NEC

User's Manual

μPD789088 Subseries

8-Bit Single-Chip Microcontrollers

μPD789086
μPD789088
μPD78F9088

Document No. U15332EJ3V0UD00 (3rd edition)
Date Published April 2004 NS CP(K)

[MEMO]

NOTES FOR CMOS DEVICES

① VOLTAGE APPLICATION WAVEFORM AT INPUT PIN

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

② HANDLING OF UNUSED INPUT PINS

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

③ PRECAUTION AGAINST ESD

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

④ STATUS BEFORE INITIALIZATION

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

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

- **The information in this document is current as of February, 2004. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".
  The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.
  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).
  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

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

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

M8E 02. 11-1

Regional Information

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

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

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

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

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

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

- **Sucursal en España**
  Madrid, Spain
  Tel: 091-504 27 87
- **Succursale Française**
  Vélizy-Villacoublay, France
  Tel: 01-30-67 58 00
- **Filiale Italiana**
  Milano, Italy
  Tel: 02-66 75 41
- **Branch The Netherlands**
  Eindhoven, The Netherlands
  Tel: 040-244 58 45
- **Tyskland Filial**
  Taeby, Sweden
  Tel: 08-63 80 820
- **United Kingdom Branch**
  Milton Keynes, UK
  Tel: 01908-691-133

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

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

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

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

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

J04.1

[MEMO]

INTRODUCTION

Target Readers

This manual is intended for users who wish to understand the functions of the μPD789088 Subseries and to design and develop application systems and programs using these microcontrollers.
The target devices are the following μPD789088 Subseries products.

Purpose

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

Organization

The μPD789088 Subseries User's Manual is divided into two parts: this manual and instructions (common to the 78K/0S Series)

μPD789088 Subseries User's Manual	78K/0S Series Instructions User's Manual
• Pin functions • Internal block functions • Interrupt functions • Other on-chip peripheral functions • Electrical specifications	• CPU functions • Instruction set • Explanation of each Instruction

How to Read This Manual

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

◊ To understand the functions in general:
→ Read this manual in the order of the contents.
The mark ★ shows major revised points.
◊ How to interpret register format:
→ Where the bit number is enclosed in angle brackets (<>), the bit name is reserved for the assembler and is defined as an sfr variable by the #pragma sfr directive for the C compiler.
◊ When you know a register name and want to confirm its details:
→ See **APPENDIX C REGISTER INDEX**.
◊ To know the 78K/0S Series instruction function in detail:
→ Refer to **78K/0S Series User's Manual Instructions (U11047E)**.
◊ To learn the electrical specifications of the μPD789088 Subseries
→ Refer to **CHAPTER 18 ELECTRICAL SPECIFICATIONS**.

Conventions

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

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

Documents Related to Devices

Document Name	Document No.
μPD789088 Subseries User's Manual	This manual
78K/0S Series Instructions User's Manual	U11047E

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

Document Name		Document No.
RA78K0S Assembler Package	Operation	U16656E
	Language	U14877E
	Structured Assembly Language	U11623E
CC78K0S C Compiler	Operation	U16654E
	Language	U14872E
SM78K Series System Simulator Ver. 2.52	Operation	U16768E
	External Part User Open Interface Specifications	U15802E
ID78K0S-NS Ver. 2.52 Integrated Debugger	Operation	U16584E
PM plus Ver.5.10		U16569E

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

Document Name	Document No.
IE-78K0S-NS In-Circuit Emulator	U13549E
IE-78K0S-NS-A In-Circuit Emulator	U15207E
IE-789088-NS-EM1 Emulation Board	U16398E

Documents Related to Flash Memory Writing

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

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

Other Related Documents

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

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

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

CONTENTS

LIST OF FIGURES (1/4)

LIST OF FIGURES (2/4)

LIST OF FIGURES (3/4)

LIST OF FIGURES (4/4)

LIST OF TABLES (1/2)

LIST OF TABLES (2/2)

CHAPTER 1 GENERAL

1.1 Features

- ROM and RAM capacity

Item / Product Name	Program Memory (ROM)		Data Memory	
			Internal High-Speed RAM	Internal Low-Speed RAM
μPD789086	Mask ROM	16 KB	256 bytes	128 bytes
μPD789088		32 KB	320 bytes	256 bytes
μPD78F9088	Flash memory	32 KB		

- Minimum instruction execution time can be changed to high-speed (0.4 μs), medium-speed (0.8 μs), and low-speed (1.6 μs) (@ 5.0 MHz operation with system clock, V_{DD} = 2.7 to 5.5 V)
- I/O ports: 24
- Serial interface: 1 channel: Switchable between 3-wire serial I/O and UART modes
- Timers: 5 channels
  - 16-bit timer: 1 channel
  - 8-bit timer: 3 channels
  - Watchdog timer: 1 channel
- On-chip key return signal detector
- On-chip power-on-clear circuit and low-voltage detector
- Power supply voltage: V_{DD} = 1.8 to 5.5 V

1.2 Applications

Preset remote controllers, compact home electrical appliances, game machines, etc.

1.3 Ordering Information

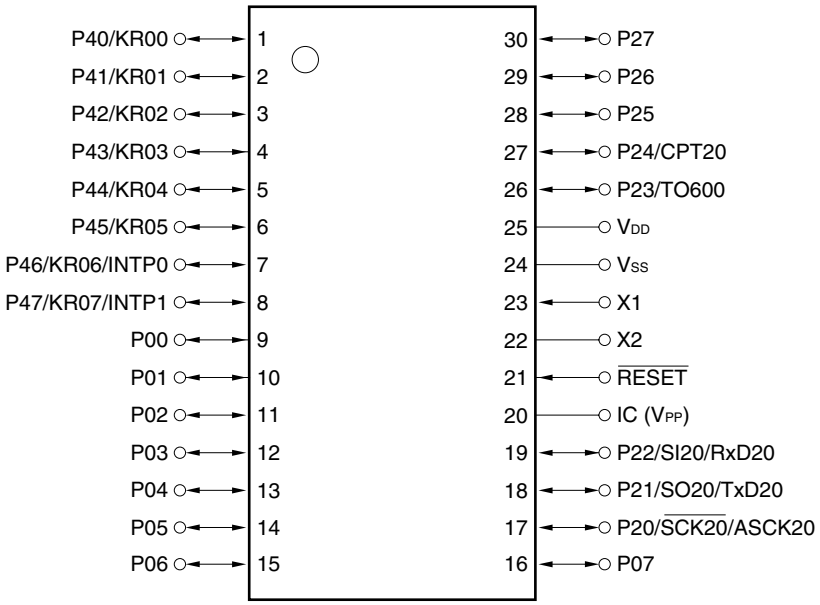
Part Number	Package	Internal ROM
μPD789086MC-×××-5A4	30-pin plastic SSOP (7.62 mm (300))	Mask ROM
μPD789088MC-×××-5A4	30-pin plastic SSOP (7.62 mm (300))	Mask ROM
μPD78F9088M1MC-5A4	30-pin plastic SSOP (7.62 mm (300))	Flash memory

Remark ××× indicates ROM code suffix.

1.4 Pin Configuration (Top View)

30-pin plastic SSOP (7.62 mm (300))

μPD789086MC-×××-5A4
μPD789088MC-×××-5A4
μPD78F9088M1MC-5A4



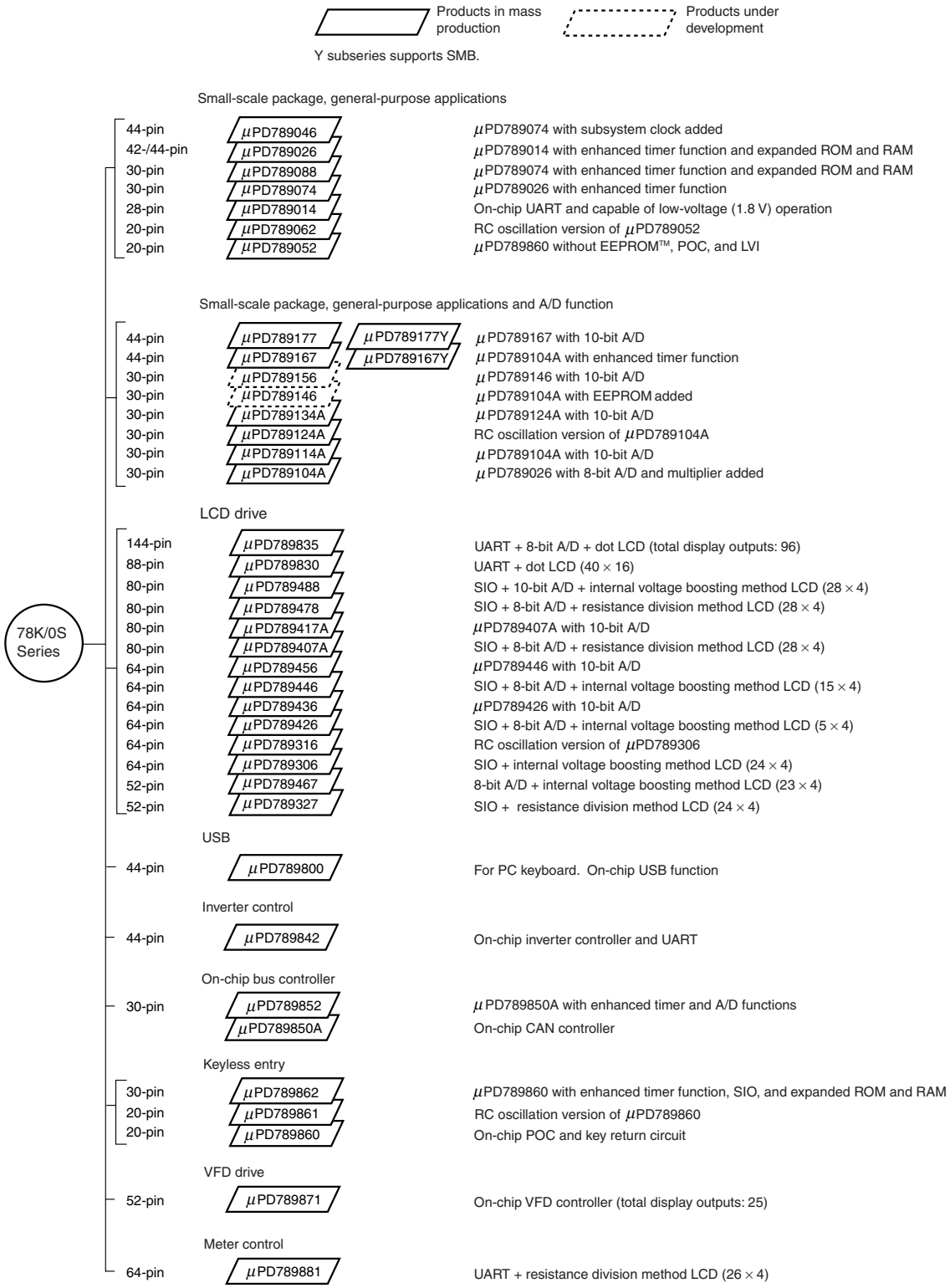
Caution Connect the IC (Internally Connected) pin directly to V_{SS}.

Remark Pin connections in parentheses are intended for the μPD78F9088.

ASCK20:	Asynchronous serial input	$\overline{SCK20}$:	Serial clock input/output
CPT20:	Capture trigger input	SI20:	Serial data input
IC:	Internally connected	SO20:	Serial data output
INTP0, INTP1:	External Interrupt Input	TO600:	Timer output
KR00 to KR07:	Key return	TxD20:	Transmit data
P00 to P07:	Port 0	V_{DD}:	Power supply
P20 to P27:	Port 2	V_{PP}:	Programming power supply
P40 to P47:	Port 4	V_{SS}:	Ground
$\overline{RESET}$:	Reset	X1, X2:	Crystal 1, 2
RxD20:	Receive data		

1.5 78K/0S Series Lineup

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



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

The major functional differences between the subseries are listed below.

Series for general-purpose applications and LCD drive

Subseries \ Function		ROM Capacity (Bytes)	Timer				8-Bit A/D	10-Bit A/D	Serial Interface	I/O	V_{DD} MIN.Value	Remarks
			8-Bit	16-Bit	Watch	WDT						
Small-scale package, general-purpose applications	μPD789046	16 K	1 ch	1 ch	1 ch	1 ch	–	–	1 ch (UART: 1 ch)	34	1.8 V	–
	μPD789026	4 K to 16 K			–							
	μPD789088	16 K to 32 K	3 ch							24		
	μPD789074	2 K to 8 K	1 ch									
	μPD789014	2 K to 4 K	2 ch	–						22		
	μPD789062	4 K							–	14		RC-oscillation version
	μPD789052											–
Small-scale package, general-purpose applications + A/D converter	μPD789177	16 K to 24 K	3 ch	1 ch	1 ch	1 ch	–	8 ch	1 ch (UART: 1 ch)	31	1.8 V	–
	μPD789167						8 ch	–				
	μPD789156	8 K to 16 K	1 ch		–		–	4 ch		20		On-chip EEPROM
	μPD789146						4 ch	–				
	μPD789134A	2 K to 8 K					–	4 ch				RC-oscillation version
	μPD789124A						4 ch	–				
	μPD789114A						–	4 ch				–
	μPD789104A						4 ch	–				
LCD drive	μPD789835	24 K to 60 K	6 ch	–	1 ch	1 ch	3 ch	–	1 ch (UART: 1 ch)	37	1.8 V[Note]	Dot LCD supported
	μPD789830	24 K	1 ch	1 ch			–			30	2.7 V	
	μPD789489	32 K to 48 K	3 ch					8 ch	2 ch (UART: 1 ch)	45	1.8 V	–
	μPD789479	24 K to 48 K					8 ch	–				
	μPD789417A	12 K to 24 K					–	7 ch	1 ch (UART: 1 ch)	43		
	μPD789407A						7 ch	–				
	μPD789456	12 K to 16 K	2 ch				–	6 ch		30		
	μPD789446						6 ch	–				
	μPD789436						–	6 ch		40		
	μPD789426						6 ch	–				
	μPD789316	8 K to 16 K					–		2 ch (UART: 1 ch)	23		RC-oscillation version
	μPD789306											–
	μPD789467	4 K to 24 K		–			1 ch		–	18		
	μPD789327						–		1 ch	21		

Note Flash memory version: 3.0 V

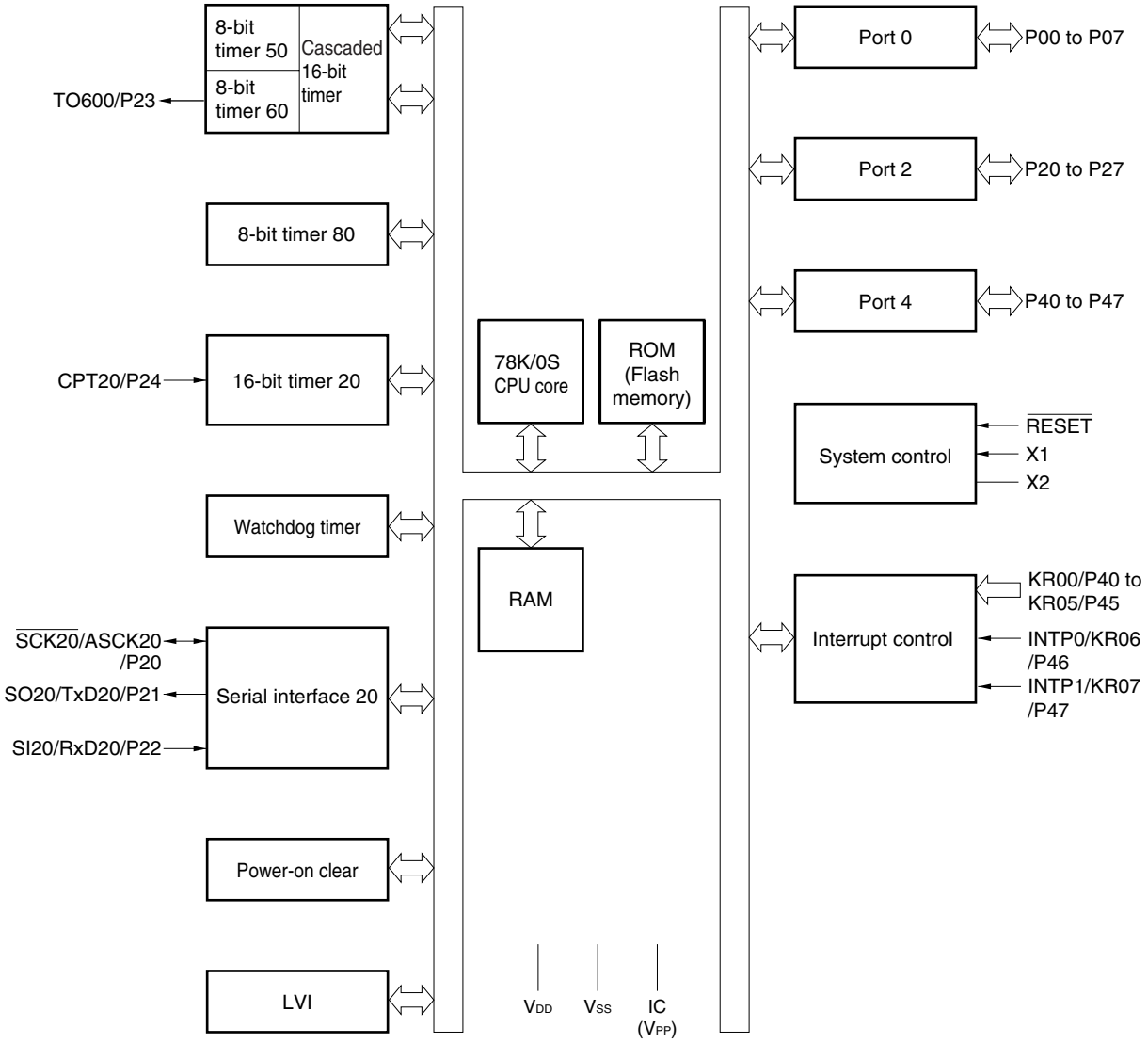
Series for ASSP

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

Notes 1. 10-bit timer: 1 channel

2. Flash memory version: 3.0 V

1.6 Block Diagram



Remarks 1. Internal ROM and RAM capacities vary depending on the product.

2. Pin connections in parentheses are intended for the μPD78F9088.

1.7 Overview of Functions

Item		μPD789086	μPD789088	μPD78F9088
Internal memory	ROM	Mask ROM		Flash memory
		16 KB	32 KB	
	High-speed RAM	256 bytes	320 bytes	
	Low-speed RAM	128 bytes	256 bytes	
Minimum instruction execution time (@ 5.0 MHz operation with system clock)		0.4 μs (V_{DD} = 2.7 to 5.5 V[Note 1]) 0.8 μs (V_{DD} = 2.0 to 5.5 V[Note 1, 2]) 1.6 μs (V_{DD} = 1.8 to 5.5 V[Note 1, 2])		
System clock multiply function		x2 circuit (operation supply voltage: 2.0 to 3.6 V, use of this function can be selected by means of software)		
General-purpose registers		8 bits × 8 registers		
Instruction set		• 16-bit operations • Bit manipulation (set, reset, and test)		
I/O ports		CMOS I/O: 24		
Serial interface		Switchable between 3-wire serial I/O and UART modes: 1 channel		
Timer		• 16-bit timer: 1 channel • 8-bit timer: 3 channels • Watchdog timer: 1 channel		
Timer output		1 output		
Power-on-clear (POC) circuit		Reset is generated at 2.15 ±0.15 V (use of POC can be selected by means of software).		
Low-voltage detector (LVI)		1.5 ±0.1 V is detected.		
Vectored interrupt sources	Maskable	Internal: 8, external: 3		
	Non-maskable	Internal: 1		
Power supply voltage		V_{DD} = 1.8 to 5.5 V[Note 1, 2]		
Operating ambient temperature		T_A = –40 to +85°C		
Package		30-pin plastic SSOP (7.62 mm (300))		

Notes 1. When the clock multiplication circuit is used, the operating voltage range is 2.0 to 3.6 V.

2. When the POC circuit is used, the POC voltage will be the minimum operating voltage.

The outline of the timer is as follows.

		16-Bit Timer 20	8-Bit Timer 50	8-Bit Timer 60	8-Bit Timer 80	Watchdog Timer
Operation mode	Interval timer	–	1 channel	1 channel	1 channel	1 channel[Note]
	External event counter	–	–	–	–	–
Function	Timer output	–	–	1 output	–	–
	PWM output	–	–	1 output	–	–
	Square wave output	–	–	1 output	–	–
	Carrier generator output	–	1 output		–	–
	Capture	1 input	–	–	–	–
	Interrupt request	1	1	1	1	1

Note The watchdog timer provides the watchdog timer function and interval timer function. Use either of the functions.

CHAPTER 2 PIN FUNCTIONS

2.1 Pin Function List

(1) Port pins

Pin Name	I/O	Function	After Reset	Alternate Function
P00 to P07	I/O	Port 0 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register B0 (PUB0).	Input	–
P20	I/O	Port 2 8-bit I/O port Input/output can be specified in 1-bit units.	Input	$\overline{\text{SCK20}}$/ASCK20
P21				SO20/TxD20
P22				SI20/RxD20
P23				TO600
P24				CPT20
P25 to P27				–
P40 to P45	I/O	Port 4 8-bit I/O port Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register B4 (PUB4).	Input	KR00 to KR05
P46				KR06/INTP0
P47				KR07/INTP1

(2) Non-port pins

Pin Name	I/O	Function	After Reset	Alternate Function
INTP0	Input	External interrupt request input for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified	Input	P46/KR06
INTP1				P47/KR07
SI20	Input	Serial interface serial data input	Input	P22/RxD20
SO20	Output	Serial interface serial data output	Input	P21/TxD20
$\overline{\text{SCK20}}$	I/O	Serial interface serial clock input/output	Input	P20/ASCK20
ASCK20	Input	Serial clock input for asynchronous serial interface	Input	P20/$\overline{\text{SCK20}}$
RxD20	Input	Serial data input for asynchronous serial interface	Input	P22/SI20
TxD20	Output	Serial data output for asynchronous serial interface	Input	P21/SO20
TO600	Output	8-bit timer 60 output	Input	P23
CPT20	Input	Capture edge input	Input	P24
KR00 to KR05	Input	Key return signal detector	Input	P40 to P45
KR06				P46/INTP0
KR07				P47/INTP1
X1	Input	Connecting crystal resonator for main system clock oscillation	–	–
X2	–		–	–
$\overline{\text{RESET}}$	Input	System reset input	Input	–
V_{DD}	–	Positive power supply	–	–
V_{SS}	–	Ground potential	–	–
IC	–	Internally connected. Connect directly to V_{SS}.	–	–
V_{PP}	–	This pin is used to set the flash memory programming mode and applies a high voltage when a program is written or verified.	–	–

2.2 Description of Pin Functions

2.2.1 P00 to P07 (Port 0)

These pins constitute an 8-bit I/O port and can be set to input or output port mode in 1-bit units by using port mode register 0 (PM0). When these pins are used as an input port, an on-chip pull-up resistor can be used in 1-bit units by setting pull-up resistor option register B0 (PUB0).

2.2.2 P20 to P27 (Port 2)

These pins constitute an 8-bit I/O port. In addition, these pins provide a function to perform input/output to/from the timer and to input/output the data and clock of the serial interface.

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

(1) Port mode

In port mode, P20 to P27 function as an 8-bit I/O port. Port 2 can be set to input or output mode in 1-bit units by using port mode register 2 (PM2).

(2) Control mode

In this mode, P20 to P27 function as the timer input/output and the serial interface data and clock input/output.

(a) TO600

This is the timer output pin of the 8-bit timer 60.

(b) CPT20

This is the capture edge input pin of the 16-bit timer counter 20.

(c) SI20, SO20

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

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

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

(e) RxD20, TxD20

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

(f) ASCK20

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

Caution When using P20 to P27 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 10-2 Serial Interface 20 Operation Mode Settings.

2.2.3 P40 to P47 (Port 4)

These pins constitute an 8-bit I/O port. In addition, these pins function as key return signal detection and external interrupt input.

Port 4 can be set to the following operation modes in 1-bit units.

(1) Port mode

When this port is used as an input port, an on-chip pull-up resistor can be used in 1-bit units by setting pull-up resistor option register B4 (PUB4).

(2) Control mode

In this mode, P30 and P31 function as key return signal detection and external interrupt input.

(a) KR00 to KR01

This is the key return signal detection pin.

(b) INTP0, INTP1

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

2.2.4 $\overline{\text{RESET}}$

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

2.2.5 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.

2.2.6 V_{DD}

This pin supplies positive power.

2.2.7 V_{SS}

This pin is the ground potential pin.

2.2.8 V_{PP} (μPD78F9088 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.

Handle the pins in either of the following ways.

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

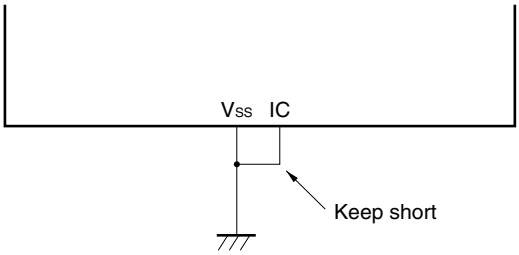
If the wiring between the V_{PP} pin and V_{SS} pin is long, or external noise is superimposed on the V_{PP} pin, the user program may not run correctly.

2.2.9 IC (mask ROM version only)

The IC (Internally Connected) pin is used to set the μPD789086 and 789088 to test mode before shipment. In normal operation mode, directly connect this pin to the V_{SS} pin with as short a wiring length as possible.

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

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



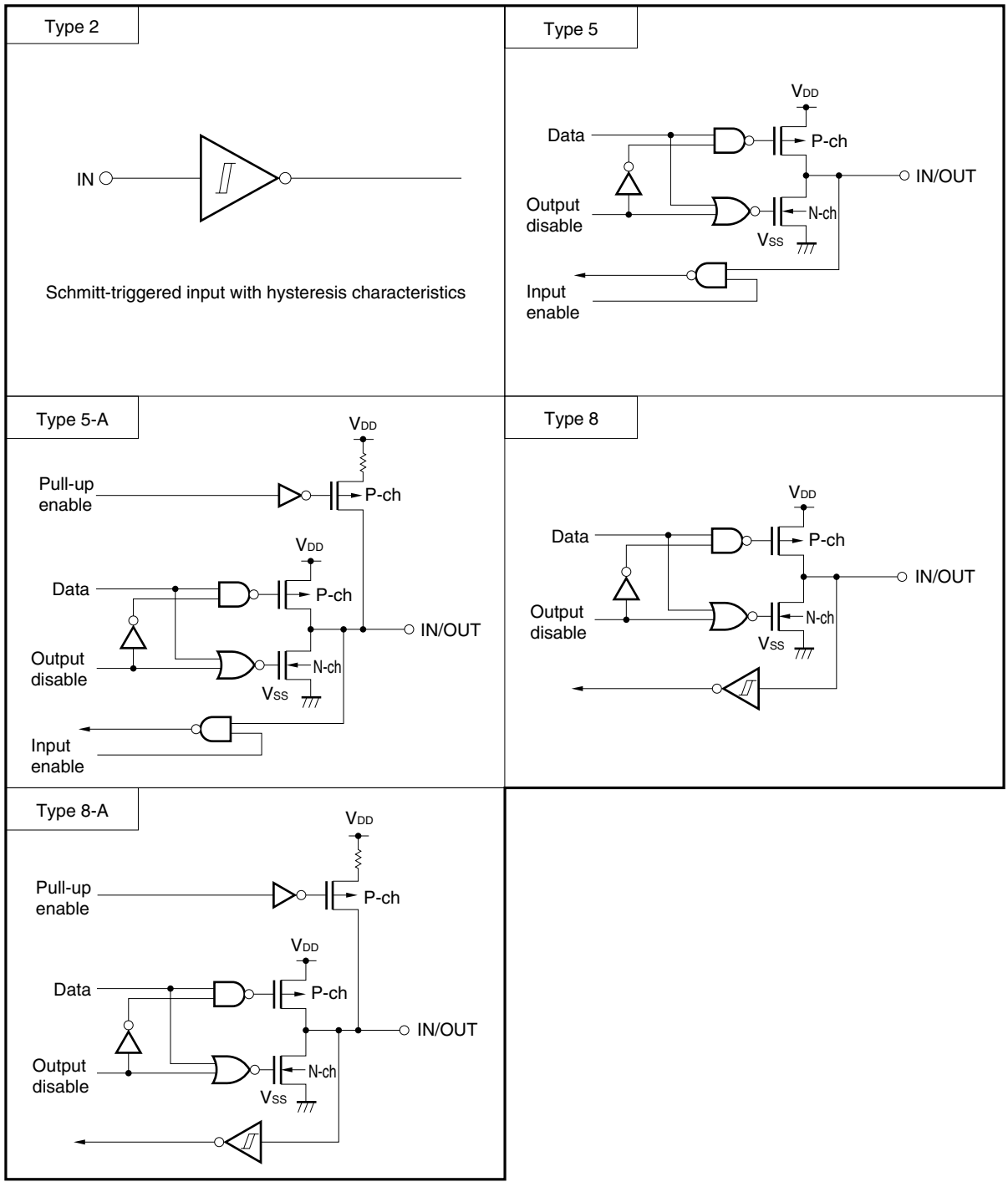
2.3 Pin I/O Circuits and Recommended Connection of Unused Pins

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

Table 2-1. Types of Pin I/O Circuits and Recommended Connection of Unused Pins

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
P00 to P07	5-A	I/O	Input: Independently connect to V_{DD} or V_{SS} via a resistor. Output: Leave open.
P20/$\overline{SCK20}$/ASCK20	8		
P21/SO20/TxD20	5		
P22/SI20/RxD20	8		
P23/TO600	5		
P24/CPT20	8		
P25 to P27	5		
P40 to P45	8-A		
P46/KR06/INTP0			Input: Independently connect to V_{SS} via a resistor. Output: Leave open.
P47/KR07/INTP1			
$\overline{RESET}$	2	Input	–
IC	–	–	Connect directly to V_{SS}.
V_{PP}			Independently connect a 10 kΩ pull-down resistor or directly connect to V_{SS}.

Figure 2-1. Pin I/O Circuits



CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Space

The μPD789088 Subseries can each access up to 64 KB of memory space. Figures 3-1 through 3-3 show the memory maps.

Figure 3-1. Memory Map (μPD789086)

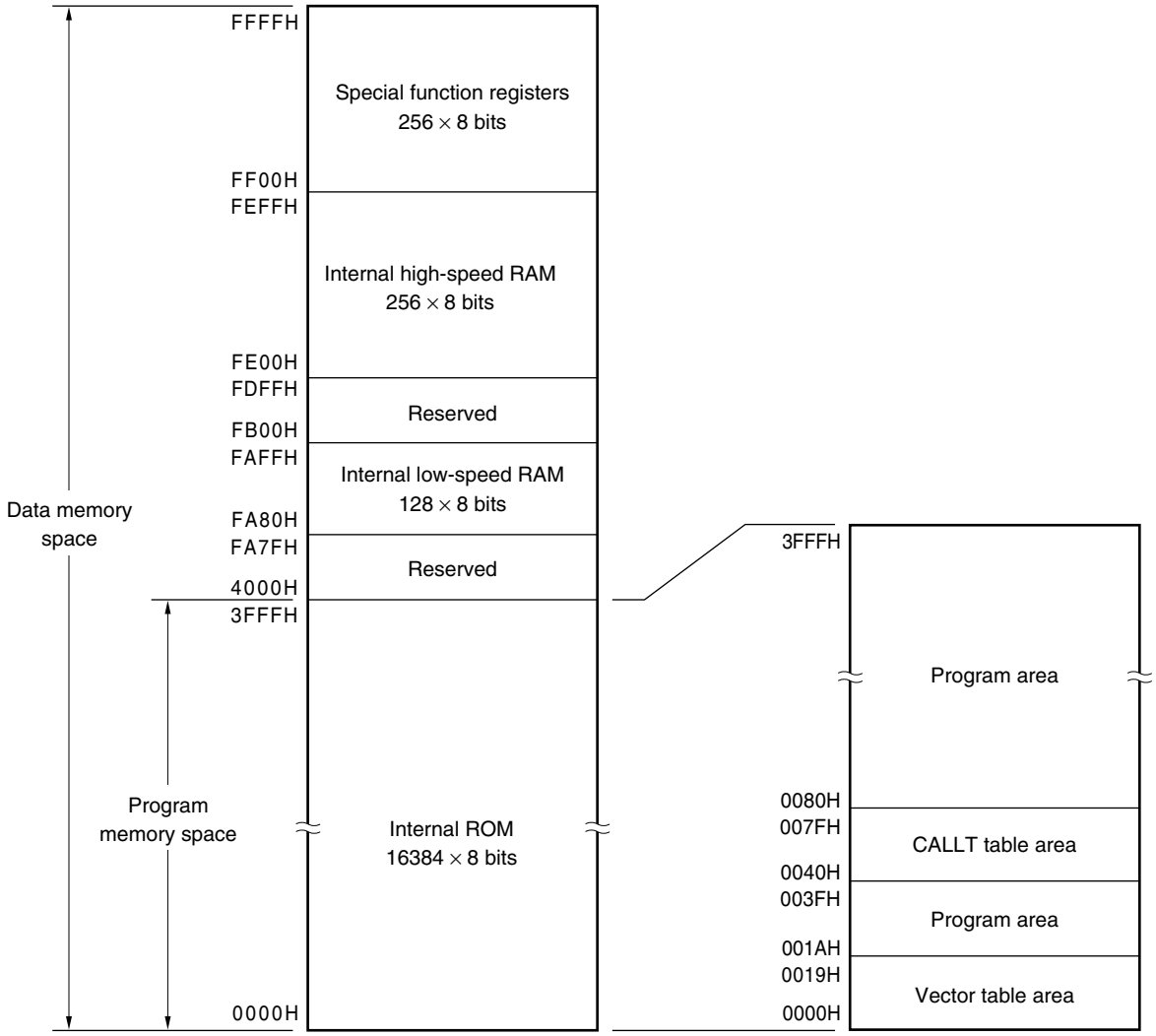


Figure 3-2. Memory Map (μPD789088)

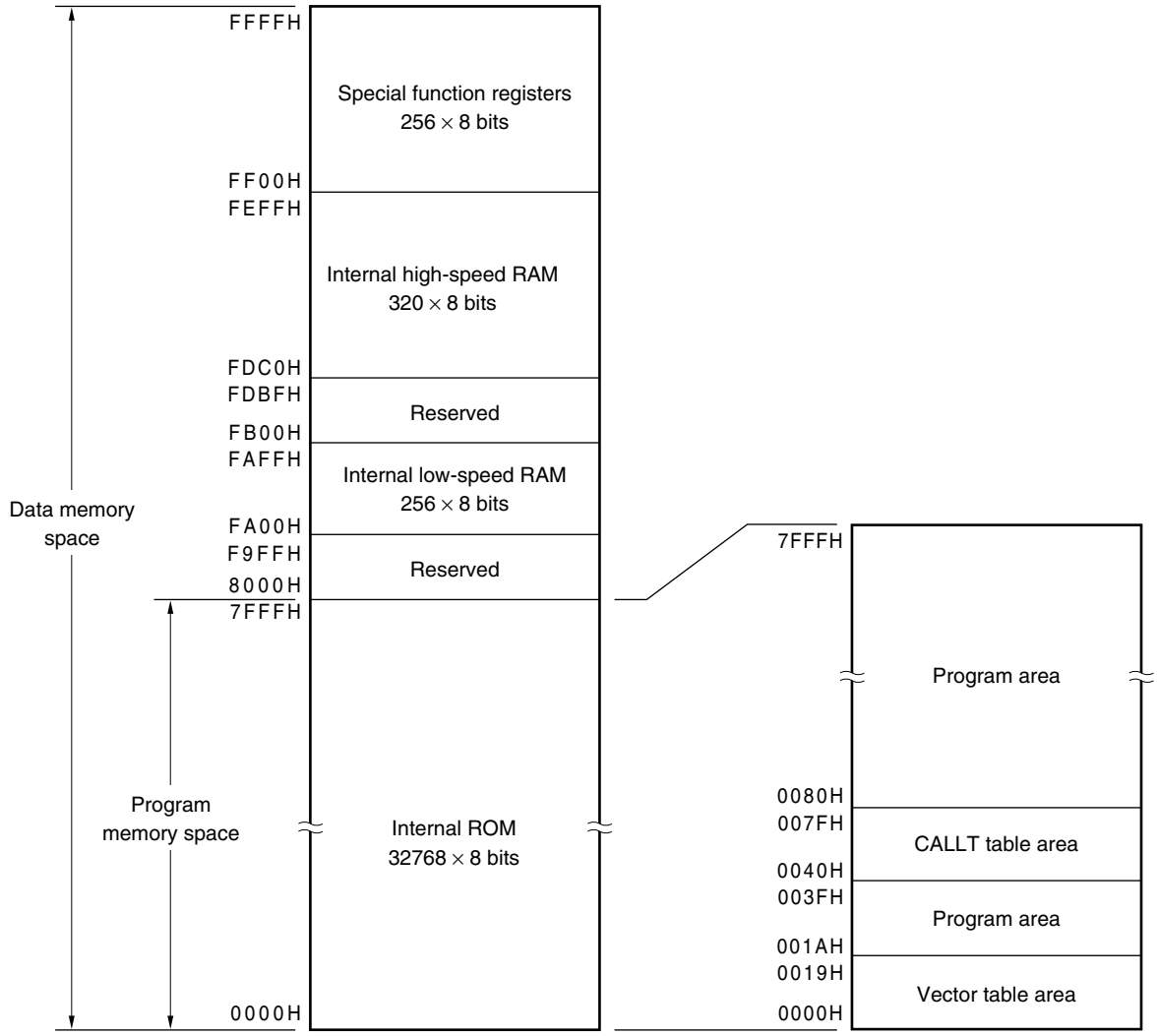
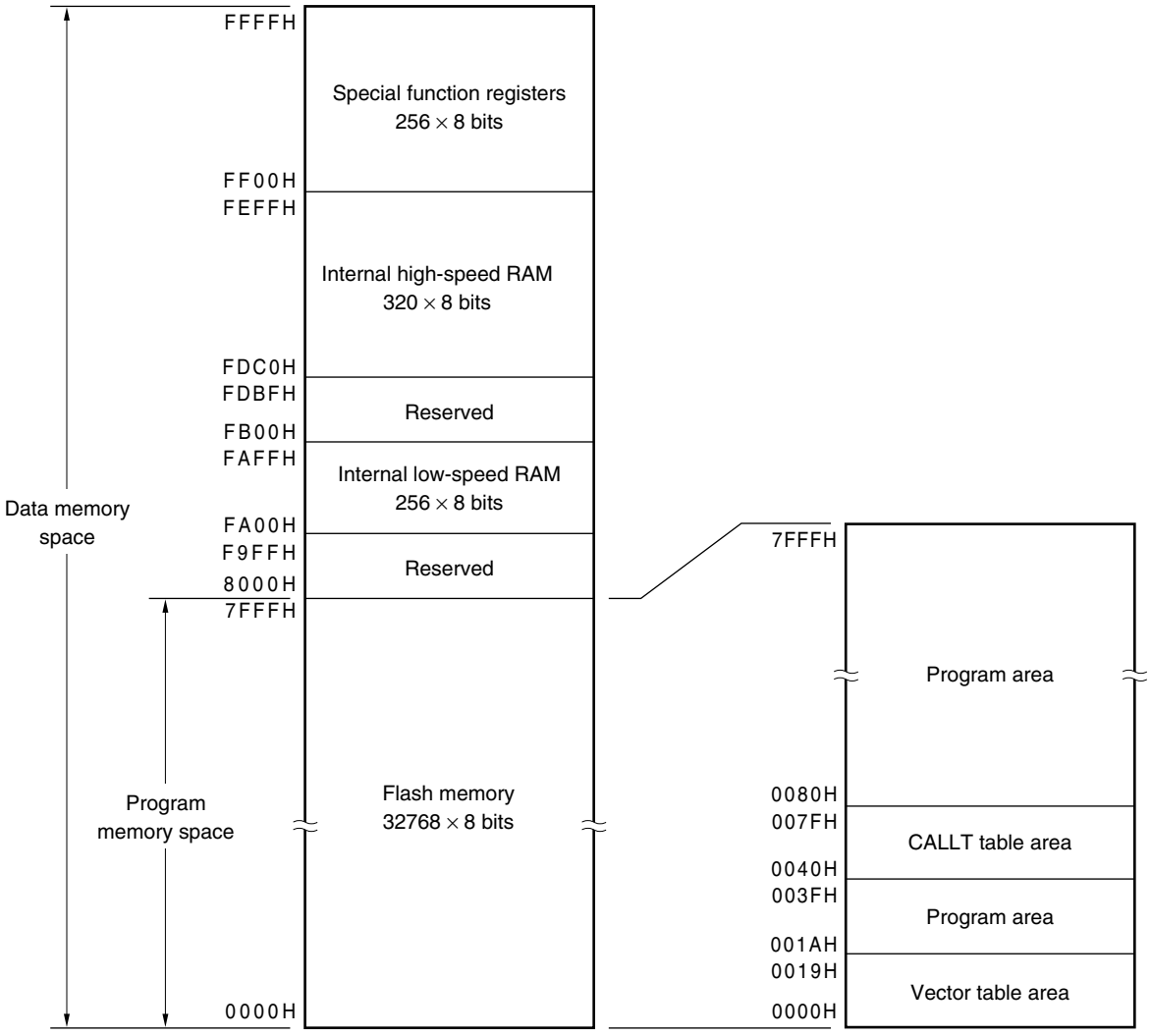


Figure 3-3. Memory Map (μPD78F9088)



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 μPD789088 Subseries provide the following internal ROMs (or flash memory) containing the following capacities.

Table 3-1. Internal ROM Capacity

Part Number	Internal ROM	
	Structure	Capacity
μPD789086	Mask ROM	16,384 × 8 bits
μPD789088		32,768 × 8 bits
μPD78F9088	Flash memory	32,768 × 8 bits

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

(1) Vector table area

The 26-byte area of addresses 0000H to 0019H is reserved as a vector table area. This area stores program start addresses to be used when branching by $\overline{\text{RESET}}$ input or interrupt request generation. Of a 16-bit program address, the lower 8 bits are stored in an even address, and the higher 8 bits are stored in an odd address.

Table 3-2. Vector Table

Vector Table Address	Interrupt Request	Vector Table Address	Interrupt Request
0000H	$\overline{\text{RESET}}$ input	000CH	INTTM60
0004H	INTWDT	000EH	INTTM80
0006H	INTP0	0010H	INTTM20
0008H	INTP1	0012H	INTKR00
000AH	INTTM50	0014H	INTSR20/INTCSI20
		0016H	INTST20

(2) CALLT instruction table area

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

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

The μPD789088 Subseries includes internal high-speed RAM and internal low-speed RAM with capacities as shown below for each product.

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

The internal low-speed RAM cannot be used as a stack memory.

Table 3-3. Capacities of Internal High-Speed RAM and Internal Low-Speed RAM

Part Number	Internal High-Speed Capacity	Internal Low-Speed Capacity
μPD789086	256 × 8 bits	128 × 8 bits
μPD789088	320 × 8 bits	256 × 8 bits
μPD78F9088		

3.1.3 Special function register (SFR) area

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

3.1.4 Data memory addressing

Each of the μPD789088 Subseries is provided with a wide range of addressing modes to make memory manipulation as efficient as possible. The data memory area (FE00H to FFFFH) can be accessed using a unique addressing mode according to its use, such as a special function register (SFR). Figures 3-4 through 3-6 illustrate the data memory addressing.

Figure 3-4. Data Memory Addressing (μPD789086)

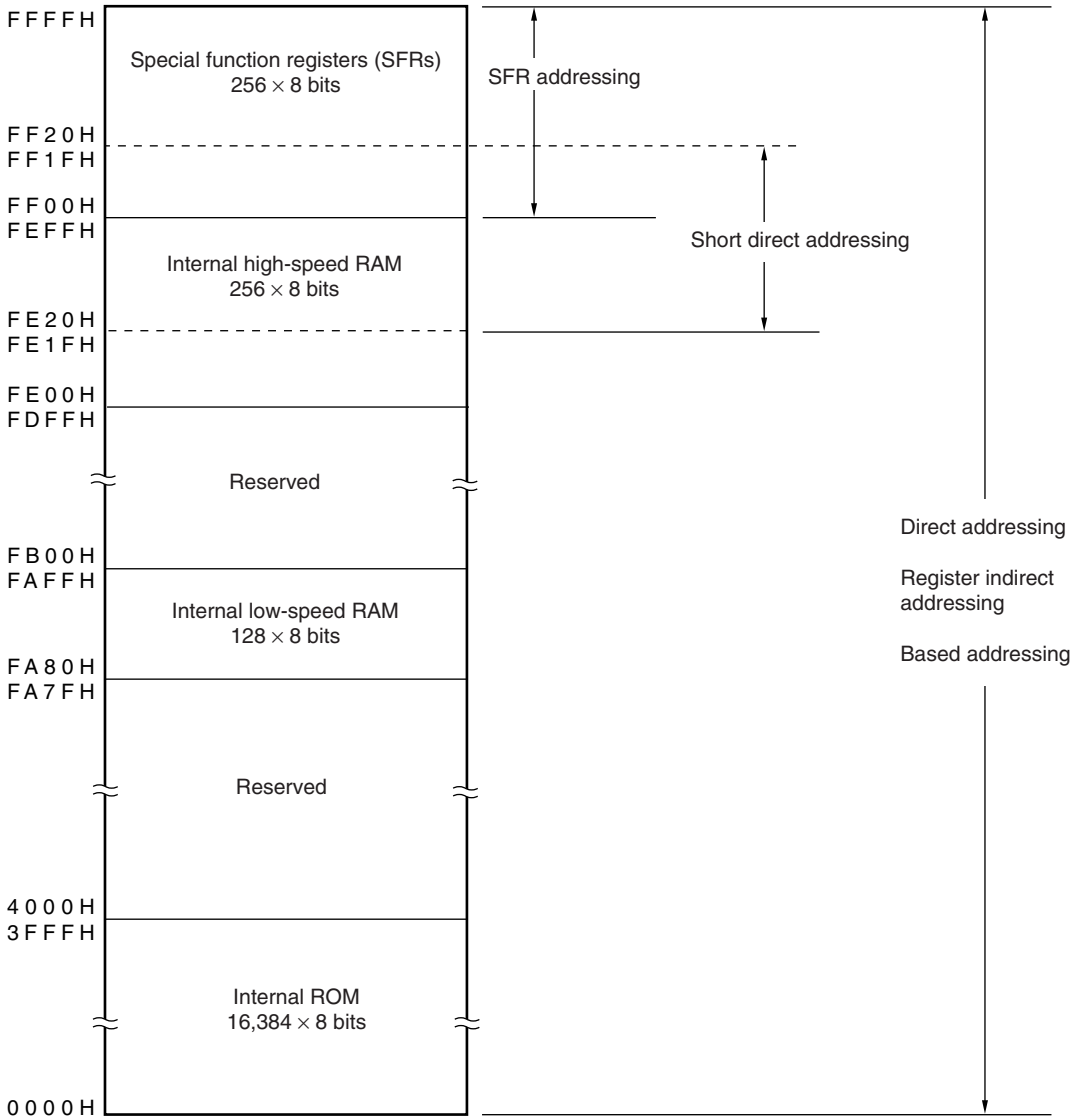


Figure 3-5. Data Memory Addressing (μPD789088)

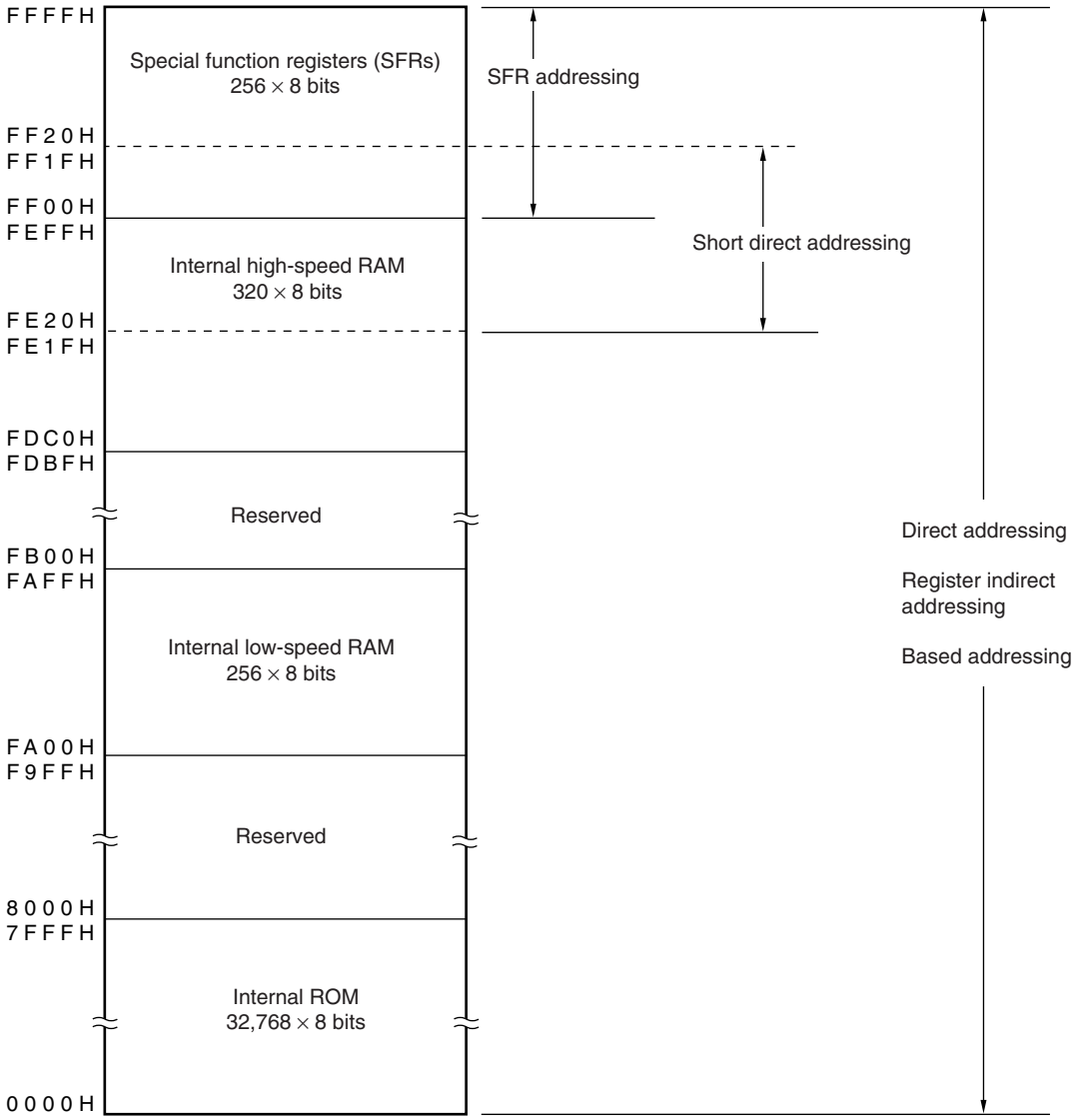
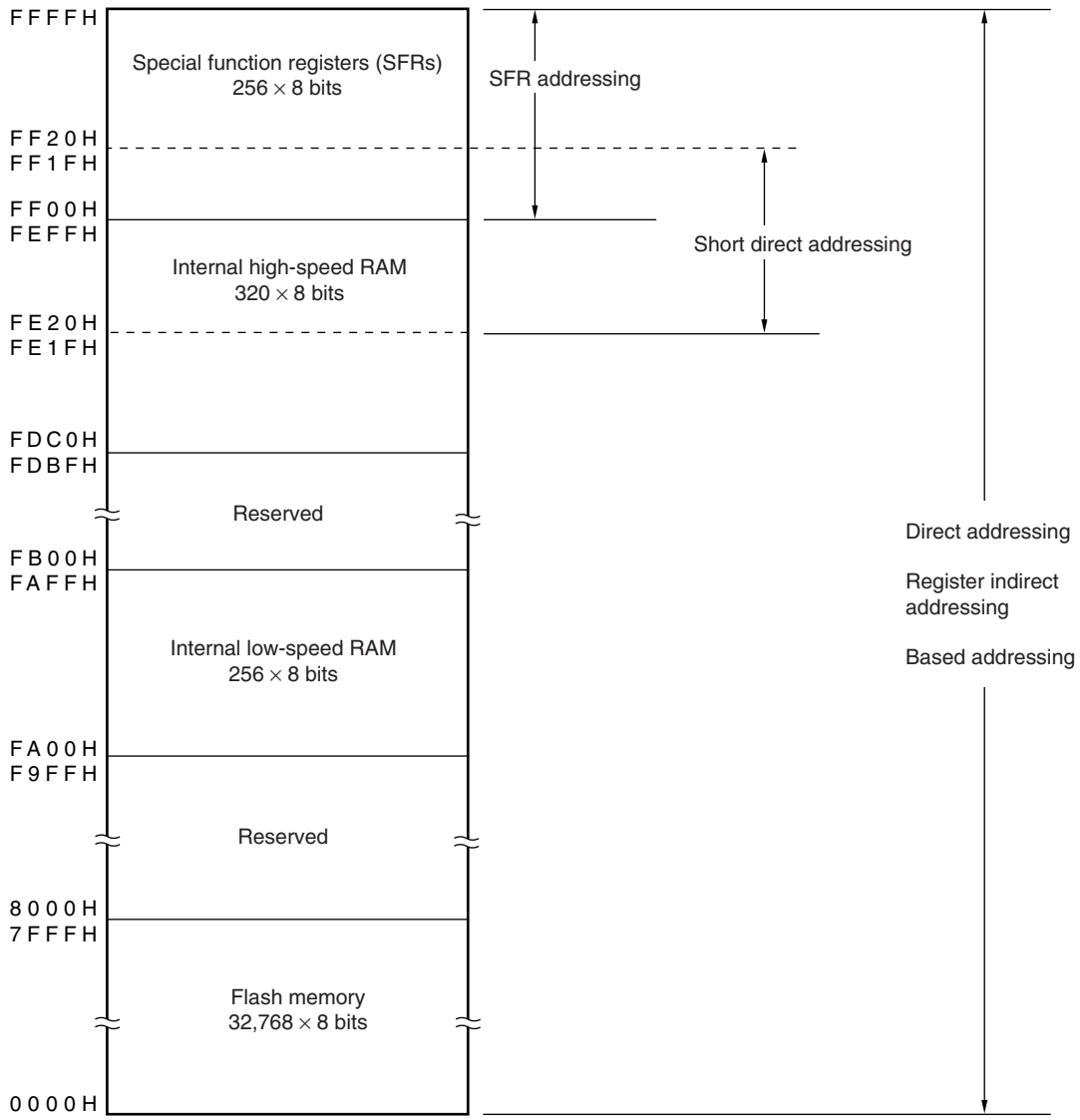


Figure 3-6. Data Memory Addressing (μPD78F9088)



3.2 Processor Registers

The μPD789088 Subseries provide the following on-chip processor registers:

3.2.1 Control registers

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

(1) Program counter (PC)

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

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

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

Figure 3-7. Program Counter Configuration

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

(2) Program status word (PSW)

The program status word is an 8-bit register consisting of various flags to be set/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 3-8. Program Status Word Configuration

	7							0
PSW	IE	Z	0	AC	0	0	1	CY

(a) Interrupt enable flag (IE)

This flag controls interrupt request acknowledge operations of 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. Interrupt request acknowledgment is controlled with an interrupt mask flag for various interrupt sources.

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

(b) Zero flag (Z)

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

(c) Auxiliary carry flag (AC)

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

(d) Carry flag (CY)

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

(3) Stack pointer (SP)

This is a 16-bit register to hold the start address of the memory stack area. Only the internal high-speed RAM area can be set as the stack area.

Figure 3-9. Stack Pointer Configuration

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

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

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

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

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

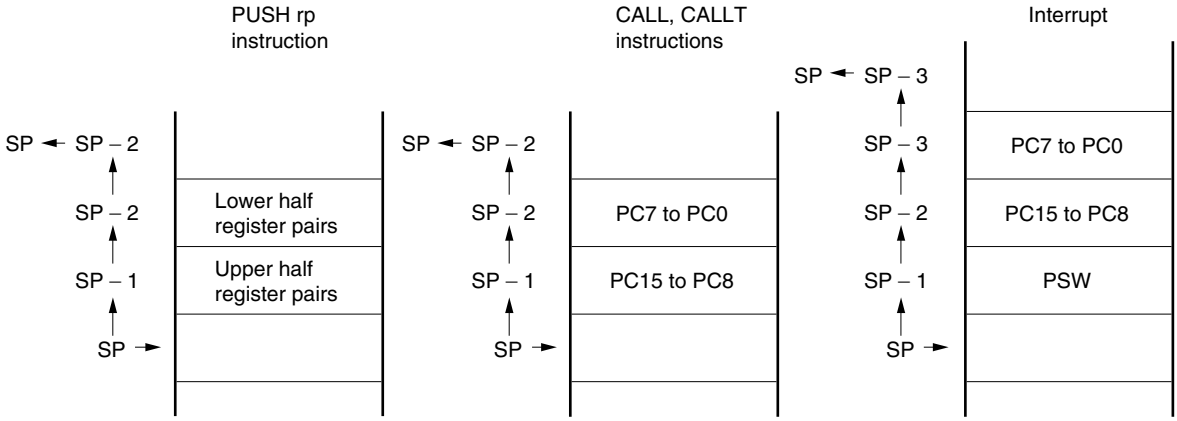
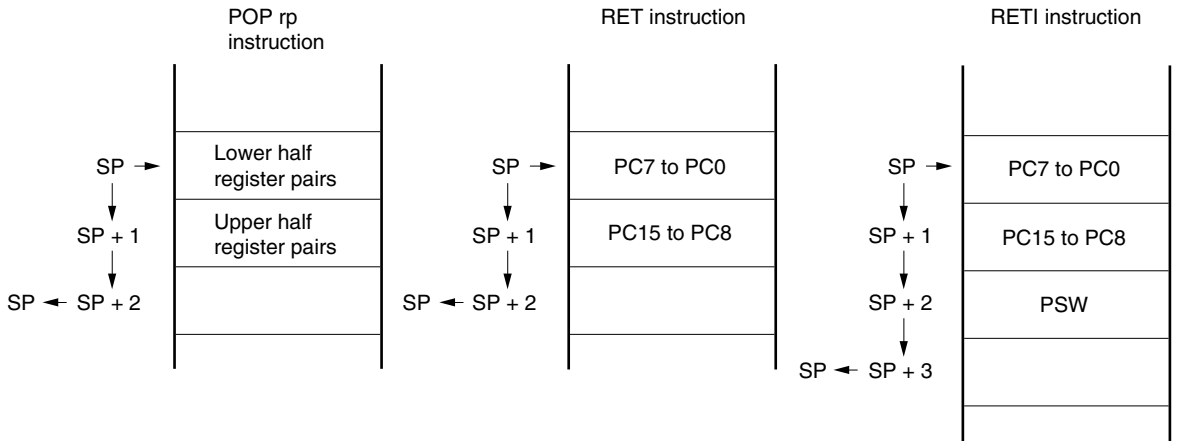


Figure 3-11. Data to Be Restored from Stack Memory



3.2.2 General-purpose registers

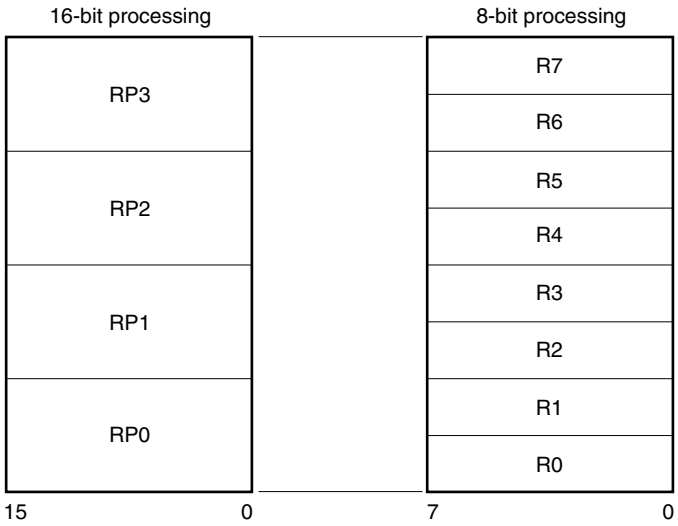
A general-purpose register consists of eight 8-bit registers (X, A, C, B, E, D, L, and H).

In addition each register being 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).

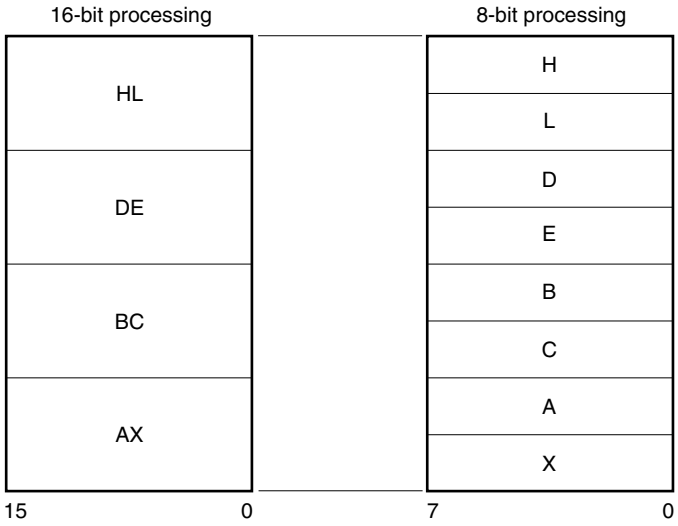
Registers can be described in terms of function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

Figure 3-12. General-Purpose Register Configuration

(a) Absolute names



(b) Function names



3.2.3 Special function registers (SFRs)

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

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

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

Each manipulation bit unit can be specified as follows.

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

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

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

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

- Symbol
  Indicates the addresses of the implemented special function registers. The symbols are reserved for the assembler and are defined as an sfr variable by the #pragma sfr directive for the C compiler. Therefore, these symbols can be used as instruction operands if an assembler or integrated debugger is used.

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

- Number of bits manipulated simultaneously
  Indicates the bit units (1, 8, and 16) in which the special function register 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	Number of Bits Manipulated Simultaneously			After Reset
				1 Bit	8 Bits	16 Bits	
FF00H	Port register 0	P0	R/W	√	√	–	00H
FF02H	Port register 2	P2		√	√	–	
FF04H	Port register 4	P4		√	√	–	
FF16H	16-bit compare register 20	CR20Note 1	W	–	√Note 2	√Note 3	FFFFH
FF17H							
FF18H	16-bit timer counter 20	TM20Note 1	R	–	√Note 2	√Note 3	0000H
FF19H							
FF1AH	16-bit capture register 20	TCP20Note 1		–	√Note 2	√Note 3	Undefined
FF1BH							
FF20H	Port mode register 0	PM0	R/W	√	√	–	FFH
FF22H	Port mode register 2	PM2		√	√	–	
FF24H	Port mode register 4	PM4		√	√	–	
FF30H	Pull-up resistor option register B0	PUB0		√	√	–	00H
FF34H	Pull-up resistor option register B4	PUB4		√	√	–	
FF42H	Timer clock selection register 2	TCL2		–	√	–	
FF48H	16-bit timer mode control register 20	TMC20		√	√	–	
FF60H	8-bit compare register 80	CR80	W	–	√	–	Undefined
FF61H	8-bit timer counter 80	TM80	R	–	√	–	00H
FF62H	8-bit timer mode control register 80	TMC80	R/W	√	√	–	
FF63H	8-bit compare register 50	CR50	W	–	√	–	Undefined
FF64H	8-bit timer counter 50	TM50	R	–	√	–	00H
FF65H	8-bit timer mode control register 50	TMC50	R/W	√	√	–	
FF66H	8-bit compare register 60	CR60	W	–	√	–	Undefined
FF67H	8-bit H width compare register 60	CRH60		–	√	–	
FF68H	8-bit timer counter 60	TM60	R	–	√	–	00H
FF69H	8-bit timer mode control register 60	TMC60	R/W	√	√	–	
FF6AH	Carrier generator output control register 60	TCA60		√	√	–	
FF6BH	REM signal control register	RSCR0		√	√	–	
FF6CH	Clock multiplication control register	CMC0		√	√	–	
FF6DH	TM50 source clock control register	ADSC5		√	√	–	

Notes 1. These SFRs are for 16-bit access only.

2. CR20, TM20, and TCP20 are designed only for 16-bit access. In direct addressing, however, 8-bit access can also be performed.

3. 16-bit access is allowed only in short direct addressing.

Table 3-4. Special Function Registers (2/2)

Address	Special Function Register (SFR) Name	Symbol	R/W	Number of Bits Manipulated Simultaneously			After Reset
				1 Bit	8 Bits	16 Bits	
FF70H	Asynchronous serial interface mode register 20	ASIM20	R/W	√	√	–	00H
FF71H	Asynchronous serial interface status register 20	ASIS20	R	√	√	–	
FF72H	Serial operation mode register 20	CSIM20	R/W	√	√	–	
FF73H	Baud rate generator control register 20	BRGC20		–	√	–	
FF74H	Transmission shift register 20	TXS20 / SIO20	W	–	√	–	FFH
	Receive buffer register 20	RXB20	R	–	√	–	Undefined
FFDDH	Power-on-clear register 10	POCF10	R/W	√	√	–	Retained[Note 1]
FFDEH	Low-voltage detection register 10	LVIF10		√	√	–	Retained[Note 2]
FFE0H	Interrupt request flag register 0	IF0		√	√	–	00H
FFE1H	Interrupt request flag register 1	IF1		√	√	–	
FFE4H	Interrupt mask flag register 0	MK0		√	√	–	FFH
FFE5H	Interrupt mask flag register 1	MK1		√	√	–	
FFECH	External interrupt mode register 0	INTM0		–	√	–	00H
FFF5H	Key return mode register 0	KRM0		√	√	–	
FFF9H	Watchdog timer mode register	WDTM		√	√	–	
FFFAH	Oscillation stabilization time selection register	OSTS		–	√	–	04H
FFFBH	Processor clock control register	PCC		√	√	–	02H

Notes 1. This value becomes 04H only after reset by power-on clear (refer to **Figure 11-2 Format of Power-on-Clear Register 10**.)

2. This value becomes 04H when V_{DD} < LVI detection voltage (refer to **Figure 12-2 Format of Low-Voltage Detection Register 10**.)

3.3 Instruction Address Addressing

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

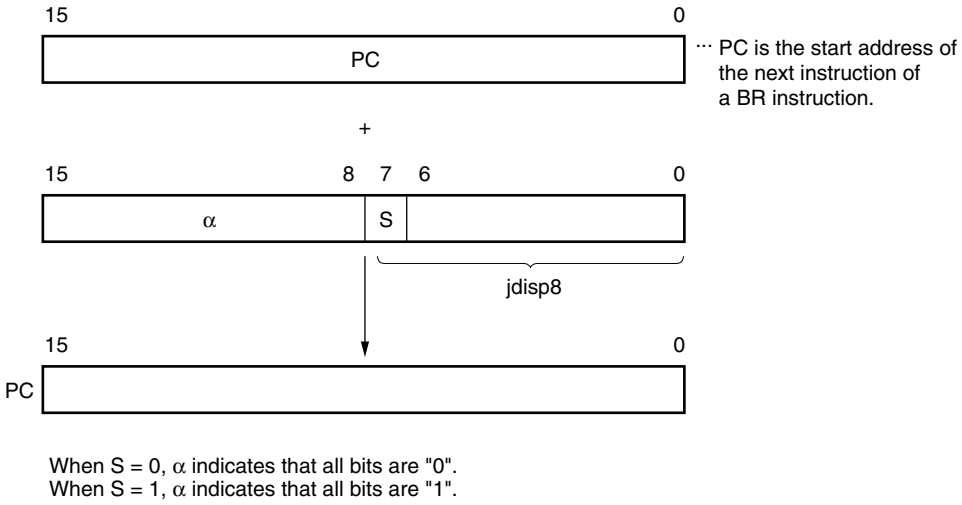
3.3.1 Relative addressing

[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) to branch. The displacement value is treated as signed two's complement data (–128 to +127) and bit 7 becomes the 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 that all bits are "0".
When S = 1, α indicates that all bits are "1".

3.3.2 Immediate addressing

[Function]

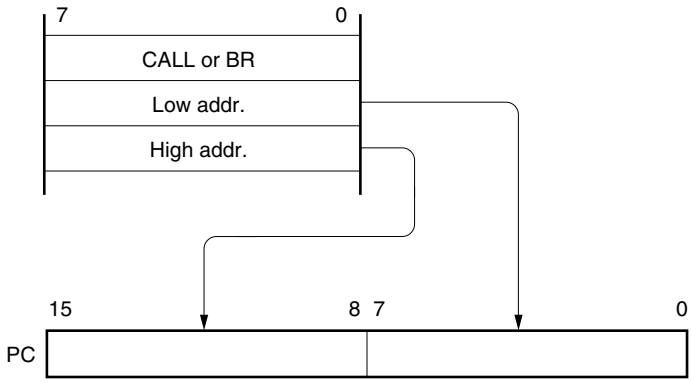
Immediate data in the instruction word is transferred to the program counter (PC) to branch.

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

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

[Illustration]

In case of CALL !addr16 and BR !addr16 instructions



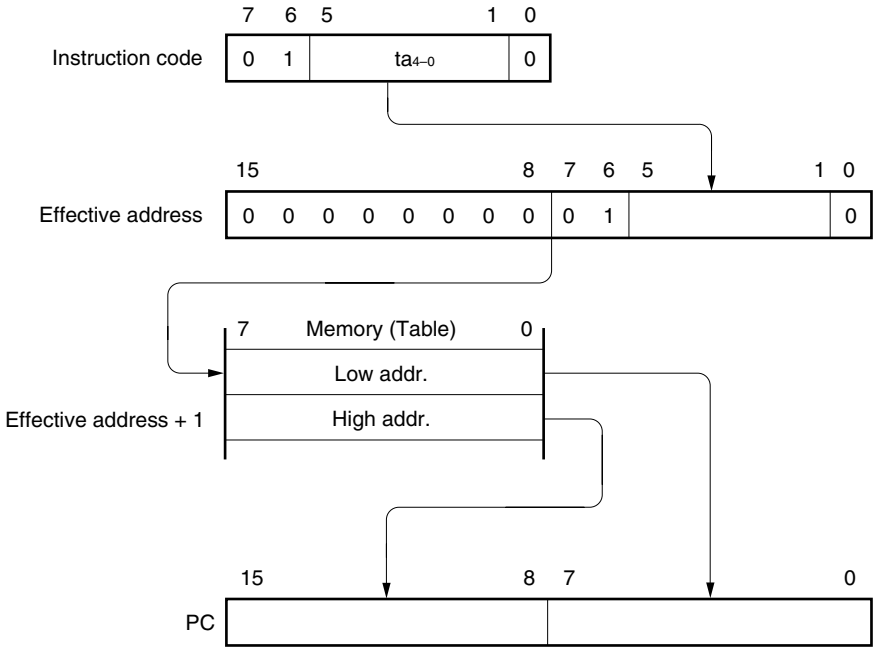
3.3.3 Table indirect addressing

[Function]

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

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

[Illustration]



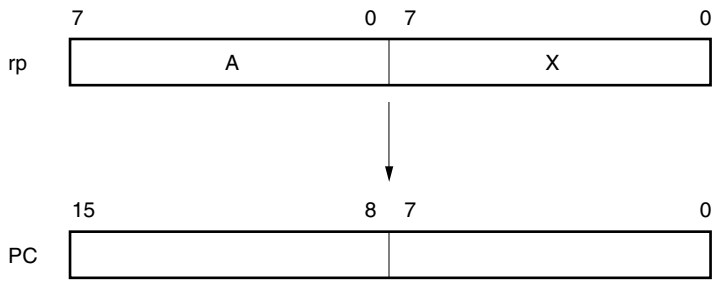
3.3.4 Register addressing

[Function]

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

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

[Illustration]



3.4 Operand Address Addressing

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

3.4.1 Direct addressing

[Function]

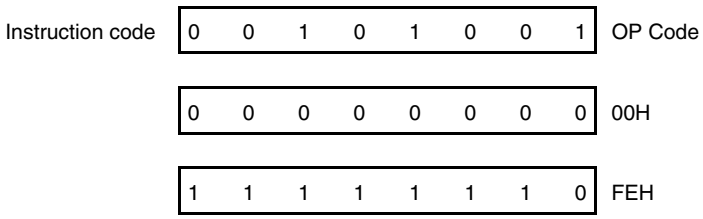
The memory indicated by immediate data in an instruction word is directly addressed.

[Operand format]

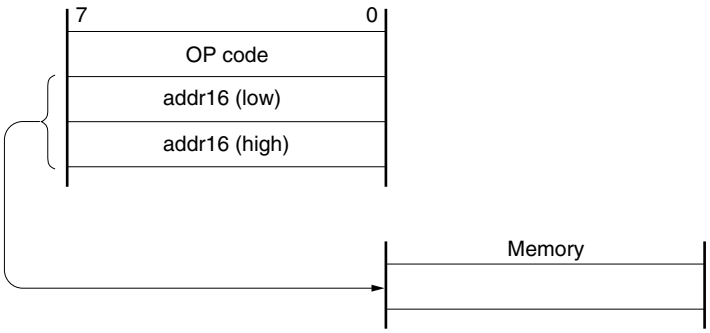
Identifier	Description
addr16	Label or 16-bit immediate data

[Description example]

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



[Illustration]



3.4.2 Short direct addressing

[Function]

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

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

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

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

[Operand format]

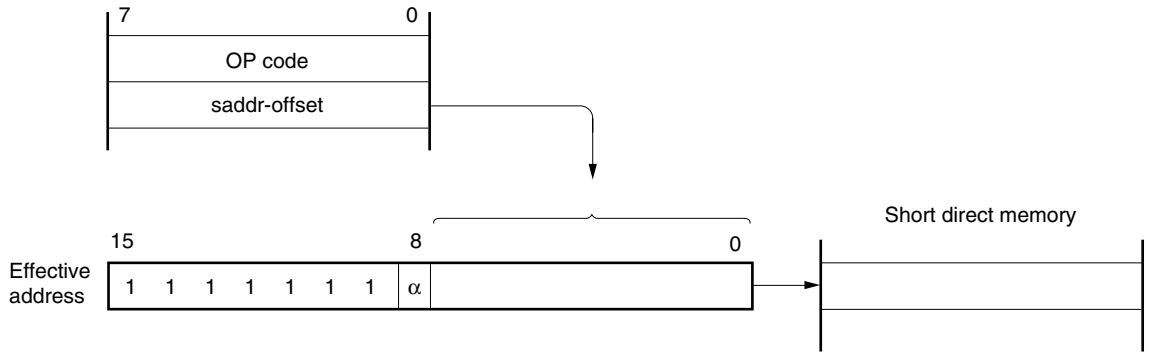
Identifier	Description
saddr	Label or immediate data indicating FE20H to FF1FH
saddrp	Label or immediate data indicating FE20H to FF1FH (even address only)

★ **[Description example]**

MOV FE90H, A; When transferring register A value to saddr (FE90H)

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

[Illustration]



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

3.4.3 Special function register (SFR) addressing

[Function]

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

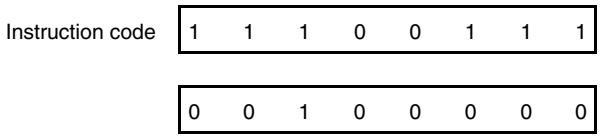
This addressing is applied to the 256-byte spaces FF00H to FFFFH and FFE0H to FFFFH. However, SFRs mapped at FF00H to FF1FH can also be accessed with short direct addressing.

[Operand format]

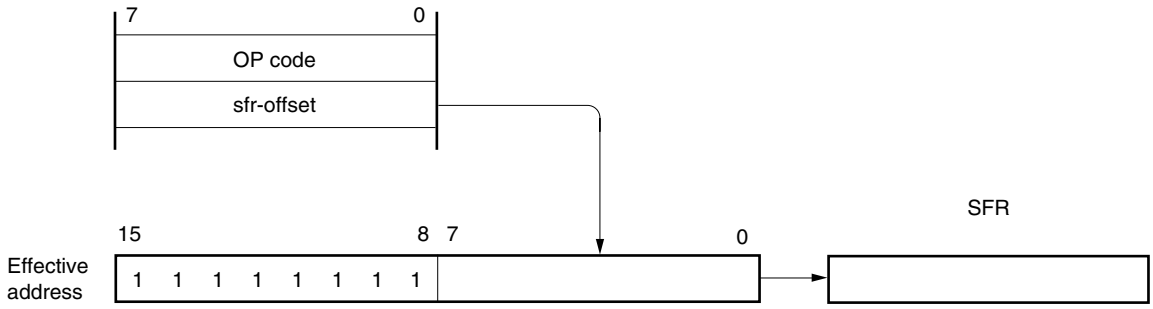
Identifier	Description
sfr	Special function register name

[Description example]

MOV PM0, A; When selecting PM0 for sfr



[Illustration]



3.4.4 Register addressing

[Function]

A general-purpose register is accessed as an operand.

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

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

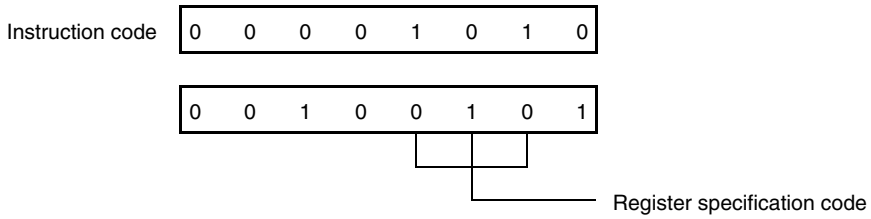
[Operand format]

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

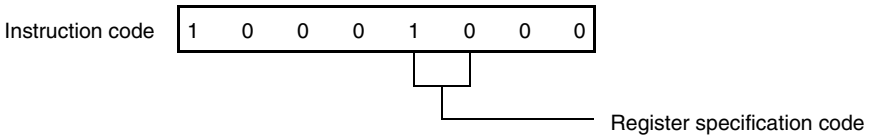
'r' and 'rp' can be described with absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

[Description example]

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



INCW DE; When selecting the DE register pair for rp



3.4.5 Register indirect addressing

[Function]

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

[Operand format]

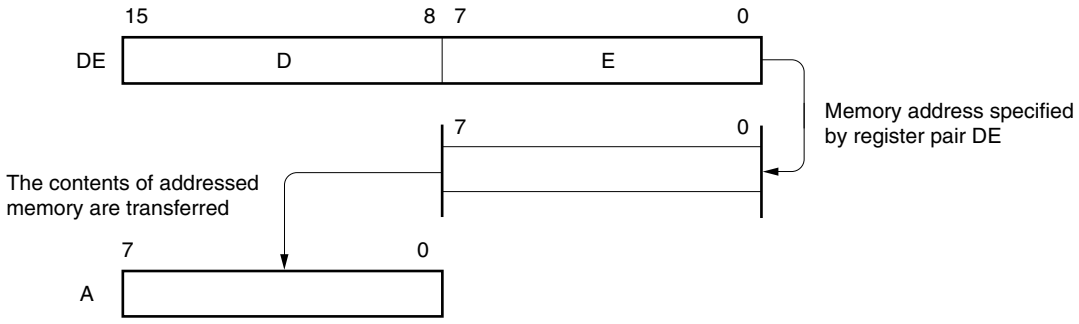
Identifier	Description
–	[DE], [HL]

[Description example]

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

Instruction code	0	0	1	0	1	0	1	1

[Illustration]



3.4.6 Based addressing

[Function]

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

[Operand format]

Identifier	Description
–	[HL+byte]

[Description example]

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

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

3.4.7 Stack addressing

[Function]

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

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

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

[Description example]

In the case of PUSH DE

Instruction code	1	0	1	0	1	0	1	0

CHAPTER 4 PORT FUNCTIONS

4.1 Function of Port

The μPD789088 Subseries is provided with the ports shown in Figure 4-1. These ports enable several types of control. Table 4-1 lists the functions of each port.

These ports, while originally designed as digital input/output ports, have alternate functions. For the alternate functions, refer to **2.1 Pin Function List**.

Figure 4-1. Port Types

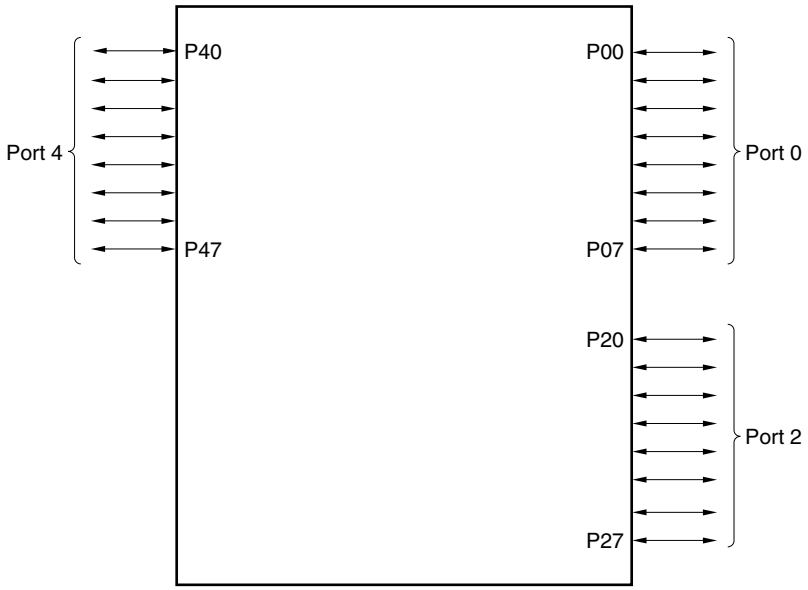


Table 4-1. Port Functions

Port Name	Pin Name	Function
Port 0	P00 to P07	I/O port. Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register B0 (PUB0).
Port 2	P20 to P27	I/O port. Input/output can be specified in 1-bit units.
Port 4	P40 to P47	I/O port. Input/output can be specified in 1-bit units. When used as an input port, an on-chip pull-up resistor can be specified by means of pull-up resistor option register B4 (PUB4).

4.2 Configuration of Port

Ports include the following hardware.

Table 4-2. Configuration of Port

Parameter	Configuration
Control registers	Port mode registers (PM0, PM2, PM4) Pull-up resistor option registers (PUB0, PUB4)
Ports	CMOS I/O: 24
Pull-up resistors	Software control: 16

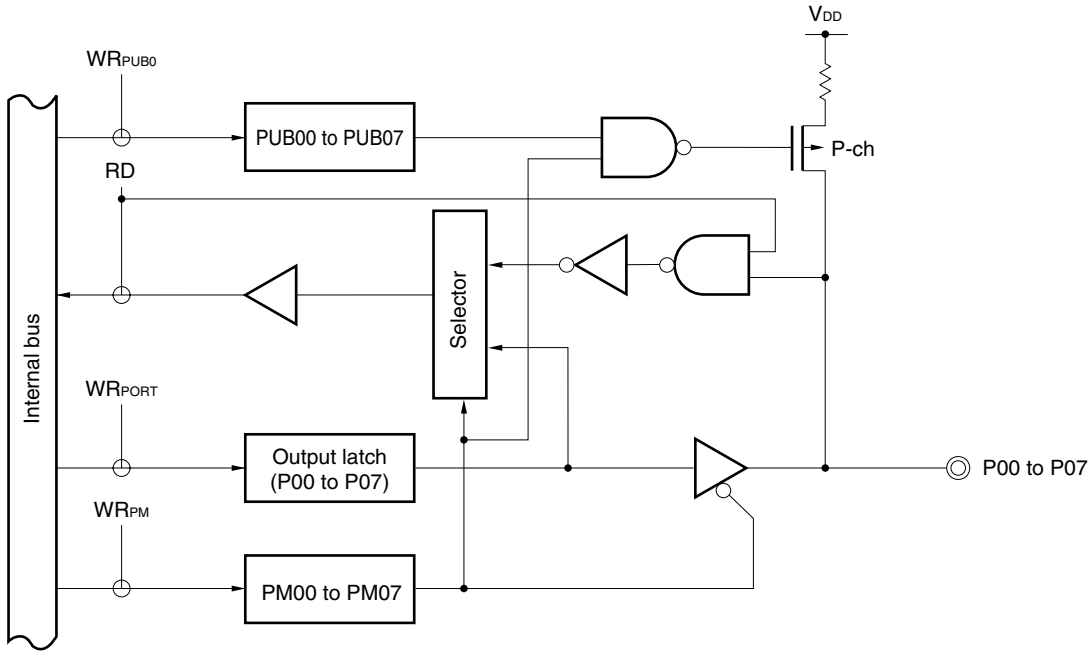
4.2.1 Port 0

This is an 8-bit I/O port with an output latch. Port 0 can be set to input or output mode in 1-bit units by using port mode register 0 (PM0). On-chip pull-up resistors can be connected to the pins used as an input port in 1-bit units by using pull-up resistor option register B0 (PUB0).

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

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

Figure 4-2. Block Diagram of P00 to P07



PUB0: Pull-up resistor option register B0
PM: Port mode register
RD: Port 0 read signal
WR: Port 0 write signal

4.2.2 Port 2

This is an 8-bit I/O port with output latches. Port 2 can be set to input or output mode in 1-bit units by using port mode register 2 (PM2).

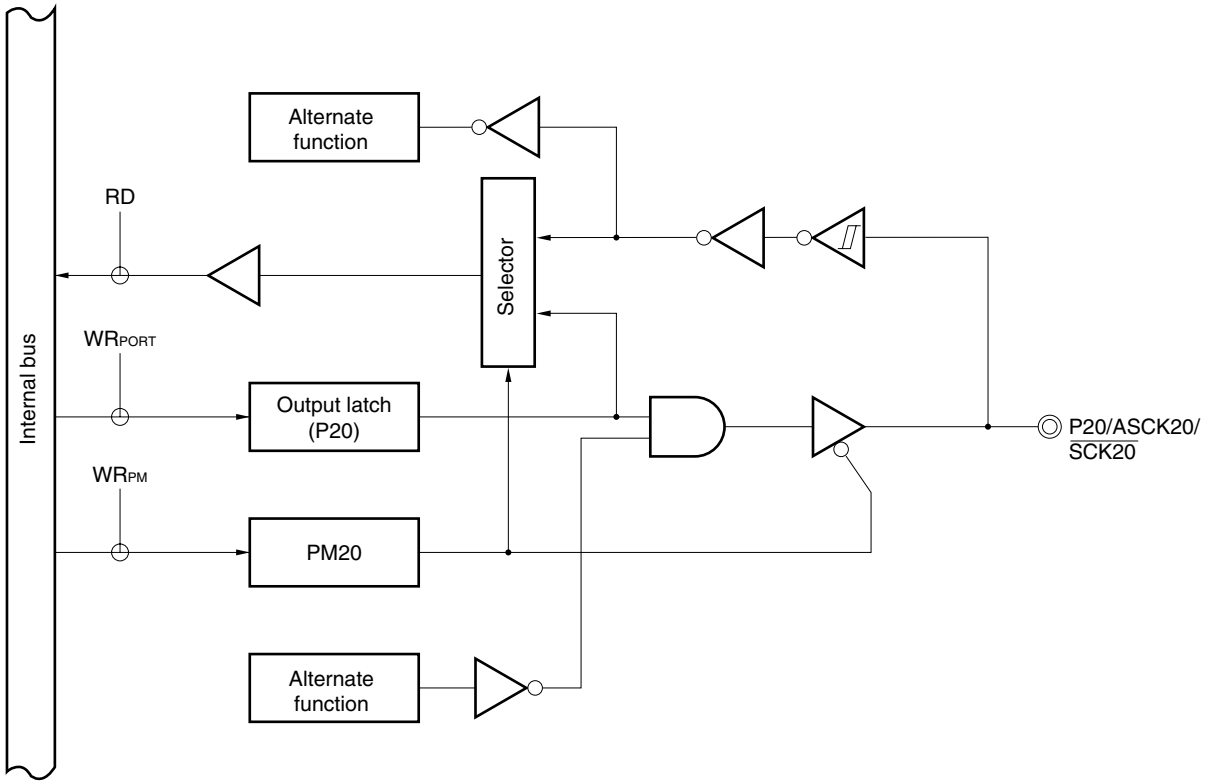
The port is also used as serial interface I/O and timer I/O.

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

Figures 4-3 through 4-7 show block diagrams of port 2.

Caution When using the pins of port 2 for the serial interface, the I/O and output latches must be set according to the function to be used. For details of the settings, see Table 10-2 Serial Interface 20 Operation Mode Settings.

Figure 4-3. Block Diagram of P20

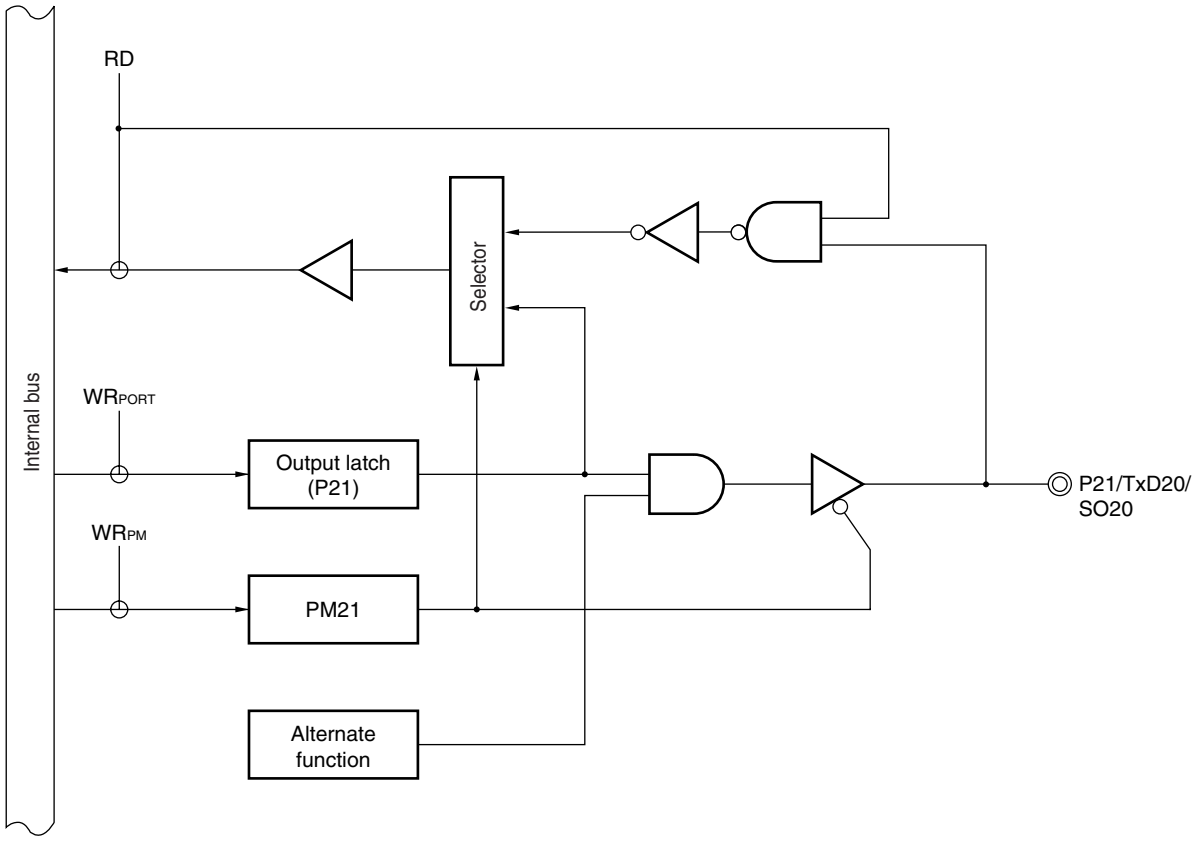


PM: Port mode register

RD: Port 2 read signal

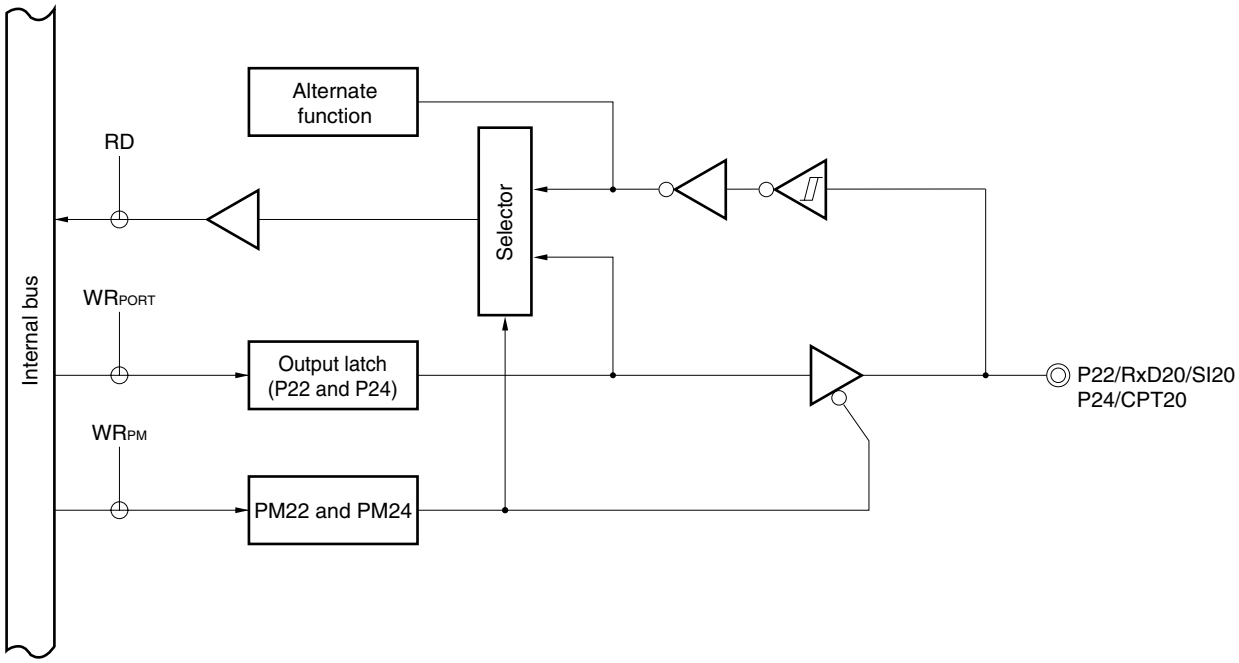
WR: Port 2 write signal

Figure 4-4. Block Diagram of P21



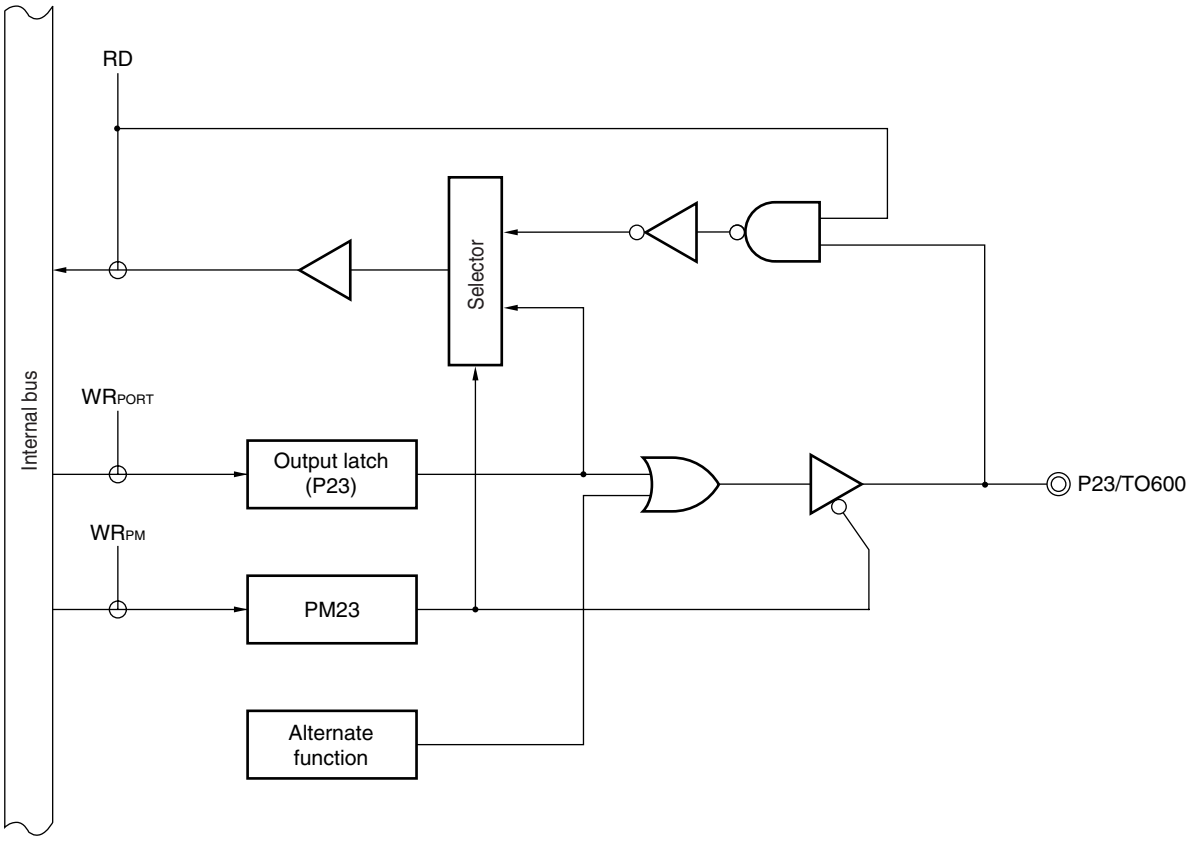
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

Figure 4-5. Block Diagram of P22 and P24



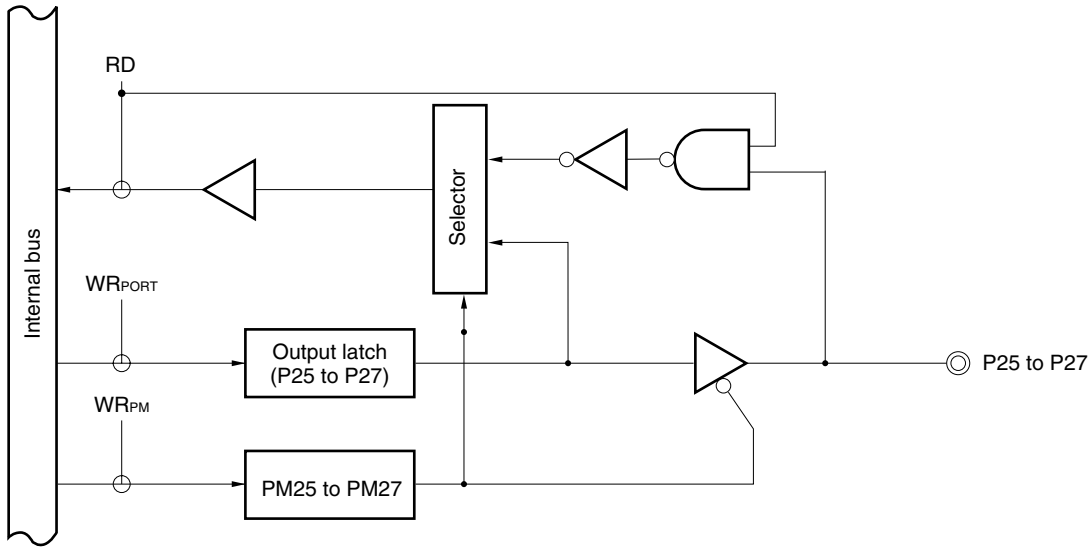
PM: Port mode register
RD: Port 2 read signal
WR: Port 2 write signal

Figure 4-6. Block Diagram of P23



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

Figure 4-7. Block Diagram of P25 to P27



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

4.2.3 Port 4

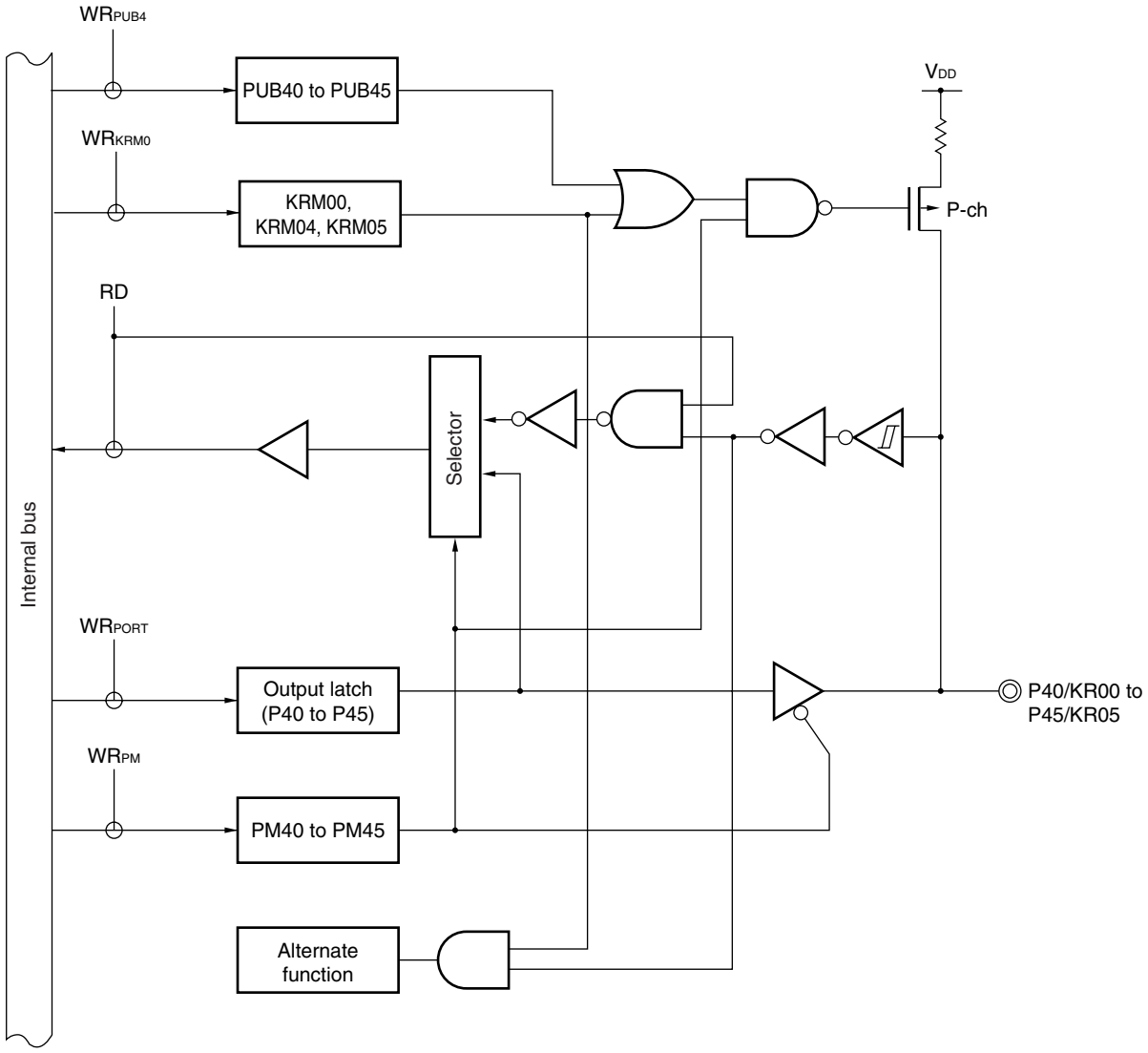
This is an 8-bit I/O port with an output latch. Port 4 can be set to input or output mode in 1-bit units by using port mode register 4 (PM4). On-chip pull-up resistors can be connected to the pins used as an input port in 1-bit units by using pull-up resistor option register B4 (PUB4).

The port is also used as key return signal detection and external interrupt input.

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

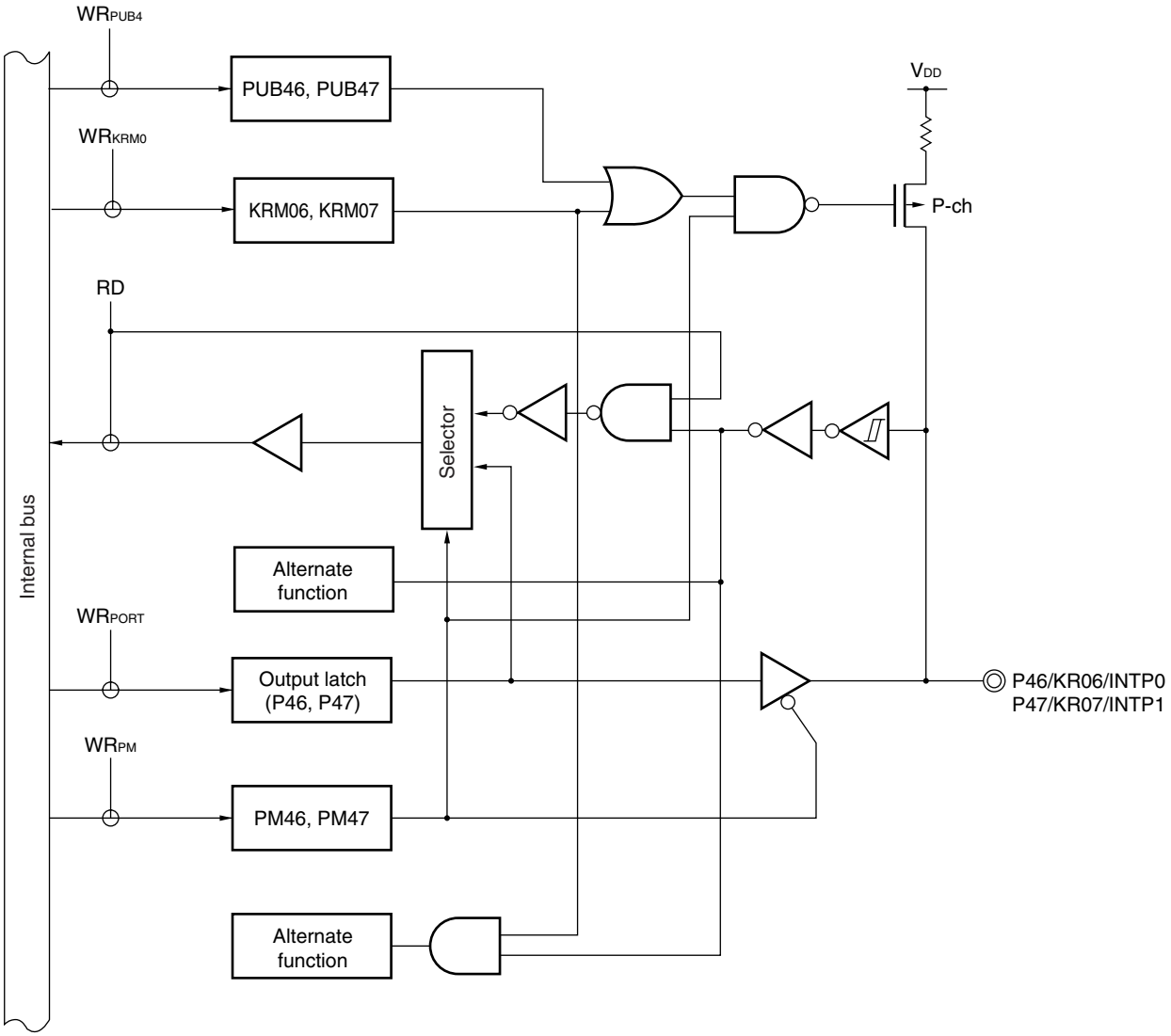
Figures 4-8 through 4-9 show block diagrams of port 4.

Figure 4-8. Block Diagram of P40 to P45



PUB4: Pull-up resistor option register B4
KRM0: Key return mode register 0
PM: Port mode register
RD: Port 4 read signal
WR: Port 4 write signal

Figure 4-9. Block Diagram of P46 and P47



PUB4: Pull-up resistor option register B4
KRM0: Key return mode register 0
PM: Port mode register
RD: Port 4 read signal
WR: Port 4 write signal

4.3 Registers Controlling Port Function

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

- Port mode registers (PM0, PM2, PM4)
- Pull-up resistor option registers (PUB0 and PUB4)

(1) Port mode registers (PM0, PM2, PM4)

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

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

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

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

Caution When port 4 is acting as an output port, and its output level is changed, an interrupt request flag is set, because this port is also used as the input for an external interrupt. To use port 2 in output mode, therefore, the interrupt mask flag must be set to 1 in advance.

Figure 4-10. Format of Port Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	FF24H	FFH	R/W

PMmn	Pmn pin input/output mode selection (m = 0, 2, 4; n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

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

Pin Name	Alternate Function		PM××	P××
	Name	Input/Output		
P23	TO600	Output	0	0
P24	CPT20	Input	1	×
P40 to P45	KR00 to KR05	Input	1	×
P46	KR06	Input	1	×
	INTP0	Input	1	×
P47	KR07	Input	1	×
	INTP1	Input	1	×

Caution When using the pins of port 2 for the serial interface, the I/O or output latch must be set according to the function to be used. For details of the settings, see Table 10-2 Serial Interface 20 Operation Mode Settings.

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

(2) Pull-up resistor option registers (PUB0, PUB4)

These registers specify whether an on-chip pull-up resistor connected to each pin of ports 0 and 4 is used. An on-chip pull-up resistor can be connected only to the pins for which use of an on-chip pull-up resistor is specified and to the bits set to input mode by PUB0 and PUB4. An on-chip pull-up resistor is automatically disconnected from the pins set to output mode regardless of the setting of PUB0 and PUB4.

PUB0 and PUB4 are set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears PUB0 and PUB4 to 00H.

Figure 4-11. Format of Pull-up Resistor Option Register

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
PUB0	PUB07	PUB06	PUB05	PUB04	PUB03	PUB02	PUB01	PUB00	FF30H	00H	R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
PUB4	PUB47	PUB46	PUB45	PUB44	PUB43	PUB42	PUB41	PUB40	FF34H	00H	R/W

PUBmn	Pmn on-chip pull-up resistor selection (m = 0 or 4; n = 0 to 7)
0	On-chip pull-up resistor is not used.
1	On-chip pull-up resistor is used.

4.4 Operation of Port Functions

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

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

4.4.2 Reading from I/O port

(1) In output mode

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

(2) In input mode

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

4.4.3 Arithmetic operation of I/O port

(1) In output mode

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

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

(2) In input mode

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

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

CHAPTER 5 CLOCK GENERATOR

5.1 Function of Clock Generator

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

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

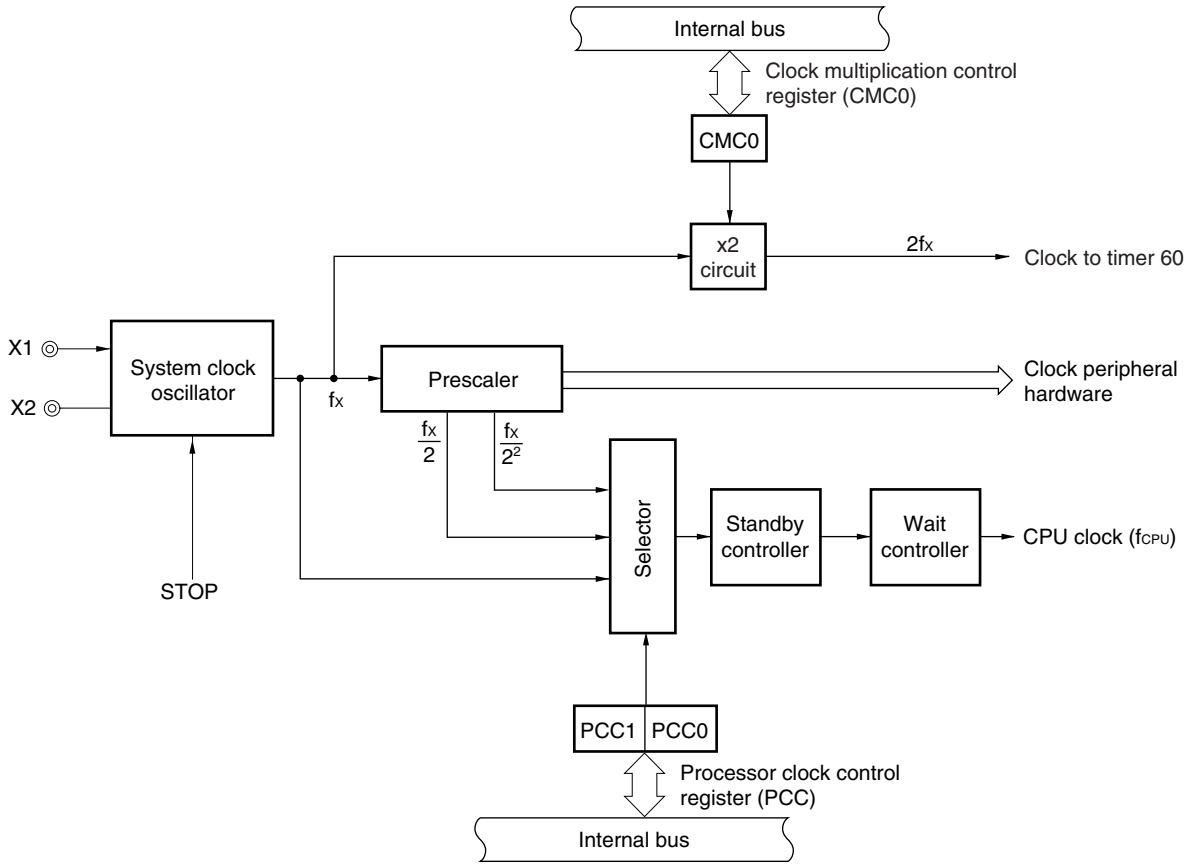
5.2 Configuration of Clock Generator

The clock generator includes the following hardware.

Table 5-1. Configuration of Clock Generator

Item	Configuration
Control registers	Processor clock control register (PCC) Clock multiplication control register (CMC0)
Oscillator	Crystal/ceramic oscillator

Figure 5-1. Block Diagram of Clock Generator



5.3 Registers Controlling Clock Generator

The clock generator is controlled by the following two registers.

- Processor clock control register (PCC)
- Clock multiplication control register (CMC0)

(1) Processor clock control register (PCC)

PCC selects the CPU clock and the division ratio.

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

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

Figure 5-2. Format of Processor Clock Control Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PCC	0	0	0	0	0	0	PCC1	PCC0	FFFBH	02H	R/W

PCC1	PCC0	CPU clock (f_{CPU}) selection	Minimum instruction execution time: $2/f_{CPU}$
			Operation at f_X = 5.0 MHz
0	0	f_X	0.4 μs
0	1	$f_X/2$	0.8 μs
1	0	$f_X/2^2$	1.6 μs
1	1	Setting prohibited	

Caution Bits 2 to 7 must all be set to 0.

Remark f_X: System clock oscillation frequency

(2) Clock multiplication control register (CMC0)

CMC0 controls the operation of the clock multiplication circuit.

CMC0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets CMC0 to 00H.

Figure 5-3. Format of Clock Multiplication Control Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
CMC0	0	0	0	0	0	0	0	CMC0	FF6CH	00H	R/W

CMC0	Control of system clock x2 multiplication circuit
0	Operation stopped (supply of x2 system clock ($2f_X$) to timer 60 is disabled)
1	Operation enabled (x2 system clock ($2f_X$) is supplied to timer 60)

Cautions 1. Bits 1 to 7 must all be set to 0.

2. The operation voltage range when the clock multiplication circuit is used is 2.0 to 3.6 V.

5.4 System Clock Oscillators

5.4.1 System clock oscillator

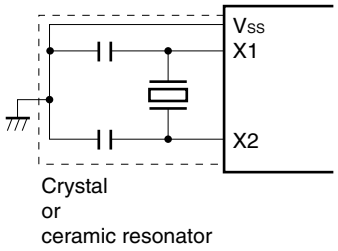
The system clock oscillator is oscillated by the crystal or ceramic resonator (5.0 MHz TYP.) connected across the X1 and X2 pins.

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

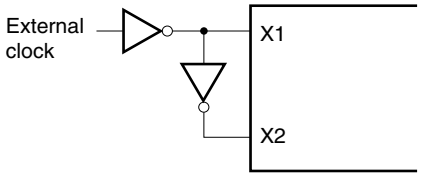
Figure 5-4 shows the external circuit of the system clock oscillator.

Figure 5-4. External Circuit of System Clock Oscillator

(a) Crystal or ceramic oscillation



(b) External clock



Cautions 1. When using the system clock oscillator, wire as follows in the area enclosed by the broken lines in Figure 5-4 to avoid an adverse effect from wiring capacitance.

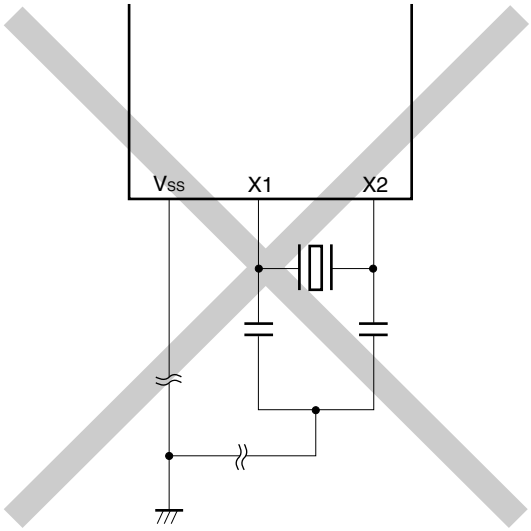
- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS}. Do not ground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

5.4.2 Example of incorrect resonator connection

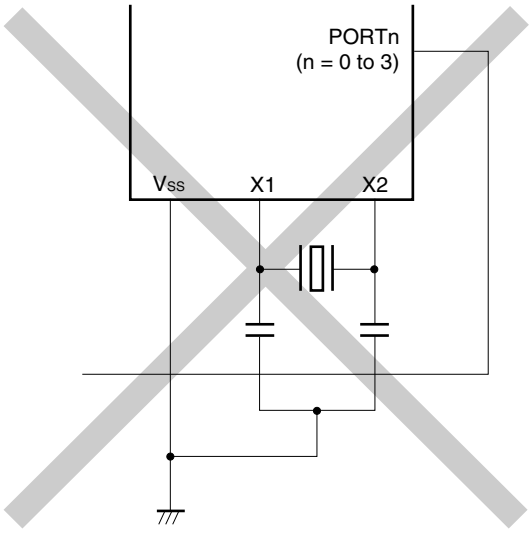
Figure 5-5 shows an example of incorrect resonator connections.

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

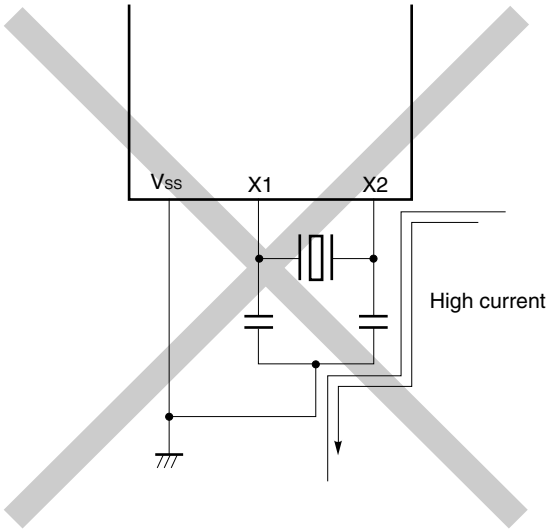
(a) Wiring too long



(b) Crossed signal line



(c) Wiring near high fluctuating current



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

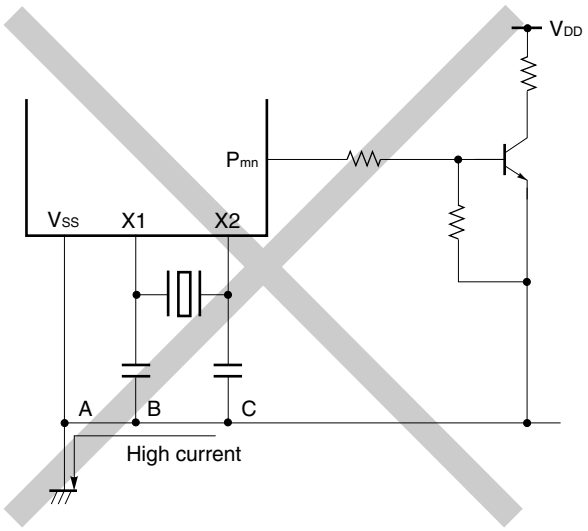
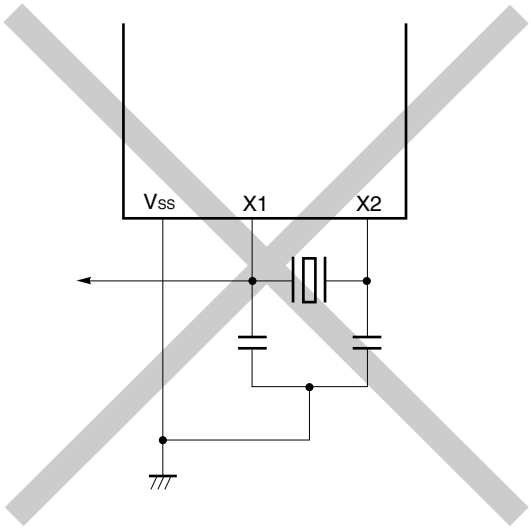


Figure 5-5. Example of Incorrect Resonator Connection (2/2)

(e) Signal is fetched



5.4.3 Frequency divider

The frequency divider divides the system clock oscillator output (fx) and generates clocks.

5.5 Operation of Clock Generator

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

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

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

(a) The slow mode (1.6 μs @ 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) Three types of minimum instruction execution time (f_{CPU}) (0.4 μs, 0.8 μs, 1.6 μs @ 5.0 MHz operation) can be selected by the PCC setting.

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

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

5.6 Changing Setting of CPU Clock

5.6.1 Time required for switching CPU clock

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

The time indicated in Table 5-2 (max.) is required until the CPU clock actually switches (i.e. switching does not occur immediately after the PCC register is rewritten). Until this time has elapsed, therefore, it is impossible to ascertain whether the clock before or after the switch is operating.

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

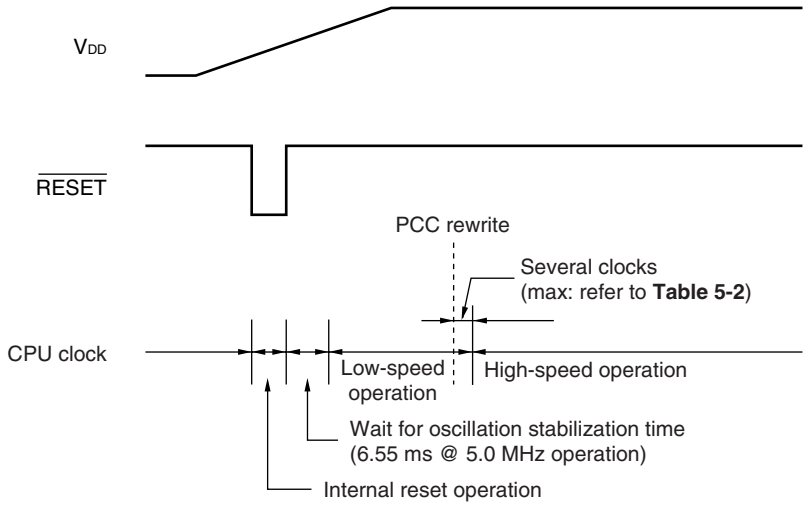
Set Value Before Switching		Set Value After Switching					
PCC1	PCC0	PCC1	PCC0	PCC1	PCC0	PCC1	PCC0
		0	0	0	1	1	0
0	0			16 clocks		16 clocks	
	1	8 clocks				8 clocks	
1	0	4 clocks		4 clocks			
	1	Setting prohibited					

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

5.6.2 Examples of switching CPU clock

The following figure illustrates how the CPU clock is switched.

Figure 5-6. Examples of Switching Between System Clock and CPU Clock



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

After that, the CPU starts instruction execution at the slow speed of the system clock (1.6 μs @ 5.0 MHz operation).

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

<3> A few clock later, the CPU clock is switched to high-speed operation (0.4 μs @ 5.0 MHz operation) and the fastest operation is started.

CHAPTER 6 16-BIT TIMER 20

The 16-bit timer counter references the free running counter and provides the functions such as timer interrupt. In addition, the count value can be captured by a trigger pin.

6.1 Function of 16-Bit Timer 20

16-bit timer 20 has the following functions.

- Timer interrupt
- Count value capture

(1) Timer interrupt

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

(2) Count value capture

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

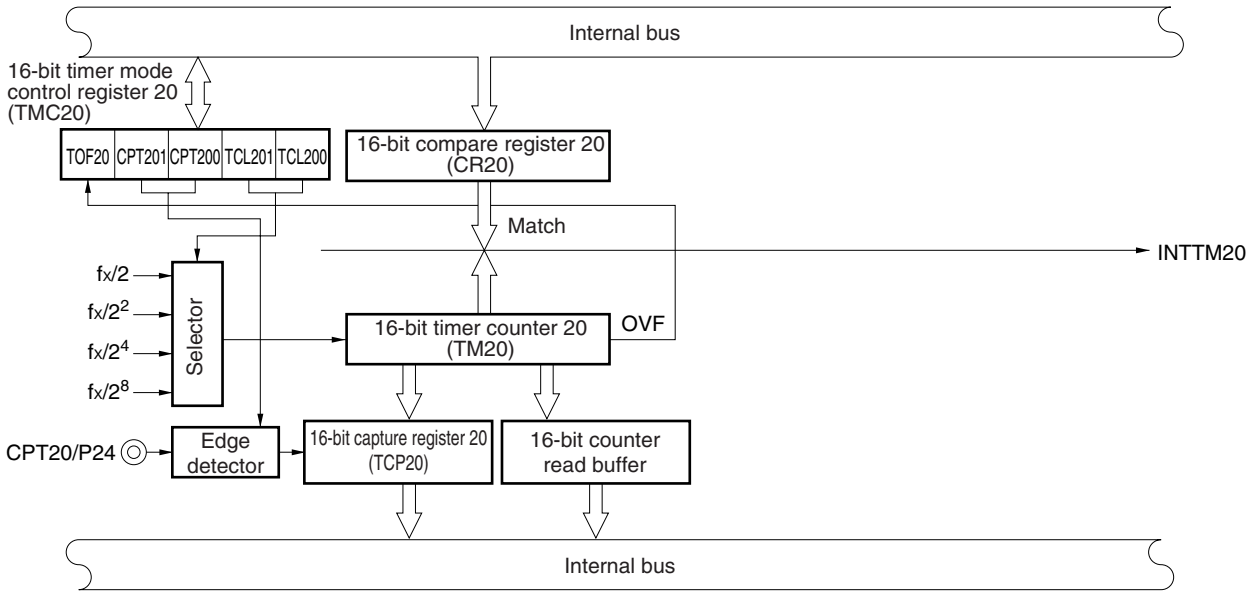
6.2 Configuration of 16-Bit Timer 20

16-bit timer 20 includes the following hardware.

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

Item	Configuration
Timer counter	16 bits × 1 (TM20)
Registers	Compare register: 16 bits × 1 (CR20) Capture register: 16 bits × 1 (TCP20)
Timer output	None
Control registers	16-bit timer mode control register 20 (TMC20) Port mode register 2 (PM2) Port register 2 (P2)

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 (INTTM20).

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

$\overline{\text{RESET}}$ input sets CR20 to FFFFH.

Cautions 1. Although this register is manipulated with a 16-bit memory manipulation instruction, an 8-bit memory manipulation instruction can be used. When manipulated with an 8-bit memory manipulation instruction, the accessing method should be direct addressing.

2. When rewriting CR20 during count operation, set CR20 to interrupt disable from interrupt mask flag register 0 (MK10) beforehand. Also, set the timer output data to inversion disable using 16-bit timer mode control register 20 (TMC20).

When CR20 is rewritten in the interrupt-enabled state, an interrupt request may occur at the moment of rewrite.

(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 free running during count clock input.

$\overline{\text{RESET}}$ input sets TM20 to 0000H and after which it resumes free running.

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

2. Although this register is manipulated with a 16-bit memory manipulation instruction, an 8-bit memory manipulation instruction can be used. When manipulated with an 8-bit memory manipulation instruction, the accessing method should be direct addressing.

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

4. The TM20 read value is not guaranteed when $f_X/2^2$ is selected for the CPU clock and $f_X/2$ is selected for the count clock of 16-bit timer 20.

(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 TCP20 to undefined.

Caution Although this register is manipulated with a 16-bit memory manipulation instruction, an 8-bit memory manipulation instruction can be used. When manipulated with an 8-bit memory manipulation instruction, the accessing method should be direct addressing.

(4) 16-bit counter read buffer

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

6.3 Registers Controlling 16-Bit Timer 20

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

- 16-bit timer mode control register 20 (TMC20)
- Port mode register 2 (PM2)
- Port register 2 (P2)

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

16-bit timer mode control register 20 (TMC20) controls the setting of the counter clock, capture edge, etc.

TMC20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC20 to 00H.

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

Symbol	7	<6>	5	4	3	2	1	0	Address	After reset	R/W
TMC20	0	TOF20	CPT201	CPT200	0	TCL201	TCL200	0	FF48H	00H	R/W

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 CPT20
1	0	Falling edge of CPT20
1	1	Both edges of CPT20

TCL201	TCL200	16-bit timer counter 20 count clock selection
0	0	$f_X/2$ (2.5 MHz)
0	1	$f_X/2^2$ (1.25 MHz)
1	0	$f_X/2^4$ (312.5 kHz)
1	1	$f_X/2^8$ (19.5 kHz)

Cautions 1. Bits 0, 3, and 7 must all be set to 0.

2. The TM20 read value is not guaranteed when $f_X/2^2$ is selected for the CPU clock and $f_X/2$ is selected for the count clock of 16-bit timer 20.

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 2 (PM2)

This register sets the input/output of port 2 in 1-bit units.

To use the P24/CPT20 pin for timer input, set PM24 to 1.

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

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

Figure 6-3. Format of Port Mode Register 2

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W

PM24	P24 pin I/O mode selection
0	Output mode (output buffer on)
1	Input mode (output buffer off)

6.4 Operation of 16-Bit Timer 20

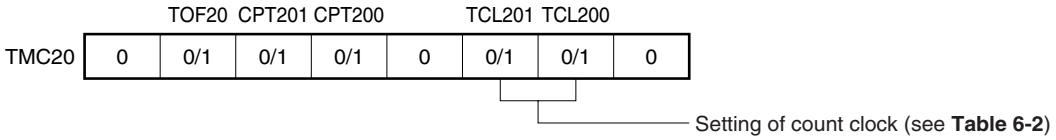
6.4.1 Operation as timer interrupt

16-bit timer 20 can generate interrupts repeatedly each time the free-running counter value reaches the value set to CR20. Since this counter is not cleared and holds the count even after an interrupt is generated, the interval time is equal to one cycle of the count clock set in TCL201 and TCL200.

To operate the 16-bit timer 20 as a timer interrupt, the following settings are required.

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

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



Caution If both the CPT201 and CPT200 flags are set to 0, the 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 (INTTM20) is generated.

Table 6-3 shows the interval time, and Figure 6-5 shows the timing of the timer interrupt operation.

Caution When rewriting CR20 during count operation, be sure set TMMK20 to interrupt disable (by setting bit 6 of interrupt mask flag register 0 (MK0) to 1).
When CR20 is rewritten in the interrupt-enabled state, an interrupt request may occur at the moment of rewrite.

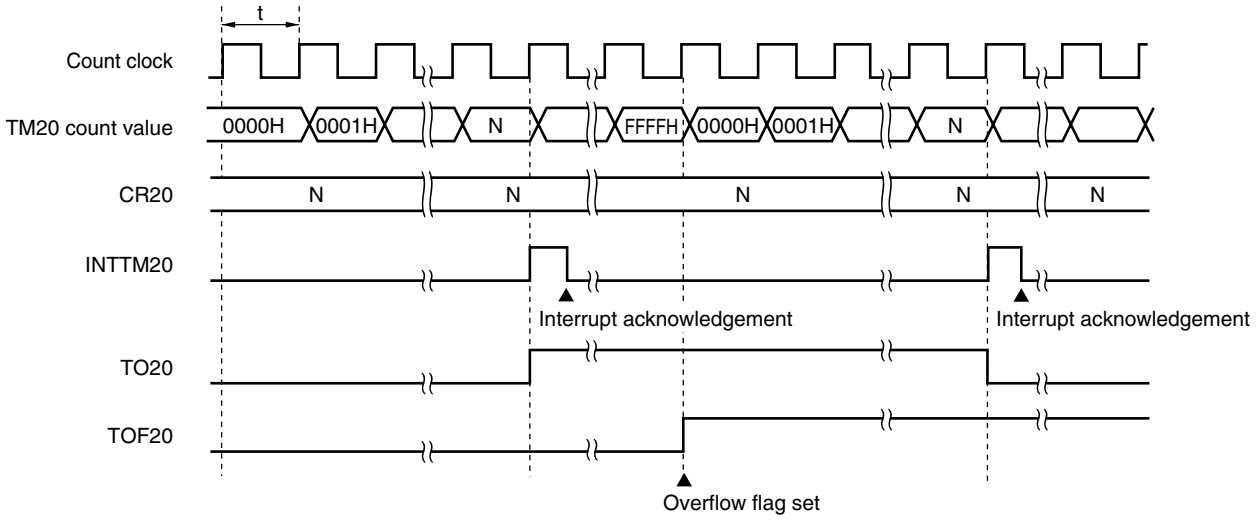
Table 6-2. Interval Time of 16-Bit Timer 20

TCL201	TCL200	Count Clock Cycle	Interval Time
0	0	2/fx (1.6 μs)	2^{17}/fx (26.2 ms)
0	1	2^{2}/fx (0.8 μs)	2^{18}/fx (52.4 ms)
1	0	2^{4}/fx (3.2 μs)	2^{20}/fx (209.7 ms)
1	1	2^{8}/fx (51.2 μs)	2^{24}/fx (3.36 s)

Remarks 1. fx: System clock oscillation frequency

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

Figure 6-5. Timing of Timer Interrupt Operation



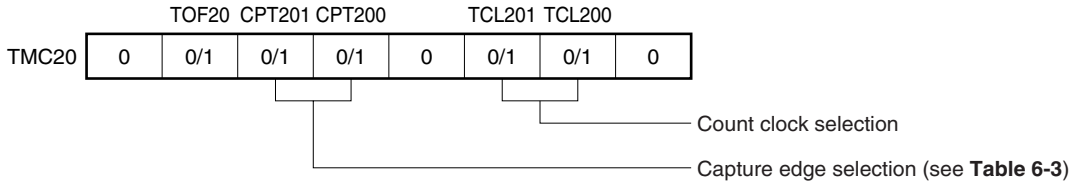
Remark N = 0000H to FFFFH

6.4.2 Capture operation

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

Set as shown in Figure 6-6 to allow 16-bit timer 20 to start the capture operation.

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



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

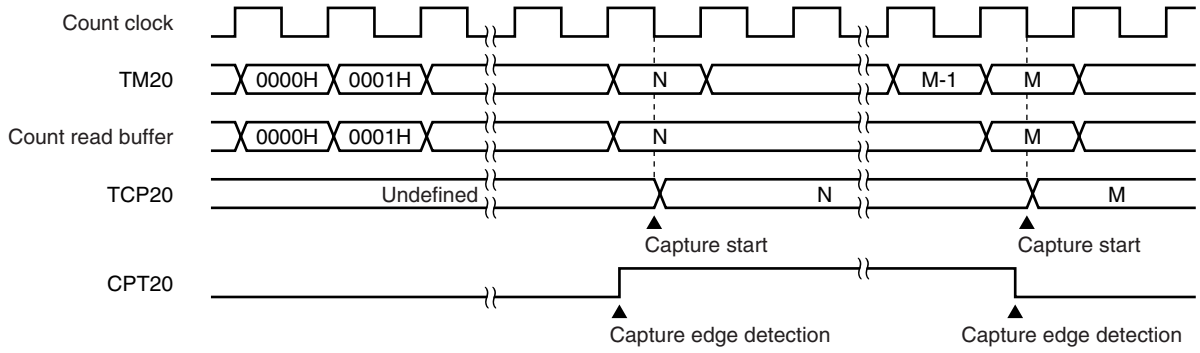
Table 6-3 and Figure 6-7 show the setting contents of the capture edge and capture operation timing, respectively.

Table 6-3. Settings of Capture Edge

CPT201	CPT200	Capture Edge Selection
0	0	Capture operation prohibited
0	1	CPT20 pin rising edge
1	0	CPT20 pin falling edge
1	1	CPT20 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 6-7. Capture Operation Timing (Both Edges of CPT20 Pin Are Specified)



Remark N, M = 0000H to FFFFH

6.4.3 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.

Buffer operation is then held pending 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 the pending state is performed at the CPU clock falling edge after the read signal of the TM20 higher byte falls.

$\overline{\text{RESET}}$ input sets TM20 to 0000H and restarts free running.

Figure 6-8 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 stabilization time.

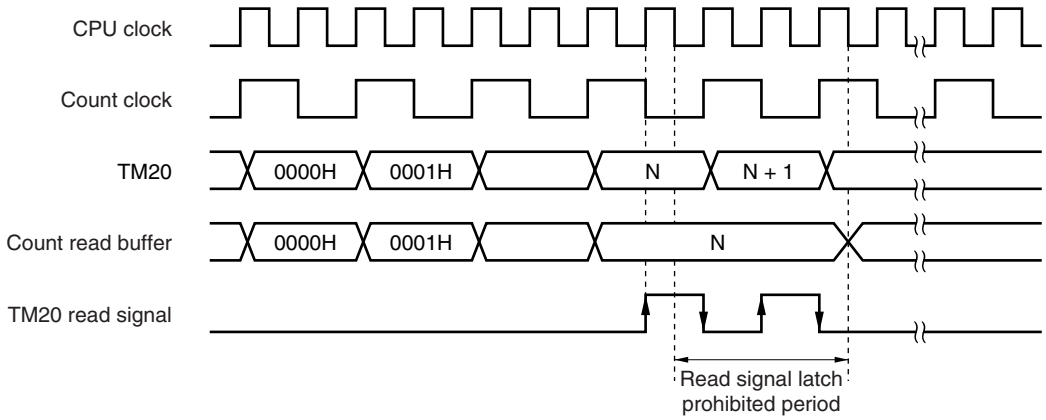
2. Although TM20 is a dedicated 16-bit transfer instruction register, an 8-bit transfer instruction can be used.

Execute an 8-bit transfer instruction by direct addressing.

3. When using an 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.

4. The TM20 read value is not guaranteed when $f_X/2^2$ is selected for the CPU clock and $f_X/2$ is selected for the count clock of 16-bit timer 20.

Figure 6-8. 16-Bit Timer Counter 20 Readout Timing



Remark N = 0000H to FFFFH

★ 6.5 Cautions on Using 16-Bit Timer 20

6.5.1 Restrictions when rewriting 16-bit compare register 20

(1) Disable interrupts (TMMK20 = 1) before rewriting the compare register (CR20).
If CR20 is rewritten with interrupts enabled, an interrupt request may be generated immediately.

(2) Depending on the timing of rewriting the compare register (CR20), the interval time may become twice as long as the intended time.
To avoid this problem, rewrite the compare register using either of the following procedures.

<Countermeasure A> When rewriting using 8-bit access

<1> Disable interrupts (TMMK20 = 1).
<2> First rewrite the higher 1 byte of CR20 (16 bits).
<3> Then rewrite the lower 1 byte of CR20 (16 bits).
<4> Clear the interrupt request flag (TMIF20).
<5> Enable timer interrupts after half a cycle or more of the count clock has elapsed from the beginning of the interrupt.

<Program example A> (count clock = 64/fx, CPU clock = fx)

```
TM20_VCT: SET1 TMMK20      ;  Disable timer interrupts (6 clocks)
          MOV  A,#xxH      ;  Set the rewrite value of higher byte (6 clocks)
          MOV  !0FF17H,A   ;  Rewrite CR20 higher byte (8 clocks)
          MOV  A,#yyH      ;  Set the rewrite value of lower byte (6 clocks)
          MOV  !0FF16H,A   ;  Rewrite CR20 lower byte (8 clocks)
          CLR1 TMIF20      ;  Clear interrupt request flag (6 clocks)
          CLR1 TMMK20      ;  Enable timer interrupts (6 clocks)
```

Total: 32 clocks or more[Note]

Note Because the INTTM20 signal becomes high level for half a cycle of the count clock after an interrupt is generated, an interrupt is set again if TMMK20 is cleared to 0 during this period.

<Countermeasure B> When rewriting using 16-bit access

<1> Disable interrupts (TMMK20 = 1).
<2> Rewrite CR20 (16 bits).
<3> Wait for one cycle or more of the count clock.
<4> Clear the interrupt request flag (TMIF20).
<5> Enable timer interrupts.

<Program example B> (count clock = 64/fx, CPU clock = fx)

```
TM20_VCT   SET1 TMMK20      ;  Disable timer interrupts
           MOVW AX,#xxyyH   ;  Set the rewrite value of CR20
           MOVW CR20,AX     ;  Rewrite CR20
           NOP
           NOP
            :               ;  32 NOP instructions (wait for 64/fx)Note
           NOP
           NOP
           CLR1 TMIF20      ;  Clear interrupt request flag
           CLR1 TMMK20      ;  Enable timer interrupts
```

Note Clear the interrupt request flag (TMIF20) after waiting for one cycle or more of the count clock from the instruction rewriting CR20 (MOVW CR20, AX).

CHAPTER 7 8-BIT TIMERS 50, 60

7.1 Function of 8-Bit Timers 50, 60

8-bit timers 50 and 60 are incorporated in the μPD789088 Subseries. The operation modes listed in the following table can be set via mode register settings.

Table 7-1. Operation Modes

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

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

The following functions can be used in this mode.

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

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

Operation as a 16-bit timer is enabled during cascade connection mode.
The following functions can be used in this mode.

- Interval timer with 16-bit resolution
- Square wave output with 16-bit resolution

(3) Carrier generator mode

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

(4) PWM output mode (pulse generator mode) (timer 60 only)

The timer output status inverts repeatedly due to the settings of TM60, CR60, and CRH60, and pulses of any duty ratio (pulse width) are output (the cycle and pulse width are both programmable).

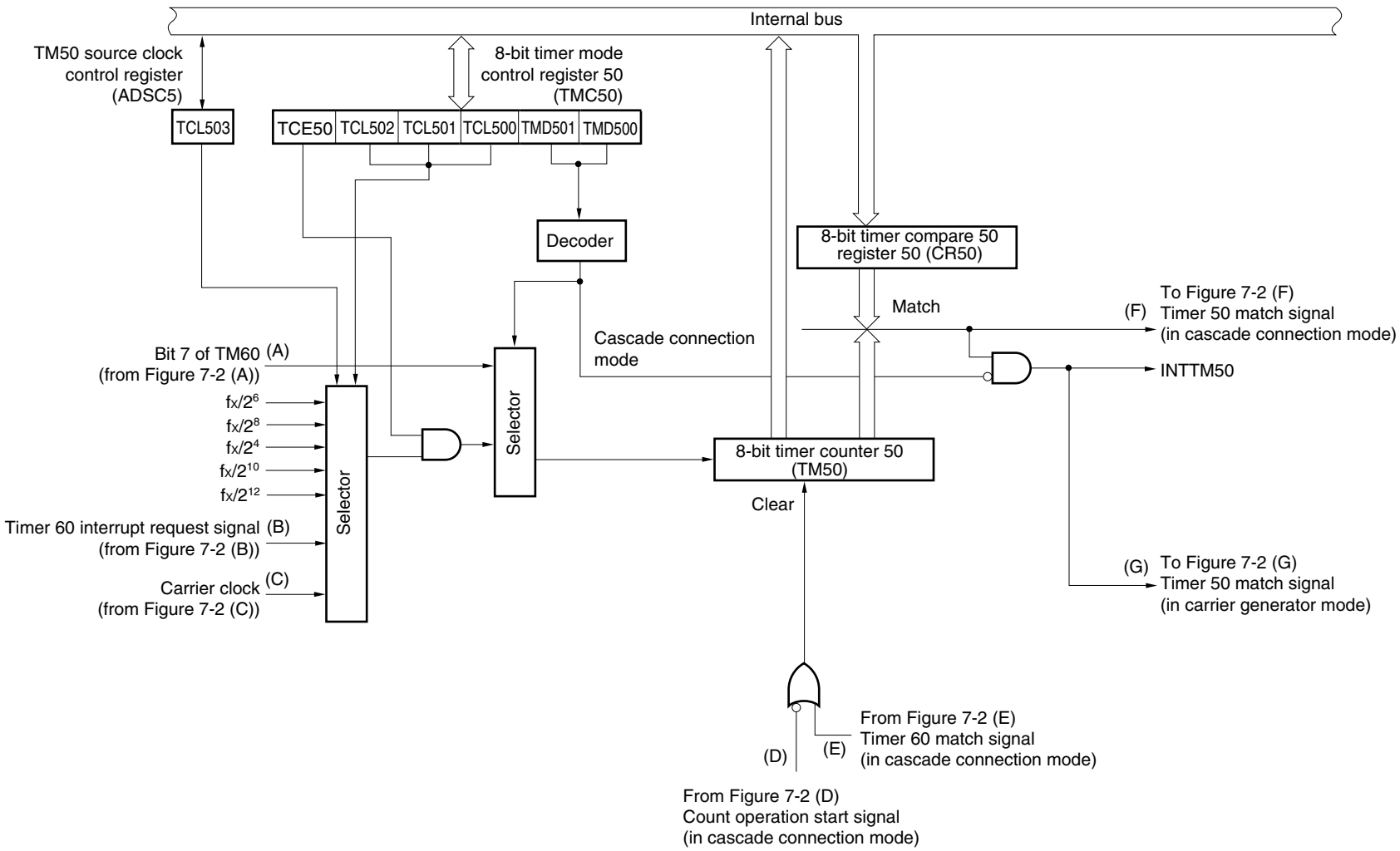
7.2 Configuration of 8-Bit Timers 50, 60

8-bit timers 50 and 60 include the following hardware.

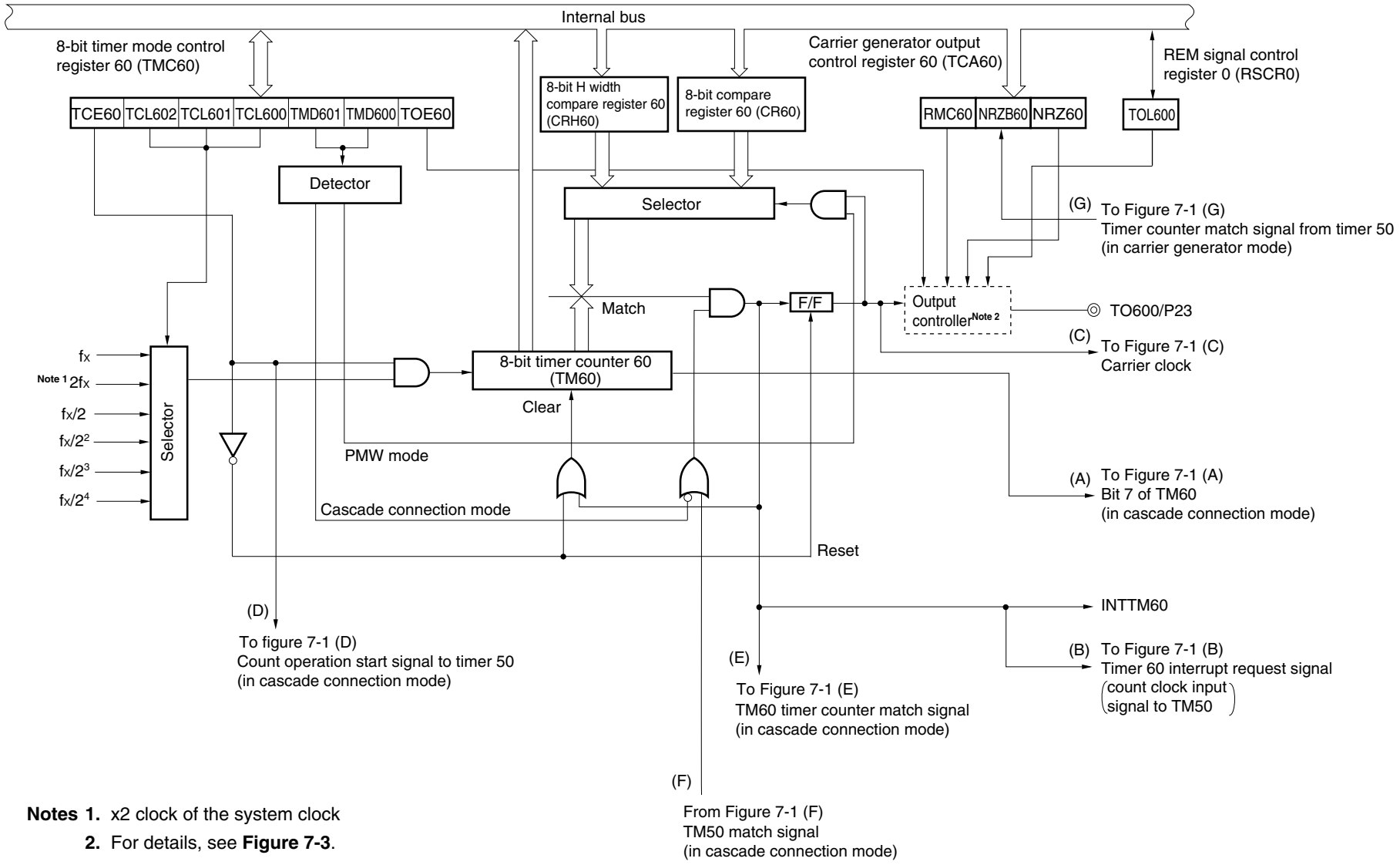
Table 7-2. Configuration of 8-Bit Timers 50, 60

Item	Configuration
Timer counters	8 bits × 2 (TM50, TM60)
Registers	Compare registers: 8 bits × 3 (CR50, CR60, CRH60)
Timer outputs	1 (TO600)
Control registers	8-bit timer mode control register 50 (TMC50) TM50 source clock control register (ADSC5) 8-bit timer mode control register 60 (TMC60) Carrier generator output control register 60 (TCA60) REM signal control register (RSCR0) Port mode register 2 (PM2) Port register 2 (P2)

★ **Figure 7-1. Block Diagram of Timer 50**



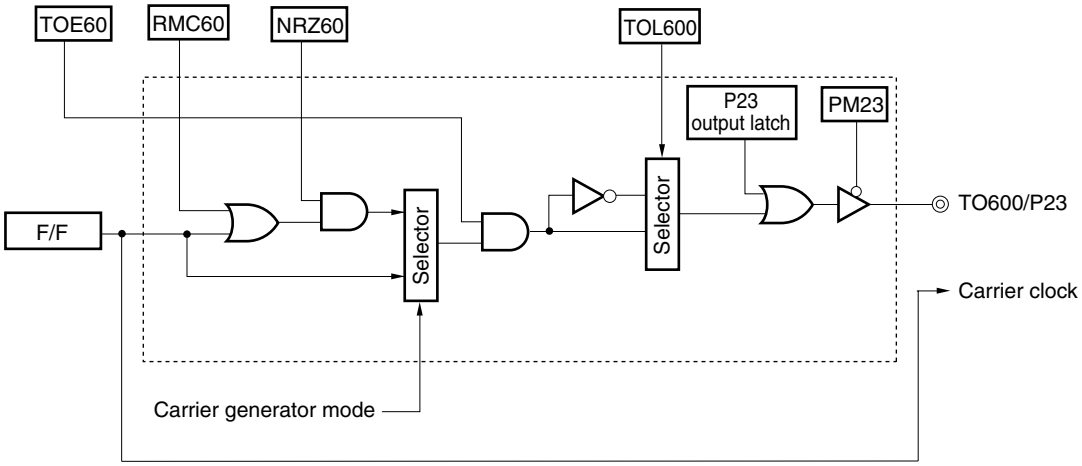
★ **Figure 7-2. Block Diagram of Timer 60**



Notes 1. x2 clock of the system clock

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

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



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

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

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

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

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

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

★ In carrier generator mode and PWM output mode, this registers is used for low-level width setting.

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

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

(3) 8-bit H width compare register 60 (CRH60)

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

★ The value set in CRH60 is constantly compared with TM60 count value, and an interrupt request (INTTM60) is generated if they match.

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

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

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

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

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

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

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

(a) Discrete mode

(i) TM50

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

(ii) TM60

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

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

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

(c) Carrier generator mode

(i) TM50

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

(ii) TM60

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

(d) PWM output mode (TM60 only)

- Reset
- When the TCE60 flag is cleared to 0
- When a match occurs between TM60 and CR60
- When a match occurs between TM60 and CRH60
- When the TM60 count value overflows

7.3 Registers Controlling 8-Bit Timers 50, 60

8-bit timers 50 and 60 are controlled by the following seven registers.

- 8-bit timer mode control register 50 (TMC50)
- TM50 source clock control register (ADSC5)
- 8-bit timer mode control register 60 (TMC60)
- Carrier generator output control register 60 (TCA60)
- REM signal control register (RSCR0)
- Port mode register 2 (PM2)
- Port register 2 (P2)

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

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

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

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

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

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
TMC50	TCE50	0	TCL502	TCL501	TCL500	TMD501	TMD500	0	FF65H	00H	R/W

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

TCL502	TCL501	TCL500	TCL503	Selection of timer 50 count clock[Note 2]
0	0	0	×	$f_X/2^6$ (78.1 kHz)
0	0	1	×	$f_X/2^8$ (19.5 kHz)
0	1	0	×	$f_X/2^4$ (313 kHz)
0	1	1	0	$f_X/2^{10}$ (4.88 kHz)
0	1	1	1	$f_X/2^{12}$ (1.22 kHz)
1	0	0	×	Timer 60 match signal
1	0	1	×	Carrier clock generated by timer 60
Other than above				Setting prohibited

TMD501	TMD500	TMD601	TMD600	Selection of operation mode for timer 50 and timer 60[Note 3]
0	0	0	0	Discrete mode (8-bit timer counter mode)
0	1	0	1	Cascade connection mode (16-bit timer counter mode)
0	0	1	1	Carrier generator mode
0	0	1	0	Timer 50: 8-bit counter mode Timer 60: PWM output mode
Other than above				Setting prohibited

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

2. The count clock selection is set to both the TMC50 register and ADSC5 register.

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

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

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

<1> Stop TM50 count operation.

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

<3> Start count operation.

3. Bits 0 and 6 must both be set to 0.

Remarks 1. f_X: System clock oscillation frequency

2. ×: Don't care

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

(2) TM50 source clock control register 50 (ADSC5)

TM50 source clock control register 50 (ADSC5) is used to control the timer 50 count clock.

ADSC5 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets ADSC5 to 00H.

Figure 7-5. Format of TM50 Source Clock Control Register 50

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ADSC5	0	0	0	0	0	0	0	TCL503	FF6DH	00H	R/W

TCL502	TCL501	TCL500	TCL503	Selection of timer 50 count clock[Note]
0	0	0	×	$f_X/2^6$ (78.1 kHz)
0	0	1	×	$f_X/2^8$ (19.5 kHz)
0	1	0	×	$f_X/2^4$ (313 kHz)
0	1	1	0	$f_X/2^{10}$ (4.88 kHz)
0	1	1	1	$f_X/2^{12}$ (1.22 kHz)
1	0	0	×	Timer 60 match signal
1	0	1	×	Carrier clock generated by timer 60
Other than above				Setting prohibited

Note The count clock selection is set to both the TMC50 register and ADSC5 register.

Remark ×: Don't care

(3) 8-bit timer mode control register 60 (TMC60)

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

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

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

Figure 7-6. Format of 8-Bit Timer Mode Control Register 60

Symbol	<7>	6	5	4	3	2	1	<0>	Address	After reset	R/W
TMC60	TCE60	0	TCL602	TCL601	TCL600	TMD601	TMD600	TOE60	FF69H	00H	R/W

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

TCL602	TCL601	TCL600	Selection of timer 60 count clock
0	0	0	f_X (5.0 MHz)
0	0	1	$2f_X$ (10.0 MHz) (×2 clock)[Note 2]
0	1	0	$f_X/2$ (2.5 MHz)
0	1	1	$f_X/2^2$ (1.25 MHz)
1	0	0	$f_X/2^3$ (625 kHz)
1	0	1	$f_X/2^4$ (312.5 kHz)
Other than above			Setting prohibited

TMD501	TMD500	TMD601	TMD600	Selection of operation mode for timer 50 and timer 60[Note 3]
0	0	0	0	Discrete mode (8-bit timer counter mode)
0	1	0	1	Cascade connection mode (16-bit timer counter mode)
0	0	1	1	Carrier generator mode
0	0	1	0	Timer 50: 8-bit counter mode Timer 60: PWM output mode
Other than above				Setting prohibited

TOE60	Control of timer output
0	Output disabled
1	Output enabled only for TO600

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

2. When using the ×2 clock, set the clock multiplication control register (CMC0) to 01H. The maximum operation voltage at this time is 3.6 V.

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

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

<1> Stop the TM60 count operation.

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

<3> Start count operation.

Remarks 1. f_X: System clock oscillation frequency

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

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

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

TCA60 is set with a 1-bit or 8-bit memory manipulation instruction.

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

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

Symbol	7	6	5	4	3	<2>	<1>	<0>	Address	After reset	R/W
TCA60	0	0	0	0	0	RMC60	NRZB60	NRZ60	FF6AH	00H	R/W[Note]

RMC60	Control of remote control output
0	When NRZ60 = 1, a carrier pulse is output to TO600 pin. When NRZ60 = 0, a low level is output to TO600 pin.
1	When NRZ60 = 1, high-level signal is output to TO600 pin. When NRZ60 = 0, a low level is output to TO600 pin.

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

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

Note Bit 0 is write-only

Cautions 1. At the count start, input the data required by the program to NRZ60 and NRZB60 in advance.

★ **2. When timer 60 output is disabled (TOE60 = 0), use of a 1-bit memory manipulation instruction for TCA60 is disabled (only an 8-bit memory manipulation instruction can be used).**

★ **3. When timer 60 output is enabled (TOE60 = 1), write to NRZ60 is invalid.**

(5) REM signal control register (RSCR0)

The REM signal control register (RSCR0) is used to control inversion of the timer 60 output (TO600).

RSCR0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets RSCR0 to 00H.

Figure 7-8. Format of REM Signal Control Register

Symbol	7	6	5	4	3	2	1	<0>	Address	After reset	R/W
RSCR0	0	0	0	0	0	0	0	TOL600	FF6BH	00H	R/W

TOL600	Inversion control of timer 60 output (TO600)
0	Not inverted (TO600)
1	Inverted ($\overline{\text{TO600}}$)

Note Set TOL600 to 0 when the P23/TO600 pin is used as a port function pin.

(6) Port mode register 2 (PM2)

This register is used to set the I/O mode of port 2 in 1-bit units.

When using the P23/TO600 pin as a timer output, set the PM23 and P23 output latch to 0.

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

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

Figure 7-9. Format of Port Mode Register 2

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W

PM2n	I/O mode of P2n pin (n = 0 to 7)
0	Output mode (output buffer is on)
1	Input mode (output buffer is off)

7.4 Operation of 8-Bit Timers 50, 60

7.4.1 Operation as 8-bit timer counter

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

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

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

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

<1> Disable operation of 8-bit timer counter n0 (TMn0) (TCEn0 = 0).
<2> Disable timer output of TO600 (TOE60 = 0) in case of timer 60.
<3> Set a count value in CRn0.
<4> Set the operation mode of timer n0 to 8-bit timer counter mode (see **Figures 7-4** and **7-6**).
<5> Set the count clock for timer n0 (see **Tables 7-3** and **7-4**).
<6> Enable the operation of TMn0 (TCEn0 = 1).

When the count value of 8-bit timer counter n0 (TMn0) matches the value set in CRn0, TMn0 is cleared to 00H and continues counting. At the same time, an interrupt request signal (INTTMn0) is generated.
Tables 7-3 and 7-4 show interval time, and Figures 7-10 to 7-15 show the timing of the interval timer operation.

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

Remark n = 5, 6

Table 7-3. Interval Time of Timer 50

TMC50			ADSC5	Minimum Interval Time	Maximum Interval Time	Resolution
TCL502	TCL501	TCL500	TCL503			
0	0	0	×	2^6/fx (12.8 μs)	2^{14}/fx (3.28 ms)	2^6/fx (12.8 μs)
0	0	1	×	2^8/fx (51.2 μs)	2^{16}/fx (13.1 ms)	2^8/fx (51.2 μs)
0	1	0	×	2^4/fx (3.2 μs)	2^{12}/fx (819 μs)	2^4/fx (3.2 μs)
0	1	1	0	2^{10}/fx (205 μs)	2^{18}/fx (52.4 ms)	2^{10}/fx (205 μs)
0	1	1	1	2^{12}/fx (819 μs)	2^{20}/fx (210 ms)	2^{12}/fx (819 μs)
1	0	0	×	Input cycle of timer 60 match signal	Input cycle of timer 60 match signal × 2^8	Input cycle of timer 60 match signal
1	0	1	×	Carrier clock cycle generated by timer 60	Carrier clock cycle generated by timer 60 × 2^8	Carrier clock cycle generated by timer 60

Remarks 1. fx: System clock oscillation frequency

2. ×: Don't care

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

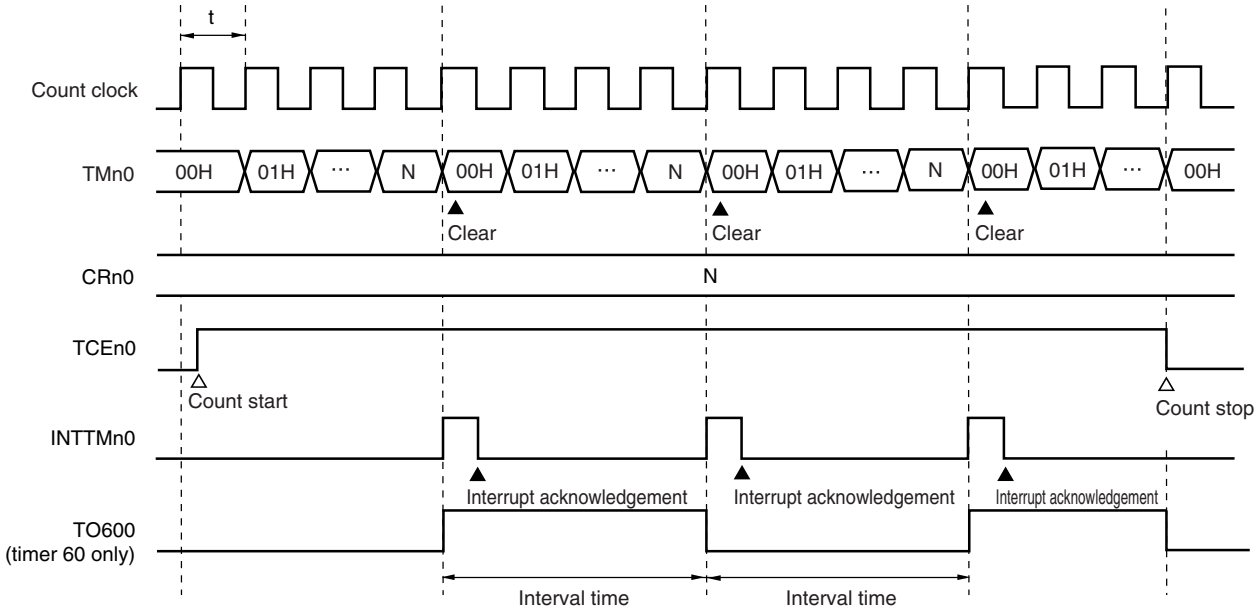
Table 7-4. Interval Time of Timer 60

TCL602	TCL601	TCL600	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	1/fx (0.2 μs)	2^8/fx (51.2 μs)	1/fx (0.2 μs)
0	0	1	1/2fx (0.1 μs)	2^7/fx (25.6 μs)	1/2fx (0.1 μs)
0	1	0	2/fx (0.4 μs)	2^9/fx (102 μs)	2/fx (0.4 μs)
0	1	1	2^2/fx (0.8 μs)	2^{10}/fx (205 μs)	2^2/fx (0.8 μs)
1	0	0	2^3/fx (1.6 μs)	2^{11}/fx (410 μs)	2^3/fx (1.6 μs)
1	0	1	2^4/fx (3.2 μs)	2^{12}/fx (819 μs)	2^4/fx (3.2 μs)

Remarks 1. fx: System clock oscillation frequency

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

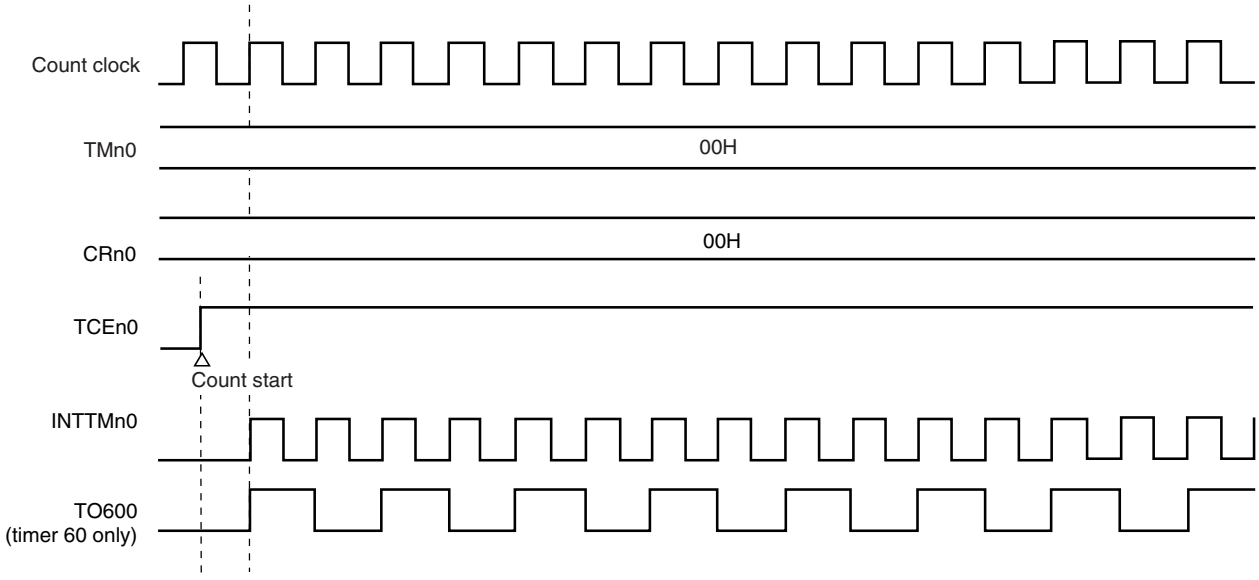
Figure 7-10. Timing of Interval Timer Operation with 8-Bit Resolution (Basic Operation)



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

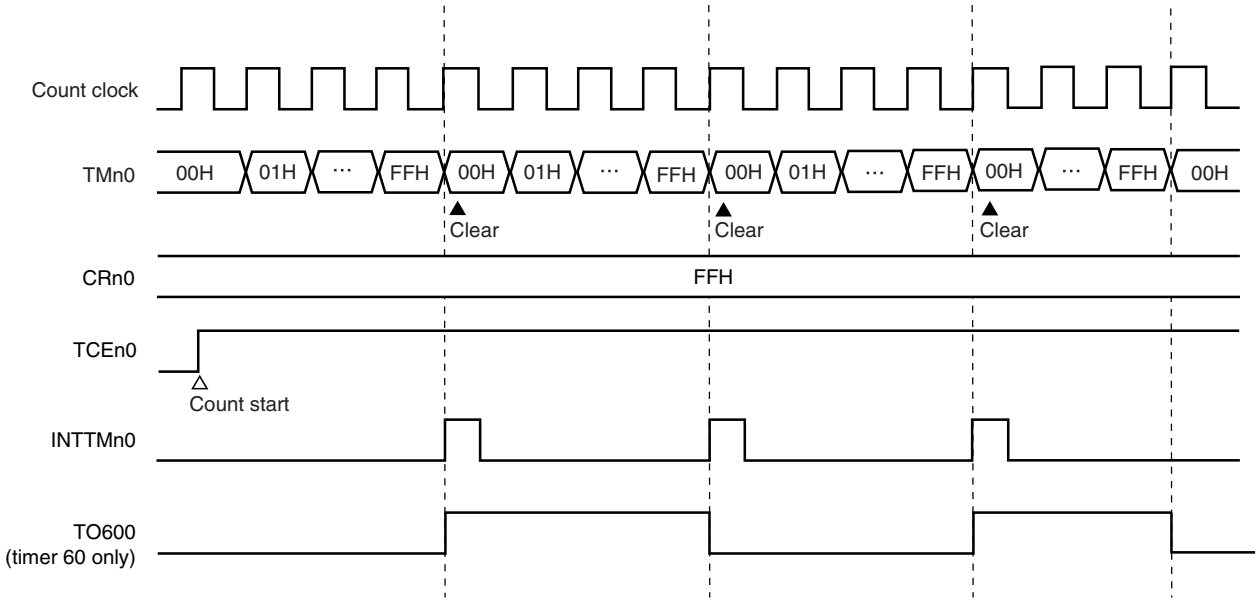
2. n = 5, 6

Figure 7-11. Timing of Interval Timer Operation with 8-Bit Resolution (When CRn0 Is Set to 00H)



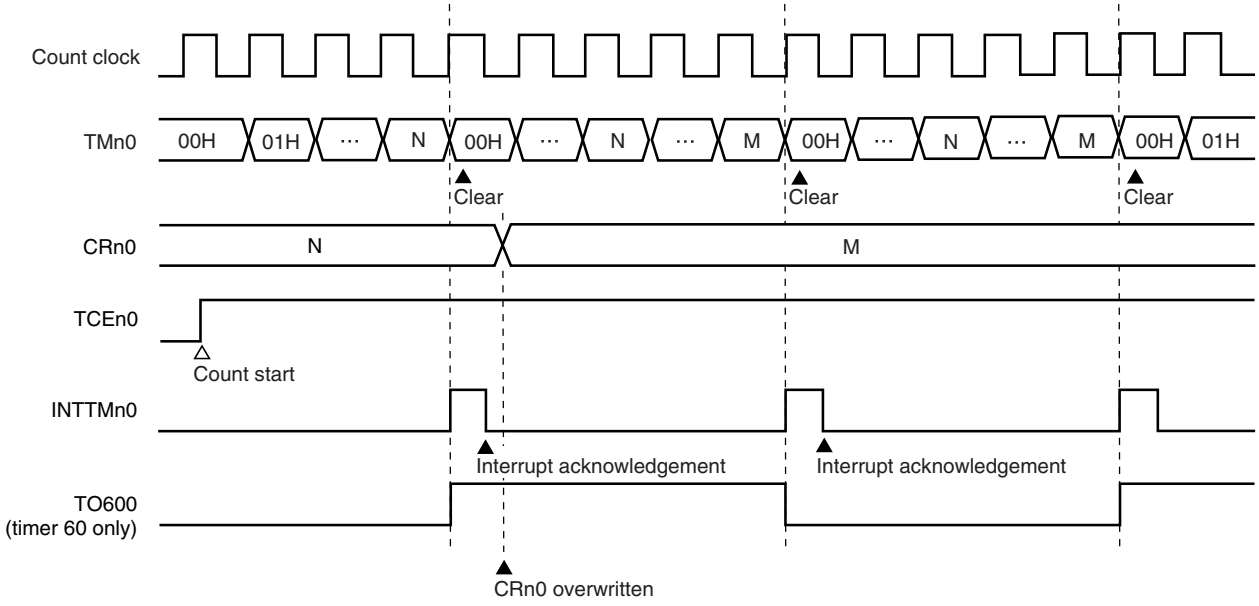
Remark n = 5, 6

Figure 7-12. Timing of Interval Timer Operation with 8-Bit Resolution (When CRn0 Is Set to FFH)



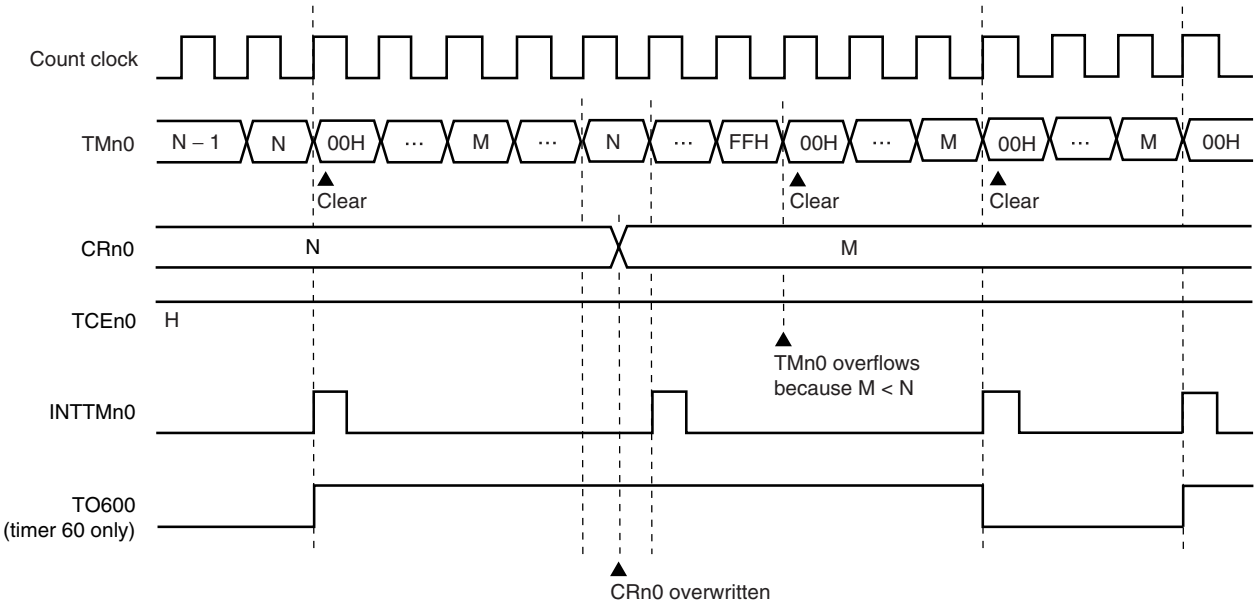
Remark n = 5, 6

Figure 7-13. Timing of Interval Timer Operation with 8-Bit Resolution (When CRn0 Changes from N to M (N < M))



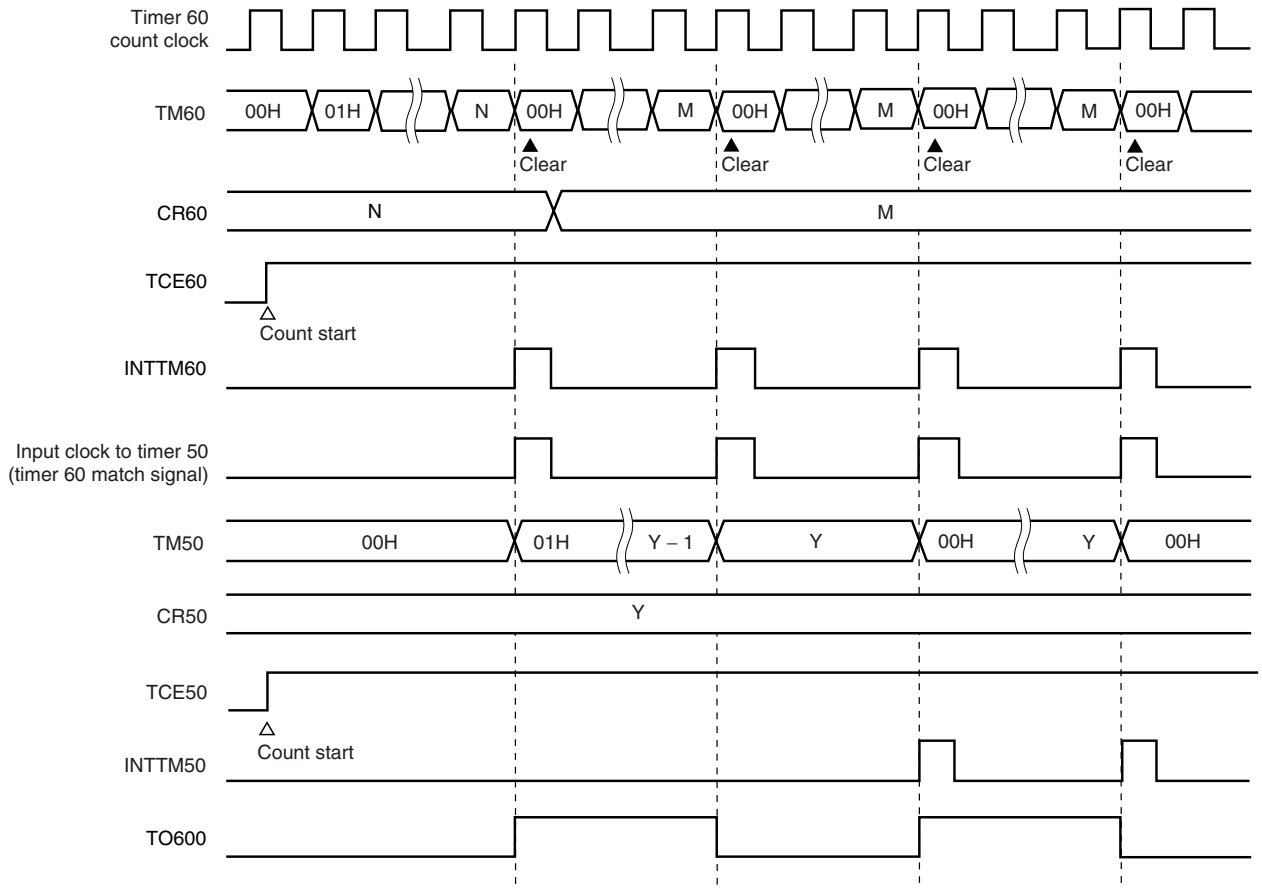
Remark n = 5, 6

Figure 7-14. Timing of Interval Timer Operation with 8-Bit Resolution (When CRn0 Changes from N to M (N > M))



Remark n = 5, 6

Figure 7-15. Timing of Interval Timer Operation with 8-Bit Resolution (When Timer 60 Match Signal Is Selected for Timer 50 Count Clock)



(2) Operation as square-wave output with 8-bit resolution (timer 60 only)

Square waves of any frequency can be output at an interval specified by the value preset in 8-bit compare register 60 (CR60).

To operate timer 60 for square-wave output, settings must be made in the following sequence.

<1> Disable operation of 8-bit timer counter 60 (TM60) (TCE60 = 0).
<2> Set a count value in CR60.
<3> Set the operation mode of timer 60 to 8-bit timer counter mode (see **Figures 7-4** and **7-6**).
<4> Set the count clock for timer 60 (see **Table 7-5**).
<5> Enable output of TO600 (TOE60 = 1).
<6> Set P23 to output mode (PM23 = 0).
<7> Set the output latches of P23 to 0.
<8> Enable the operation of TM60 (TCE60 = 1).

When the count value of TM60 matches the value set in CR60, the TO600 pin output will be inverted. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurs, TM60 is cleared to 00H and continues counting. At the same time, an interrupt request signal (INTTM60) is generated.

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

Table 7-5 shows the square-wave output range, and Figure 7-16 shows the timing of square-wave output.

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

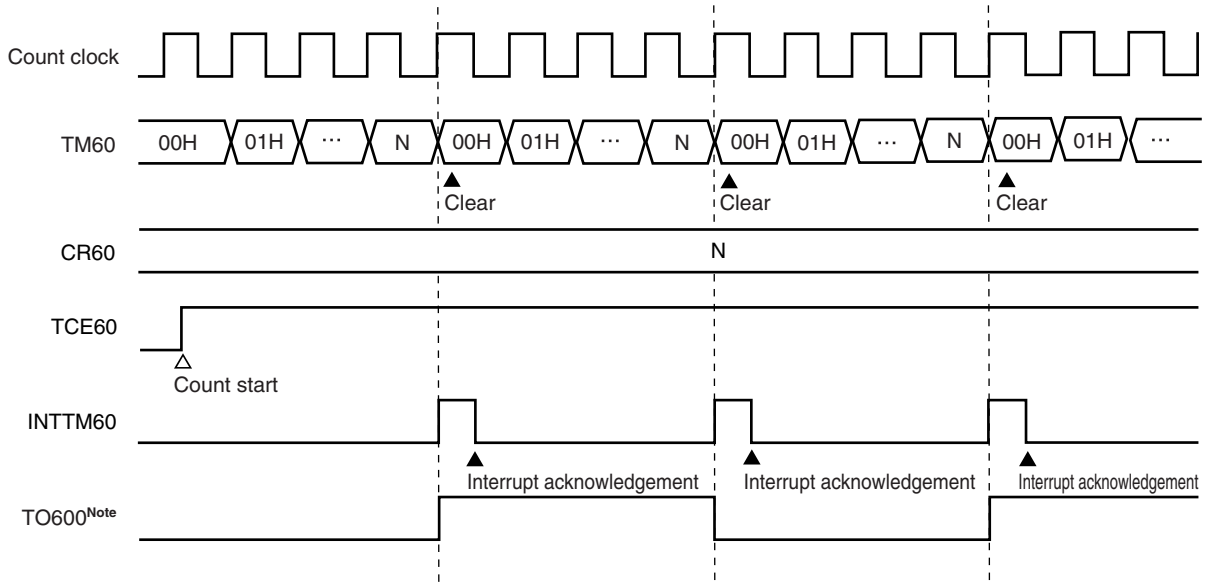
Table 7-5. Square-Wave Output Range of Timer 60

TCL602	TCL601	TCL600	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	0	$1/f_X$ (0.2 μs)	$2^8/f_X$ (51.2 μs)	$1/f_X$ (0.2 μs)
0	0	1	$1/2f_X$ (0.1 μs)	$2^7/f_X$ (25.6 μs)	$1/2f_X$ (0.1 μs)
0	1	0	$2/f_X$ (0.4 μs)	$2^9/f_X$ (102 μs)	$2/f_X$ (0.4 μs)
0	1	1	$2^2/f_X$ (0.8 μs)	$2^{10}/f_X$ (205 μs)	$2^2/f_X$ (0.8 μs)
1	0	0	$2^3/f_X$ (1.6 μs)	$2^{11}/f_X$ (410 μs)	$2^3/f_X$ (1.6 μs)
1	0	1	$2^4/f_X$ (3.2 μs)	$2^{12}/f_X$ (819 μs)	$2^4/f_X$ (3.2 μs)

Remarks 1. f_X: System clock oscillation frequency

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

Figure 7-16. Timing of Square-Wave Output with 8-Bit Resolution



Note The initial value of TO600 is low level when output is enabled (TOE60 = 1).

7.4.2 Operation as 16-bit timer counter

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

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

- Interval timer with 16-bit resolution
- Square-wave output with 16-bit resolution

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

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

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

<1> Disable operation of 8-bit timer counter 50 (TM50) and 8-bit timer counter 60 (TM60) (TCE50 = 0, TCE60 = 0).
<2> Disable timer output of TO600 (TOE60 = 0).
<3> Set the count clock for timer 60 (see **Table 7-6**).
<4> Set the operation mode of timer 50 and timer 60 to 16-bit timer counter mode (see **Figures 7-4** and **7-6**).
<5> Set a count value in CR50 and CR60.
<6> Enable the operation of TM50 and TM60 (TCE60 = 1[Note]).

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

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

Table 7-6 shows interval time, and Figure 7-17 shows the timing of the interval timer operation.

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

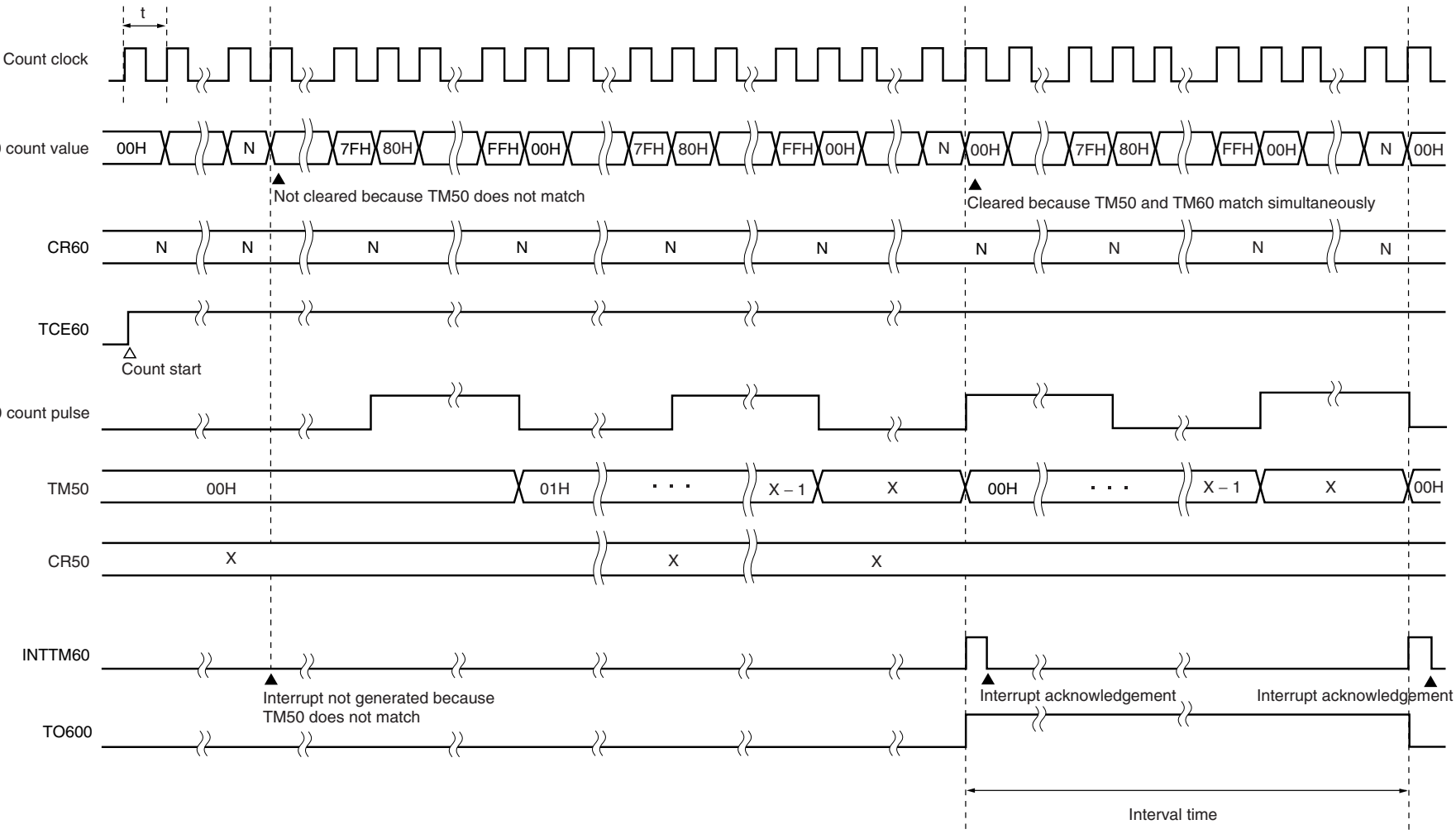
Table 7-6. Interval Time with 16-Bit Resolution

TCL602	TCL601	TCL600	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	0	$1/f_X$ (0.2 μs)	$2^{16}/f_X$ (13.1 ms)	$1/f_X$ (0.2 μs)
0	0	1	$1/2f_X$ (0.1 μs)	$2^{15}/f_X$ (6.55 ms)	$1/2f_X$ (0.1 μs)
0	1	0	$2/f_X$ (0.4 μs)	$2^{17}/f_X$ (26.2 ms)	$2/f_X$ (0.4 μs)
0	1	1	$2^2/f_X$ (0.8 μs)	$2^{18}/f_X$ (52.4 ms)	$2^2/f_X$ (0.8 μs)
1	0	0	$2^3/f_X$ (1.6 μs)	$2^{19}/f_X$ (105 ms)	$2^3/f_X$ (1.6 μs)
1	0	1	$2^4/f_X$ (3.2 μs)	$2^{20}/f_X$ (210 ms)	$2^4/f_X$ (3.2 μs)

Remarks 1. f_X: System clock oscillation frequency

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

Figure 7-17. Timing of Interval Timer Operation with 16-Bit Resolution



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

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

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

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

<1> Disable operation of 8-bit timer counter 50 (TM50) and 8-bit timer counter 60 (TM60) (TCE50 = 0, TCE60 = 0).
<2> Disable timer output of TO600 (TOE60 = 0).
<3> Set the count clock for timer 60 (see **Table 7-7**).
<4> Set the operation mode of timers 50 and 60 to 16-bit timer counter mode (see **Figures 7-4** and **7-6**).
<5> Set P23 to the output mode (PM23 = 0), set the P23 output latch to 0, and set TO600 to output enable (TOE60 = 1).
<6> Set count values in CR50 and CR60.
<7> Enable the operation of TM60 (TCE60 = 1[Note]).

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

When the count values of TM50 and TM60 simultaneously match the values set in CR50 and CR60 respectively, the TO600 pin output will be inverted. Through application of this mechanism, square waves of any frequency can be output. As soon as a match occurs, TM50 and TM60 are cleared to 00H and counting continues. At the same time, an interrupt request signal (INTTM60) is generated (INTTM50 is not generated).

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

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

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

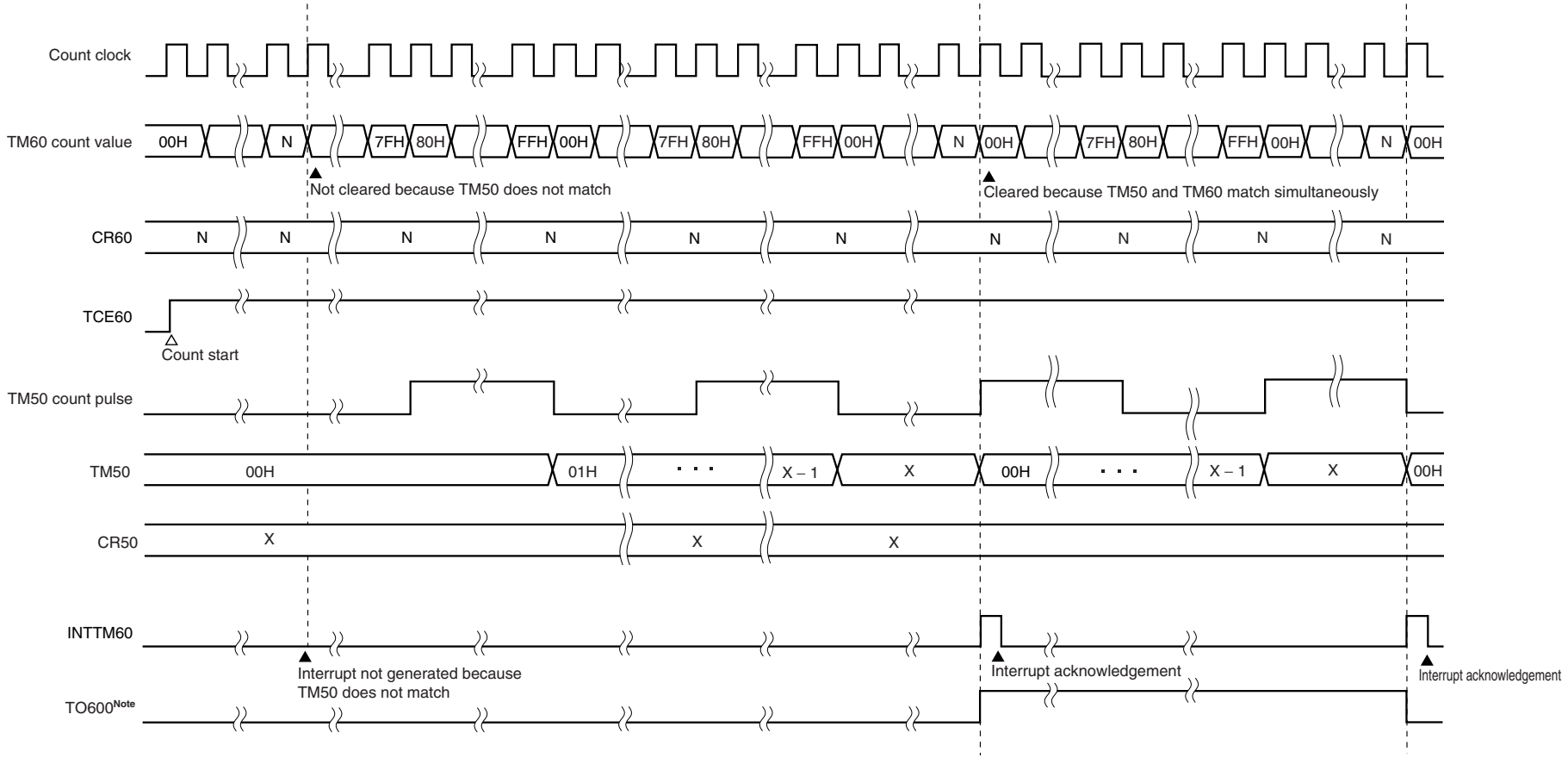
Table 7-7. Square-Wave Output Range with 16-Bit Resolution

TCL602	TCL601	TCL600	Minimum Pulse Width	Maximum Pulse Width	Resolution
0	0	0	$1/f_X$ (0.2 μs)	$2^{16}/f_X$ (13.1 ms)	$1/f_X$ (0.2 μs)
0	0	1	$1/2f_X$ (0.1 μs)	$2^{15}/f_X$ (6.55 ms)	$1/2f_X$ (0.1 μs)
0	1	0	$2/f_X$ (0.4 μs)	$2^{17}/f_X$ (26.2 ms)	$2/f_X$ (0.4 μs)
0	1	1	$2^2/f_X$ (0.8 μs)	$2^{18}/f_X$ (52.4 ms)	$2^2/f_X$ (0.8 μs)
1	0	0	$2^3/f_X$ (1.6 μs)	$2^{19}/f_X$ (105 ms)	$2^3/f_X$ (1.6 μs)
1	0	1	$2^4/f_X$ (3.2 μs)	$2^{20}/f_X$ (210 ms)	$2^4/f_X$ (3.2 μs)

Remarks 1. f_X: System clock oscillation frequency

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

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



Note The initial value of TO600 is low level when output is enabled.

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

7.4.3 Operation as carrier generator

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

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

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

<2> Disable timer output of TO600 (TOE60 = 0).

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

<4> Set the operation mode of timer 50 and timer 60 to carrier generator mode (see **Figures 7-4** and **7-6**).

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

<6> Set remote control output to carrier pulse (RMC60 (bit 2 of carrier generator output control register 60 (TCA60)) = 0).

Input the required value to NRZB60 (bit 1 of TCA60) by program.

Input a value to NRZ60 (bit 0 of TCA60) before it is reloaded from NRZB60.

<7> Set P23 to the output mode (PM23 = 0), set the P23 output latch to 0, and set TO600 to output enable (TOE60 = 1).

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

The operation of the carrier generator is as follows.

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

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

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

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

<5> When NRZ60 is 1, a carrier clock is output from the TO600 pin.

★ When NRZ60 is 0, a low level is output.

★ <6> Write the value to be transferred to NRZ60 next time to NRZB60.

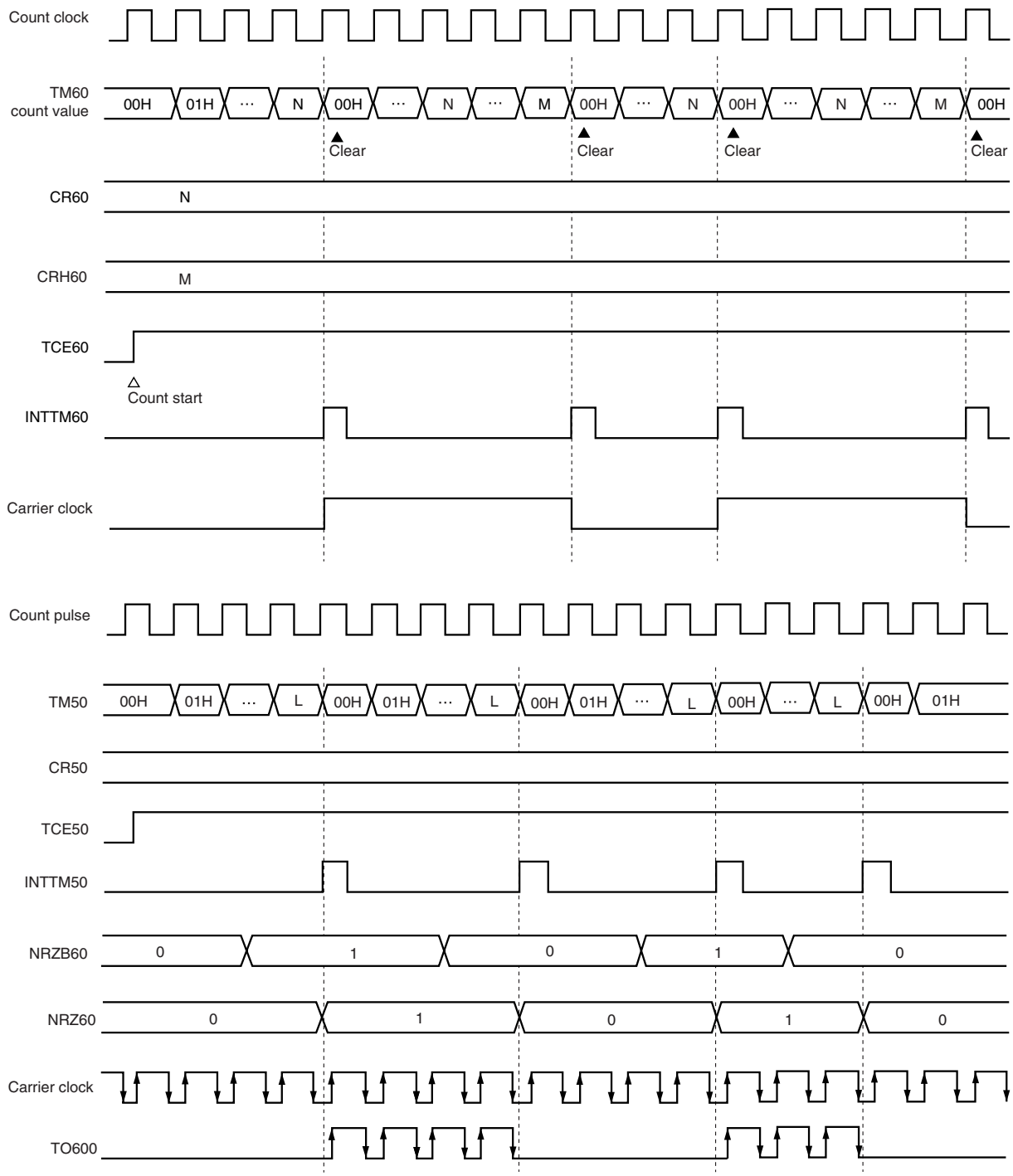
★ <7> Generate the desired carrier signal by repeating <4> to <6>.

Cautions 1. TCA60 cannot be set with a 1-bit memory manipulation instruction when timer 60 output is disabled (TOE60 = 0). Be sure to use an 8-bit memory manipulation instruction.

2. When setting the carrier generator operation again after stopping it once, reset NRZB60 because the previous value is not retained. In this case also a 1-bit memory manipulation instruction cannot be used when timer 60 output is disabled (TOE60 = 0). Be sure to use an 8-bit memory manipulation instruction.

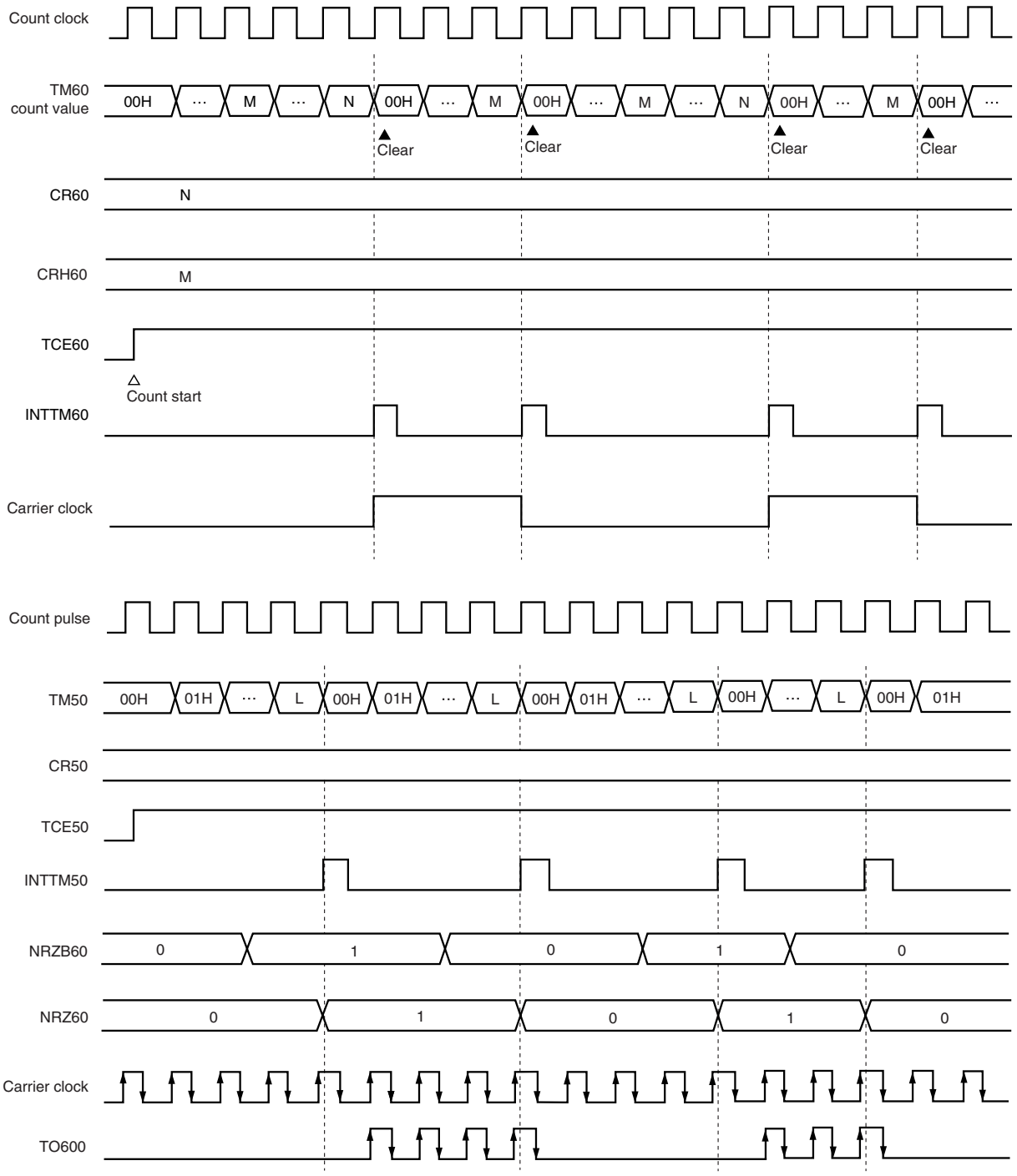
Figures 7-19 to 7-21 show the operation timing of the carrier generator.

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



Remark 00H ≤ N < M ≤ FFH, L = 00H to FFH

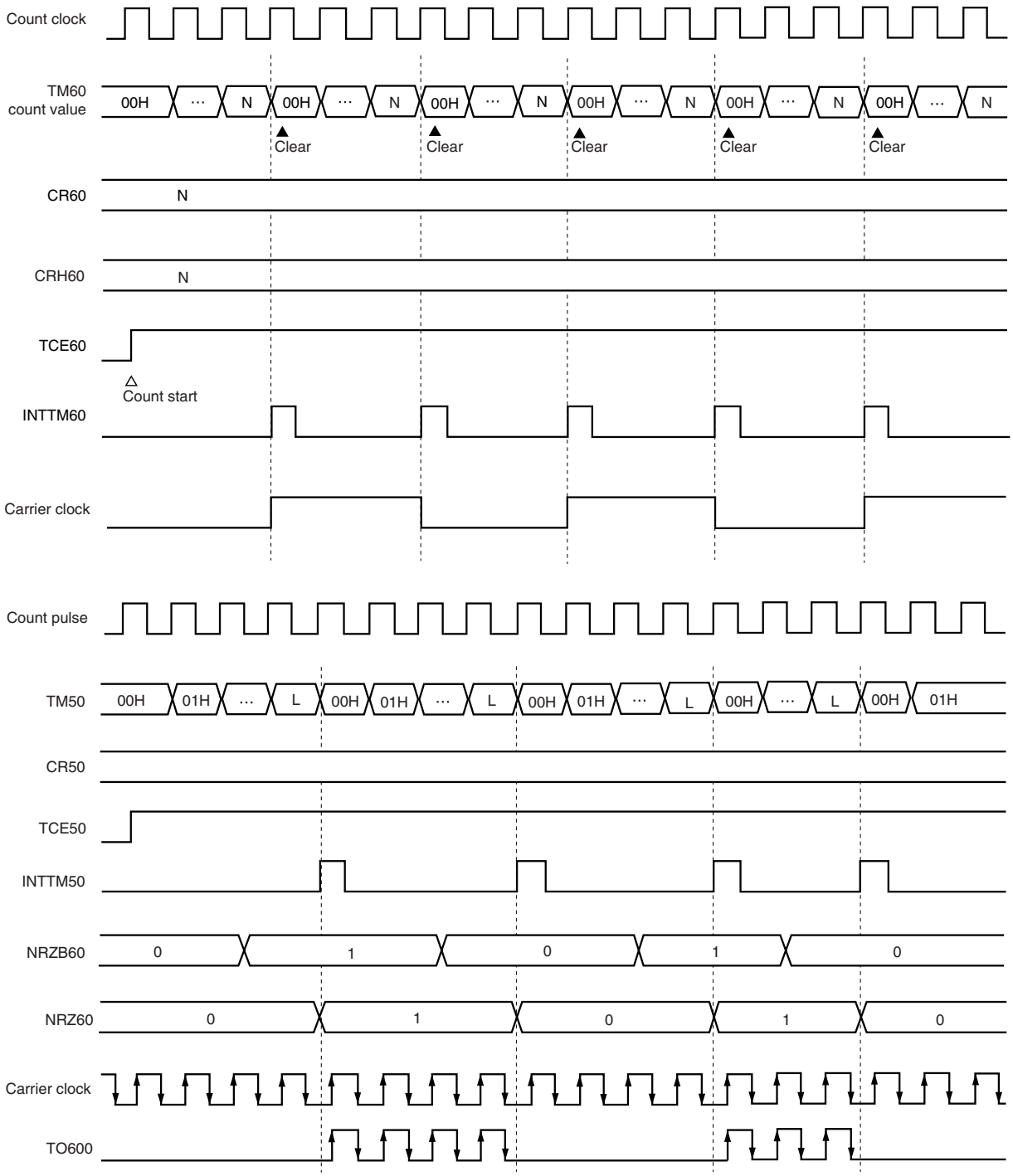
Figure 7-20. Timing of Carrier Generator Operation
(When CR60 = N, CRH60 = M (M < N), Phases of Carrier Clock and NRZ60 Are Asynchronous)



Remarks 1. 00H ≤ M < N ≤ FFH, L = 00H to FFH

★ **2.** This timing chart shows an example in which the value of NRZ60 is changed while the carrier clock is high.

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



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

7.4.4 Operation as PWM output (timer 60 only)

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

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

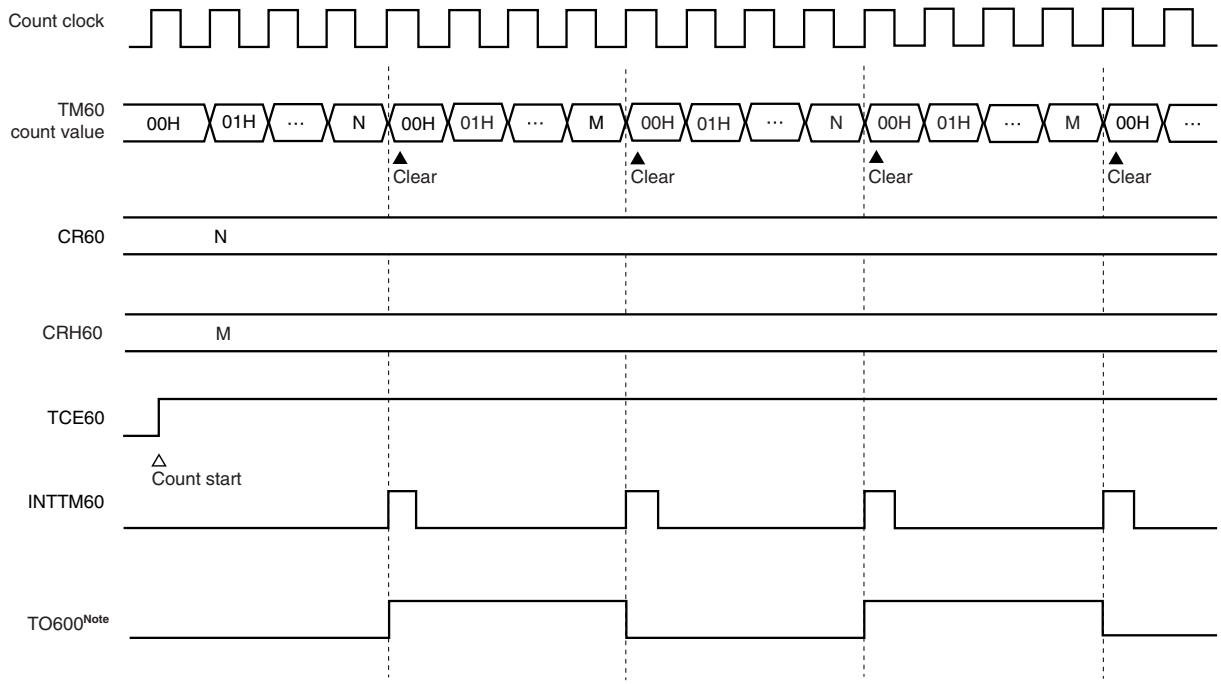
<1> Disable operation of TM60 (TCE60 = 0).
<2> Disable timer output of TO600 (TOE60 = 0).
<3> Set count values in CR60 and CRH60.
<4> Set the operation mode of timer 60 to the PWM output mode (see **Figure 7-6**).
<5> Set the count clock for timer 60.
<6> Set P23 to the output mode (PM23 = 0) and the P23 output latch to 0 and enable timer output of TO600 (TOE60 = 1).
<7> Enable the operation of TM60 (TCE60 = 1).

The operation in the PWM output mode is as follows.

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

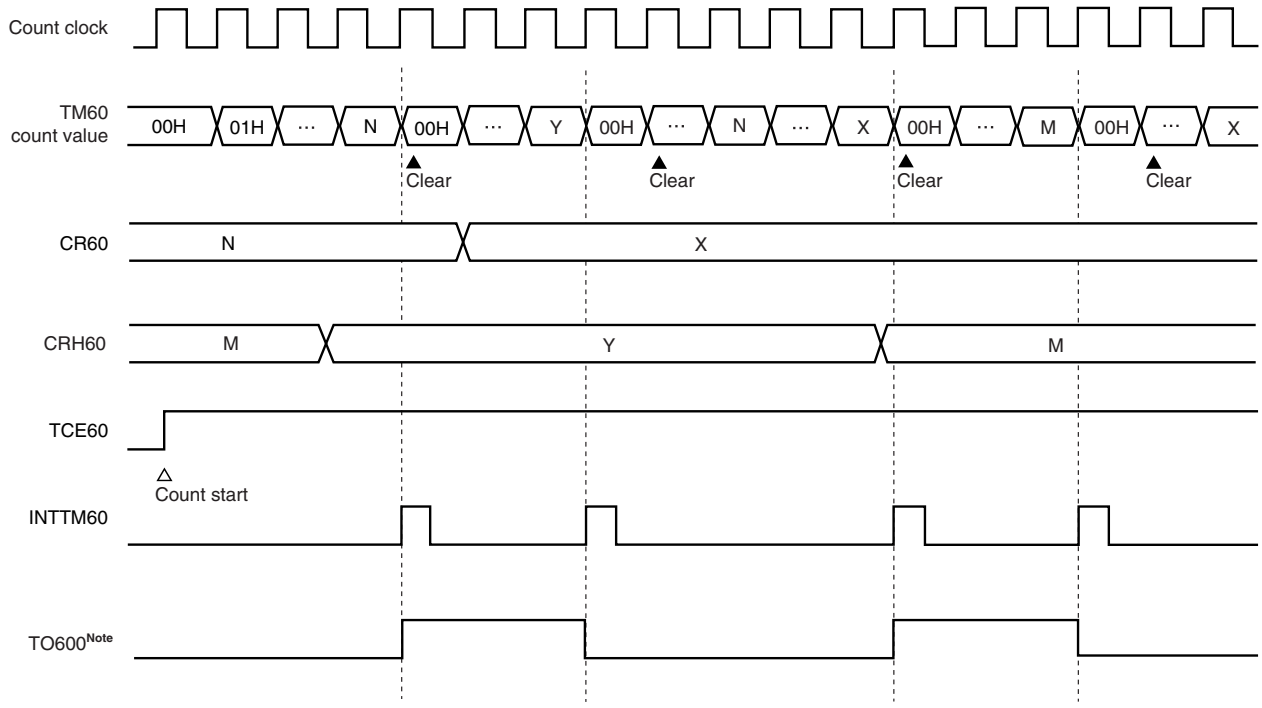
A pulse of any duty ratio and cycle is output by repeating <1> to <4> above. Figures 7-22 and 7-23 show the operation timing in the PWM output mode.

Figure 7-22. PWM Output Mode Timing (Basic Operation)



Note The initial value of TO600 is low level when output is enabled (TOE60 = 1).

Figure 7-23. PWM Output Mode Timing (When CR60 and CRH60 Are Overwritten)



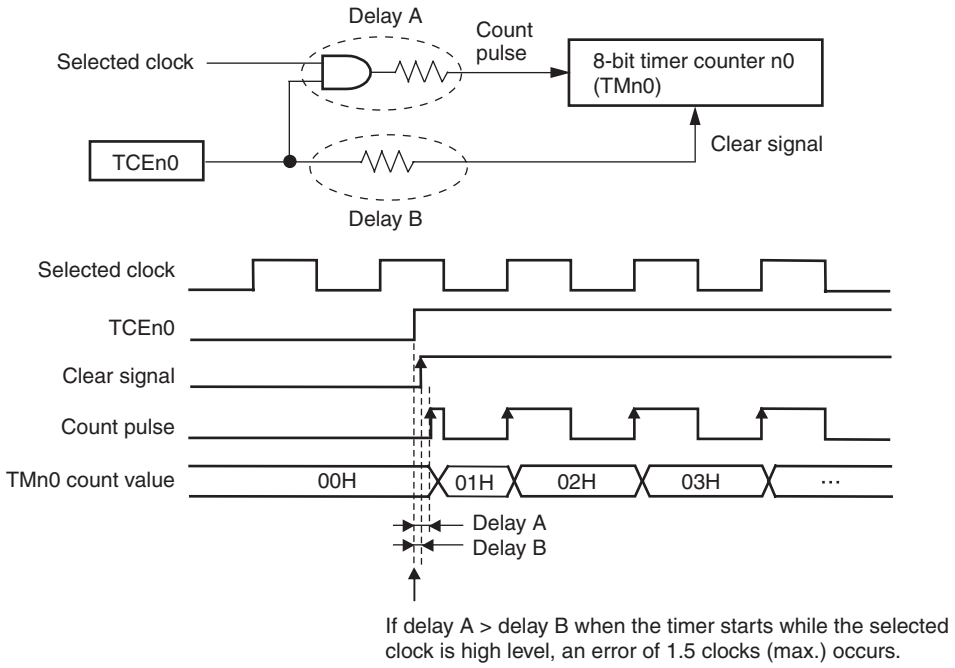
Note The initial value of TO600 is low level when output is enabled (TOE60 = 1).

7.5 Notes on Using 8-Bit Timers 50, 60

★ **(1) Error on starting timer**

An error of up to 1.5 clocks (max.) is included in the time between the timer being started and a match signal being generated. This is because the counter may be incremented by detecting a rising edge at the timing at which the timer starts while the count clock is high level (see **Figure 7-24**).

Figure 7-24. Case in Which Error of 1.5 Clocks (Max.) Occurs

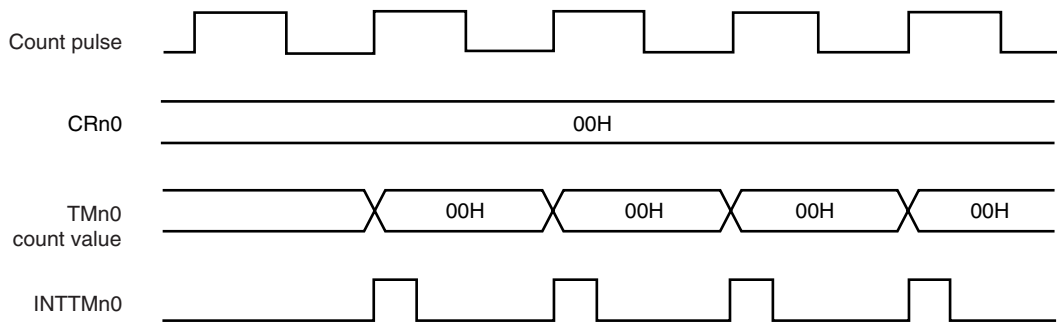


Remark n = 5, 6

(2) Setting of 8-bit compare register n0

8-bit compare register n0 (CRn0) can be set to 00H. Therefore, one pulse can be counted.

Figure 7-25. Timing of 1-Pulse Count Operation (8-Bit Resolution)



Remark n = 5, 6

CHAPTER 8 8-BIT TIMER 80

8.1 Function of 8-Bit Timer 80

8-bit timer 80 has the following functions.

- Interval timer

(1) 8-bit interval timer

When 8-bit timer 80 is used as an interval timer, it generates an interrupt at a time interval set in advance.

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

Minimum Interval Time	Maximum Interval Time	Resolution
2^4/fx (3.2 μs)	2^{12}/fx (819 μs)	2^4/fx (3.2 μs)
2^8/fx (51.2 μs)	2^{16}/fx (13.1 ms)	2^8/fx (51.2 μs)
2^{16}/fx (13.1 ms)	2^{24}/fx (3.36 s)	2^{16}/fx (13.1 ms)

Remarks 1. fx: System clock oscillation frequency

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

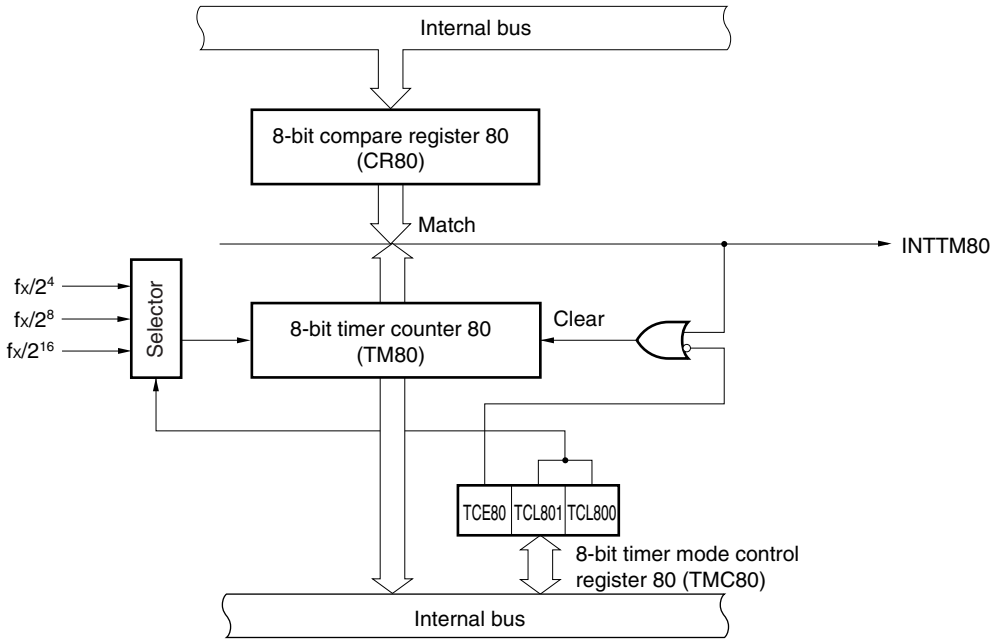
8.2 Configuration of 8-Bit Timer 80

8-bit timer 80 includes the following hardware.

Table 8-2. Configuration of 8-Bit Timer 80

Item	Configuration
Timer counter	8 bits × 1 (TM80)
Register	Compare register: 8 bits × 1 (CR80)
Timer output	None
Control register	8-bit timer mode control register 80 (TMC80)

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



(1) 8-bit compare register 80 (CR80)

A value specified in CR80 is compared with the count in 8-bit timer counter 80 (TM80). If they match, an interrupt request (INTTM80) is issued.

CR80 is set with an 8-bit memory manipulation instruction. Any value from 00H to FFH can be set.

$\overline{\text{RESET}}$ input makes CR80 undefined.

Caution Before rewriting CR80, stop the timer operation. If CR80 is rewritten while the timer operation is enabled, the match interrupt request signal may be generated immediately.

(2) 8-bit timer counter 80 (TM80)

TM80 is used to count the number of pulses.

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

$\overline{\text{RESET}}$ input clears TM80 to 00H.

8.3 Register Controlling 8-Bit Timer 80

The following register is used to control 8-bit timer 80.

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

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

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

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

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

Figure 8-2. Format of 8-Bit Timer Mode Control Register 80

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
TMC80	TCE80	0	0	0	0	TCL801	TCL800	0	FF62H	00H	R/W

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

TCL801	TCL800	8-bit timer counter 80 count clock selection
0	0	$f_X/2^4$ (312.5 kHz)
0	1	$f_X/2^8$ (19.5 kHz)
1	0	$f_X/2^{16}$ (76.3 kHz)
1	1	Setting prohibited

Cautions 1. Always stop the timer before setting TMC80.

2. Bits 0 and 3 to 6 must all be set to 0.

Remarks 1. f_X: System clock oscillation frequency

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

8.4 Operation of 8-Bit Timer 80

8.4.1 Operation as interval timer

The interval timer repeatedly generates an interrupt at a time interval specified by the count value preset in 8-bit compare register 80 (CR80).

To operate 8-bit timer 80 as an interval timer, settings must be made in the following sequence.

<1> Disable operation of 8-bit timer counter 80 (TM80) (TCE80 (bit 7 of 8-bit timer mode control register 80 (TMC80)) = 0).
<2> Set the count clock of 8-bit timer 80 (see **Table 8-3**).
<3> Set a count value in CR80.
<4> Enable the operation of TM80 (TCE80 = 1).

When the count value of 8-bit timer counter 80 (TM80) matches the value set in CR80, TM80 is cleared to 0 and continues counting. At the same time, an interrupt request signal (INTTM80) is generated.

Table 8-3 shows interval time, and Figure 8-3 shows the timing of interval timer operation.

Cautions 1. Stop the timer operation before rewriting CR80. If CR80 is rewritten while the timer operation is enabled, a match signal may be generated immediately (an interrupt request will be generated if interrupts are enabled).

2. If setting the count clock to TMC80 and enabling the operation of TM80 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 as an interval timer, therefore, make the settings in the above sequence.

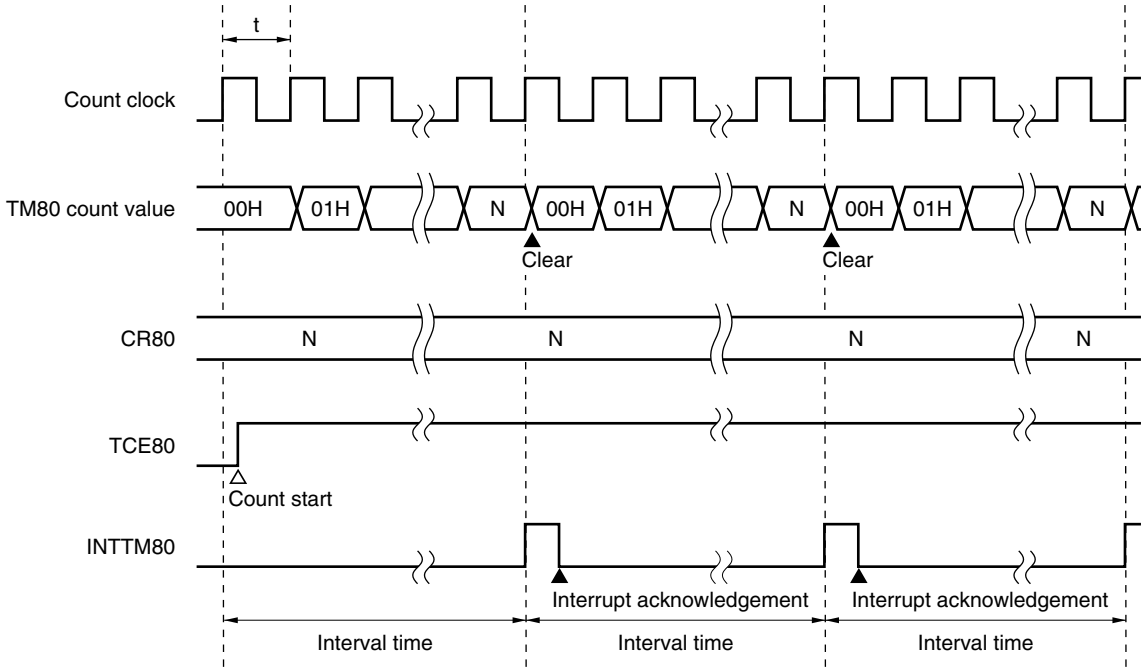
Table 8-3. Interval Time of 8-Bit Timer 80

TCL801	TCL800	Minimum Interval Time	Maximum Interval Time	Resolution
0	0	2^4/fx (3.2 μs)	2^{12}/fx (819 μs)	2^4/fx (3.2 μs)
0	1	2^8/fx (51.2 μs)	2^{16}/fx (13.1 ms)	2^8/fx (51.2 μs)
1	0	2^{16}/fx (13.1 ms)	2^{24}/fx (3.36 s)	2^{16}/fx (13.1 ms)
1	1	Setting prohibited		

Remarks 1. fx: System clock oscillation frequency

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

Figure 8-3. Interval Timer Operation Timing



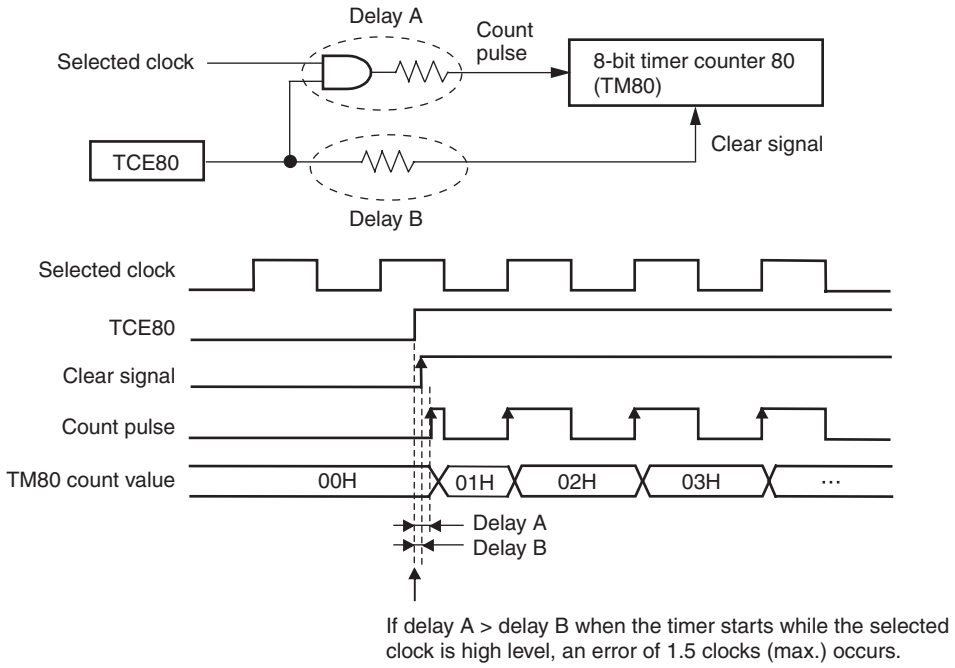
Remark Interval time = (N + 1) × t
N = 00H to FFH

8.5 Notes on Using 8-Bit Timer 80

★ **(1) Error on starting timer**

An error of up to 1.5 clocks (max.) is included in the time between the timer being started and a match signal being generated. This is because the counter may be incremented by detecting a rising edge at the timing at which the timer starts while the count clock is high level (see **Figure 8-4**).

Figure 8-4. Case in Which Error of 1.5 Clocks (Max.) Occurs

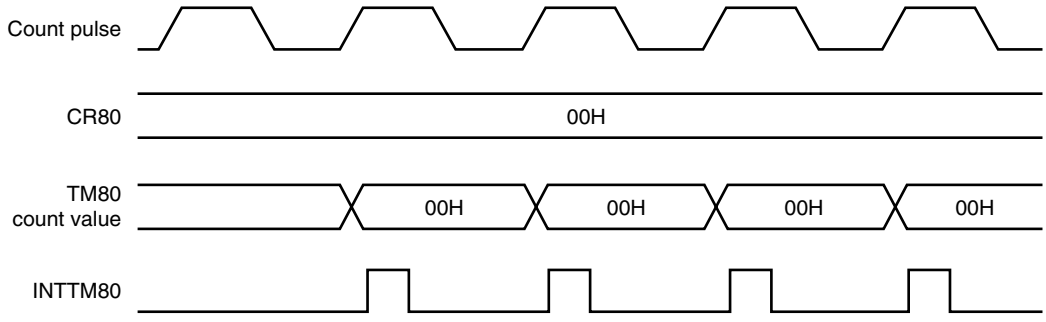


If delay A > delay B when the timer starts while the selected clock is high level, an error of 1.5 clocks (max.) occurs.

(2) Setting of 8-bit compare register 80

8-bit compare register 80 (CR80) can be set to 00H. Therefore, one pulse can be counted.

Figure 8-5. Timing of 1-Pulse Count



Caution Before rewriting CR80, stop the timer operation. If CR80 is rewritten while the timer operation is enabled, a match interrupt request signal may be generated immediately.

(3) Notes on setting STOP mode

Be sure to stop timer operation (TCE80 = 0) before executing the STOP instruction.

CHAPTER 9 WATCHDOG TIMER

9.1 Function of Watchdog Timer

The watchdog timer has the following functions:

- Watchdog timer
- Interval timer

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

(1) Watchdog timer

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

Table 9-1. 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

fx: System clock oscillation frequency

(2) Interval timer

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

Table 9-2. Interval Time

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

fx: System clock oscillation frequency

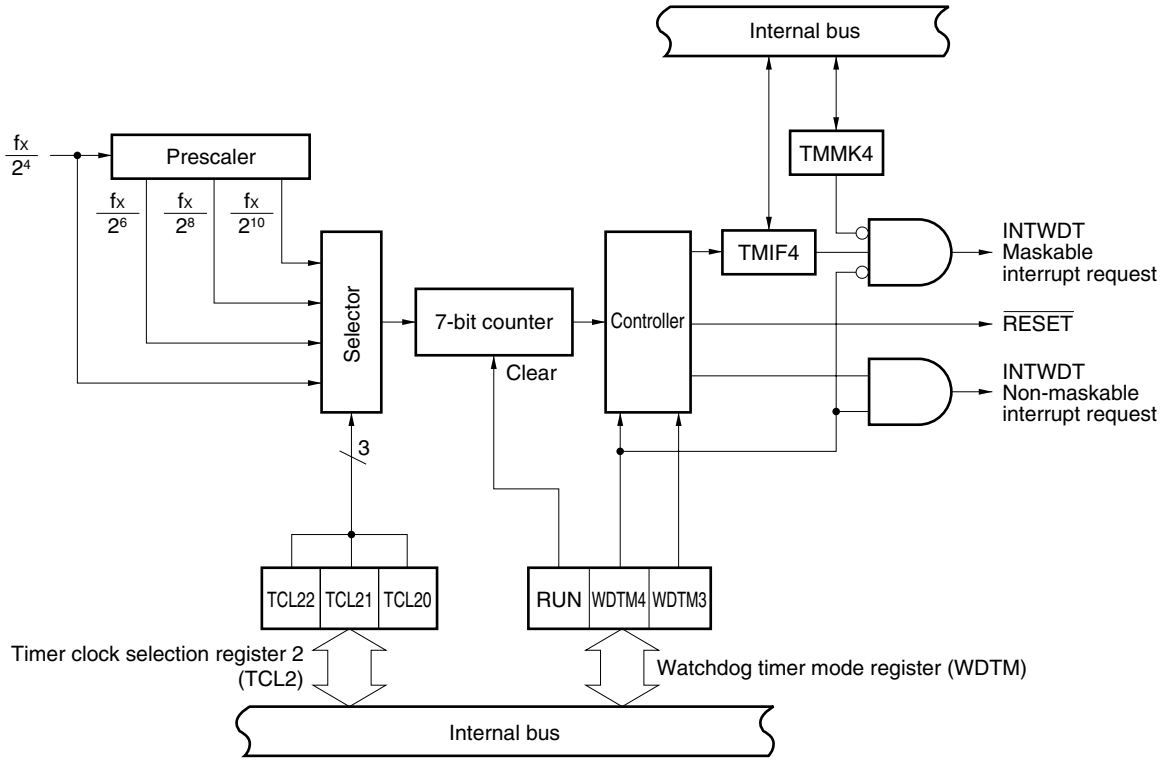
9.2 Configuration of Watchdog Timer

The watchdog timer includes the following hardware.

Table 9-3. Configuration of Watchdog Timer

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

Figure 9-1. Block Diagram of Watchdog Timer



9.3 Registers Controlling Watchdog Timer

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

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

(1) Timer clock selection register 2 (TCL2)

This register sets the watchdog timer count clock.

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

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

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

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

TCL22	TCL21	TCL20	Count clock selection	Interval time
0	0	0	$f_X/2^4$ (313 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 the operation mode of the watchdog timer, and enables/disables counting of the watchdog timer.

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

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

Figure 9-3. Format of Watchdog Timer Mode Register

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

RUN	Watchdog timer operation selection[Note 1]
0	Stops counting.
1	Clears counter and starts counting.

WDTM4	WDTM3	Watchdog timer operation mode selection[Note 2]
0	0	Operation stop
0	1	Interval timer mode (Generates a maskable interrupt upon overflow occurrence.)[Note 3]
1	0	Watchdog timer mode 1 (Generates a non-maskable interrupt upon overflow occurrence.)
1	1	Watchdog timer mode 2 (Starts a reset operation upon overflow occurrence.)

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

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

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

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

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

9.4 Operation of Watchdog Timer

9.4.1 Operation as watchdog timer

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

The count clock (inadvertent loop detection time interval) of the watchdog timer can be selected by bits 0 to 2 (TCL20 to TCL22) of timer clock selection register 2 (TCL2). By setting bit 7 (RUN) of WDTM to 1, the watchdog timer is started. Set RUN to 1 within the set inadvertent loop detection time interval after the watchdog timer has been started. By setting RUN to 1, the watchdog timer can be cleared and start counting. If RUN is not set to 1, and the inadvertent loop detection time is exceeded, a system reset signal or a non-maskable interrupt is generated, depending on the value of bit 3 (WDTM3) of WDTM.

The watchdog timer continues operation in HALT mode, but stops in STOP mode. Therefore, first set RUN to 1 to clear the watchdog timer before executing the STOP instruction.

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

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

TCL22	TCL21	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

fx: System clock oscillation frequency

9.4.2 Operation as interval timer

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

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

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

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

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

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

Table 9-5. Interval Generated Using Interval Timer

TCL22	TCL21	TCL20	Interval Time	At fx = 5.0 MHz
0	0	0	$2^{11} \times 1/fx$	410 μs
0	1	0	$2^{13} \times 1/fx$	1.64 ms
1	0	0	$2^{15} \times 1/fx$	6.55 ms
1	1	0	$2^{17} \times 1/fx$	26.2 ms

fx: System clock oscillation frequency

CHAPTER 10 SERIAL INTERFACE 20

10.1 Function of Serial Interface 20

Serial interface 20 has the following three modes.

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

(1) Operation stop mode

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

(2) Asynchronous serial interface (UART) mode

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

Serial interface 20 contains a UART-dedicated baud rate generator, enabling communication over a wide range of baud rates. It is also possible to define baud rates by dividing the frequency of the clock input to the ASCK20 pin and the square-wave output signal of timer 60.

(3) 3-wire serial I/O mode (switchable between MSB-first and LSB-first transmission)

This mode is used to transmit 8-bit data, using three lines: a serial clock ($\overline{\text{SCK20}}$) line and two serial data lines (SI20 and SO20).

As it supports simultaneous transmission and reception, 3-wire serial I/O mode requires less processing time for data transmission than asynchronous serial interface mode.

Because, in 3-wire serial I/O mode, it is possible to select whether 8-bit data transmission begins with the MSB or LSB, serial interface 20 can be connected to any device regardless of whether that device is designed for MSB-first or LSB-first transmission.

3-wire serial I/O mode is useful for connecting peripheral I/O circuits and display controllers having conventional synchronous serial interfaces, such as those of the 75X/XL, 78K, and 17K Series devices.

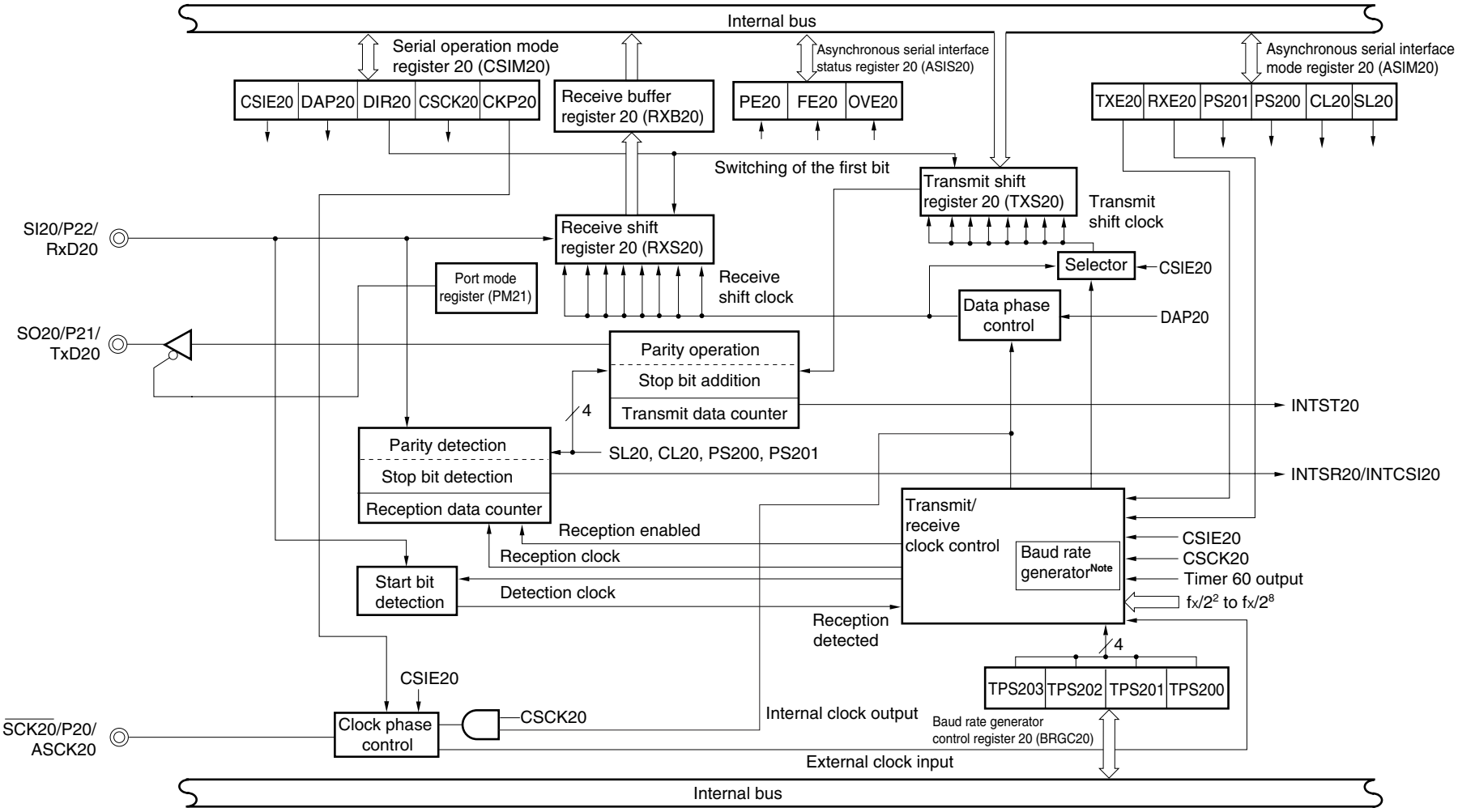
10.2 Configuration of Serial Interface 20

Serial interface 20 includes the following hardware.

Table 10-1. Configuration of Serial Interface 20

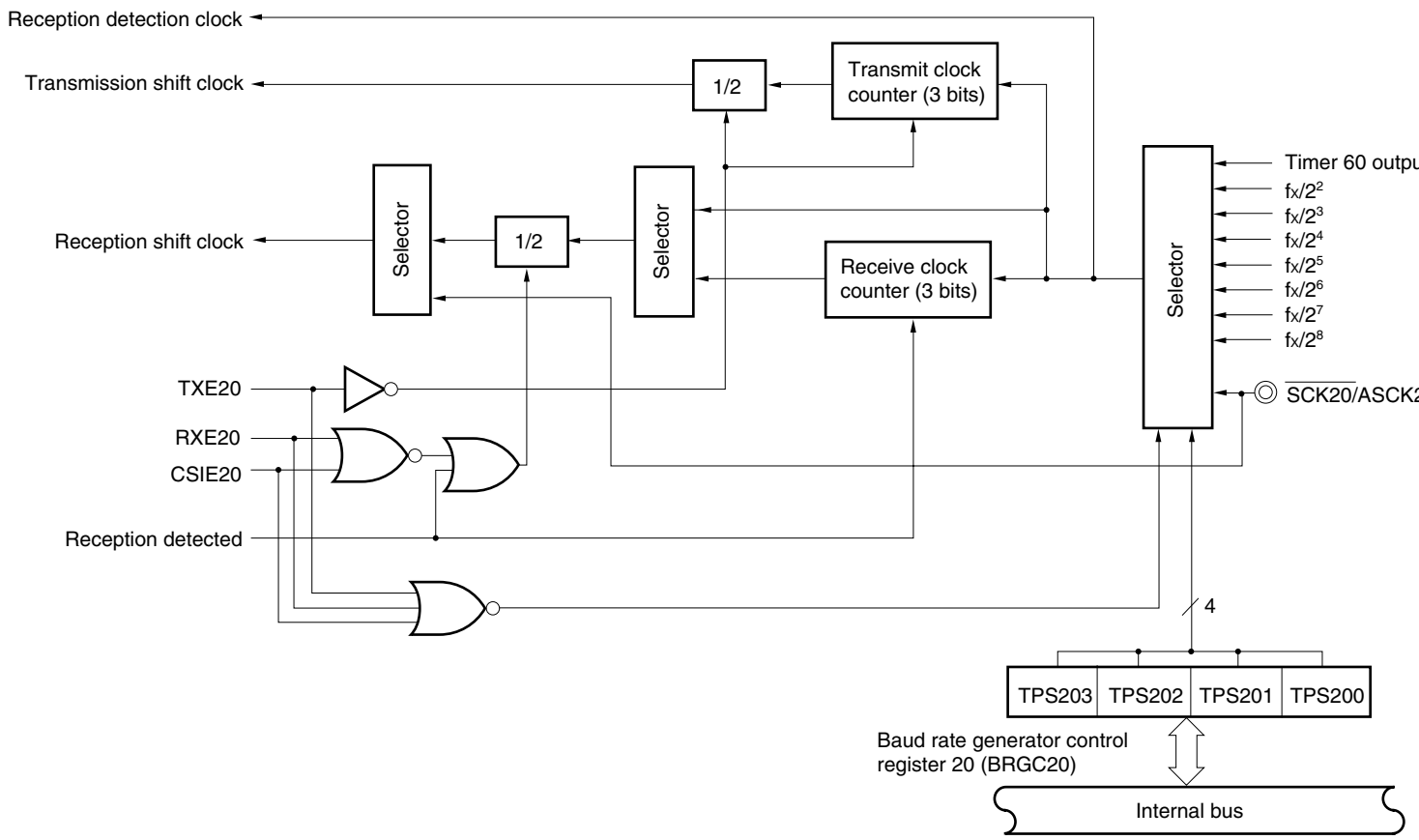
Item	Configuration
Registers	Transmit shift register 20 (TXS20) Receive shift register 20 (RXS20) Receive buffer register 20 (RXB20)
Control registers	Serial operation mode register 20 (CSIM20) Asynchronous serial interface mode register 20 (ASIM20) Asynchronous serial interface status register 20 (ASIS20) Baud rate generator control register 20 (BRGC20) Port mode register 2 (PM2) Port register 2 (P2)

★ **Figure 10-1. Block Diagram of Serial Interface 20**



Note See **Figure 10-2** for the configuration of the baud rate generator.

Figure 10-2. Block Diagram of Baud Rate Generator 20



(1) Transmit shift register 20 (TXS20)

TXS20 is a register in which transmit data is prepared. The transmit data is output from TXS20 bit-serially.

When the data length is seven bits, bits 0 to 6 of the data in TXS20 will be transmit data. Writing data to TXS20 triggers transmission.

TXS20 can be written with an 8-bit memory manipulation instruction, but cannot be read.

$\overline{\text{RESET}}$ input sets TXS20 to FFH.

Caution Do not write to TXS20 during transmission.

TXS20 and receive buffer register 20 (RXB20) are mapped at the same address, so that any attempt to read from TXS20 results in a value being read from RXB20.

(2) Receive shift register 20 (RXS20)

RXS20 is a register in which serial data, received at the RxD20 pin, is converted to parallel data. Once one entire byte has been received, RXS20 feeds the receive data to receive buffer register 20 (RXB20).

RXS20 cannot be manipulated directly by a program.

(3) Receive buffer register 20 (RXB20)

RXB20 holds a reception data. A new reception data is transferred from receive shift register 20 (RXS20) every 1-byte data reception.

When the data length is seven bits, the receive data is sent to bits 0 to 6 of RXB20, in which the MSB is always fixed to 0.

RXB20 can be read with an 8-bit memory manipulation instruction, but cannot be written.

$\overline{\text{RESET}}$ input makes RXB20 undefined.

Caution RXB20 and transmit shift register 20 (TXS20) are mapped at the same address, so that any attempt to write to RXB20 results in a value being written to TXS20.

(4) Transmit controller

The transmit controller controls transmission. For example, it adds start, parity, and stop bits to the data in transmit shift register 20 (TXS20), according to the setting of asynchronous serial interface mode register 20 (ASIM20).

(5) Receive controller

The receive controller controls reception according to the setting of asynchronous serial interface mode register 20 (ASIM20). It also checks for errors, such as parity errors, during reception. If an error is detected, asynchronous serial interface status register 20 (ASIS20) is set according to the status of the error.

10.3 Registers Controlling Serial Interface 20

Serial interface 20 is controlled by the following six registers.

- Serial operation mode register 20 (CSIM20)
- Asynchronous serial interface mode register 20 (ASIM20)
- Asynchronous serial interface status register 20 (ASIS20)
- Baud rate generator control register 20 (BRGC20)
- Port mode register 2 (PM2)
- Port register 2 (P2)

(1) Serial operation mode register 20 (CSIM20)

CSIM20 is set when serial interface 20 is used in 3-wire serial I/O mode.

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears CSIM20 to 00H.

Figure 10-3. Format of Serial Operation Mode Register 20

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	0	0	0	DAP20	DIR20	CSCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control
0	Operation disabled
1	Operation enabled

DAP20	3-wire serial I/O mode data phase selection
0	Outputs at the falling edge of $\overline{\text{SCK20}}$.
1	Outputs at the rising edge of $\overline{\text{SCK20}}$.

DIR20	First-bit specification
0	MSB
1	LSB

CSCK20	3-wire serial I/O mode clock selection
0	External clock input to the $\overline{\text{SCK20}}$ pin
1	Output of the dedicated baud rate generator

CKP20	3-wire serial I/O mode clock phase selection
0	Clock is active low, and $\overline{\text{SCK20}}$ is at high level in the idle state.
1	Clock is active high, and $\overline{\text{SCK20}}$ is at low level in the idle state.

Cautions 1. Bits 4 to 6 must all be set to 0.

2. CSIM20 must be cleared to 00H when UART mode is selected.

★ **3. Switching operation modes must be performed after the serial transmit/receive operation is stopped.**

(2) Asynchronous serial interface mode register 20 (ASIM20)

ASIM20 is set when serial interface 20 is used in asynchronous serial interface mode.

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIM20 to 00H.

Figure 10-4. Format of Asynchronous Serial Interface Mode Register 20

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stop
1	Transmit operation enable

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

PS201	PS200	Parity bit specification
0	0	No parity
0	1	Always add 0 parity at transmission. Parity check is not performed at reception (No parity error is generated).
1	0	Odd parity
1	1	Even parity

CL20	Transmit data character length specification
0	7 bits
1	8 bits

SL20	Transmit data stop bit length
0	1 bit
1	2 bits

Cautions 1. Bits 0 and 1 must both be set to 0.

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

3. Switching operation modes must be performed after the serial transmit/receive operation is stopped.

Table 10-2. Serial Interface 20 Operation Mode Settings

(1) Operation stop mode

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

(2) 3-wire serial I/O mode

ASIM20		CSIM20			PM22	P22	PM21	P21	PM20	P20	First Bit	Shift Clock	P22/SI20/ RxD20 Pin Function	P21/SO20/ TxD20 Pin Function	P20/$\overline{\text{SCK20}}$/ ASCK20 Pin Function
TXE20	RXE20	CSIE20	DIR20	CSCK20											
0	0	1	0	0	×Note 2	×Note 2	0	1	1	×	MSB	External clock	SI20Note 2	SO20 (CMOS output)	$\overline{\text{SCK20}}$ input
				1					0	1		Internal clock			$\overline{\text{SCK20}}$ output
		1	1	0					1	×	LSB	External clock			$\overline{\text{SCK20}}$ input
				1					0	1		Internal clock			$\overline{\text{SCK20}}$ output
Other than above											Setting prohibited				

(3) Asynchronous serial interface mode

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

Notes 1. These pins can be used for port functions.

2. When only transmission is used, this pin can be used as P22 (CMOS input/output).

Remark ×: Don't care.

(3) Asynchronous serial interface status register 20 (ASIS20)

ASIS20 indicates the type of a reception error, if it occurs while asynchronous serial interface mode is set.

ASIS20 is read with a 1-bit or 8-bit memory manipulation instruction.

The contents of ASIS20 are undefined in 3-wire serial I/O mode.

$\overline{\text{RESET}}$ input clears ASIS20 to 00H.

Figure 10-5. Format of Asynchronous Serial Interface Status Register 20

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ASIS20	0	0	0	0	0	PE20	FE20	OVE20	FF71H	00H	R

PE20	Parity error flag
0	No parity error has occurred.
1	A parity error has occurred (parity mismatch in transmit parity and receive parity).

FE20	Flaming error flag
0	No framing error has occurred.
1	A framing error has occurred (no stop bit detected).[Note 1]

OVE20	Overrun error flag
0	No overrun error has occurred.
1	An overrun error has occurred[Note 2]. (Before data was read from the receive buffer register, the subsequent receive operation was completed.)

Notes 1. Even when the stop bit length is set to 2 bits by setting bit 2 (SL20) of asynchronous serial interface mode register 20 (ASIM20), the stop bit detection at reception is performed with 1 bit.

2. Until receive buffer register 20 (RXB20) is read when an overrun error occurs, an overrun error continues to occur.

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

BRGC20 is used to specify the serial clock for serial interface 20.

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

$\overline{\text{RESET}}$ input clears BRGC20 to 00H.

Figure 10-6. Format of Baud Rate Generator Control Register 20

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC20	TPS203	TPS202	TPS201	TPS200	0	0	0	0	FF73H	00H	R/W

TPS203	TPS202	TPS201	TPS200	Selection of baud rate generator source clock	n
0	0	0	0	$f_X/2^2$ (1.25 MHz)	2
0	0	0	1	$f_X/2^3$ (625 kHz)	3
0	0	1	0	$f_X/2^4$ (313 kHz)	4
0	0	1	1	$f_X/2^5$ (156 kHz)	5
0	1	0	0	$f_X/2^6$ (78.1 kHz)	6
0	1	0	1	$f_X/2^7$ (39.1 kHz)	7
0	1	1	0	$f_X/2^8$ (19.5 kHz)	8
0	1	1	1	Timer 60 output	–
1	0	0	0	External clock input to the ASCK20 pin[Note]	–
Other than above				Setting prohibited	

Note An external clock can be used only in UART mode.

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

2. Be sure to set timer 60 to square-wave output mode when timer 60 output is selected.

3. When the external input clock is selected, set port mode register 2 (PM2) to input mode.

Remarks 1. f_X: System clock oscillation frequency

2. n: Value determined by setting TPS200 through TPS203 ($2 \leq n \leq 8$)

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

UART baud rate transmit/receive clock is generated from any of the following three signals.

- A divided system clock signal
- A square-wave signal output from timer 60
- A divided ASCK20 pin clock input signal.

★ Serial clock in 3-wire serial I/O mode is generated from any of the following two signals.

- A divided system clock signal
- A square-wave signal output from timer 60

Remark When using the clock input from the $\overline{SCK20}$ pin, set with bit 1 (CSCK20) of serial operation mode register 20 (CSIM20).

(a) Generation of UART baud rate transmit/receive clock form system clock

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

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

f_X: System clock oscillation frequency

n: Value determined by values of TPS200 through TPS203 as shown in Figure 10-6 ($2 \leq n \leq 8$)

Table 10-3. Example of Relationship Between System Clock and Baud Rate

Baud Rate (bps)	n	BRGC20 Set Value	Error (%)	
			f_X = 5.0 MHz	f_X = 4.9152 MHz
1,200	8	60H	1.73	0
2,400	7	50H		
4,800	6	40H		
9,600	5	30H		
19,200	4	20H		
38,400	3	10H		
76,800	2	00H		

(b) Generation of UART baud rate transmit/receive clock by square-wave output from timer 60

The transmit/receive clock is generated by dividing the square wave output from timer 60. The baud rate generated by square-wave output from timer 60 is estimated by using the following expression.

$$[\text{Baud rate}] = \frac{f_{TO60}}{16} \text{ [bps]}$$

f_{TO60}: Frequency of timer 60 square-wave output

Table 10-4. Relationship Between Timer 60 Square-Wave Output Frequency and Baud Rate (When BRGC20 Is Set to 70H)

Baud Rate (bps)	Timer 60 Square-Wave Output 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

(c) Generation of baud rate transmit/receive clock from external clock input from ASCK20 pin

The transmit/receive clock is generated by scaling the clock input from the ASCK20 pin. The baud rate of a clock generated from the clock input from the ASCK20 pin is estimated by using the following expression.

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

f_{ASCK}: Frequency of clock input from the ASCK20 pin

Table 10-5. Relationship Between ASCK20 Pin Input Frequency and Baud Rate (When BRGC20 Is Set to 80H)

Baud Rate (bps)	ASCK20 Pin Input Frequency (kHz)
75	1.2
150	2.4
300	4.8
600	9.6
1,200	19.2
2,400	38.4
4,800	76.8
9,600	153.6
19,200	307.2
31,250	500.0
38,400	614.4

★ **(d) Generation of serial clock from system clock in 3-wire serial I/O mode**

The serial clock is generated by dividing the system clock. The frequency of the serial clock can be obtained by the following expression. If the serial clock is externally input to the $\overline{SCK20}$ pin, it is unnecessary to set BRGC20.

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

f_X: System clock oscillation frequency

n: Values in Figure 10-6 determined by the settings of TPS200 to TPS203 ($2 \leq n \leq 8$)

★ **(e) Generation of serial clock from square-wave output of timer 60 in 3-wire serial I/O mode**

The square-wave output of timer 60 is generated as the serial clock. The frequency of the serial clock can be obtained by the following expression. If the serial clock is externally input to the $\overline{SCK20}$ pin, it is unnecessary to set BRGC20.

$$\text{Serial clock frequency} = \frac{f_{TO60}}{2} \text{ [Hz]}$$

f_{TO60}: Frequency of timer 60 square-wave output

10.4 Operation of Serial Interface 20

Serial interface 20 provides the following three types of modes.

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

10.4.1 Operation stop mode

In operation stop mode, serial transfer is not executed; therefore, the power consumption can be reduced. The P20/$\overline{\text{SCK20}}$/ASCK20, P21/SO20/TxD20, and P22/SI20/RxD20 pins can be used as normal I/O ports.

(1) Register setting

Operation stop mode is set by serial operation mode register 20 (CSIM20) and asynchronous serial interface mode register 20 (ASIM20).

(a) Serial operation mode register 20 (CSIM20)

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears CSIM20 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	0	0	0	DAP20	DIR20	CSCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control
0	Operation disabled
1	Operation enabled

Caution Bits 4 to 6 must all be set to 0.

(b) Asynchronous serial interface mode register 20 (ASIM20)

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIM20 to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stop
1	Transmit operation enable

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

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

10.4.2 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 communication at a desired baud rate from many options. In addition, the baud rate can also be defined by dividing the timer 60 output or the clock input to the ASCK20 pin.

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

(1) Register setting

UART mode is set by serial operation mode register 20 (CSIM20), asynchronous serial interface mode register 20 (ASIM20), asynchronous serial interface status register 20 (ASIS20), baud rate generator control register 20 (BRGC20), port mode register 2 (PM2), and port register 2 (P2).

(a) Serial operation mode register 20 (CSIM20)

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears CSIM20 to 00H.

Set CSIM20 to 00H when UART mode is selected.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	0	0	0	DAP20	DIR20	CSCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control
0	Operation disabled
1	Operation enabled

DAP20	3-wire serial I/O mode data phase selection
0	Outputs at the falling edge of $\overline{\text{SCK20}}$.
1	Outputs at the rising edge of $\overline{\text{SCK20}}$.

DIR20	First-bit specification
0	MSB
1	LSB

CSCK20	3-wire serial I/O mode clock selection
0	External clock input to the $\overline{\text{SCK20}}$ pin
1	Output of the dedicated baud rate generator

CKP20	3-wire serial I/O mode clock phase selection
0	Clock is active low, and $\overline{\text{SCK20}}$ is high level in the idle state.
1	Clock is active high, and $\overline{\text{SCK20}}$ is low level in the idle state.

Cautions 1. Bits 4 to 6 must all be set to 0.

★ **2. Switching operation modes must be performed after the serial transmit/receive operation is stopped.**

(b) Asynchronous serial interface mode register 20 (ASIM20)

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIM20 to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stopped
1	Transmit operation enabled

RXE20	Receive operation control
0	Receive operation stopped
1	Receive operation enabled

PS201	PS200	Parity bit specification
0	0	No parity
0	1	Always add 0 parity at transmission. Parity check is not performed at reception. (No parity error is generated.)
1	0	Odd parity
1	1	Even parity

CL20	Character length specification
0	7 bits
1	8 bits

SL20	Transmit data stop bit length specification
0	1 bit
1	2 bits

Cautions 1. Bits 0 and 1 must both be set to 0.

2. Switching operation modes must be performed after the serial transmit/receive operation is stopped.

(c) Asynchronous serial interface status register 20 (ASIS20)

ASIS20 is read with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIS20 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
ASIS20	0	0	0	0	0	PE20	FE20	OVE20	FF71H	00H	R

PE20	Parity error flag
0	Parity error not generated
1	Parity error generated (when the transmit parity and receive parity do not match)

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

OVE20	Overrun error flag
0	Overrun error not generated
1	Overrun error 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 (SL20) of asynchronous serial interface mode register 20 (ASIM20), the stop bit detection at reception is performed with 1 bit.

2. Until receive buffer register 20 (RXB20) is read when an overrun error occurs, an overrun error continues to occur.

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

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

$\overline{\text{RESET}}$ input clears BRGC20 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC20	TPS203	TPS202	TPS201	TPS200	0	0	0	0	FF73H	00H	R/W

TPS203	TPS202	TPS201	TPS200	Selection of baud rate generator source clock	n
0	0	0	0	$f_X/2^2$ (1.25 MHz)	2
0	0	0	1	$f_X/2^3$ (625 kHz)	3
0	0	1	0	$f_X/2^4$ (313 kHz)	4
0	0	1	1	$f_X/2^5$ (156 kHz)	5
0	1	0	0	$f_X/2^6$ (78.1 kHz)	6
0	1	0	1	$f_X/2^7$ (39.1 kHz)	7
0	1	1	0	$f_X/2^8$ (19.5 kHz)	8
0	1	1	1	Timer 60 output	–
1	0	0	0	External clock input to ASCK20 pin[Note]	–
Other than above				Setting prohibited	

Note An external clock can be used only in UART mode.

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

2. Be sure to set timer 60 to square-wave output mode when timer 60 output is selected.

3. When the external input clock is selected, set port mode register 2 (PM2) to input mode.

Remarks 1. fx: System clock oscillation frequency

2. n: Value determined by setting TPS200 through TPS203 ($2 \leq n \leq 8$)

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

Select which of the following baud rate transmit/receive clocks are to be generated.

- A divided system clock signal
- A square-wave signal output from timer 60
- A divided ASCK20 pin clock input signal.

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

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

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

fx: System clock oscillation frequency

n: Value determined by setting TPS200 through TPS203 as shown in the above table ($2 \leq n \leq 8$)

Table 10-6. Example of Relationship Between System Clock and Baud Rate

Baud Rate (bps)	n	BRGC20 Set Value	Error (%)	
			fx = 5.0 MHz	fx = 4.9152 MHz
1,200	8	60H	1.73	0
2,400	7	50H		
4,800	6	40H		
9,600	5	30H		
19,200	4	20H		
38,400	3	10H		
76,800	2	00H		

(ii) Generation of baud rate transmit/receive clock by square-wave output from timer 60

The transmit/receive clock is generated by dividing the square wave output from timer 60. The baud rate generated by square-wave output from timer 60 is estimated by using the following expression.

$$[\text{Baud rate}] = \frac{f_{TO60}}{16} \text{ [bps]}$$

f_{TO60}: Frequency of timer 60 square-wave output

Table 10-7. Relationship Between Timer 60 Square-Wave Output Frequency and Baud Rate (When BRGC20 Is Set to 70H)

Baud Rate (bps)	Timer 60 Square-Wave Output 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

(iii) Generation of baud rate transmit/receive clock from external clock input from ASCK20 pin

The transmit/receive clock is generated by scaling the clock input from the ASCK20 pin. The baud rate of the clock generated from the clock input from the ASCK20 pin is estimated by using the following expression.

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

f_{ASCK}: Frequency of clock input from the ASCK20 pin

Table 10-8. Relationship Between ASCK20 Pin Input Frequency and Baud Rate (When BRGC20 Is Set to 80H)

Baud Rate (bps)	ASCK20 Pin Input Frequency (kHz)
75	1.2
150	2.4
300	4.8
600	9.6
1,200	19.2
2,400	38.4
4,800	76.8
9,600	153.6
19,200	307.2
31,250	500.0
38,400	614.4

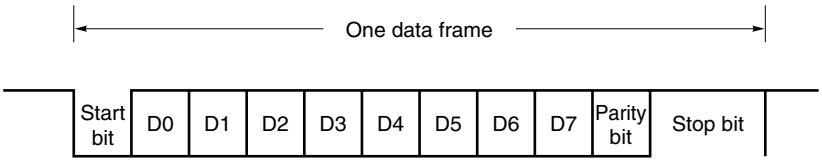
(2) Communication operation

(a) Data format

The transmit/receive data format is as shown in Figure 10-7. One data frame consists of a start bit, character bits, a parity bit, and stop bit(s).

The specification of the character bit length in one data frame, parity selection, and specification of the stop bit length is carried out with asynchronous serial interface mode register 20 (ASIM20).

Figure 10-7. Format of Asynchronous Serial Interface Transmit/Receive Data



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

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

The serial transfer rate is selected by ASIM20 and baud rate generator control register 20 (BRGC20).

If a serial data receive error is generated, the receive error contents can be determined by reading the status of asynchronous serial interface status register 20 (ASIS20).

★ **(b) Parity types and operation**

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

(i) Even parity

- **At transmission**

  The parity bit is determined so that the number of character bits with a value of "1" in the transmit data including the parity bit may be even. The parity bit value should be as follows.

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

- **At reception**

  The number of character bits with a value of "1" in the receive data including the parity bit is counted, and if the number is odd, a parity error is generated.

(ii) Odd parity

- **At transmission**

  Conversely to even parity, the parity bit is determined so that the number of character bits with a value of "1" in the transmit data including the parity bit may be odd. The parity bit value should be as follows.

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

- **At reception**

  The number of character bits with a value of "1" in the receive data including the 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.

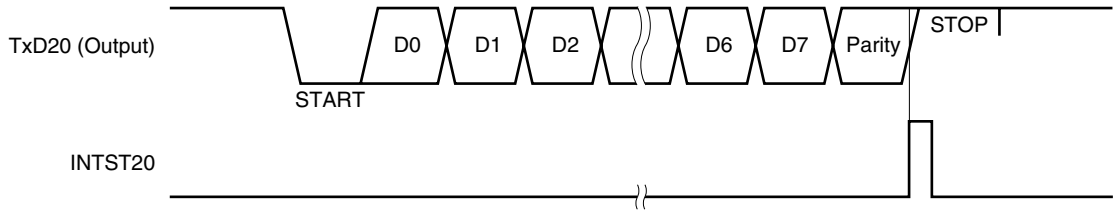
(c) Transmission

★ A transmit operation is enabled by setting bit 7 (TXE20) of asynchronous serial interface mode register 20 (ASIM20) to 1, and started by writing transmit data to transmit shift register 20 (TXS20). The start bit, parity bit, and stop bit(s) are added automatically.

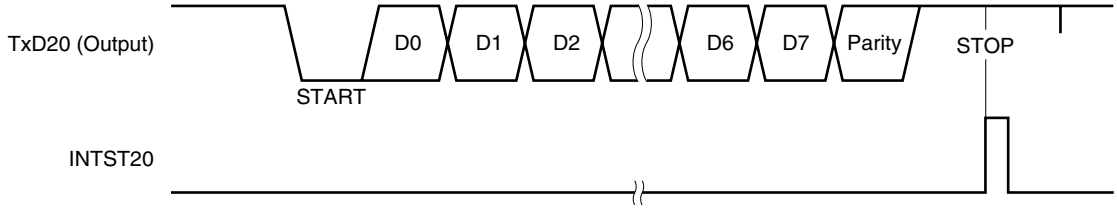
When the transmit operation starts, the data in TXS20 is shifted out, and when TXS20 is empty, a transmission completion interrupt (INTST20) is generated.

Figure 10-8. Asynchronous Serial Interface Transmission Completion Interrupt Timing

(a) Stop bit length: 1



(b) Stop bit length: 2



Caution Do not rewrite asynchronous serial interface mode register 20 (ASIM20) during a transmit operation. If the ASIM20 register is rewritten during transmission, subsequent transmission may not be performed (the normal state is restored by $\overline{\text{RESET}}$ input).

(d) Reception

★ A receive operation is performed via level detection.

When bit 6 (RXE20) of asynchronous serial interface mode register 20 (ASIM20) is set to 1, a receive operation is enabled and sampling of the RxD20 pin input is performed.

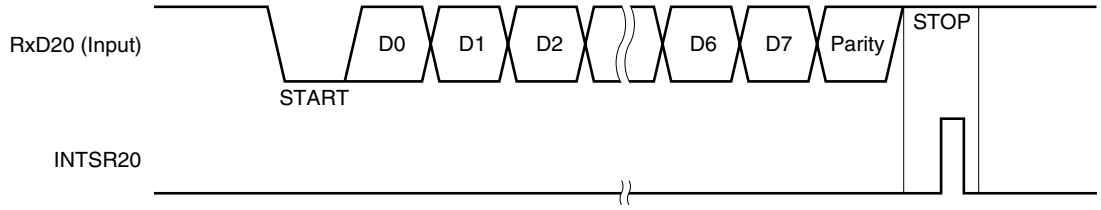
RxD20 pin input sampling is performed using the serial clock specified by ASIM20.

When the RxD20 pin input becomes low, the 3-bit counter starts counting, and at the time when half the time determined by the specified baud rate has passed, the data sampling start timing signal is output. If the RxD20 pin input sampled again as a result of this start timing signal is low, it is identified as a start bit, the 3-bit counter is initialized and starts counting, and data sampling is performed. When character data, a parity bit, and one stop bit are detected after the start bit, reception of one frame of data ends.

When one frame of data has been received, the receive data in the shift register is transferred to receive buffer register 20 (RXB20), and a reception completion interrupt (INTSR20) is generated.

If the RXE20 bit is reset to 0 during the receive operation, the receive operation is stopped immediately. In this case, the contents of RXB20 and asynchronous serial interface status register 20 (ASIS20) are not changed, and INTSR20 is not generated.

Figure 10-9. Asynchronous Serial Interface Reception Completion Interrupt Timing



★ **Caution If a receive operation is enabled when the RxD20 pin input is low level, a receive operation is immediately started. Therefore, be sure to enable a receive operation after making the RxD20 pin input high level.**

(e) Receive errors

The following three errors may occur during a receive operation: a parity error, a framing error, and an overrun error. After data reception, an error flag is set in asynchronous serial interface status register 20 (ASIS20). Receive error causes are shown in Table 10-9.

It is possible to determine what kind of error was generated during reception by reading the contents of ASIS20 in the reception error interrupt servicing (see **Figure 10-5**).

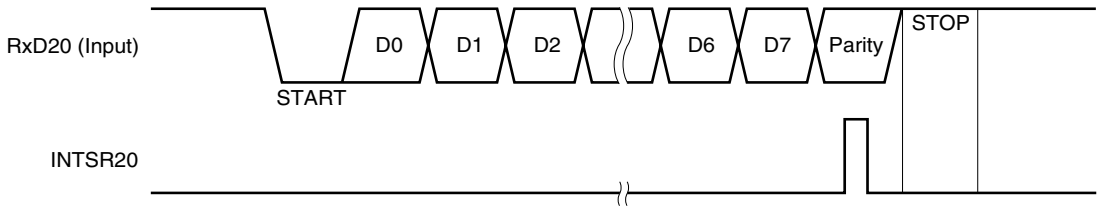
The contents of ASIS20 are reset to 0 by reading receive buffer register 20 (RXB20) or receiving the next data (if there is an error in the next data, the corresponding error flag is set).

Table 10-9. Receive Error Causes

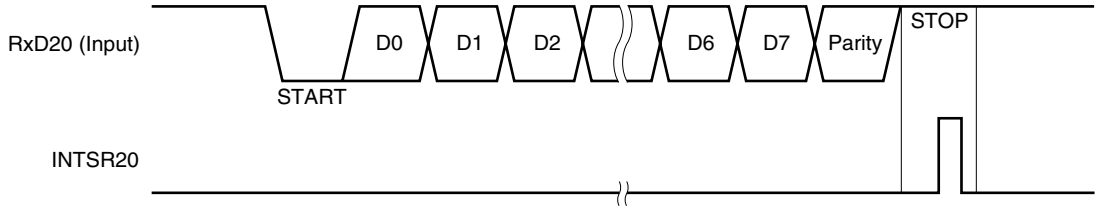
Receive Errors	Cause
Parity error	Transmission-time parity and reception data parity do not match.
Framing error	Stop bit not detected
Overrun error	Reception of next data is completed before data is read from the receive buffer register.

Figure 10-10. Receive Error Timing

(a) Parity error generated



(b) Framing error or overrun error generated



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

2. Be sure to read receive buffer register 20 (RXB20) even if a receive error is generated. If RXB20 is not read, an overrun error will be generated when the next data is received, and the receive error state will continue indefinitely.

(f) Reading receive data

When the reception completion interrupt (INTSR20) occurs, receive data can be read by reading the value of receive buffer register 20 (RXB20).

To read the receive data stored in receive buffer register 20 (RXB20), read while reception is enabled (RXE20 = 1).

Remark However, if it is necessary to read receive data after reception has stopped (RXE20 = 0), read using either of the following methods.

(a) Read after setting RXE20 = 0 after waiting for one cycle or more of the source clock selected by BRGC20.
(b) Read after bit 2 (DIR20) of serial operation mode register 20 (CSIM20) is set (1).

Program example of (a) (BRGC20 = 00H (source clock = fx/2))

```
INTREX:                 ;<Reception completion interrupt routine>
       NOP              ;2 clocks
       CLR1 RXE20       ;Reception stopped
       MOV  A, RXB20    ;Read receive data
```

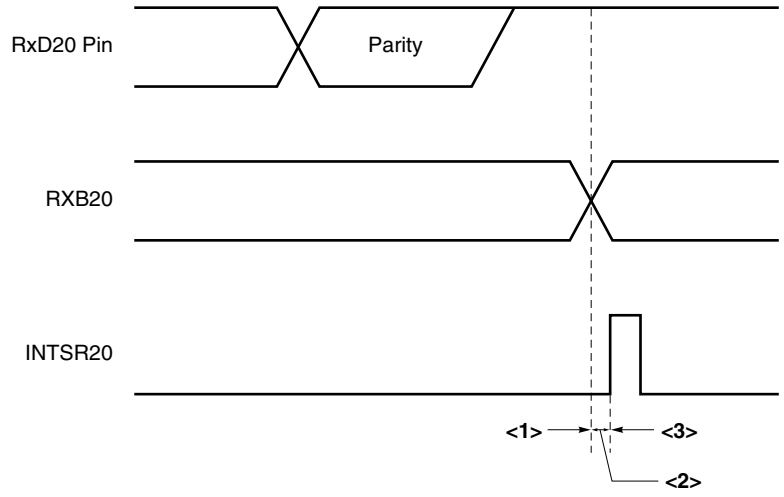
Program example of (b)

```
INTRXE:                 ;<Reception completion interrupt routine>
       SET1 CSIM20.2    ;DIR20 flag is set to LSB first
       CLR1 RXE20       ;Reception stopped
       MOV  A, RXB20    ;Read receive data
```

(3) Cautions related to UART mode

(a) When bit 7 (TXE20) of asynchronous serial interface mode register 20 (ASIM20) is cleared during transmission, be sure to set transmit shift register 20 (TXS20) to FFH, then set TXE20 to 1 before executing the next transmission.

(b) When bit 6 (RXE20) of asynchronous serial interface mode register 20 (ASIM20) is cleared during reception, receive buffer register 20 (RXB20) and the reception completion interrupt (INTSR20) are as follows.



When RXE20 is set to 0 at a time indicated by **<1>**, RXB20 holds the previous data and INTSR20 is not generated.
When RXE20 is set to 0 at a time indicated by **<2>**, RXB20 renews the data and INTSR20 is not generated.
When RXE20 is set to 0 at a time indicated by **<3>**, RXB20 renews the data and INTSR20 is generated.

10.4.3 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., that incorporate a conventional clocked synchronous serial interface, such as the 75XL Series, 78K Series, 17K Series, etc.

Communication is performed using three lines: the serial clock ($\overline{SCK20}$), serial output (SO20), and serial input (SI20).

(1) Register setting

3-wire serial I/O mode settings are performed using serial operation mode register 20 (CSIM20), asynchronous serial interface mode register 20 (ASIM20), and baud rate generator control register 20 (BRGC20).

(a) Serial operation mode register 20 (CSIM20)

CSIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{RESET}$ input clears CSIM20 to 00H.

Symbol	<7>	6	5	4	3	2	1	0	Address	After reset	R/W
CSIM20	CSIE20	0	0	0	DAP20	DIR20	CSCK20	CKP20	FF72H	00H	R/W

CSIE20	3-wire serial I/O mode operation control
0	Operation disabled
1	Operation enabled

DAP20	3-wire serial I/O mode data phase selection
0	Outputs at the falling edge of $\overline{SCK20}$.
1	Outputs at the rising edge of $\overline{SCK20}$.

DIR20	First-bit specification
0	MSB
1	LSB

CSCK20	3-wire serial I/O mode clock selection
0	External clock input to the $\overline{SCK20}$ pin
1	Output of the dedicated baud rate generator

CKP20	3-wire serial I/O mode clock phase selection
0	Clock is active low, and $\overline{SCK20}$ is at high level in the idle state.
1	Clock is active high, and $\overline{SCK20}$ is at low level in the idle state.

Cautions 1. Bits 4 to 6 must all be set to 0.

2. When the external input clock is selected, set port mode register 2 (PM2) to input mode.

★ **3. Switching operation modes must be performed after the serial transmit/receive operation is stopped.**

(b) Asynchronous serial interface mode register 20 (ASIM20)

ASIM20 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears ASIM20 to 00H.

When 3-wire serial I/O mode is selected, ASIM20 must be set to 00H.

Symbol	<7>	<6>	5	4	3	2	1	0	Address	After reset	R/W
ASIM20	TXE20	RXE20	PS201	PS200	CL20	SL20	0	0	FF70H	00H	R/W

TXE20	Transmit operation control
0	Transmit operation stop
1	Transmit operation enable

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

PS201	PS200	Parity Bit specification
0	0	No parity
0	1	Always add 0 parity at transmission. Parity check is not performed at reception. (No parity error is generated.)
1	0	Odd parity
1	1	Even parity

CL20	Character length specification
0	7 bits
1	8 bits

SL20	Transmit data sop bit length specification
0	1 bit
1	2 bits

Cautions 1. Bits 0 and 1 must both be set to 0.

2. Switching operation modes must be performed after the serial transmit/receive operation is stopped.

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

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

$\overline{\text{RESET}}$ input clears BRGC20 to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
BRGC20	TPS203	TPS202	TPS201	TPS200	0	0	0	0	FF73H	00H	R/W

TPS203	TPS202	TPS201	TPS200	Selection of baud rate generator source clock	n
0	0	0	0	$f_X/2^2$ (1.25 MHz)	2
0	0	0	1	$f_X/2^3$ (625 kHz)	3
0	0	1	0	$f_X/2^4$ (313 kHz)	4
0	0	1	1	$f_X/2^5$ (156 kHz)	5
0	1	0	0	$f_X/2^6$ (78.1 kHz)	6
0	1	0	1	$f_X/2^7$ (39.1 kHz)	7
0	1	1	0	$f_X/2^8$ (19.5 kHz)	8
0	1	1	1	Timer 60 output	–
Other than above				Setting prohibited	

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

Remarks 1. fx: System clock oscillation frequency

2. n: Value determined by setting TPS200 through TPS203 ($2 \leq n \leq 8$)

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

★ **(i) Generation of serial clock from system clock in 3-wire serial I/O mode**

The serial clock is generated by dividing the system clock. The frequency of the serial clock can be obtained by the following expression. If the serial clock is externally input to the $\overline{\text{SCK20}}$ pin, it is unnecessary to set BRGC20.

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

fx: System clock oscillation frequency

n: Values in above Table determined by the settings of TPS200 to TPS203 ($2 \leq n \leq 8$)

★ **(ii) Generation of serial clock from square-wave output of timer 60 in 3-wire serial I/O mode**

The square-wave output of timer 60 is generated as the serial clock. The frequency of the serial clock can be obtained by the following expression. If the serial clock is externally input to the $\overline{\text{SCK20}}$ pin, it is unnecessary to set BRGC20.

$$\text{Serial clock frequency} = \frac{f_{TO60}}{2} \text{ [Hz]}$$

f_{TO60}: Frequency of timer 60 square-wave output

(2) Communication operation

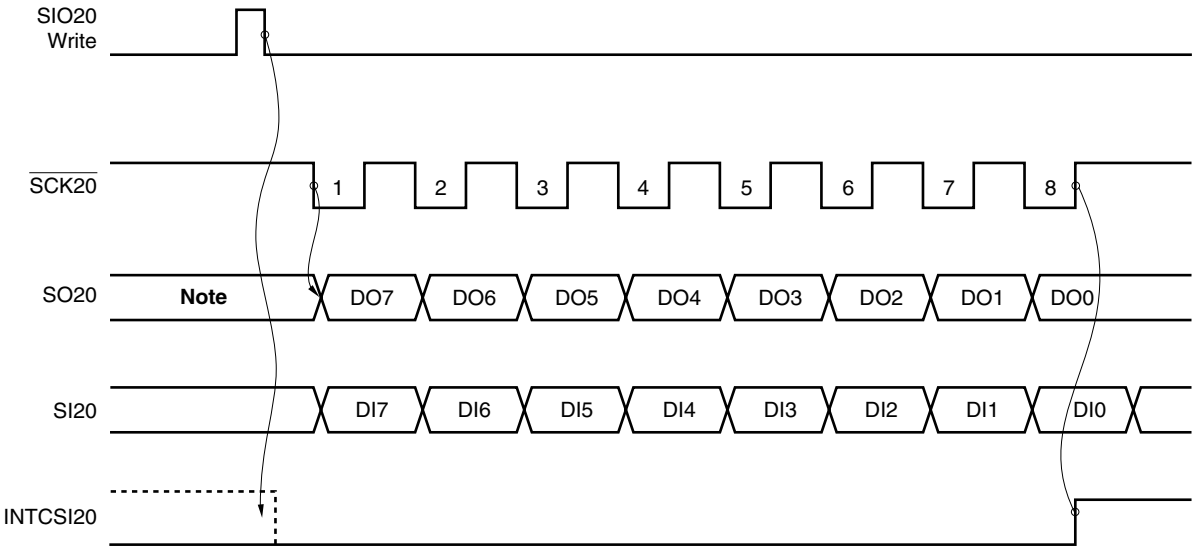
In 3-wire serial I/O mode, data transmission/reception is performed in 8-bit units. Data is transmitted/received bit by bit in synchronization with the serial clock.

Transmit shift register 20 (TXS20/SIO20) and receive shift register 20 (RXS20) shift operations are performed in synchronization with the fall of the serial clock ($\overline{SCK20}$). Then transmit data is held in the SO20 latch and output from the SO20 pin. Also, receive data input to the SI20 pin is latched in receive buffer register 20 (RXB20/SIO20) on the rise of $\overline{SCK20}$.

At the end of an 8-bit transfer, the operation of TXS20/SIO20 and RXS20 stops automatically, and the interrupt request signal (INTCSI20) is generated.

Figure 10-11. 3-Wire Serial I/O Mode Timing (1/5)

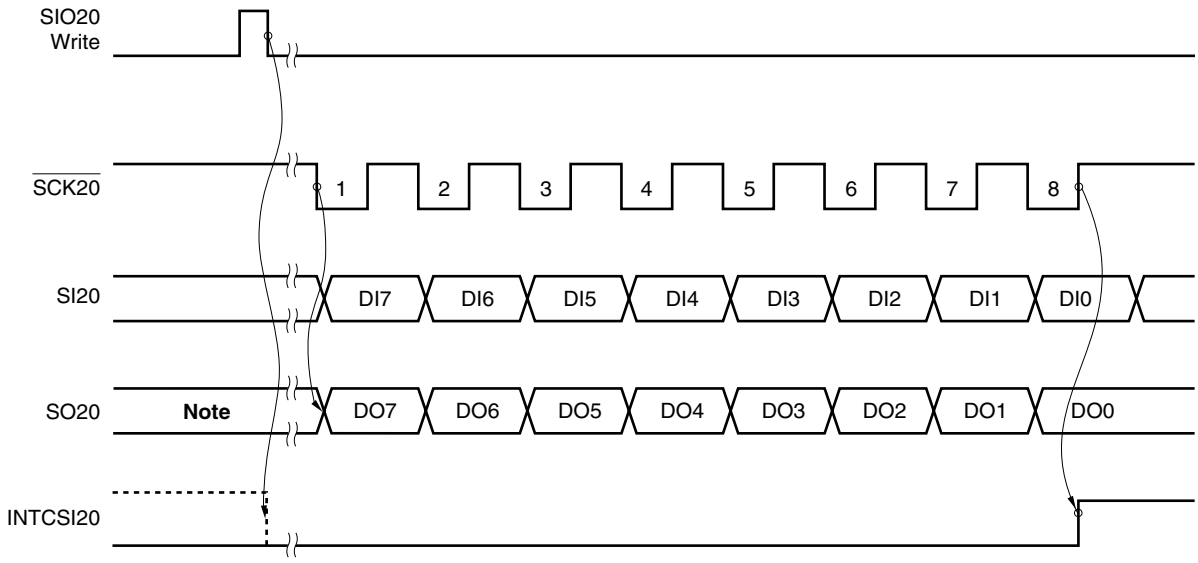
(i) Master operation timing (when DAP20 = 0, CKP20 = 0)



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

Figure 10-11. 3-Wire Serial I/O Mode Timing (2/5)

(ii) Slave operation timing (when DAP20 = 0, CKP20 = 0)



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

(iii) Master operation (when DAP20 = 0, CKP20 = 1)

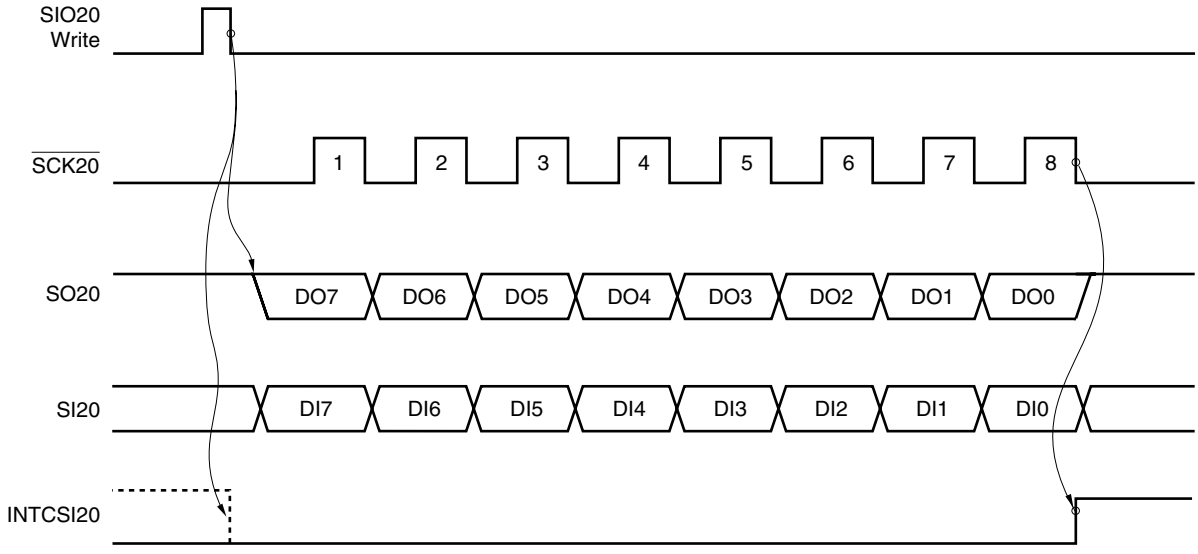
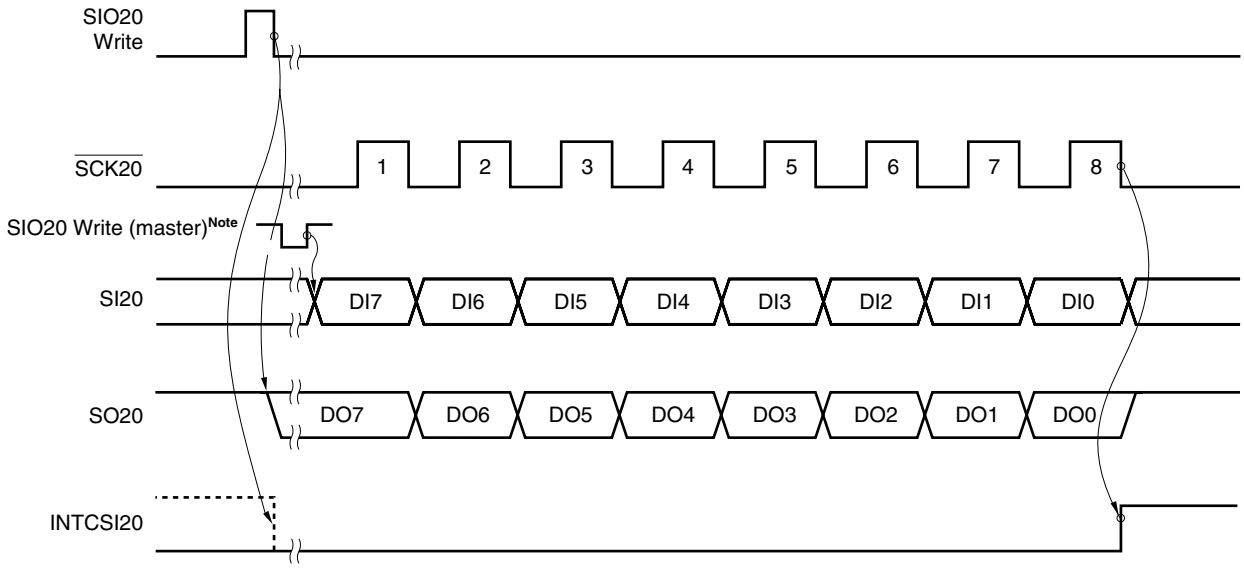


Figure 10-11. 3-Wire Serial I/O Mode Timing (3/5)

★ **(iv) Slave operation (when DAP20 = 0, CKP20 = 1)**



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

(v) Master operation (when DAP20 = 1, CKP20 = 0)

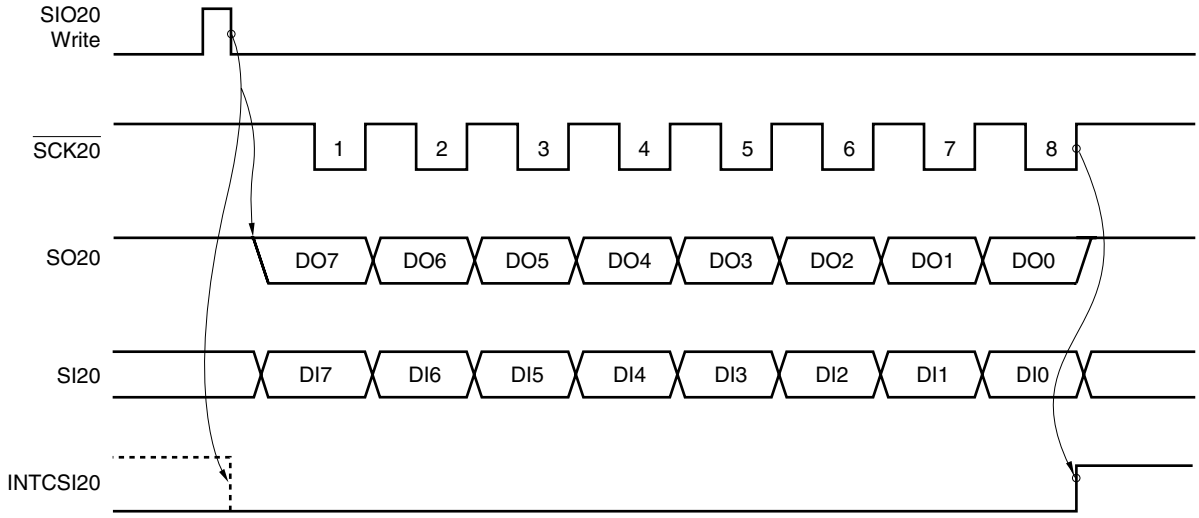
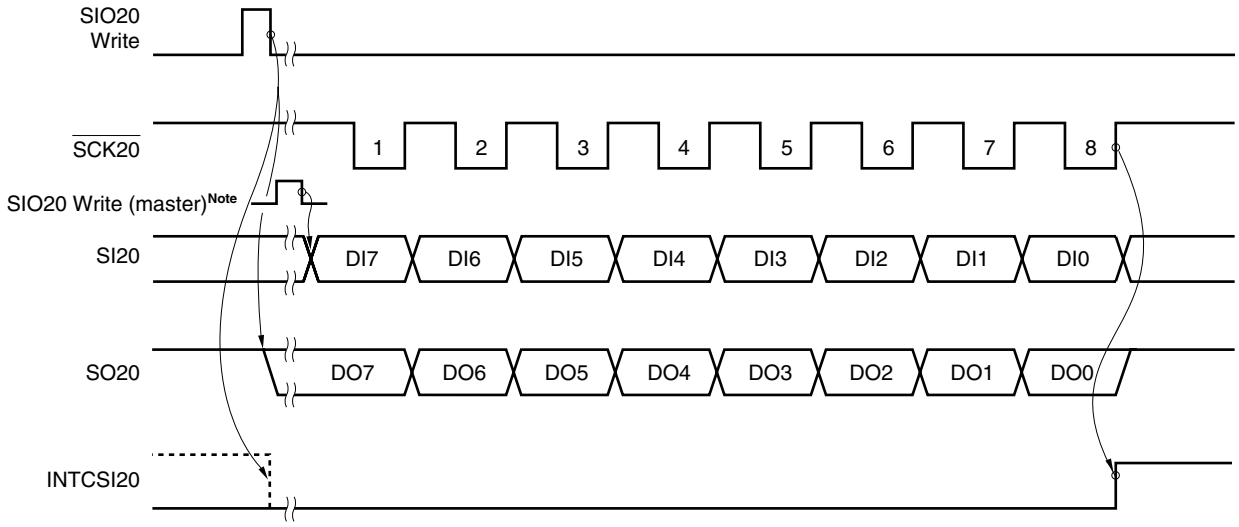


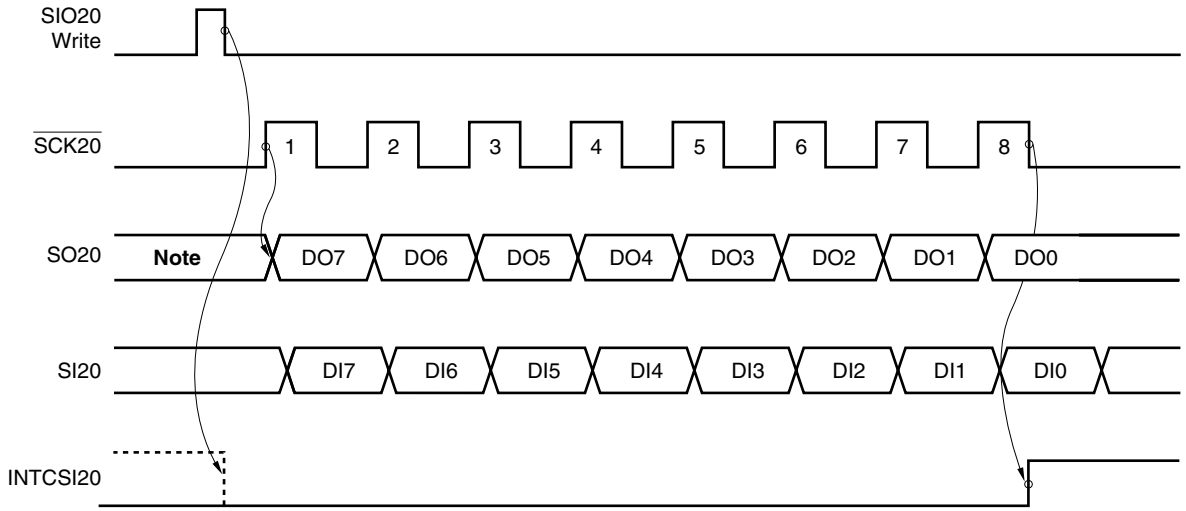
Figure 10-11. 3-Wire Serial I/O Mode Timing (4/5)

(vi) Slave operation (when DAP20 = 1, CKP20 = 0)



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

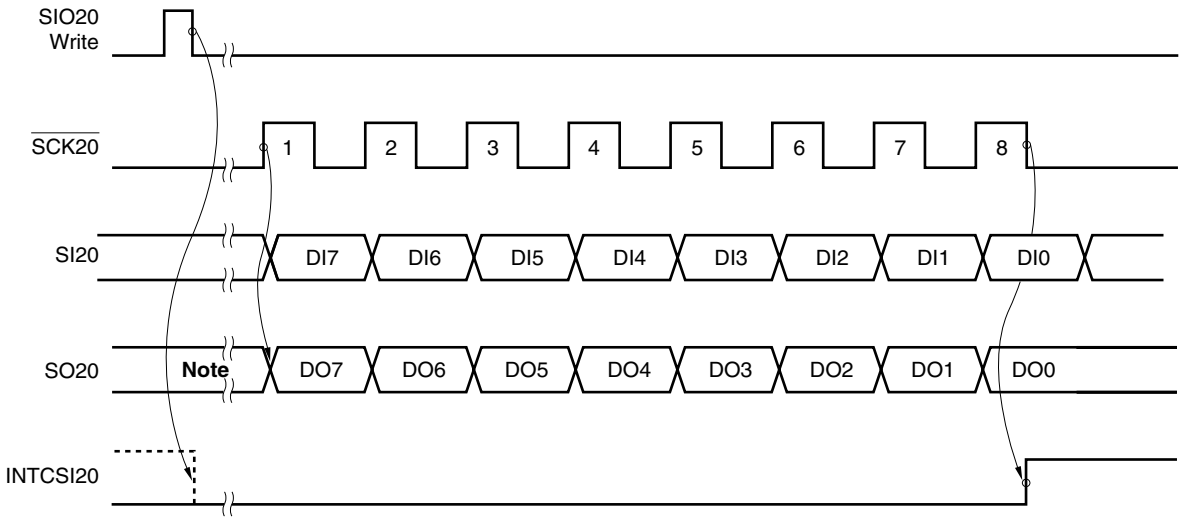
(vii) Master operation (when DAP20 = 1, CKP20 = 1)



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

Figure 10-11. 3-Wire Serial I/O Mode Timing (5/5)

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



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

(3) Transfer start

Serial transfer is started by setting transfer data to the transmit shift register (TXS20/SIO20) when the following two conditions are satisfied.

- Serial operation mode register 20 (CSIM20) bit 7 (CSIE20) = 1
- Internal serial clock is stopped or $\overline{\text{SCK20}}$ is un-active level after 8-bit serial transfer.

Caution If CSIE20 is set to 1 after data is written to TXS20/SIO20, transfer does not start.

The termination of 8-bit transfer stops the serial transfer automatically and generates the interrupt request signal (INTCSI20).

CHAPTER 11 POWER-ON-CLEAR CIRCUITS

11.1 Function of Power-on-Clear Circuit

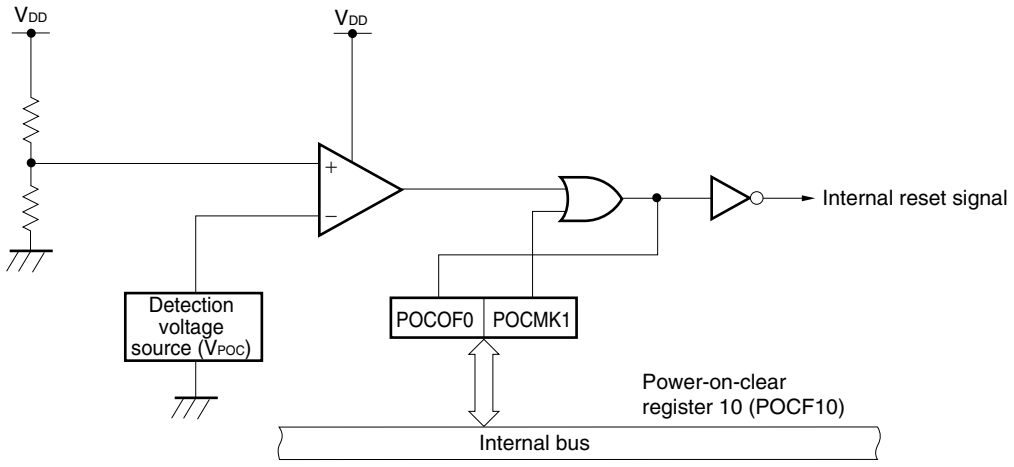
The power-on-clear (POC) circuits include the following function.

- Compares the detection voltage (V_{POC}) with the power supply voltage (V_{DD}) and generates an internal reset signal if $V_{DD} < V_{POC}$.
- Whether or not to use the POC circuit can be selected by means of software.
- This circuit can operate even in STOP mode.

11.2 Configuration of Power-on-Clear Circuit

Figure 11-1 shows the block diagram of the power-on-clear circuits.

Figure 11-1. Block Diagram of Power-on-Clear Circuit



11.3 Register Controlling Power-on-Clear Circuit

The power-on-clear circuits are controlled by the following register.

- Power-on-clear register 10 (POCF10)

(1) Power-on-clear register 10 (POCF10)

POCF10 controls POC circuit operation.

POCF10 is set with a 1-bit or 8-bit memory manipulation instruction.

Figure 11-2. Format of Power-on-Clear Register 10

Symbol	7	6	5	4	3	<2>	1	0	Address	After reset	R/W
POCF10	0	0	0	0	0	POCOF10	POCMK1	0	FFDDH	Retained[Note]	R/W

POCOF10	POC output detection flag
0	Non-generation of reset signal by POC or in cleared state due to a write operation to POCF10
1	Generation of reset signal by POC

POCMK1	POC control
0	Generation of reset signal by POC is enabled (the POC circuit is used)
1	Generation of reset signal by POC is disabled (the POC circuit is not used)

Note This value is 04H only after a power-on-clear reset. POCF10 is not affected by any other reset. Write any value to POCF10 to clear to 00H.

POCF10 is also cleared to 00H when the power is turned off. However, POCMK1 is cleared to 0 on power application, which means that a reset by power-on is generated, resulting in the register being cleared to 04H.

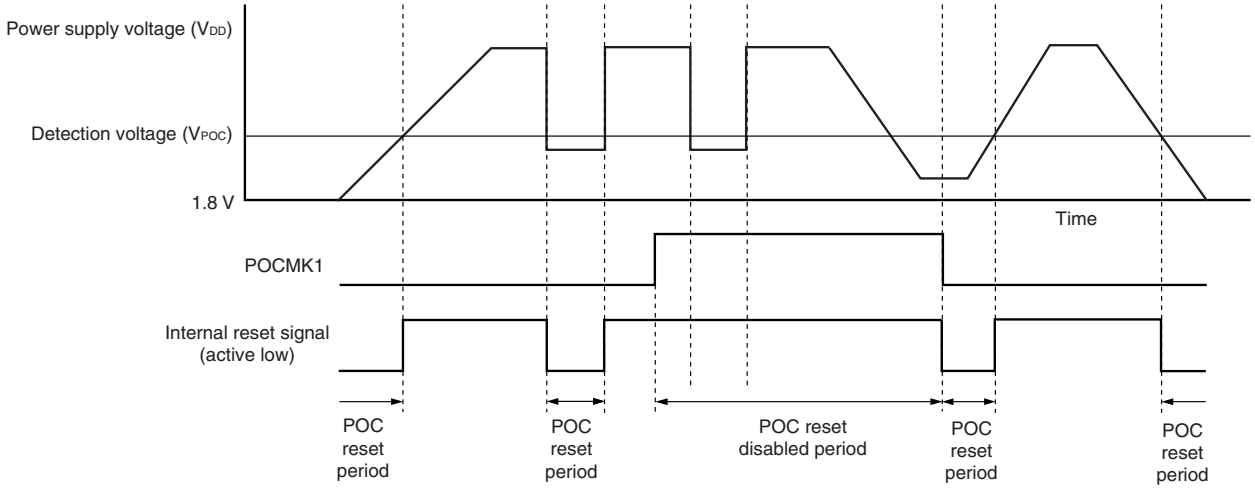
Caution Bits 0 and 3 to 7 must all be set to 0.

11.4 Operation of Power-on-Clear Circuit

The POC circuit compares the detection voltage (V_{POC}) with the power supply voltage (V_{DD}) and generates an internal reset signal if $V_{DD} < V_{POC}$.

When a reset is generated via the power-on-clear circuit in bit 2 (POCOF0) on power-on-clear register 10 (POCF10) is set (1). This bit is then cleared (0) by an instruction written to POCF10. After a power-on-clear reset (i.e. after program execution has started from address 0000H), a power failure can be detected by detecting POCOF0.

Figure 11-3. Timing of Internal Reset Signal Generation of POC Circuit



CHAPTER 12 LOW-VOLTAGE DETECTOR

12.1 Function of Low-Voltage Detector

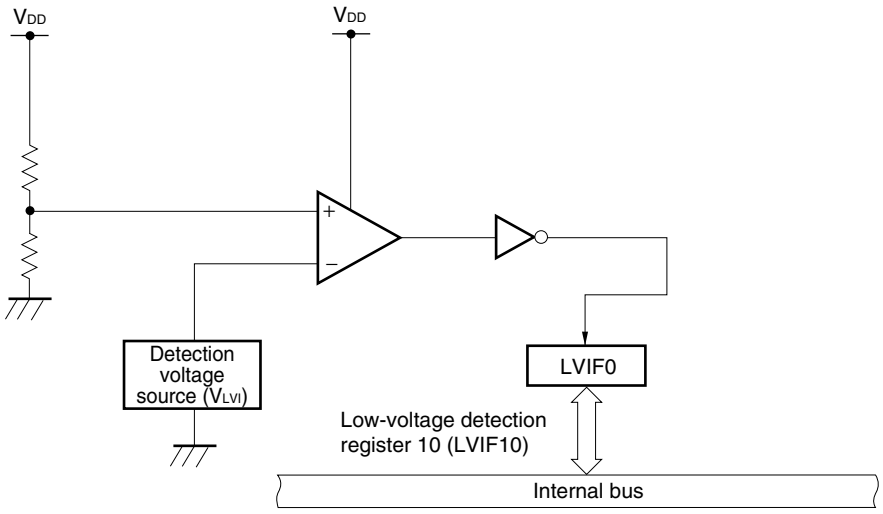
The low-voltage detector (LVI) includes the following function.

- The low-voltage detector compares the detection voltage (V_{LVI}) with the power supply voltage (V_{DD}) and sets a flag (LVIF0) in the low-voltage detection register (0 →1) if $V_{DD} < V_{LVI}$.
- This circuit can be used to judge the necessity of RAM data initialization.
- This circuit can operate even in STOP mode.

12.2 Configuration of Low-Voltage Detector

Figure 12-1 shows the block diagram of the low-voltage detector.

Figure 12-1. Block Diagram of Low-Voltage Detector



12.3 Register Controlling Low-Voltage Detector

The low-voltage detector is controlled by the following register.

- Low-voltage detection register 10 (LVIF10)

(1) Low-voltage detection register 10 (LVIF10)

LVIF10 controls LVI circuit operation.

LVIF10 is set with a 1-bit or 8-bit memory manipulation instruction.

Figure 12-2. Format of Low-Voltage Detection Register 10

Symbol	7	6	5	4	3	<2>	1	0	Address	After reset	R/W
LVIF10	0	0	0	0	0	LVIF0	0	0	FFDEH	Retained[Note]	R/W

LVIF0	Low-voltage detection flag
0	Non-detection of low voltage by LVI or in cleared state due to a write operation to LVIF10
1	Detection of low voltage by LVI

Note This value is 04H only when V_{DD} < LVI detection voltage. LVIF10 is not affected by any other reset. Write any value to POCF10 to clear to 00H.

LVIF10 is set to 04H on power application because V_{DD} is always less than LVI detection voltage.

Caution Bits 0, 1 and 3 to 7 must all be set to 0.

12.4 Operation of Low-Voltage Detector

The LVI circuit compares the detection voltage (V_{LVI}) with the power supply voltage (V_{DD}) and sets a flag (LVIF0) in the low-voltage detection register (0 →1) if V_{DD} < V_{LVI}.

When a low voltage is detected via the LVI circuit, bit 2 (LVIF0) of low-voltage detection register 10 (LVIF10) is set (1). This bit is then cleared (0) by an instruction written to LVIF10.

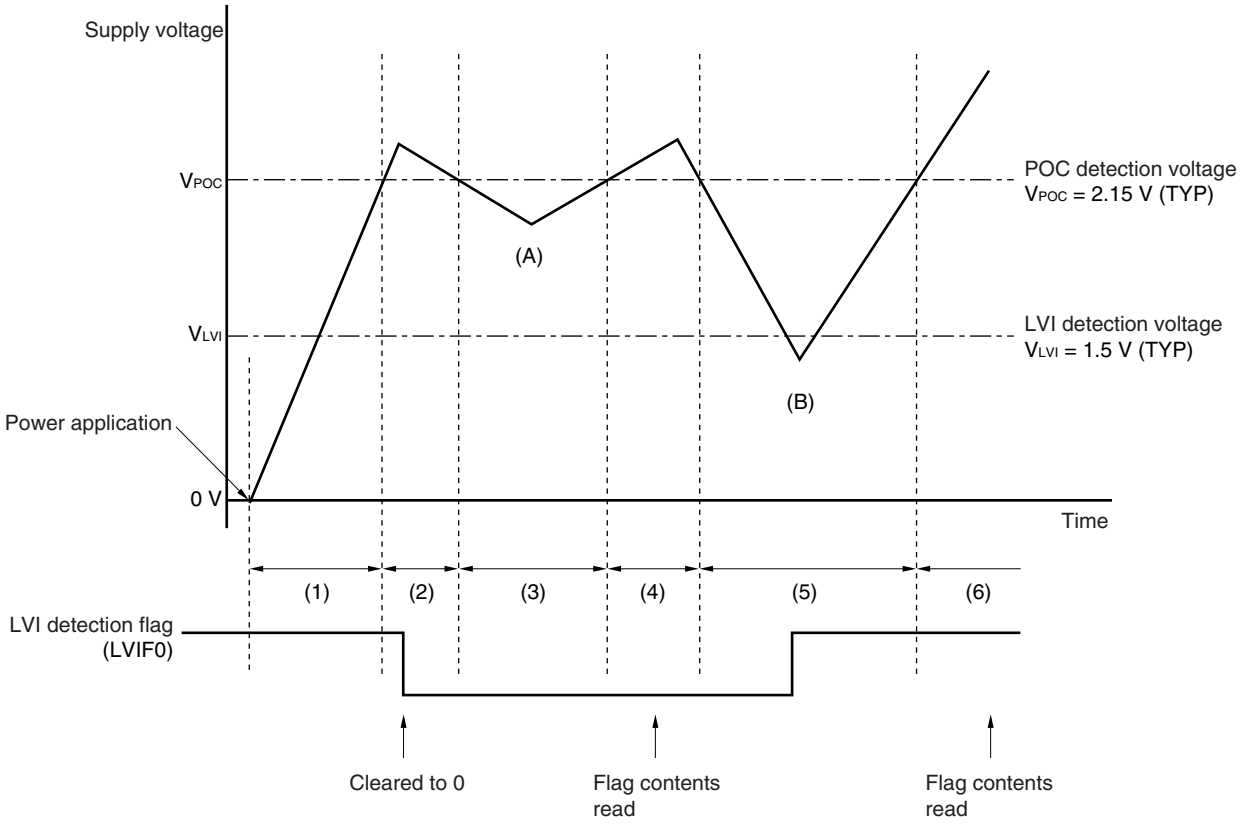
An example of how to use the LVI detection flag (LVIF0) is explained below.

Whether the voltage level has lowered to the point that RAM data is destroyed can be judged by checking this flag at a time such as when the battery is exchanged or when the battery voltage has dropped.

LVIF0 = 0: RAM data is not destroyed (normal)

LVIF0 = 1: RAM data is destroyed or RAM data is not set at power application (RAM data must be initialized)

Figure 12-3. Supply Voltage Transition and Detection Voltage



(1) The setting of a battery, etc., causes the supply voltage to rise, and when V_{POC} (POC detection voltage) is exceeded, reset is released.
Because the supply voltage was initially 0 V (equal to or lower than V_{LVI} (LVI detection voltage), LVIF0 is 1.

(2) This is the voltage at which operation is enabled. Program so that LVIF0 is cleared to 0 after the required data is written to RAM.

(3) When the supply voltage falls to V_{POC} or lower, a reset is executed.
However, because the voltage is higher than V_{LVI} at point (A), LVIF0 remains 0.

(4) When the voltage rises again to V_{POC} or higher, reset is released.
LVIF0 is confirmed by software to be 0 after reset release, and so it can be judged that RAM data hasn't been destroyed. It is therefore not necessary to initialize RAM data by software in this case.

(5) When the supply voltage falls to V_{POC} or lower, a reset is executed.
Because the voltage is equal to or lower than V_{POC} at point (B), LVIF0 changes from 0 to 1.

(6) LVIF0 is confirmed by software to be 1 after reset release, and so it can be judged that RAM data could have been destroyed. In this case, RAM data must be initialized by software.

CHAPTER 13 INTERRUPT FUNCTIONS

13.1 Interrupt Function Types

The following two types of interrupt functions are used.

(1) Non-maskable interrupt

This interrupt is acknowledged even if interrupts are disabled. It does not undergo interrupt priority control and is given top priority over all other interrupt requests.

★ A standby release signal is generated and HALT mode is released.

An interrupt from the watchdog timer is the only non-maskable interrupt source.

(2) Maskable interrupt

These interrupts undergo mask control. If two or more interrupts are simultaneously generated, each interrupt has a predetermined priority as shown in Table 13-1.

★ A standby release signal is generated and STOP and HALT modes are released.

There are three external sources and eight internal sources of maskable interrupts.

13.2 Interrupt Sources and Configuration

There are a total of 12 non-maskable and maskable interrupt sources (see **Table 13-1**).

Table 13-1. Interrupt Sources

Interrupt Type	Priority[Note 1]	Interrupt Source		Internal/External	Vector Table Address	Basic Configuration Type[Note 2]
		Name	Trigger			
Non-maskable	–	INTWDT	Watchdog timer overflow (with watchdog timer mode 1 selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (with the interval timer mode selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTTM50	Match between TM50 and CR50	Internal	000AH	(B)
	4	INTTM60	Match between TM60 and CR60 (in 8-bit counter mode), and between TM50, TM60 and CR50, CR60 (in 16-bit timer mode)		000CH	
	5	INTTM80	Generation of matching signal of 8-bit timer 80		000EH	
	6	INTTM20	Generation of matching signal of 16-bit timer 20		0010H	
	7	INTKR00	Key return signal detection	External	0012H	(C)
	8	INTSR20	End of serial interface 20 UART reception	Internal	0014H	(B)
		INTCSI20	End of serial interface 20 3-wire SIO transfer reception			
	9	INTST20	End of serial interface 20 UART transmission		0016H	

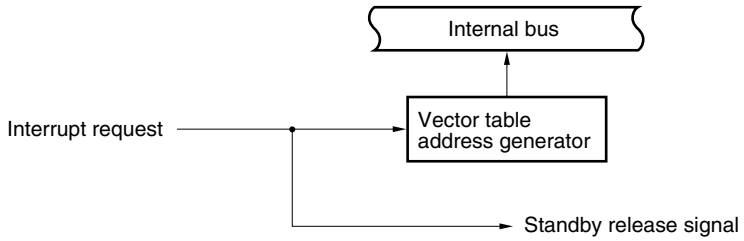
Notes 1. Priority is the priority order when several maskable interrupt requests are generated at the same time. 0 is the highest and 9 is the lowest.

2. Basic configuration types (A), (B), and (C) correspond to (A), (B), and (C) in Figure 13-1.

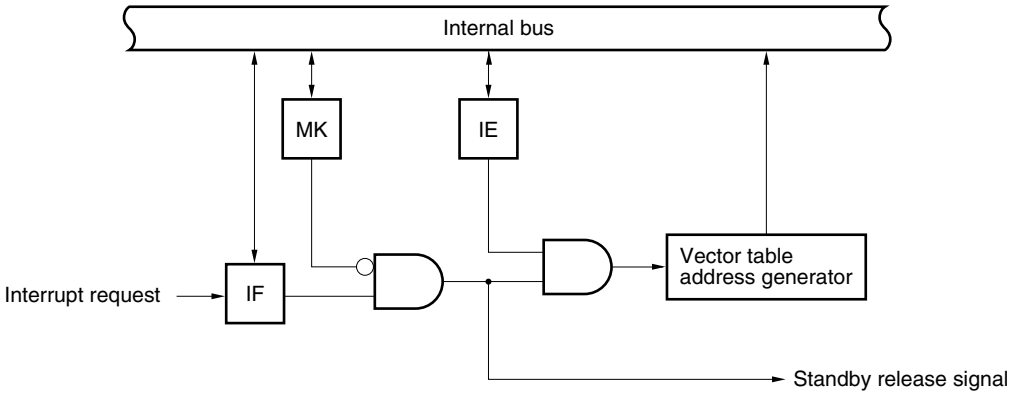
Remark There are two interrupt sources for the watchdog timer (INTWDT): non-maskable interrupts and maskable interrupts. Either one (but not both) should be selected for actual use.

Figure 13-1. Basic Configuration of Interrupt Function

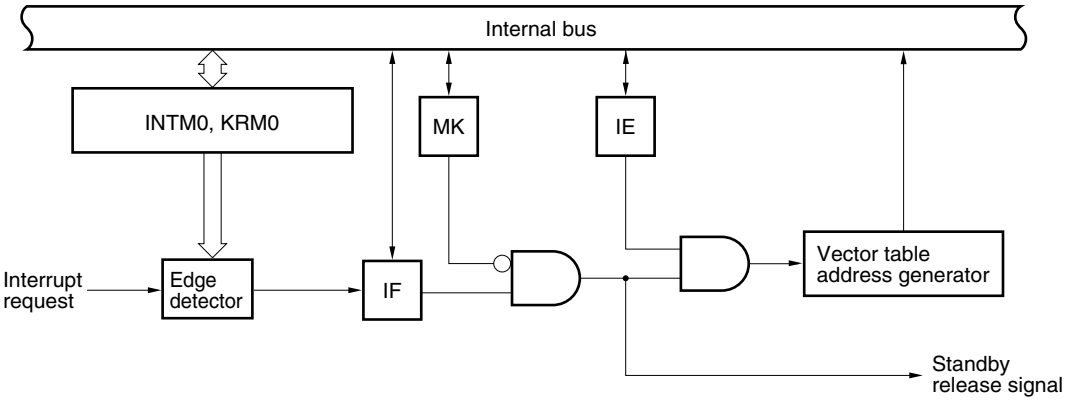
(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt



IF: Interrupt request flag
IE: Interrupt enable flag
MK: Interrupt mask flag
INTM0: External interrupt mode register 0
KRM0: Key return mode register 0

13.3 Interrupt Function Control Registers

The interrupt functions are controlled by the following five types of registers:

- Interrupt request flag registers 0 and 1 (IF0 and IF1)
- Interrupt mask flag registers 0 and 1 (MK0 and MK1)
- External interrupt mode register 0 (INTM0)
- Key return mode register 0 (KRM0)
- Program status word (PSW)

Table 13-2 lists interrupt requests, the corresponding interrupt request flags, and interrupt mask flags.

Table 13-2. Interrupt Request Signals and Corresponding Flags

Interrupt Request Signal	Interrupt Request Flag	Interrupt Mask Flag
INTWDT	TMIF4	TMMK4
INTP0	PIF0	PMK0
INTP1	PIF1	PMK1
INTTM50	TMIF50	TMMK50
INTTM60	TMIF60	TMMK60
INTTM80	TMIF80	TMMK80
INTTM20	TMIF20	TMMK20
INTKR00	KRIF00	KRMK00
INTSR20/INTCSI20	SRIF20/CSIIF20	SRMK20/CSIMK20
INTST20	STIF20	STMK20

(1) Interrupt request flag registers 0 and 1 (IF0 and IF1)

An interrupt request flag is set to 1 when the corresponding interrupt request is issued, or when the related instruction is executed. It is cleared to 0 when the interrupt request is acknowledged, when a $\overline{\text{RESET}}$ signal is input, or when a related instruction is executed.

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

$\overline{\text{RESET}}$ input clears IF0 and IF1 to 00H.

Figure 13-2. Format of Interrupt Request Flag Register

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
IF0	KRIF00	TMIF20	TMIF80	TMIF60	TMIF50	PIF1	PIF0	TMIF4	FFE0H	00H	R/W
	7	6	5	4	3	2	<1>	<0>			
IF1	0	0	0	0	0	0	STIF20	SRIF20/ CSIIF20	FFE1H	00H	R/W

××IF×	Interrupt request flag
0	No interrupt request signal has been issued.
1	An interrupt request signal has been issued; an interrupt request has been made.

Cautions 1. Bits 2 to 7 of IF1 must all be set to 0.

2. The TMIF4 flag can be read- and write-accessed only when the watchdog timer is being used as an interval timer. It must be cleared to 0 if the watchdog timer is used in watchdog timer mode 1 or 2.

3. When port 4 is being used as an output port, and its output level is changed, an interrupt request flag is set, because this port is also used as an external interrupt input. To use port 2 in output mode, therefore, the interrupt mask flag must be preset to 1.

(2) Interrupt mask flag registers 0 and 1 (MK0 and MK1)

The interrupt mask flags are used to enable and disable the corresponding maskable interrupts.

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

$\overline{\text{RESET}}$ input sets MK0 and MK1 to FFH.

Figure 13-3. Format of Interrupt Mask Flag Register

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	After reset	R/W
MK0	KRMK00	TMMK20	TMMK80	TMMK60	TMMK50	PMK1	PMK0	TMMK4	FFE4H	FFH	R/W
	7	6	5	4	3	2	<1>	<0>			
MK1	1	1	1	1	1	1	STMK20	SRMK20/ CSIMK20	FFE5H	FFH	R/W

××MK	Interrupt handling control
0	Enables interrupt servicing.
1	Disables interrupt servicing.

Cautions 1. Bits 2 to 7 of MK1 must all be set to 1.

2. When the watchdog timer is being used in watchdog timer mode 1 or 2, any attempt to read the TMMK4 flag results in an undefined value being detected.

3. When port 4 is being used as an output port, and its output level is changed, an interrupt request flag is set, because this port is also used as an external interrupt input. To use port 4 in output mode, therefore, the interrupt mask flag must be preset to 1.

(3) External interrupt mode register 0 (INTM0)

INTM0 is used to specify a valid edge for INTP0 and INTP1.

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

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

Figure 13-4. Format of External Interrupt Mode Register 0

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
INTM0	0	0	ES11	ES10	ES01	ES00	0	0	FFECH	00H	R/W

ES11	ES10	INPT1 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES01	ES00	INPT0 valid edge selection
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

Cautions 1. Bits 0, 1, 6, and 7 must all be set to 0.

2. Before setting INTM0, set the corresponding interrupt mask flag to 1 to disable interrupts.

To enable interrupts, clear to 0 the corresponding interrupt request flag, then the corresponding interrupt mask flag.

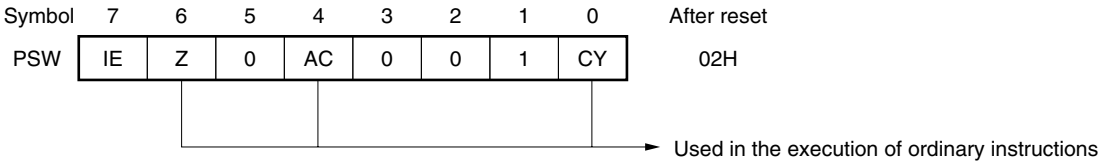
(4) Program status word (PSW)

The program status word is used to hold the instruction execution result and the current status of the interrupt requests. The IE flag, used to enable and disable maskable interrupts, is mapped to PSW.

PSW can be read- and write-accessed in 8-bit units, as well as using bit manipulation instructions and dedicated instructions (EI and DI). When a vector interrupt is acknowledged, the PSW is automatically saved to a stack, and the IE flag is reset to 0.

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

Figure 13-5. Program Status Word Configuration



IE	Whether to enable/disable interrupt acknowledgement
0	Disabled
1	Enabled

(5) Key return mode register 0 (KRM0)

This register is used to set the pin that is to detect the key return signal (rising edge of port 4).

KRM0 is set with a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets KRM0 to 00H.

Figure 13-6. Format of Key Return Mode Register 0

Symbol	<7>	<6>	<5>	<4>	3	2	1	<0>	Address	After reset	R/W
KRM0	KRM07	KRM06	KRM05	KRM04	0	0	0	KRM00	FFF5H	00H	R/W

KRM00	Control of key return signal detection
0	Key return signal not detected
1	Key return signal detected (P40 to P43 falling edge detection)

KRM0n	Control of key return signal detection
0	Key return signal not detected
1	Key return signal detected (P4n falling edge detection)

Remark n = 4 to 7

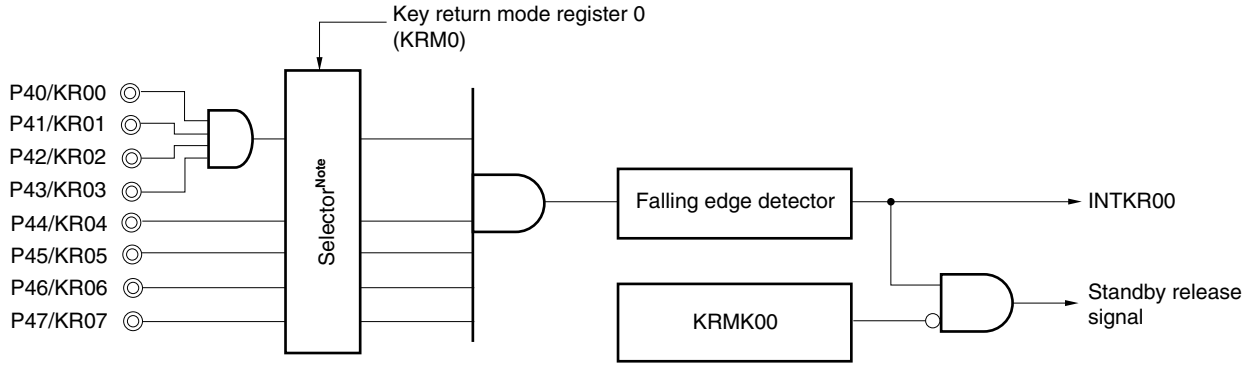
Cautions 1. Bits 1 to 3 must all be set to 0.

2. Before setting KRM0, set (1) bit 7 (KRMK00) of MK0 to disable interrupts. After KRM0 is set, clear (0) KRMK00 after clearing (0) bit 7 (KRIF00) of IF0 to enable interrupts.

3. On-chip pull-up resistors are automatically connected in input mode to the pins specified for key return signal detection (P40 to P47). Although these resistors are disconnected when the mode changes to output, key return signal detection continues unchanged.

4. A key return signal cannot be detected while any one of the pins specified for key return detection is low level even when a falling edge occurred at another pin.

Figure 13-7. Block Diagram of Key Return Signal Detector



Note For selecting the pin to be used as the falling edge input.

13.4 Interrupt Processing Operation

13.4.1 Non-maskable interrupt request acknowledgement operation

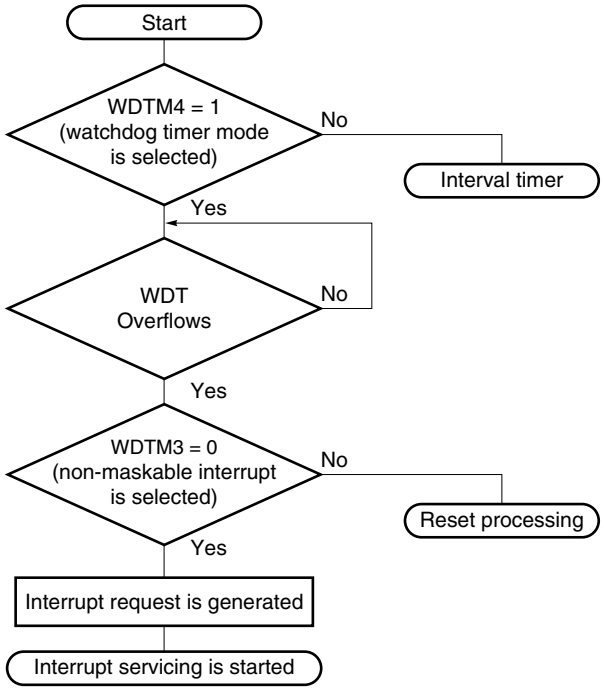
The non-maskable interrupt request is unconditionally acknowledged even when interrupts are disabled. It is not subject to interrupt priority control and takes precedence over all other interrupts.

When the non-maskable interrupt request is acknowledged, PSW and PC are saved to the stack in that order, the IE flag is reset to 0, the contents of the vector table are loaded to the PC, and then program execution branches.

Figure 13-8 shows the flowchart from non-maskable interrupt request generation to acknowledgement. Figure 13-9 shows the timing of non-maskable interrupt request acknowledgement. Figure 13-10 shows the acknowledgement operation if multiple non-maskable interrupts are generated.

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

Figure 13-8. Flowchart from Non-Maskable Interrupt Request Generation to Acknowledgement



WDTM: Watchdog timer mode register
WDT: Watchdog timer

Figure 13-9. Timing of Non-Maskable Interrupt Request Acknowledgement

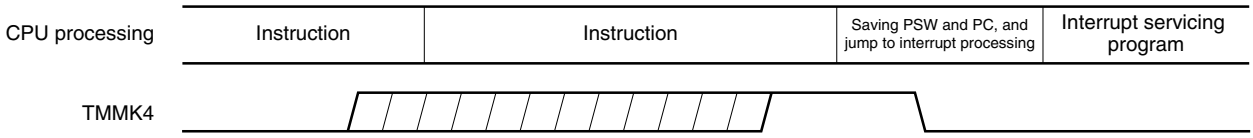
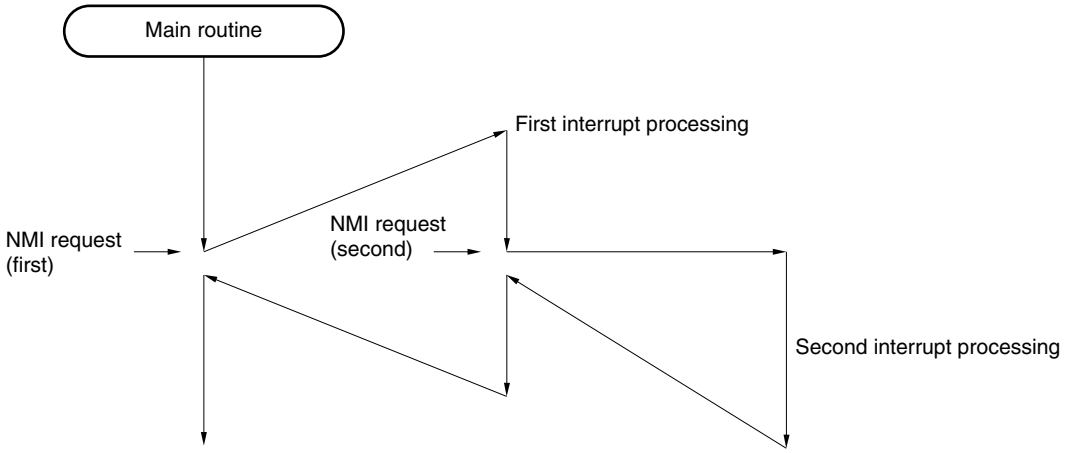


Figure 13-10. Acknowledgement of Non-Maskable Interrupt Request



13.4.2 Maskable interrupt request acknowledgement operation

A maskable interrupt request can be acknowledged when the interrupt request flag is set to 1 and the corresponding interrupt mask flag is cleared to 0. A vectored interrupt request is acknowledged in the interrupt enabled status (when the IE flag is set to 1).

The time required to start the interrupt servicing after a maskable interrupt request has been generated is shown in Table 13-3.

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

Table 13-3. Time from Generation of Maskable Interrupt Request to Servicing

Minimum Time	Maximum Time[Note]
9 clocks	19 clocks

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

Remark 1 clock: $\frac{1}{f_{CPU}}$ (f_{CPU}: CPU clock)

When two or more maskable interrupt requests are generated at the same time, they are acknowledged starting from the interrupt request assigned the highest priority.

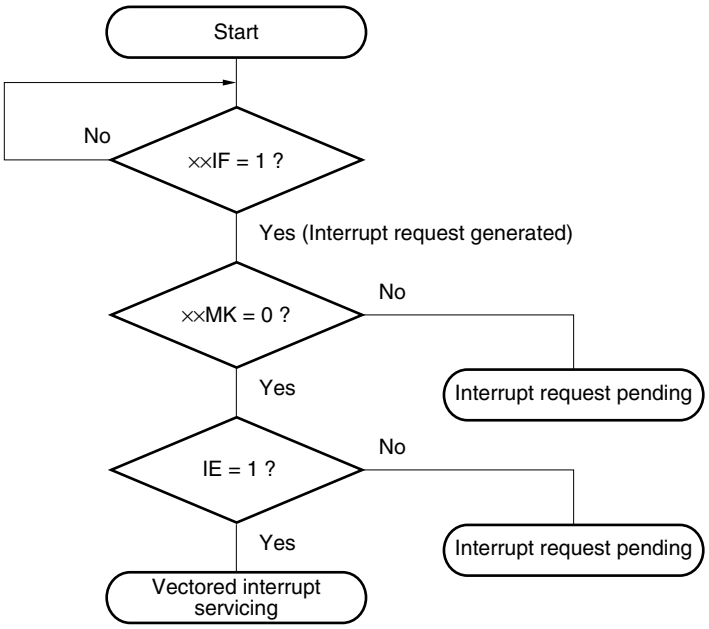
A pending interrupt is acknowledged when a status in which it can be acknowledged is set.

Figure 13-11 shows the algorithm of interrupt requests acknowledgement.

When a maskable interrupt request is acknowledged, the contents of the PSW and PC are saved to the stack in that order, the IE flag is reset to 0, and the data in the vector table determined for each interrupt request is loaded to the PC, and execution branches.

To return from interrupt servicing, use the RETI instruction.

Figure 13-11. Interrupt Request Acknowledgement Processing Algorithm

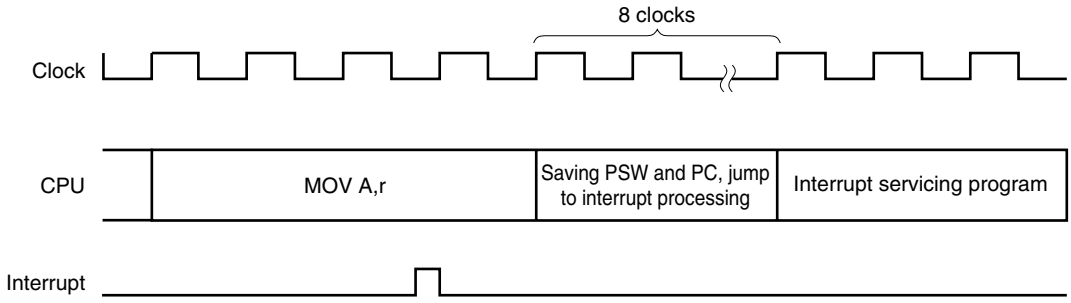


××IF: Interrupt request flag

××MK: Interrupt mask flag

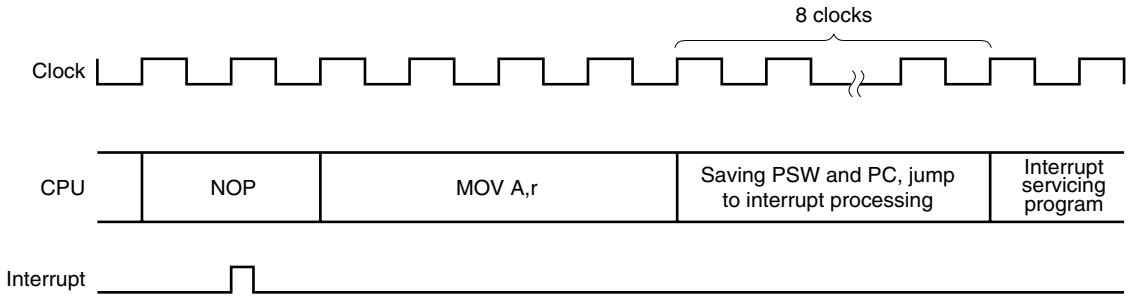
IE: Flag to control maskable interrupt request acknowledgement (1 = enable, 0 = disable)

Figure 13-12. Interrupt Request Acknowledgement Timing (Example of MOV A,r)



If an interrupt request flag (××IF) is set before an instruction clock n (n = 4 to 10) under execution becomes n − 1, the interrupt is acknowledged after the instruction under execution is complete. Figure 13-12 shows an example of the interrupt request acknowledgement timing for an 8-bit data transfer instruction MOV A,r. Since this instruction is executed for 4 clocks, if an interrupt occurs for 3 clocks after the execution starts, the interrupt acknowledgement processing is performed after the MOV A,r instruction is executed.

Figure 13-13. Interrupt Request Acknowledgement Timing (When Interrupt Request Flag Is Set at the Last Clock During Instruction Execution)



If an interrupt request flag (××IF) is set at the last clock of the instruction, the interrupt acknowledgement processing starts after the next instruction is executed. Figure 13-13 shows an example of the interrupt acknowledgement timing for an interrupt request flag that is set at the second clock of NOP (2-clock instruction). In this case, the MOV A,r instruction after the NOP instruction is executed, and then the interrupt acknowledgement processing is performed.

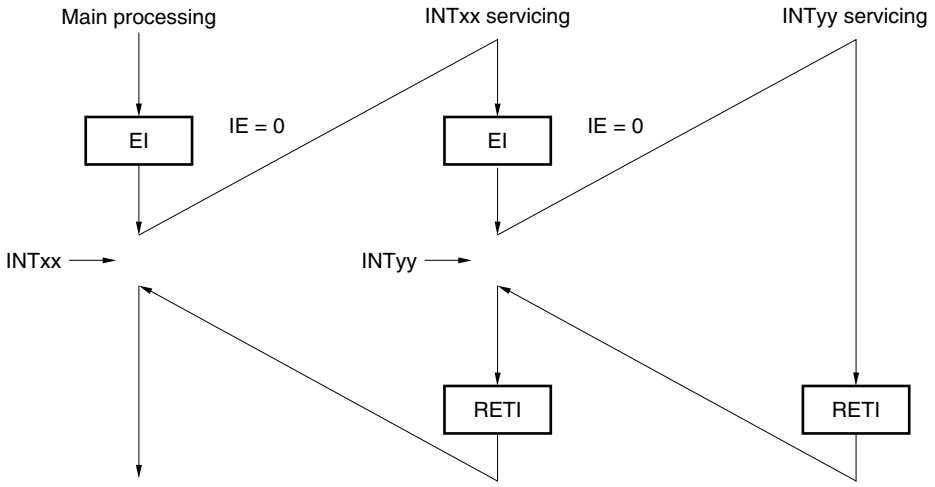
Caution Interrupt requests are reserved while interrupt request flag register 0 or 1 (IF0 or IF1) or interrupt mask flag register 0 or 1 (MK0 or MK1) is being accessed.

13.4.3 Multiple interrupt servicing

Multiple interrupt servicing in which another interrupt is acknowledged while an interrupt is being processed can be performed using a priority order system. When two or more interrupts are generated at once, interrupt servicing is performed according to the priority assigned to each interrupt request in advance (see **Table 13-1**).

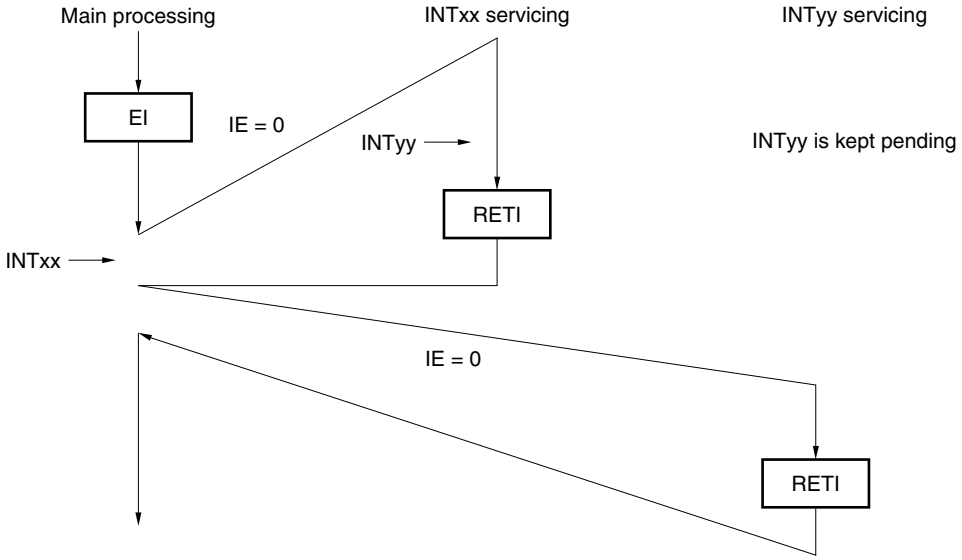
Figure 13-14. Example of Multiple Interrupts

Example 1. A multiple interrupt is acknowledged



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

Example 2. Multiple interrupts are not generated because interrupts are not enabled



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

IE = 0: Interrupt request acknowledgement disabled

13.4.4 Interrupt request reserve

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

- Manipulation instruction for interrupt request flag registers 0 and 1 (IF0 and IF1)
- Manipulation instruction for interrupt mask flag registers 0 and 1 (MK0 and MK1)

CHAPTER 14 STANDBY FUNCTION

14.1 Standby Function and Configuration

14.1.1 Standby function

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

(1) HALT mode

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

(2) STOP mode

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

The low voltage (V_{DD} = 1.2 V max. (target value)) of the data memory can be retained. Therefore, this mode is useful for retaining the contents of the data memory at an extremely low current consumption.

STOP mode can be released by an interrupt request, so that this mode can be used for intermittent operation. However, some time is required until the system clock oscillator stabilizes after 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 latches of the I/O ports and output buffers are also retained.

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

14.1.2 Standby function control register

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

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

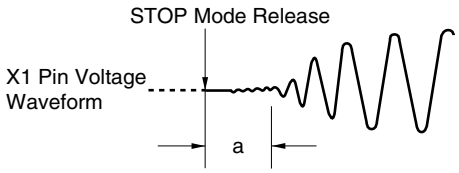
$\overline{\text{RESET}}$ input sets OSTS to 04H. However, the oscillation stabilization time after $\overline{\text{RESET}}$ input is 2^{15}/fx, instead of 2^{17}/fx.

Figure 14-1. Format of Oscillation Stabilization Time Selection Register

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

OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection
0	0	0	2^{12}/fx (819 μs)
0	1	0	2^{15}/fx (6.55 ms)
1	0	0	2^{17}/fx (26.2 ms)
Other than above			Setting prohibited

Caution The wait time after STOP mode is released does not include the time from STOP mode release to clock oscillation start ("a" in the figure below), regardless of release by $\overline{\text{RESET}}$ input or by interrupt generation.



Remarks 1. fx: System clock oscillation frequency

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

14.2 Operation of Standby Function

14.2.1 HALT mode

(1) HALT mode

HALT mode is set by executing the HALT instruction.

The operation statuses in HALT mode are shown in the following table.

Table 14-1. Operation Statuses in HALT Mode

Item	HALT Mode Operation Status
System clock generator	System clock oscillation enabled Clock supply to CPU stopped
CPU	Operation disabled
Port (output latch)	Remains in the state existing before the selection of HALT mode
16-bit timer 20	Operation enabled
8-bit timers 50, 60, 80	Operation enabled
Watchdog timer	Operation enabled
Serial interface 20	Operation enabled
Power-on-clear circuit	Operation enabled[Note 1]
Low voltage detector (LVI)	Operation enabled
External interrupt	Operation enabled[Note 2]
Key return detector	Operation enabled[Note 2]

Notes 1. Only when the use of the POC circuit is specified (POCMK1 = 0)

2. Maskable interrupt that is not masked

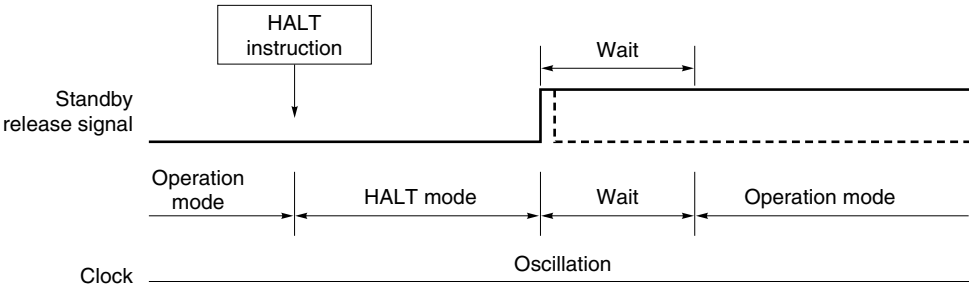
(2) Releasing HALT mode

HALT mode can be released by the following three sources:

(a) Releasing by unmasked interrupt request

HALT mode is released by an unmasked interrupt request. In this case, if interrupt request acknowledgement is enabled, vectored interrupt servicing is performed. If interrupt acknowledgement is disabled, the instruction at the next address is executed.

Figure 14-2. Releasing HALT Mode by Interrupt



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

2. The wait time is as follows:

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

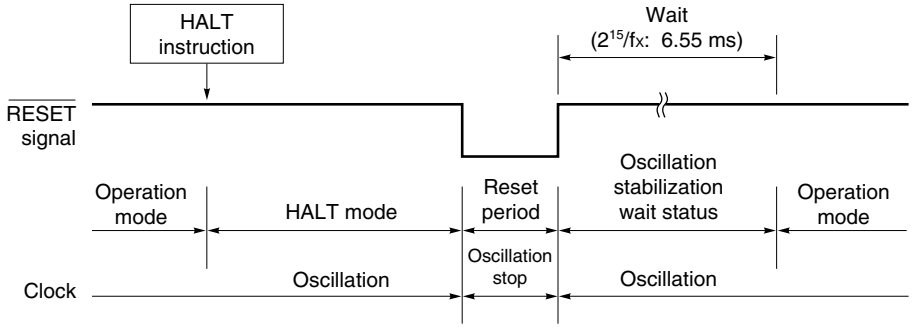
(b) Releasing by non-maskable interrupt request

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

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

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

Figure 14-3. Releasing HALT Mode by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

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

Table 14-2. Operation After Releasing HALT Mode

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

×: Don't care

14.2.2 STOP mode

(1) Setting and operation status of STOP mode

STOP mode is set by executing the STOP instruction.

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

The operation statuses in STOP mode are shown in the following table.

Table 14-3. Operation Statuses in STOP Mode

Item	STOP Mode Operation Status
Clock generator	System clock oscillation stopped
CPU	Operation stopped
Port (output latch)	Remains in the state existing before STOP mode has been set
16-bit timer 20	Operation stopped
8-bit timers 50, 60, 80	Operation stopped
Watchdog timer	Operation stopped
Serial interface 20	Operation enabled only when external clock is input to serial clock
Power-on-clear circuit	Operation enabled[Note 1]
Low voltage detector (LVI)	Operation enabled
External interrupt	Operation enabled[Note 2]
Key return detector	Operation enabled[Note 2]

Notes 1. Only when the use of the POC circuit is specified (POCMK1 = 0)

2. Maskable interrupt that is not masked

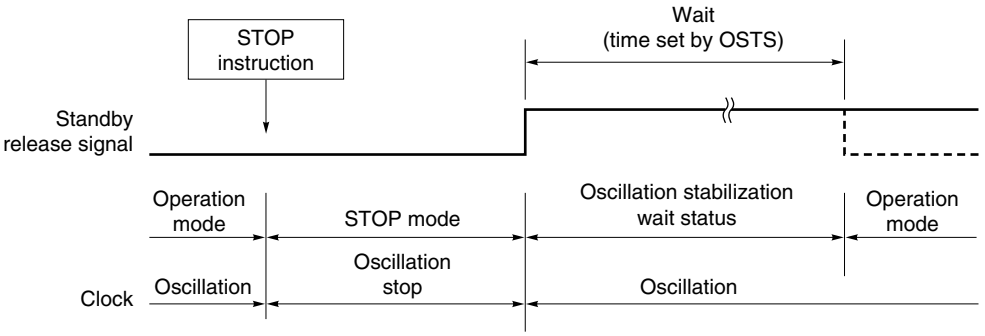
(2) Releasing STOP mode

STOP mode can be released by the following two sources:

(a) Releasing by unmasked interrupt request

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

Figure 14-4. Releasing STOP Mode by Interrupt

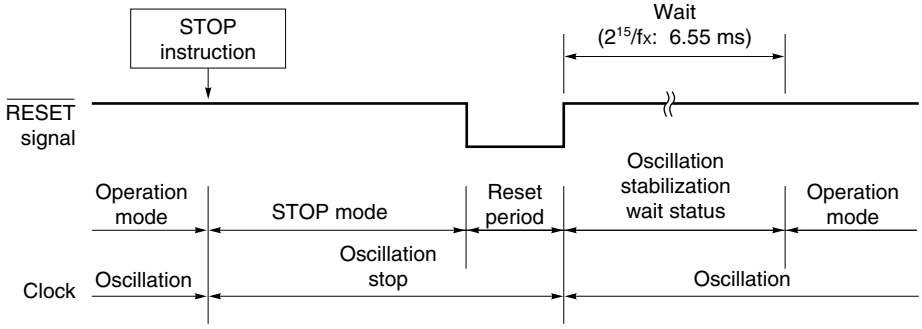


Remark The broken lines indicate the case where the interrupt request that has released standby mode is acknowledged.

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

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

Figure 14-5. Releasing STOP Mode by $\overline{\text{RESET}}$ Input



Remarks 1. fx: System clock oscillation frequency

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

Table 14-4. Operation After Releasing STOP Mode

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

×: Don't care

CHAPTER 15 RESET FUNCTION

The following three operations are available to generate reset signals.

(1) External reset signal input via $\overline{\text{RESET}}$ pin
(2) Internal reset by detection of watchdog timer inadvertent program loop time
(3) Internal reset using power-on-clear circuit (POC)

The external and internal reset signals are functionally equivalent. When $\overline{\text{RESET}}$ is input, program execution begins from the addresses written at addresses 0000H and 0001H.

If a low-level signal is applied to the $\overline{\text{RESET}}$ pin, or if the watchdog timer overflows, a reset occurs, causing each item of the hardware to enter the states listed in Table 15-1. While a reset is being applied, or while the oscillation frequency is stabilizing immediately after the end of a reset sequence, each pin remains in the high-impedance state.

If a high-level signal is applied to the $\overline{\text{RESET}}$ pin, the reset sequence is terminated, and program execution is started after the oscillation stabilization time has elapsed. A reset sequence caused by a watchdog timer overflow is terminated automatically and program execution is started after the oscillation stabilization time has elapsed.

Reset by power-on-clear (POC) is cleared if the supply voltage rises beyond a specific level, and the program execution is started after the oscillation stabilization time has elapsed.

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 15-1. Block Diagram of Reset Function

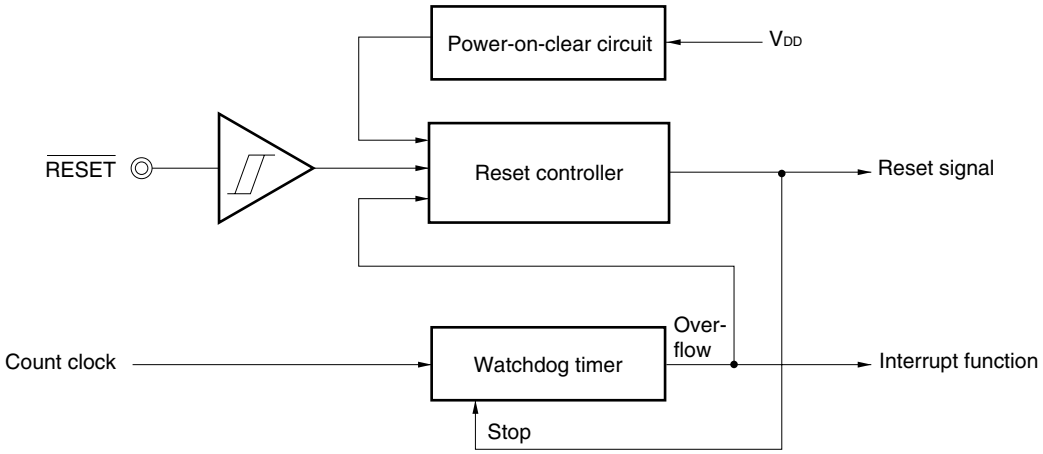


Figure 15-2. Reset Timing by $\overline{\text{RESET}}$ Input

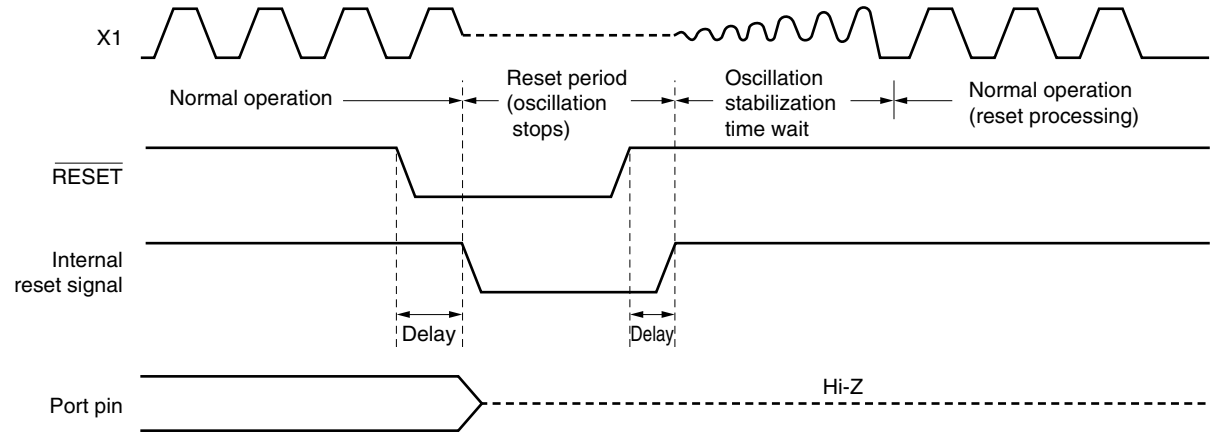


Figure 15-3. Reset Timing by Watchdog Timer Overflow

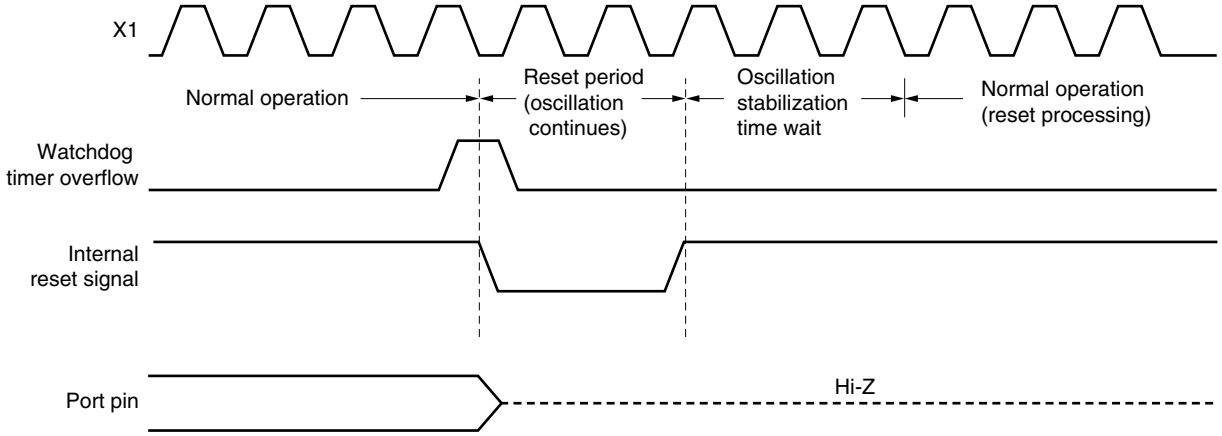


Figure 15-4. Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode

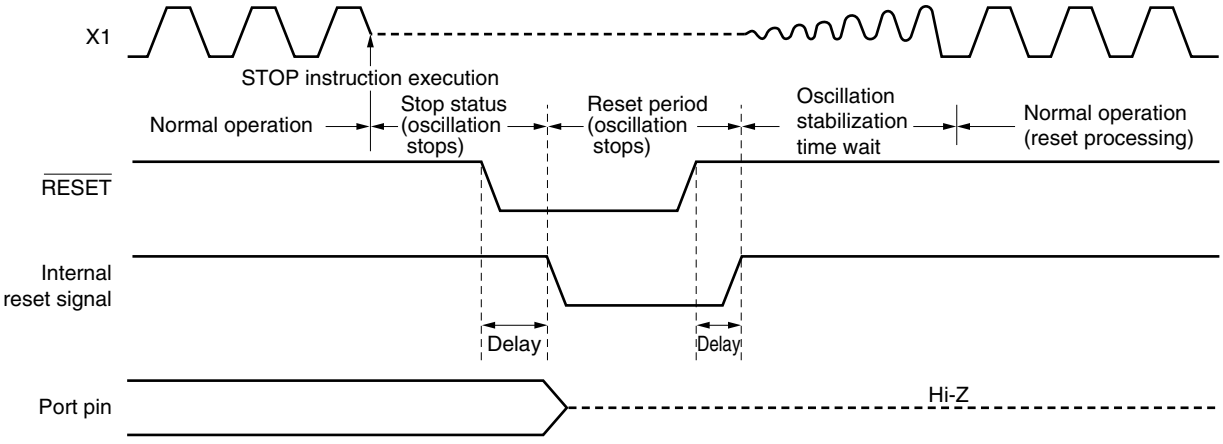
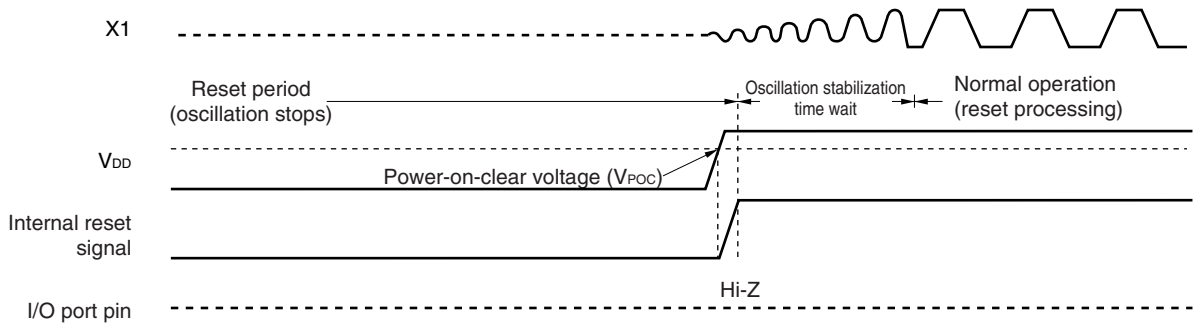
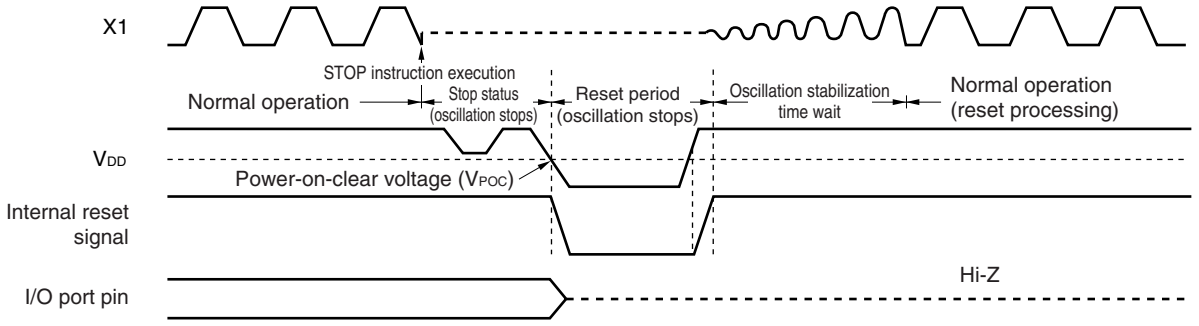


Figure 15-5. Reset Timing by Power-on Clear

(a) At power application



(b) In STOP mode



(c) In normal operation mode (including HALT mode)

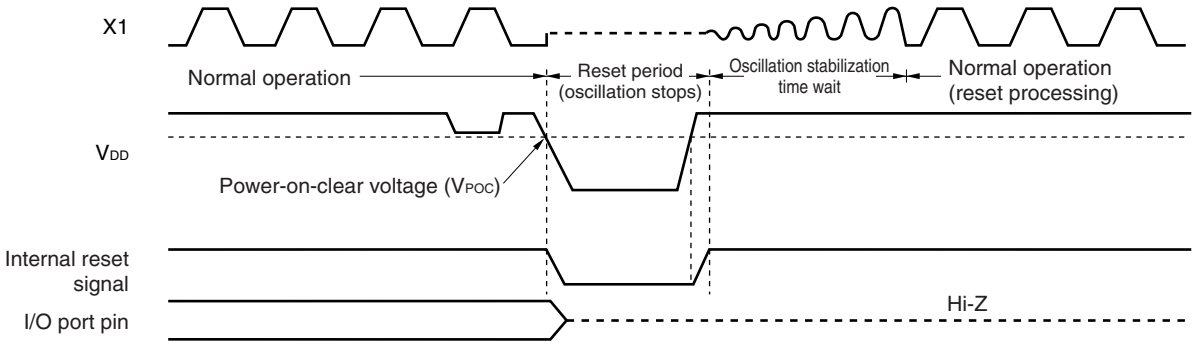


Table 15-1. Status of Hardware After Reset

Hardware		Status After Reset
Program counter (PC)[Note 1]		Loaded with the contents of the reset vector table (0000H, 0001H)
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined[Note 2]
	General-purpose registers	Undefined[Note 2]
Ports (P0, P2, P4) (output latch)		00H
Port mode registers (PM0, PM2, PM4)		FFH
Pull-up resistor option registers (PUB0, PUB4)		00H
Processor clock control register (PCC)		02H
Clock multiplication control register (CMC0)		00H
Oscillation stabilization time selection register (OSTS)		04H
16-bit timer 20	Timer counter (TM20)	0000H
	Compare register (CR20)	FFFFH
	Control register (TMC20)	00H
	Capture register (TCP20)	Undefined
8-bit timers 50, 60, 80	Timer counter (TM50, TM60, TM80)	00H
	Compare register (CR50, CR60, CRH60, CR80)	Undefined
	Mode control register (TMC50, TMC60, TMC80)	00H
	Carrier generator output control register (TCA60)	00H
	REM signal control register (RSCR0)	00H
	TM50 source clock control register (ADSC5)	00H
Watchdog timer	Clock selection register (TCL2)	00H
	Mode register (WDTM)	00H
Serial interface 20	Serial operation mode register (CSIM20)	00H
	Asynchronous serial interface mode register (ASIM20)	00H
	Asynchronous serial interface status register (ASIS20)	00H
	Baud rate generator control register (BRGC20)	00H
	Transmit shift register (TXS20)	FFH
	Receive buffer register (RXB20)	Undefined
Power-on-clear circuit	Power-on-clear register 10 (POCF10)	Retained[Note 3]
Low-voltage detector	Low-voltage detection register 10 (LVIF10)	Retained[Note 4]
Interrupts	Request flag registers (IF0, IF1)	00H
	Mask flag registers (MK0, MK1)	FFH
	External interrupt mode register (INTM0)	00H
	Key return mode register 0 (KRM0)	00H

Notes 1. While a reset signal is being input, and during the oscillation stabilization period, the contents of the PC will be undefined, while the remainder of the hardware will be the same as after the reset.

2. In standby mode, the RAM enters the hold state after a reset.

3. This value is 04H only after a power-on-clear reset.

4. This value is 04H only when V_{DD} < LVI detection voltage.

CHAPTER 16 μPD78F9088

The μPD78F9088 replaces the internal ROM of the μPD789086 and 789088 with flash memory. The differences between the flash memory and the mask ROM versions are shown in Table 16-1.

Table 16-1. Differences Between Flash Memory and Mask ROM Versions

Item		Flash Memory Version	Mask ROM Version	
		μPD78F9088	μPD789086	μPD789088
Internal memory	ROM structure	Flash memory	Mask ROM	
	ROM capacity	32 KB	16 KB	32 KB
	High-speed RAM	320 bytes	256 bytes	320 bytes
	Low-speed RAM	256 bytes	128 bytes	256 bytes
IC pin		Not provided	Provided	
V_{PP} pin		Provided	Not provided	
Electrical characteristics		Refer to **CHAPTER 18 ELECTRICAL SPECIFICATIONS**.		

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

16.1 Flash Memory Characteristics

Flash memory programming is performed by connecting a dedicated flash programmer (Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4)) to the target system with the μPD78F9088 mounted on the target system (on-board). A flash memory program adapter (FA adapter), which is a target board used exclusively for programming, is also provided.

Remark FL-PR3, FL-PR4, and the program adapter are the products made by Naito Densei Machida Mfg. Co., Ltd. (TEL +81-45-475-4191).

Programming using flash memory has the following advantages.

- Software can be modified after the microcontroller is solder-mounted on the target system.
- Distinguishing software facilities low-quantity, varied model production
- Easy data adjustment when starting mass production

16.1.1 Programming environment

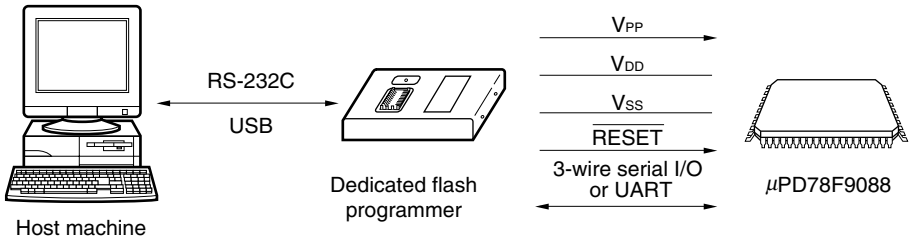
The following shows the environment required for μPD78F9088 flash memory programming.

When Flashpro III (part no. FL-PR3, PG-FP3) or Flashpro IV (part no. FL-PR4, PG-FP4) is used as a dedicated flash programmer, a host machine is required to control the dedicated flash programmer. Communication between the host machine and flash programmer is performed via RS-232C/USB (Rev. 1.1).

For details, refer the manuals for Flashpro III/Flashpro IV.

Remark USB is supported by Flashpro IV only.

Figure 16-1. Environment for Writing Program to Flash Memory



16.1.2 Communication mode

Use the communication mode shown in Table 16-2 to perform communication between the dedicated flash programmer and μPD78F9088.

Table 16-2. Communication Mode List

Communication Mode	TYPE Setting[Note 1]					Pins Used	Number of V_{PP} Pulses
	COMM PORT	SIO Clock	CPU CLOCK		Multiple Rate		
			In Flashpro	On Target Board			
3-wire serial I/O	SIO ch-0 (3-wired, sync.)	100 Hz to 1.25 MHz[Note 2]	1, 2, 4, or 5 MHz[Note 3]	1 to 5 MHz[Note 2]	1.0	SI20/RxD20/P22 SO20/TxD20/P21 $\overline{SCK20}$/ASCK20/P20	0
UART	UART ch-0 (Async.)	4,800 to 76,800 bps [Notes 2, 4]	5 MHz[Note 5]	4.91 or 5 MHz[Note 2]	1.0	RxD20/SI20/P22 TxD20/SO20/P21	8

Notes 1. Selection items for TYPE settings on the dedicated flash programmer (Flashpro III (part no. FL-PR3, PG-FP3)/Flashpro IV (part no. FL-PR4, PG-FP4)).

2. The possible setting range differs depending on the voltage. For details, refer to **CHAPTER 18 ELECTRICAL SPECIFICATIONS**.

3. 2 or 4 MHz only for Flashpro III

4. Because signal wave slew also affects UART communication, in addition to the baud rate error, thoroughly evaluate the slew and baud rate error.

5. Optional for Flashpro IV. However, when using Flashpro III, be sure to select the clock of the resonator on the board. UART cannot be used with the clock supplied by Flashpro III.

Figure 16-2. Communication Mode Selection Format

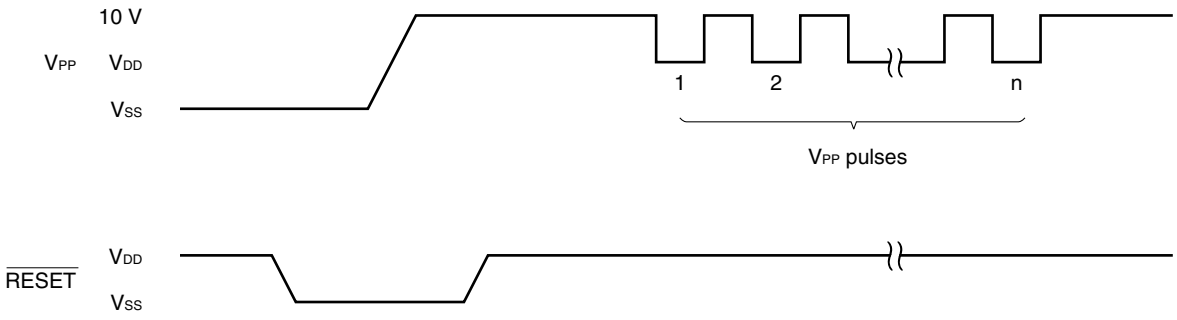
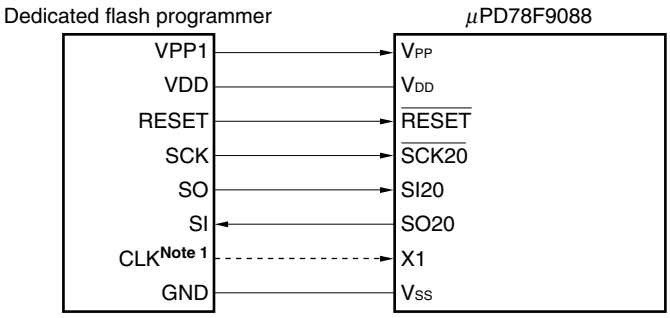
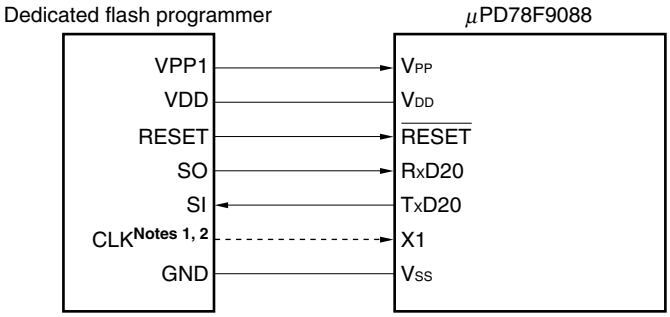


Figure 16-3. Example of Connection with Dedicated Flash Programmer

(a) 3-wire serial I/O



(b) UART



Notes 1. Connect this pin when the system clock is supplied from the dedicated flash programmer. If a resonator is already connected to the X1 pin, the CLK pin does not need to be connected.

2. When using UART with Flashpro III, the clock of the resonator connected to the X1 pin must be used, so connection to the CLK pin is not necessary.

Caution The V_{DD} pin, if already connected to the power supply, must be connected to the VDD pin of the dedicated flash programmer. Before using the power supply connected to the V_{DD} pin, supply voltage before starting programming.

If Flashpro III (FL-PR3, PG-FP3)/Flashpro IV (FL-PR4, PG-FP4) is used as a dedicated flash programmer, the following signals are generated for the μPD78F9088. For details, refer to the manual of Flashpro III/Flashpro IV.

Table 16-3. Pin Connection List

Signal Name	I/O	Pin Function	Pin Name	3-Wire Serial I/O	UART
VPP1	Output	Write voltage	V_{PP}	◎	◎
VPP2	–	–	–	×	×
VDD	I/O	V_{DD} voltage generation/ voltage monitoring	V_{DD}	◎Note	◎Note
GND	–	Ground	V_{SS}	◎	◎
CLK	Output	Clock output	X1	○	○
RESET	Output	Reset signal	$\overline{\text{RESET}}$	◎	◎
SI	Input	Receive signal	SO20/TxD20	◎	◎
SO	Output	Transmit signal	SI20/RxD20	◎	◎
SCK	Output	Transfer clock	$\overline{\text{SCK20}}$	◎	×
HS	Input	Handshake signal	–	×	×

Note V_{DD} voltage must be supplied before programming is started.

Remark ◎: Pin must be connected.
○: If the signal is supplied on the target board, pin need not be connected.
×: Pin need not be connected.

16.1.3 On-board pin processing

When performing programming on the target system, provide a connector on the target system to connect the dedicated flash programmer.

An on-board function that allows switching between normal operation mode and flash memory programming mode may be required in some cases.

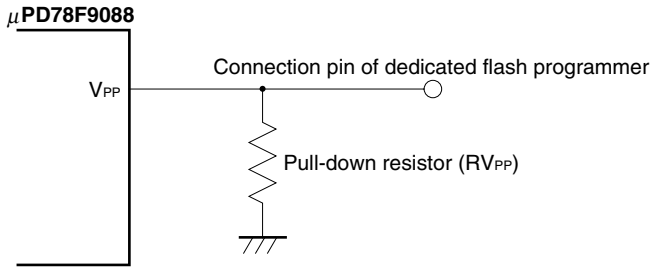
<V_{PP} pin>

In normal operation mode, input 0 V to the V_{PP} pin. In flash memory programming mode, a write voltage of 10.0 V (TYP.) is supplied to the V_{PP} pin, so perform either of the following.

(1) Connect a pull-down resistor (RV_{PP} = 10 kΩ) to the V_{PP} pin.
(2) Use the jumper on the board to switch the V_{PP} pin input to either the writer or directly to GND.

A V_{PP} pin connection example is shown below.

Figure 16-4. V_{PP} Pin Connection Example



<Serial interface pin>

The following shows the pins used by the serial interface.

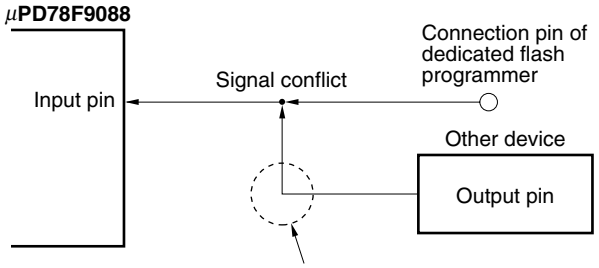
Serial Interface	Pins Used
3-wire serial I/O	SI20, SO20, $\overline{SCK20}$
UART	RxD20, TxD20

When connecting the dedicated flash programmer to a serial interface pin that is connected to another device on-board, signal conflict or abnormal operation of the other devices may occur. Care must therefore be taken with such connections.

(1) Signal conflict

If the dedicated flash programmer (output) is connected to a serial interface pin (input) that is connected to another device (output), a signal conflict occurs. To prevent this, isolate the connection with the other device or set the other device to the output high impedance status.

Figure 16-5. Signal Conflict (Input Pin of Serial Interface)

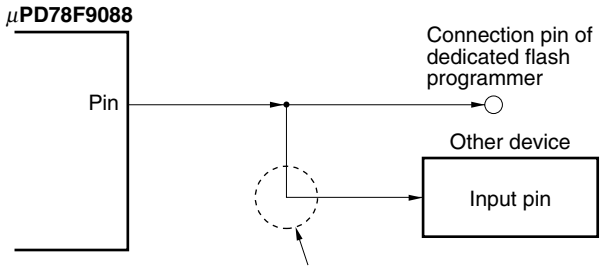


In the flash memory programming mode, the signal output by another device and the signal sent by the dedicated flash programmer conflict, therefore, isolate the signal of the other device.

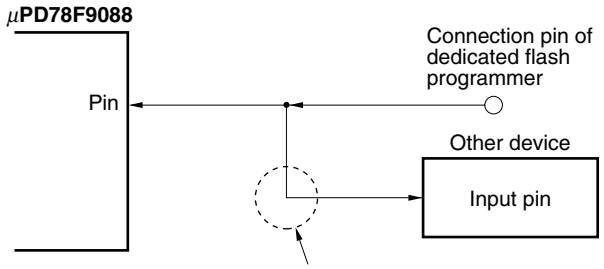
(2) Abnormal operation of other device

If the dedicated flash programmer (output or input) is connected to a serial interface pin (input or output) that is connected to another device (input), a signal is output to the device, and this may cause an abnormal operation. To prevent this abnormal operation, isolate the connection with the other device or set so that the input signals to the other device are ignored.

Figure 16-6. Abnormal Operation of Other Device



If the signal output by the μPD78F9088 affects another device in the flash memory programming mode, isolate the signals of the other device.



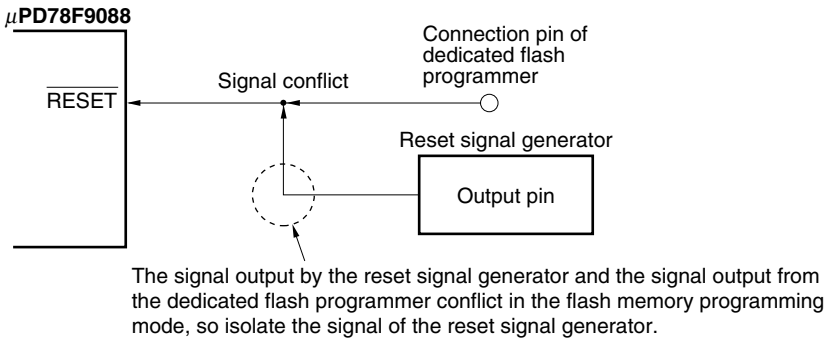
If the signal output by the dedicated flash programmer affects another device in the flash memory programming mode, isolate the signals of the other device.

<$\overline{\text{RESET}}$ pin>

If the reset signal of the dedicated flash programmer is connected to the $\overline{\text{RESET}}$ pin connected to the reset signal generator on-board, a signal conflict occurs. To prevent this, isolate the connection with the reset signal generator.

If the reset signal is input from the user system in the flash memory programming mode, a normal programming operation cannot be performed. Therefore, do not input reset signals from other than the dedicated flash programmer.

Figure 16-7. Signal Conflict ($\overline{\text{RESET}}$ Pin)



<Port pins>

When the μPD78F9088 enters the flash memory programming mode, all the pins other than those that communicate with flash programmer are in the same status as immediately after reset.

If the external device does not recognize initial statuses such as the output high impedance status, therefore, connect the external device to V_{DD} or V_{SS} via a resistor.

<Resonator>

When using the on-board clock, connect X1, X2 as required in the normal operation mode.

When using the clock output of the flash programmer, connect it directly to X1, disconnecting the main resonator on-board, and leave the X2 pin open. The subclock conforms to the normal operation mode.

<Power supply>

To use the power output from the flash programmer, connect the V_{DD} pin to VDD of the flash programmer, and V_{SS} pin to GND of the flash programmer, respectively.

To use the on-board power supply, make connection in accordance with the normal operation mode. However, because the voltage is monitored by the flash programmer, be sure to connect VDD of the flash programmer.

16.1.4 Connection of adapter for flash writing

The following figure shows an example of recommended connection when the adapter for flash writing is used.

Figure 16-8. Wiring Example for Flash Writing Adapter with 3-Wire Serial I/O

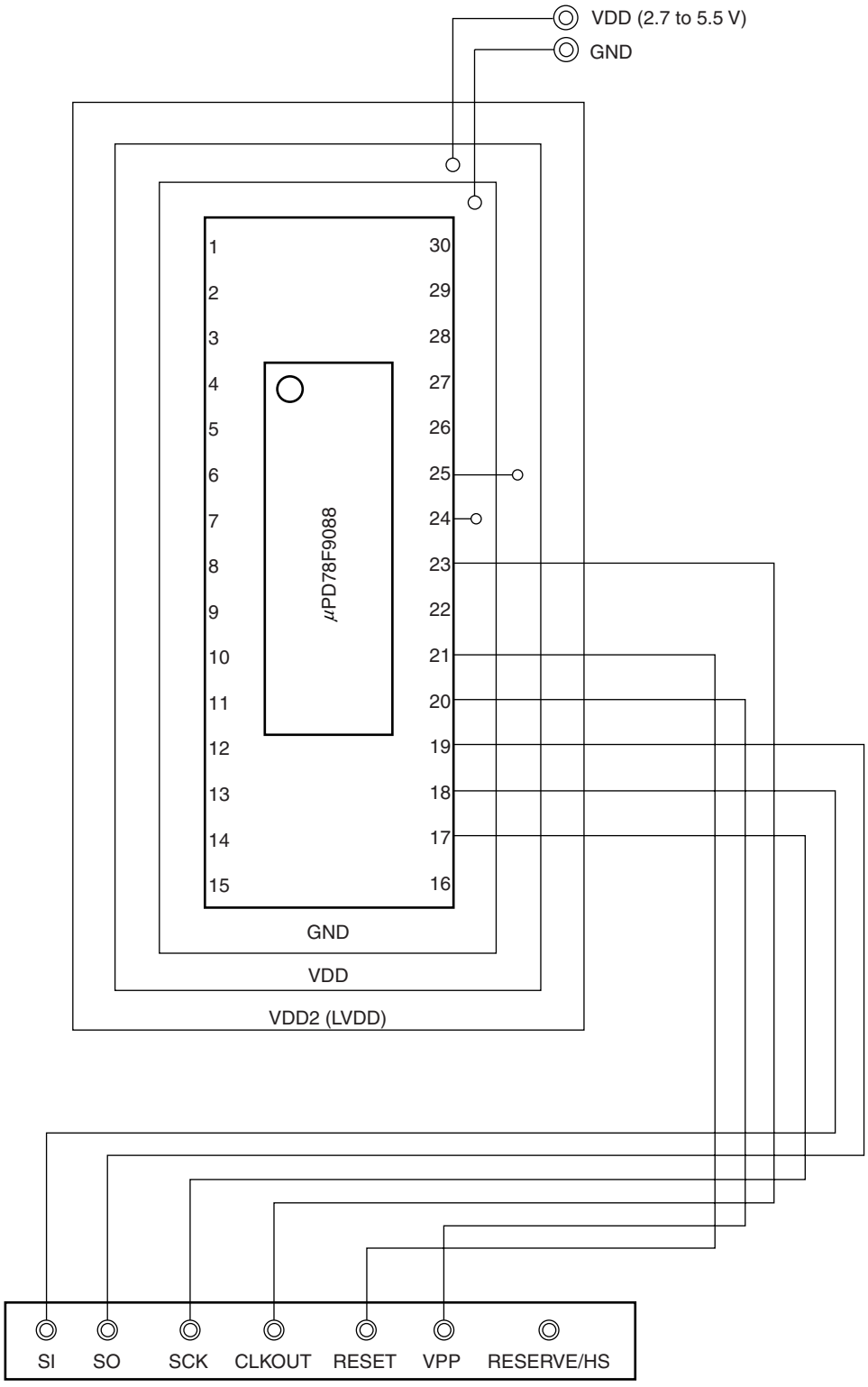
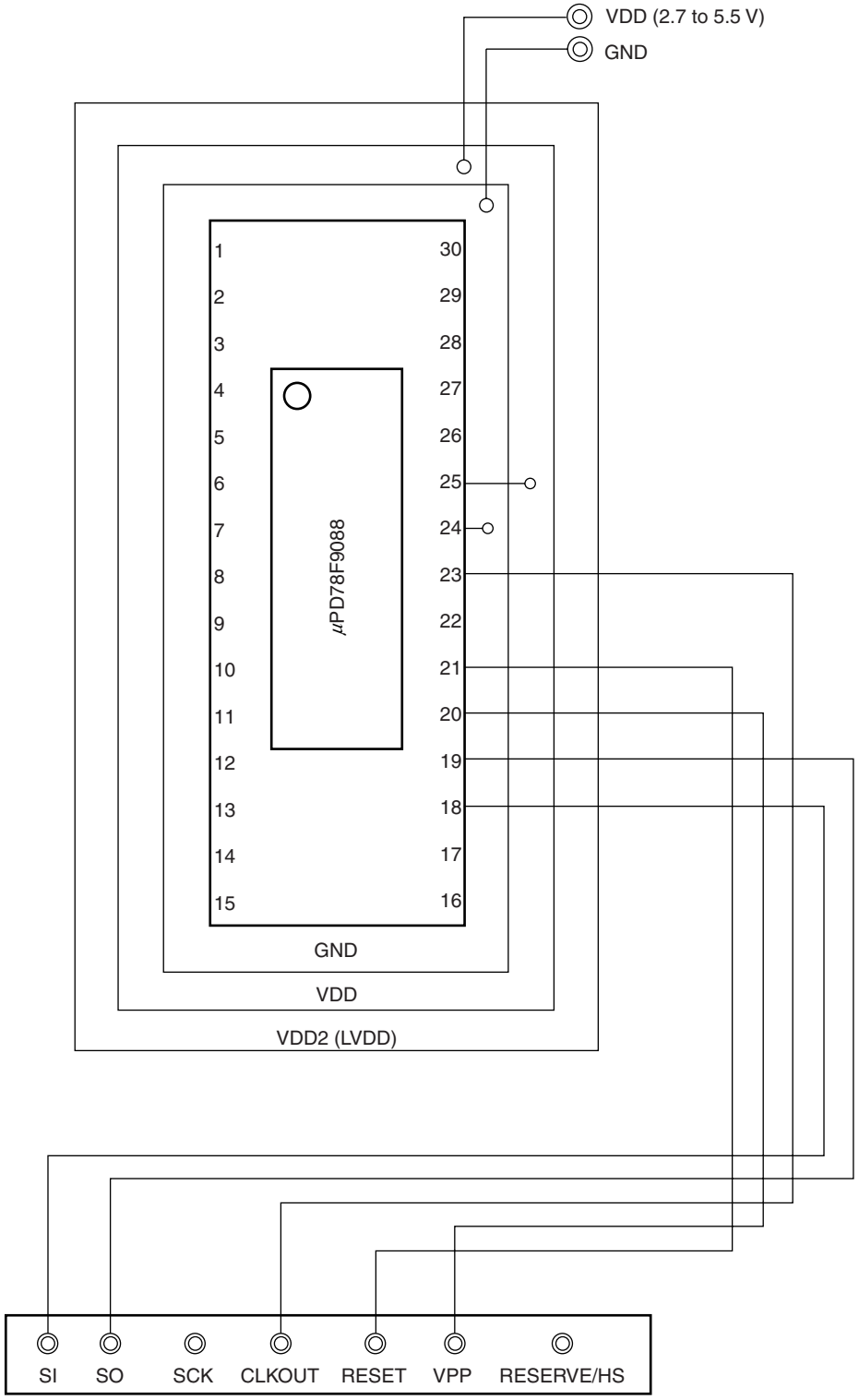


Figure 16-9. Wiring Example for Flash Writing Adapter with UART



CHAPTER 17 INSTRUCTION SET OVERVIEW

This chapter lists the instruction set of the μPD789088 Subseries. For details of the operation and machine language (instruction code) of each instruction, refer to **78K/0S Series User's Manual Instructions (U11047E)**.

17.1 Operation

17.1.1 Operand identifiers and description methods

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

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

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

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

Table 17-1. Operand Identifiers and Description Methods

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

Remark For symbols of special function registers, see **Table 3-4 Special Function Registers**.

17.1.2 Description of "Operation" column

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

17.1.3 Description of "Flag" column

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

17.2 Operation List

Mnemonic	Operand	Bytes	Clocks	Operation	Flag
					Z AC CY
MOV	r, #byte	3	6	r ← byte	
	saddr, #byte	3	6	(saddr) ← byte	
	sfr, #byte	3	6	sfr ← byte	
	A, r Note 1	2	4	A ← r	
	r, A Note 1	2	4	r ← A	
	A, saddr	2	4	A ← (saddr)	
	saddr, A	2	4	(saddr) ← A	
	A, sfr	2	4	A ← sfr	
	sfr, A	2	4	sfr ← A	
	A, !addr16	3	8	A ← (addr16)	
	!addr16, A	3	8	(addr16) ← A	
	PSW, #byte	3	6	PSW ← byte	× × ×
	A, PSW	2	4	A ← PSW	
	PSW, A	2	4	PSW ← A	× × ×
	A, [DE]	1	6	A ← (DE)	
	[DE], A	1	6	(DE) ← A	
	A, [HL]	1	6	A ← (HL)	
	[HL], A	1	6	(HL) ← A	
	A, [HL + byte]	2	6	A ← (HL + byte)	
	[HL + byte], A	2	6	(HL + byte) ← A	
XCH	A, X	1	4	A ↔ X	
	A, r Note 2	2	6	A ↔ r	
	A, saddr	2	6	A ↔ (saddr)	
	A, sfr	2	6	A ↔ sfr	
	A, [DE]	1	8	A ↔ (DE)	
	A, [HL]	1	8	A ↔ (HL)	
	A, [HL, byte]	2	8	A ↔ (HL + byte)	

Notes 1. Except r = A.

2. Except r = A, X.

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

Mnemonic	Operand	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
MOVW	rp, #word	3	6	rp ← word			
	AX, saddrp	2	6	AX ← (saddrp)			
	saddrp, AX	2	8	(saddrp) ← AX			
	AX, rp Note	1	4	AX ← rp			
	rp, AX Note	1	4	rp ← AX			
XCHW	AX, rp Note	1	8	AX ↔ rp			
ADD	A, #byte	2	4	A, CY ← A + byte	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte	×	×	×
	A, r	2	4	A, CY ← A + r	×	×	×
	A, saddr	2	4	A, CY ← A + (saddr)	×	×	×
	A, !addr16	3	8	A, CY ← A + (addr16)	×	×	×
	A, [HL]	1	6	A, CY ← A + (HL)	×	×	×
	A, [HL + byte]	2	6	A, CY ← A + (HL + byte)	×	×	×
ADDC	A, #byte	2	4	A, CY ← A + byte + CY	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) + byte + CY	×	×	×
	A, r	2	4	A, CY ← A + r + CY	×	×	×
	A, saddr	2	4	A, CY ← A + (saddr) + CY	×	×	×
	A, !addr16	3	8	A, CY ← A + (addr16) + CY	×	×	×
	A, [HL]	1	6	A, CY ← A + (HL) + CY	×	×	×
	A, [HL + byte]	2	6	A, CY ← A + (HL + byte) + CY	×	×	×
SUB	A, #byte	2	4	A, CY ← A – byte	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte	×	×	×
	A, r	2	4	A, CY ← A – r	×	×	×
	A, saddr	2	4	A, CY ← A – (saddr)	×	×	×
	A, !addr16	3	8	A, CY ← A – (addr16)	×	×	×
	A, [HL]	1	6	A, CY ← A – (HL)	×	×	×
	A, [HL + byte]	2	6	A, CY ← A – (HL + byte)	×	×	×

Note Only when rp = BC, DE, or HL.

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

Mnemonic	Operand	Bytes	Clocks	Operation	Flag		
					Z	AC	CY
SUBC	A, #byte	2	4	A, CY ← A – byte – CY	×	×	×
	saddr, #byte	3	6	(saddr), CY ← (saddr) – byte – CY	×	×	×
	A, r	2	4	A, CY ← A – r – CY	×	×	×
	A, saddr	2	4	A, CY ← A – (saddr) – CY	×	×	×
	A, !addr16	3	8	A, CY ← A – (addr16) – CY	×	×	×
	A, [HL]	1	6	A, CY ← A – (HL) – CY	×	×	×
	A, [HL + byte]	2	6	A, CY ← A – (HL + byte) – CY	×	×	×
AND	A, #byte	2	4	A ← A ∧ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ∧ byte	×		
	A, r	2	4	A ← A ∧ r	×		
	A, saddr	2	4	A ← A ∧ (saddr)	×		
	A, !addr16	3	8	A ← A ∧ (addr16)	×		
	A, [HL]	1	6	A ← A ∧ (HL)	×		
	A, [HL + byte]	2	6	A ← A ∧ (HL + byte)	×		
OR	A, #byte	2	4	A ← A ∨ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ∨ byte	×		
	A, r	2	4	A ← A ∨ r	×		
	A, saddr	2	4	A ← A ∨ (saddr)	×		
	A, !addr16	3	8	A ← A ∨ (addr16)	×		
	A, [HL]	1	6	A ← A ∨ (HL)	×		
	A, [HL + byte]	2	6	A ← A ∨ (HL + byte)	×		
XOR	A, #byte	2	4	A ← A ⩛ byte	×		
	saddr, #byte	3	6	(saddr) ← (saddr) ⩛ byte	×		
	A, r	2	4	A ← A ⩛ r	×		
	A, saddr	2	4	A ← A ⩛ (saddr)	×		
	A, !addr16	3	8	A ← A ⩛ (addr16)	×		
	A, [HL]	1	6	A ← A ⩛ (HL)	×		
	A, [HL + byte]	2	6	A ← A ⩛ (HL + byte)	×		

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

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

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

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

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

17.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

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

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

Note Except r = A.

(2) 16-bit instructions

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

1st Operand \ 2nd Operand	#word	AX	rp[Note]	saddrp	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	
rp	MOVW	MOVW[Note]				INCW DECW PUSH POP
saddrp		MOVW				
sp		MOVW				

Note Only when rp = BC, DE, or HL.

(3) Bit manipulation instructions

SET1, CLR1, NOT1, BT, BF

1st Operand \ 2nd Operand	$addr16	None
A.bit	BT BF	SET1 CLR1
sfr.bit	BT BF	SET1 CLR1
saddr.bit	BT BF	SET1 CLR1
PSW.bit	BT BF	SET1 CLR1
[HL].bit		SET1 CLR1
CY		SET1 CLR1 NOT1

(4) Call instructions/branch instructions

CALL, CALLT, BR, BC, BNC, BZ, BNZ, DBNZ

1st Operand \ 2nd Operand	AX	!addr16	[addr5]	$addr16
Basic instructions	BR	CALL BR	CALLT	BR BC BNC BZ BNZ
Compound instructions				DBNZ

(5) Other instructions

RET, RETI, NOP, EI, DI, HALT, STOP

CHAPTER 18 ELECTRICAL SPECIFICATIONS

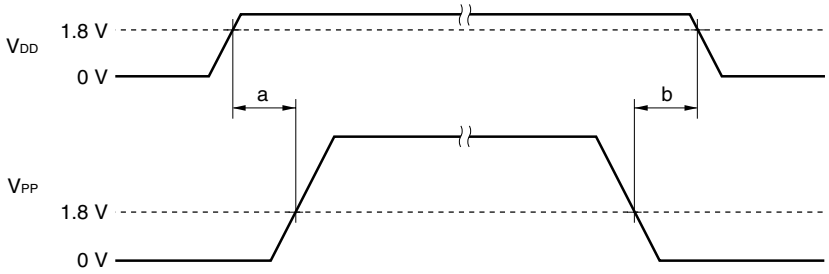
Absolute Maximum Ratings (T_A = 25°C)

Parameter	Symbol	Conditions	Ratings	Unit
Supply voltage	V_{DD}		−0.3 to +6.5	V
	V_{PP}	μPD78F9088[Note 2]	−0.3 to +10.5	V
Input voltage	V_I		−0.3 to V_{DD} + 0.3[Note 1]	V
Output voltage	V_O	P00 to P07, P20 to P27, P40 to P47	−0.3 to V_{DD} + 0.3[Note 1]	V
Output current, high	I_{OH}	Pin P23/TO600	−30	mA
		Per pin (except P23/TO600)	−10	mA
		Total for all pins (except P23/TO600)	−30	mA
Output current, low	I_{OL}	Per pin	30	mA
		Total for all pins (except P23/TO600)	80	mA
Operating ambient temperature	T_A	In normal operation mode	−40 to +85	°C
		During flash memory programming	10 to 40	°C
Storage temperature	T_{stg}	μPD78908x	−65 to +150	°C
		μPD78F9088	−40 to +125	°C

Notes 1. 6.5 V or lower

2. Make sure that the following conditions of the V_{PP} voltage application timing are satisfied when the flash memory is written.

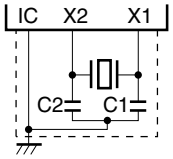
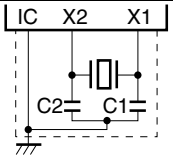
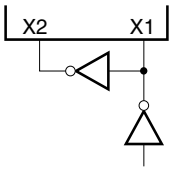
- **When supply voltage rises**
  V_{PP} must exceed V_{DD} 10 μs or more after V_{DD} has reached the lower-limit value (1.8 V) of the operating voltage range (see a in the figure below).
- **When supply voltage drops**
  V_{DD} must be lowered 10 μs or more after V_{PP} falls below the lower-limit value (1.8 V) of the operating voltage range of V_{DD} (see b in the figure below).



Caution Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

System Clock Oscillator Characteristics (T_A = –40 to +85°C, V_{DD} = 1.8 to 5.5 V)

Resonator	Recommended Circuit	Parameter	Conditions	MIN.	TYP.	MAX.	Unit
Ceramic resonator		Oscillation frequency (f_X)Note 1		1.0		5.0	MHz
		Oscillation stabilization timeNote 2	After V_{DD} has reached the MIN. oscillation voltage range			4	ms
Crystal resonator		Oscillation frequency (f_X)Note 1		1.0		5.0	MHz
		Oscillation stabilization timeNote 2	After V_{DD} has reached the MIN. oscillation voltage range			30	ms
External clock		X1 input frequency (f_X)Note 1		1.0		5.0	MHz
		X1 input high-/low-level width (t_{XH}, t_{XL})		85		500	ns

Notes 1. Indicates only oscillator characteristics. Refer to **AC Characteristics** for instruction execution time.

2. Time required to stabilize oscillation after reset or STOP mode release. Use the resonator that stabilizes oscillation during the oscillation wait time.

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

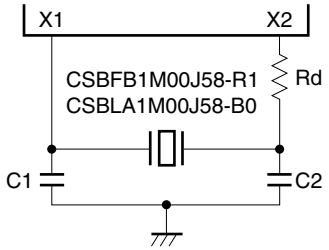
- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with other signal lines.**
- **Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as V_{SS}.**
- **Do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

★ Recommended Oscillator Constant

Ceramic resonator (T_A = –40 to +85°C) (μPD789086, 789088)

Manufacturer	Part Number	Frequency (MHz)	Recommended Circuit Constant (pF)		Oscillation Voltage Range (V_{DD})		Remarks
			C1	C2	MIN.	MAX.	
Murata Mfg. (standard product)	CSBFB1M00J58-R1[Note]	1.0	100	100	2.4	5.5	Rd = 1.0 kΩ
	CSBLA1M00J58-B0[Note]						
	CSTCC2M00G56-R0	2.0	–	–	1.8		On-chip capacitor
	CSTLS2M00G56-B0						
	CSTCR4M00G53-R0	4.0					
	CSTLS4M00G53-B0						
	CSTCR4M19G53-R0	4.194					
	CSTLS4M19G53-B0						
	CSTCR4M91G53-R0	4.915					
	CSTLS4M91G53-B0						
	CSTCR5M00G53-R0	5.0					
	CSTLS5M00G53-B0						
TDK	FCR4.0MC5	4.0	–	–	2.5	5.5	On-chip capacitor
	FCR4.19MC5	4.19					
	FCR5.0MC5	5.0					
Kyocera	PBRC2.00AR-A	2.0	47	47	1.8	5.5	–
	PBRC4.00HR	4.0	–	–			On-chip capacitor
	PBRC4.19HR	4.19					
	PBRC4.91HR	4.91					
	PBRC5.00HR	5.0					

Note A limiting resistor (Rd = 1.0 kΩ) is required when CSBFB1M00J58-R1 or CSBLA1M00J58-B0 (1.0 MHz) manufactured by Murata Mfg. Co., Ltd. is used as the ceramic resonator (see the figure below). This is not necessary when using one of the other recommended resonators.

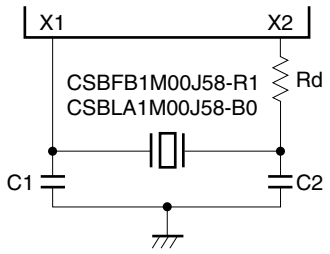


Caution The oscillator constant is a reference value based on evaluation in specific environments by the resonator manufacturer. If the oscillator characteristics need to be optimized in the actual application, request the resonator manufacturer for evaluation on the implementation circuit. Note that the oscillation voltage and oscillation frequency merely indicate the characteristics of the oscillator. Use the internal operation conditions of the μPD789086, 789088 within the specifications of the DC and AC characteristics.

Ceramic resonator (T_A = –40 to +85°C) (μPD78F9088)

Manufacturer	Part Number	Frequency (MHz)	Recommended Circuit Constant (pF)		Oscillation Voltage Range (V_{DD})		Remarks
			C1	C2	MIN.	MAX.	
Murata Mfg. (standard product)	CSBFB1M00J58-R1[Note]	1.0	100	100	2.4	5.5	Rd = 1.0 kΩ
	CSBLA1M00J58-B0[Note]						
	CSTCC2M00G56-R0	2.0	–	–	1.8		On-chip capacitor
	CSTLS2M00G56-B0						
	CSTCR4M00G53-R0	4.0					
	CSTLS4M00G53-B0						
	CSTCR4M19G53-R0	4.194					
	CSTLS4M19G53-B0						
	CSTCR4M91G53-R0	4.915					
	CSTLS4M91G53-B0				1.9		
	CSTCR5M00G53-R0	5.0			1.8		
	CSTLS5M00G53-B0				1.9		
Murata Mfg. (low-voltage drive type)	CSTLS4M91G53093-B0	4.915	–	–	1.8	5.5	On-chip capacitor
	CSTLS5M00G53093-B0	5.0					
TDK	FCR4.0MC5	4.0	–	–	2.5	5.5	On-chip capacitor
	FCR4.19MC5	4.19					
	FCR5.0MC5	5.0					
Kyocera	PBRC2.00AR-A	2.0	47	47	1.8	5.5	–
	PBRC4.00HR	4.0	–	–			On-chip capacitor
	PBRC4.19HR	4.19					
	PBRC4.91HR	4.91					
	PBRC5.00HR	5.0					

Note A limiting resistor (Rd = 1.0 kΩ) is required when CSBFB1M00J58-R1 or CSBLA1M00J58-B0 (1.0 MHz) manufactured by Murata Mfg. Co., Ltd. is used as the ceramic resonator (see the figure below). This is not necessary when using one of the other recommended resonators.



Caution The oscillator constant is a reference value based on evaluation in specific environments by the resonator manufacturer. If the oscillator characteristics need to be optimized in the actual application, request the resonator manufacturer for evaluation on the implementation circuit. Note that the oscillation voltage and oscillation frequency merely indicate the characteristics of the oscillator. Use the internal operation conditions of the μPD78F9088 within the specifications of the DC and AC characteristics.

DC Characteristics (T_A = –40 to +85°C, V_{DD} = 1.8 to 5.5 V) (1/2)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
★ Output current, low	I_{OL}	Per pin (except P23/TO600)				10	mA
		P23/TO600	V_{DD} = 3.0 V, V_{OL} = 1.0 V	7	15	24	mA
★		Total for all pins (except P23/TO600)				80	mA
★ Output current, high	I_{OH}	Per pin (except P23/TO600)				–1	mA
		P23/TO600	V_{DD} = 3.0 V, V_{OH} = 2.0 V	–7	–15	–24	mA
★		Total for all pins (except P23/TO600)				–15	mA
Input voltage, high	V_{IH1}	P00 to P07, P21, P23, P25 to P27	V_{DD} = 2.7 to 5.5 V	0.7 V_{DD}		V_{DD}	V
			V_{DD} = 1.8 to 2.7 V	0.8 V_{DD}		V_{DD}	V
	V_{IH2}	$\overline{RESET}$, P20, P22, P24, P40 to P47	V_{DD} = 1.8 to 5.5 V	0.8 V_{DD}		V_{DD}	V
	V_{IH3}	X1, X2		V_{DD} – 0.1		V_{DD}	V
Input voltage, low	V_{IL1}	P00 to P07, P21, P23, P25 to P27	V_{DD} = 2.7 to 5.5 V	0		0.3 V_{DD}	V
			V_{DD} = 1.8 to 2.7 V	0		0.2 V_{DD}	V
	V_{IL2}	$\overline{RESET}$, P20, P22, P24, P40 to P47	V_{DD} = 1.8 to 5.5 V	0		0.2 V_{DD}	V
	V_{IL3}	X1, X2		0		0.1	V
Output voltage, high	V_{OH11}	P00 to P07, P20 to P22, P24 to P27, P40 to P47	1.8 ≤ V_{DD} ≤ 5.5 V, I_{OH} = –100 μA	V_{DD} – 0.5			V
	V_{OH12}		1.8 ≤ V_{DD} ≤ 5.5 V, I_{OH} = –500 μA	V_{DD} – 0.7			V
	V_{OH21}	P23/TO600	1.8 ≤ V_{DD} ≤ 5.5 V, I_{OH} = –400 μA	V_{DD} – 0.5			V
	V_{OH22}		1.8 ≤ V_{DD} ≤ 5.5 V, I_{OH} = –2 mA	V_{DD} – 0.7			V
Output voltage, low	V_{OL11}	P00 to P07, P20 to P27, P40 to P47	1.8 ≤ V_{DD} ≤ 5.5 V, I_{OL} = 400 μA			0.5	V
	V_{OL12}		1.8 ≤ V_{DD} ≤ 5.5 V, I_{OL} = 2 mA			0.7	V

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

DC Characteristics (T_A = –40 to +85°C, V_{DD} = 1.8 to 5.5 V) (2/2)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Input leakage current, high	I_{LIH1}	$V_{IN} = V_{DD}$	P00 to P07, P20 to P27, $\overline{\text{RESET}}$, P40 to P47			3	μA
	I_{LIH2}		X1, X2			20	μA
Input leakage current, low	I_{LIL1}		P00 to P07, P20 to P27, $\overline{\text{RESET}}$, P40 to P47			–3	μA
	I_{LIL2}		X1, X2			–20	μA
Software pull-up resistors	R_1	P00 to P07		50	100	200	kΩ
	R_2	P40 to P47		20	35	50	kΩ
Supply current[Note] Ceramic/crystal oscillation (μPD78908x)	I_{DD1}	5.0 MHz crystal oscillation operating mode	V_{DD} = 5.5 V		1.6	2.5	mA
	I_{DD2}	5.0 MHz crystal oscillation HALT mode	V_{DD} = 5.5 V		0.6	1.2	mA
	I_{DD3}	STOP mode (POC circuit operating)	V_{DD} = 5.5 V		1.5	10	μA
Supply current[Note] Ceramic/crystal oscillation (μPD78F9088)	I_{DD1}	5.0 MHz crystal oscillation operating mode	V_{DD} = 5.5 V		5.0	10	mA
	I_{DD2}	5.0 MHz crystal oscillation HALT mode	V_{DD} = 5.5 V		0.6	2.4	mA
	I_{DD3}	STOP mode (POC circuit operating)	V_{DD} = 5.5 V		2.0	20	μA

Note Current flowing through ports (including current flowing through on-chip pull-up resistors) is not included.

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

AC Characteristics

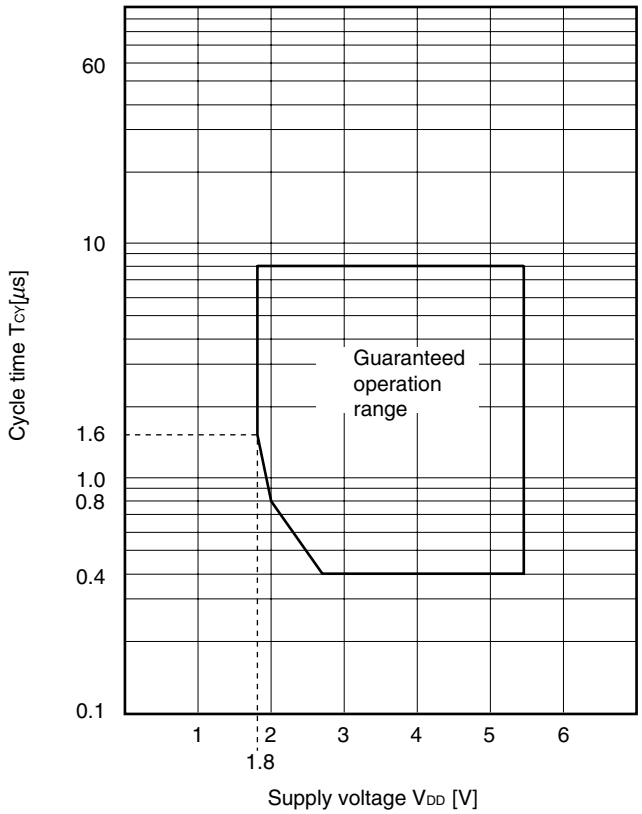
(1) Basic operation (T_A = –40 to +85°C, V_{DD} = 1.8 to 5.5 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Cycle time (minimum instruction execution time)	T_{CY}	V_{DD} = 2.7 to 5.5 V[Note 1]		0.4		8	μs
		V_{DD} = 2.0 to 2.7 V[Note 2]	μPD78908x	0.8		8	μs
			μPD78F9088	0.8		8	μs
		V_{DD} = 1.8 to 2.0 V[Note 2]		1.6		8	μs
CPT20 input high-/low-level width	t_{CPH}, t_{CPL}			10			μs
Interrupt input high-/low-level width	t_{INTH}, t_{INTL}	INTP0, INTP1		10			μs
Key return pin low-level width	T_{KRIL}	KR00 to KR07		10			μs
$\overline{\text{RESET}}$ low-level width	t_{RSL}			10			μs

Notes 1. When the x2 clock is used via the clock multiple circuit, the operating voltage range will be 2.0 to 3.6 V.

2. When the POC circuit is ued, the POC voltage will be the minimum operating voltage.

T_{CY} vs V_{DD}



(2) Serial interface (T_A = –40 to +85°C, V_{DD} = 1.8 to 5.5 V)

(i) 3-wire serial I/O mode ($\overline{SCK20}$...Internal clock output)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
$\overline{SCK20}$ cycle time	t_{KCY1}	V_{DD} = 2.7 to 5.5 V		800			ns
		V_{DD} = 1.8 to 5.5 V		3200			ns
$\overline{SCK20}$ high-/low-level width	t_{KH1}, t_{KL1}	V_{DD} = 2.7 to 5.5 V		$t_{KCY1}/2 - 50$			ns
		V_{DD} = 1.8 to 5.5 V		$t_{KCY1}/2 - 150$			ns
SI20 setup time (to $\overline{SCK20}$↑)	t_{SIK1}	V_{DD} = 2.7 to 5.5 V		150			ns
		V_{DD} = 1.8 to 5.5 V		500			ns
SI20 hold time (from $\overline{SCK20}$↑)	t_{KSI1}	V_{DD} = 2.7 to 5.5 V		400			ns
		V_{DD} = 1.8 to 5.5 V		600			ns
SO20 output delay time from $\overline{SCK20}$↓	t_{KSO1}	R = 1 kΩ, C = 100 pF[Note]	V_{DD} = 2.7 to 5.5 V	0		250	ns
			V_{DD} = 1.8 to 5.5 V	0		1000	ns

Note R and C are the load resistance and load capacitance of the SO output line.

(ii) 3-wire serial I/O mode ($\overline{SCK20}$...External clock input)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
$\overline{SCK20}$ cycle time	t_{KCY2}	V_{DD} = 2.7 to 5.5 V		800			ns
		V_{DD} = 1.8 to 5.5 V		3200			ns
$\overline{SCK20}$ high-/low-level width	t_{KH2}, t_{KL2}	V_{DD} = 2.7 to 5.5 V		400			ns
		V_{DD} = 1.8 to 5.5 V		1600			ns
SI20 setup time (to $\overline{SCK20}$↑)	t_{SIK2}	V_{DD} = 2.7 to 5.5 V		100			ns
		V_{DD} = 1.8 to 5.5 V		150			ns
SI20 hold time (from $\overline{SCK20}$↑)	t_{KSI2}	V_{DD} = 2.7 to 5.5 V		400			ns
		V_{DD} = 1.8 to 5.5 V		600			ns
SO20 output delay time from $\overline{SCK20}$↓	t_{KSO2}	R = 1 kΩ, C = 100 pF[Note]	V_{DD} = 2.7 to 5.5 V	0		300	ns
			V_{DD} = 1.8 to 5.5 V	0		1000	ns

Note R and C are the load resistance and load capacitance of the SO output line.

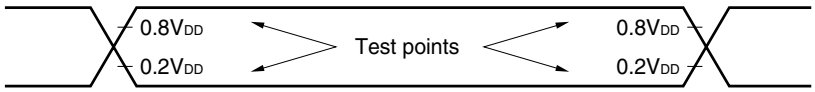
(iii) UART mode (Dedicated baud rate generator output)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate		V_{DD} = 2.7 to 5.5 V			78125	bps
		V_{DD} = 1.8 to 5.5 V			19531	bps

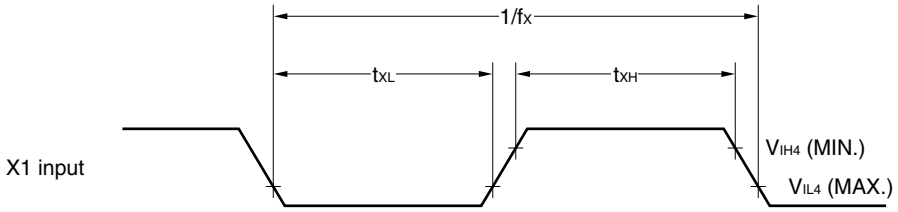
(iv) UART mode (external clock input)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
ASCK20 cycle time	t_{KCY3}	V_{DD} = 2.7 to 5.5 V	800			ns
		V_{DD} = 1.8 to 5.5 V	3200			ns
ASCK20 high-/low-level width	t_{KH3}, t_{KL3}	V_{DD} = 2.7 to 5.5 V	400			ns
		V_{DD} = 1.8 to 5.5 V	1600			ns
Transfer rate		V_{DD} = 2.7 to 5.5 V			39063	bps
		V_{DD} = 1.8 to 5.5 V			9766	bps
ASCK20 rise/fall time	t_R, t_F				1	μs

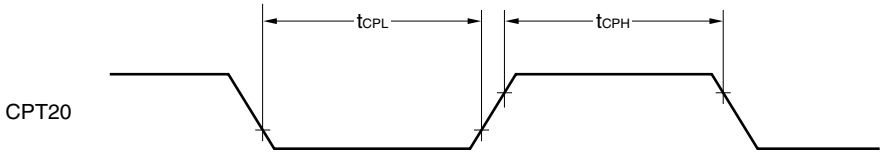
AC Timing Test Points (Excluding X1 Input)



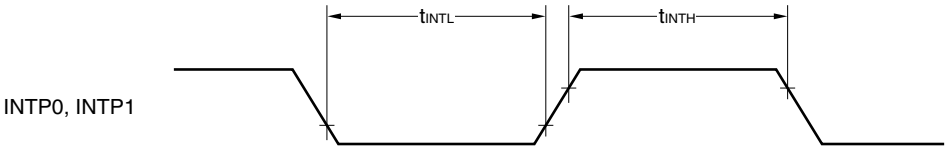
Clock Timing



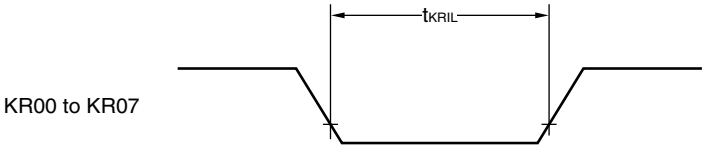
Capture Input Timing



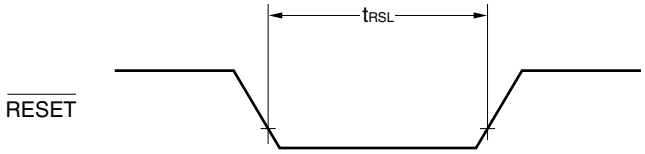
Interrupt Input Timing



Key Return Input Timing

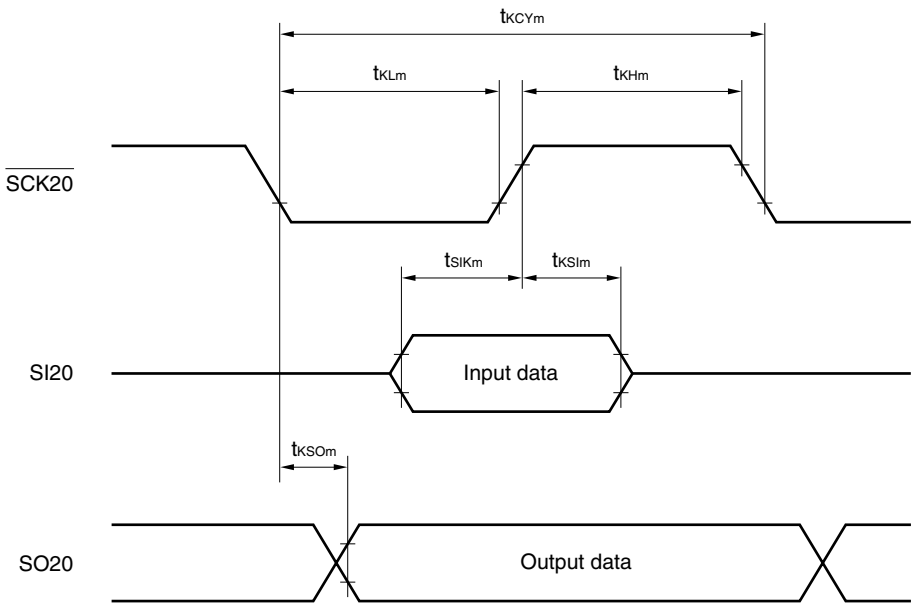


$\overline{\text{RESET}}$ Input Timing



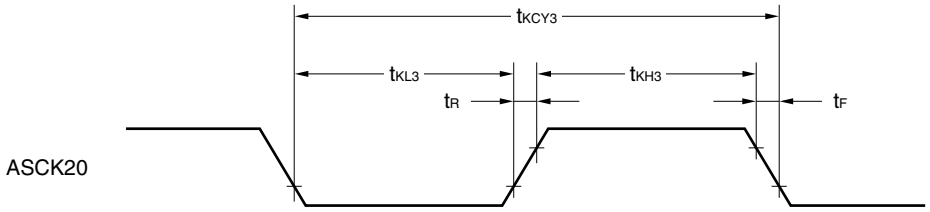
Serial Transfer Timing

3-wire serial I/O mode:



m = 1, 2

UART mode (external clock input):



Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics
(T_A = –40 to +85°C, V_{DD} = 1.8 to 5.5 V)

Parameter		Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Data retention supply voltage		V_{DDDR}			1.2		5.5	V
Low voltage detection voltage	POC	V_{POC}	Response time: 2 ms[Note 1]	μPD78908x	2.0	2.15	2.3	V
				μPD78F9088	2.0	2.15	2.3	V
	LVI	V_{LVI}			1.4	1.5	1.6	V
Power supply rise time		t_{PTh}	V_{DD}: 0 V → 1.8 V		0.01		100	ms
Release signal set time		t_{SREL}	STOP cancelled by $\overline{RESET}$		10			μs
Oscillation stabilization wait time[Note 2]		t_{WAIT}	Cancelled by $\overline{RESET}$			2^{15}/fx		s
			Cancelled by interrupt request			**Note 3**		s

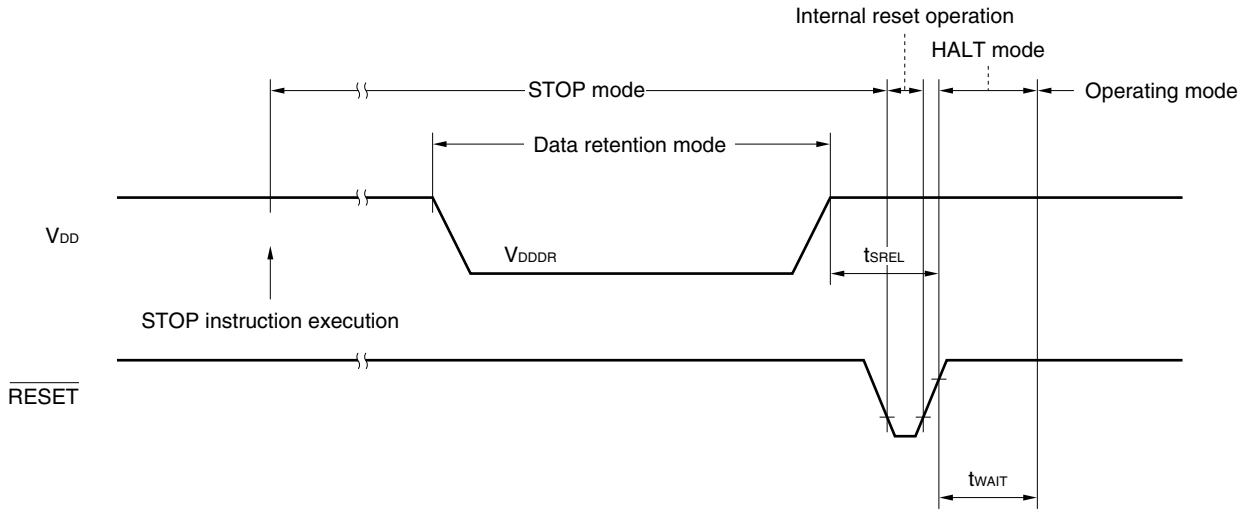
Notes 1. The response time is the time until the output is inverted following detection of voltage by POC, or the time until operation stabilizes after the shift from the operation stopped state to the operating state.

2. The oscillation stabilization wait time is the amount of time the CPU operation is stopped in order to avoid unstable operation at the start of oscillation. Program operation does not start until both the oscillation stabilization wait time and the time until oscillation starts have elapsed.

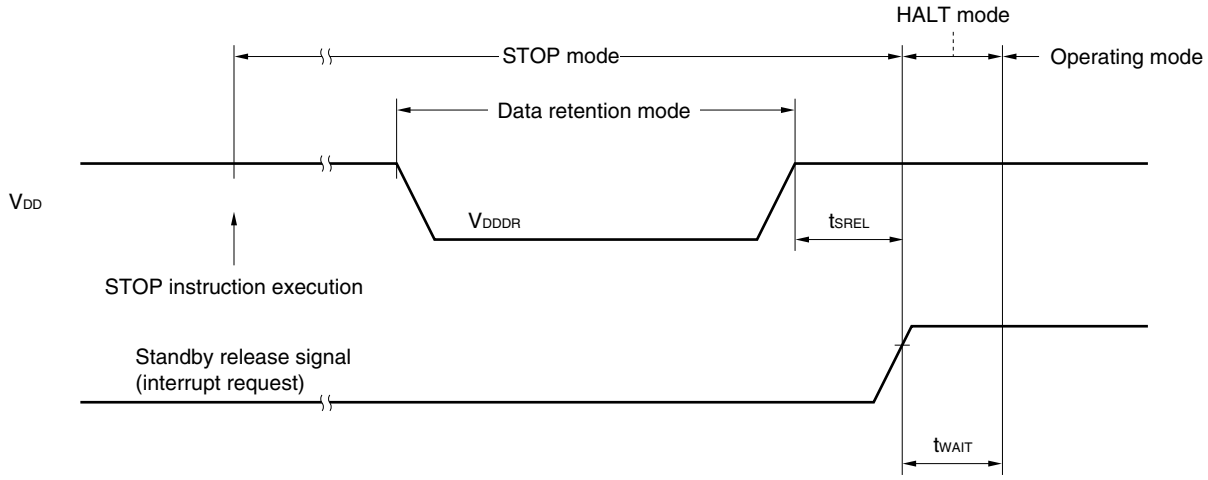
3. 2^{12}/fx, 2^{15}/fx, or 2^{17}/fx can be selected using bits 0 to 2 (OSTS0 to OSTS2) of the oscillation stabilization time selection register (OSTS).

Remark fx: System clock oscillation frequency

Data Retention Timing (STOP Mode Release by $\overline{RESET}$)



Data Retention Timing (Standby Release Signal: STOP Mode Release by Interrupt Signal)

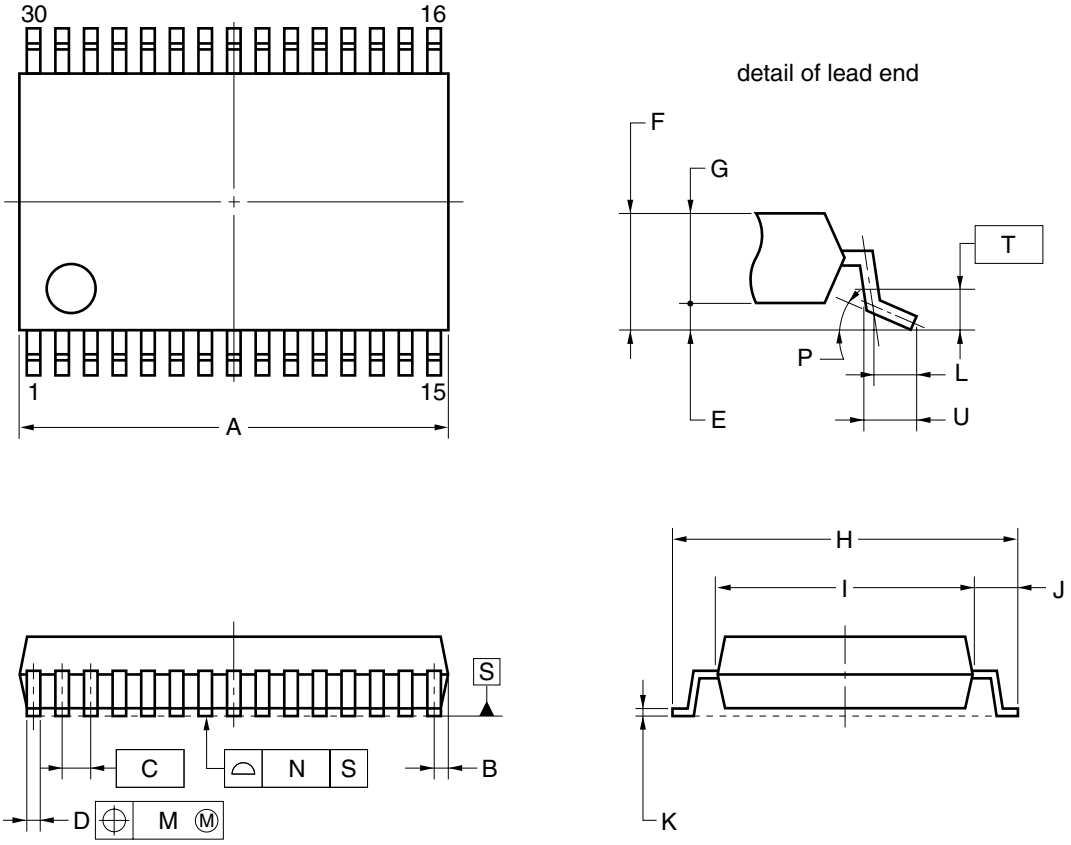


Writing and Erasing Characteristics (T_A = 10 to 40°C, V_{DD} = 1.8 to 5.5 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Write operating frequency	f_X	V_{DD} = 2.7 to 5.5 V	1		5	MHz
		V_{DD} = 2.0 to 5.5 V	1		2.5	MHz
		V_{DD} = 1.8 to 5.5 V	1		1.25	MHz
Write current (V_{DD} pin)	I_{DDW}	When V_{PP} supply voltage = V_{PP1} (at 5.0 MHz operation)			13[Note]	mA
Write current (V_{PP} pin)	I_{PPW}	When V_{PP} supply voltage = V_{PP1}			7.5	mA
Erase current (V_{DD} pin)	I_{DDE}	When V_{PP} supply voltage = V_{PP1} (at 5.0 MHz operation)			13[Note]	mA
Erase current (V_{PP} pin)	I_{PPE}	When V_{PP} supply voltage = V_{PP1}			100	mA
Unit erase time	t_{er}		1	1	1	s
Total erase time	t_{era}				20	s
Number of overwrites		Erase and write is considered as 1 cycle			20	times
V_{PP} supply voltage	V_{PP0}	Normal operation	0		0.2V_{DD}	V
	V_{PP1}	Flash memory programming	9.7	10.0	10.3	V

Note Excludes current flowing through ports (including on-chip pull-up resistors)

30-PIN PLASTIC SSOP (7.62 mm (300))



NOTE

Each lead centerline is located within 0.13 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	9.85±0.15
B	0.45 MAX.
C	0.65 (T.P.)
D	$0.24^{+0.08}_{-0.07}$
E	0.1±0.05
F	1.3±0.1
G	1.2
H	8.1±0.2
I	6.1±0.2
J	1.0±0.2
K	0.17±0.03
L	0.5
M	0.13
N	0.10
P	$3^{\circ}{}^{+5^{\circ}}_{-3^{\circ}}$
T	0.25
U	0.6±0.15

S30MC-65-5A4-2

CHAPTER 20 RECOMMENDED SOLDERING CONDITIONS

The μPD789088 Subseries should be soldered and mounted under the following recommended conditions.

For soldering methods and conditions other than those recommended below, contact an NEC Electronics sales representative.

For technical information, see the following website.

Semiconductor Device Mount Manual (http://www.necel.com/pkg/en/mount/index.html)

Table 20-1. Surface Mounting Type Soldering Conditions

μPD789086MC-×××-5A4: 30-pin plastic SSOP (7.62 mm (300))
μPD789088MC-×××-5A4: 30-pin plastic SSOP (7.62 mm (300))
μPD78F9088M1MC-5A4: 30-pin plastic SSOP (7.62 mm (300))

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C, Time: 30 seconds max. (at 210°C or higher), Count: Twice or less, Exposure limit: 3 days[Note] (after that, prebake at 125°C for 10 hours)	IR35-103-2
VPS	Package peak temperature: 215°C, Time: 40 seconds max. (at 200°C or higher), Count: Twice or less, Exposure limit: 3 days[Note] (after that, prebake at 125°C for 10 hours)	VP15-103-2
Wave soldering	Soldering bath temperature: 260°C max., Time: 10 seconds max., Count: 1, Preheating temperature: 120°C max. (package surface temperature), Exposure limit: 3 days[Note] (after that, prebake at 125°C for 10 hours)	WS60-103-1
Partial heating	Pin temperature: 300°C max., Time: 3 seconds max. (per pin row)	–

Note After opening the dry peak, store it at 25°C or less and 65% RH or less for the allowable storage period.

Caution Do not use different soldering methods together (except for partial heating).

APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for development of systems using the μPD789088 Subseries.
Figure A-1 shows development tools.

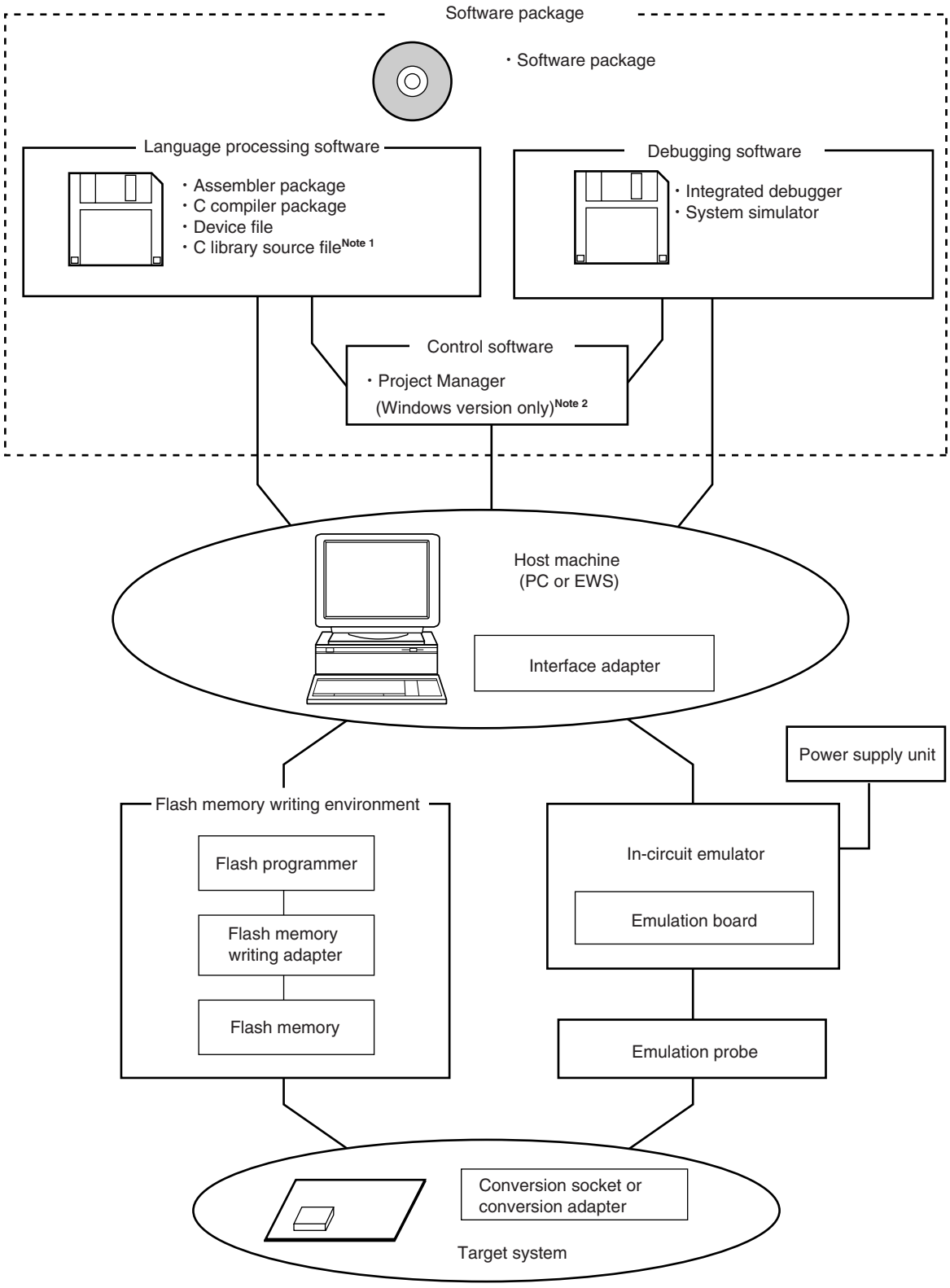
- Support to PC98-NX Series

  Unless specified otherwise, the products supported by IBM PC/AT™ compatibles can be used in PC98-NX Series. When using the PC98-NX Series, refer to the explanation of IBM PC/AT compatibles.

- Windows™

  Unless specified otherwise, “Windows” indicates the following operating systems.
  - Windows 3.1
  - Windows 95
  - Windows 98
  - Windows NT™ Ver.4.0
  - Windows 2000
  - Windows XP

Figure A-1. Development Tools



Notes 1. C library source file is not included in the software package.

2. Project Manager is included in the assembler package.
Project Manager is used only in the Windows environment.

A.1 Software Package

SP78K0S Software package	Software tools for development of the 78K/0S Series are combined in this package. The following tools are included. RA78K0S, CC78K0S, ID78K0S-NS, SM78K0S, and device files
	Part number: μS××××SP78K0S

Remark ×××× in the part number differs depending on the OS used

μS××××SP78K0S

××××	Host Machine	OS	Supply Media
AB17	PC-9800 series, IBM PC/AT compatible	Japanese Windows	CD-ROM
BB17		English Windows	

A.2 Language Processing Software

RA78K0S Assembler package	Program that converts program written in mnemonic into object codes that can be executed by microcontroller. In addition, automatic functions to generate symbol table and optimize branch instructions are also provided. Used in combination with optional device file (DF789088). **<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 (DF789088). **<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
DF789088[Note 1] Device file	File containing the information inherent to the device. Used in combination with optional RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S.
	Part number: μS××××DF789088
CC78K0S-L[Note 2] C library source file	Source file of functions for generating object library included in C compiler package. Necessary for changing object library included in C compiler package according to customer's specifications. Since this is a source file, its working environment does not depend on any particular operating system.
	Part number: μS××××CC78K0S-L

Notes 1. DF789088 is a common file that can be used with RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S.

2. CC78K0S-L is not included in the software package (SP78K0S).

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

μS××××RA78K0S
μS××××CC78K0S

××××	Host Machine	OS	Supply Media
AB13	PC-9800 series, IBM PC/AT compatible	Japanese Windows	3.5" 2HD FD
BB13		English Windows	
AB17		Japanese Windows	CD-ROM
BB17		English Windows	
3P17	HP9000 series 700™	HP-UX™ (Rel. 10.10)	
3K17	SPARCstation™	SunOS™ (Rel. 4.1.4), Solaris™ (Rel. 2.5.1)	

μS××××DF789088
μS××××CC78K0S-L

××××	Host Machine	OS	Supply Media
AB13	PC-9800 series, IBM PC/AT compatible	Japanese Windows	3.5" 2HD FD
BB13		Japanese Windows	
3P16	HP9000 series 700	HP-UX™ (Rel. 10.10)	DAT
3K13	SPARCstation	SunOS™ (Rel. 4.1.4),	3.5" 2HD FD
3K15		Solaris™ (Rel. 2.5.4)	1/4-inch CGMT

A.3 Control Software

Project Manager	Control software created for efficient development of the user program in the Windows environment. User program development operations such as editor startup, build, and debugger startup can be performed from the Project Manager. <Caution> The Project Manager is included in the assembler package (RA78K0S). The Project Manager is used only in the Windows environment.

A.4 Flash Memory Writing Tools

Flashpro III (FL-PR3, PG-FP3) Flashpro IV (FL-PR4, PG-FP4) Flash programmer	Dedicated flash programmer for microcomputers incorporating flash memory
FA-30MC Flash memory writing adapter	Adapter for writing to flash memory and connected to Flashpro III or Flashpro IV. • FA-30MC: For 30-pin plastic SSOP (MC-5A4 type)

Remark The FL-PR3, FL-PR4, and FA-30MC are products made by Naito Densei Machida Mfg. Co., Ltd. (TEL +81-45-475-4191).

A.5 Debugging Tools (Hardware)

IE-78K0S-NS In-circuit emulator		In-circuit emulator for debugging hardware and software of application system using 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-78K0S-NS-A In-circuit emulator		The IE-78K0S-NS-A provides a coverage function in addition to the IE-78K0S-NS functions, thus enhancing the debug functions, including the tracer and timer functions.
IE-70000-MC-PS-B AC adapter		Adapter for supplying power from AC 100 to 240 V outlet.
IE-70000-98-IF-C Interface adapter		Adapter necessary when using PC-9800 series PC (except notebook type) as host machine (C bus supported)
IE-70000-CD-IF-A PC card interface		PC card and interface cable necessary when using notebook PC as host machine (PCMCIA socket supported)
IE-70000-PC-IF-C Interface adapter		Interface adapter necessary when using IBM PC/AT compatible as host machine (ISA bus supported)
IE-70000-PCI-IF-A Interface adapter		Adapter necessary when using personal computer incorporating PCI bus as host machine
IE-789088-NS-EM1 Emulation board		Board for emulating the peripheral hardware inherent to the device. Used in combination with in-circuit emulator.
NP-30GS NP-H36GS Emulation probe		Cable to connect the in-circuit emulator and target system. Used in combination with the NGS-30 when supporting 30-pin plastic SSOP (MC-5A4 type).
	NGS-30 Conversion socket	Conversion socket to connect the NP-36GS or NP-H36GS and a target system board on which a 30-pin plastic SSOP (MC-5A4 type) can be mounted.
NP-30MC Emulation probe		Cable to connect the in-circuit emulator and target system. Used in combination with the YSPACK30BK or NSPACK30BK.
	YSPACK30BK NSPACK30BK Conversion adapter	Conversion adapter to connect the NP-30MC and a target system board on which a 30-pin plastic SSOP (MC-5A4 type) can be mounted

Remarks 1. The NP-36GS, NP-H36GS, NP-30MC, and NGS-30 are products made by Naito Densei Machida Mfg. Co., Ltd. (TEL +81-45-475-4191).

2. The YSPACK30BK and NSPACK30BK are products made by TOKYO ELETECH CORPORATION.
For further information, contact: Daimaru Kogyo, Ltd.
Tokyo Electronics Department (TEL +81-3-3820-7112)
Osaka Electronics Department (TEL +81-6-6244-6672)

A.6 Debugging Tools (Software)

ID78K0S-NS Integrated debugger	This debugger supports the in-circuit emulators IE-78K0S-NS and IE-78K0S-NS-A for the 78K/0S Series. The ID78K0S-NS is Windows-based software. It has improved C-compatible debugging functions and can display the results of tracing with the source program using an integrating window function that associates the source program, disassemble display, and memory display with the trace result. Used in combination with optional device file (DF789088).
	Part number: μS××××ID78K0S-NS
SM78K0S System simulator	This is a system simulator for the 78K/0S Series. The SM78K0S is Windows-based software. It can be used to debug the target system at C source level or assembler level while simulating the operation of the target system on the host machine. Using SM78K0S, the logic and performance of the application can be verified independently of hardware development. Therefore, the development efficiency can be enhanced and the software quality can be improved. Used in combination with optional device file (DF789088).
	Part number: μS××××SM78K0S
DF789088[Note] Device file	File containing the information inherent to the device. Used in combination with the optional RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S.
	Part number: μS××××DF789088

Note DF789088 is a common file that can be used with RA78K0S, CC78K0S, ID78K0S-NS, and SM78K0S.

Remark ×××× in the part number differs depending on the operating systems and supply medium to be used.

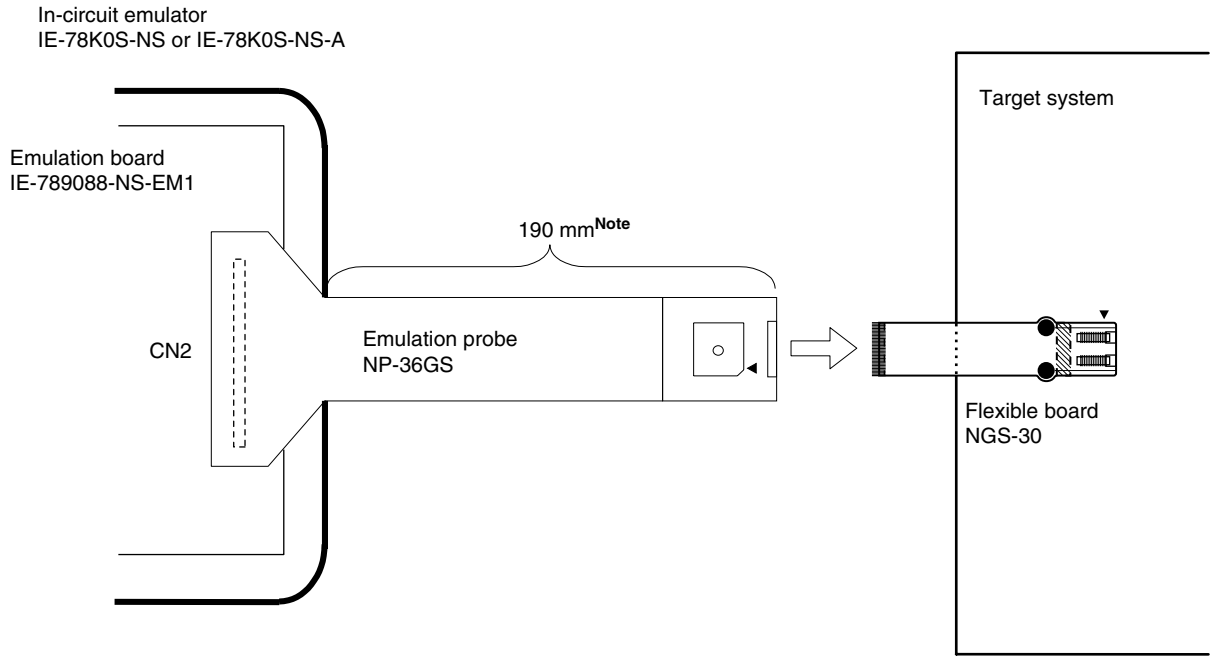
μS××××ID78K0S-NS
μS××××SM78K0S

××××	Host Machine	OS	Supply Media
AB13	PC-9800 series IBM PC/AT compatibles	Japanese Windows	3.5" 2HD FD
BB13		English Windows	
AB17		Japanese Windows	CD-ROM
BB17		English Windows	

APPENDIX B NOTES ON TARGET SYSTEM DESIGN

Figures B-1 to B-4 show the conditions when connecting the emulation probe to the flexible board and the conversion adapter. Follow the configuration below and consider the shape of parts to be mounted on the target system when designing a system.

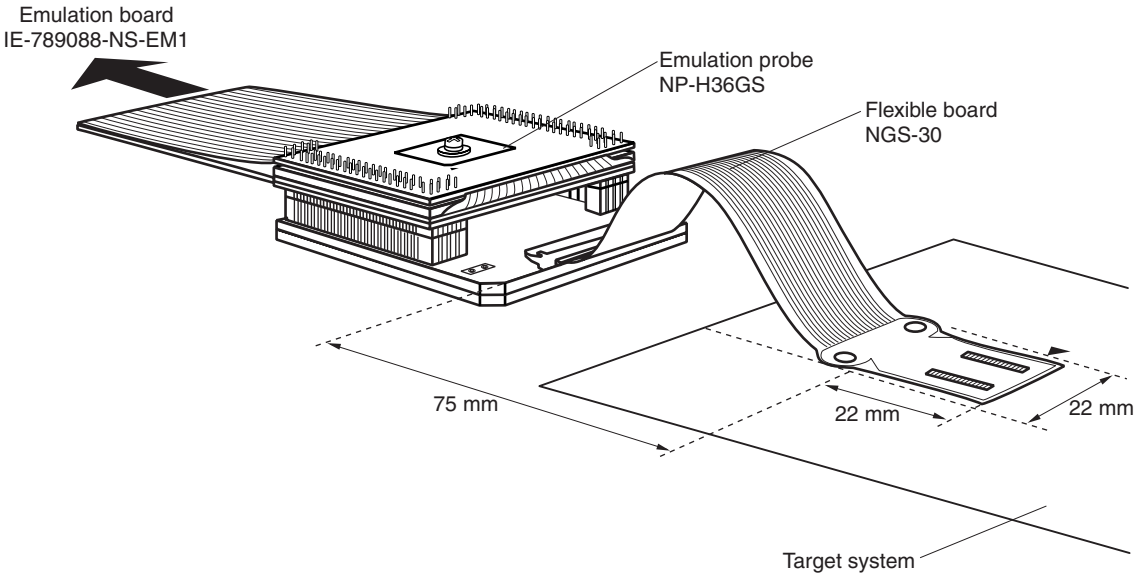
Figure B-1. Distance Between In-Circuit Emulator and Flexible Board (NP-36GS)



Note Distance when NP-36GS is used. When NP-H36GS is used, the distance is 390 mm.

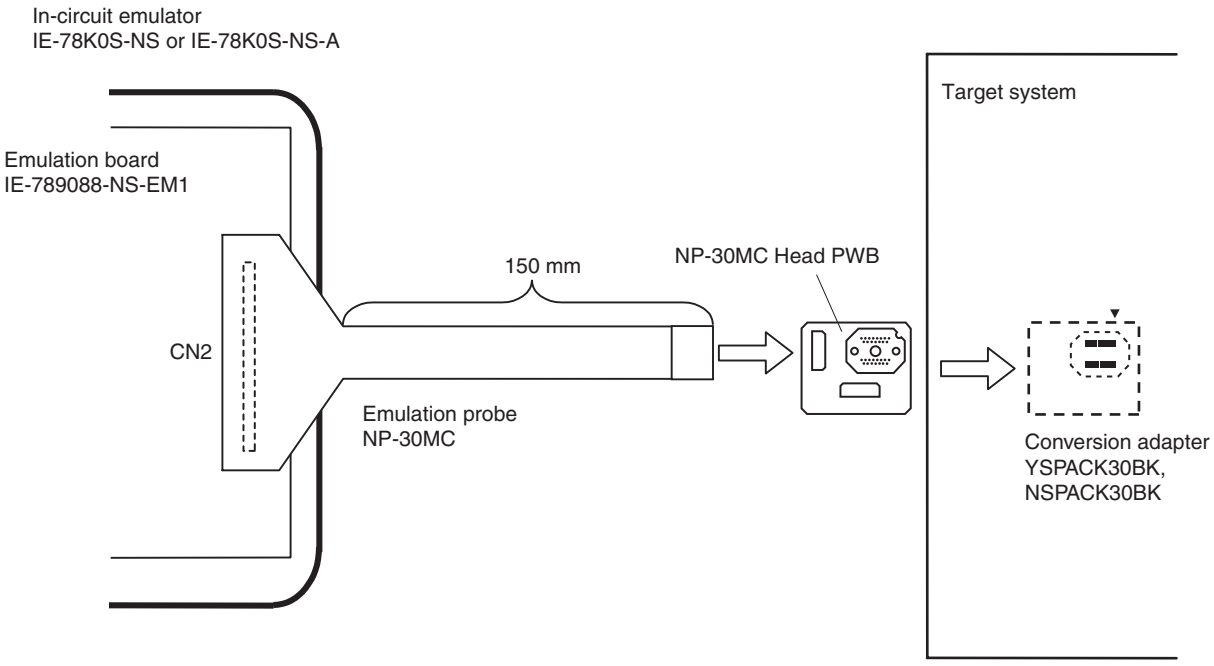
Remark NP-36GS, NP-H36GS, and NGS-30 are products of Naito Densei Machida Mfg. Co., Ltd.

Figure B-2. Connection Condition of Target System (NP-H36GS)



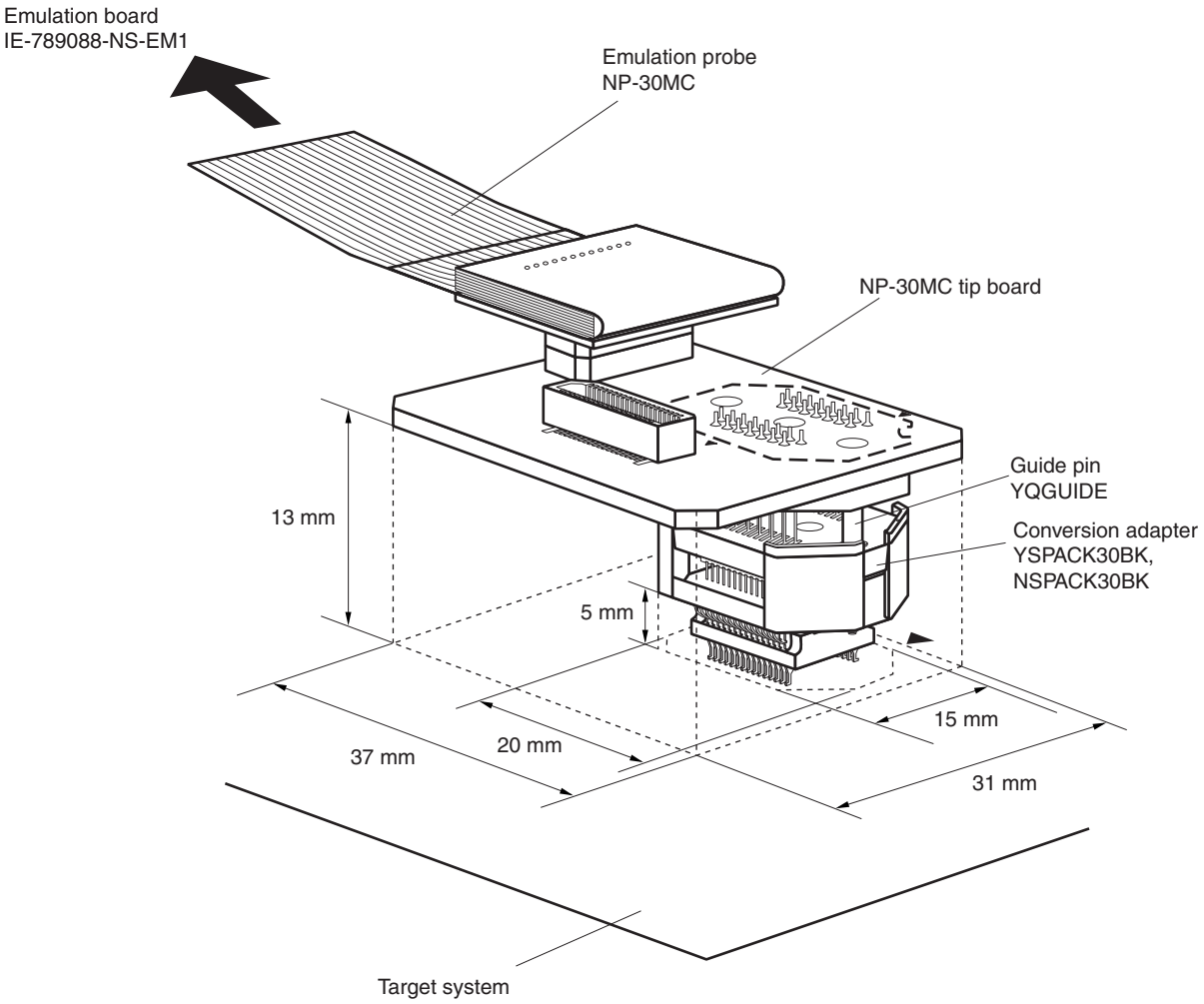
Remark NP-H36GS and NGS-30 are a products of Naito Densei Machida Mfg.Co., Ltd.

Figure B-3. Distance Between In-Circuit Emulator and Conversion Adapter (NP-30MC)



Remark NP-30MC is a product of Naito Densei Machida Mfg. Co., Ltd.
YSPACK30BK and NSPACK30BK are products of TOKYO ELETECH CORPORATION.

Figure B-4. Connection Condition of Target System (NP-30MC)



Remark NP-30MC is a product of Naito Densei Machida Mfg. Co., Ltd.
YSPACK30BK, NSPACK30BK, and YQGUIDE are products of TOKYO ELETECH CORPORATION.

APPENDIX C REGISTER INDEX

C.1 Register Name Index (Alphabetic Order)

[R]

[S]

[T]

[W]

C.2 Register Symbol Index (Alphabetic Order)

[R]

[T]

[W]

APPENDIX D REVISION HISTORY

★ D.1 Major Revisions in This Edition

Page	Description
p. 54	**CHAPTER 3 CPU ARCHITECTURE** • Modification of [Description example] in **3.4.2 Short direct addressing**
p. 89	**CHAPTER 6 16-BIT TIMER 20** • Modification of description of **6.5 Cautions on Using 16-Bit Timer 20**
pp. 93, 94, 95, 101, 116, 118, 122	**CHAPTER 7 8-BIT TIMERS 50, 60** • Modification of **Figure 7-1 Block Diagram of Timer 50** • Modification of **Figure 7-2 Block Diagram of Timer 60** • Modification of **Figure 7-3 Block Diagram of Output Controller (Timer 60)** • Modification of cautions in **7.3 (4) Carrier generator output control register 60 (TCA60)** • Modification of description of **7.4.3 Operation as carrier generator** • Addition of remark2 in **Figure 7-20 Timing of Carrier Generator Operation (When CR60 = N, CRH60 = M (M < N))** • Modification of description of **7.5 (1) Error on starting timer**
p. 128	**CHAPTER 8 8-BIT TIMER 80** • Modification of description of **8.5 (1) Error on starting timer**
pp. 136, 139, 144, 146, 148, 155, 156, 157, 161, 163, 166	**CHAPTER 10 SERIAL INTERFACE 20** • Modification of **Figure 10-1 Block Diagram of Serial Interface 20** • Addition of caution3 in **Figure 10-3 Format of Serial Operation Mode Register 20** • Addition of description in **10.3 (4) Baud rate generator control register 20 (BRGC20)** • Addition of **(d) Generation of serial clock from system clock in 3-wire serial I/O mode** and **(e) Generation of serial clock from square-wave output of timer 60 in 3-wire serial I/O mode** in **10.3 (4)** • Addition of caution2 in **10.4.2 (1) (a) Serial operation mode register 20 (CSIM20)** • Modification of description of **10.4.2 (2) (b) Parity types and operation** • Modification of description of **10.4.2 (2) (c) Transmission** • Modification of description of **10.4.2 (2) (d) Reception** • Addition of caution3 in **10.4.3 (1) (a) Serial operation mode register 20 (CSIM20)** • Addition of **(i) Generation of serial clock from system clock in 3-wire serial I/O mode** and **(ii) Generation of serial clock from square-wave output of timer 60 in 3-wire serial I/O mode** in **10.4.3 (1) (c)** • Modification of **Figure 10-11 3-Wire Serial I/O Mode Timing (iv) Slave operation (when DAP20 = 0, CKP20 = 1)**
p. 175	**CHAPTER 13 INTERRUPT FUNCTIONS** • Modification of description of **13.1 Interrupt Function Types**
pp. 223, 225	**CHAPTER 18 ELECTRICAL SPECIFICATIONS** • Addition of **Recommended Oscillator Constant** • Modification of values of **Output current, low** and **Output current, high** in **DC Characteristics**

D.2 Revisions up to Previous Edition

The revision history is described below. The “Applied to” column indicates the chapter in each edition.

Edition	Major Revision from Previous Edition	Applied to:
2nd	Update of relate documents	**INTRODUCTION**
	Update of diagram in **1.5 78K/0S Series Lineup**	**CHAPTER 1 GENERAL**
	Modification of pin handling of V_{PP} pin in **2.2.8 V_{PP} (μPD78F9088 only)** and **Table 2-1 Types of Pin I/O Circuits and Recommended Connection of Unused Pins**	**CHAPTER 2 PIN FUNCTIONS**
	Modification of indication of minimum instruction execution time in **Figure 5-2 Format of Processor Clock Control Register and 5.5 Operation of Clock Generator**	**CHAPTER 5 CLOCK GENERATOR**
	Modification of description of **6.4.1 Operation as timer interrupt**	**CHAPTER 6 16-BIT TIMER 20**
	Addition of **(f) Reading receive data** of UART in **10.4.2**	**CHAPTER 10 SERIAL INTERFACE 20**
	Revision of contents about flash memory programming as **16.1 Flash Memory Characteristics**	**CHAPTER 16 μPD78F9088**
	Addition of electrical specifications	**CHAPTER 18 ELECTRICAL SPECIFICATIONS**
	Addition of package drawing	**CHAPTER 19 PACKAGE DRAWING**
	Addition of recommended soldering conditions	**CHAPTER 20 RECOMMENDED SOLDERING CONDITIONS**
	Overall revision of contents of development tools Deletion of embedded software	**APPENDIX A DEVELOPMENT TOOLS**
	Addition of notes on target system design	**APPENDIX B NOTES ON TARGET SYSTEM DESIGN**