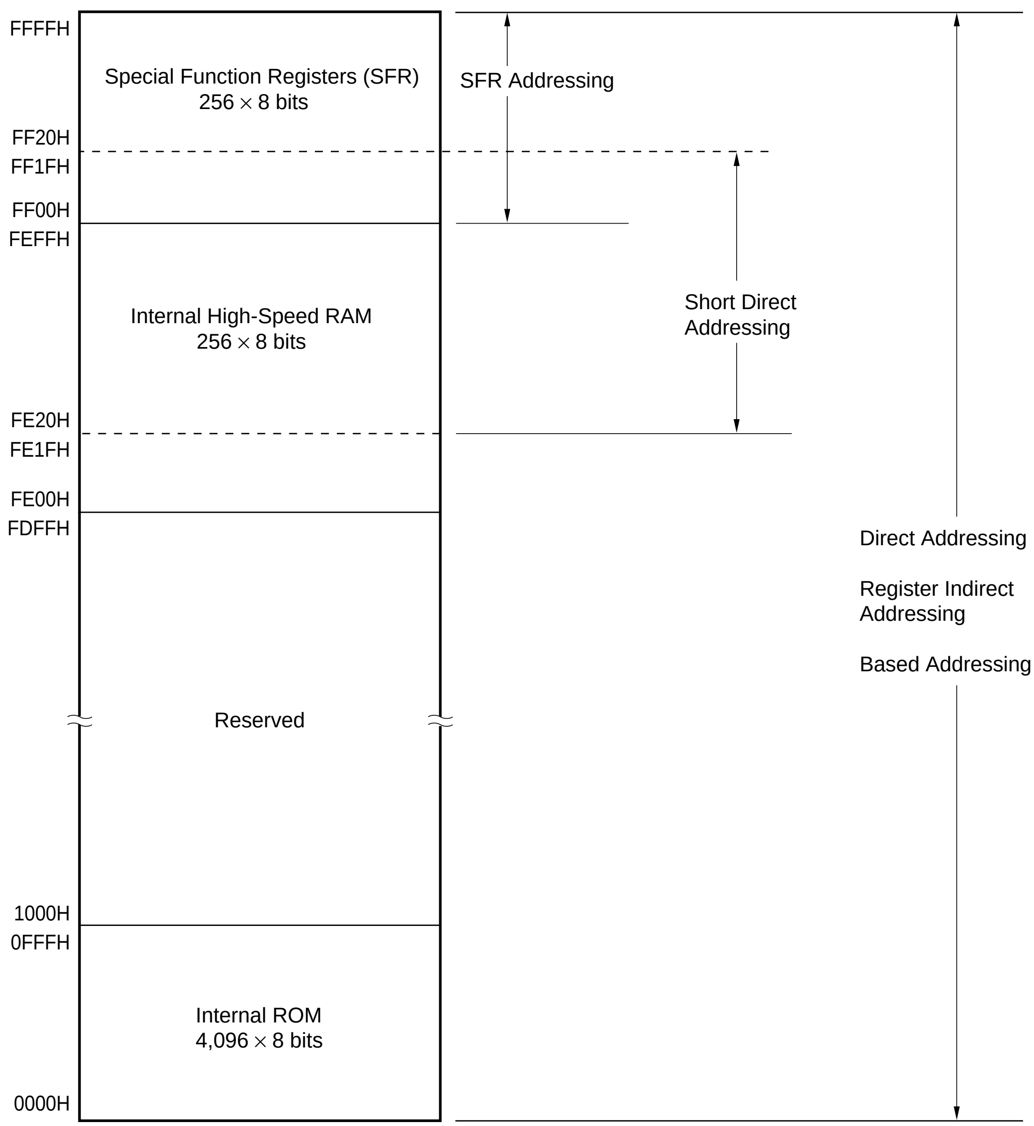


Figure 3-7. Data Memory Addressing (μPD789024)

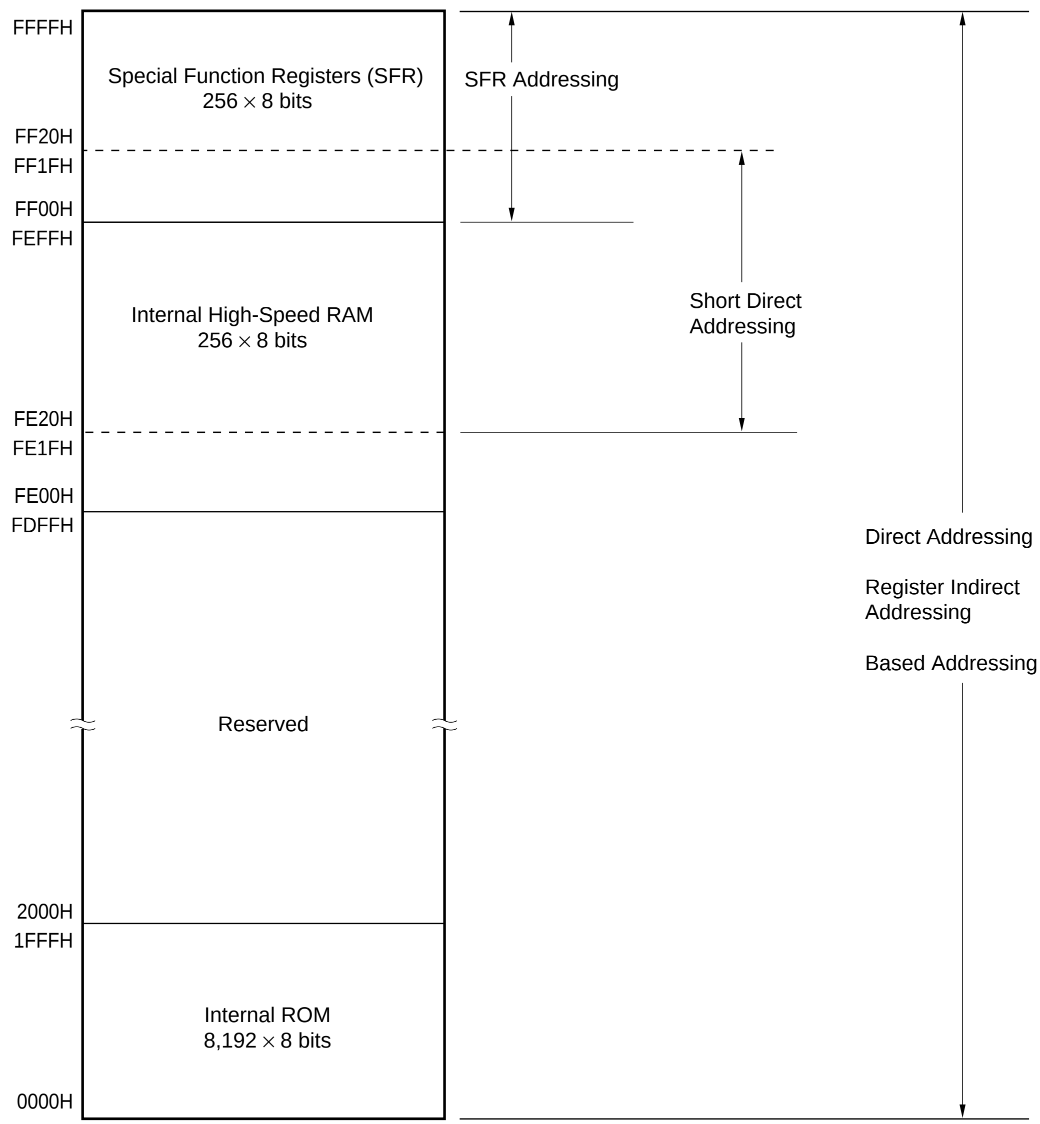


Figure 3-8. Data Memory Addressing (μPD789025)

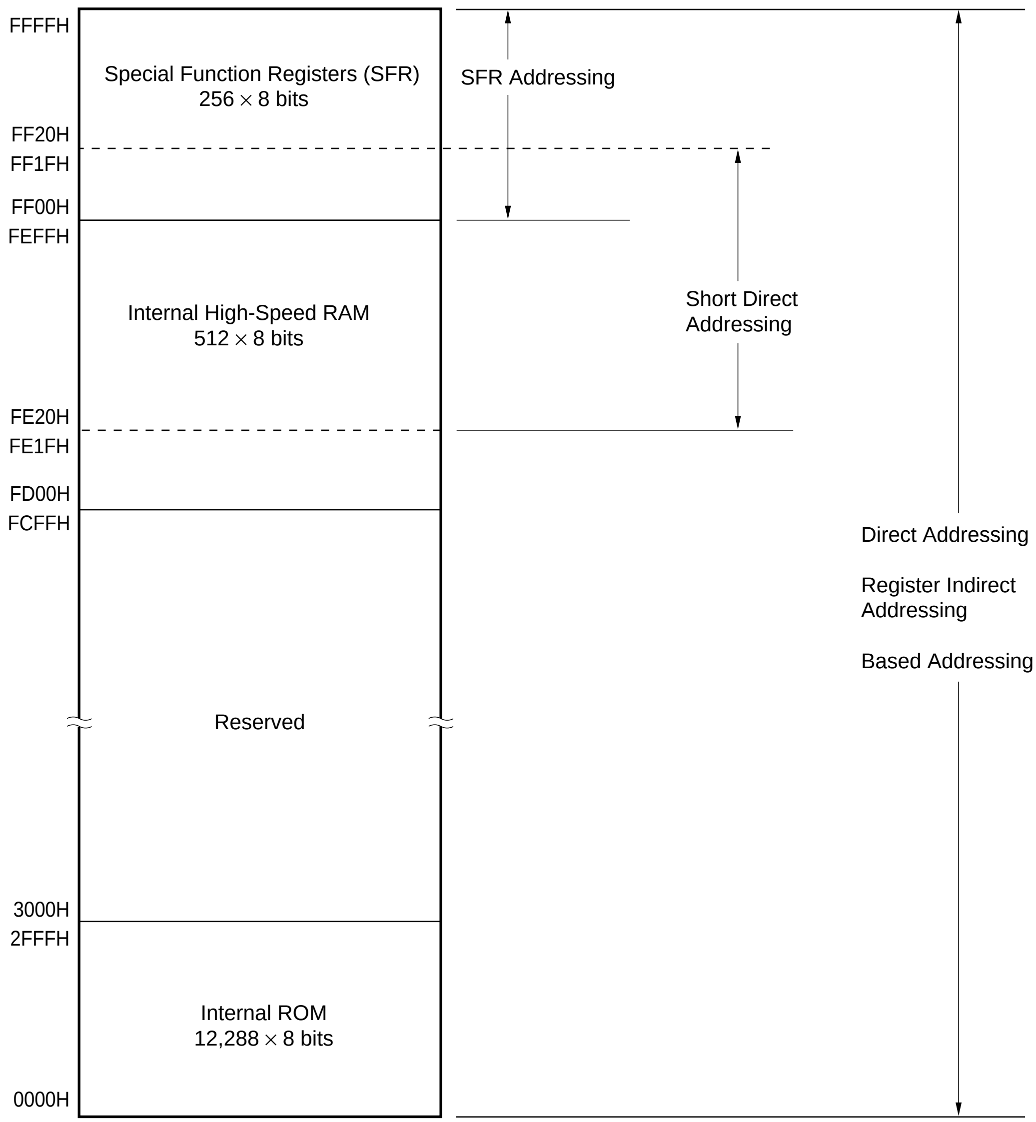


Figure 3-9. Data Memory Addressing (μPD789026)

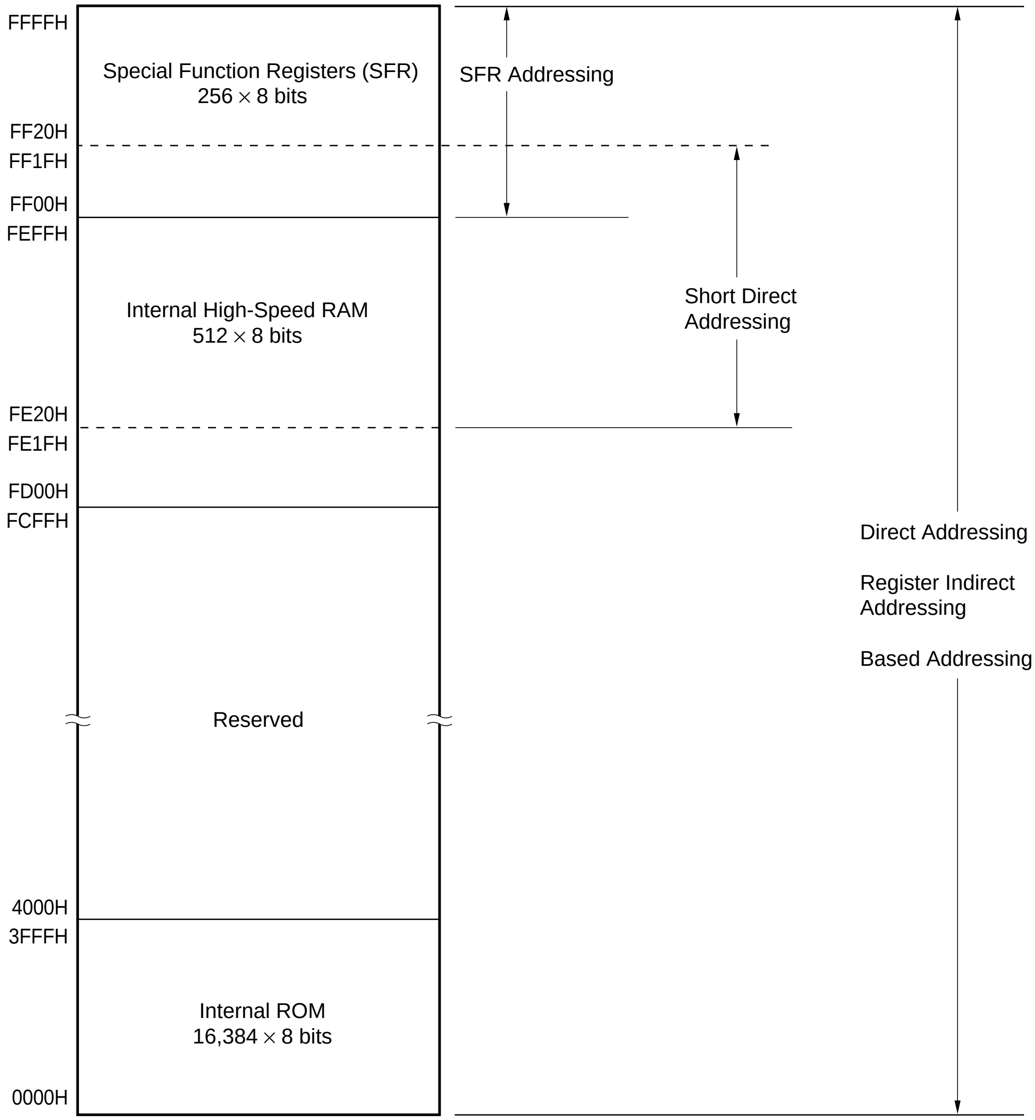
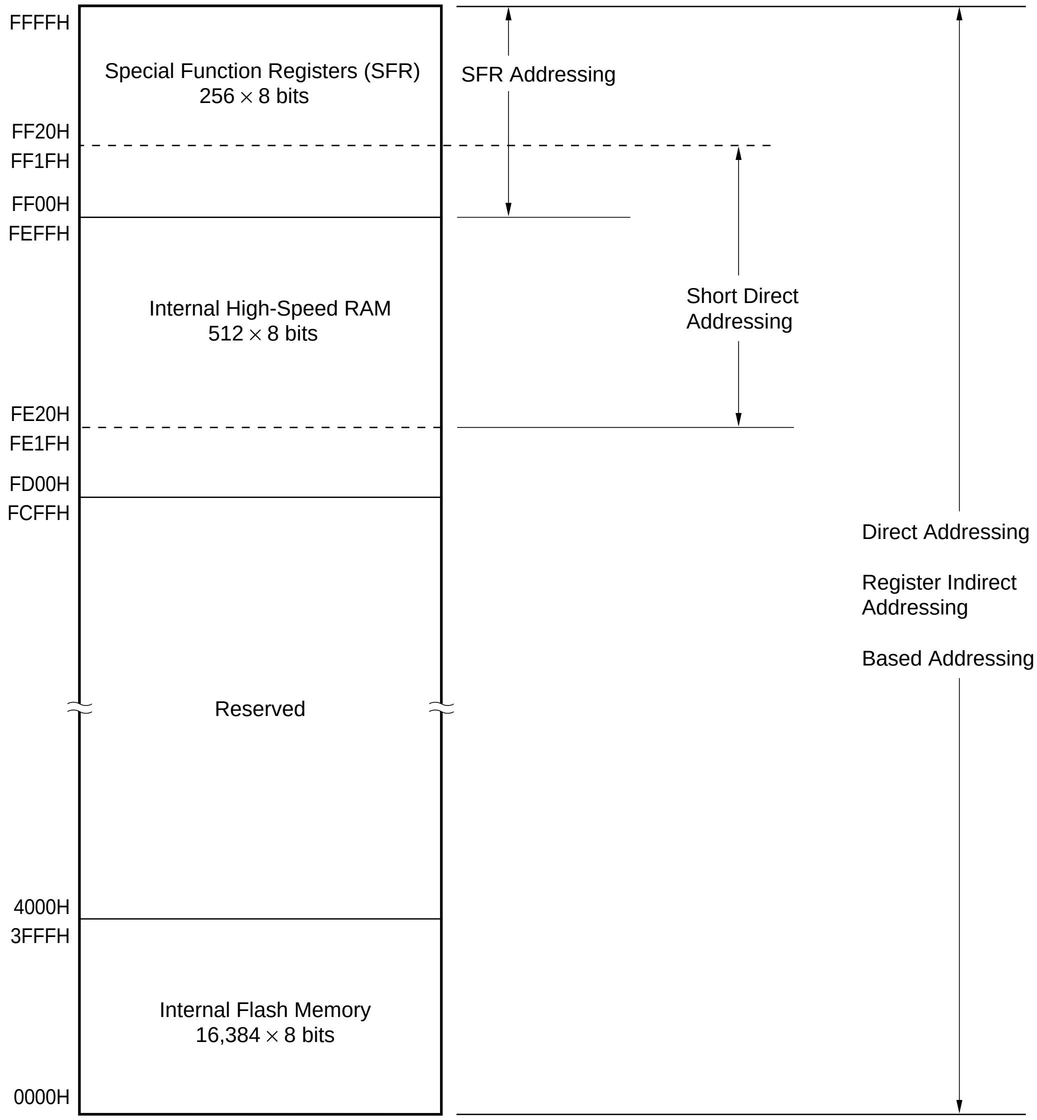


Figure 3-10. Data Memory Addressing (μPD78F9026A)



3.2 Processor Registers

The μPD789026 Subseries provides the following on-chip processor registers:

3.2.1 Control registers

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

(1) Program counter (PC)

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

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

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

Figure 3-11. 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 PSW to 02H.

Figure 3-12. 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 the 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 and interrupt request acknowledgement is controlled with an interrupt mask flag for each interrupt source.

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

(b) Zero flag (Z)

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

(c) Auxiliary carry flag (AC)

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

(d) Carry flag (CY)

This flag stores overflow and underflow upon add/subtract instruction execution. It stores the shift-out value upon rotate instruction execution and functions as a bit accumulator during bit 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 3-13. Stack Pointer Configuration

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

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

Each stack operation saves/restores data as shown in Figures 3-14 and 3-15.

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

Figure 3-14. Data to be Saved to Stack Memory

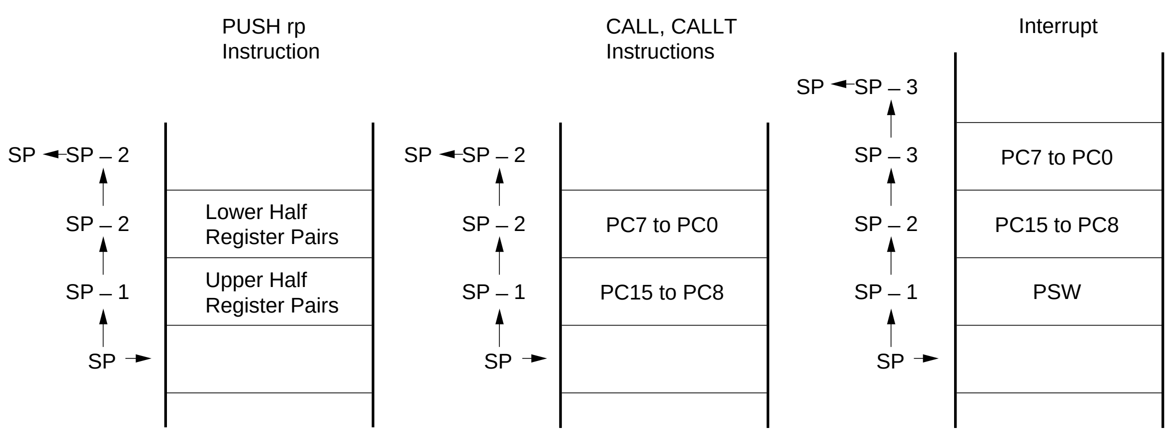
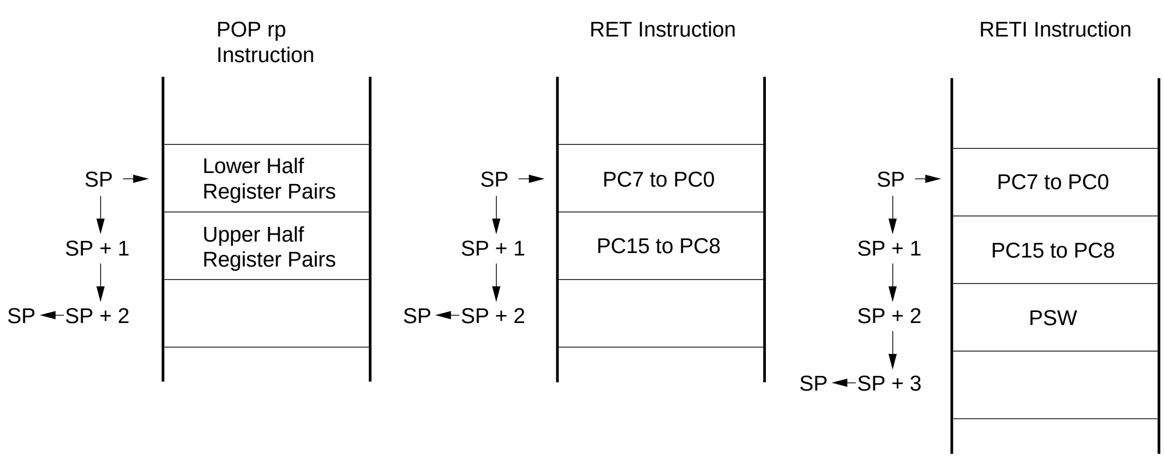


Figure 3-15. Data to be Restored from Stack Memory



3.2.2 General-purpose registers

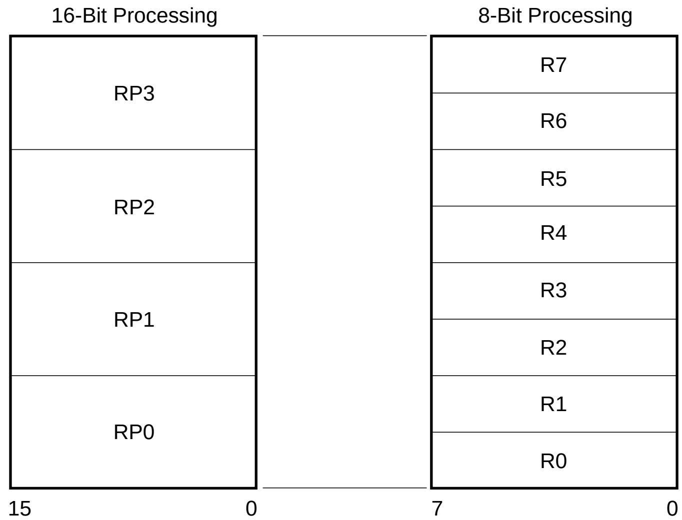
A general-purpose register consists of eight 8-bit registers (X, A, C, B, E, D, L, and H).

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).

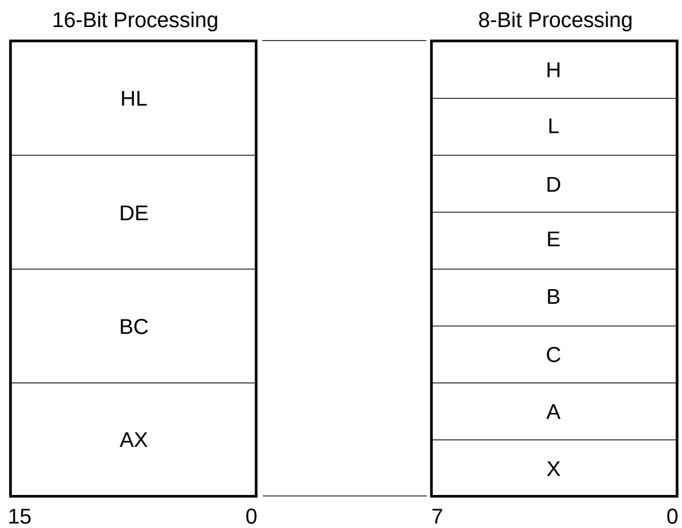
They can be written in terms of functional names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

Figure 3-16. General-Purpose Register Configuration

(a) Absolute Names



(b) Functional Names



3.2.3 Special function register (SFR)

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

It is allocated in the 256-byte area FF00H to FFFFH.

The special function register can be manipulated, like the general-purpose register, 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
  Writes 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
  Writes 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
  Writes a symbol reserved with assembler for the 16-bit manipulation instruction operand. When specifying an address, write an even address.

Table 3-4 lists the special function register. The meanings of the symbols in this table are as follows:

- Symbol
  Indicates the addresses of the incorporated 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 3-4. 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	O	O	–	00H
FF01H	Port 1	P1			O	O	–	
FF02H	Port 2	P2			O	O	–	
FF03H	Port 3	P3			O	O	–	
FF04H	Port 4	P4			O	O	–	
FF05H	Port 5	P5			O	O	–	
FF10H	Transmit shift register 00	TXS00	SIO00	W	–	O	–	FFH
	Receive buffer register 00	RXB00		R	–	O	–	Undefined
FF16H	16-bit compare register 20	CR20		W	–	O Note 1	O Note 2	FFFFH
FF17H								
FF18H	16-bit timer counter 20	TM20		R	–	O Note 1	O Note 2	0000H
FF19H								
FF1AH	16-bit capture register 20	TCP20			–	O Note 1	O Note 2	Undefined
FF1BH								
FF20H	Port mode register 0	PM0		R/W	O	O	–	FFH
FF21H	Port mode register 1	PM1			O	O	–	
FF22H	Port mode register 2	PM2			O	O	–	
FF23H	Port mode register 3	PM3			O	O	–	
FF24H	Port mode register 4	PM4			O	O	–	
FF25H	Port mode register 5	PM5			O	O	–	
FF42H	Timer clock select register 2	TCL2			–	O	–	00H
FF50H	8-bit compare register 00	CR00		W	–	O	–	Undefined
FF51H	8-bit timer counter 00	TM00		R	–	O	–	00H
FF53H	8-bit timer mode control register 00	TMC00		R/W	O	O	–	
FF5BH	16-bit timer mode control register 20	TMC20			O	O	–	
FF70H	Asynchronous serial interface mode register 00	ASIM00			O	O	–	
FF71H	Asynchronous serial interface status register 00	ASIS00		R	O	O	–	
FF72H	Serial operation mode register 00	CSIM00		R/W	O	O	–	
FF73H	Baud rate generator control register 00	BRGC00			–	O	–	

Notes 1. CR20, TM20, and TCP20 are designed for 16-bit access. They can also be accessed in 8-bit mode, however. In 8-bit access mode, use direct addressing.

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

Table 3-4. 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	
FFE0H	Interrupt request flag register 0	IF0	R/W	O	O	–	00H
FFE1H	Interrupt request flag register 1	IF1		O	O	–	
FFE4H	Interrupt mask flag register 0	MK0		O	O	–	FFH
FFE5H	Interrupt mask flag register 1	MK1		O	O	–	
FFECH	External interrupt mode register 0	INTM0		–	O	–	00H
FFF5H	Key return mode register 00	KRM00		O	O	–	
FFF7H	Pull-up resistor option register	PUO		O	O	–	
FFF9H	Watchdog timer mode register	WDTM		O	O	–	
FFFAH	Oscillation settling time select register	OSTS		–	O	–	04H
FFFBH	Processor clock control register	PCC		O	O	–	02H

3.3 Instruction Address Addressing

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

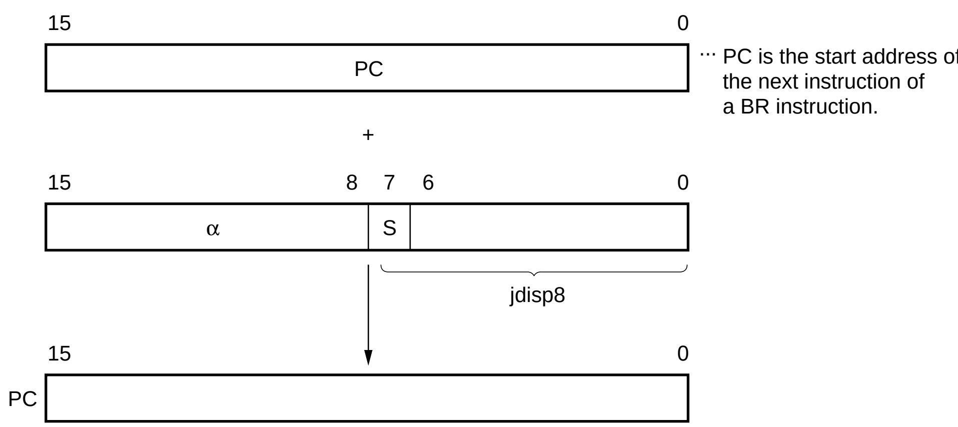
3.3.1 Relative addressing

[Function]

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

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

[Illustration]



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

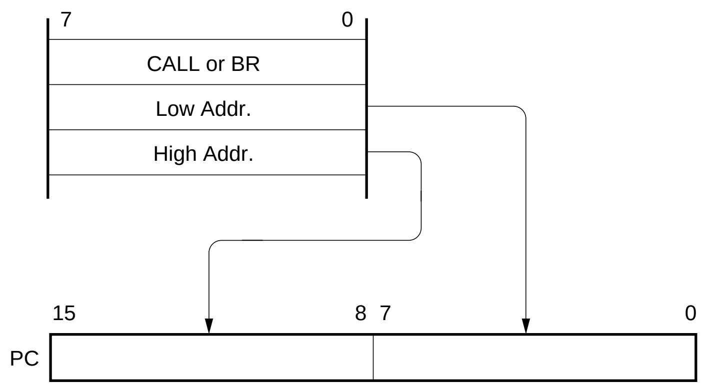
3.3.2 Immediate addressing

[Function]

Immediate data in the instruction word is transferred to the program counter (PC) and branched.
This function is carried out when the CALL !addr16 or BR !addr16 instruction is executed.
CALL !addr16 and BR !addr16 instructions can branch to all the memory spaces.

[Illustration]

In case of CALL !addr16 or BR !addr16 instruction



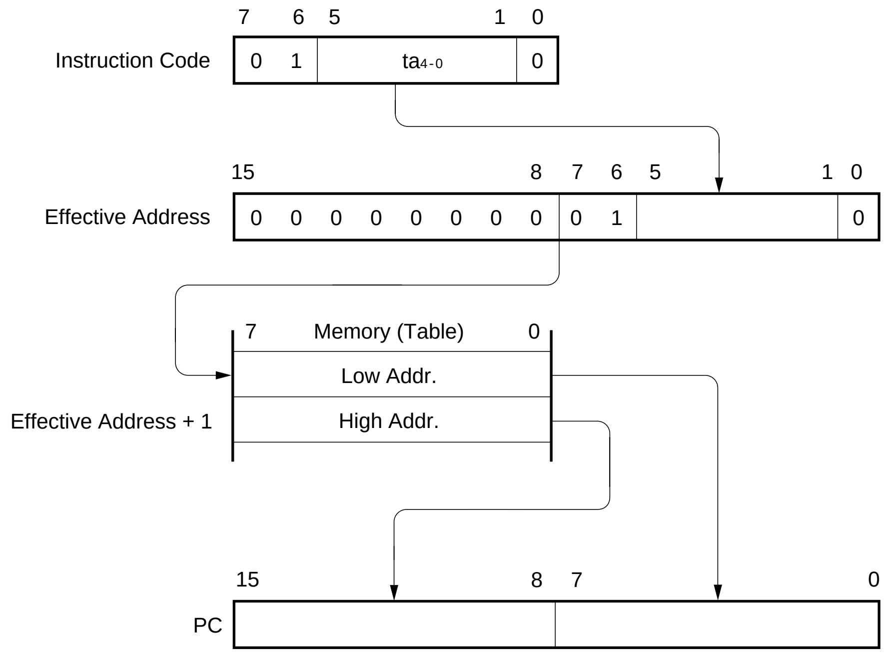
3.3.3 Table indirect addressing

[Function]

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

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

[Illustration]



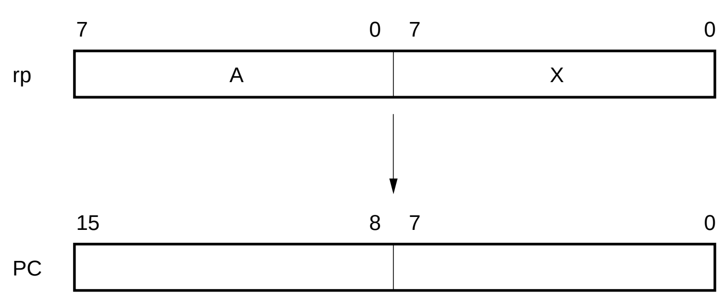
3.3.4 Register addressing

[Function]

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

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

[Illustration]



3.4 Operand Address Addressing

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

3.4.1 Direct addressing

[Function]

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

[Operand format]

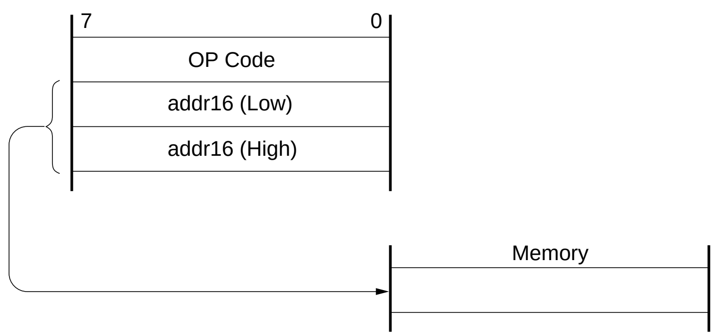
Identifier	Description
addr16	Label or 16-bit immediate data

[Example]

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

Instruction Code	0	0	1	0	1	0	0	1	OP Code
	0	0	0	0	0	0	0	0	00H
	1	1	1	1	1	1	1	0	FEH

[Illustration]



3.4.2 Short direct addressing

[Function]

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

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

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

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

[Operand format]

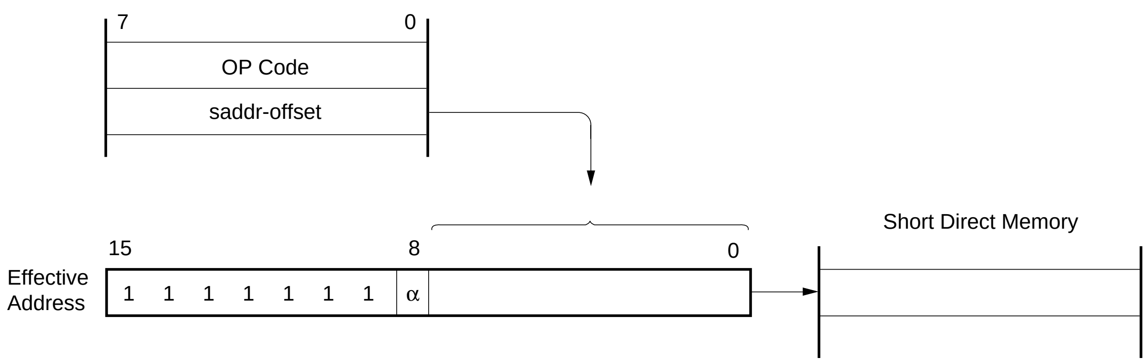
Identifier	Description
saddr	Label or FE20H to FF1FH immediate data
saddrp	Label or FE20H to FF1FH immediate data (even address only)

[Example]

MOV FE30H, #50H; When setting saddr to FE30H and the immediate data to 50H

Instruction Code	1	1	1	1	0	1	0	1	OP Code
	0	0	1	1	0	0	0	0	30H (saddr-offset)
	0	1	0	1	0	0	0	0	50H (immediate data)

[Illustration]



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

3.4.3 Special function register (SFR) addressing

[Function]

The memory-mapped special function register (SFR) is addressed with 8-bit immediate data in an instruction word.

This addressing is applied to the 240-byte spaces FF00H to FFCFH and FFE0H to FFFFH. However, the SFR mapped at FF00H to FF1FH can also be accessed with short direct addressing.

[Operand format]

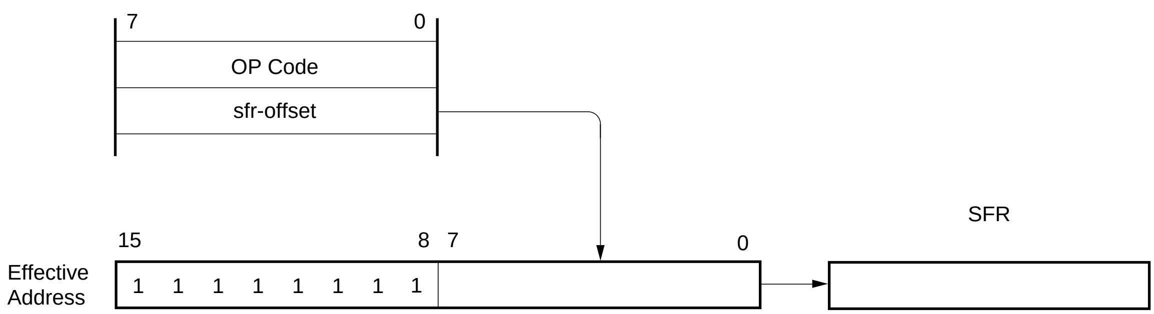
Identifier	Description
sfr	Special function register name

[Example]

MOV PM0, A; When selecting PM0 for sfr

Instruction Code	1	1	1	0	0	1	1	1
	0	0	1	0	0	0	0	0

[Illustration]



3.4.4 Register addressing

[Function]

The general-purpose register is accessed as an operand. The general-purpose register to be accessed is specified with register specification 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 written 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).

[Example]

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

Instruction Code	0	0	0	0	1	0	1	0
	0	0	1	0	0	1	0	1

Register Specification Code

INCW DE; When selecting the DE register pair for rp

Instruction Code	1	0	0	0	1	0	0	0

Register Specification Code

3.4.5 Register indirect addressing

[Function]

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

[Operand format]

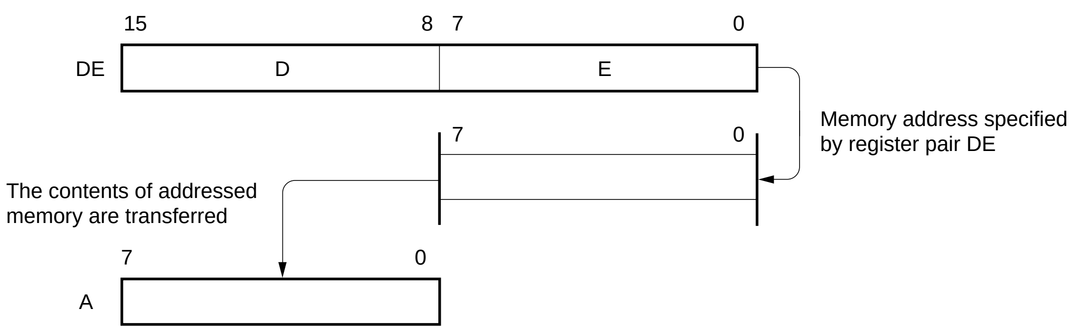
Identifier	Description
–	[DE], [HL]

[Example]

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

Instruction Code	0	0	1	0	1	0	1	1

[Illustration]



3.4.6 Based addressing

[Function]

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

[Operand format]

Identifier	Description
–	[HL+byte]

[Example]

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

Instruction Code								
	0	0	1	0	1	1	0	1
	0	0	0	1	0	0	0	0

3.4.7 Stack addressing

[Function]

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

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

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

[Example]

In the case of PUSH DE

Instruction Code								
	1	0	1	0	1	0	1	0

CHAPTER 4 PORT FUNCTIONS

4.1 Functions of Ports

The μPD789026 Subseries provides the ports shown in Figure 4-1, enabling various methods of control.

Alternate functions are provided in addition to the digital I/O port function. For more information on these alternate functions, see **Chapter 2**.

Figure 4-1. Port Types

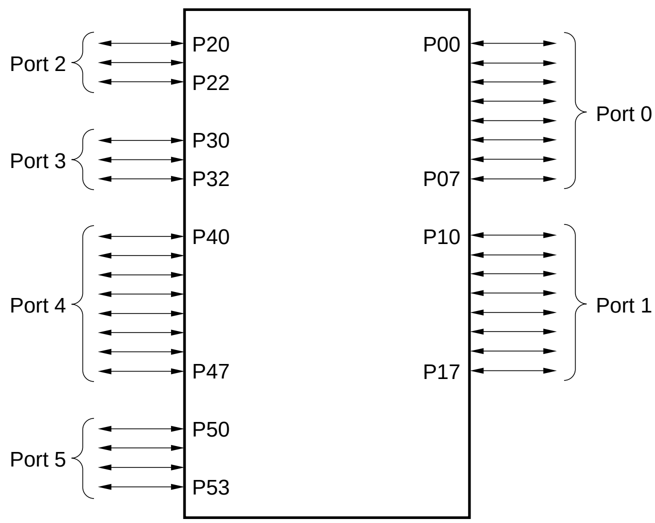


Table 4-1. Port Functions

Pin Name	Input/Output	Function	After Reset	Alternate Function
P00 to P07	Input/output	Port 0 8-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	–
P10 to P17	Input/output	Port 1 8-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	–
P20	Input/output	Port 2 3-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	$\overline{\text{SCK0}}$/ASCK
P21				SO0/TxD
P22				SI0/RxD
P30	Input/output	Port 3 3-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	INTP0
P31				INTP1
P32				INTP2/CPT2
P40 to P47	Input/output	Port 4 8-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	KR0 to KR7
P50	Input/output	Port 5 4-bit I/O port I/O specifiable in 1-bit units When used as input port, on-chip pull-up resistor can be connected by setting of the pull-up resistor option register (PUO). LEDs can be driven directly.	Input	TI0/TO0
P51				TO2
P52, P53				–

4.2 Port Configuration

Ports have the following hardware configuration.

Table 4-2. Port Configuration

Parameter	Configuration
Control register	Port mode register (PMm: m = 0 to 5) Pull-up resistor option register (PUO)
Port	Total: 34 (input/output: 34)
Pull-up resistor	Total: 34 (on-chip pull-up resistor can be connected by software)

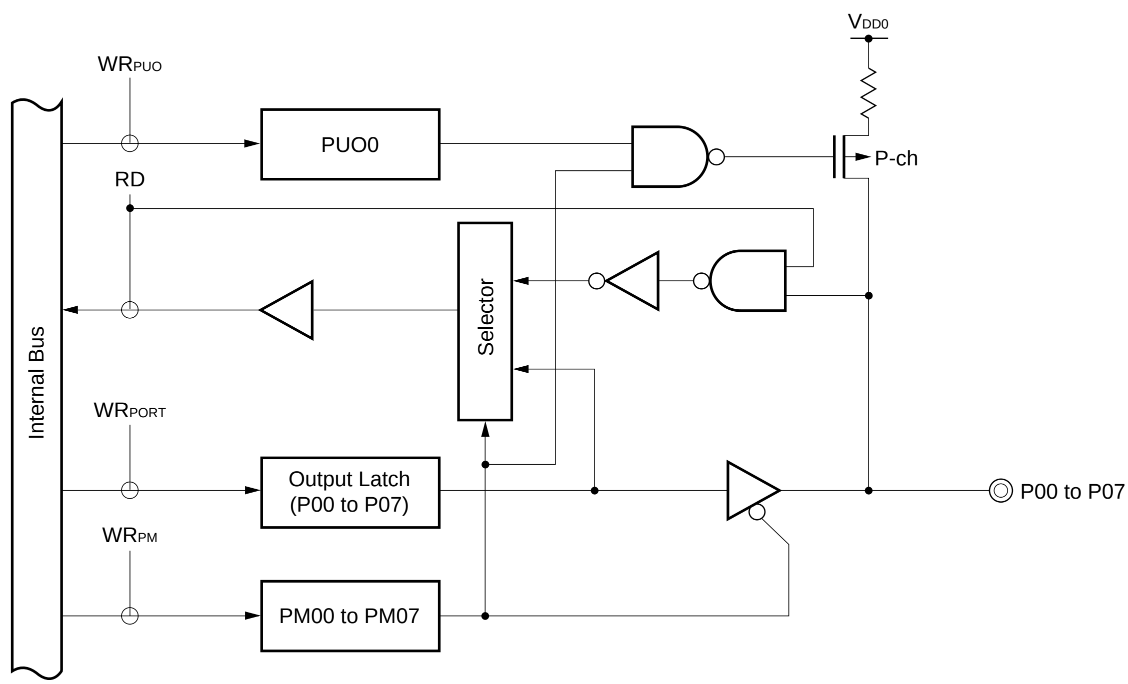
4.2.1 Port 0

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

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

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

Figure 4-2. Block Diagram of P00 to P07



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

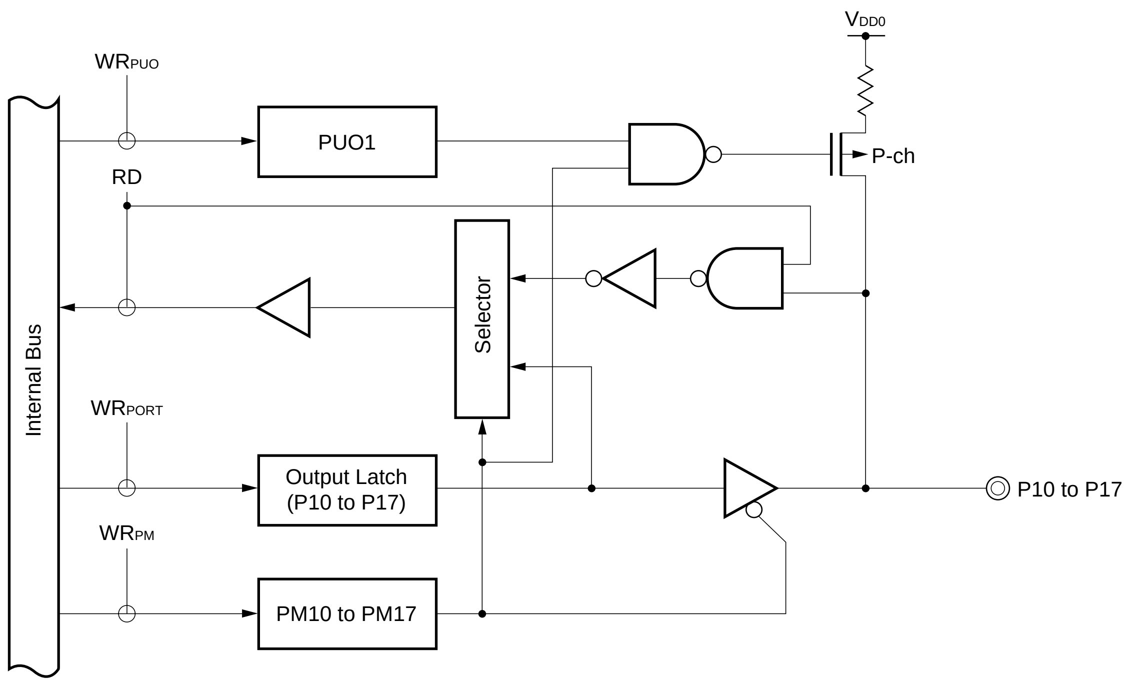
4.2.2 Port 1

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

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

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

Figure 4-3. Block Diagram of P10 to P17



PUO : Pull-up resistor option register

PM : Port mode register

RD : Port 1 read signal

WR : Port 1 write signal

4.2.3 Port 2

This is a 3-bit I/O port with output latch. Port 2 can be specified in the input or output mode in 1-bit units by using the port mode register 2 (PM2). When using P20 to P22 pins as input port pins, on-chip pull-up resistors can be connected in 3-bit units by using the pull-up resistor option register (PUO).

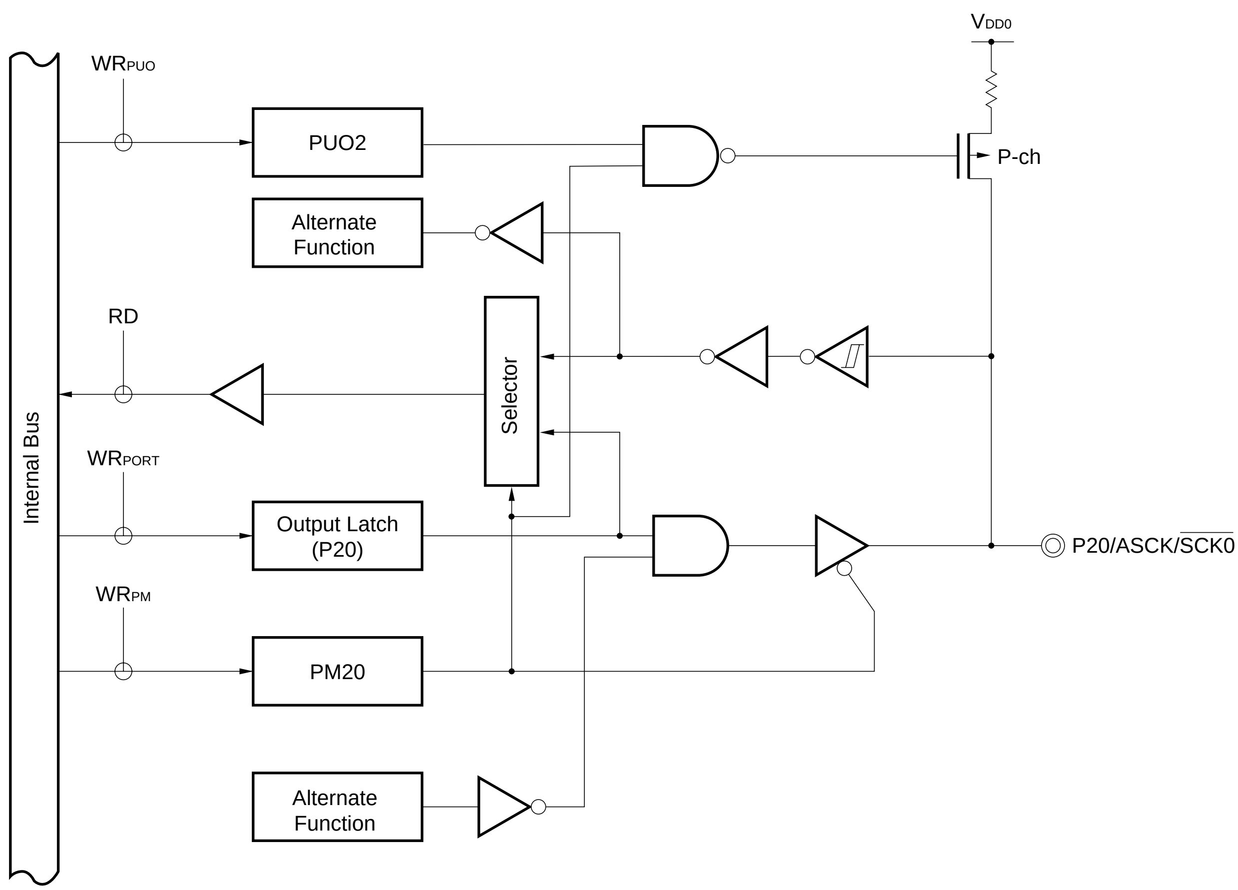
The pins of this port are also used as the data I/O and clock I/O pins of the serial interface.

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

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

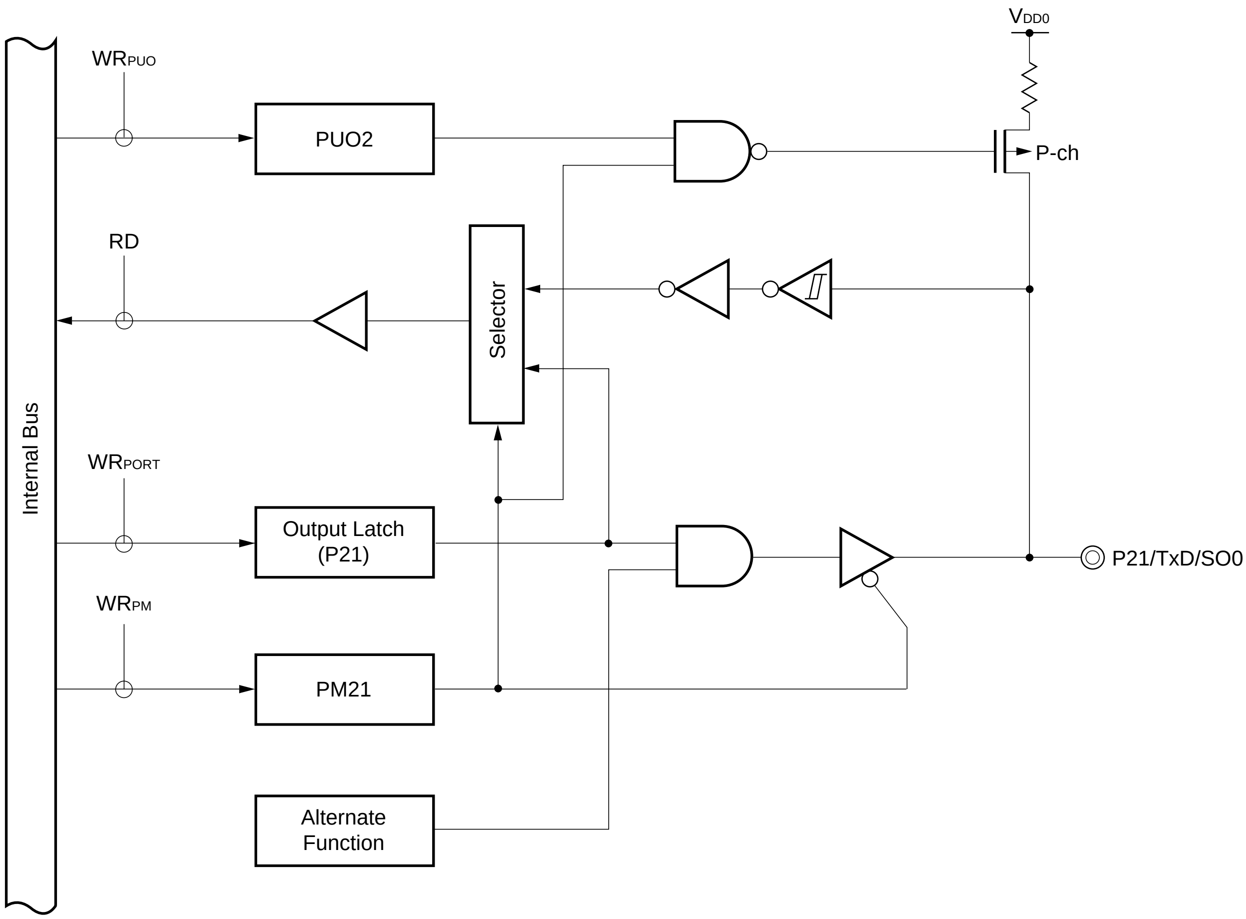
Caution When using the pins of port 2 as the serial interface, the I/O mode and output latch must be set according to the function to be used. For details of the settings, see Table 9-2.

Figure 4-4. Block Diagram of P20



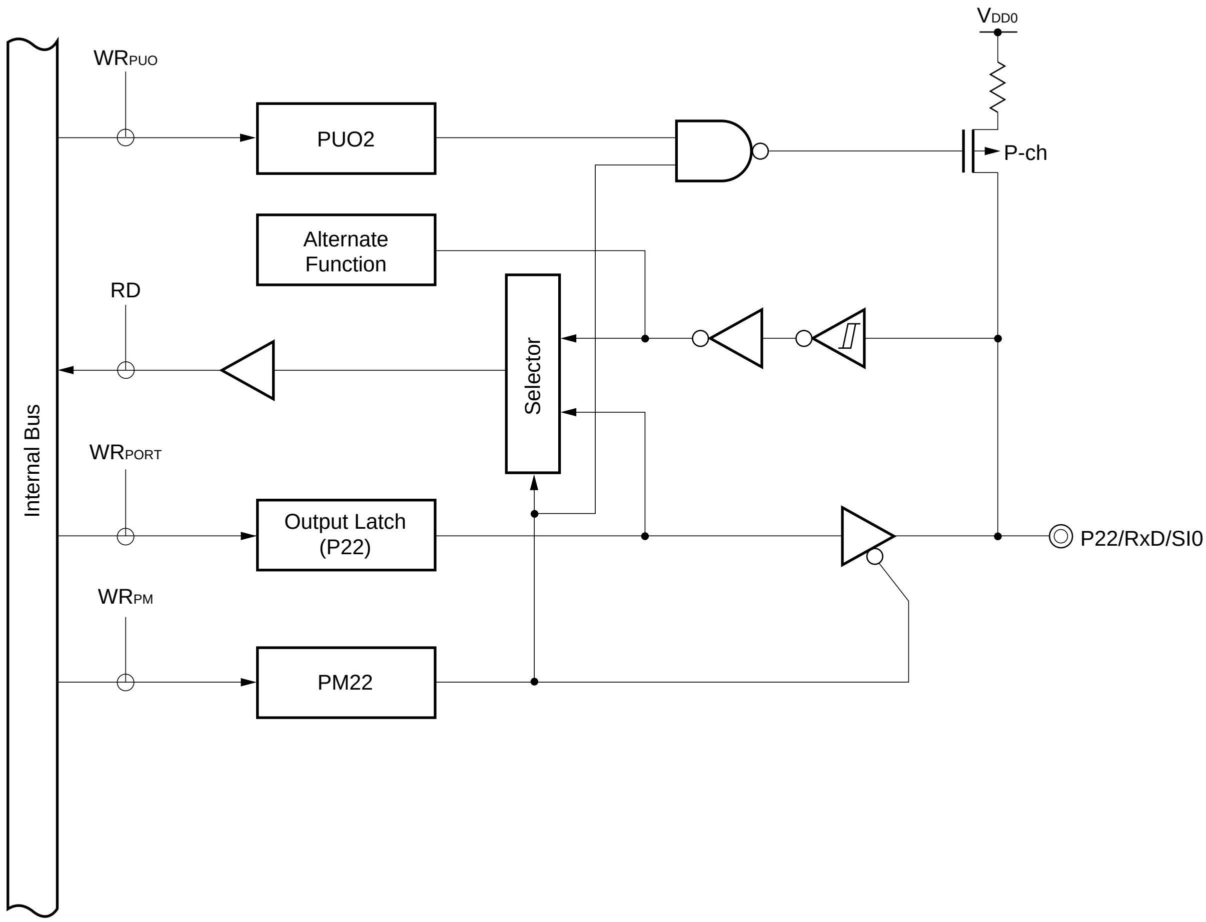
PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 2 read signal
WR : Port 2 write signal

Figure 4-5. Block Diagram of P21



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

Figure 4-6. Block Diagram of P22



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

4.2.4 Port 3

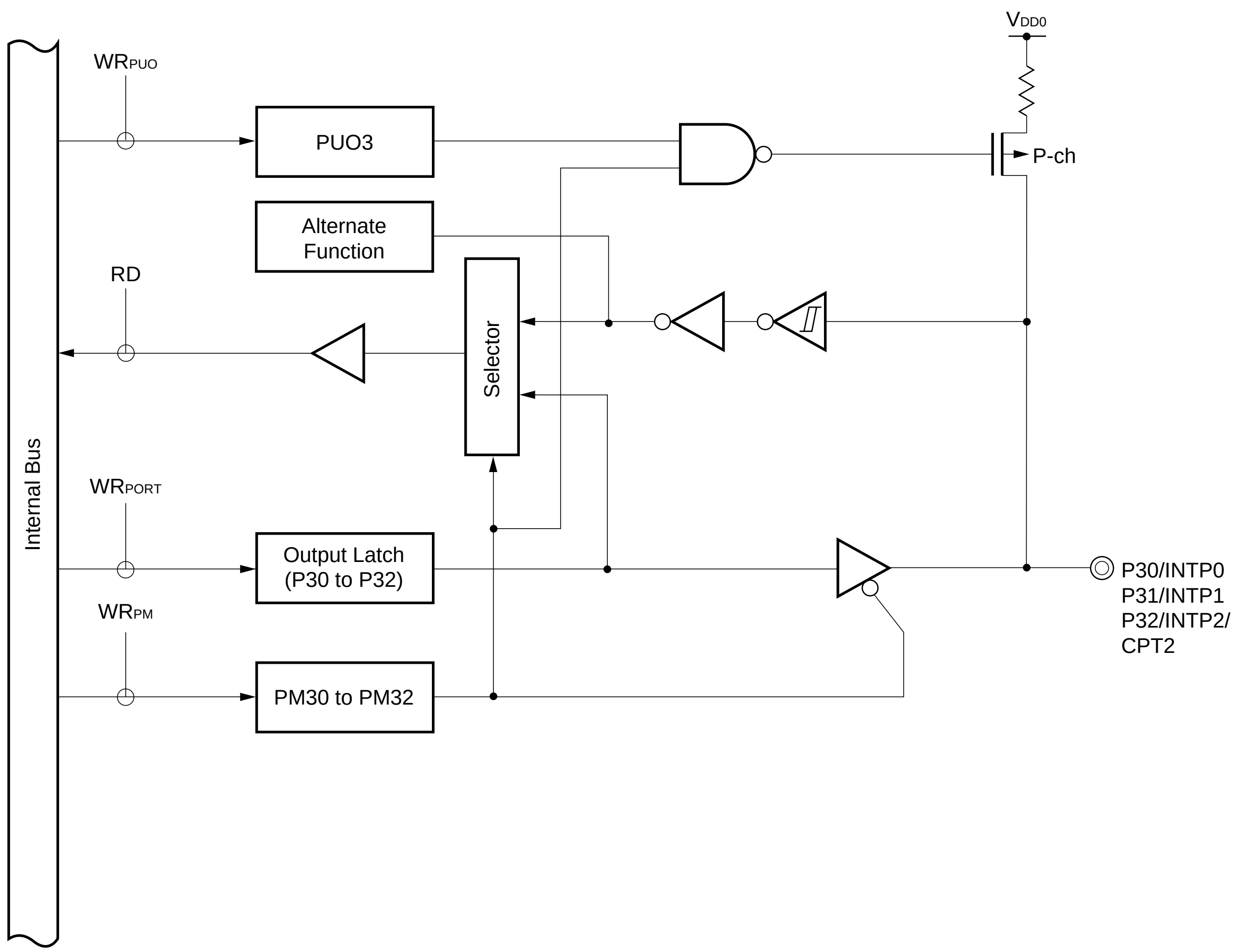
This is a 3-bit I/O port with output latch. It can be specified in input or output mode in 1-bit units by using the port mode register 3 (PM3). When using P30 to P32 pins as input port pins, on-chip pull-up resistors can be connected in 3-bit units by using the pull-up resistor option register (PUO).

The pins of this port are also used as the external interrupt input and capture edge input.

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

Figure 4-7 shows the block diagram of port 3.

Figure 4-7. Block Diagram of P30 to P32



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

4.2.5 Port 4

This is an 8-bit I/O port with output latch. Port 4 can be specified in the input or output mode in 1-bit units by using port mode register 4 (PM4). When using P40 to P47 pins as input port pins, on-chip pull-up resistors can be connected in 8-bit units by using the pull-up resistor option register (PUO).

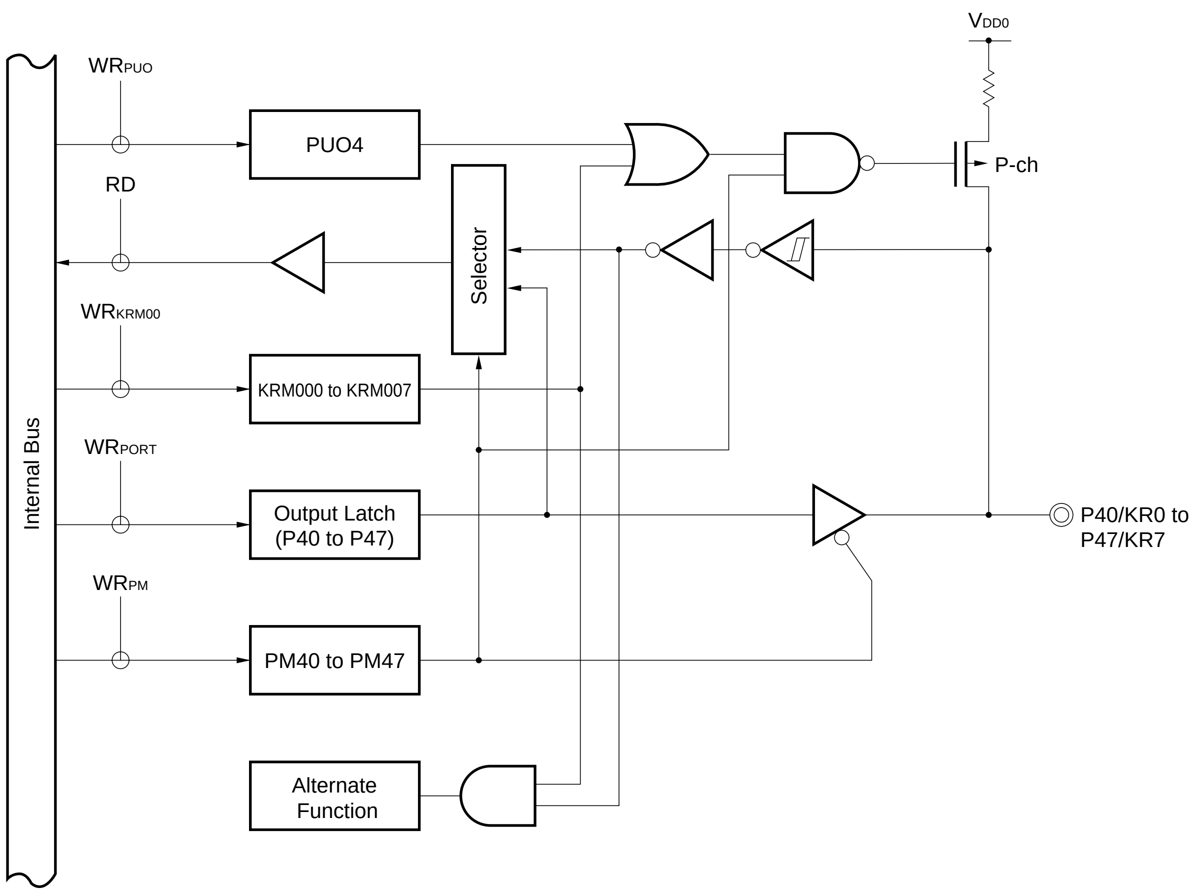
The pins of this port are also used as the key return input.

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

Figure 4-8 shows the block diagram of port 4.

Caution When using port 4 for the key return function, it is necessary to set key return mode register 00. For details of the settings, see Section 10.3 (5).

Figure 4-8. Block Diagram of P40 to P47



KRM00 : Key return mode register 00
PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 4 read signal
WR : Port 4 write signal

4.2.6 Port 5

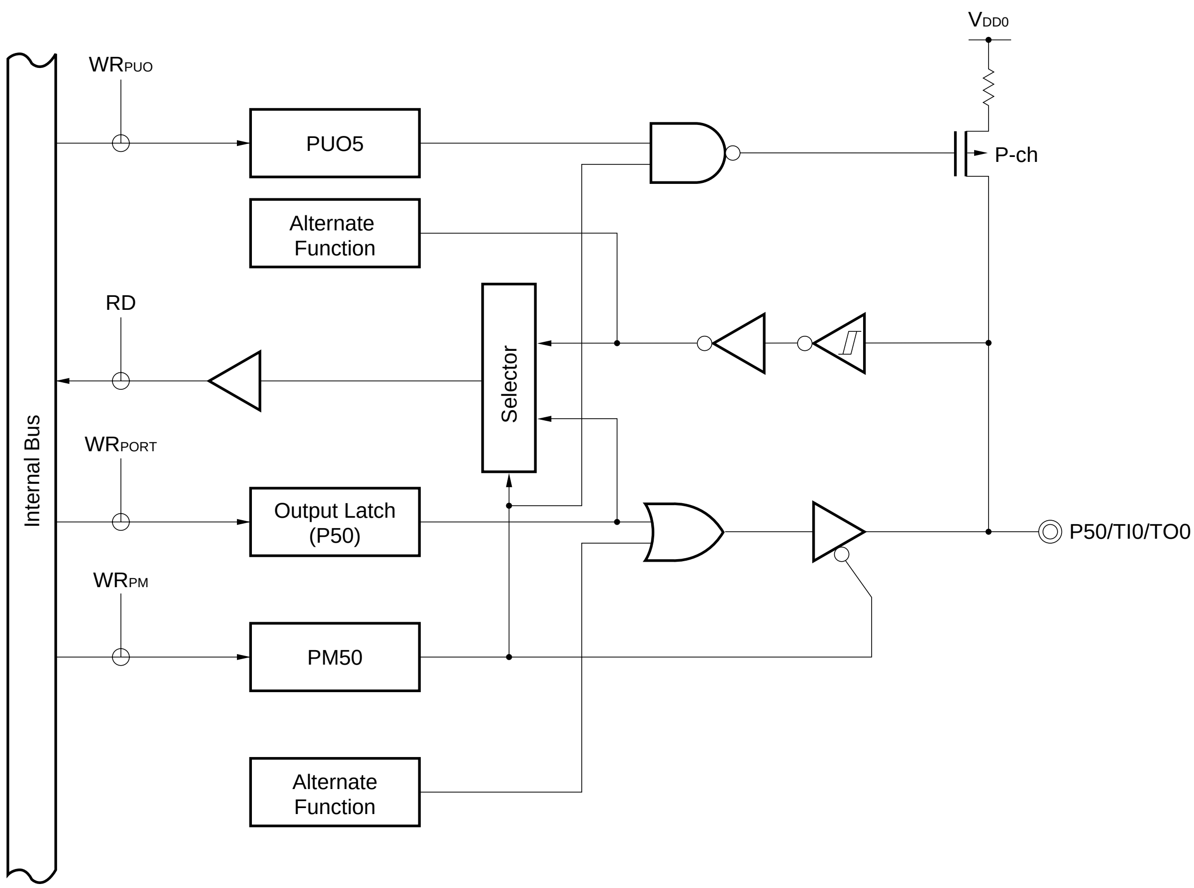
This is a 4-bit I/O port with output latch. Port 5 can be specified in the input or output mode in 1-bit units by using the port mode register 5 (PM5). When using P50 to P53 pins as input port pins, on-chip pull-up resistors can be connected in 4-bit units by using the pull-up resistor option register (PUO).

The pins of this port are also used as the data I/O pins of the timer.

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

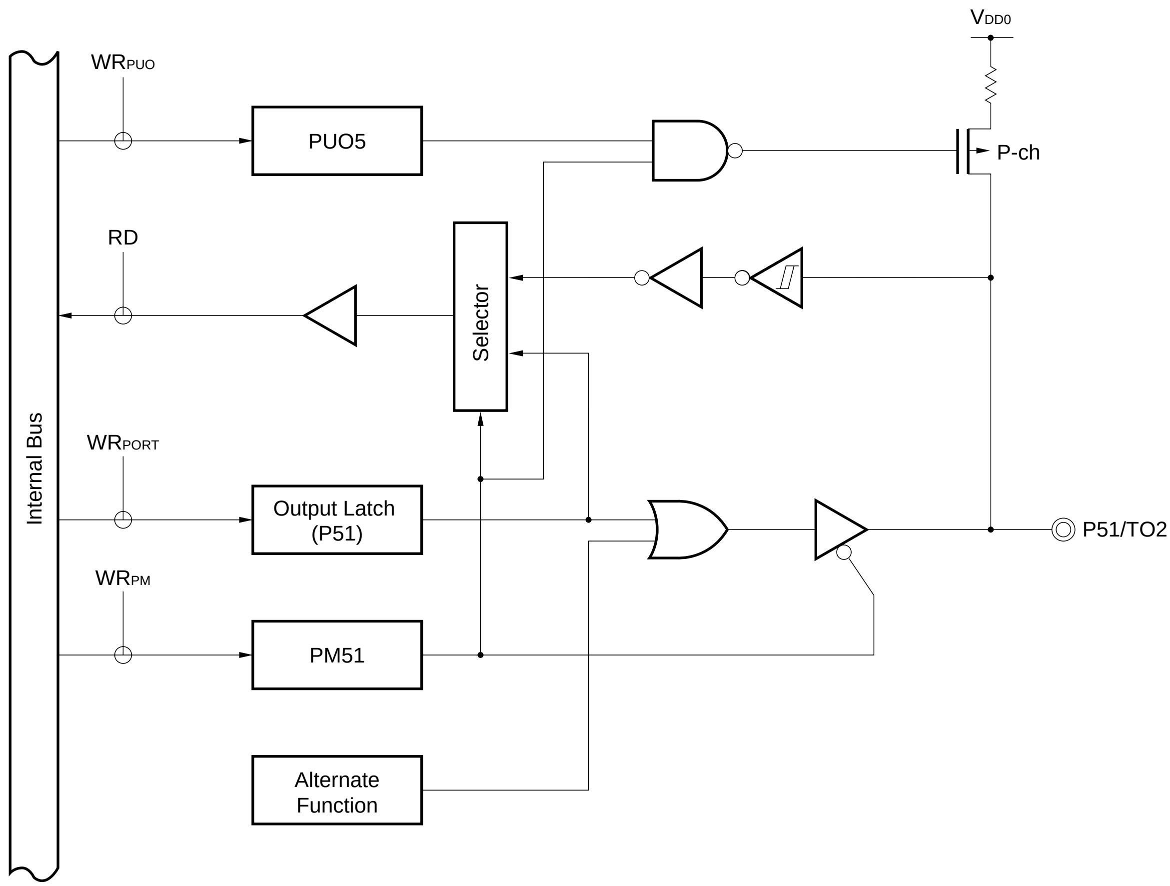
Figures 4-9 through 4-11 show the block diagrams of port 5.

Figure 4-9. Block Diagram of P50



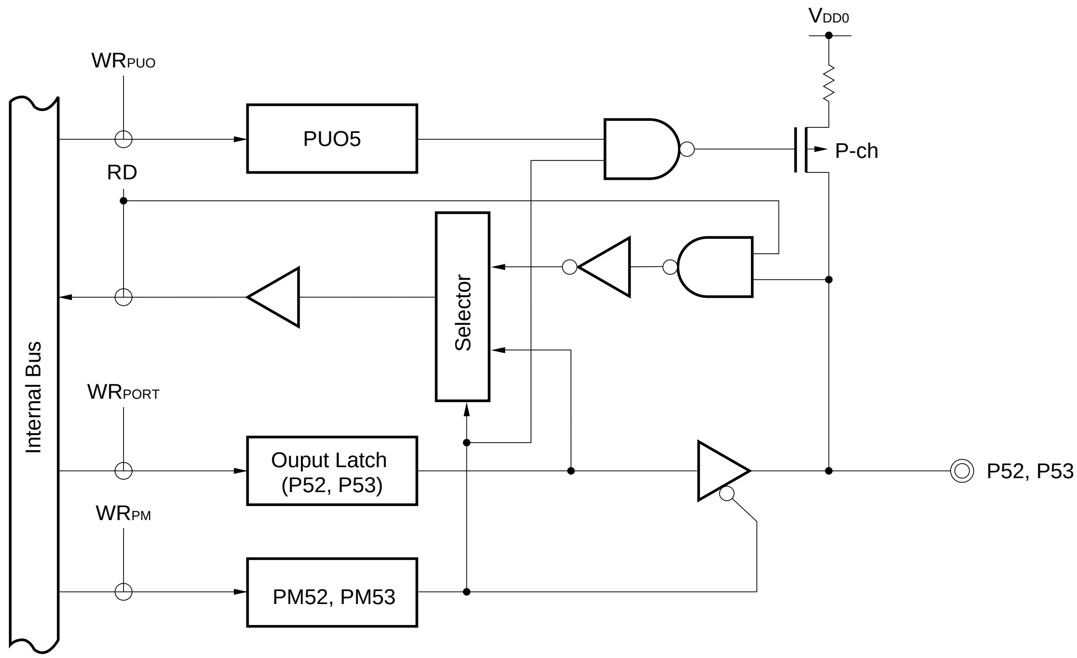
PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 5 read signal
WR : Port 5 write signal

Figure 4-10. Block Diagram of P51



PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 5 read signal
WR : Port 5 write signal

Figure 4-11. Block Diagram of P52 and P53



PUO : Pull-up resistor option register
PM : Port mode register
RD : Port 5 read signal
WR : Port 5 write signal

4.3 Port Function Control Registers

The following two types of registers control the ports.

- Port mode registers (PM0 to PM5)
- Pull-up resistor option register (PUO)

(1) Port mode registers (PM0 to PM5)

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

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

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

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

Caution As port 3 has an alternate function as external interrupt input, when the port function output mode is specified and the output level is changed, the interrupt request flag is set. When the output mode is used, therefore, the interrupt mask flag should be set to 1 beforehand.

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

Pin Name	Alternate Function		PM××	P××
	Name	Input/Output		
P30	INTP0	Input	1	×
P31	INTP1	Input	1	×
P32	INTP2	Input	1	×
	CPT2	Input	1	×
P40 to P47[Note]	KR0 to KR7	Input	1	×
P50	TI0	Input	1	×
	TO0	Output	0	0
P51	TO2	Output	0	0

Note When an alternate function is used, set the key return mode register 00 (KRM00) to 1 (see **Section 10.3 (5)**).

Caution When Port 2 is used for serial interface pin, the I/O mode and output latch must be set according to its function. For details of the settings, see Table 9-2.

Remark × : Don't care
PM×× : Port mode register
P×× : Output latch of port

Figure 4-12. Port Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	FF21H	FFH	R/W
PM2	1	1	1	1	1	PM22	PM21	PM20	FF22H	FFH	R/W
PM3	1	1	1	1	1	PM32	PM31	PM30	FF23H	FFH	R/W
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	FF24H	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, 1, 4 : n = 0 to 7 m = 2, 3 : n = 0 to 2 m = 5 : n = 0 to 3
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

(2) Pull-up resistor option register (PUO)

The pull-up resistor option register (PUO) sets whether an on-chip pull-up resistor on each port is used or not.

On the port which is specified to use the on-chip pull-up resistor in the PUO, the pull-up resistor can be internally used only for the bits set in the input mode. No on-chip pull-up resistors can be used for the bits set in the output mode in spite of setting the PUO. On-chip pull-up resistors cannot be used either when the pins are used as the alternate-function output pins.

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

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

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

Symbol	7	6	<5>	<4>	<3>	<2>	<1>	<0>	Address	After Reset	R/W
PUO	0	0	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	FFF7H	00H	R/W

PUOm	Port m On-Chip Pull-Up Resistor Selection (m = 0 to 5)
0	On-chip pull-up resistor not used
1	On-chip pull-up resistor used

4.4 Operation of Port Functions

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

4.4.1 Writing to I/O port

(1) In output mode

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

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 in the input mode and not subject to manipulation become undefined.

4.4.2 Reading from I/O port

(1) In output mode

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

(2) In input mode

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

4.4.3 Arithmetic operation of I/O port

(1) In output mode

An arithmetic operation can be performed with the contents of an 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 in the input mode and not subject to manipulation become undefined.

[MEMO]

CHAPTER 5 CLOCK GENERATION CIRCUIT

5.1 Function of Clock Generation Circuit

The clock generation circuit generates the clock to be supplied to the CPU and peripheral hardware. The system clock oscillator consists of the following type.

- System clock oscillator
  This circuit oscillates at frequencies of 1.0 to 5.0 MHz. Oscillation can be stopped by executing the STOP instruction.

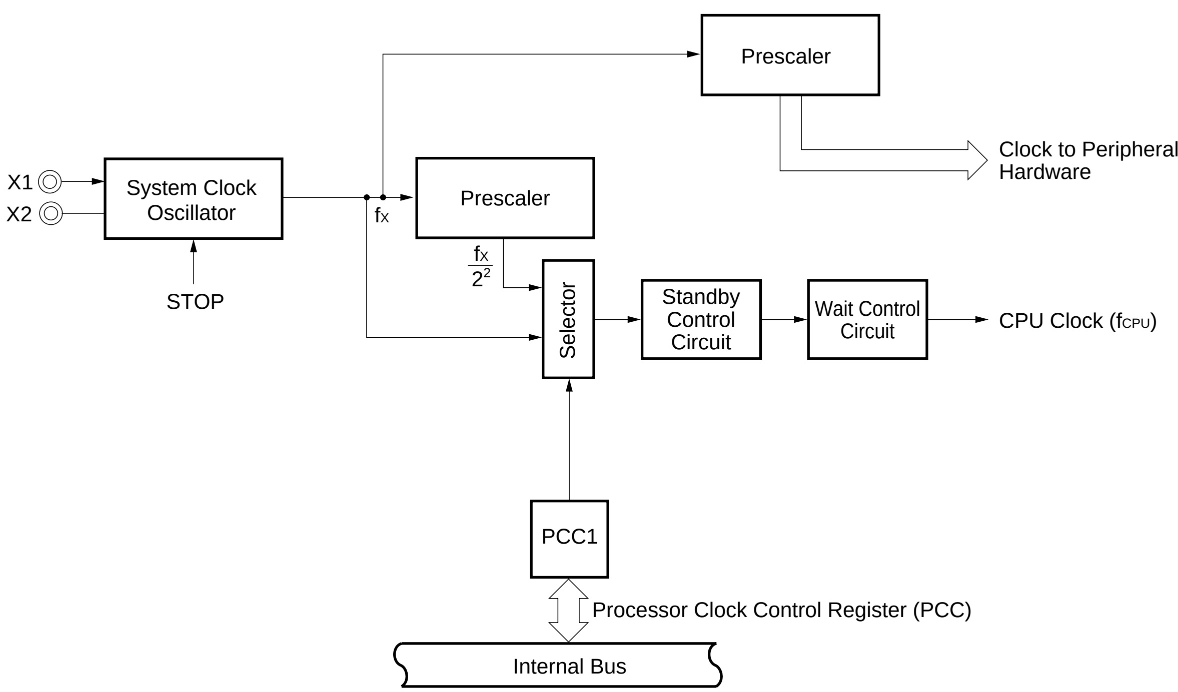
5.2 Configuration of Clock Generation Circuit

The clock generation circuit consists of the following hardware.

Table 5-1. Configuration of Clock Generation Circuit

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

Figure 5-1. Block Diagram of Clock Generation Circuit



5.3 Register Controlling Clock Generation Circuit

The clock generation circuit is controlled by the following register:

- Processor clock control register (PCC)

(1) Processor clock control register (PCC)

PCC sets CPU clock selection 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 5-2. Processor Clock Control Register Format

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

PCC1	CPU Clock (f_{CPU}) Selection
0	f_X (0.2 μs)
1	$f_X/2^2$ (0.8 μs)

Caution Be sure to set bit 0 and bits 2 to 7 to 0.

Remarks 1. f_X: System clock oscillation frequency

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

3. Minimum instruction execution time: 2 f_{CPU}

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

5.4 System Clock Oscillation Circuits

5.4.1 System clock oscillation circuit

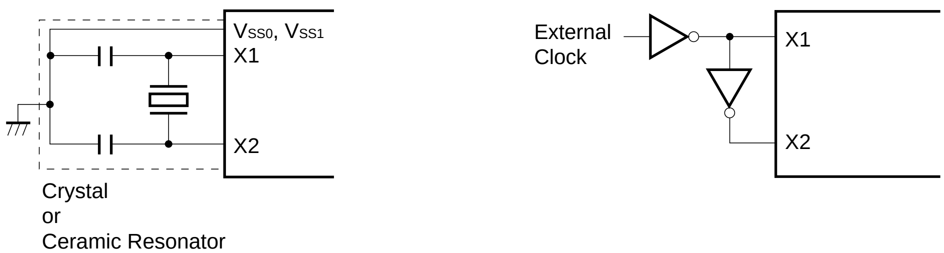
The system clock oscillation circuit is oscillated by the crystal or ceramic resonator (5.0 MHz TYP.) connected across the X1 and X2 pins.

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 5-3 shows the external circuit of the system clock oscillation circuit.

Figure 5-3. External Circuit of System Clock Oscillation Circuit

(a) Crystal or ceramic oscillation **(b) External clock**



Caution When using the system clock oscillator circuit, to avoid influence of wiring capacity, etc., wire the portion enclosed by the broken line in Figure 5-3 as follows:

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

Figure 5-4 shows incorrect examples of resonator connection.

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

(a) Too long wiring **(b) Crossed signal line**

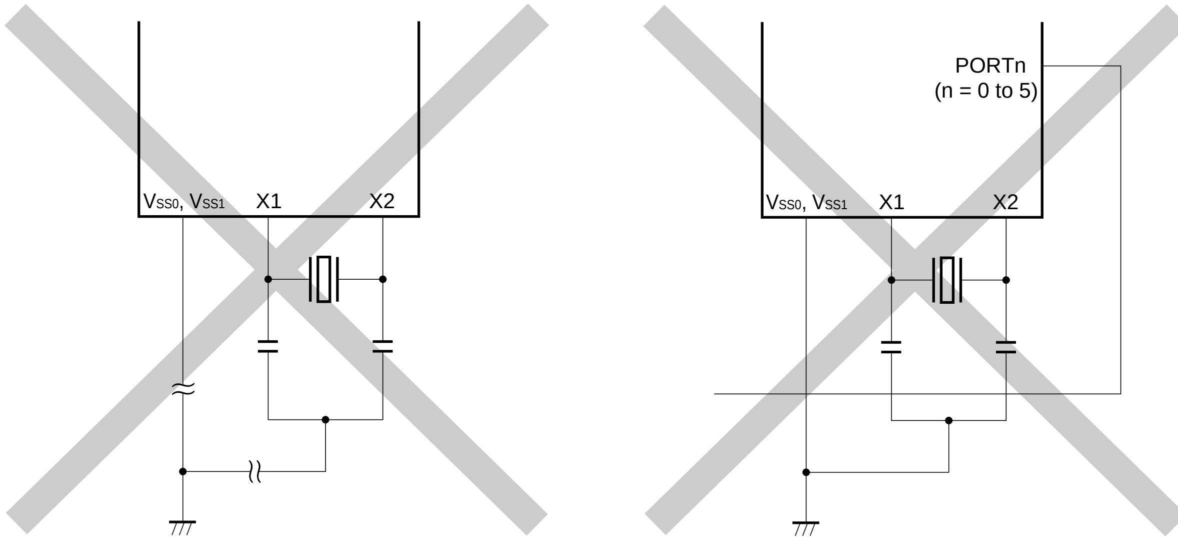
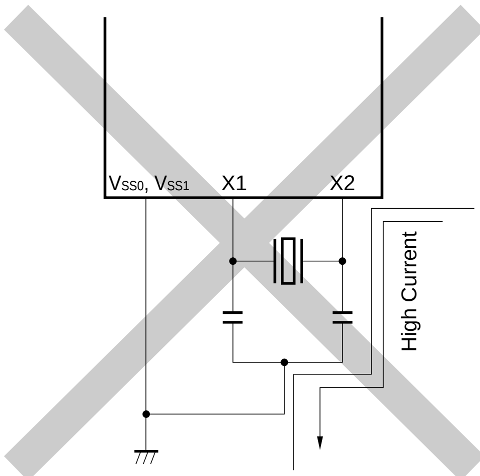
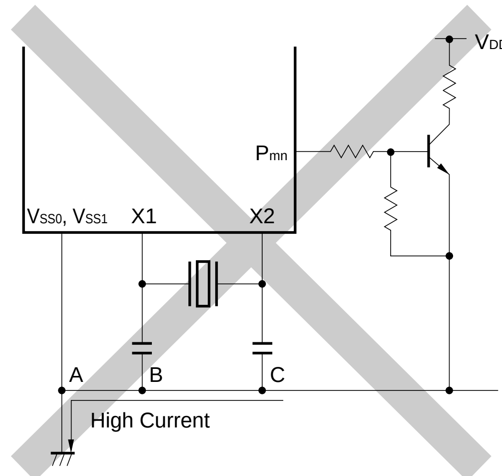


Figure 5-4. Incorrect Examples of Resonator Connection (2/2)

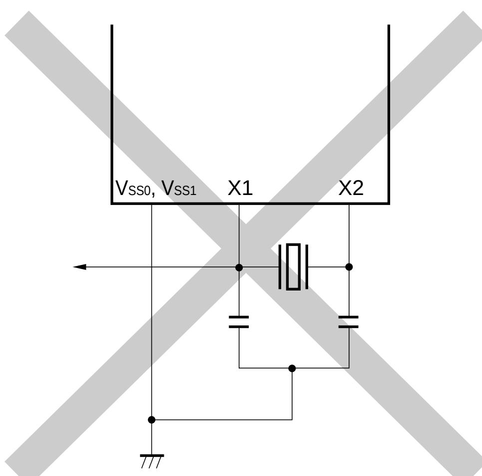
(c) Wiring near high alternating current



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



(e) Signal is extracted



5.4.2 Divider circuit

The divider circuit divides the output of the system clock oscillation circuit (f_X) to generate various clocks.

5.5 Operation of Clock Generation Circuit

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

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

The operation of the clock generation circuit is determined by the processor clock control register (PCC), as follows:

(a) The slow mode 2 f_{CPU} (1.6 μs: at 5.0-MHz operation) of the 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 system clock is stopped .

(b) Two types of CPU clocks f_{CPU} (0.2 μs and 0.8 μs: at 5.0-MHz operation) can be selected by the PCC setting.

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

(d) The clock to the peripheral hardware is supplied by dividing the system clock. The other peripheral hardware is stopped when the system clock is stopped (except, however, the external clock input operation).

5.6 Changing Setting of System Clock and CPU Clock

5.6.1 Time required for switching between system clock and CPU clock

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

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

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

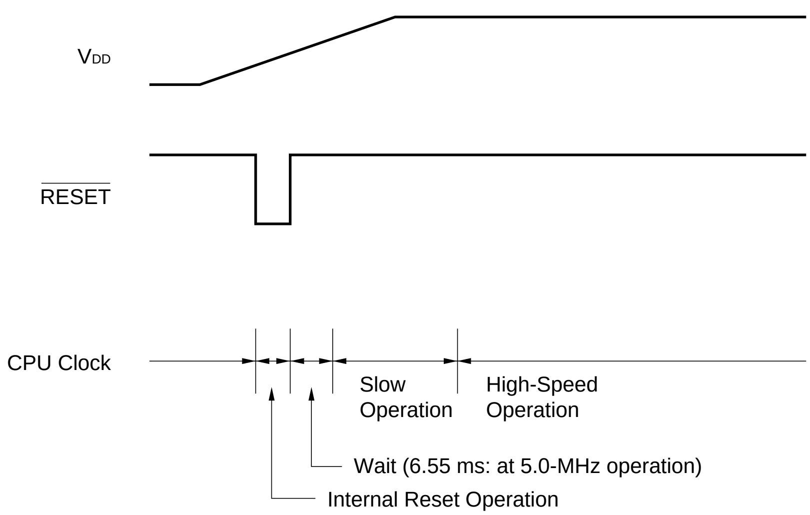
Set Value before Switching	Set Value after Switching	
PCC1	PCC1	PCC1
	0	1
0		4 clocks
1	2 clocks	

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

5.6.2 Switching CPU clock

The following figure illustrates how the CPU clock switches.

Figure 5-5. Switching CPU Clock



<1> The CPU is reset when the $\overline{\text{RESET}}$ pin is made low on power application. The effect of resetting is released when the $\overline{\text{RESET}}$ pin is later made high, and the system clock starts oscillating. At this time, the time during which oscillation settles ($2^{15}/f_X$) is automatically secured.
After that, the CPU starts instruction execution at the slow speed of the system clock (1.6 μs: at 5.0-MHz operation).

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

[MEMO]

6.1 16-Bit Timer Functions

16-bit timer 20 has the following functions.

- Timer interrupt
- Timer output
- Count value capture

(1) Timer interrupt

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

(2) Timer output

Timer output control is possible when a count value and compare value matches.

(3) Count value capture

A TM20 count value is latched synchronizing with the capture trigger and retained.

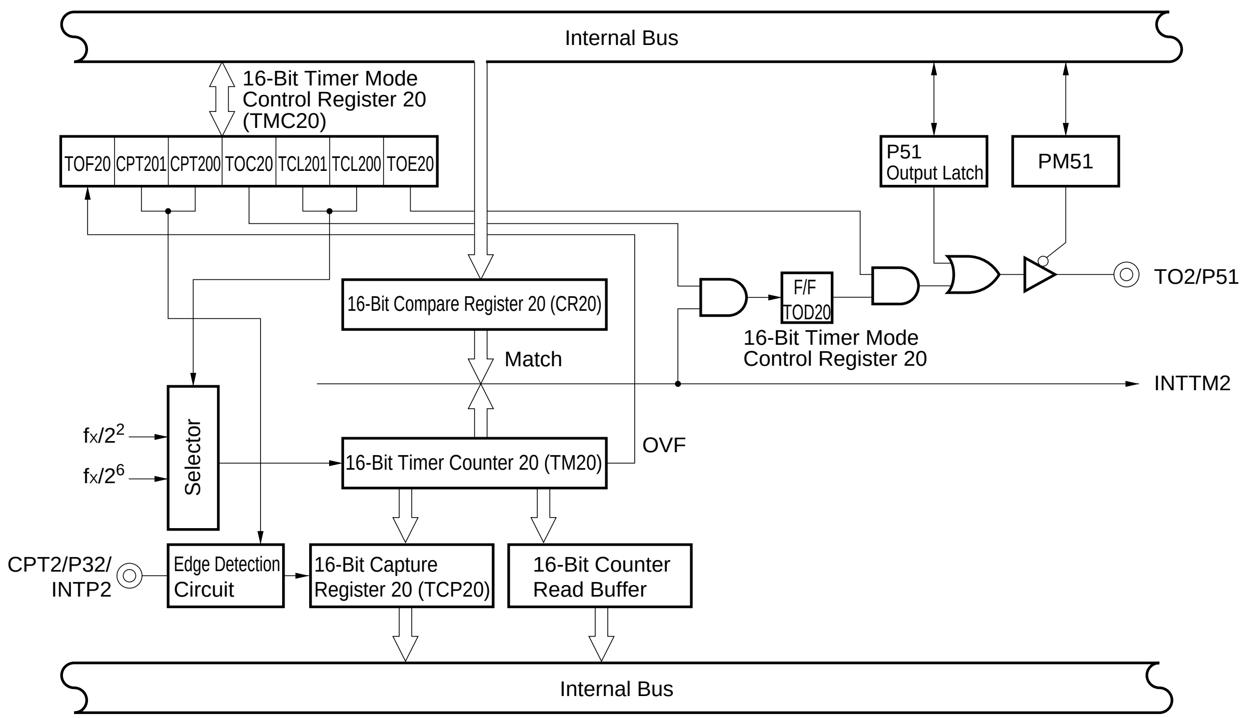
6.2 16-Bit Timer Configuration

16-bit timer 20 is configured in the following hardware.

Table 6-1. Configuration of 16-Bit Timer 20

Item	Configuration
Timer counter	16 bits × 1 (TM20)
Register	Compare register : 16 bits × 1 (CR20) Capture register : 16 bits × 1 (TCP20)
Timer output	1 (TO20)
Control register	16-bit timer mode control register 20 (TMC20) Port mode register 5 (PM5)

Figure 6-1. Block Diagram of 16-Bit Timer 20



(1) 16-bit compare register 20 (CR20)

This register compares the value set to CR20 with the count value of 16-bit timer counter 20 (TM20), and when they match, generates an interrupt request (INTTM2).

CR20 is set with a 16-bit memory manipulation instruction. The 0000H to FFFFH values can be set.

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

Cautions 1. This register is manipulated with a 16-bit memory manipulation instruction, however an 8-bit memory manipulation instruction can be used. When manipulated with an 8-bit memory manipulation instruction, accessing method should be direct addressing. This register can be accessed only in short direct addressing mode when a 16-bit memory manipulation instruction is used.

2. When rewriting CR20 during count operation, set CR20 to interrupt disable from interrupt mask flag register 0 (MK0) beforehand. Beside, set the timer output data to inversion disable by 16-bit timer mode control register 20 (TMC20).

★ **If CR20 is rewritten with the interrupt enabled, an interrupt request may be issued immediately.**

(2) 16-bit timer counter 20 (TM20)

This is a 16-bit register that counts count pulses.

TM20 is read with a 16-bit memory manipulation instruction.

This register is in free running during count clock input.

$\overline{\text{RESET}}$ input clears this register to 0000H and after that to be in free running.

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

2. This register is manipulated with a 16-bit memory manipulation instruction, however an 8-bit memory manipulation instruction can be used. When manipulated with an 8-bit memory manipulation instruction, accessing method should be direct addressing. This register can be accessed only in short direct addressing mode when a 16-bit memory manipulation instruction is used.

3. When manipulated with an 8-bit memory manipulation instruction, readout should be performed in the order from low-order byte to high-order byte and must be in pairs.

(3) 16-bit capture register 20 (TCP20)

This is a 16-bit register that captures the contents of 16-bit timer counter 20 (TM20).

TCP20 is set with a 16-bit memory manipulation instruction.

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

Caution This register is manipulated with a 16-bit memory manipulation instruction, however an 8-bit memory manipulation instruction can be used. When manipulated with an 8-bit memory manipulation instruction, accessing method should be direct addressing. This register can be accessed only in short direct addressing mode when a 16-bit memory manipulation instruction is used.

(4) 16-bit counter read buffer

This buffer latches a counter value and retains a count value of 16-bit timer counter 20 (TM20).

6.3 Registers Controlling 16-Bit Timer

The following two types of registers control 16-bit timer 20.

- 16-bit timer mode control register 20 (TMC20)
- Port mode register 5 (PM5)

(1) 16-bit timer mode control register 20 (TMC20)

16-bit timer mode control register 20 (TMC20) controls the setting of a count clock, capture edge, etc.
TMC20 is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears TMC20 to 00H.

Figure 6-2. 16-Bit Timer Mode Control Register 20 Format

Symbol	7	<6>	5	4	3	2	1	<0>	Address	After Reset	R/W
TMC20	TOD20	TOF20	CPT201	CPT200	TOC20	TCL201	TCL200	TOE20	FF5BH	00H	R/W[Note]

TOD20	Timer Output Data
0	Timer output data is 0.
1	Timer output data is 1.

TOF20	Overflow Flag Set
0	Clear by reset and software
1	Set by overflow of 16-bit timer

CPT201	CPT200	Capture Edge Selection
0	0	Capture operation disabled
0	1	Rising edge of CPT2
1	0	Falling edge of CPT2
1	1	Both edges of CPT2

TOC20	Timer Output Data Inverse Control
0	Inverse disabled
1	Inverse enabled

TCL201	TCL200	16-bit Timer Counter 20 Count Clock Selection
0	0	$f_X/2^2$ (1.25 MHz)
0	1	$f_X/2^6$ (78.1 kHz)
Other than above		Setting prohibited

TOE20	16-bit Timer 20 Output Control
0	Output disabled (port mode)
1	Output enabled

Note Bit 7 is read-only.

Remarks **1.** f_X: System clock oscillation frequency

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

(2) Port mode register 5 (PM5)

This register sets the input/output of port 5 in 1-bit units.

To use the P51/TO2 pin for timer output, set the output latch of PM51 and P51 to 0.

PM5 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM5 to FFH.

Figure 6-3. Port Mode Register 5 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM5	1	1	1	1	PM53	PM52	PM51	PM50	FF25H	FFH	R/W

PM51	P51 Pin Input/Output Selection
0	Output mode (output buffer on)
1	Input mode (output buffer off)

6.4 16-Bit Timer Operation

6.4.1 Operation as timer interrupt

In the timer interrupt function, interrupts are repeatedly generated at the count value set to 16-bit compare register 20 (CR20) in advance based on the intervals of the value set in TCL201 and TCL200.

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

- Set count values in CR20
- Set 16-bit timer mode control register 20 (TMC20) as shown in Figure 6-4.

Figure 6-4. Settings of 16-Bit Timer Mode Control Register 20 at Timer Interrupt Operation

★

	TOD20	TOF20	CPT201	CPT200	TOC20	TCL201	TCL200	TOE20
TMC20	-	0/1	0/1	0/1	0/1	0	0/1	0/1

Setting of Count Clock (see **Table 6-2**)

Caution If 0 is set both to CPT201 and CPT200 flags, capture edge becomes setting prohibited.

When the count value of 16-bit timer counter 20 (TM20) coincides with the value set to CR20, counting of TM20 continues and an interrupt request signal (INTTM2) is generated.

Table 6-2 shows interval time, and Figure 6-5 shows timing of timer interrupt operation.

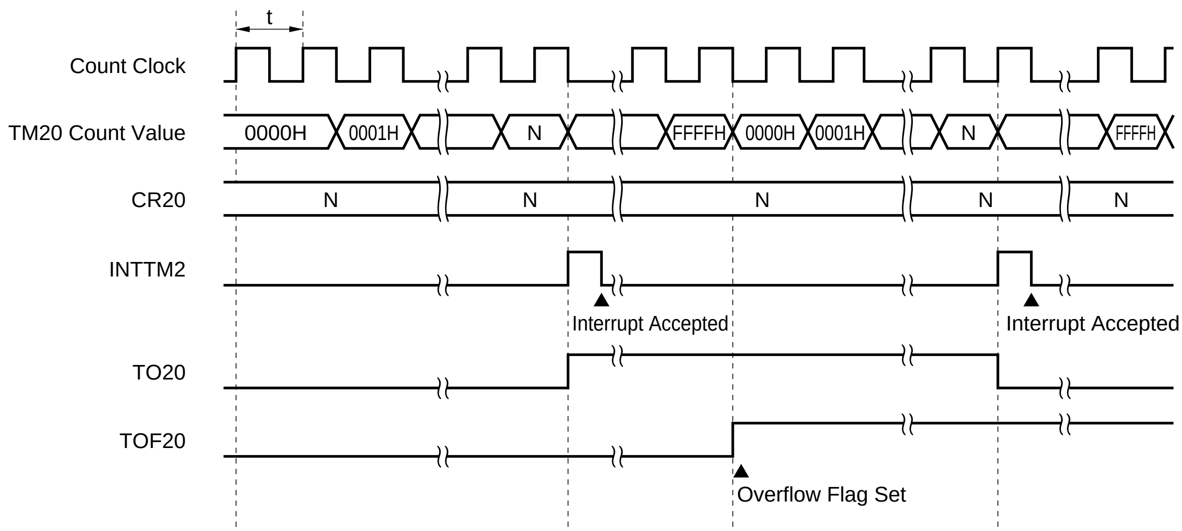
★ **Caution Perform the following processing when rewriting CR20 during count operation.**
<1> Disable the interrupt (TMMK20 (bit 7 of interrupt mask flag register 0 (MK0)) = 1).
<2> Disable inversion control of timer output data (TOC20 = 0).
If CR20 is rewritten with the interrupt enabled, an interrupt request may be issued immediately.

Table 6-2. Interval Time of 16-Bit Timer 20

TCL201	TCL200	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)
Other than above		Setting prohibited	

Remarks 1. f_X: System clock oscillation frequency
2. The parenthesized values apply to operation at f_X = 5.0 MHz.

Figure 6-5. Timer Interrupt Operation Timing



Remark N = 0000H to FFFFH

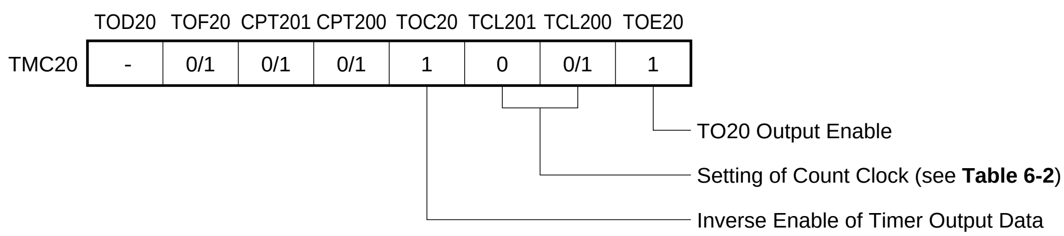
6.4.2 Operation as timer output

Timer outputs are repeatedly generated at the count value set to 16-bit compare register 20 (CR20) in advance based on the intervals of the value set in TCL201 and TCL200.

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

- Set P51 to output mode (PM51 = 0).
- Set 0 to the output latch of P51.
- Set the count value in CR20.
- Set 16-bit timer mode control register 20 (TMC20) as shown in Figure 6-6.

Figure 6-6. Settings of 16-Bit Timer Mode Control Register 20 at Timer Output Operation

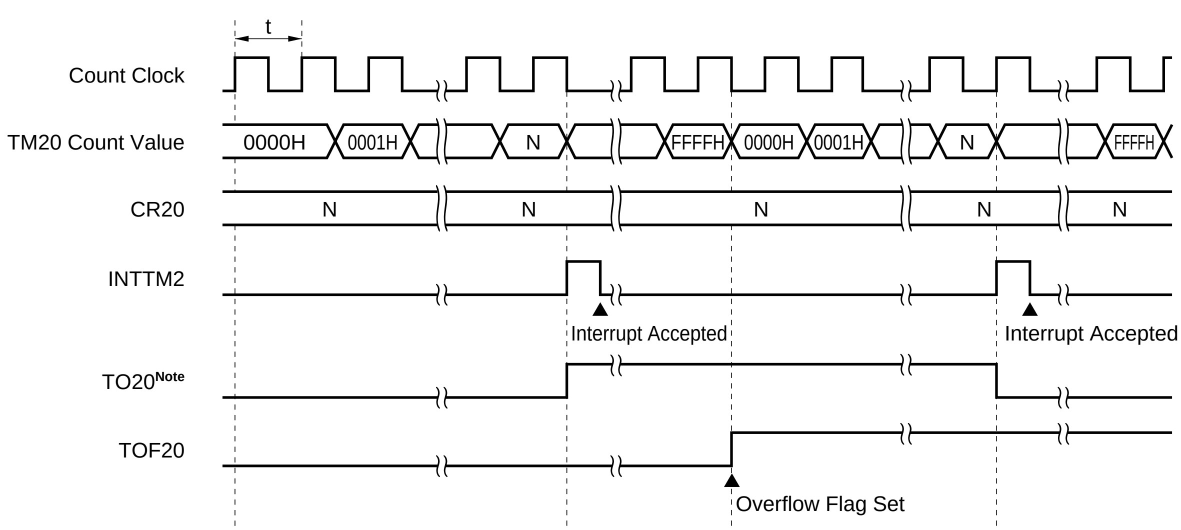


Caution If both CPT201 flag and CPT200 flag are set to 0, the capture edge becomes operation prohibited.

When the count value of 16-bit timer counter 20 (TM20) matches the value set in CR20, the output status of the TO2/P51 pin is inverted. This enables timer output. At that time, TM20 count is continued and an interrupt request signal (INTTM2) is generated.

Figure 6-7 shows the timing of timer output (see Table 6-2 for the interval time of the 16-bit timer).

Figure 6-7. Timer Output Timing



Note The TO20 initial value becomes low level during output enable (TOE20 = 1).

Remark N = 0000H to FFFFH

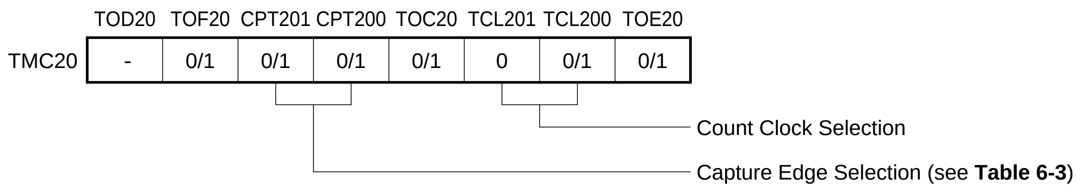
6.4.3 Capture operation

The capture operation functions to capture and latch the count value of 16-bit timer counter 20 (TM20) synchronizing with a capture trigger.

Set as shown in Figure 6-8 to allow the 16-bit timer to start the capture operation.

Figure 6-8. Setting Contents of 16-Bit Timer Mode Control Register 20 during Capture Operation

★



16-bit capture register 20 (TCP20) starts capture operation after being detected a CPT20 capture trigger edge, and latches and retains the count value of 16-bit timer counter 20. TCP20 fetches count value within 2 clocks and retains the count value until the next capture edge detection.

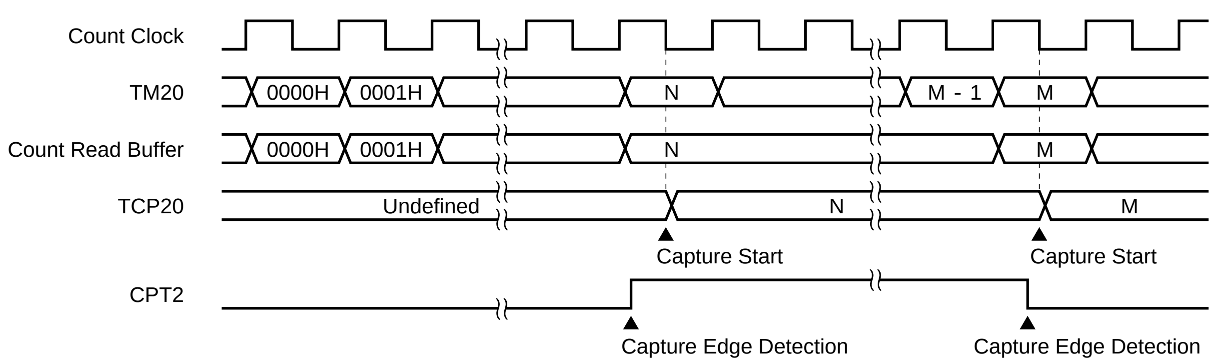
Table 6-3 and Figure 6-9 show the setting contents of capture edge and capture operation timing, respectively.

Table 6-3. Setting Contents of Capture Edge

CPT201	CPT200	Capture Edge Selection
0	0	Capture operation prohibited
0	1	CPT2 pin rising edge
1	0	CPT2 pin falling edge
1	1	CPT2 pin both edges

Caution Because TCP20 is rewritten when a capture trigger edge is detected during TCP20 read, disable the capture trigger detection during TCP20 read.

Figure 6-9. Capture Operation Timing (Both Edges of CPT2 Pin are Specified)



6.4.4 16-bit timer counter 20 readout

The count value of 16-bit timer counter 20 (TM20) is read out by a 16-bit manipulation instruction.

TM20 readout is performed through a counter read buffer. The counter read buffer latches the TM20 count value. And buffer operation is pended at the CPU clock falling edge after the read signal of the TM20 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 TM20 higher byte falls.

$\overline{\text{RESET}}$ input clears TM20 to 0000H and starts freerunning.

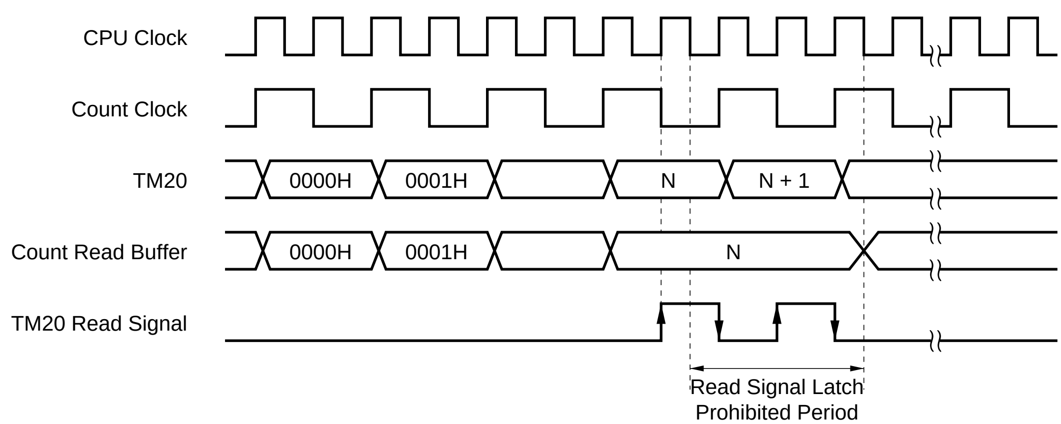
Figure 6-10 shows the timing of 16-bit timer counter 20 readout.

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

2. Though TM20 is a dedicated register of a 16-bit transfer instruction, an 8-bit transfer instruction can be used.
When using the 8-bit transfer instruction, execute by direct addressing.

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

Figure 6-10. 16-Bit Timer Counter 20 Readout Timing



[MEMO]

7.1 8-Bit Timer/Event Counter Functions

8-bit timer/event counter 00 (TM00) has the following functions:

- Interval timer
- External event counter
- Square wave output

(1) 8-bit interval timer

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

Table 7-1. Interval Time of 8-Bit Timer/Event Counter 00

Minimum Interval Time	Maximum Interval Time	Resolution
$1/f_X$ (200 ns)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (200 ns)
$2^5/f_X$ (6.4 μs)	$2^{13}/f_X$ (1.64 ms)	$2^5/f_X$ (6.4 μs)

Remarks 1. f_X: System clock oscillation frequency

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

(2) External event counter

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

(3) Square wave output

A square wave of arbitrary frequency can be output.

Table 7-2. Square Wave Output Range of 8-Bit Timer/Event Counter 00

Minimum Pulse Width	Maximum Pulse Width	Resolution
$1/f_X$ (200 ns)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (200 ns)
$2^5/f_X$ (6.4 μs)	$2^{13}/f_X$ (1.64 ms)	$2^5/f_X$ (6.4 μs)

Remarks 1. f_X: System clock oscillation frequency

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

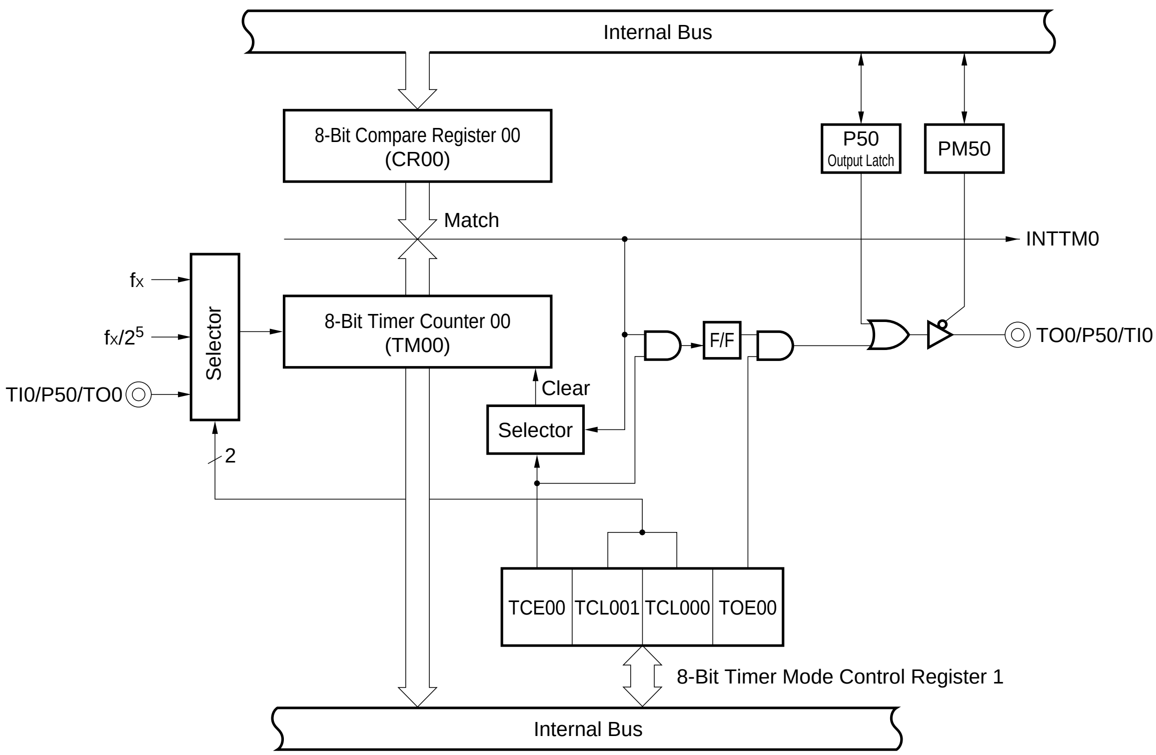
7.2 8-Bit Timer/Event Counter Configuration

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

Table 7-3. Configuration of 8-Bit Timer/Event Counter 00

Item	Configuration
Timer counter	8 bits × 1 (TM00)
Register	Compare register: 8 bits × 1 (CR00)
Timer output	1 (TO0)
Control register	8-bit timer mode control register 00 (TMC00) Port mode register 5 (PM5)

Figure 7-1. Block Diagram of 8-Bit Timer/Event Counter 00



(1) 8-bit compare register 00 (CR00)

This is an 8-bit register to compare the value set to CR00 with the 8-bit timer register 00 (TM00) count value, and if they match, generates an interrupt request (INTTM0).

CR00 is set with an 8-bit memory manipulation instruction. The 00H to FFH values can be set.

$\overline{\text{RESET}}$ input makes CR00 undefined.

★ **Caution Before rewriting CR00, stop the timer operation. If CR00 is rewritten while the timer operation is enabled, the coincidence interrupt request signal may be generated immediately.**

(2) 8-bit timer counter 00 (TM00)

This is an 8-bit register to count pulses.

TM00 is read with an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TM00 to 00H.

7.3 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 register 00 (TMC00)
- Port mode register 5 (PM5)

(1) 8-bit timer mode control register 00 (TMC00)

TMC00 determines whether to enable or disable 8-bit timer counter 00 (TM00), specifies the count clock for TM00, and controls the operation of the output control circuit of 8-bit timer/event counter 00.

TMC00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TMC00 to 00H.

Figure 7-2. 8-Bit Timer Mode Control Register 00 Format

Symbol	<7>	6	5	4	3	2	1	<0>	Address	After Reset	R/W
TMC00	TCE00	0	0	0	0	TCL001	TCL000	TOE00	FF53H	00H	R/W

TCE00	8-Bit Timer Counter 00 Operation Control
0	Operation disabled (TM00 is cleared to 0.)
1	Operation enabled

TCL001	TCL000	8-Bit Timer Counter 00 Count Clock Selection
0	0	f_X (5.0 MHz)
0	1	$f_X/2^5$ (156 kHz)
1	0	Rising edge of TI0Note
1	1	Falling edge of TI0Note

TOE00	8-Bit Timer/Event Counter 00 Output Control
0	Output disabled (port mode)
1	Output enabled

Note When inputting a clock signal eternally, timer output cannot be used.

Caution Always stop the timer before setting TMC00.

Remarks 1. f_X: System clock oscillation frequency

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

(2) Port mode register 5 (PM5)

This register sets port 5 input/output in 1-bit units.

When using the P50/TI0/TO0 pin for timer output, set PM50 and the output latch of P50 to 0.

PM5 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets PM5 to FFH.

Figure 7-3. Port Mode Register 5 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
PM5	1	1	1	1	PM53	PM52	PM51	PM50	FF25H	FFH	R/W

PM50	P50 Pin Input/Output Mode Selection
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

7.4 8-Bit Timer/Event Counter Operation

★ 7.4.1 Operation as interval timer

Interval timer repeatedly generates an interrupt at time intervals specified by the count value set to 8-bit compare register 00 (CR00) in advance.

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

<1> Disable the operation of 8-bit timer counter 00 (TM00) (TCE00 (bit 7 of 8-bit timer mode control register 00 (TMC00)) = 0).
<2> Set the count clock of the 8-bit timer/event counter (see **Table 7-4**).
<3> Set a count value in CR00.
<4> Enable the operation of TM00 (TCE00 = 1).

When the count value of the 8-bit timer counter 00 (TM00) coincides with the value set to CR00, the value of TM00 is cleared to 0 and TM00 continues counting. At the same time, an interrupt request signal (INTTM0) is generated.

Table 7-4 shows interval time, and Figure 7-4 shows the timing of interval timer operation.

Cautions 1. Before rewriting CR00, stop the timer operation. If CR00 is rewritten while the timer operation is enabled, the coincidence interrupt request signal may be generated immediately.

2. If setting the count clock in TMC00 and enabling the operation of TM00 are performed at the same time with an 8-bit memory manipulation instruction, the error one cycle after the timer has been started may exceed one clock. To use the 8-bit timer/event counter as an interval timer, therefore, perform the setting in the above sequence.

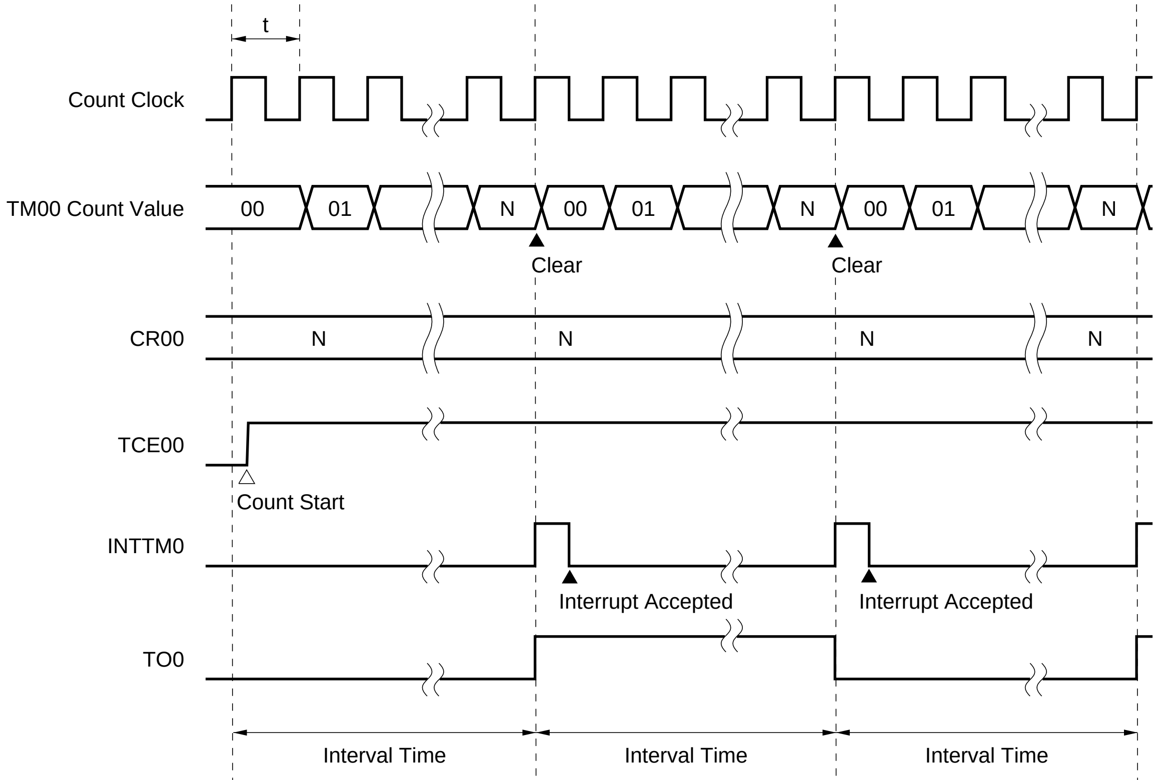
Table 7-4. Interval Time of 8-Bit Timer/Event Counter 00

TCL001	TCL000	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	$2^5/f_X$ (6.4 μs)	$2^{13}/f_X$ (1.64 ms)	$2^5/f_X$ (6.4 μs)
1	0	TI0 input cycle	$2^8 \times$ TI0 input cycle	TI0 input edge cycle
1	1	TI0 input cycle	$2^8 \times$ TI0 input cycle	TI0 input edge cycle

Remarks 1. f_X: System clock oscillation frequency

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

Figure 7-4. Interval Timer Operation Timing



Remark Interval time = (N + 1) × t
where N = 00H to FFH

★ ### 7.4.2 Operation as external event counter

The external event counter counts the number of external clock pulses input to the TI0/P50/TO0 pin by using timer counter 00 (TM00).

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

<1> Set P50 to input mode (PM50 = 1).
<2> Disable the operation of 8-bit timer counter 00 (TM00) (TCE00 (bit 7 of 8-bit timer mode control register 00 (TMC00)) = 0).
<3> Specify the rising or falling edge of TI0 (see **Table 7-4**).
<4> Set a count value in CR00.
<5> Enable the operation of TM00 (TCE00 = 1).

Each time the valid edge specified by the bit 1 or 2 (TCL001 or TCL000) of TMC00 is input, the value of 8-bit timer counter 00 (TM00) is incremented.

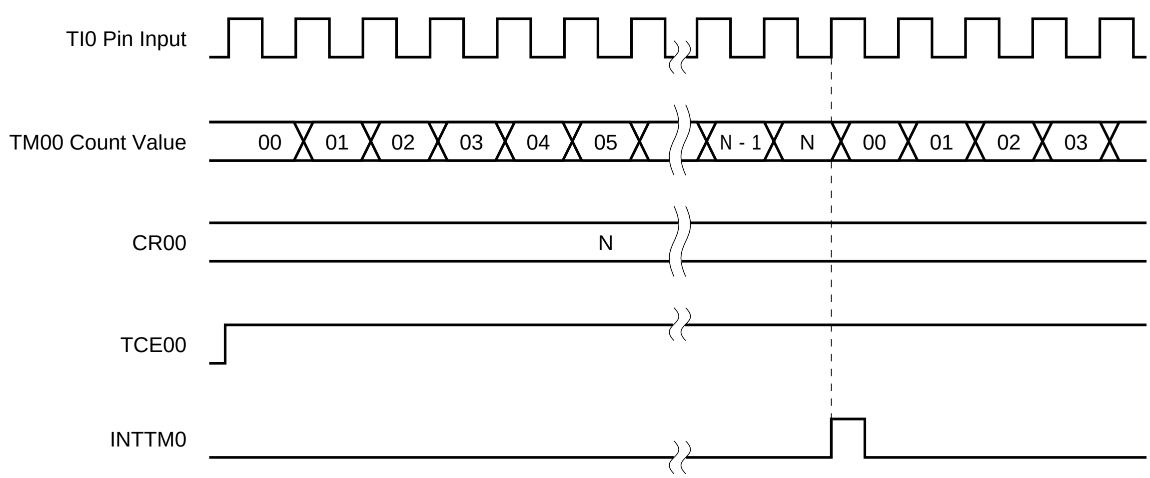
When the count value of TM00 coincides with the value set to CR00, the value of TM00 is cleared to 0 and TM00 continues counting. At the same time, an interrupt request signal (INTTM0) is generated.

Figure 7-5 shows the timing of external event counter operation (with rising edge specified).

Cautions 1. Before rewriting CR00, stop the timer operation. If CR00 is rewritten while the timer operation is enabled, the coincidence interrupt request signal may be generated immediately.

2. If setting the count clock in TMC00 and enabling the operation of TM00 are performed at the same time with an 8-bit memory manipulation instruction, the error one cycle after the timer has been started may exceed one clock. To use the 8-bit timer/event counter as an external event counter, therefore, perform the setting in the above sequence.

Figure 7-5. External Event Counter Operation Timing (with Rising Edge Specified)



Remark N = 00H to FFH

★ 7.4.3 Operation as square wave output

The 8-bit timer/event counter can generate the output square waves of arbitrary frequency at intervals specified by the count value set to 8-bit compare register 00 (CR00) in advance.

To operate 8-bit timer/event counter 00 as square wave output, the following settings are required.

<1> Set P50 to output mode (PM50 = 0).
<2> Set 0 for the output latch of P50.
<3> Disable the operation of 8-bit timer counter 00 (TM00) (TCE00 (bit 7 of 8-bit timer mode control register 00 (TMC00)) = 1).
<4> Set a count clock for 8-bit timer/event counter 00 and enable output of TO0 (TOE00 (bit 0 of TMC00) = 1).
<5> Set a count value in CR00.
<6> Enable the operation of TM00 (TCE00 = 1).

When the count value of 8-bit timer counter 00 (TM00) matches the value set in CR00, the TO0/P50/TI0 pin output will be inverted, respectively. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurred, the TM00 value will be cleared to 0 then resume to count, generating an interrupt request signal (INTTM0).

Setting 0 to the bit 7 in TMC00, that is, TCE00 makes the square-wave output clear to 0.

Table 7-5 lists square wave output range, and Figure 7-6 shows timing of square wave output.

Cautions 1. Before rewriting CR00, stop the timer operation. If CR00 is rewritten while the timer operation is enabled, the coincidence interrupt request signal may be generated immediately.

2. If setting the count clock in TMC00 and enabling the operation of TM00 are performed at the same time with an 8-bit memory manipulation instruction, the error one cycle after the timer has been started may exceed one clock. To use the 8-bit timer/event counter as a square wave output, therefore, perform the setting in the above sequence.

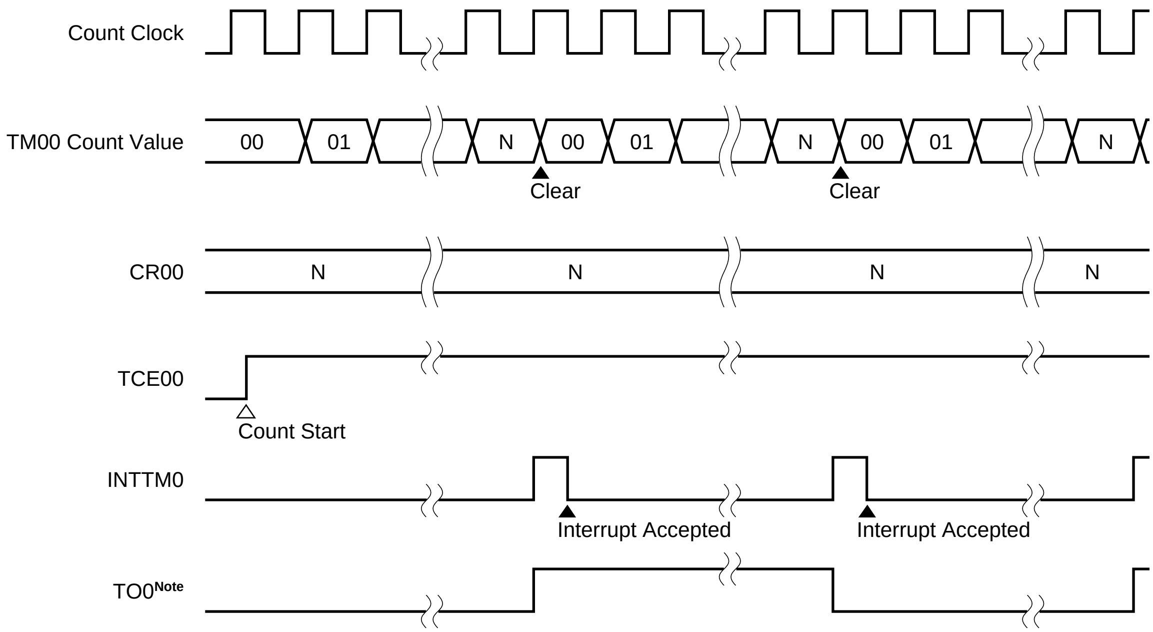
Table 7-5. Square Wave Output Range of 8-Bit Timer/Event Counter 00

TCL001	TCL000	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	$2^5/f_X$ (6.4 μs)	$2^{13}/f_X$ (1.64 ms)	$2^5/f_X$ (6.4 μs)

Remarks 1. f_X: System clock oscillation frequency

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

Figure 7-6. Square Wave Output Timing



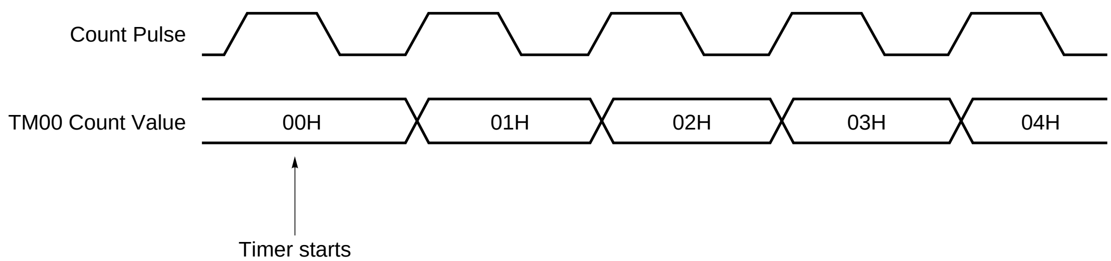
Note The initial value of TO0 at output enable (TOE00 = 1) becomes low-level.

7.5 Notes on Using 8-Bit Timer/Event Counters

(1) Error on starting timer

An error of up to 1 clock occurs after the timer has been started until a coincidence signal is generated. This is because 8-bit timer counter 00 (TM00) started asynchronously with the count pulse.

Figure 7-7. 8-Bit Timer Counter 00 Start Timing

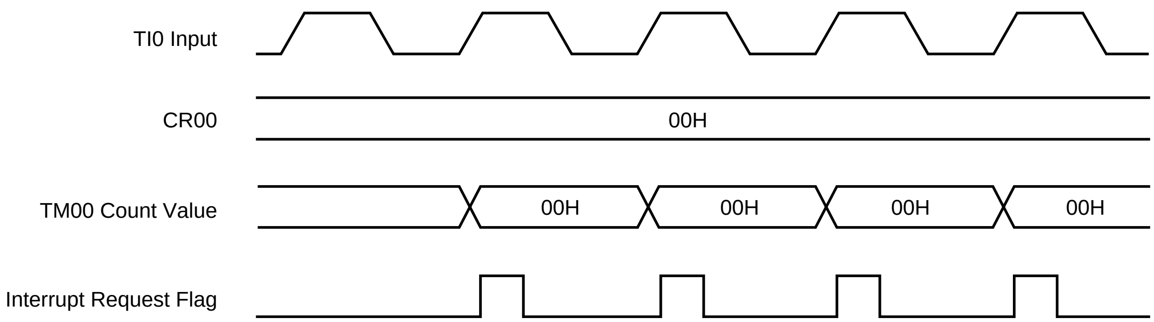


(2) Setting of 8-bit compare register

8-bit compare register 00 (CR00) can be set to 00H.

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

Figure 7-8. External Event Counter Operation Timing



8.1 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 the $\overline{\text{RESET}}$ signal can be generated.

Table 8-1. 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: System clock oscillation frequency

(2) Interval timer

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

Table 8-2. 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: System clock oscillation frequency

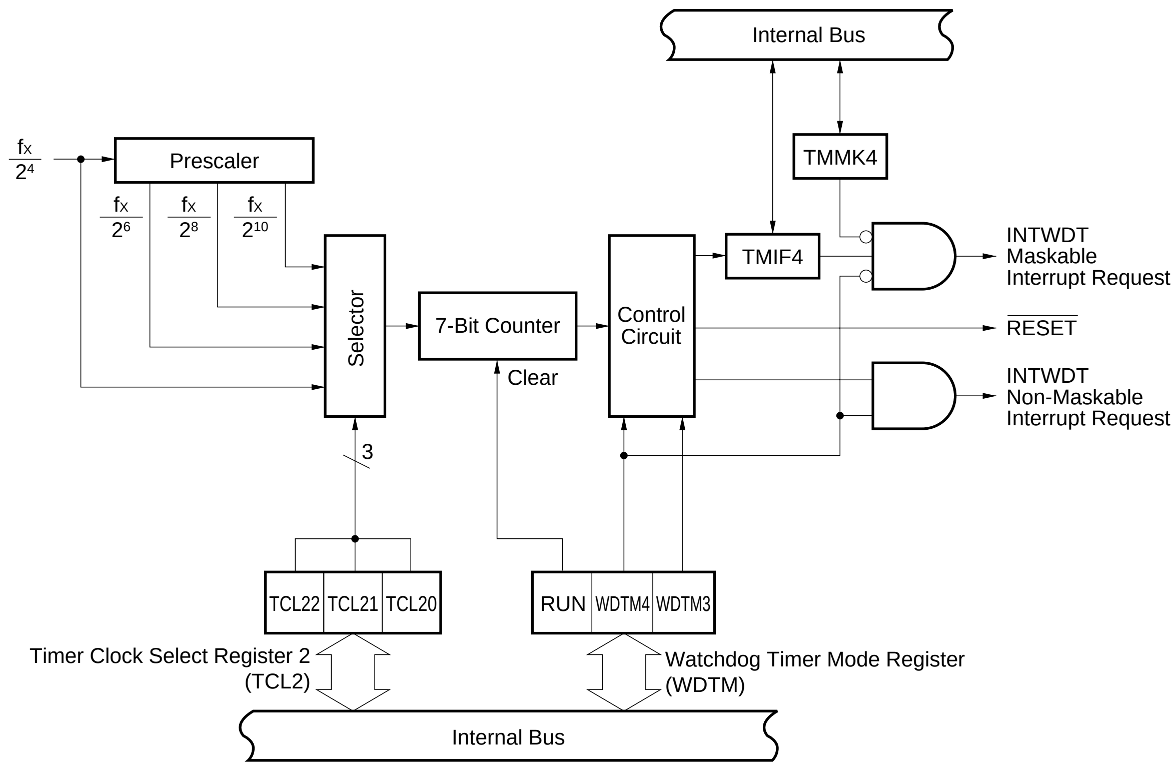
8.2 Watchdog Timer Configuration

The watchdog timer consists of the following hardware.

Table 8-3. Configuration of Watchdog Timer

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

Figure 8-1. Block Diagram of Watchdog Timer



8.3 Watchdog Timer Control Registers

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

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

(1) Timer clock select 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 8-2. Timer Clock Select Register 2 Format

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 Time
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: 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 8-3. Watchdog Timer Mode Register Format

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

RUN	Selects Operation of Watchdog Timer[Note 1]
0	Stops counting.
1	Clears counter and starts counting.

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

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

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

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

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

2. In watchdog timer mode 1 or 2, set WDTM4 to 1 after confirming TMIF4 (bit 0 of interrupt request flag register 0 (IF0)) being set to 0. When watchdog timer mode 1 or 2 is selected under the condition where TMIF4 is 1, a non-maskable interrupt occurs at the completion of rewriting.

8.4 Operation of Watchdog Timer

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

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

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

Table 8-4. 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: System clock oscillation frequency

8.4.2 Operation as interval timer

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

Select a count clock (or interval time) by setting bits 0 to 2 (TCL20 to TCL22) of timer clock select 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 the 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 the HALT mode, but stops in the STOP mode. Therefore, set RUN to 1 before entering the STOP mode to clear the interval timer, and then execute the STOP instruction.

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

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

Table 8-5. Interval Time of Interval Timer

TCL22	TCL21	TCL20	Interval 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: System clock oscillation frequency

CHAPTER 9 SERIAL INTERFACE 00

9.1 Serial Interface 00 Functions

Serial interface 00 employs 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 carried out. It enables power consumption reduction.

(2) Asynchronous serial interface (UART) mode

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

A UART-dedicated baud rate generator is incorporated, allowing communication over a wide range of baud rates. In addition, the baud rate can be defined by scaling the input clock to the ASCK pin.

(3) 3-wire serial I/O mode (MSB/LSB start bit switchable)

In this mode, 8-bit data transfer is carried out with three lines, one for serial clock ($\overline{SCK0}$) and two for serial data (SI0, SO0).

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

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

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

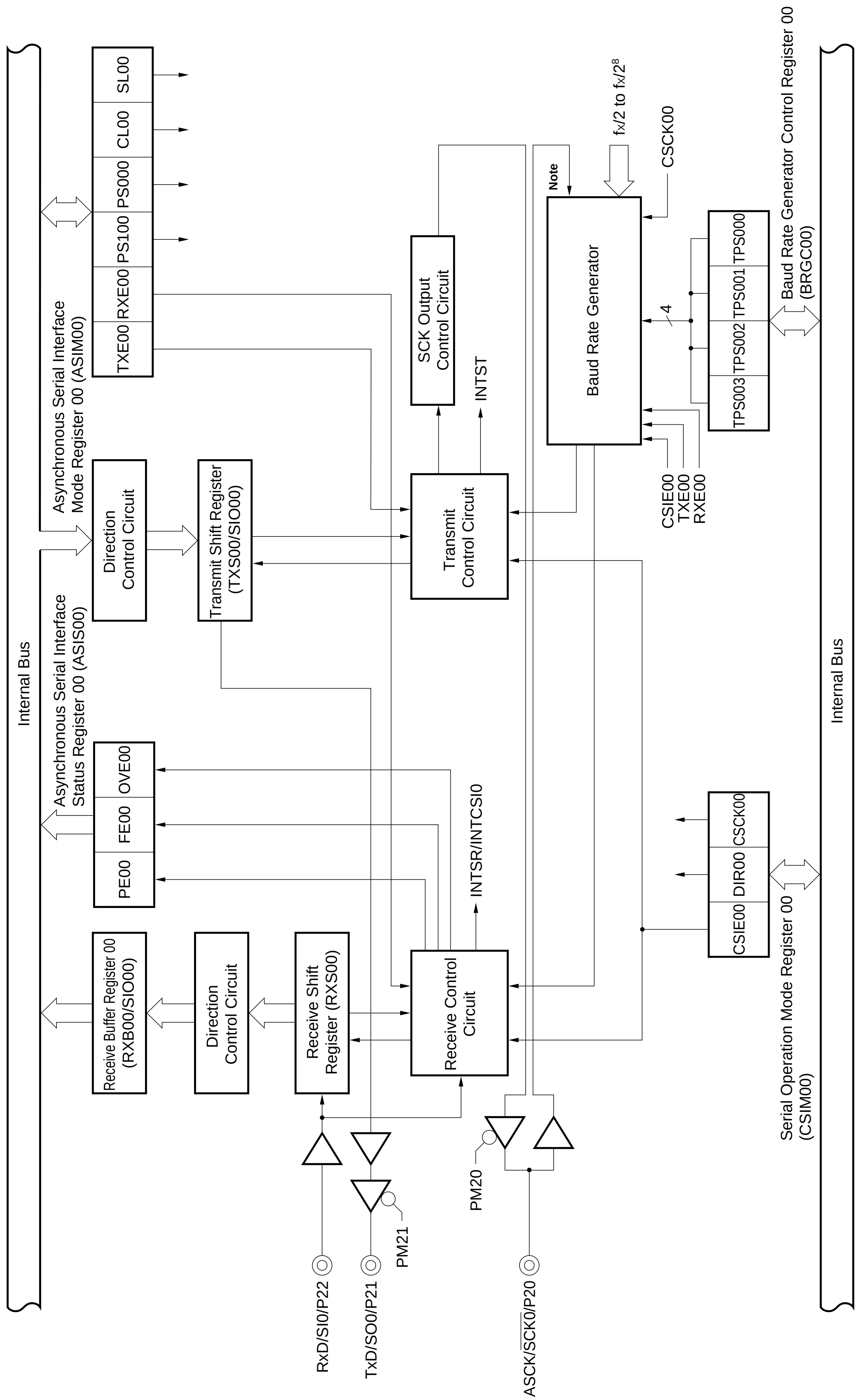
9.2 Serial Interface 00 Configuration

Serial interface 00 has the following hardware configuration.

Table 9-1. Configuration of Serial Interface 00

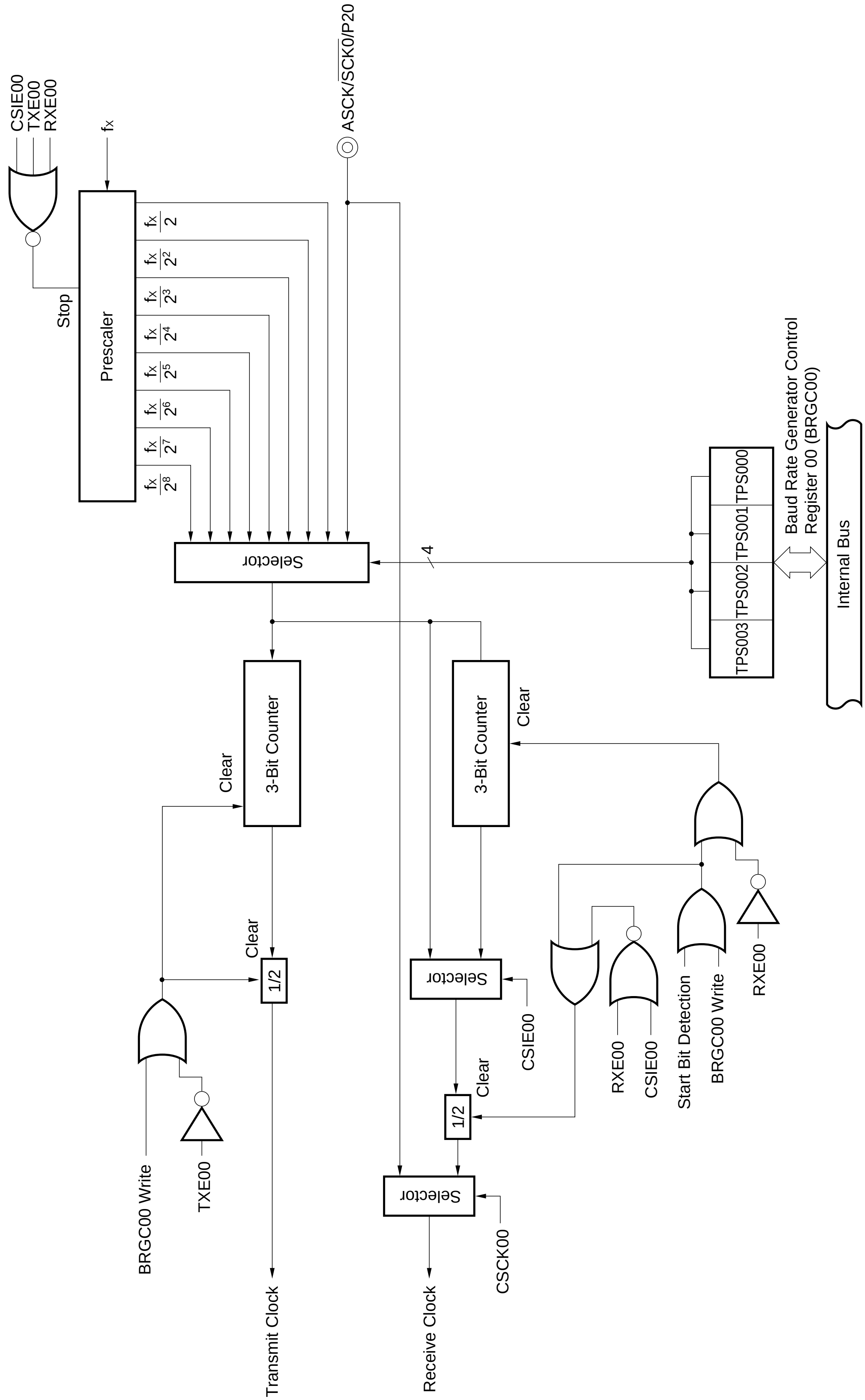
Item	Configuration
Register	Transmission shift register 00 (TXS00) Receive shift register 00 (RXS00) Receive buffer register 00 (RXB00)
Control register	Serial operation mode register 00 (CSIM00) Asynchronous serial interface mode register 00 (ASIM00) Asynchronous serial interface status register 00 (ASIS00) Baud rate generator control register 00 (BRGC00)

Figure 9-1. Block Diagram of Serial Interface 00



Note For the configuration of the baud rate generator, see Figure 9-2.

Figure 9-2. Block Diagram of Baud Rate Generator



(1) Transmit shift register 00 (TXS00)

This register is used to specify data to be transmitted. Data written to TXS00 is transmitted as serial data.

If the data length is specified as 7 bits, bits 0 to 6 of the data written to TXS00 are transferred as the transmit data. The transmit operation is started by writing data to TXS00.

TXS00 is written to with an 8-bit memory manipulation instruction. It cannot be read.

$\overline{\text{RESET}}$ input sets TXS00 to FFH.

Caution Do not write to TXS00 during transmission.
TXS00 and receive buffer register 00 (RXB00) are allocated to the same address, and when reading is performed, RXB00 values are read.

(2) Receive shift register 00 (RXS00)

This register is used to convert serial data input to the RxD pin into parallel data. Each time one byte of data is received, it is transferred to receive buffer register 00 (RXB00).

RXS00 cannot be manipulated directly by program.

(3) Receive buffer register 00 (RXB00)

This register is used to hold received data. Each time one byte of data is received, a new byte of data is transferred from receive shift register 00 (RXS00).

If the data length is specified as 7 bits, receive data is transferred to bits 0 to 6 of RXB00, and the MSB of RXB00 always becomes 0.

RXB00 can be read with an 8-bit memory manipulation instruction. It cannot be written to.

$\overline{\text{RESET}}$ input makes RXB00 undefined.

Caution RXB00 and transmit shift register 00 (TXS00) are allocated to the same address, and when writing is performed, the values are written to TXS00.

(4) Transmit control circuit

This circuit controls transmit operations by adding a start bit, parity bit, and stop bit to data written to transmit shift register 00 (TXS00), according to the data set to asynchronous serial interface mode register 00 (ASIM00).

(5) Receive control circuit

This circuit controls receive operations according to the data set to asynchronous serial interface mode register 00 (ASIM00). It performs also parity error check, etc., during receive operations, and when an error is detected, it sets the value to asynchronous serial interface status register 00 (ASIS00) depending on the nature of the error.

9.3 Serial Interface 00 Control Register

The following four types of registers are used to control serial interface 00.

- Serial operation mode register 00 (CSIM00)
- Asynchronous serial interface mode register 00 (ASIM00)
- Asynchronous serial interface status register 00 (ASIS00)
- Baud rate generator control register 00 (BRGC00)

(1) Serial operation mode register 00 (CSIM00)

This register is set when using serial interface 00 in the 3-wire serial I/O mode.
CSIM00 is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears CSIM00 to 00H.

Figure 9-3. Serial Operation Mode Register 00 Format

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM00	CSIE00	0	0	0	0	DIR00	CSCK00	0	FF72H	00H	R/W

CSIE00	Operation Control in 3-Wire Serial I/O Mode
0	Operation stop
1	Operation enable

DIR00	Start Bit Specification
0	MSB
1	LSB

CSCK00	Clock Selection in 3-Wire Serial I/O Mode
0	Input clock to $\overline{\text{SCK0}}$ pin from external
1	Dedicated baud rate generator output

Cautions 1. Be sure to set bit 0 and bits 3 to 6 to 0.

2. Set 00H to CSIM00 at the UART mode.

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

This register is set when using serial interface 00 in the asynchronous serial interface mode.

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

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

Figure 9-4. Asynchronous Serial Interface Mode Register 00 Format

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

TXE00	Transmit Operation Control
0	Transmit operation stop
1	Transmit operation enable

RXE00	Receive Operation Control
0	Receive operation stop
1	Receive operation enable

PS001	PS000	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

CL00	Character Length Specification
0	7 bits
1	8 bits

SL00	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

Cautions 1. Be sure to set bits 0 and 1 to 0.

2. Set 00H to ASIM00 at the 3-wire serial I/O mode.

3. Switching operating modes must be performed after the halt of serial transmit/receive operation.

Table 9-2. Serial Interface 00 Operating Mode Settings

(1) Operation stop mode

ASIM00		CSIM00			PM22	P22	PM21	P21	PM20	P20	Start Bit	Shift Clock	P22/SI0/RxD Pin Function	P21/SO0/TxD Pin Function	P20/$\overline{SCK0}$/ASCK Pin Function
TXE00	RXE00	CSIE00	DIR00	CSCK00											
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

ASIM00		CSIM00			PM22	P22	PM21	P21	PM20	P20	Start Bit	Shift Clock	P22/SI0/RxD Pin Function	P21/SO0/TxD Pin Function	P20/$\overline{SCK0}$/ASCK Pin Function
TXE00	RXE00	CSIE00	DIR00	CSCK00											
0	0	1	0	0	1[Note 2]	×[Note 2]	0	1	1	×	MSB	External clock	SI0[Note 2]	SO0 (CMOS output)	$\overline{SCK0}$ input
				1					0	1		Internal clock			$\overline{SCK0}$ output
		1	1	0					1	×	LSB	External clock			$\overline{SCK0}$ input
				1					0	1		Internal clock			$\overline{SCK0}$ output
Other than above											Setting prohibited				

(3) Asynchronous serial interface mode

ASIM00		CSIM00			PM22	P22	PM21	P21	PM20	P20	Start Bit	Shift Clock	P22/SI0/RxD Pin Function	P21/SO0/TxD Pin Function	P20/$\overline{SCK0}$/ASCK Pin Function
TXE00	RXE00	CSIE00	DIR00	CSCK00											
1	0	0	0	0	×[Note 1]	×[Note 1]	0	1	1	×	LSB	External clock	P22	TxD (CMOS output)	ASCK input
									×[Note 1]	×[Note 1]		Internal clock			P20
0	1	0	0	0	1	×	×[Note 1]	×[Note 1]	1	×		External clock	RxD	P21	ASCK input
									×[Note 1]	×[Note 1]		Internal clock			P20
1	1	0	0	0	1	×	0	1	1	×		External clock		TxD (CMOS output)	ASCK input
									×[Note 1]	×[Note 1]		Internal clock			P20
Other than above											Setting prohibited				

Notes 1. Can be used as port function.

2. If used only for transmission, can be used as P22 (CMOS input/output).

Remark ×: Don't care.

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

This register indicates types of error when a reception error is generated in the asynchronous interface mode.

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

The contents of ASIS00 become undefined in the 3-wire serial I/O mode.

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

Figure 9-5. Asynchronous Serial Interface Status Register 00 Format

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

PE00	Parity Error Flag
0	Parity error not generated
1	Parity error generated (when the parity of transmit data does not coincide.)

FE00	Flaming Error Flag
0	Flaming error not generated
1	Flaming error generated (when stop bit is not detected.)[Note 1]

OVE00	Overrun Error Flag
0	Overrun error not generated
1	Overrun error generated[Note 2] (when the next receive operation is completed before the data is read from the receive buffer register.)

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

2. When an overrun error occurs, be sure to read out receive buffer register 00 (RXB00). Unless RXB00 is read out, overrun errors occur at each data reception.

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

This register is used to set the serial clock of serial interface 00.

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

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

Figure 9-6. Baud Rate Generator Control Register 00 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
BRGC00	TPS003	TPS002	TPS001	TPS000	0	0	0	0	FF73H	00H	R/W

TPS003	TPS002	TPS001	TPS000	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	Input clock from external to ASCK pin[Note]	–
Other than above				Setting prohibited	

Note Only used in UART mode.

Cautions 1. When writing to BRGC00 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 BRGC00 during communication operation.

2. Be sure not to select n = 1 during an operation at f_X = 5.0 MHz because n = 1 exceeds the baud rate limit.

3. When selecting an input clock from an external source, set port mode register 2 (PM2) to the input mode.

Remarks 1. f_X : System clock oscillation frequency

2. n : Value determined by setting TPS000 through TPS003 ($1 \leq n \leq 8$)

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 from the ASCK pin.

(a) Generation of baud rate transmit/receive clock by means of system clock

The transmit/receive clock is generated by scaling the system clock. The baud rate generated from the system clock is found from the following expression.

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1} \times 8} \text{ [Hz]}$$

f_X : System clock oscillation frequency

n : Value determined by values of TPS000 through TPS003 as shown in Figure 9-6 ($2 \leq n \leq 8$)

Table 9-3. Example of Relationship between System Clock and Baud Rate

Baud Rate (bps)	n	BRGC00 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 Be sure not to select n = 1 during an operation at f_X = 5.0 MHz because n = 1 exceeds the baud rate limit.

(b) Generation of baud rate transmit/receive clock by means of external clock from ASCK pin

The transmit/receive clock is generated by scaling the clock input from the ASCK pin. The baud rate generated from the clock input from the ASCK pin is found from the following expression.

$$[\text{Baud rate}] = \frac{f_{ASCK}}{16} \text{ [Hz]}$$

f_{ASCK}: Frequency of clock input to the ASCK pin

Table 9-4. Relationship between ASCK Pin Input Frequency and Baud Rate (When BRGC00 is Set to 80H)

Baud Rate (bps)	ASCK 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

9.4 Serial Interface 00 Operation

Serial interface 00 provides the following three types of modes.

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

9.4.1 Operation stop mode

In the operation stop mode, serial transfer is not executed, therefore, the power consumption can be reduced. The P20/$\overline{\text{SCK0}}$/ASCK, P21/SO0/TxD, and P22/SI0/RxD pins can be used as normal I/O ports.

(1) Register setting

Operation stop mode is set by serial operation mode register 00 (CSIM00) and asynchronous serial interface mode register 00 (ASIM00).

(a) Serial operation mode register 00 (CSIM00)

CSIM00 is set with a 1-bit or 8-bit memory manipulation instruction.
$\overline{\text{RESET}}$ input clears CSIM00 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM00	CSIE00	0	0	0	0	DIR00	CSCK00	0	FF72H	00H	R/W

CSIE00	Operation Control in 3-Wire Serial I/O Mode
0	Operation stop
1	Operation enable

Caution Be sure to set bit 0 and bits 3 to 6 to 0.

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

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

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

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

TXE00	Transmit Operation Control
0	Transmit operation stop
1	Transmit operation enable

RXE00	Receive Operation Control
0	Receive operation stop
1	Receive operation enable

Caution Be sure to set bits 0 and 1 to 0.

9.4.2 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 a UART-dedicated baud rate generator that enables communications at a desired transfer rate from many options. In addition, the baud rate can also be defined by dividing the input clock to the ASCK 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

The UART mode is set by serial operation mode register 00 (CSIM00), asynchronous serial interface mode register 00 (ASIM00), asynchronous serial interface status register 00 (ASIS00), and baud rate generator control register 00 (BRGC00).

(a) Serial operation mode register 00 (CSIM00)

CSIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears CSIM00 to 00H.

Set 00H to CSIM00 when UART mode is selected.

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM00	CSIE00	0	0	0	0	DIR00	CSCK00	0	FF72H	00H	R/W

CSIE00	Operation Control in 3-Wire Serial I/O Mode
0	Operation stop
1	Operation enable

DIR00	Start Bit Specification
0	MSB
1	LSB

CSCK00	Clock Selection in 3-Wire Serial I/O Mode
0	Input clock to $\overline{\text{SCK0}}$ pin from external
1	Dedicated baud rate generator output

Caution Be sure to set bit 0 and bits 3 to 6 to 0.

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

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

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

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

TXE00	Transmit Operation Control
0	Transmit operation stop
1	Transmit operation enable

RXE00	Receive Operation Control
0	Receive operation stop
1	Receive operation enable

PS001	PS000	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

CL00	Character Length Specification
0	7 bits
1	8 bits

SL00	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

Cautions 1. Be sure to set bits 0 and 1 to 0.

2. Switching operating modes must be performed after the halt of serial transmit/receive operation.

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

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

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

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

PE00	Parity Error Flag
0	Parity error not generated
1	Parity error generated (when the parity of transmit data does not coincide.)

FE00	Flaming Error Flag
0	Framing error not generated
1	Framing error generated (when stop bit is not detected.)[Note 1]

OVE00	Overrun Error Flag
0	Overrun error not generated
1	Overrun error generated[Note 2] (when the next receive operation is completed before the data is read from the receive buffer register.)

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

2. Be sure to read reception buffer register 00 (RXB00) when an overrun error occurs. If not, every time the data is received an overrun error is generated.

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

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

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

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
BRGC00	TPS003	TPS002	TPS001	TPS000	0	0	0	0	FF73H	00H	R/W

TPS003	TPS002	TPS001	TPS000	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	Input clock from external to ASCK pin	-
Other than above				Setting prohibited	

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 operation.

2. Be sure not to select n = 1 during an operation at f_X = 5.0 MHz because n = 1 exceeds the baud rate limit.

3. When selecting an input clock from an external source, set port mode register 2 (PM2) to the input mode.

Remarks 1. f_X : System clock oscillation frequency

2. n : Value determined by setting TPS000 through TPS003 ($1 \leq n \leq 8$)

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 from the ASCK pin.

(i) Generation of baud rate transmit/receive clock by means of system clock

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

$$[\text{Baud rate}] = \frac{f_X}{2^{n+1} \times 8} \text{ [Hz]}$$

f_X : System clock oscillation frequency

n : Value determined by setting TPS000 through TPS003 as shown in the above table ($2 \leq n \leq 8$)

Table 9-5. Example of Relationship between System Clock and Baud Rate

Baud Rate (bps)	n	BRGC00 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 Be sure not to select n = 1 during an operation at f_X = 5.0 MHz because n = 1 exceeds the baud rate limit.

(ii) Generation of baud rate transmit/receive clock by means of external clock from ASCK pin

The transmit/receive clock is generated by scaling the clock input from the ASCK pin. The baud rate generated from the clock input from the ASCK 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 ASCK pin

Table 9-6. Relationship between ASCK Pin Input Frequency and Baud Rate (When BRGC00 is Set to 80H)

Baud Rate (bps)	ASCK 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

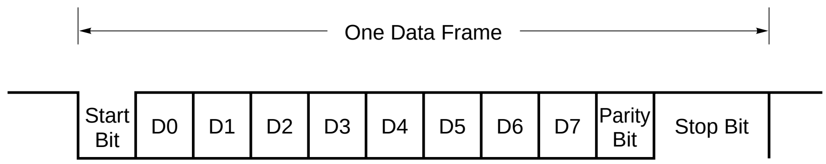
(2) Communication operation

(a) Data format

The transmit/receive data format is as shown in Figure 9-7. One data frame consists of a start bit, character bits, parity bit, and stop bit(s).

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

Figure 9-7. Asynchronous Serial Interface Transmit/Receive Data Format



- Start bit...................... 1 bit
- Character bits............ 7 bits/8 bits
- Parity bit .... 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 low-order 7 bits (bits 0 to 6) are valid; in transmission the most significant bit (bit 7) is ignored, and in reception the most significant bit (bit 7) is always "0".

The serial transfer rate is selected by means of ASIM00 and baud rate generator control register 00 (BRGC00).

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

(b) Parity types and operation

The parity bit is used to detect a bit error in the communication data. Normally, the same 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 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.

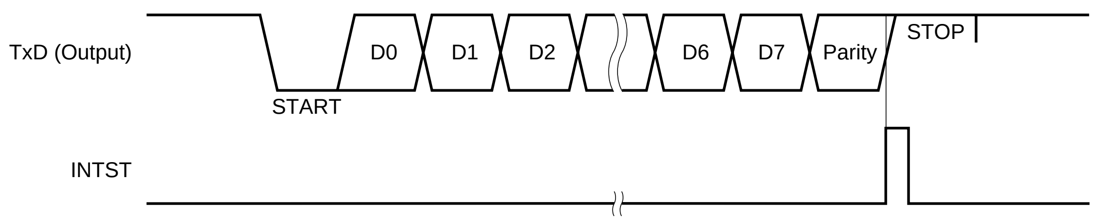
(c) Transmission

A transmit operation is started by writing transmit data to transmit shift register 00 (TXS00). The start bit, parity bit, and stop bit(s) are added automatically.

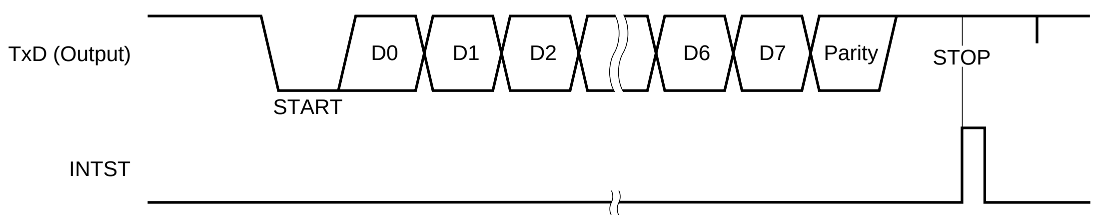
When the transmit operation starts, the data in TXS00 is shifted out, and when TXS00 is empty, a transmission completion interrupt (INTST) is generated.

Figure 9-8. 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 00 (ASIM00) during a transmit operation. If the ASIM00 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 (INTST) or the interrupt request flag (STIF00) set by INTST.

(d) Reception

When bit 6 (RXE00) of asynchronous serial interface mode register 00 (ASIM00) is set to 1, a receive operation is enabled and sampling of the RxD pin input is performed.

RxD pin input sampling is performed using the serial clock specified by ASIM00.

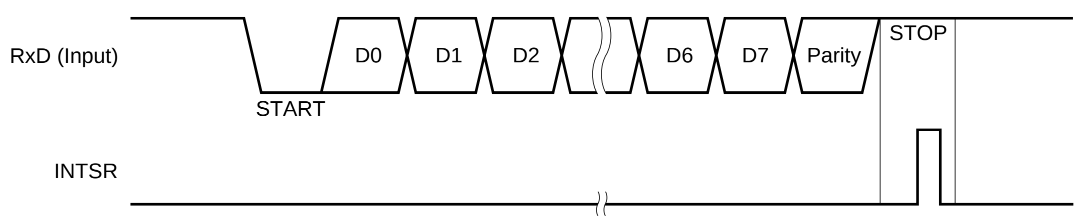
When the RxD pin input becomes low, the 3-bit counter starts counting, and at the time when half the time determined by the specified baud rate has passed, the data sampling start timing signal is output. If the RxD pin input sampled again as a result of this start timing signal is low, it is identified 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 receive buffer register 00 (RXB00), and a reception completion interrupt (INTSR) is generated.

If an error is generated, the receive data in which the error was generated is still transferred to RXB00, and INTSR is generated.

If the RXE00 bit is reset to 0 during the receive operation, the receive operation is stopped immediately. In this case, the contents of RXB00 and asynchronous serial interface status register 00 (ASIS00) are not changed, and INTSR is not generated.

Figure 9-9. Asynchronous Serial Interface Reception Completion Interrupt Timing



Caution Be sure to read receive buffer register 00 (RXB00) even if a receive error occurs. If RXB00 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, or overrun error. Upon data reception, an error flag is set in asynchronous serial interface status register 00 (ASIS00). Receive error causes are shown in Table 9-7.

It is possible to determine what kind of error was generated during reception by reading the contents of ASIS00 in the reception error interrupt servicing (see **Figures 9-9** and **9-10**).

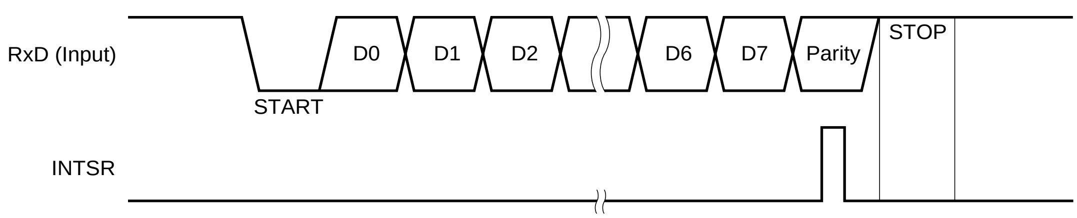
The contents of ASIS00 are reset to 0 by reading receive buffer register 00 (RXB00) or receiving the next data (if there is an error in the next data, the corresponding error flag is set).

Table 9-7. Receive Error Causes

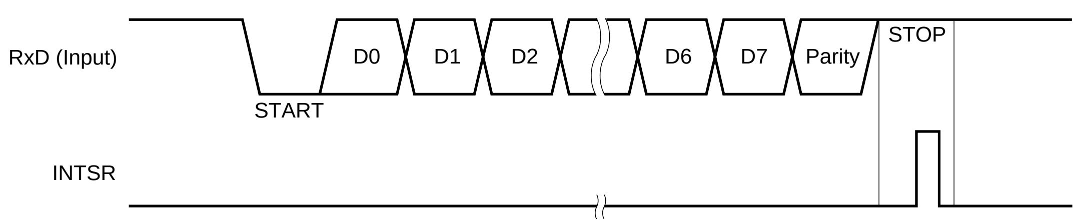
Receive Error	Cause
Parity error	Transmission-time parity specification and reception data parity do not match
Framing error	Stop bit not detected
Overrun error	Reception of next data is completed before data is read from receive buffer register

Figure 9-10. Receive Error Timing

(a) Parity error generated



(b) Framing error or overrun error generated



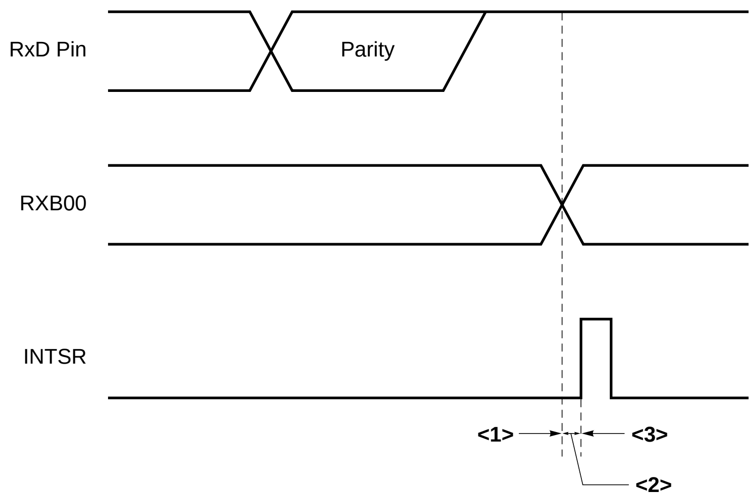
Cautions 1. The contents of the ASIS00 register are reset to 0 by reading receive buffer register 00 (RXB00) or receiving the next data. To ascertain the error contents, read ASIS00 before reading RXB00.

2. Be sure to read receive buffer register 00 (RXB00) even if a receive error is generated. If RXB00 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 (TXE00) of asynchronous serial interface mode register 00 (ASIM00) is cleared during transmission, be sure to set transmit shift register 00 (TXS00) to FFH, then set TXE00 to 1 before executing the next transmission.

(b) When bit 6 (RXE00) of asynchronous serial interface mode register 00 (ASIM00) is cleared during reception, receive buffer register 00 (RXB00) and receive completion interrupt (INTSR) are as follows.



When RXE00 is set to 0 at a time indicated by **<1>**, RXB00 holds the previous data and INTSR is not generated.
When RXE00 is set to 0 at a time indicated by **<2>**, RXB00 renews the data and INTSR is not generated.
When RXE00 is set to 0 at a time indicated by **<3>**, RXB00 renews the data and INTSR is generated.

9.4.3 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 75X/XL Series, 78K Series, 17K Series, etc.

Communication is performed using three lines: the serial clock ($\overline{\text{SCK0}}$), serial output (SO0), and serial input (SI0).

(1) Register setting

3-wire serial I/O mode settings are performed using serial operation mode register 00 (CSIM00), asynchronous serial interface mode register 00 (ASIM00), and baud rate generator control register 00 (BRGC00).

(a) Serial operation mode register 00 (CSIM00)

CSIM00 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears CSIM00 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After Reset	R/W
CSIM00	CSIE00	0	0	0	0	DIR00	CSCK00	0	FF72H	00H	R/W

CSIE00	Operation Control in 3-Wire Serial I/O Mode
0	Operation stop
1	Operation enable

DIR00	Start Bit Specification
0	MSB
1	LSB

CSCK00	Clock Selection in 3-Wire Serial I/O Mode
0	Input clock to $\overline{\text{SCK0}}$ pin from external
1	Dedicated baud rate generator output

Caution Be sure to set bit 0 and bits 3 to 6 to 0.

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

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

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

When the 3-wire serial I/O mode is selected, 00H must be set to ASIM00.

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

TXE00	Transmit Operation Control
0	Transmit operation stop
1	Transmit operation enable

RXE00	Receive Operation Control
0	Receive operation stop
1	Receive operation enable

PS001	PS000	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

CL00	Character Length Specification
0	7 bits
1	8 bits

SL00	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

Cautions 1. Be sure to set bits 0 and 1 to 0.

2. Switching operating modes must be performed after serial transmit/receive operation is halted.

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

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

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

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
BRGC00	TPS003	TPS002	TPS001	TPS000	0	0	0	0	FF73H	00H	R/W

TPS003	TPS002	TPS001	TPS000	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 BRGC00 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 BRGC00 during communication operation.

2. Be sure not to select n = 1 during an operation at f_X = 5.0 MHz because n = 1 exceeds the baud rate limit.

3. When selecting an input clock from an external source, set port mode register 2 (PM2) to the input mode.

Remarks 1. f_X : System clock oscillation frequency

2. n : Value determined by setting TPS000 through TPS003 ($1 \leq n \leq 8$)

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 the TPS000 to TPS003 bits to set the frequency of the serial clock. To obtain the frequency to be set, use the following formula. When the serial clock is input from external, setting BRGC00 is not necessary.

$$\text{Serial clock frequency} = \frac{f_X}{2^{n+1}} \text{ [Hz]}$$

f_X : System clock oscillation frequency

n : Value determined by setting TPS000 through TPS003 as shown in the above table ($1 \leq n \leq 8$)

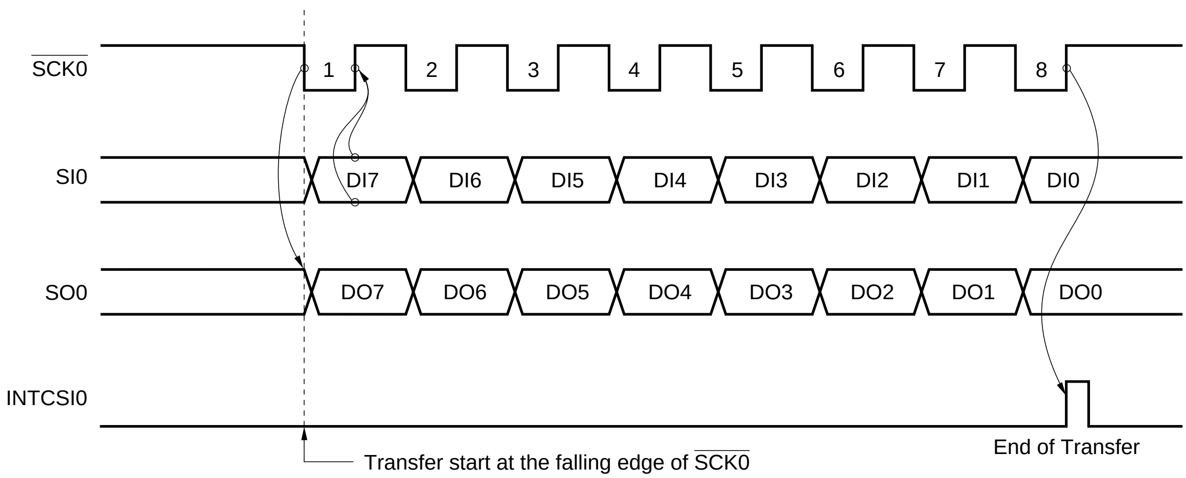
(2) Communication operation

In the 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.

Transmit shift register 00 (TXS00/SIO00) and receive shift register 00 (RXS00) shift operations are performed in synchronization with the fall of the serial clock ($\overline{\text{SCK0}}$). Then transmit data is held in the SO0 latch and output from the SO0 pin. Also, receive data input to the SI0 pin is latched in receive buffer register 00 (RXB00/SIO00) on the rise of $\overline{\text{SCK0}}$.

At the end of an 8-bit transfer, the operations of TXS00/SIO00 and RXS00 stop automatically, and the interrupt request signal (INTCSI0) is generated.

Figure 9-11. 3-Wire Serial I/O Mode Timing



(3) Transfer start

Serial transfer is started by setting transfer data to transmit shift register 00 (TXS00/SIO00) when the following two conditions are satisfied.

- Serial operation mode register 00 (CSIM00) bit 7 (CSIE00) = 1
- Internal serial clock is stopped or $\overline{\text{SCK0}}$ is at high level after 8-bit serial transfer.

Caution If CSIE00 is set to "1" after data write to TXS00/SIO00, transfer does not start.

A termination of 8-bit transfer stops the serial transfer automatically and generates the interrupt request signal (INTCSI0).

CHAPTER 10 INTERRUPT FUNCTIONS

10.1 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 10-1.
A standby release signal is generated.
The maskable interrupt has four sources of external interrupts and five sources of internal interrupts.

10.2 Interrupt Sources and Configuration

There are total of ten non-maskable and maskable interrupts in the interrupt sources (see **Table 10-1**).

Table 10-1. Interrupt Source List

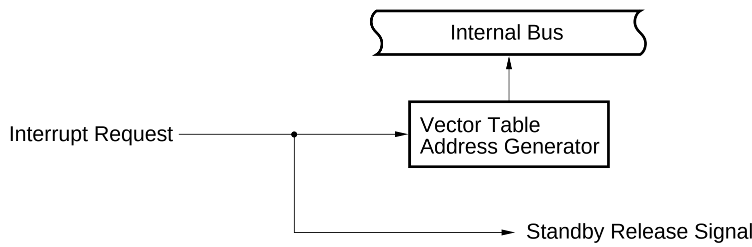
Interrupt Type	Priority[Note 1]	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	–	INTWDT	Watchdog timer overflow (watchdog timer mode 1 selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (interval timer mode selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTSR	End of serial interface 00 UART reception	Internal	000CH	(B)
		INTCSI0	End of serial interface 00 3-wire transfer			
	5	INTST	End of serial interface 00 UART transmission		000EH	
	6	INTTM0	Generation of 8-bit timer/event counter 00 match signal		0010H	
	7	INTTM2	Generation of 16-bit timer 20 match signal		0014H	
	8	INTKR	Key return signal detection	External	002AH	(C)

Notes 1. Priorities are intended for the priority for two or more simultaneously generated maskable interrupts. 0 is the highest priority and 8 is the lowest priority.

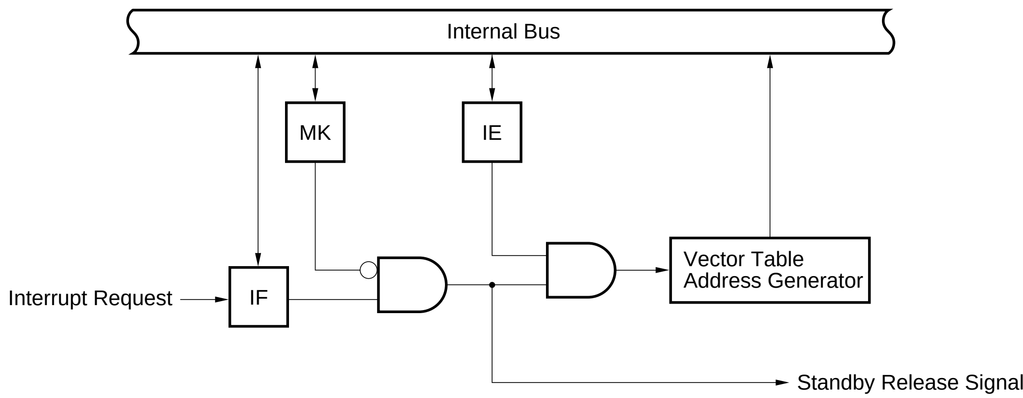
2. Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 10-1.

Figure 10-1. Basic Configuration of Interrupt Function

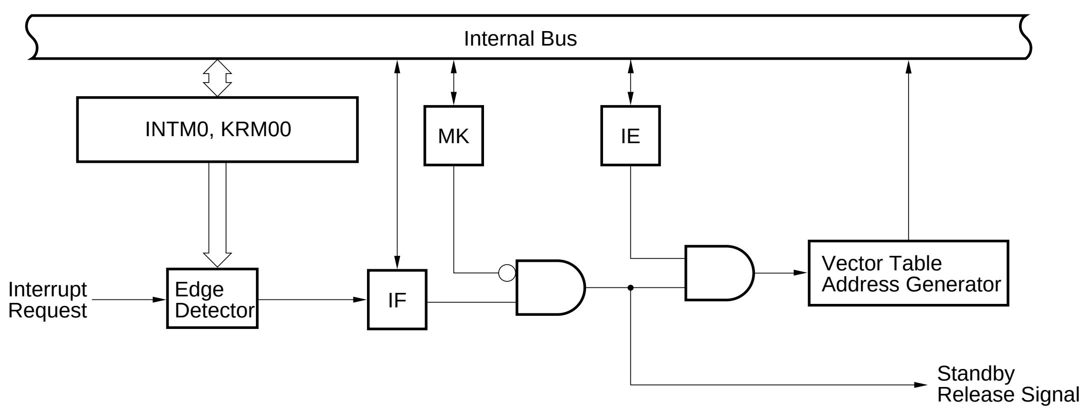
(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt



INTM0 : External interrupt mode register 0
KRM00 : Key return mode register 00
IF : Interrupt request flag
IE : Interrupt enable flag
MK : Interrupt mask flag

10.3 Interrupt Function Control Registers

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

- Interrupt request flag registers (IF0 and IF1)
- Interrupt mask flag registers (MK0 and MK1)
- External interrupt mode register 0 (INTM0)
- Program status word (PSW)
- Key return mode register 00 (KRM00)

Table 10-2 gives a listing of interrupt request flag and interrupt mask flag names corresponding to interrupt requests.

Table 10-2. Flags Corresponding to Interrupt Request Signal Name

Interrupt Request Signal Name	Interrupt Request Flag	Interrupt Mask Flag
INTWDT	TMIF4	TMMK4
INTP0	PIF0	PMK0
INTP1	PIF1	PMK1
INTP2	PIF2	PMK2
INTSR/INTCSI0	SRIF00	SRMK00
INTST	STIF00	STMK00
INTTM0	TMIF00	TMMK00
INTTM2	TMIF20	TMMK20
INTKR	KRIF00	KRMK00

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

The interrupt request flag is set to 1 when the corresponding interrupt request is generated or an instruction is executed. It is cleared to 0 upon acknowledgement of an interrupt request, upon $\overline{\text{RESET}}$ input, or when an 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 10-2. Interrupt Request Flag Register Format

Symbol	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After Reset	R/W
IF0	0	TMIF00	STIF00	SRIF00	PIF2	PIF1	PIF0	TMIF4	FFE0H	00H	R/W

	<7>	6	5	4	3	2	1	<0>			
IF1	TMIF20	0	0	0	0	0	0	KRIF00	FFE1H	00H	R/W

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

Cautions 1. Be sure to clear bit 7 of IF0 and bits 1 to 6 of IF1 to 0.

2. TMIF4 flag is R/W enabled only when a watchdog timer is used as an interval timer. If the watchdog timer mode 1 or 2 is used, set TMIF4 flag to 0.

3. Because port 3 has an alternate function as the external interrupt input, when the output level is changed by specifying the output mode of the port function, an interrupt request flag is set. Therefore, 1 should be set in the interrupt mask flag before using the output mode.

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

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

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 10-3. Interrupt Mask Flag Register Format

Symbol	7	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After Reset	R/W
MK0	1	TMMK00	STMK00	SRMK00	PMK2	PMK1	PMK0	TMMK4	FFE4H	FFH	R/W

Symbol	<7>	6	5	4	3	2	1	<0>	Address	After Reset	R/W
MK1	TMMK20	1	1	1	1	1	1	KRMK00	FFE5H	FFH	R/W

××MK×	Interrupt Servicing Control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

Cautions 1. Be sure to set bit 7 of MK0 and bits 1 to 6 of MK1 to 1.

2. IF the TMMK4 flag is read when a watchdog timer is used in watchdog timer mode 1 or 2, its value becomes undefined.

3. Because port 3 has an alternate function as the external interrupt input, when the output level is changed by specifying the output mode of the port function, an interrupt request flag is set. Therefore, 1 should be set in the interrupt mask flag before using the output mode.

(3) External interrupt mode register 0 (INTM0)

This register is used to set the valid edge of INTP0 to INTP2.

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

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

Figure 10-4. External Interrupt Mode Register 0 Format

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. Be sure to set bits 0 and 1 to 0.

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

(4) Program status word (PSW)

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

Besides 8-bit unit read/write, this register can carry out operations with a bit manipulation instruction and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged, PSW is automatically saved into a stack, and the IE flag is reset to 0. It is restored from the stack with the RETI and POP PSW instructions.

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

Figure 10-5. Program Status Word Configuration

Symbol	7	6	5	4	3	2	1	0	After Reset
PSW	IE	Z	0	AC	0	0	1	CY	02H

Z, AC, CY → Used when normal instruction is executed

IE	Interrupt Acknowledge Enable/Disable
0	Disable
1	Enable

(5) Key return mode register 00 (KRM00)

KRM00 is used to specify the pin at which a key return signal is detected.
KRM00 is set with a 1-bit or 8-bit memory manipulation instruction.
Bit 0 (KRM000) is set for the four pins from KR0/P40 to KR3/P43. Bits 4 to 7 (KRM004 to KRM007) are set in 1-bit units for pins KR4/P44 to KR7/P47, respectively.
$\overline{\text{RESET}}$ input clears KRM00 to 00H.

Figure 10-6. Key Return Mode Register 00 Format

Symbol	7	6	5	4	3	2	1	0	Address	After Reset	R/W
KRM00	KRM007	KRM006	KRM005	KRM004	0	0	0	KRM000	FFF5H	00H	R/W

KRM00n	Key Return Signal Detection Selection
0	Undetected
1	Detected (at the falling edge of port 4)

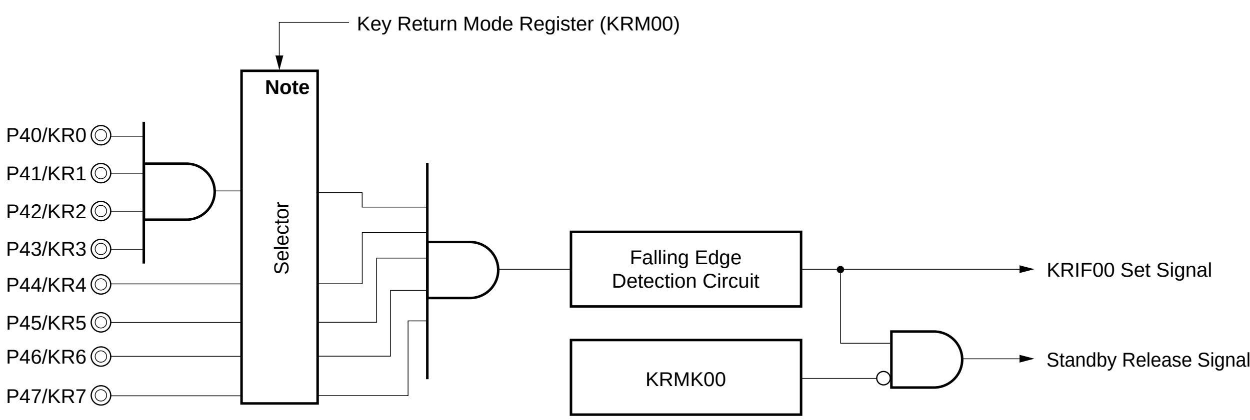
Cautions 1. Be sure to set bits 1 to 3 to 0.

2. When KRM00 is set to 1, the corresponding pin is connected to a pull-up resistor unless it is in output mode. In output mode, the pull-up resistor is not connected.

**3. Before setting KRM00, set bit 0 of MK1 (KRMK00 = 1) to disable interrupts.
To enable interrupts, clear bit 0 of IF1 (KRIF00 = 0), then bit 0 of MK1 (KRMK00 = 0).**

Remark n = 0, 4 to 7

Figure 10-7. Falling Edge Detection Circuit



Note Selector used to select the pin to be used for falling edge input

10.4 Interrupt Processing Operation

10.4.1 Non-maskable interrupt request acceptance operation

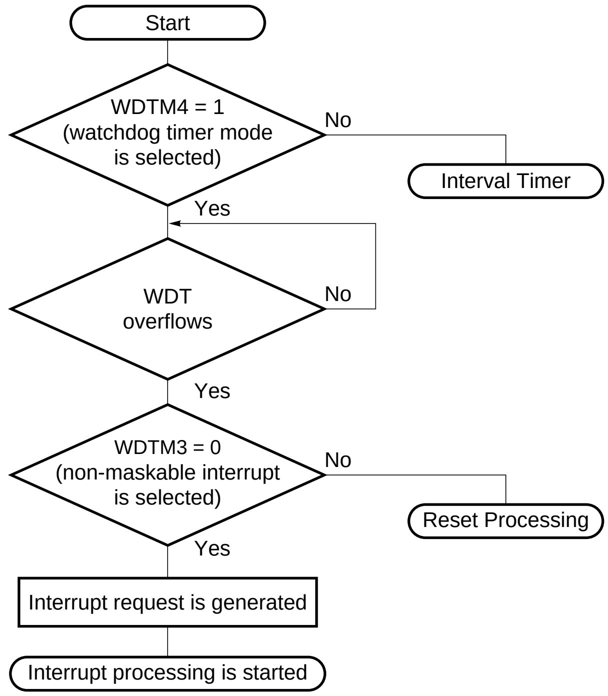
The non-maskable interrupt request is unconditionally accepted even when interrupts are disabled. It is not subject to interrupt priority control and takes precedence over all other interrupts.

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 10-8 shows the flowchart from non-maskable interrupt request generation to acceptance. Figure 10-9 shows the timing of non-maskable interrupt request acceptance. Figure 10-10 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 10-8. Flowchart from Non-Maskable Interrupt Request Generation to Acceptance**



WDTM : Watchdog timer mode register
WDT : Watchdog timer

Figure 10-9. Non-Maskable Interrupt Request Acceptance Timing

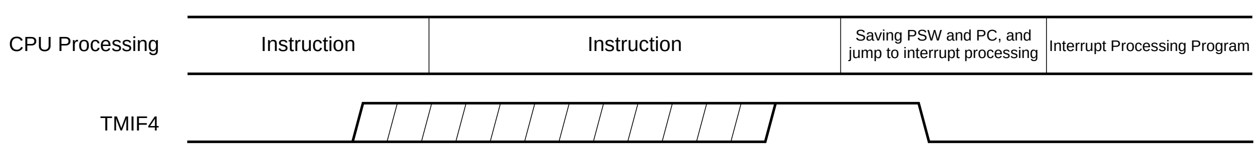
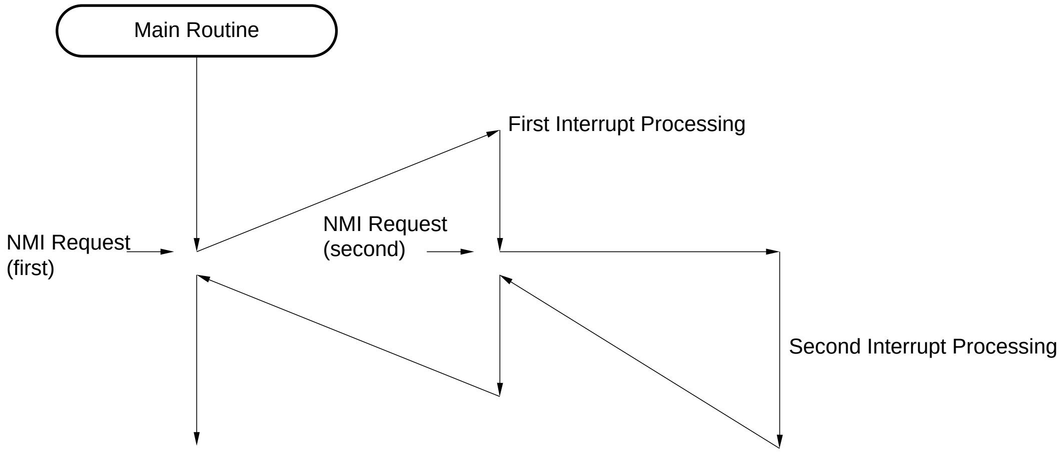


Figure 10-10. Accepting Non-Maskable Interrupt Request



10.4.2 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 10-3.

See Figures 10-12 and 10-13 for the interrupt request acceptance timing.

Table 10-3. 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.

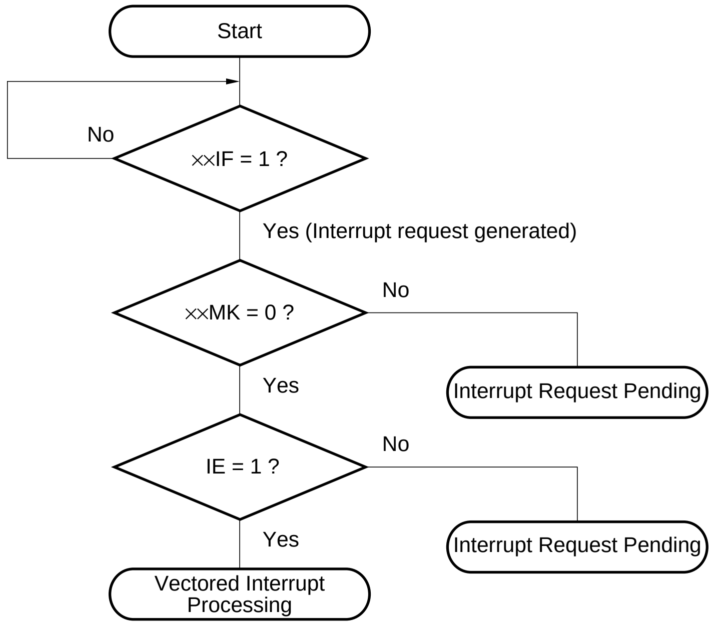
A pended interrupt is accepted when the status where it can be accepted is set.

Figure 10-11 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 10-11. Interrupt Request Acceptance Program Algorithm

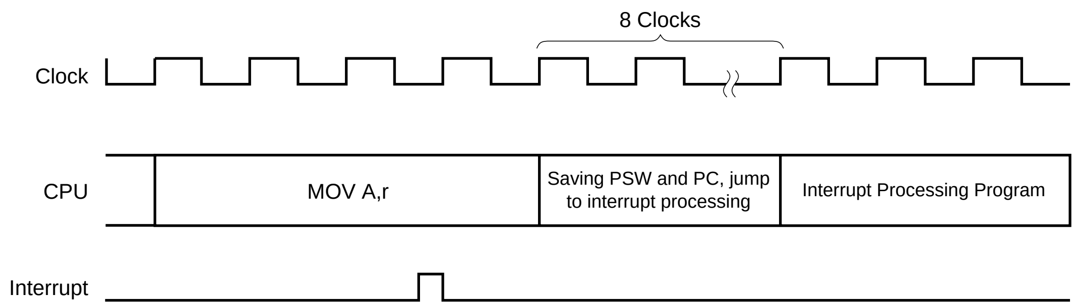


××IF : Interrupt request flag

××MK : Interrupt mask flag

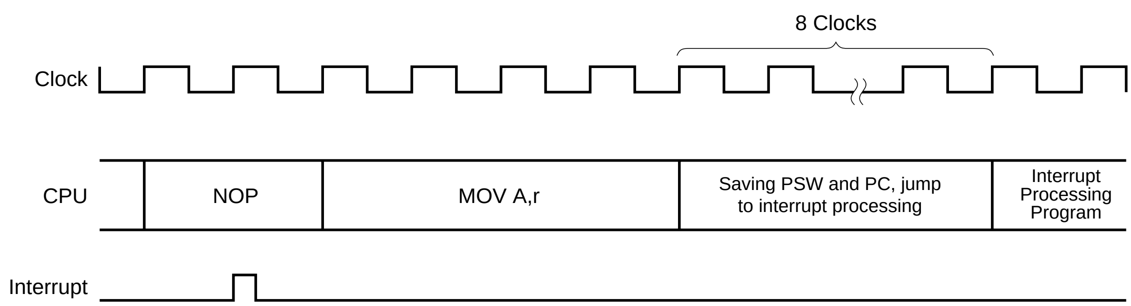
IE : Flag to control maskable interrupt request acceptance (1 = enable, 0 = disable)

Figure 10-12. 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 10-12 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 10-13. 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 10-13 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.

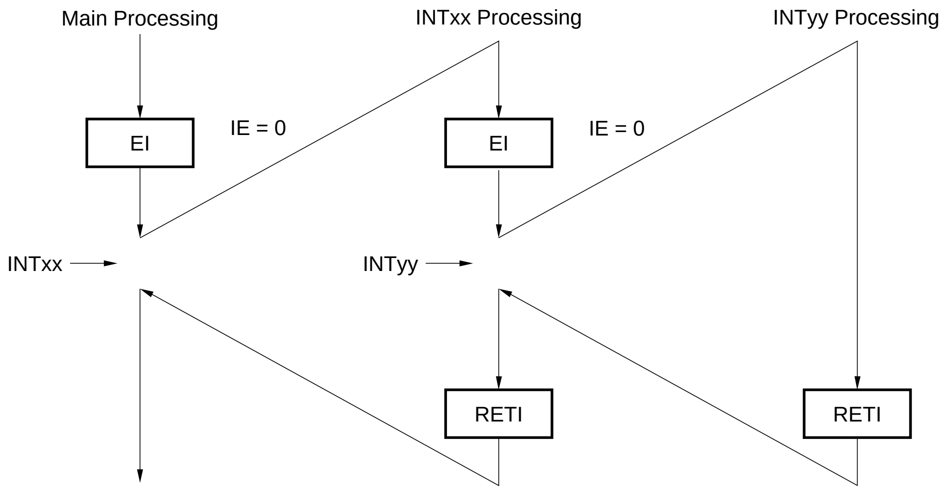
Caution Interrupt requests are reserved while the interrupt request flag register (IF0 or IF1) or the interrupt mask flag register (MK0 or MK1) is being accessed.

10.4.3 Nesting processing

Nesting 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 10-1**).

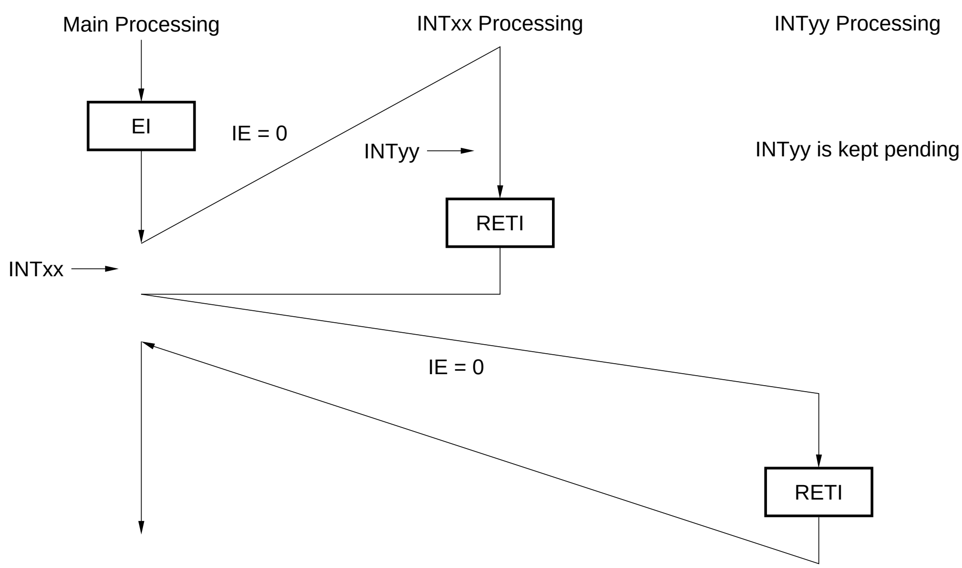
Figure 10-14. Example of Nesting

Example 1. Nesting is accepted



During interrupt INTxx servicing, interrupt request INTyy is accepted, and a nesting is generated. An EI instruction is issued before each interrupt request acceptance, and the interrupt request acceptance enable state is set.

Example 2. A nesting 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 nesting is not generated. The INTyy request is reserved and accepted after the INTxx processing is performed.

IE = 0: Interrupt request acceptance disabled

10.4.4 Interrupt request reserve

Some instructions may reserve the acceptance of an interrupt 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)

[MEMO]

CHAPTER 11 STANDBY FUNCTION

11.1 Standby Function and Configuration

11.1.1 Standby function

The standby function is to reduce the power consumption of the system and can be effected in the following two modes:

(1) HALT mode

This mode is set when the HALT instruction is executed. The HALT mode stops the operation clock of the CPU. The system clock oscillation circuit continues oscillating. This mode does not reduce the current drain as much as the STOP mode, but is useful for resuming processing immediately when an interrupt request is generated, or for intermittent operations.

(2) STOP mode

This mode is set when the STOP instruction is executed. The STOP mode stops the system clock oscillation circuit and stops the entire system. The current drain of the CPU can be substantially reduced in this mode.

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

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

In both modes, the previous contents of the registers, flags, and data memory before setting standby mode are all retained. In addition, the statuses of the output latch of the I/O ports and output buffer are also retained.

Caution To set the STOP mode, be sure to stop the operations of the peripheral hardware, and then execute the STOP instruction.

11.1.2 Standby function control register

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

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

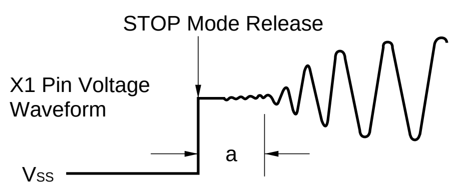
$\overline{\text{RESET}}$ input sets OSTS to 04H. However, the oscillation settling time after $\overline{\text{RESET}}$ input is $2^{15}/f_X$, instead of $2^{17}/f_X$.

Figure 11-1. Oscillation Settling Time Select Register Format

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

OSTS2	OSTS1	OSTS0	Oscillation Settling Time Selection
0	0	0	$2^{12}/f_X$ (819 μs)
0	1	0	$2^{15}/f_X$ (6.55 ms)
1	0	0	$2^{17}/f_X$ (26.2 ms)
Other than above			Setting prohibited

Caution The wait time after the STOP mode is released does not include the time from STOP mode release to clock oscillation start ("a" in the figure below), regardless of 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.

11.2 Operation of Standby Function

11.2.1 HALT mode

(1) HALT mode

HALT mode is set by executing the HALT instruction.

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

Table 11-1. HALT Mode Operating Status

Item	HALT Mode Operating Status
Clock generation circuit	System clock oscillation enabled Clock supply to CPU stopped
CPU	Operation stopped
Port (Output latch)	Retains the status before setting the HALT mode
16-bit timer	Operation enabled
8-bit timer/event counter	Operation enabled
Watchdog timer	Operation enabled
Serial interface	Operation enabled
External interrupt	Operation enabled
Key return	Only the pin set to key return mode is enabled

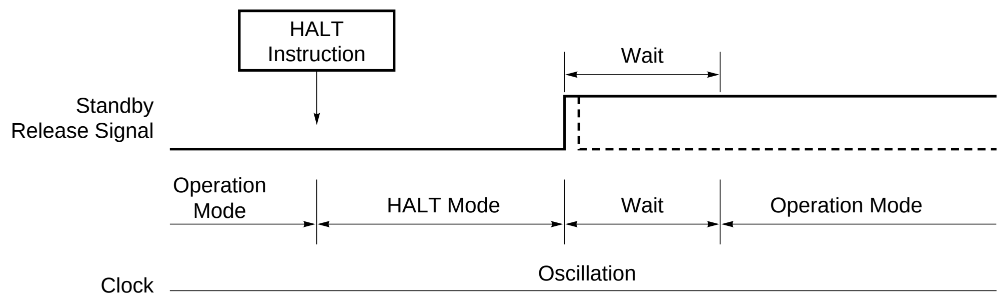
(2) Releasing HALT mode

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

(a) Releasing by unmasked interrupt request

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

Figure 11-2. Releasing HALT Mode by Interrupt



Remarks 1. The broken line indicates the case where the interrupt request that has released the standby mode is 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

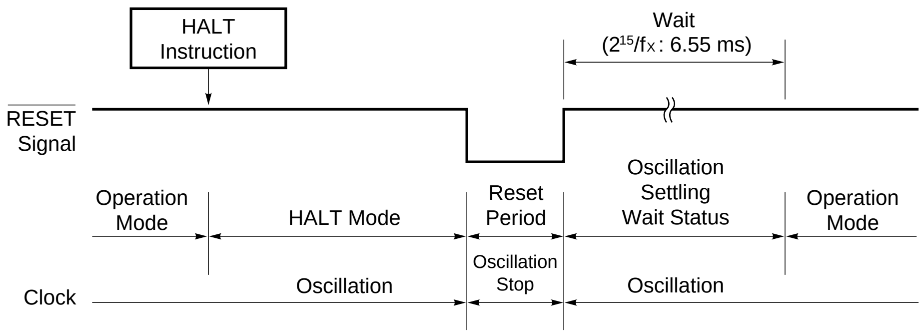
(b) Releasing by non-maskable interrupt request

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

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

When the HALT mode is released by the $\overline{\text{RESET}}$ signal, execution branches to the reset vector address in the same manner as the ordinary reset operation, and program execution is started.

Figure 11-3. Releasing HALT Mode by $\overline{\text{RESET}}$ Input



Remarks 1. f_X: System clock oscillation frequency

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

Table 11-2. 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

11.2.2 STOP mode

(1) Setting and operation status of STOP mode

The STOP mode is set by executing the STOP instruction.

Caution Because the standby mode can be released by an interrupt request signal, the standby mode is released as soon as it is set if there is an interrupt source whose interrupt request flag is set and interrupt mask flag is reset. When the STOP mode is set, therefore, the HALT mode is set immediately after the STOP instruction has been executed, the wait time set by the oscillation settling time select register (OSTS) elapses, and then operation mode is set.

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

Table 11-3. STOP Mode Operating Status

Item	STOP Mode Operation Status
Clock generation circuit	Stops system clock oscillation
CPU	Stops operation
Port (Output latch)	Retains the status before setting the STOP mode
16-bit timer	Stops operation
8-bit timer/event counter	Operation is enabled only when TI0 is selected as the count clock
Watchdog timer	Stops operation
Serial interface	Enables operation only when input clock from external is selected as serial clock
External interrupt	Operation enabled
Key return	Only the pin set to key return mode is enabled

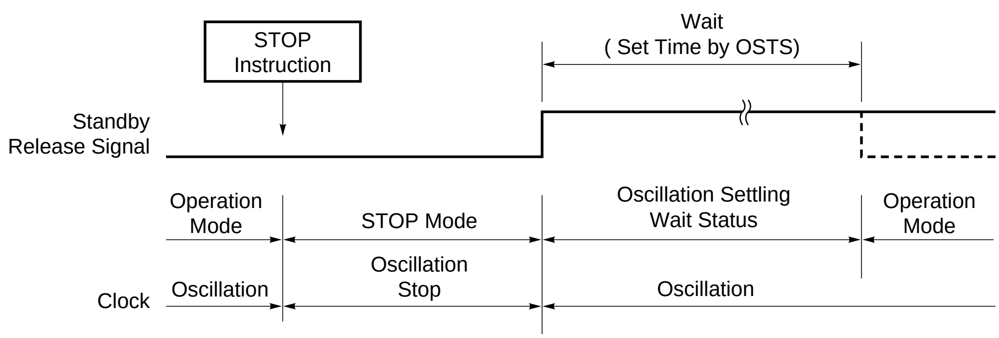
(2) Releasing STOP mode

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

(a) Releasing by unmasked interrupt request

The STOP mode can be released by an unmasked interrupt request. In this case, if the interrupt is enabled to be 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 11-4. Releasing STOP Mode by Interrupt

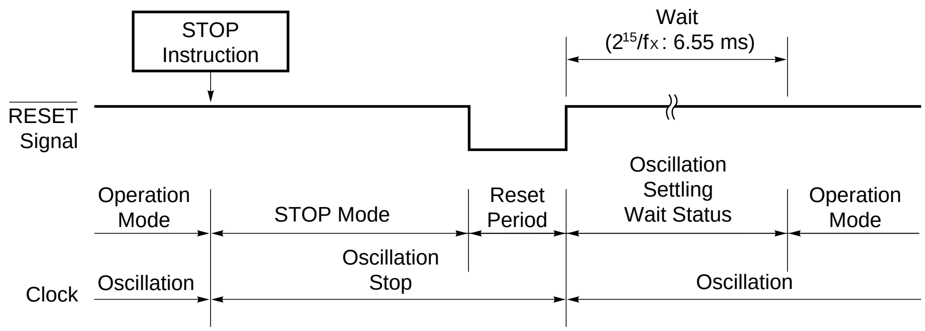


Remark The broken line indicates the case where the interrupt request that has released the standby mode is accepted.

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

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

Figure 11-5. Releasing STOP Mode by $\overline{\text{RESET}}$ Input



Remarks 1. f_X: System clock oscillation frequency

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

Table 11-4. 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 12 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 12-1. 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 ($2^{15}/f_X$) 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 ($2^{15}/f_X$) has elapsed (see **Figures 12-2** through **12-4**).

Cautions 1. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin.

2. When the STOP mode is cleared by reset, the STOP mode contents are held during reset input. However, the port pins become high impedance.

Figure 12-1. Block Diagram of Reset Function

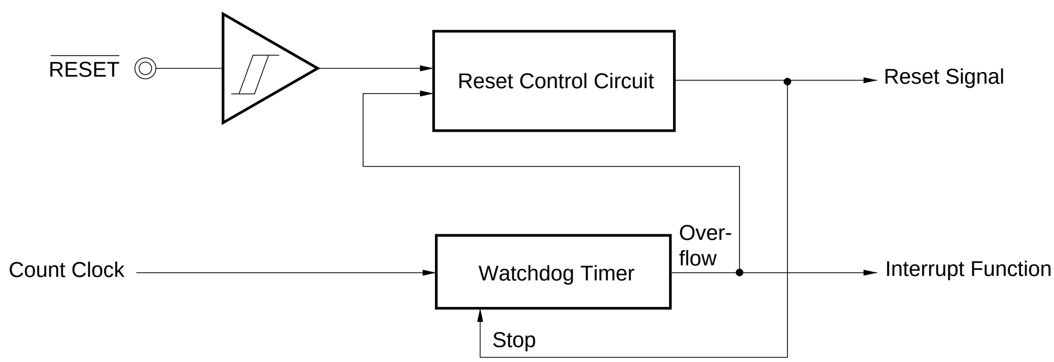


Figure 12-2. Reset Timing by $\overline{\text{RESET}}$ Input

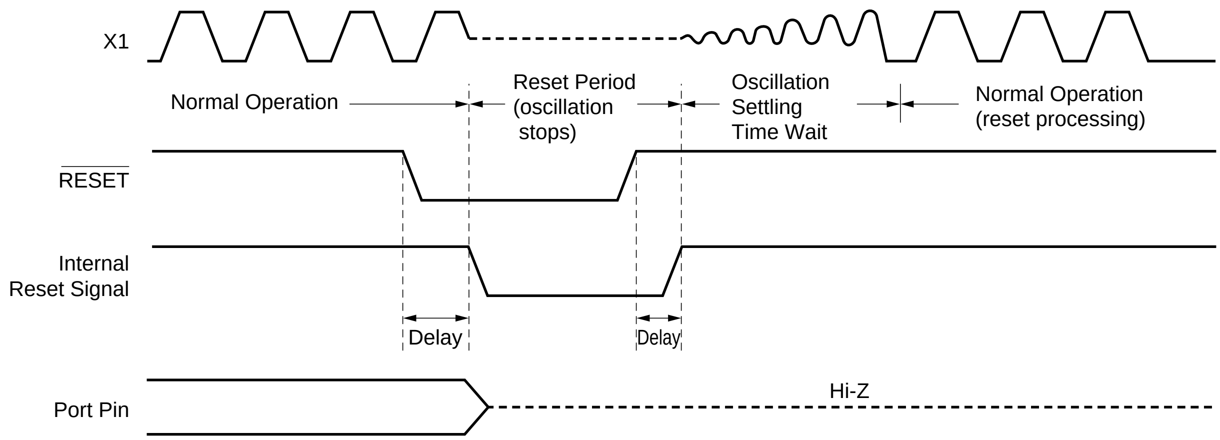


Figure 12-3. Reset Timing by Overflow in Watchdog Timer

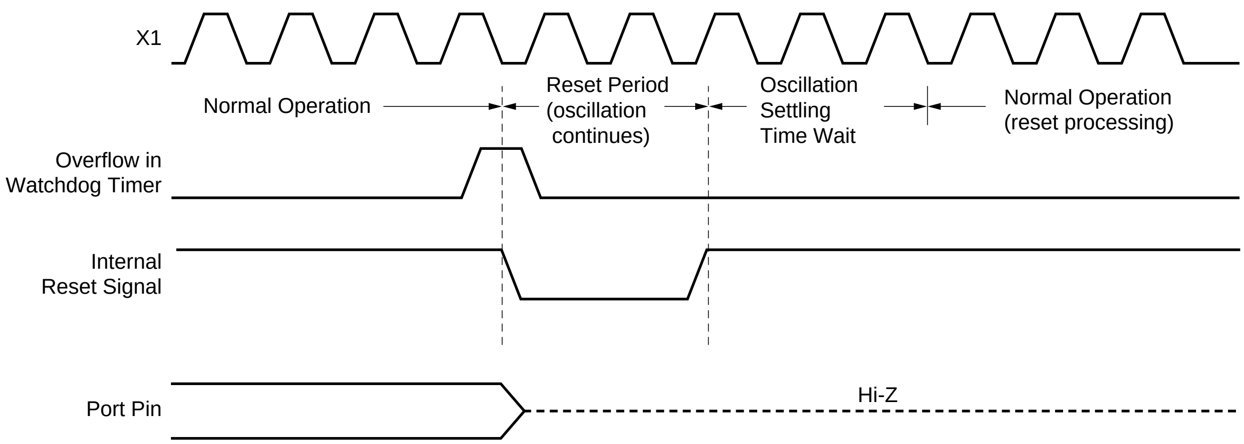


Figure 12-4. Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode

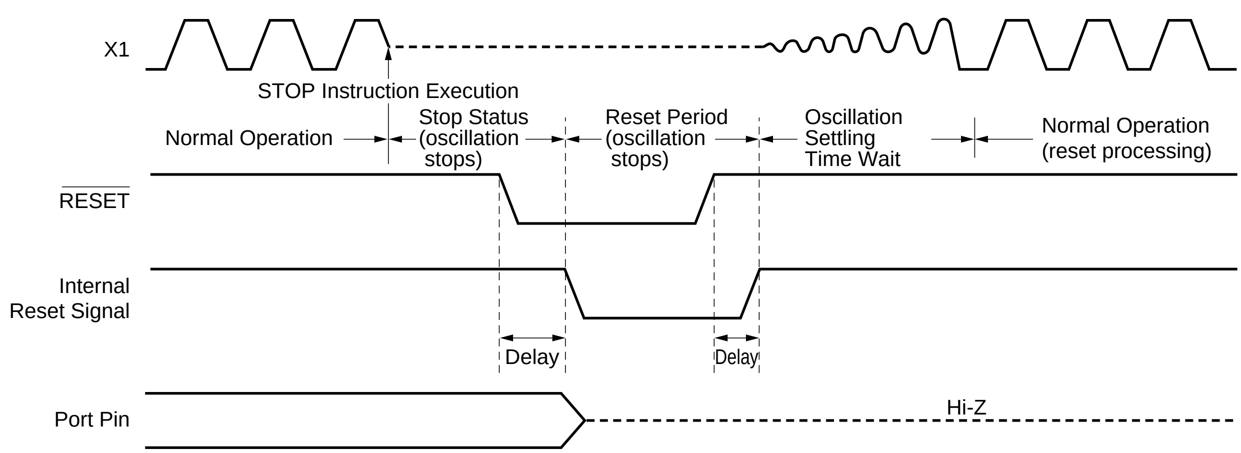


Table 12-1. Hardware Status after Reset

Hardware		State after Reset
Program counter (PC)[Note 1]		The contents of reset vector tables (0000H and 0001H) are set.
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined[Note 2]
	General-purpose register	Undefined[Note 2]
Ports (P0 to P5) (Output latch)		00H
Port mode registers (PM0 to PM5)		FFH
Pull-up resistor option register (PUO)		00H
Processor clock control register (PCC)		02H
Oscillation settling time select register (OSTS)		04H
16-bit timer	Timer counter (TM20)	0000H
	Compare register (CR20)	FFFFH
	Mode control register (TMC20)	00H
	Capture register (TCP20)	Undefined
8-bit timer/event counter	Timer counter (TM00)	00H
	Compare register (CR00)	00H
	Mode control register (TMC00)	00H
Watchdog timer	Timer clock select register (TCL2)	00H
	Mode register (WDTM)	00H
Serial interface	Mode register (CSIM00)	00H
	Asynchronous serial interface mode register (ASIM00)	00H
	Asynchronous serial interface status register (ASIS00)	00H
	Baud rate generator control register (BRGC00)	00H
	Transmit shift register (TXS00)	FFH
	Receive buffer register (RXB00)	Undefined
Interrupt	Request flag registers (IF0, IF1)	00H
	Mask flag registers (MK0, MK1)	FFH
	External interrupt mode register (INTM0)	00H
	Key return mode register (KRM00)	00H

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

2. In post-reset values are retained in the standby mode.

[MEMO]

CHAPTER 13 μPD78F9026A

The μPD78F9026A is a version with an internal ROM of the mask ROM models replaced with a flash memory. The differences between the μPD78F9026A and the mask ROM models are shown in Table 13-1.

Table 13-1. Differences between μPD78F9026A and Mask ROM Models

Item		Flash Memory Model	Mask ROM Model			
		μPD78F9026A	μPD789022	μPD789024	μPD789025	μPD789026
Internal memory	ROM	16 Kbytes (flash memory)	4 Kbytes	8 Kbytes	12 Kbytes	16 Kbytes
	Internal high-speed RAM	512 bytes	256 bytes		512 bytes	
IC pin		Not provided	Provided			
V_{PP} pin		Provided	Not provided			
Electric characteristics		Refer to Data Sheet.				

Caution The flash memory and masked ROM products have different noise immunity and noise radiation characteristics. Do not use ES products for evaluation when considering switching from flash memory products to those using masked ROM upon the transition from preproduction to mass-production. CS products (masked ROM products) should be used in this case.

13.1 Flash Memory Programming

The program memory provided to the μPD78F9026A is flash memory.

The flash memory can be written on-board, i.e., with the μPD78F9026A mounted on the target system.

To do so, connect a dedicated flash writer (Flashpro III (Part number: FL-PR3, PG-FP3)) to the host machine and target system.

Remark FL-PR3 is a product of Naito Densei Machida Mfg. Co., Ltd.

13.1.1 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 13-2. To select a communication mode, the format shown in Figure 13-1 is used. Each communication mode is selected depending on the number of V_{PP} pulses shown in Table 13-2.

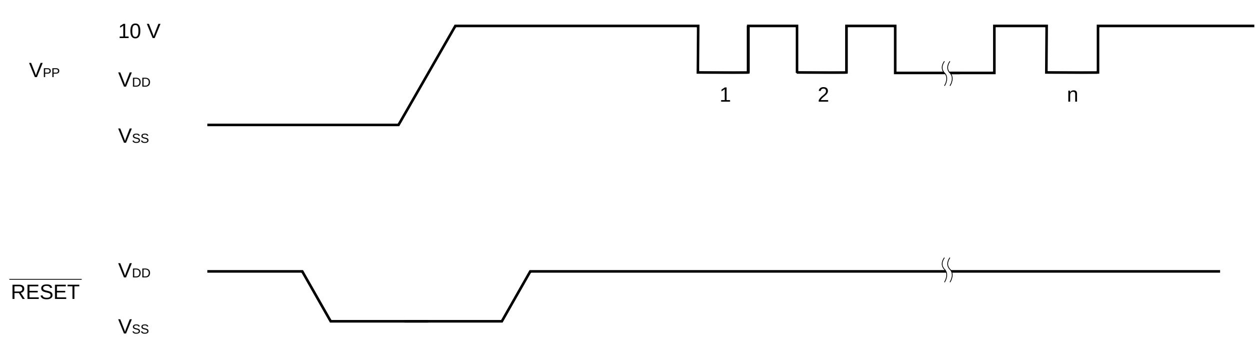
Table 13-2. Communication Modes

Communication Mode	Pins Used	Number of V_{PP} Pulses
3-wire serial I/O	$\overline{\text{SCK0}}$/ASCK/P20 SO0/TxD/P21 SI0/RxD/P22	0
UART	TxD/SO0/P21 RxD/SI0/P22	8
Pseudo 3-wire mode[Note]	P00 (Serial clock input) ★ P01 (Serial data output) ★ P02 (Serial data input)	12
	P40/KR0 (serial clock input) P41/KR1 (serial data output) P42/KR2 (serial data input)	13

Note Serial transfer is performed by controlling the port with software.

Caution Be sure to select a communication mode by the number of V_{PP} pulses shown in Table 13-2.

Figure 13-1. Communication Mode Selection Format



13.1.2 Flash memory programming function

An operation such as writing the flash memory is performed when a command or data is transmitted/received in the selected communication mode. The major flash memory programming functions are listed in Table 13-3.

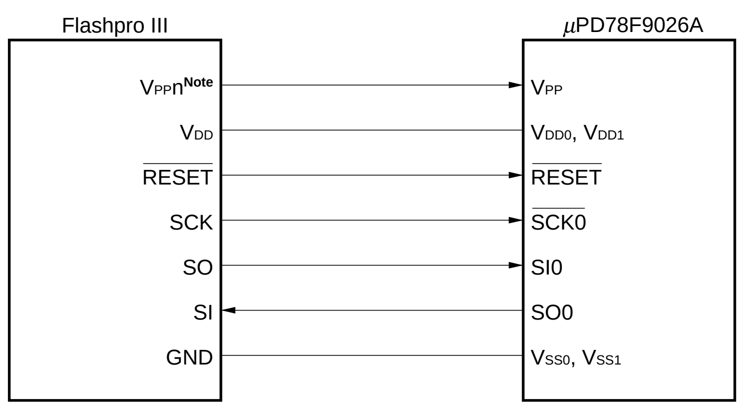
Table 13-3. Major Flash Memory Programming Functions

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

13.1.3 Connection Example of Flashpro III

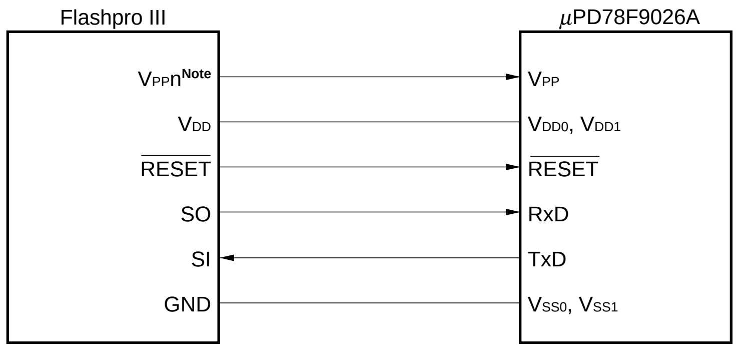
Connection with Flashpro III differs depending on the communication mode (3-wire serial I/O, UART, or pseudo 3-wire mode). Figures 13-2 through 13-4 show the connection in the respective modes.

Figure 13-2. Connection Example of Flashpro III in 3-Wire Serial I/O Mode



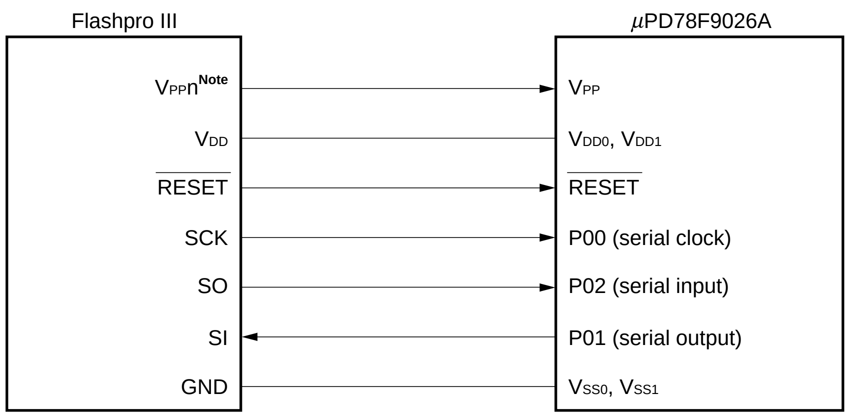
Note n = 1, 2

★ **Figure 13-3. Connection Example of Flashpro III in UART Mode**



Note n = 1, 2

★ **Figure 13-4. Connection Example of Flashpro III in Pseudo 3-Wire Mode (When using P0)**



Note n = 1, 2

★ **13.1.4 Setting example when using Flashpro III (PG-FP3)**

When writing data to flash memory by using Flashpro III (PG-FP3), set as follows.

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

Table 13-4. Setting Example When Using PG-FP3

Communication Mode	Setting Example with PG-FP3		Number of V_{PP} Pulses[Note 1]
3-wire serial I/O	COMM PORT	SIO-ch0	0
	CPU CLK	On Target Board	
		In Flashpro	
	On Target Board	4.1943 MHz	
	SIO CLK	1.0 MHz	
	In Flashpro	4.0 MHz	
	SIO CLK	1.0 MHz	
UART	COMM PORT	UART-ch0	8
	CPU CLK	On Target Board	
	On Target Board	4.1943 MHz	
	UART BPS	9,600 bps[Note 2]	
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 MHz	
	In Flashpro	4.0 MHz	
	SIO CLK	1.0 MHz	
	COMM PORT	Port B	13
	CPU CLK	On Target Board	
		In Flashpro	
	On Target Board	4.1943 MHz	
★	SIO CLK	1 kHz	
	In Flashpro	4.0 MHz	
★	SIO CLK	1 kHz	

Notes 1. This is the number of V_{PP} pulses supplied from Flashpro III during initialization of serial communication. The pin used for the communication is decided depending on this number.

2. Select from 9,600 bps, 19,200 bps, 38,400 bps, or 76,800 bps.

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

[MEMO]

CHAPTER 14 INSTRUCTION SET

This chapter lists the instruction set of the μPD789026 Subseries. For the details of the operation and machine language (instruction code) of each instruction, refer to **78K/0S Series User's Manual — Instruction (U11047E)**.

14.1 Operation

14.1.1 Operand identifiers and writing methods

Operands are written in "Operand" column of each instruction in accordance with the writing method of the instruction operand identifier (refer to the assembler specifications for detail). When there are two or more writing methods, select one of them. Alphabetic letters in capitals and symbols, #, !, $, and [ ] are key words and are written as they are. Each symbol has the following meaning.

- # : Immediate data specification
- ! : Absolute address specification
- $: Relative address specification
- [ ] : Indirect address specification

In the case of immediate data, write an appropriate numeric value or a label. When using a label, be sure to write the #, !, $ and [ ] symbols.

For operand register identifiers, r and rp, either functional names (X, A, C, etc.) or absolute names (names in parentheses in the table below, R0, R1, R2, etc.) can be used.

Table 14-1. Operand Identifiers and Writing Methods

Identifier	Writing Method
r rp sfr	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7) AX (RP0), BC (RP1), DE (RP2), HL (RP3) Special-function register symbol
saddr saddrp	FE20H to FF1FH Immediate data or labels FE20H to FF1FH Immediate data or labels (even addresses only)
addr16 addr5	0000H to FFFFH Immediate data or labels (only even addresses for 16-bit data transfer instructions) 0040H to 007FH Immediate data or labels (even addresses only)
word byte bit	16-bit immediate data or label 8-bit immediate data or label 3-bit immediate data or label

Remark See **Table 3-4** for symbols of special function registers.

14.1.2 Description of "Operation" column

A	:	A register; 8-bit accumulator
X	:	X register
B	:	B register
C	:	C register
D	:	D register
E	:	E register
H	:	H register
L	:	L register
AX	:	AX register pair; 16-bit accumulator
BC	:	BC register pair
DE	:	DE register pair
HL	:	HL register pair
PC	:	Program counter
SP	:	Stack pointer
PSW	:	Program status word
CY	:	Carry flag
AC	:	Auxiliary carry flag
Z	:	Zero flag
IE	:	Interrupt request enable flag
NMIS	:	Flag indicating non-maskable interrupt servicing in progress
( )	:	Memory contents indicated by address or register contents in parentheses
$\times_H$, $\times_L$	:	High-order 8 bits and low-order 8 bits of 16-bit register
$\wedge$	:	Logical product (AND)
$\vee$	:	Logical sum (OR)
$\veebar$	:	Exclusive logical sum (exclusive OR)
—	:	Inverted data
addr16	:	16-bit immediate data or label
jdisp8	:	Signed 8-bit data (displacement value)

14.1.3 Description of "Flag" column

(Blank)	:	Unchanged
0	:	Cleared to 0
1	:	Set to 1
$\times$	:	Set/cleared according to the result
R	:	Previously saved value is restored

14.2 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	Flag AC	Flag 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	Flag AC	Flag CY
CMP	A,#byte	2	4	A – byte	×	×	×
	saddr,#byte	3	6	(saddr) – byte	×	×	×
	A,r	2	4	A – r	×	×	×
	A,saddr	2	4	A – (saddr)	×	×	×
	A,!addr16	3	8	A – (addr16)	×	×	×
	A,[HL]	1	6	A – (HL)	×	×	×
	A,[HL+byte]	2	6	A – (HL+byte)	×	×	×
ADDW	AX,#word	3	6	AX,CY ← AX + word	×	×	×
SUBW	AX,#word	3	6	AX,CY ← AX – word	×	×	×
CMPW	AX,#word	3	6	AX – word	×	×	×
INC	r	2	4	r ← r + 1	×	×	
	saddr	2	4	(saddr) ← (saddr) + 1	×	×	
DEC	r	2	4	r ← r – 1	×	×	
	saddr	2	4	(saddr) ← (saddr) – 1	×	×	
INCW	rp	1	4	rp ← rp + 1			
DECW	rp	1	4	rp ← rp – 1			
ROR	A,1	1	2	$(CY,A_7 \leftarrow A_0,A_{m-1} \leftarrow A_m) \times 1$			×
ROL	A,1	1	2	$(CY,A_0 \leftarrow A_7,A_{m+1} \leftarrow A_m) \times 1$			×
RORC	A,1	1	2	$(CY \leftarrow A_0,A_7 \leftarrow CY,A_{m-1} \leftarrow A_m) \times 1$			×
ROLC	A,1	1	2	$(CY \leftarrow A_7,A_0 \leftarrow CY,A_{m+1} \leftarrow A_m) \times 1$			×
SET1	saddr.bit	3	6	(saddr.bit) ← 1			
	sfr.bit	3	6	sfr.bit ← 1			
	A.bit	2	4	A.bit ← 1			
	PSW.bit	3	6	PSW.bit ← 1	×	×	×
	[HL].bit	2	10	(HL).bit ← 1			
CLR1	saddr.bit	3	6	(saddr.bit) ← 0			
	sfr.bit	3	6	sfr.bit ← 0			
	A.bit	2	4	A.bit ← 0			
	PSW.bit	3	6	PSW.bit ← 0	×	×	×
	[HL].bit	2	10	(HL).bit ← 0			
SET1	CY	1	2	CY ← 1			1
CLR1	CY	1	2	CY ← 0			0
NOT1	CY	1	2	$CY \leftarrow \overline{CY}$			×

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

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

Remark One instruction clock cycle is one CPU clock cycle (f_{CPU}) selected by processor clock control register (PCC).

14.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, INC, DEC, ROR, ROL, RORC, ROLC, PUSH, POP, DBNZ

2nd Operand / 1st Operand	#byte	A	r	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL+byte]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV[Note] XCH[Note] ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV											INC DEC
B, C											DBNZ		
sfr	MOV	MOV											
saddr	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV									DBNZ		INC DEC
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											
[HL+byte]		MOV											

Note Except r = A.

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

2nd Operand / 1st 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

2nd Operand / 1st 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

★

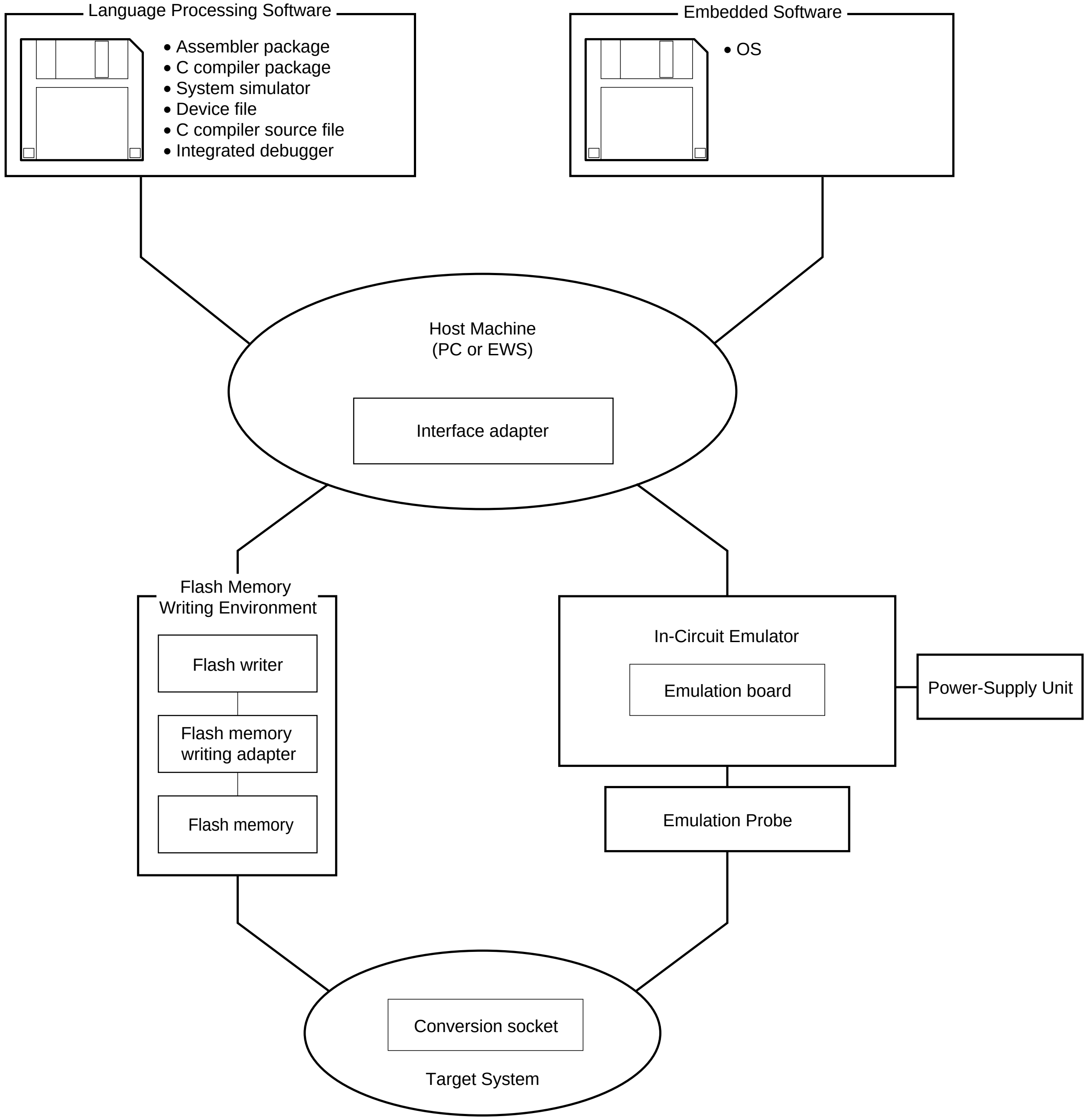
APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for development of systems using the μPD789026 Subseries. Figure A-1 shows development tools.

- Compatibility with PC98-NX series
  Unless stated otherwise, products which are supported for the IBM PC/AT™ compatibles can also be used with the PC98-NX series. When using the PC98-NX series, therefore, refer to the explanations for the IBM PC/AT compatibles.

- Windows
  Unless stated otherwise, "Windows" refers to the following operating systems.
  - Windows 3.1
  - Windows 95
  - Windows NT™ Ver. 4.0

Figure A-1. Development Tools



A.1 Language Processing Software

RA78K0S Assembler package	Program that converts program written in mnemonic into object 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 (DF789026). **<Caution when used under PC environment>** The assembler package is a DOS-based application but may be used under the Windows environment by using Project Manager of Windows (included in the assembler package).
	Part number: μS××××RA78K0S
CC78K0S C compiler package	Program that converts program written in C language into object codes that can be executed by microcontroller. Used in combination with optional assembler package (RA78K0S) and device file (DF789026). **<Caution when used under PC environment>** The C compiler package is a DOS-based application but may be used under the Windows environment by using Project Manager of Windows (included in the assembler package).
	Part number: μS××××CC78K0S
DF789026[Note] Device file	File containing the information inherent to the device. Used in combination with other optional tools (RA78K0S, CC78K0S, SM78K0S).
	Part number: μS××××DF789026
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 package 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 DF789026 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××××DF789026
μS××××CC78K0S-L

××××	Host Machine	OS	Supply Media
AA13	PC-9800 series	Japanese Windows[Note]	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows[Note]	3.5" 2HC FD
3P16	HP9000 series 700™	HP-UX™ (Rel.10.10)	DAT (DDS)
3K13	SPARCstation™	SunOS™ (Rel.4.1.1), Solaris™ (Rel.2.5.1)	3.5" 2HC FD
3K15			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 writer	Flash writer dedicated to microcontrollers with flash memory.
FA-42CU FA-44GB FA-44GB-8ES Flash memory writing adapter	Flash memory writing adapter. Used in connection with Flashpro III. • FA-42CU: For 42-pin plastic shrink DIP (CU type) • FA-44GB: For 44-pin plastic QFP (GB-3BS type) • FA-44GB-8ES: For 44-pin plastic LQFP (GB-8ES type)

Remark FL-PR3, FA-42CU, FA-44GB, and FA-44GB-8ES are products of Naito Densei Machida Mfg. Co., Ltd. For further information, contact: Naito Densei Machida Mfg. Co., Ltd. (044-822-3813)

A.3 Debugging Tools

A.3.1 Hardware

IE-78K0S-NS In-circuit emulator	In-circuit emulator for debugging hardware and software upon developing the 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 (C bus compatible).
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 (PCMCIA socket compatible).
IE-70000-PC-IF-C Interface adapter	This adapter is needed when IBM PC/AT compatibles are used as a host machine of IE-78K0S-NS (ISA bus compatible).
IE-70000-PCI-IF Interface adapter	This adapter is needed when a personal computer incorporating the PCI bus is used as a host machine of IE-78K0S-NS.
IE-789026-NS-EM1 Emulation board	Emulation board for emulating the peripheral hardware inherent to the device. Used in combination with in-circuit emulator.
NP-44GB[Note] Emulation probe	Emulation probe for connecting the in-circuit emulator and target system. This is for the 44-pin plastic QFP (GB-3BS type) and the 44-pin plastic LQFP (GB-8ES type).
EV-9200G-44 Conversion adapter	This conversion adapter is used to connect a target system board designed to allow mounting of the 44-pin plastic QFP (GB-3BS type) and the 44-pin plastic LQFP (GB-8ES type) and the NP-44GB.
NP-44GB-TQ Emulation probe	Emulation probe for connecting the in-circuit emulator and target system. This is for the 44-pin plastic QFP (GB-3BS type) and the 44-pin plastic LQFP (GB-8ES type).
TGB-044SAP Conversion adapter	This conversion adapter is used to connect a target system board designed to allow mounting of the 44-pin plastic QFP (GB-3BS type) and the 44-pin plastic LQFP (GB-8ES type) and the NP-44GB-TQ.

Remarks 1. NP-44GB and NP-44GB-TQ are products of Naito Densei Machida Mfg. Co., Ltd.
For further information, contact: Naito Densei Machida Mfg. Co., Ltd. (044-822-3813)

2. TGB-044SAP is a product of TOKYO ELETECH CORPORATION.
For further information, contact: Daimaru Kougyou, Ltd.
Tokyo Electronics Department (03-3820-7112)
Osaka Electronics Department (06-6244-6672)

A.3.2 Software

ID78K0S-NS Integrated debugger (Supports in-circuit emulator IE78K0S-NS)	Control program for debugging 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 optional device file (DF789026).
	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 compatibles	Japanese Windows[Note]	3.5" 2HC FD

Note Also operates under the DOS environment.

SM78K0S System simulator	Debugs program at C source level or assembler level while simulating operation of target system on host machine. SM78K0S runs 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 optional device file (DF789026).
	Part number: μS××××SM78K0S
DF789026[Note] Device file	File containing the information inherent to the device. Used in combination with other optional tools (RA78K0S, CC78K0S, SM78K0S).
	Part number: μS××××DF789026

Note DF789026 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 system to be used.

μS××××SM78K0S

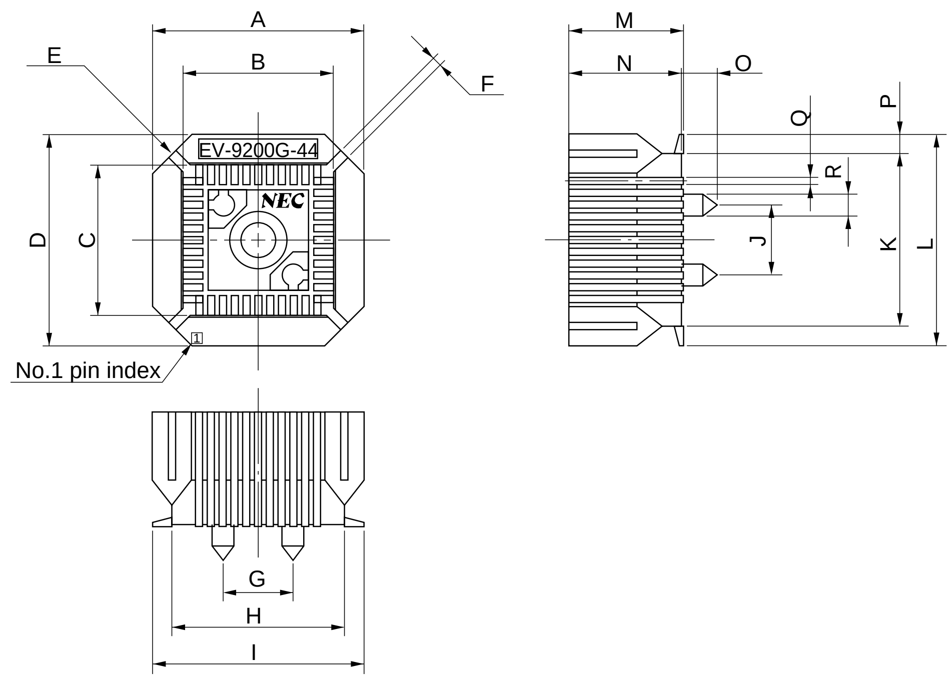
××××	Host Machine	OS	Supply Media
AA13	PC-9800 series	Japanese Windows[Note]	3.5" 2HD FD
AB13	IBM PC/AT compatibles	Japanese Windows[Note]	3.5" 2HC FD

Note Also operates under the DOS environment.

A.4 Conversion Socket (EV-9200G-44) Drawing and Recommended Footprint

Figure A-2. EV-9200G-44 Package Drawing (Reference)

Based on EV-9200G-44
(1) Package drawing (in mm)

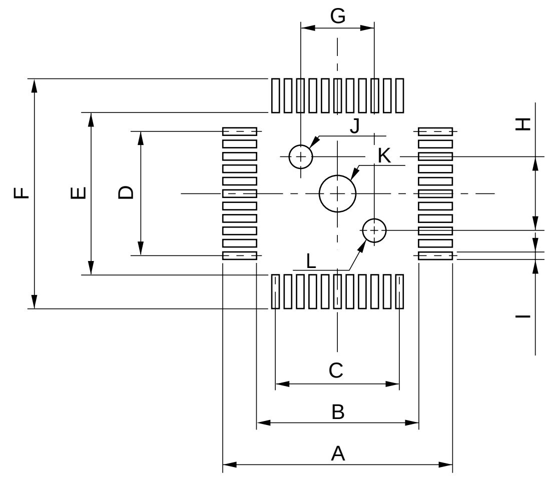


EV-9200G-44-G0

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
O	7.8	0.307
N	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 Footprint (Reference)

Based on EV-9200G-44
(2) Pad drawing (in mm)



EV-9200G-44-P1E

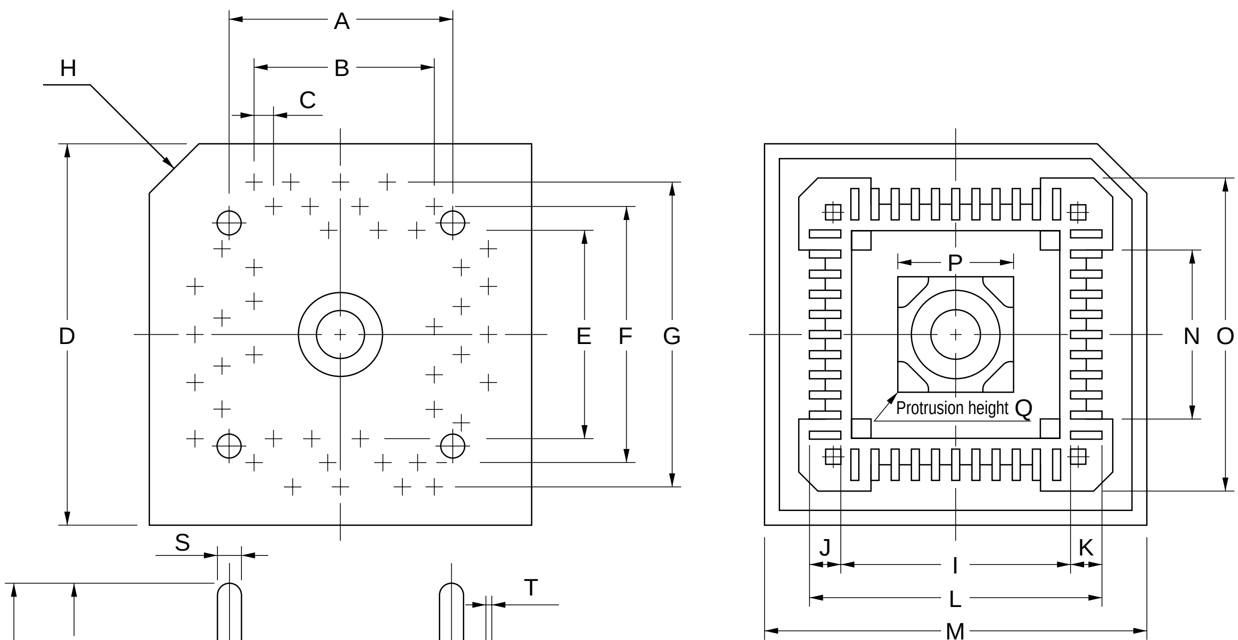
ITEM	MILLIMETERS	INCHES
A	15.7	0.618
B	11.0	0.433
C	$0.8\pm0.02 \times 10 = 8.0\pm0.05$	$0.031^{+0.002}_{-0.001} \times 0.394 = 0.315^{+0.002}_{-0.002}$
D	$0.8\pm0.02 \times 10 = 8.0\pm0.05$	$0.031^{+0.002}_{-0.001} \times 0.394 = 0.315^{+0.002}_{-0.002}$
E	11.0	0.433
F	15.7	0.618
G	5.00 ± 0.08	$0.197^{+0.003}_{-0.004}$
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	$\phi1.57\pm0.03$	$\phi0.062^{+0.001}_{-0.002}$

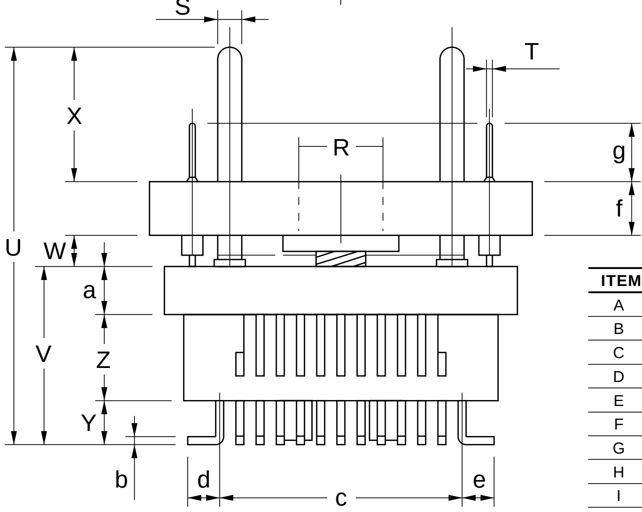
Caution Dimensions of mount pad for EV-9200 and that for target device (QFP) may be different in some parts. For the recommended mount pad dimensions for QFP, refer to "SEMICONDUCTOR DEVICE MOUNTING TECHNOLOGY MANUAL" (C10535E).

A.5 Conversion Adapter (TGB-044SAP) Drawing

Figure A-4. TGB-044SAP Package Drawing (Reference)

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 μPD789026 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 under the PC environment>** The MX78K0S is a DOS-based application. Use this software in the DOS pane when running it on Windows.

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

μS××××MX78K0S

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

Note Also operates under the DOS environment.

[MEMO]

APPENDIX C REGISTER INDEX

C.1 Register Name Index

C.2 Register Symbol Index

[T]

[W]

APPENDIX D REVISION HISTORY

Here is the revision history of this manual. "Chapter" indicates the chapter of the previous edition.

Edition	Revision from Previous Edition	Chapter
Second edition	Change of μPD789025 and μPD789026 from "under development" to "developed"	Throughout
	Change of symbols in Table 3-4	Chapter 3 CPU Architecture
	Change of asynchronous serial interface status register 00 so that it can be manipulated in 1-bit units	
	Change of block diagram of each port in Section 4.2	Chapter 4 Port Functions
	Change of symbols and flag names of 16-bit timer mode control register 20	Chapter 6 16-Bit Timer Counter
	Change of symbols and flag names of 8-bit timer mode control register 00	Chapter 7 8-Bit Timer/Event Counter
	Change of symbols and flag names of serial operation mode register 00	Chapter 9 Serial Interface 00
	Change of symbols and flag names of asynchronous serial interface mode register 00	
	Change of symbols and flag names of asynchronous serial interface status register 00	
	Change of asynchronous serial interface status register 00 so that 1-bit memory manipulation instruction can be used	
	Change of symbols and flag names of baud rate generator control register 00	
	Change of flag names of interrupt request flag register	Chapter 10 Interrupt Functions
	Change of flag names of interrupt mask flag register	
	Change of symbols and flag names of key return mode register 00	
	Addition of description on timing of maskable interrupt request acceptance	
	Addition of setting with Flashpro II	Chapter 13 μPD78F9026
Third edition	Completion of development of μPD789022 and μPD789024	Throughout
	Change of part number from μPD78F9026 to μPD78F9026A	
	Deletion of following products: μPD789022CU-×××, μPD789024CU-×××	
	Addition of GB-8ES type package to all models	
	Change of circuit type and recommended connection of unused pins in processing of input/output circuit type of each pin and unused pins	CHAPTER 2 PIN FUNCTION
	Addition of cautions on rewriting CR20 to operation as timer interrupt	CHAPTER 6 16-BIT TIMER
	Addition of cautions on rewriting CR00 to 8-bit compare register 00 (CR00)	CHAPTER 7 8-BIT TIMER/EVENT COUNTER
	Addition of description of operation to operation as interval timer	
	Addition of description of operation to operation as external event counter	
	Addition of description of operation to operation as square wave output	
	Change of flash writer from Flashpro II to Flashpro III	CHAPTER 13 μPD78F9026A
	Addition of part number of MX78K0S to embedded software	APPENDIX B EMBEDDED SOFTWARE

[MEMO]

Facsimile Message

From:

Name

Company

Tel. FAX

Address

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

Thank you for your kind support.

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

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

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

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

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

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

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

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

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

Document title: ____________________

Document number: ____________________ Page number: __________

If possible, please fax the referenced page or drawing.

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