USER'S MANUAL

NEC

Phase-out/Discontinued

μPD78356

16-BIT SINGLE-CHIP MICROCONTROLLER

HARDWARE

μPD78355
μPD78356
μPD78P356

Document No. U10669EJ3V0UM00 (3rd edition)
(Previous No. IEU-1361)
Date Published February 1996 P
Printed in Japan

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

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

License not needed : μPD78355/(A)/(A1)/(A2), 78P356KP-S
The customer must judge the need for license
: μPD78356/(A)/(A1)/(A2), 78P356GC-7EA,
78P356GD-5BB/(A)/(A1)/(A2)

Major Revisions in This Edition

Page	Description
Throughout	"The following products are under development" -> changed to "already developed" μPD78355, 78356, 78P356, 78356 (A), 78P356 (A)
p. 83, 84	**CHAPTER 5 PORT FUNCTIONS** Addition of description of the external 8-bit bus to **5.2.3 (4) Port 9**
p. 231, 232	**CHAPTER 10 ASYNCHRONOUS SERIAL INTERFACE** Addition of **10.7 Transmitting/Receiving Data Using the Macro Service**
p. 469 to 471	**APPENDIX B TOOLS** Addition of description when using the integrated debugger

The mark ★ shows major revised points.

PREFACE

Users:

This manual is for engineers who intend to learn the capabilities of the μPD78356 subseries for application program development.
The related products are μPD78356 subseries as follows.

- Standard products : μPD78355, 78356, 78P356
- Special products : μPD78355 (A)[Note], 78355 (A1)[Note], 78355 (A2)[Note], 78356 (A), 78356 (A1)[Note], 78356 (A2)[Note], 78P356 (A), 78P356 (A1)[Note], 78P356 (A2)[Note]

Note Under development

Purpose:

The purpose of this manual is to help users understand the hardware capabilities of the μPD78356 subseries as listed below.

Organization:

The μPD78356 subseries manual consists of two volumes, hardware (this manual) and instruction.

Hardware	Instruction
Pin functions	CPU functions
Internal peripheral hardware	Addressing
Interrupt	Instruction set list
Instruction set list	Detailed explanation of instructions

CAUTIONS

Cautions on the use of the μPD78356 subseries are summarized in CHAPTER 22. Read them before using the product.
For the up-to-the-minute information about this product, please contact an NEC representative.

Guidance:

Before using this manual, the user should have a general knowledge of the electronics, logical circuit, and microcontroller fields.

- **To readers using this manual for the products other than μPD78356**
  -> When there is no functional difference in the products, the manual describes only the μPD78356, and its descriptions are applicable to the products other than μPD78356 subseries.
  The description of PROM is for both a one-time PROM and EPROM.

Examples in this manual are described assuming that they are used for the "Standard" quality grade of ordinary electronic equipment. If you want to apply these examples to use which requires "Special" quality grade, consider the quality grade of parts and circuits to be actually used.

- **To check the details of a register if you know the name of the register:**
  -> Look it up in **APPENDIX C**.

- **To understand the details of functions which you want to use:**
  -> Look it up in **APPENDIX D**.

- **To understand details of the instruction functions in the μPD78356 subseries:**
  -> See the separate **μPD78356 User's Manual Instruction (IEU-853)**[Note].

- **To understand the general functions of the μPD78356 subseries:**
  -> Read the entire manual in the order of the table of contents.

- **To know the electrical specifications in the μPD78356 subseries:**
  -> Refer to the Data Sheet in the separate volume.

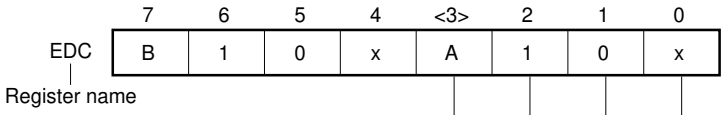
- **To know application examples of each function:**
  -> Refer to the separate Application Note.

Note This document number is that of Japanese version.

Legend:

Data weight	:	Higher digits on the left side Lower digits on the right side
Active low	:	$\overline{xxxx}$ (Pins and signal names are overscored)
Memory map address	:	Low-order address on the upper side High-order address on the lower side
Note	:	Explanation of the indicated part of the text
Caution	:	Information requesting the user's special attention
Remark	:	Supplementary information
Numeric value	:	Binary ... xxxxB or xxxx Decimal ... xxxx Hexadecimal ... xxxxH

Register format:



Bit number that is in brackets indicates, that it has been a reserved word in the NEC assembler (RA78K/III) and it has been defined with the sfrbit.h header file in the C compiler (CC78K/III).

Write operation	Read operation
Write 0 or 1. Neither value affects operation.	Reads 0 or 1.
Write 0.	
Write 1.	
Write the value corresponding to a function to use.	Reads a value according to operation status.

Never write a combination of codes marked "Setting prohibited" in the register formats in the text.

Be careful when handling characters that are easily confused:

0 (zero), O (uppercase of o)

1 (one), l (lowercase of L), I (uppercase of i)

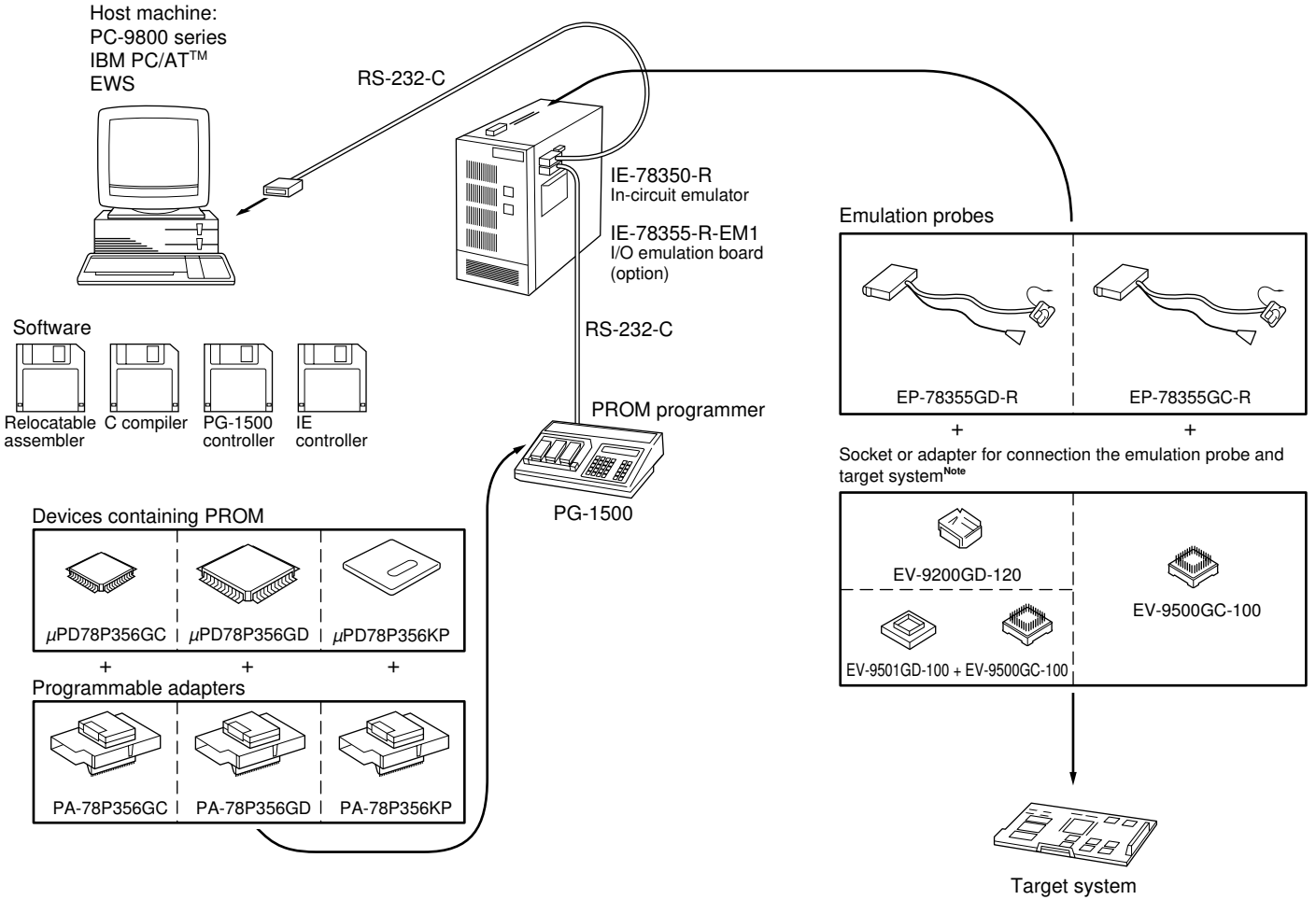
Related documents: **• Documents related to device**

Document Name		Document No.	
		English	Japanese
μPD78355, 78356 Data Sheet		To be published soon	U10155J
μPD78P356 Data Sheet		To be published soon	U10325J
μPD78355 (A), 78356 (A) Data Sheet		Planned	To be published soon
μPD78P356 (A) Data Sheet		Planned	To be published soon
μPD78356 User's Manual	Hardware	This manual	U10669J
	Instructions	IEU-1395	IEU-853
μPD78356 Special Function Register Table		IEM-1214	IEM-5576
μPD78352A Instruction Set		–	IEM-5543

• Documents related to development tools

Document Name		Document No.	
		English	Japanese
IE-78350-R User's Manual	Hardware	EEU-1366	EEU-754
	Software	EEU-1376	EEU-753
IE-784000-R User's Manual		EEU-1534	EEU-5004
IE-78355-R-EM1 User's Manual		EEU-1423	EEU-866
EP-78355GC-R User's Manual		EEU-1508	EEU-963
EP-78355GD-R User's Manual		EEU-1509	EEU-964

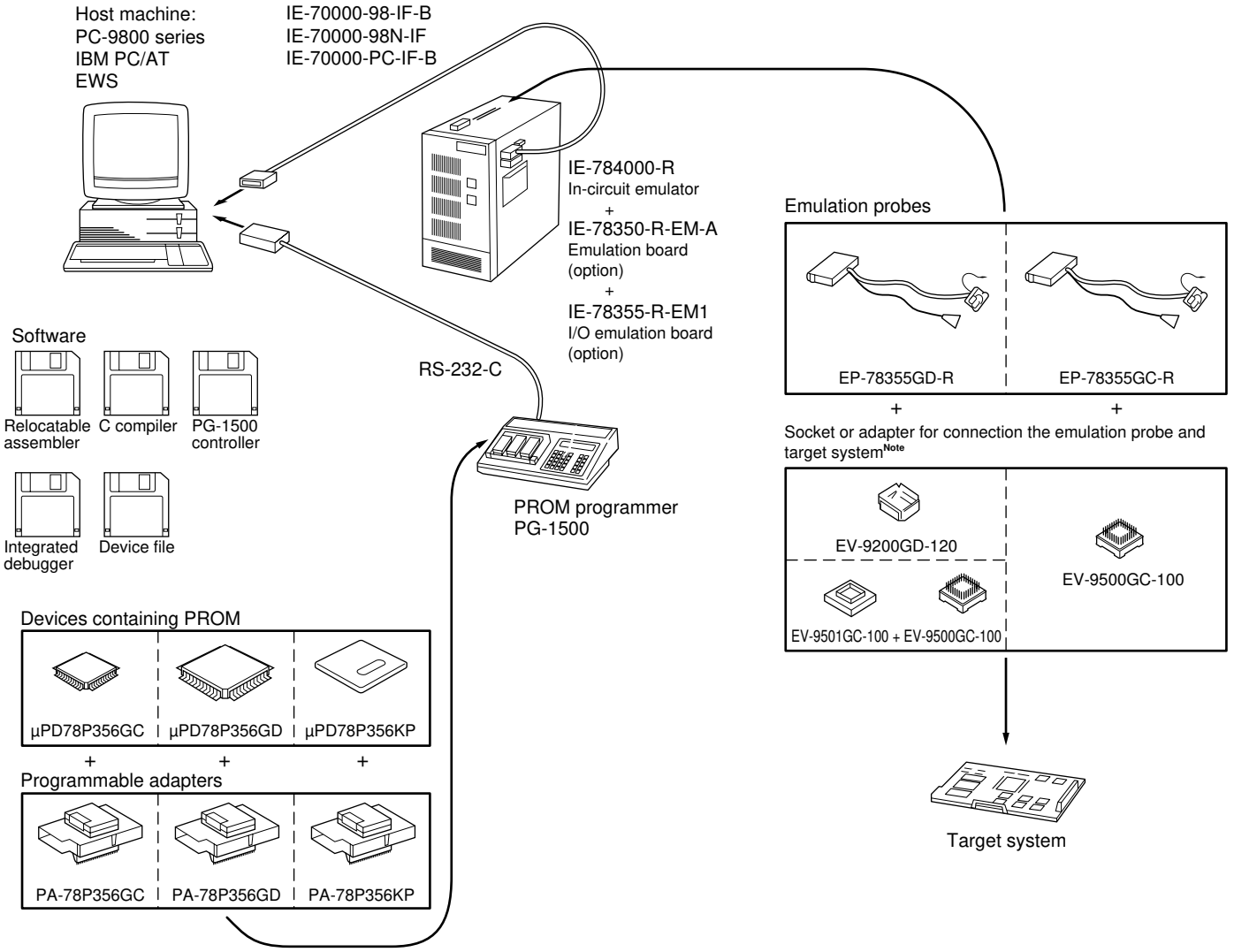
Caution The above related documents are subject to change without notice. For design purpose, etc., be sure to use the latest documents.



Note The conversion socket or the conversion adapter is supplied with the emulation probe.

Remarks 1. The PG-1500 can be directly connected to the host machine via the RS-232-C interface.

2. In this figure, the distribution media for software is represented by 3.5-inch floppy disks.

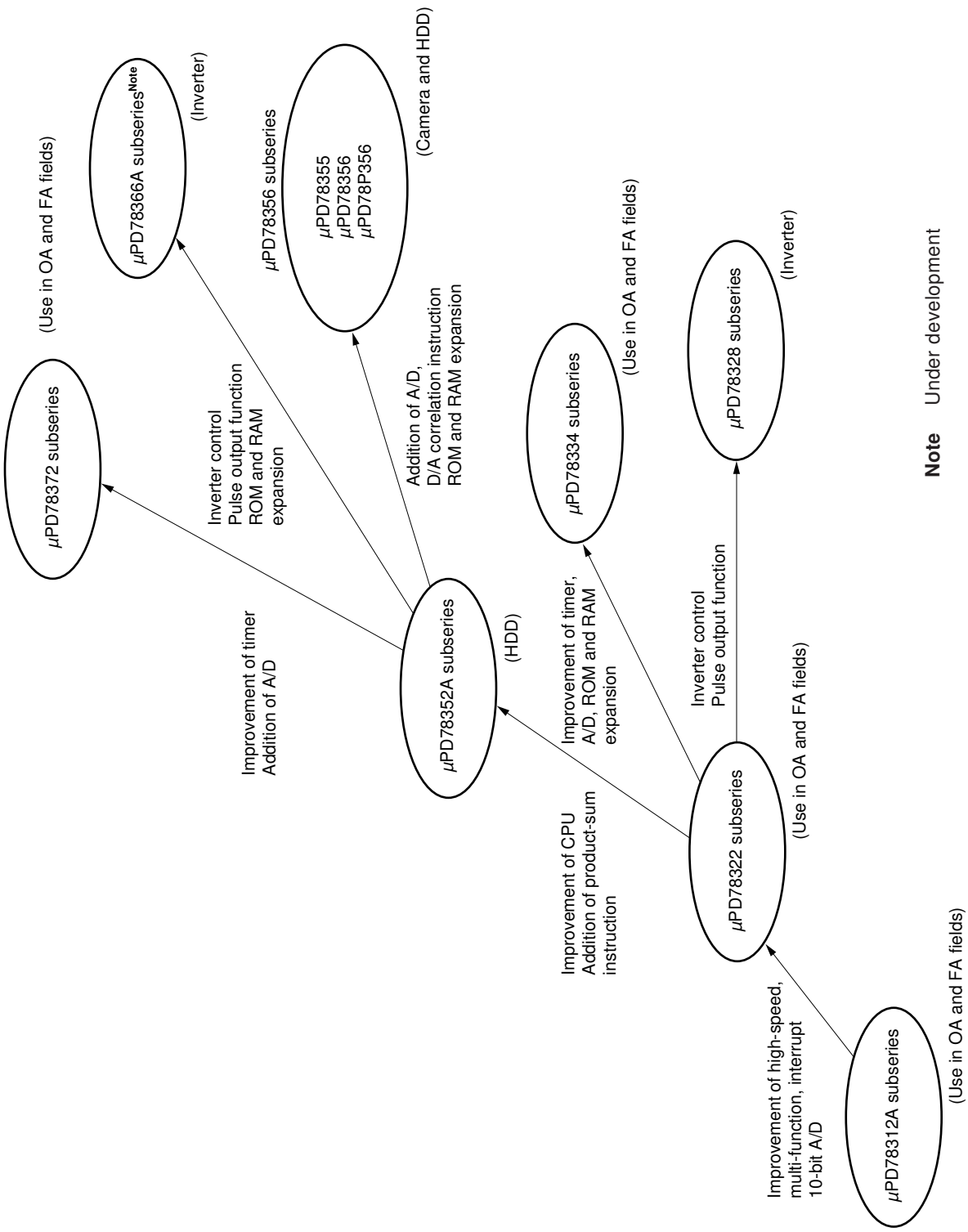


Note The conversion socket or the conversion adapter is supplied with the emulation probe.

Remarks 1. In this figure, the distribution media for software is represented by 3.5-inch floppy disks.

2. In this figure, the host machines are represented by desk-top personal computers.

Development of 78K/III series products



[MEMO]

CONTENTS

LIST OF FIGURES (1/7)

LIST OF FIGURES (2/7)

LIST OF FIGURES (3/7)

LIST OF FIGURES (4/7)

LIST OF FIGURES (5/7)

LIST OF FIGURES (6/7)

LIST OF FIGURES (7/7)

LIST OF TABLES (1/3)

LIST OF TABLES (2/3)

LIST OF TABLES (3/3)

CHAPTER 1 GENERAL

The μPD78356 subseries products are products of the 16-bit single-chip microcontroller 78K/III series. They contain a 16-bit high-speed, high-performance CPU.

The μPD78356 subseries has the following 12 models.

- Standard products : μPD78355, 78356, 78P356
- Special products : μPD78355 (A)[Note], 78355 (A1)[Note], 78355 (A2)[Note], 78356 (A), 78356 (A1)[Note], 78356 (A2)[Note], 78P356 (A), 78P356 (A1)[Note], 78P356 (A2)[Note]

Note Under development

The μPD78356 series contains a data bus whose width can be switched from 8 bits to 16 bits, and vice versa (bus sizing function).

The μPD78356 series also contains an ultrahigh-speed A/D converter and high-speed D/A converter, which are required in motor control, a real-time pulse unit, which controls pulses in real time, and large memory. Because the product-sum instruction is added, the μPD78356 series can be used in many field as high-speed, high-performance CPU.

The μPD78355 is a ROM-less version of the μPD78356.

The μPD78P356 is produced by replacing the internal mask ROM of the μPD78356 with a one-time PROM or EPROM. One-time PROM versions, in which data can be written once are ideally suited for small-scale production of many different products, and rapid development and time-to-market of a new product. EPROM products, to which programs can be re-written after previously written programs have been erased, are suited for system evaluation.

The μPD78355 (A), 78355 (A1), 78355 (A2), 78356 (A), 78356 (A1), 78356 (A2), 78P356 (A), 78P356 (A1), 78P356 (A2) are “special” quality grade versions of the μPD78355, 78356 and 78P356, respectively.

1.1 Features

- 16-bit internal architecture, 16- or 8-bit external data bus
- High-speed data processing using the pipeline control system and high-speed operation clock
  Minimum instruction execution time: 125 ns
  (internal clock frequency: 16 MHz, external clock frequency: 32 MHz)
- 115 instruction sets suited for control applications (μPD78322 upward compatible)
  – 16-bit arithmetic/logical instruction
  – Multiply/divide instruction (16 bits x 16 bits, 32 bits ÷ 16 bits)
  – Signed multiply instruction
  – Product-sum instruction (16 bits x 16 bits + 32 bits)
  – Correlation instruction
  – String instruction
  – Bit manipulation instruction and so on
- Various peripheral hardware
  – Real-time pulse unit
  – Ultrahigh-speed 10-bit resolution A/D converter: 8 channels
  (A/D conversion time is about 2 μs: Operation with an external frequency of 32 MHz)
  – 8-bit resolution D/A converter: 2 channels
  – 8/10/12-bit resolution variable PWM signal output function: 2 channels
- Discrete serial interfaces: 3 channels
  – UART
  – Synchronous serial interface/SBI
  – Synchronous serial interface (with pin switching function)
- On-chip high-speed interrupt controller
  – A 4-level priority can be specified.
  – One interrupt processing mode can be selected out of three types:
  (vectored interrupt function, macro service function, and context switching function.)
- Internal memory: ROM 48 Kbytes (μPD78356)
  Not provided (μPD78355)
  PROM 48 Kbytes (μPD78P356)
  RAM 2 Kbytes

1.2 Applications

[Standard products]

- High-speed servo control for hard disk drive or floppy disk drive
- Automatic focusing for camcoder or single-lens reflex camera
- Motor control in factory automation field

[Special products]

- Car electrical control

1.3 Ordering Information and Quality Grade

1.3.1 Standard products

(1) Ordering information

Part number	Package	Internal ROM	
μPD78355GC-7EA	100-pin plastic QFP (14 x 14 mm)	Not provided	★
μPD78355GD-5BB[Note]	120-pin plastic QFP (28 x 28 mm)	Not provided	
μPD78356GC-xxx-7EA	100-pin plastic QFP (14 x 14 mm)	Mask ROM	★
μPD78356GD-xxx-5BB	120-pin plastic QFP (28 x 28 mm)	Mask ROM	★
μPD78P356GC-7EA	100-pin plastic QFP (14 x 14 mm)	One-time PROM	★
μPD78P356GD-5BB	120-pin plastic QFP (28 x 28 mm)	One-time PROM	★
μPD78P356KP-S[Note]	120-pin ceramic WQFN	EPROM	

Note Under development

Remark xxx indicates ROM code suffix.

(2) Quality grade

Standard

Please refer to “Quality grade on NEC Semiconductor Devices” (Document number IEI-1209) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.

1.3.2 Special products

(1) Ordering information

Part number	Package	Internal ROM
μPD78355GD(A)-5BB[Note]	120-pin plastic QFP (28 x 28 mm)	Not provided
μPD78355GD(A1)-5BB[Note]	120-pin plastic QFP (28 x 28 mm)	Not provided
μPD78355GD(A2)-5BB[Note]	120-pin plastic QFP (28 x 28 mm)	Not provided
★ μPD78356GD(A)-xxx-5BB	120-pin plastic QFP (28 x 28 mm)	Mask ROM
μPD78356GD(A1)-xxx-5BB[Note]	120-pin plastic QFP (28 x 28 mm)	Mask ROM
μPD78356GD(A2)-xxx-5BB[Note]	120-pin plastic QFP (28 x 28 mm)	Mask ROM
★ μPD78P356GD(A)-5BB	120-pin plastic QFP (28 x 28 mm)	One-time PROM
μPD78P356GD(A1)-5BB[Note]	120-pin plastic QFP (28 x 28 mm)	One-time PROM
μPD78P356GD(A2)-5BB[Note]	120-pin plastic QFP (28 x 28 mm)	One-time PROM

Note Under development

Remark: xxx indicates ROM code suffix.

(2) Quality grade

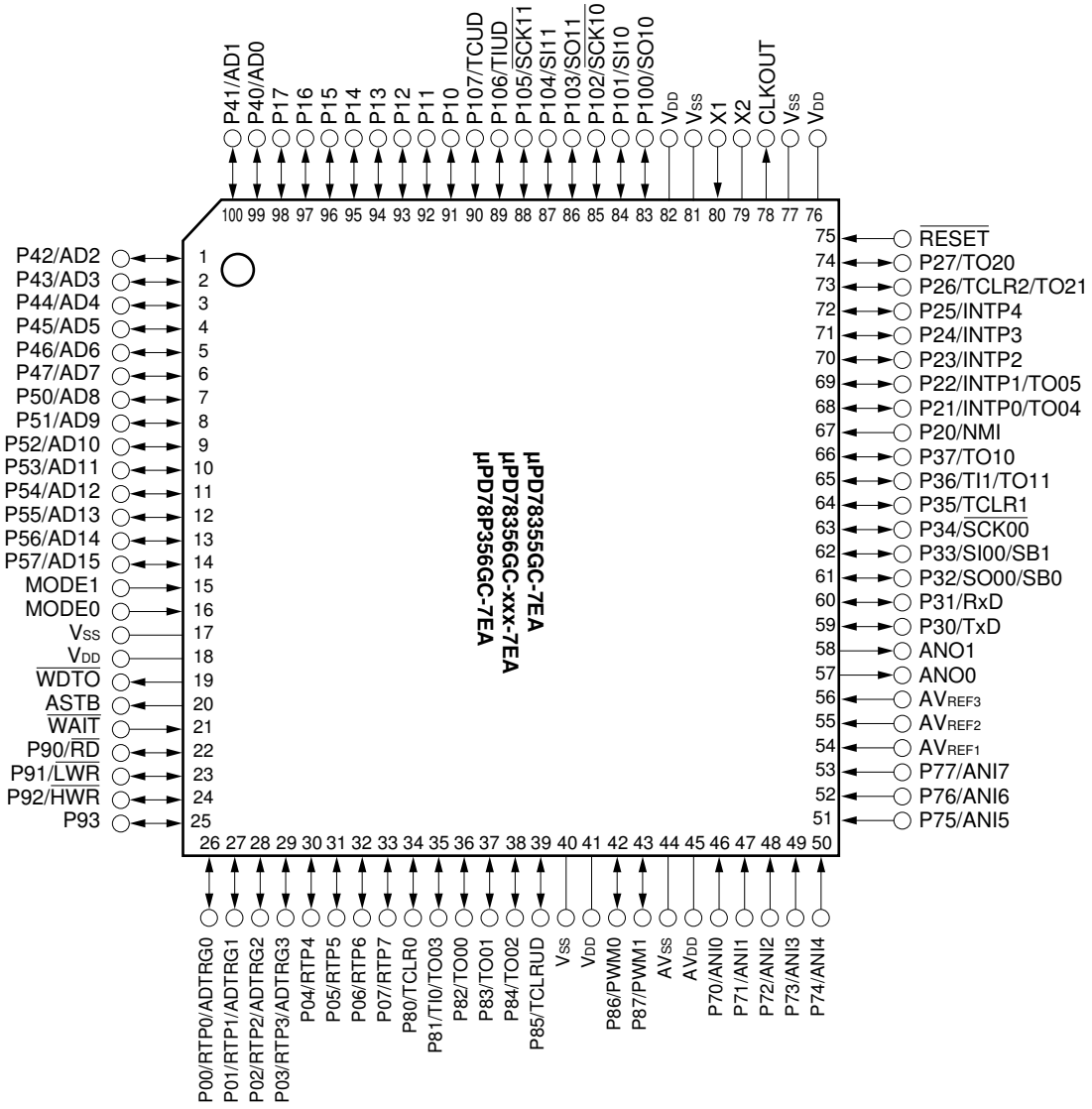
Special

Please refer to “Quality grade on NEC Semiconductor Devices” (Document number IEI-1209) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.

1.4 Pin Configuration (Top View)

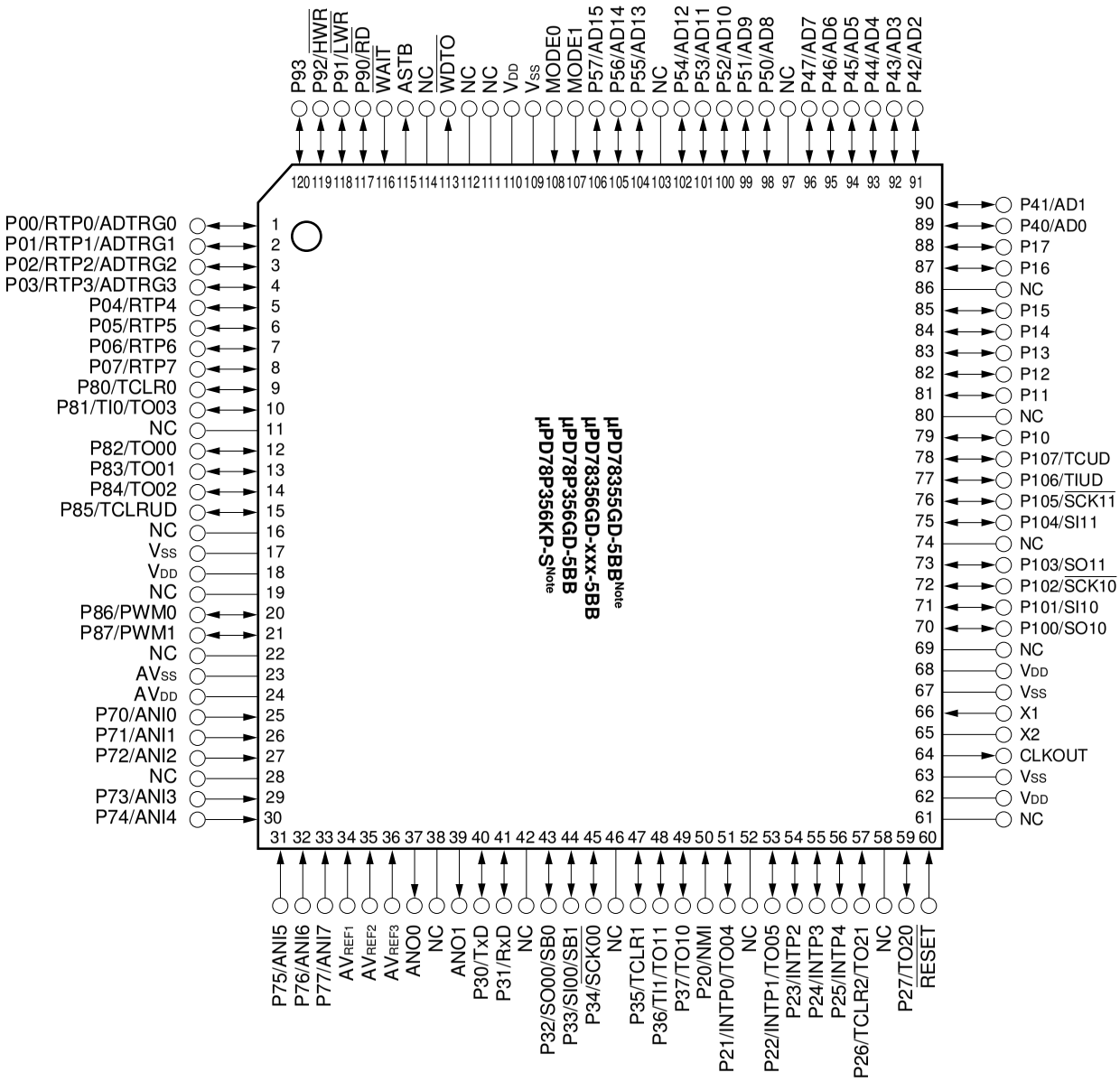
1.4.1 μPD78355 normal operation mode

(1) 100-pin plastic QFP (14 x 14 mm)



Phase-out/Discontinued

(2) 120-pin plastic QFP (28 x 28 mm), 120-pin ceramic WQFN



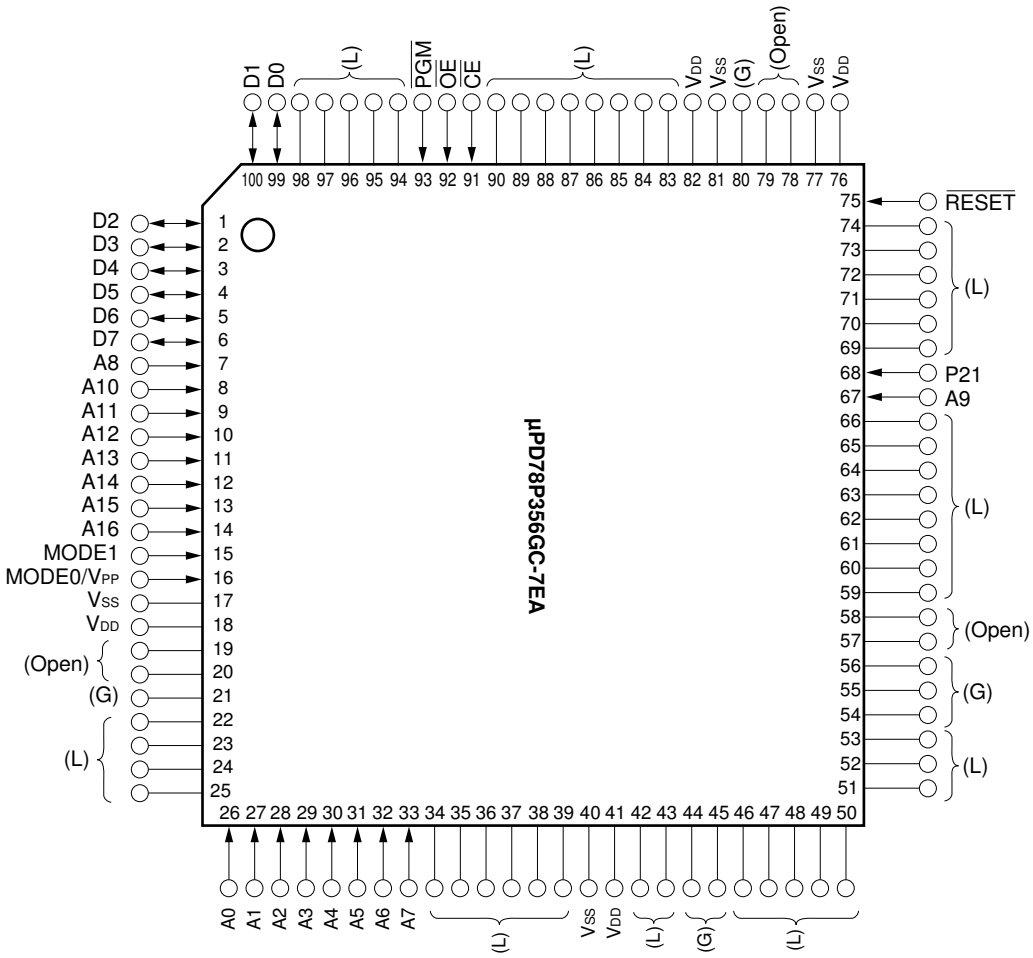
Note Under development

P00 to P07 : Port 0
P10 to P17 : Port 1
P20 to P27 : Port 2
P30 to P37 : Port 3
P40 to P47 : Port 4
P50 to P57 : Port 5
P70 to P77 : Port 7
P80 to P87 : Port 8
P90 to P93 : Port 9
P100 to P107 : Port 10
RTP0 to RTP7 : Realtime port
NMI : Nonmaskable interrupt
INTP0 to INTP4 : Interrupt from peripherals
TO00 to TO05 :
TO10, TO11 : } Timer output
TO20, TO21 :
TCLR0 to TCLR2 :
TCLRUD : } Timer clear input
TI0, TI1 : Timer input
TIUD : Count pulse input
TCUD : Control pulse input
ANI0 to ANI7 : Analog input
ADTRG0 to ADTRG3 : A/D trigger input
ANO0, ANO1 : Analog output
TxD : Transmit data
RxD : Receive data

SI00, SI10 :
SI11 : } Serial input
SO00, SO10 :
SO11 : } Serial output
SB0, SB1 : Serial bus
$\overline{SCK00}$, $\overline{SCK10}$:
$\overline{SCK11}$: } Serial clock
PWM0, PWM1 : Pulse width modulation output
$\overline{WDTO}$: Watchdog timer output
MODE0, MODE1 : Mode
AD0 to AD15 : Address/data bus
ASTB : Address strobe
$\overline{RD}$: Read strobe
$\overline{LWR}$: Low-address write strobe
$\overline{HWR}$: High-address write strobe
$\overline{WAIT}$: Wait
CLKOUT : Clock output
$\overline{RESET}$: Reset
X1, X2 : Crystal
AV_{DD} : Analog V_{DD}
AV_{SS} : Analog V_{SS}
AV_{REF1} to AV_{REF3} : Analog reference voltage
V_{DD} : Power supply
V_{SS} : Ground
NC : Non-connection

1.4.2 PROM programming mode (only μPD78P356: MODE0/V_{PP} = +5 V, MODE1 = G, P21 = G, $\overline{\text{RESET}}$ = G)

(1) 100-pin plastic QFP (14 x 14 mm)



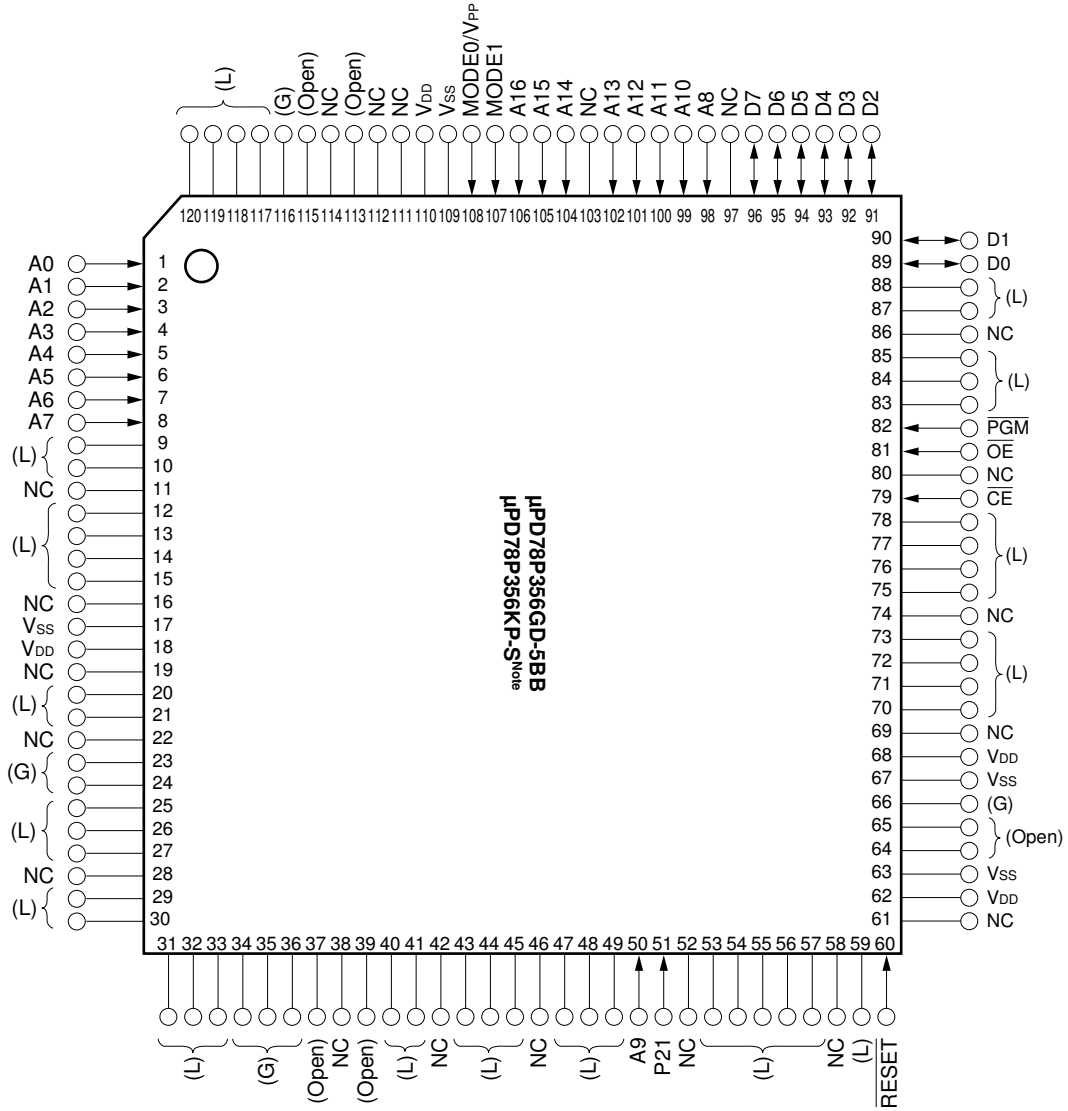
Caution Symbols in parentheses denote processing for pins not used in the PROM programming mode.

L : Connect these pins separately to the V_{SS} pins through resistors.

G : To be connected to the V_{SS} pin.

Open : Nothing should be connected on these pins.

(2) 120-pin plastic QFP (28 x 28 mm), 120-pin ceramic WQFN



Note Under development

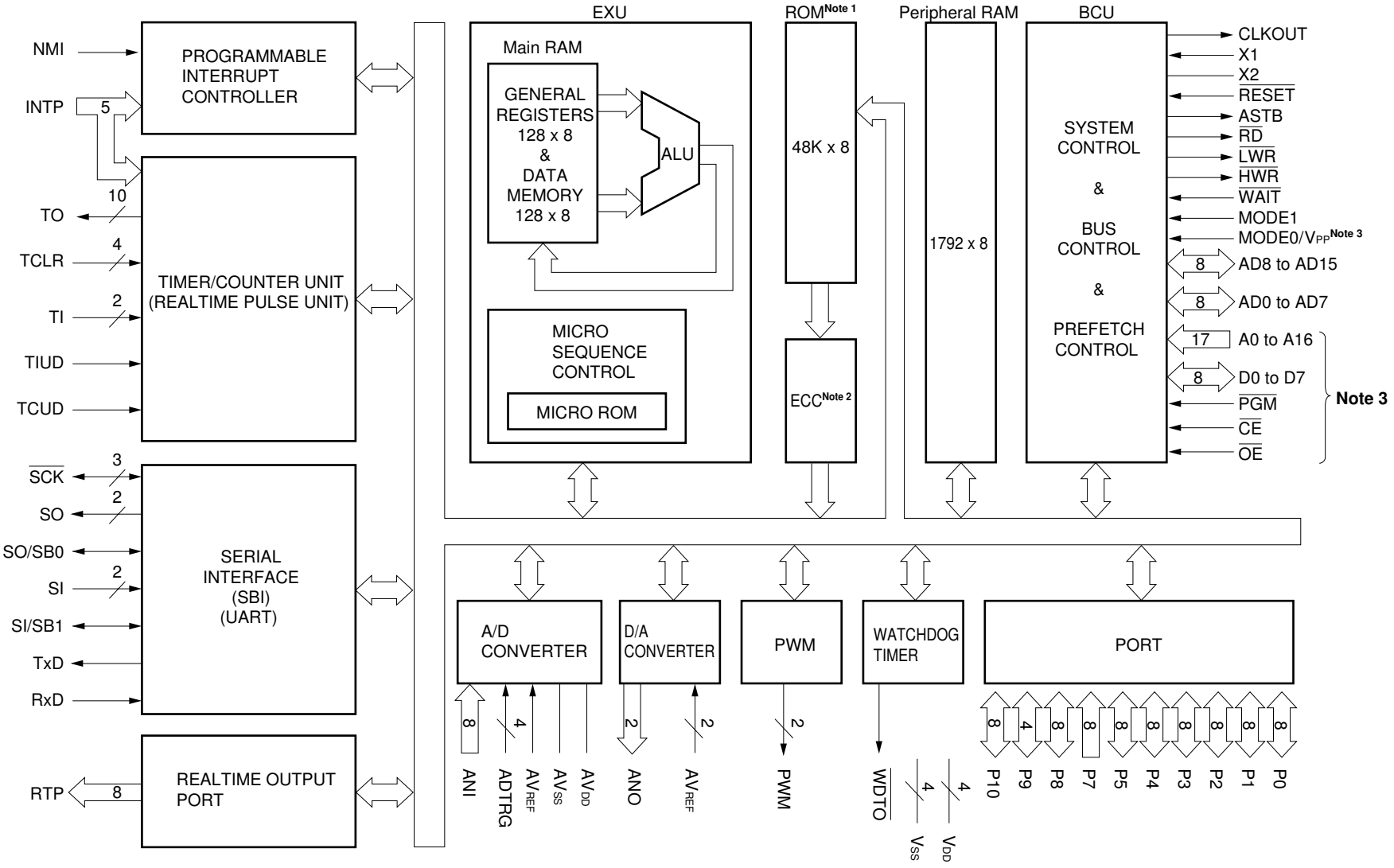
Caution Symbols in parentheses denote processing for pins not used in the PROM programming mode.

L : Connect these pins separately to the VSS pins through resistors.

G : To be connected to the VSS pin.

Open : Nothing should be connected on these pins.

A0 to A16	: Address bus		MODE0, MODE1	:	Programming mode set
D0 to D7	: Data bus		P21	:	
$\overline{CE}$	: Chip enable		$\overline{RESET}$	:	
$\overline{OE}$	: Output enable		VDD	:	Power supply
$\overline{PGM}$	: Programming mode		VSS	:	Ground
			VPP	:	Programming power supply



Notes **1.** The μPD78355 does not contain ROM.
The μPD78P356 contains 48-Kbyte PROM.

2. Only the μPD78P356 contains ECC circuit.

3. Signals for the PROM programming mode in the μPD78P356.

1.6 Function Overview

Item \ Program		μPD78355	μPD78356	μPD78P356
Minimum instruction execution time		125 ns (internal clock: 16 MHz operation, external clock: 32 MHz operation)		
Internal memory	ROM	–	48 Kbytes	–
	PROM	–	–	48 Kbytes
	RAM	2 Kbytes		
Memory space		64 Kbytes (can externally be extended)		
General register		8 bits x 16 x 8 banks		
Number of basic instructions		115		
Instruction set		• 16-bit transfer or arithmetic/logical operation • Multiply/divide operation (16 bits x 16 bits, 32 bits ÷ 16 bits) • Bit manipulate • String • Product-sum operation (16 bits x 16 bits + 32 bits) • Correlation operation		
I/O line	Input	9 (Shared for analog input: 8)		
	I/O	48	67	
Real-time pulse unit		• 16-bit timer x 5 • 10-bit timer x 1 • 16-bit compare register x 10 • 10-bit compare register x 1 • 16-bit capture/compare register x 5 • Timer output x 10		
Real-time output port		Pulse outputs synchronized with real-time pulse unit: 8		
PWM unit		8/10/12-bit resolution variable PWM outputs: 2 channels		
A/D converter		10-bit resolution: 8 channels		
D/A converter		8-bit resolution: 2 channels		
Serial interface		Serial interface with a dedicated baud rate generator UART: 1 Clock synchronous serial interface/SBI: 1 Clock synchronous serial interface (with pin switching function): 1		
Interrupt function		• External sources: 5, Internal sources: 25 (including 5 external sources) • A 4-level priority can be specified by software. • One interrupt processing mode can be selected out of three types: (vectored interrupt function, macro service function, and context switching function.)		
Bus sizing function		8 bits or 16 bits external data bus width can be selected.		
ECC circuit		Not provided		Provided
Package	Without a window	• 100-pin plastic QFP (14 x 14 mm) (only standard products) • 120-pin plastic QFP (28 x 28 mm)		
	With a window	–		120-pin ceramic WQFN (only standard products)
Others		• Watchdog timer function • Standby function (HALT/STOP)		

1.7 Differences between the μPD78355, μPD78356, and μPD78P356

The difference between the μPD78356 and μPD78355 is that the μPD78356 has an on-chip mask ROM while the μPD78355 does not.

The μPD78356 operates in the same way as the μPD78355 when the level of the MODE0 pin is held high at all times (ROM-less mode).

The μPD78P356 is a μPD78356 with its internal mask ROM replaced with a one-time PROM or EPROM. The functions of the μPD78P356 other than the PROM specifications such as the write/verify function are the same as for the μPD78356.

Table 1-1 lists the differences between the μPD78355, μPD78356, and μPD78P356.

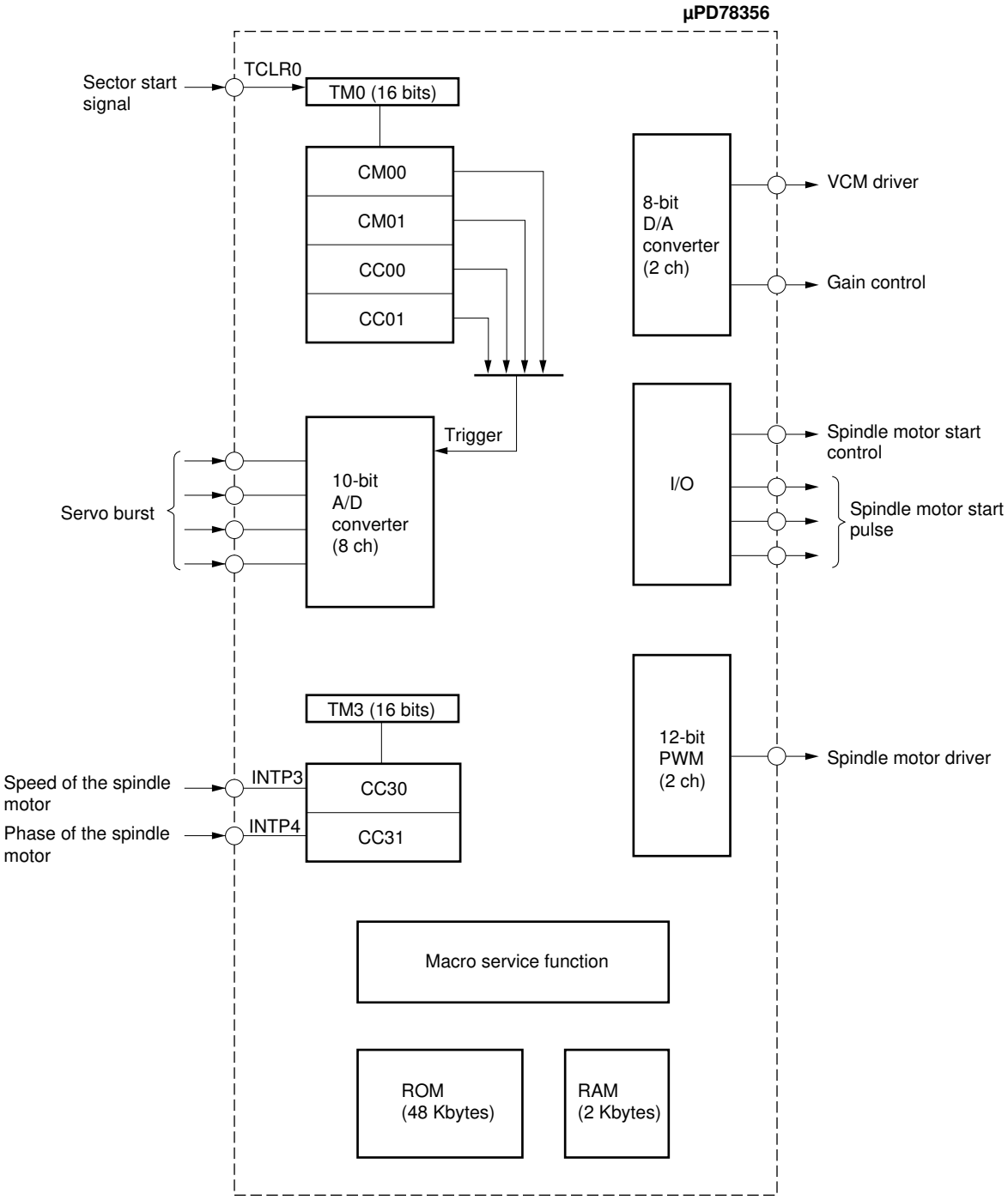
Table 1-1. Differences between the μPD78355, μPD78356, and μPD78P356

Item \ Part No.		μPD78355	μPD78356	μPD78P356
Internal memory	ROM	Not provided.	48 Kbytes	48 Kbytes (PROM)
	RAM	2 Kbytes		
I/O line	Input	9 (Shared analog input: 8)		
	I/O	48	67	
Port 4 (P40 to P47)		Always functions as the lower-order bits of the multiplexed address/data bus.	Can be specified as input or output in 8-bit units. Functions as the power-order bits of the multiplexed address/data bus (AD0 to AD7) in the external memory expansion mode.	
Port 5 (P50 to P57)		Always functions as the higher-order bits of the multiplexed address/data bus.	Can be specified as input or output in 1-bit units. Functions as the higher-order bits of the multiplexed address/data bus (AD8 to AD15) in the external memory expansion mode[Note].	
Port 9 (P90 to P93)		P90 always functions as $\overline{RD}$ strobe signal output, P91 as $\overline{LWR}$ strobe output, and P92 as $\overline{HWR}$ strobe output. P93 functions as I/O port.	Can be specified as input or output in 1-bit units. P90 functions as $\overline{RD}$ strobe signal output, P91 as $\overline{LWR}$ strobe output, and P92 as $\overline{HWR}$ strobe output in the external memory expansion mode. P93 functions as I/O port.	
Memory expansion mode register (MM)		Always in the external memory expansion mode.	Specifies input or output for port 4 in 8-bit units. Specifies the memory expansion widths for ports 4 and 5 in the external memory expansion mode.	
Port 5 mode register (PM5)		Not provided.	Specifies input or output for port 5 in 1-bit units.	
MODE0 and MODE1 settings		• With 8-bit external bus MODE0 = MODE1 = HL • With 16-bit external bus MODE0 = MODE1 = HH	• In the normal operation mode MODE0 = MODE1 = LL • In the ROM-less mode (with 8-bit external bus) MODE0 = MODE1 = HL • In the ROM-less mode (with 16-bit external bus) MODE0 = MODE1 = HH	• In the normal operation mode MODE0 = MODE1 = LL • In the programming mode (with 8-bit external bus) MODE0 = MODE1 = HL • In the ROM-less mode (with 16-bit external bus) MODE0 = MODE1 = HH
ECC circuit		Not provided		Provided
Package	Without a window	• 100-pin plastic QFP (14 x 14 mm) • 120-pin plastic QFP (28 x 28 mm)		
	With a window	–		120-pin ceramic WQFN

Note Which pins are used as an address bus depends upon the size of external memory (set by the memory expansion mode register.) (See **CHAPTER 18**.)

1.8 Example of System Configuration

Hard-disk servo control system



[MEMO]

2.1 List of Pin Functions

2.1.1 Normal operating mode

(1) Port pins

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

Pin	I/O	Function	Alternate function
P00	I/O	Port 0. 8-bit I/O port. Can be specified as input or output in 1-bit units.	RTP0/ADTRG0
P01			RTP1/ADTRG1
P02			RTP2/ADTRG2
P03			RTP3/ADTRG3
P04 to P07			RTP4 to RTP7
P10 to P17	I/O	Port 1. 8-bit I/O port. Can be specified as input or output in 1-bit units.	–
P20	I	Port 2. 8-bit I/O port. Can be specified as input or output in 1-bit units. (P20/NMI is excluded.)	NMI
P21	I/O		INTP0/TO04
P22			INTP1/TO05
P23			INTP2
P24			INTP3
P25			INTP4
P26			TCLR2/TO21
P27			TO20
P30	I/O	Port 3. 8-bit I/O port. Can be specified as input or output in 1-bit units.	TxD
P31			RxD
P32			SO00/SB0
P33			SI00/SB1
P34			$\overline{\text{SCK00}}$
P35			TCLR1
P36			TI1/TO11
P37			TO10

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

Pin	I/O	Function	Alternate function
P40 to P47[Note]	I/O	Port 4. 8-bit I/O port. Can be specified as input or output in 8-bit units.	AD0 to AD7
P50 to P57[Note]	I/O	Port 5. 8-bit I/O port. Can be specified as input or output in 1-bit units.	AD8 to AD15
P70 to P77	I	Port 7. Port used only for 8-bit input.	ANI0 to ANI7
P80	I/O	Port 8. 8-bit I/O port. Can be specified as input or output in 1-bit units.	TCLR0
P81			TI0/TO03
P82			TO00
P83			TO01
P84			TO02
P85			TCLRUD
P86			PWM0
P87			PWM1
P90[Note]	I/O	Port 9. 4-bit I/O port. Can be specified as input or output in 1-bit units.	$\overline{RD}$
P91[Note]			$\overline{LWR}$
P92[Note]			$\overline{HWR}$
P93			–
P100	I/O	Port 10. 8-bit I/O port. Can be specified as input or output in 1-bit units.	SO10
P101			SI10
P102			$\overline{SCK10}$
P103			SO11
P104			SI11
P105			$\overline{SCK11}$
P106			TIUD
P107			TCUD

Note Does not function as a port for the μPD78355 (Pin P92 does not function only when the external 16-bit bus is specified).

(2) Non-port pins

Table 2-2. Non-port Pin Functions (1/3)

Pin	I/O	Function	Alternate function
RTP0	O	Outputs a pulse in real time as triggered by a trigger signal sent from the real-time pulse unit.	P00/ADTRG0
RTP1			P01/ADTRG1
RTP2			P02/ADTRG2
RTP3			P03/ADTRG3
RTP4 to RTP7			P04 to P07
NMI	I	Nonmaskable interrupt request input	P20
INTP0		External interrupt request input	P21/TO04
INTP1			P22/TO05
INTP2			P23
INTP3			P24
INTP4			P25
TI0	I	External count clock input to timer 0	P81/TO03
TI1		External count clock input to timer 1	P36/TO11
TIUD		External count clock input to the up/down counter	P106
TCUD		Input for the control signal to determine whether the up/down counter counts up or down.	P107
TCLR0		Clear signal input to the real-time pulse unit	P80
TCLR1			P35
TCLR2			P26/TO21
TCLRUD			P85
TO00	O	Timer output from the real-time pulse unit (RPU)	P82
TO01			P83
TO02			P84
TO03			P81/TI0
TO04			P21/INTP0
TO05			P22/INTP1
TO10			P37
TO11			P36/TI1
TO20			P27
TO21			P26/TCLR2
ANI0 to ANI7	I	Analog input to the A/D converter	P70 to P77
ADTRG0		External trigger signal input to the A/D converter	P00/RTP0
ADTRG1			P01/RTP1
ADTRG2			P02/RTP2
ADTRG3			P03/RTP3

Table 2-2. Non-port Pin Functions (2/3)

Pin	I/O	Function	Alternate function
ANO0	O	Analog output from the D/A converter	–
ANO1			–
TxD	O	Serial data output from the asynchronous serial interface	P30
RxD	I	Serial data input to the asynchronous serial interface	P31
$\overline{SCK00}$	I/O	Serial clock I/O for the clock synchronous serial interface	P34
$\overline{SCK10}$			P102
$\overline{SCK11}$			P105
SI00	I	Serial data input in the 3-wire mode of the clock synchronous serial interface	P33/SB1
SI10			P101
SI11			P104
SO00	O	Serial data output in the 3-wire mode of the clock synchronous serial interface	P32/SB0
SO10			P100
SO11			P103
SB0	I/O	Serial data I/O in the SBI mode of the clock synchronous serial interface	P32/SO00
SB1			P33/SI00
PWM0	O	PWM signal output	P86
PWM1			P87
$\overline{WDTO}$	O	Output for the signal which indicates the watchdog timer overflowed. (A nonmaskable interrupt is generated.)	–
AD0 to AD7	I/O	Lower-order bits of the multiplexed address/data bus used when external memory is expanded	P40 to P47
AD8 to AD15		Higher-order bits of the multiplexed address/data bus used when external memory is expanded	P50 to P57
ASTB	O	Output for the timing signal used in externally latching address information output from the AD0 to AD15 pins, in order to access the external memory	–
$\overline{RD}$		Read strobe signal output to the external memory	P90
$\overline{LWR}$		Write strobe signal output to the low-order 8 bits in the external memory	P91
$\overline{HWR}$		Write strobe signal output to the high-order 8 bits in the external memory	P92
$\overline{WAIT}$	I	Input for the control signal which causes wait in the bus cycle	–
MODE0	I	Input for the control signal which sets the operation mode. Normally, both MODE0 and MODE1 are directly connected to the V_{SS} pin.	–
MODE1			

Table 2-2. Non-port Pin Functions (3/3)

Pin	I/O	Function	Alternate function
CLKOUT	O	System clock output	–
$\overline{\text{RESET}}$	I	System reset input	–
X1	I	Crystal input pin for the system clock. When a clock signal is provided externally, input to the X1 pin and the X2 pin should be left open.	–
X2	–		
AV_{REF1}	I	A/D converter reference voltage input	–
AV_{REF2}		D/A converter reference voltage input	–
AV_{REF3}			–
AV_{DD}	–	Analog power supply for the A/D converter	–
AV_{SS}	–	Ground for the A/D converter	–
V_{DD}	–	Positive power supply	–
V_{SS}	–	Ground	–
NC	–	Not internally connected. Connect the NC pin to the V_{SS} pin (can also be left open).	–

2.1.2 PROM programming mode (only for the μPD78P356: MODE0/V_{PP} = H, MODE1 = L, P21 = L, $\overline{\text{RESET}}$ = L)

Table 2-3. Pin Functions for PROM Programming Mode

Pin	I/O	Function
MODE0/V_{PP}	I	Power supply pin for setting the μPD78P356 to a PROM programming mode and for writing a program
MODE1		Setting the μPD78P356 to a PROM programming mode
P21		
$\overline{\text{RESET}}$		
A0 to A16		Address bus
D0 to D7	I/O	Data bus
$\overline{\text{PGM}}$	I	Program input
$\overline{\text{CE}}$		PROM enable input
$\overline{\text{OE}}$		Read strobe to PROM
V_{DD}	–	Positive power supply
V_{SS}		Ground

Caution Connect the MODE0/V_{PP}, MODE1, P21, and $\overline{\text{RESET}}$ pins to V_{DD} or V_{SS} directly.

2.2 Pin Functions

2.2.1 Normal operation mode

(1) P00 to P07 (Port 0) ... 3-state input/output

Port 0 is an 8-bit I/O port. Port 0 functions not only as a general I/O port, but also as a real-time output port or an external trigger input for the A/D converter.

By setting the port 0 mode control register (PMC0), each bit of port 0 can be placed in one of the operating modes described below. (See **5.2**.)

(a) Port mode

Port 0 functions as an 8-bit general I/O port.

By setting the port 0 mode register (PM0), each bit of port 0 can be separately specified as an input port or output port.

(b) Control mode

Port 0 functions as the control pins described below.

(i) RTP0 to RTP7

These pins function as a real-time output port.

(ii) ADTRG0 to ADTRG3

These pins function as an external trigger input for the A/D converter.

Caution: When the $\overline{\text{RESET}}$ signal is input, all the pins of port 0 are set to the input mode (output high-impedance). At this time, the contents of the output latch are undefined.

(2) P10 to P17 (Port 1) ... 3-state input/output

Port 1 is an 8-bit I/O port. Port 1 functions as a general I/O port.

By setting the port 1 mode register (PM1), each bit of port 1 can be separately specified as an input port or output port.

Caution When the $\overline{\text{RESET}}$ signal is input, all the pins of port 1 are set to the input mode (output high-impedance). At this time, the contents of the output latch are undefined.

(3) P20 to P27 (Port 2) ... 3-state input/output

Port 2 is an 8-bit I/O port. Note, however, that only P20/NMI is dedicated to input. Port 2 functions not only as a general I/O port, but also as control pins.

By setting the port 2 mode control register (PMC2), each bit of port 2 can be placed in one of the operating modes described below. (See **5.2**.)

(a) Port mode

Port 2 functions as an 8-bit general I/O port.

By setting the port 2 mode register (PM2), each bit of port 2 can be separately specified as an input port or output port. However, note that only P20 is dedicated to input.

(b) Control mode

Port 2 functions as the control pins described below.

(i) NMI

This pin functions as an input pin for an edge detection external nonmaskable interrupt request.

(ii) INTP0 to INTP4

These pins function as input pins for an edge detection external interrupt request.

(iii) TO04, TO05, TO20, TO21

These pins function as output pins for timer output from the real-time pulse unit.

(iv) TCLR2

This pin functions as an input pin for inputting a clear signal to the real-time pulse unit.

Caution When the $\overline{\text{RESET}}$ signal is input, all the pins of port 2 are set to the input mode (output high-impedance). At this time, the contents of the output latch are undefined.

(4) P30 to P37 (Port 3) ... 3-state input/output

Port 3 is an 8-bit I/O port. Port 3 functions not only as a general I/O port, but also as control pins.
By setting the port 3 mode control register (PMC3), each bit of port 3 can be placed in one of the operating modes described below. (See **5.2**.)

(a) Port mode

Port 3 functions as an 8-bit general I/O port.
By setting the port 3 mode register (PM3), each bit of port 3 can be separately specified as an input port or output port.

(b) Control mode

Port 3 functions as the control pins described below.

(i) RxD, TxD

These pins function as I/O pins for serial data transferred via the asynchronous serial interface (UART).

(ii) SO00/SB0, SI00/SB1

These pins function as I/O pins for serial data transferred via the clock synchronous serial interface.

(iii) $\overline{\text{SCK00}}$

This pin functions as an I/O pin for a serial clock signal transferred via the clock synchronous serial interface.

(iv) TO10, TO11

These pins function as output pins for timer output from the real-time pulse unit.

(v) TI1

This pin functions as an input pin for inputting an external count clock signal to timer 1 (TM1) of the real-time pulse unit.

(vi) TCLR1

This pin functions as an input pin for inputting a clear signal to the real-time pulse unit.

Caution When the $\overline{\text{RESET}}$ signal is input, all the pins of port 3 are set to the input mode (output high-impedance). At this time, the contents of the output latch are undefined.

(5) P40 to P47 (Port 4) ... 3-state input/output

The function of port 4 is different for the μPD78355, μPD78356, and μPD78P356.

(a) μPD78355

Port 4 is placed in the external memory expansion mode at all times. When an external 8-bit bus is specified, port 4 always functions as an 8-bit address/data bus (AD0 to AD7). When an external 16-bit bus is specified, port 4 always functions as the lower 8-bit address/data bus (AD0 to AD7). Port 4 does not function as a port.

(b) μPD78356, μPD78P356 (with the MODE0 and MODE1 pins set low)

Port 4 is an 8-bit I/O port. Port 4 functions not only as a general I/O port, but also as an address/data bus.

By setting the memory expansion mode register (MM), port 4 can be placed in one of the operating modes described below. (See **CHAPTER 18**.)

(i) Port mode

Port 4 functions as an 8-bit general I/O port.

By setting the memory expansion mode register (MM), port 4 can be specified as an 8-bit input port or an 8-bit output port.

(ii) External memory expansion mode

Port 4 functions as an address/data bus (AD0 to AD7) for accessing external memory. When an external 8-bit bus is specified, port 4 functions as an 8-bit address/data bus. When an external 16-bit bus is specified, port 4 functions as the lower 8-bit address/data bus. In this case, the value of the port 4 register has no effect.

Caution When the $\overline{\text{RESET}}$ signal is input, all the pins of port 4 are set to the input mode (output high-impedance) regardless of whether the port mode or external memory expansion mode is set. At this time, the contents of the output latch are undefined.

(6) P50 to P57 (Port 5) ... 3-state input/output

The function of port 5 is different for the μPD78355, μPD78356, and μPD78P356.

(a) μPD78355

Port 5 is placed in the external memory expansion mode at all times. When an external 8-bit bus is specified, port 5 always functions as an 8-bit address/data bus (AD8 to AD15). When an external 16-bit bus is specified, port 5 always functions as the higher 8-bit address/data bus (AD8 to AD15). Port 5 does not function as a port.

(b) μPD78356, μPD78P356 (with the MODE0 and MODE1 pins made low)

Port 5 is an 8-bit I/O port. Port 5 functions not only as a general I/O port, but also as an address/data bus.

By setting the memory expansion mode register (MM), port 5 can be placed in one of the operating modes described below. (See **CHAPTER 18**.)

(i) Port mode

Port 5 functions as a 2-bit, 4-bit, or 8-bit general I/O port.

By setting the port 5 mode register (PM5), each bit of port 5 can be separately specified as an input port or output port.

(ii) External memory expansion mode

Port 5 functions as an address/data bus (AD8 to AD15) for accessing external memory. When an external 8-bit bus is specified, port 5 functions as an 8-bit address/data bus. When an external 16-bit bus is specified, port 5 functions as the higher 8-bit address/data bus. In this case, the value of the port 5 register has no effect.

P50 to P57 can be specified for address output stepwise according to the size of external expansion memory. (Any remaining pins can be used for a general I/O port.)

Caution When the $\overline{\text{RESET}}$ signal is input, all the pins of port 5 are set to the input mode (output high-impedance) regardless of whether the port mode or external memory expansion mode is set. At this time, the contents of the output latch are undefined.

(7) P70 to P77 (Port 7) ... Input

Port 7 is an 8-bit special input port. Port 7 functions not only as a general input port, but also as an analog signal input for the A/D converter.

(a) Port mode

Port 7 is placed in the control mode at all times. However, by executing a read instruction for port 7, the state of each pin can be read.

(b) Control mode

Port 7 functions as input pins (ANI0 to ANI7) for inputting an analog signal to the A/D converter.

(8) P80 to P87 (Port 8) ... 3-state input/output

Port 8 is an 8-bit I/O port. Port 8 functions not only as a general I/O port, but also as PWM output pins or control pins for the real-time pulse unit.

By setting the port 8 mode control register (PMC8), each bit of port 8 can be placed in one of the operating modes described below. (See **5.2**.)

(a) Port mode

Port 8 functions as an 8-bit general I/O port.

By setting the port 8 mode register (PM8), each bit of port 8 can be separately specified as an input port or output port.

(b) Control mode

Port 8 functions as the control pins described below.

(i) TO00, TO01, TO02, TO03

These pins function as output pins for timer output from the real-time pulse unit.

(ii) TI0

This pin functions as an input pin for inputting an external count clock signal to timer 0 (TM0) of the real-time pulse unit.

(iii) TCLR0, TCLRUD

These pins function as input pins for inputting a clear signal to the real-time pulse unit.

(iv) PWM0, PWM1

These pins function as PWM signal output pins.

Caution When the $\overline{\text{RESET}}$ signal is input, all the pins of port 8 are set to the input mode (output high-impedance). At this time, the contents of the output latch are undefined.

(9) P90 to P93 (Port 9) ... 3-state input/output

The function of port 9 is different for the μPD78355, μPD78356, and μPD78P356.

(a) μPD78355

The P90 and P91 pins are placed in the external memory expansion mode at all times, and always function as the $\overline{RD}$ and $\overline{LWR}$ pins. P90 and P91 do not function as a port.

The P92 pin functions as a port when an external 8-bit bus is specified. P92 functions as the $\overline{HWR}$ pin when an external 16-bit bus is specified. When P92 functions as the $\overline{HWR}$ pin, it does not have the port function.

The P93 pin functions as a general I/O port. By setting the port 9 mode register (PM9), P93 can be specified as an input port or output port.

(b) μPD78356, μPD78P356 (with the MODE0 and MODE1 pins made low)

Port 9 is a 4-bit I/O port. Port 9 functions not only as a general I/O port, but also as control signal output pins for external memory.

By setting the memory expansion mode register (MM) and programmable wait control register (PWC), port 9 can be placed in one of the operating modes described below. (See **CHAPTER 18**.)

(i) Port mode

Port 9 functions as a 4-bit general I/O port.

By setting the port 9 mode register (PM9), each bit of port 9 can be specified as an input port or output port.

(ii) External memory expansion mode

Port 9 functions as control signal output pins for external memory.

<1> $\overline{RD}$

This pin is the lead signal output pin for external memory.

<2> $\overline{LWR}$

This pin is the 8-bit write signal output pin for external memory. When an external 16-bit bus is specified, this pin functions as the lower 8-bit write signal output pin for external memory.

<3> $\overline{HWR}$

When an external 16-bit bus is specified, this pin functions as the higher 8-bit write signal output pin for external memory.

Caution When the $\overline{RESET}$ signal is input, all the pins of port 9 are set to the input mode (output high-impedance) regardless of whether the port mode or external memory expansion mode is set. At this time, the contents of the output latch are undefined.

(10) P100 to P107 (Port 10) ... 3-state input/output

Port 10 is an 8-bit I/O port. Port 10 functions not only as a general I/O port, but also as control pins. By setting the port 10 mode control register (PMC10), each bit of port 10 can be placed in one of the operating modes described below. (See **5.2**.)

(a) Port mode

Port 10 functions as an 8-bit general I/O port.

By setting the port 10 mode register (PM10), each bit of port 10 can be separately specified as an input port or output port.

(b) Control mode

Port 10 functions as the control pins described below.

(i) SO10, SI10, SO11, SI11

These pins function as I/O pins for serial data transferred via the clock synchronous serial interface (with the pin switch function).

(ii) $\overline{\text{SCK00}}$, $\overline{\text{SCK11}}$

These pins function as I/O pins for a serial clock signal transferred via the clock synchronous serial interface (with the pin switch function).

(iii) TIUD

This pin functions as an input pin for inputting a count clock signal to the up/down counter (UDC).

(iv) TCUD

This pin functions as an input pin for inputting a count operation switch control signal to the up/down counter (UDC).

Caution When the $\overline{\text{RESET}}$ signal is input, all the pins of port 10 are set to the input mode (output high-impedance). At this time, the contents of the output latch are undefined.

(11) ASTB (Address strobe) ... Output

This pin functions as an output pin for outputting a timing signal used to access external expansion memory. This port is used to latch an address output externally on the P40/AD0 to P47/AD7 pins.

(12) $\overline{\text{WAIT}}$ (Wait) ... Input

This pin functions as an input pin for inputting a control signal used to cause the bus cycle to wait. While the low-level signal is input to the $\overline{\text{WAIT}}$ pin, the bus cycle is extended. Then, a device with a low-speed access capability can be connected.

(13) MODE0, MODE1 (Mode) ... Input

These pins function as input pins for inputting a control signal used to specify an operating mode. As indicated in the table below, the setting of these pins is different for the μPD78355, μPD78356, and μPD78P356. The levels of the MODE0 and MODE1 pins cannot be changed during operation.

μPD78355	μPD78356	μPD78P356
• External 8-bit bus MODE0 = MODE1 = HL • External 16-bit bus MODE0 = MODE1 = HH	• Normal operation mode MODE0 = MODE1 = LL • ROM-less mode (External 8-bit bus) MODE0 = MODE1 = HL • ROM-less mode (External 16-bit bus) MODE0 = MODE1 = HH	• Normal operation mode MODE0 = MODE1 = LL • Programming mode (External 8-bit bus) MODE0 = MODE1 = HL • ROM-less mode (External 16-bit bus) MODE0 = MODE1 = HH

Cautions 1. Be sure to connect the MODE0 and MODE1 pins to V_{DD} or V_{SS} directly.

2. Do not set settings other than the above to the MODE0 and MODE1 pins.

(14) $\overline{\text{RESET}}$ (Reset) ... Input

This pin is the system reset input pin and is low-level active.

(15) CLKOUT (Clock output) ... Output

This pin is the system clock output pin.

(16) X1, X2 (Crystal)

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

An external clock, when used, is input to the X1 pin (the X2 pin is left open).

(17) $\overline{\text{WDTO}}$ (Watchdog timer output) ... Output

This pin is used to output a signal indicating that the watchdog timer has generated a nonmaskable interrupt.

(18) ANO0, ANO1 (Analog output) ... Output

D/A converter analog output pins

(19) AV_{REF1} (Analog reference voltage) ... Input

Reference voltage input pin used with the A/D converter

(20) AV_{REF2}, AV_{REF3} (Analog reference voltage) ... Input

Reference voltage input pins used with the D/A converter

(21) AV_{DD} (Analog V_{DD})

A/D converter power supply pin

(22) AV_{SS} (Analog V_{SS})

A/D converter GND pin

(23) V_{DD} (Power supply)

Positive power supply pin

(24) V_{SS} (Ground)

Ground potential pin

(25) NC (Non-connection)

This pin is not internally connected. Connect this pin to V_{SS}. (This pin can also be left open.)

2.2.2 PROM programming mode (only for the μPD78P356)

(1) MODE0, MODE1, P21, $\overline{\text{RESET}}$ ... Input

Input pin that sets the μPD78P356 in the PROM programming mode.
When MODE0 is set to H, MODE1 is set to L, P21 is set to L, and $\overline{\text{RESET}}$ is set to L, the μPD78P356 enters the PROM programming mode.

(2) A0 to A16 (Address bus) ... Input

Address bus that selects internal PROM address (0000H to BFFFH).

(3) D0 to D7 (Data bus) ... Input/output

Data bus. Programs are written in or read from the internal PROM via this bus.

(4) $\overline{\text{PGM}}$ (Programming mode) ... Input

Input pin for the operation mode control signal of the internal PROM.
When this signal is active, it enables writing to the internal PROM.
When this signal is inactive, it enables reading from the internal PROM.

(5) $\overline{\text{CE}}$ (Chip enable) ... Input

This pin inputs the enable signal from the internal PROM.
When $\overline{\text{CE}}$ = H, $\overline{\text{OE}}$ = H, and $\overline{\text{PGM}}$ = L, a single-page (four-byte) program can be written in 1-byte units.
When $\overline{\text{CE}}$ = L, $\overline{\text{OE}}$ = H, and $\overline{\text{PGM}}$ = L, a single-byte program can be written in 1-byte units.
Setting $\overline{\text{OE}}$ to L when $\overline{\text{CE}}$ = L enables the contents of PROM to be read.

(6) $\overline{\text{OE}}$ (Output enable) ... Input

This pin inputs the read strobe signal into the internal PROM.
When $\overline{\text{CE}}$ = L, activate this signal. Then the contents of the PROM selected by A0 to A16 is read on to D0 to D7 in 1-byte units.

(7) V_{PP} (Programming power supply)

Power supply pin for program writing.
When V_{PP} =12.5 V, $\overline{\text{OE}}$ = H, and $\overline{\text{CE}}$ = L, programs on D0 to D7 are written into the internal PROM selected by A0 to A16.

(8) V_{DD} (Power supply)

Positive power supply pin

(9) V_{SS} (Ground)

Ground potential pin

2.3 Input/Output Circuits and Connection of Unused Pins

Table 2-4 shows the input or output circuit type of each pin and connections of unused pins. Fig. 2-1 shows input and output circuits.

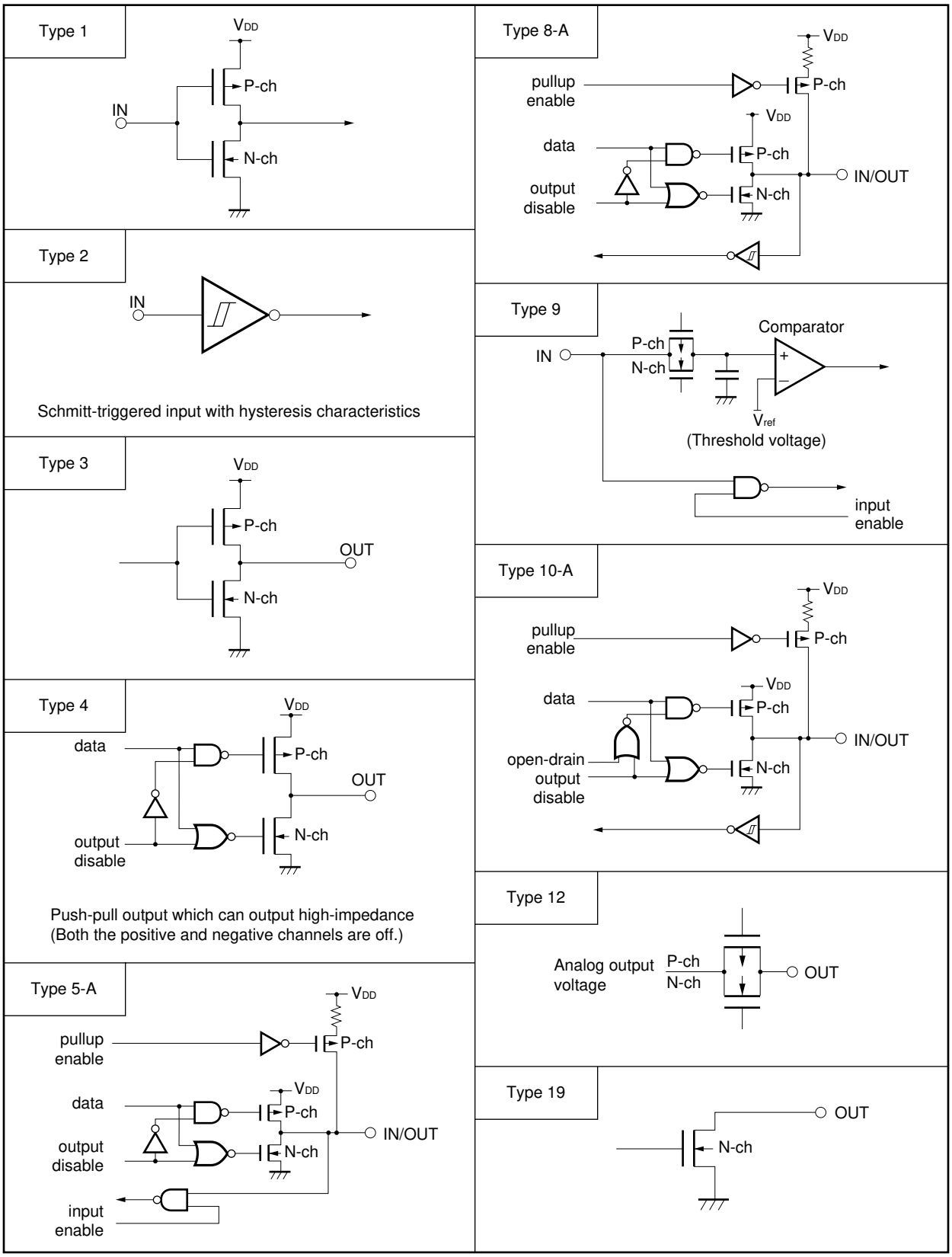
Table 2-4. Input/Output Circuits of Each Pin and Connection of Unused Pins (1/2)

Pin	I/O circuit type	Recommended connection method
P00/RTP0/ADTRG0 to P03/RTP3/ADTRG3	8-A	Input state : Each pin is connected to the V_{DD} or V_{SS} pin via a separate resistor. Output state : Open
P04/RTP4 to P07/RTP7	5-A	
P10 to P17		
P20/NMI	2	Connected to the V_{SS} pin.
P21/INTP0/TO04	8-A	Input state : Each pin is connected to the V_{DD} or V_{SS} pin via a separate resistor. Output state : Open
P22/INTP1/TO05		
P23/INTP2		
P24/INTP3		
P25/INTP4		
P26/TCLR2/TO21		
P27/TO20		
P30/TxD	5-A	
P31/RxD		
P32/SO00/SB0	10-A	
P33/SI00/SB1		
P34/$\overline{\text{SCK00}}$	8-A	
P35/TCLR		
P36/TI1/TO11		
P37/TO10	5-A	
P40/AD0 to P47/AD7		
P50/AD8 to P57/AD15		
P70/ANI0 to P77/ANI7	9	Connected to the V_{SS} pin.
P80/TCLR0	8-A	Input state : Each pin is connected to the V_{DD} or V_{SS} pin via a separate resistor. Output state : Open
P81/TI0/TO03		
P82/TO00	5-A	
P83/TO01		
P84/TO02		
P85/TCLRUD	8-A	
P86/PWM0	5-A	
P87/PWM1		

Table 2-4. Input/Output Circuits of Each Pin and Connection of Unused Pins (2/2)

Pin	I/O circuit type	Recommended connection method
P90/$\overline{RD}$	5-A	Input state : Each pin is connected to the V_{DD} or V_{SS} pin via a separate resistor. Output state : Open
P91/$\overline{LWR}$		
P92/$\overline{HWR}$		
P93		
P100/SO10		
P101/SI10	8-A	
P102/$\overline{SCK10}$		
P103/SO11	5-A	
P104/SI11	8-A	
P105/$\overline{SCK11}$		
P106/TIUD		
P107/TCUD		
ANO0, ANO1	12	Open
ASTB	4	
CLKOUT	3	
$\overline{WDTO}$	19	Connected to the V_{SS} pin.
$\overline{WAIT}$	1	Connected to the V_{DD} pin.
MODE0, MODE1	1	–
$\overline{RESET}$	2	
AV_{REF1} to AV_{REF3}, AV_{SS}	–	Connected to the V_{SS} pin.
AV_{DD}		Connected to the V_{DD} pin.
NC		Connected to the V_{SS} pin (or open).

Fig. 2-1. Input/Output Circuits of Each Pin



CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Space

The μPD78356 allows access to a memory space of up to 64 Kbytes (see **Fig. 3-1**). The MODE0 and MODE1 pins of the μPD78356 allow the user to choose between internal ROM access and external memory access (ROM-less mode).

The method of program memory mapping varies from one product to another. However, the same data memory mapping is input to all products.

(1) μPD78355 (MODE0 and MODE1 = HL or HH)

Program memory is mapped to external memory (63232 bytes: 0000H to F6FFH). This area can be shared with data memory.

Data memory is mapped to internal RAM (2048 bytes: F700H to FEFFH).

When an external 8-bit bus is used, set MODE0 and MODE1 to HL. When an external 16-bit bus is used, set MODE0 and MODE1 to HH.

(2) μPD78356 (MODE0 and MODE1 = LL)

Program memory is mapped to internal ROM (49152 bytes: 0000H to BFFFH) and external memory (14080 bytes: C000H to F6FFH). External memory is accessed in the external memory expansion mode. The external memory mapping area can be shared with data memory.

Data memory is mapped to internal RAM (2048 bytes: F700H to FEFFH).

Usually, set MODE0 and MODE1 to LL. To set the ROM-less mode, set MODE0 and MODE1 to HL when an 8-bit external bus is used, and set MODE0 and MODE1 to HH when a 16-bit external bus is used.

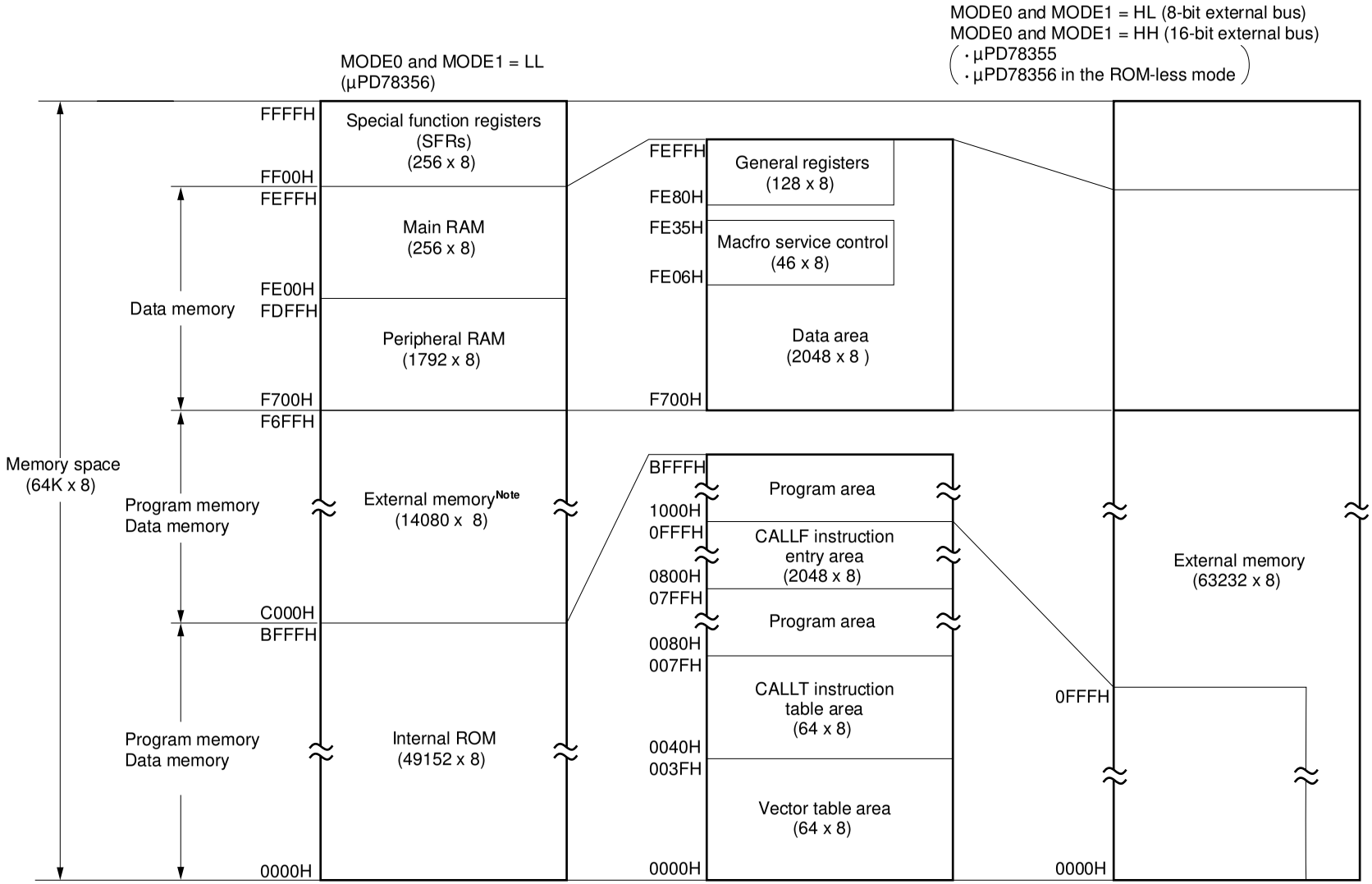
(3) μPD78P356 (MODE0 and MODE1 = LL)

Program memory is mapped to internal PROM (49152 bytes: 0000H to BFFFH) and external memory (14080 bytes: C000H to F6FFH). External memory is accessed in the external memory expansion mode. The external memory mapping area can be shared with data memory.

Data memory is mapped to internal RAM (2048 bytes: F700H to FEFFH).

Usually, set MODE0 and MODE1 to LL. To set the programming mode, set MODE0 and MODE1 to HL. To set the ROM-less mode, set MODE0 and MODE1 to HH when a 16-bit external bus is used.

Fig. 3-1. Memory Map



Note Access in external memory expansion mode

Caution When executing word access (includes stack manipulation) to the main RAM area (FE00H to FEFFH), only even address can be specified using operand.

3.1.1 Vector table area

Interrupt requests from peripheral hardware, reset inputs, external interrupt requests, and branch addresses interrupted by a break instruction are stored in the 64-byte area from 0000H to 003FH.

When an interrupt request is issued, the contents of each vector table are set to the program counter (PC) before branching. The contents of the even addresses are set to eight low-order bits of the PC and the contents of the odd addresses are set to the eight high-order bits.

When the TPF bit of the CPU control word (CCW) is set, the internal ROM area 8002H to 803FH, instead of the area 0002H to 003FH, is used as an interrupt vector table.

Table 3-1. Vector Table Area (1/2)

Interrupt source		Vector table address	
Interrupt request	Interrupt source/unit	TPF = 0	TPF = 1
$\overline{\text{RESET}}$	$\overline{\text{RESET}}$ pin input	0000H	
NMI	NMI pin input	0002H	8002H
INTWDT	Watchdog timer	0004H	8004H
INTOV0	Real-time pulse unit	0006H	8006H
INTOV3		0008H	8008H
INTP0/INTCC00	INTP0 Pin input/real-time pulse unit	000AH	800AH
INTP1/INTCC01	INTP1 Pin input/real-time pulse unit	000CH	800CH
INTP2/INTCC02	INTP2 Pin input/real-time pulse unit	000EH	800EH
INTP3/INTCC30	INTP3 Pin input/real-time pulse unit	0010H	8010H
INTP4/INTCC31	INTP4 Pin input/real-time pulse unit	0012H	8012H
INTCM00	Real-time pulse unit	0014H	8014H
INTCM01		0016H	8016H
INTCM02		0018H	8018H
INTCM03		001AH	801AH
INTCM10		001CH	801CH
INTCM11		001EH	801EH
INTCM20		0020H	8020H
INTCM21		0022H	8022H
INTCM40		0024H	8024H
INTCMUD0		0026H	8026H
INTCMUD1		0028H	8028H

Table 3-1. Vector Table Area (2/2)

Interrupt source		Vector table address	
Interrupt request	Interrupt source/unit	TPF = 0	TPF = 1
INTSER	Asynchronous serial interface	002AH	802AH
INTSR		002CH	802CH
INTST		002EH	802EH
INTCSI0	Clock synchronous serial interface	0030H	8030H
INTCSI1		0032H	8032H
INTAD	A/D converter	0034H	8034H
Op-code trap	–	003CH	
BRK instruction	–	003EH	

3.1.2 CALLT instruction table area

Thirty-two tables of the address called by a single-byte call instruction (CALLT) can be stored in the 64-byte area from 0040H to 007FH. This area is the CALLT table area.

When the TPF bit of the CPU control word (CCW) is set, the internal ROM area 8040H to 807FH, instead of the area 0040H to 007FH, is used as a CALLT instruction table.

3.1.3 CALLF instruction entry area

A subroutine can be directly called for the area 0800H to 0FFFH by using a double-byte call instruction (CALLF).

3.1.4 Internal RAM area

2048-byte RAM is built into the area F700H to FEFFH.

This area consists of the following two components:

- Peripheral RAM : F700H to FDFFH (1792 bytes)
- Main RAM : FE00H to FEFFH (256 bytes)

High-speed access to the main RAM is possible.

Macro service control words are mapped in the 46-byte area from FE06H to FE35H in the main RAM area. General registers are mapped in the 128-byte area from FE80H to FEFFH in the main RAM area. The general registers consist of eight banks.

Cautions 1. When implementing word access (includes stack manipulation) to the main RAM area (FE00H to FEFFH), the access operation performed depends on whether the selected reference address is even or odd. (See Table 3-2.)
If both even and odd addresses are accessed, incorrect operations will be performed. Therefore, select only even reference addresses. (See Examples 1 and 2.) When executing the 16-bit data transfer instruction, select only even addresses by operand. If odd addresses are specified, errors will generate in the assembler package (RA78K/III).

2. Do not perform word access in both the peripheral RAM area and main RAM area. (See Example 3.)

Table 3-2. Operations when Word Accessing in the Internal RAM Area

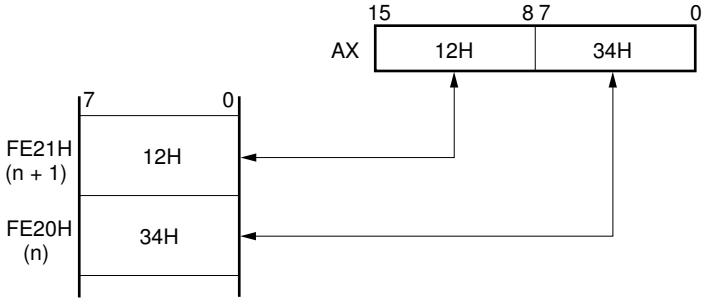
Access area \ Reference address (n)	Even	Odd
Main RAM	○	x
Peripheral RAM	○	○

Remark ○ : Accesses n address and n + 1 address
x : Accesses n address and n – 1 address

Examples 1 to 5 are examples of word accesses in the internal RAM area.

Examples 1. When writing/reading word data in/from even addresses (FE20H) in the main RAM area:
When the word data is written in even addresses (address n) in the main RAM area, the low-order 8 bits of the word data are written in the even address (address n) and the high-order 8 bits are written in the high-order odd address (address n + 1).
When the word data is read from the even address of the main RAM area (address n), the word data is read from address n and address n + 1.

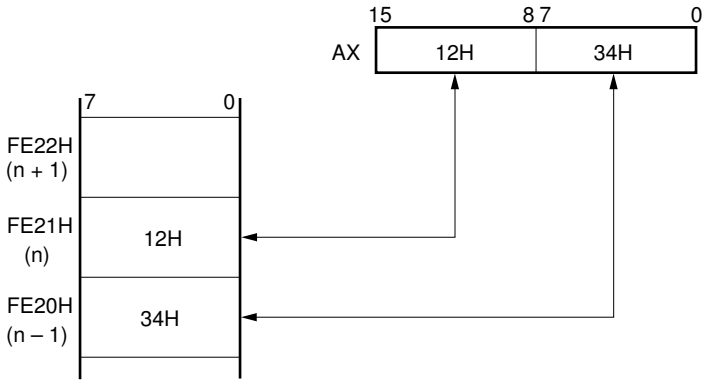
```
MOVW    AX, #1234H
MOVW    0FE20H, AX; Word data is written in FE20H
MOVW    AX, 0FE20H; Word data is read from FE20H
```



n : Reference address

2. When writing/reading word data in/from odd addresses (FE21H) in the main RAM area:
When the word data is written in odd addresses in the main RAM area, the high-order 8 bits of the word data are written in the odd address (address n) and the low-order 8 bits are written in the low-order even address (address n – 1).
When the word data is read from the odd address of the main RAM area (address n), the word data is read from address n and address n – 1.

```
MOVW    AX, #1234H
MOVW    DE, #0FE21H
MOVW    [DE], AX; Word data is written in FE21H
MOVW    AX, [DE]; Word data is read from FE21H
```

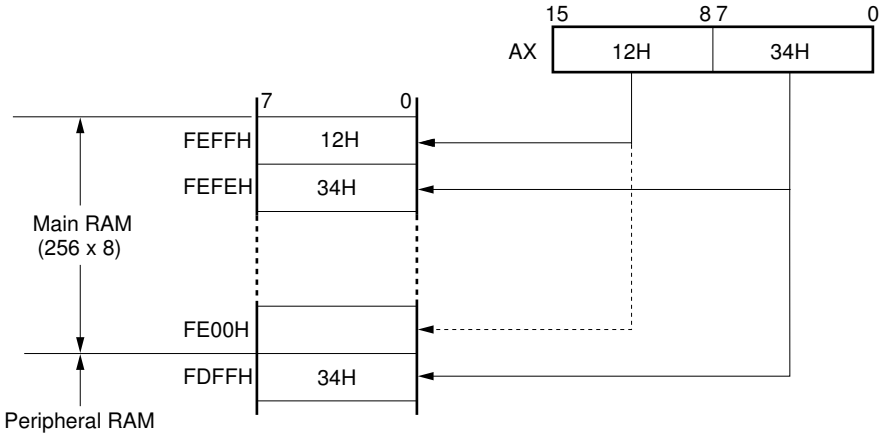


n : Reference address

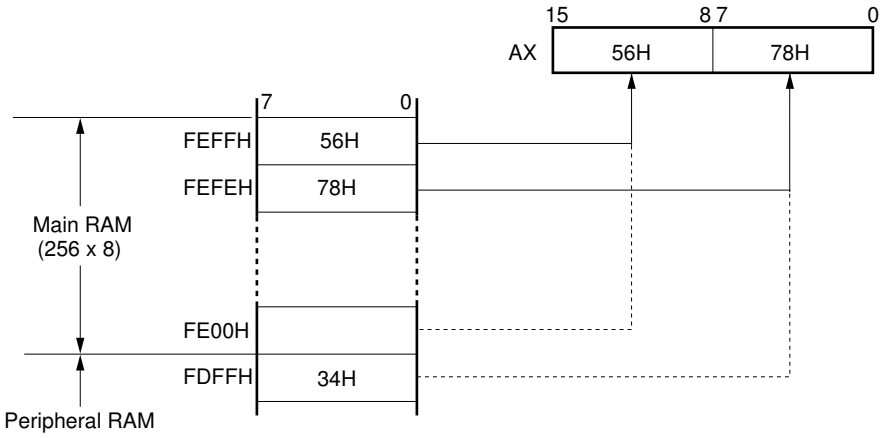
Examples 3. When writing/reading word data in/from both the peripheral RAM area and main RAM area: When the word data is written in both the peripheral RAM area and main RAM area, as it will be written in addresses separated by distance 256 bytes, incorrect operations will be performed. The word data is read from the last address (FDFFH) of the peripheral RAM area, it is read from FEFEH and FEFFH that are separated by distance 256 bytes.

```
MOVW    AX, #1234H
MOVW    DE, #0FDFFH
MOVW    [DE], AX; Word data is written in the peripheral RAM (FDFFH)
          .
          .
MOVW    DE, #0FDFFH
MOVW    AX, [DE]; Word data is read from the peripheral RAM (FDFFH)
```

(In Writing)

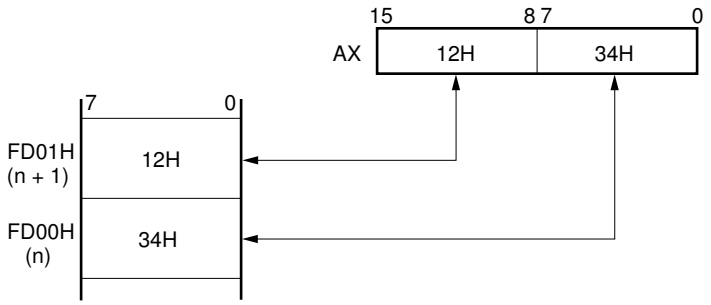


(In Reading)



Examples 4. When writing/reading word data in/from even addresses (FD00H) in the peripheral RAM area:
When the word data is written in even addresses in the peripheral RAM area, the low-order 8 bits of the word data area written in the even address (address n) and the high-order 8 bits are written in the high-order odd address (address n + 1).
When the word data is read from the even address of the peripheral RAM area (address n), the word data is read from address n and address n + 1.

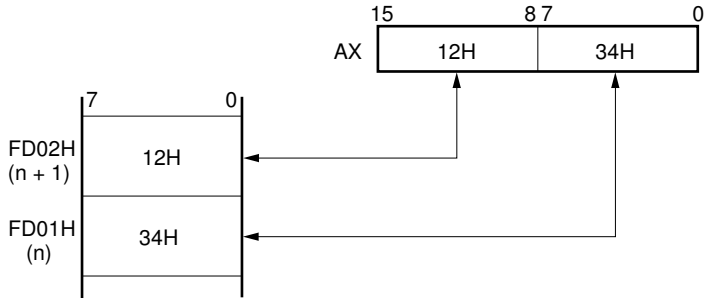
```
MOVW    AX, #1234H
MOVW    DE, #0FD00H
MOVW    [DE], AX; Word data is written in FD00H
MOVW    AX, [DE]; Word data is read from FD00H
```



n : Reference address

5. When writing/reading word data in/from odd addresses (FD01H) in the peripheral RAM area:
When the word data is written in odd addresses in the peripheral RAM area, the low-order 8 bits of the word data are written in the odd address (address n) and the high-order 8 bits are written in the high-order even address (address n + 1).
When the word data is read from the odd address of the peripheral RAM area (address n), the word data is read from address n and address n + 1.

```
MOVW    AX, #1234H
MOVW    DE, #0FD01H
MOVW    [DE], AX; Word data is written in FD01H
MOVW    AX, [DE]; Word data is read from FD01H
```



n : Reference address

3.1.5 Special function register area

Registers having special functions, such as mode and control registers for the peripheral hardware, are mapped in the area from FF00H to FFFFH.

Caution Unmapped addresses of the special function register cannot be accessed (except the external access area).

3.1.6 External memory area

The μPD78356 can expand an external memory (ROM or RAM) step by step to up to 14 Kbytes (C000H to F6FFH).

The μPD78355 can connect an external memory (ROM or RAM) in the 64-Kbyte area (0000H to F6FFH).

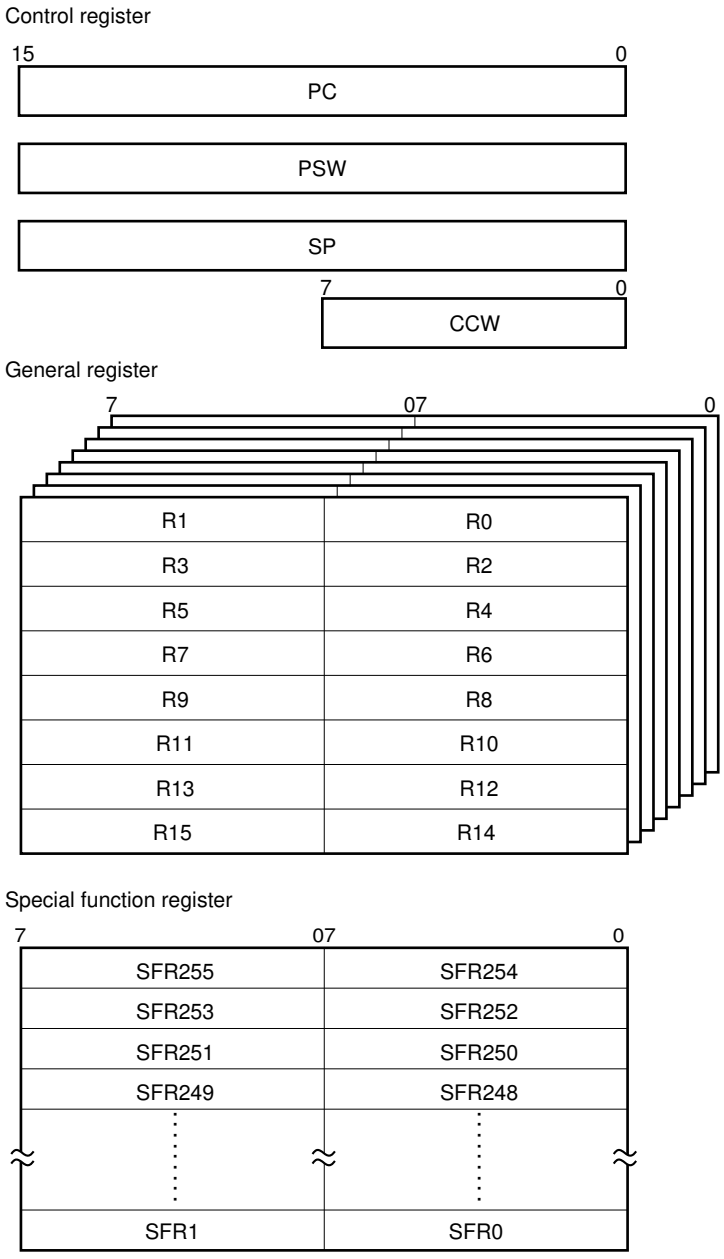
To access an external memory, the pins P40/AD0 to P47/AD7 (address/data bus), P50/AD8 to P57/AD15 (address/data bus), $\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$, and ASTB are used. An external access area is mapped in the 16-byte area from FFD0H to FFDFH of the special function register (SFR) area. SFR addressing can access an external memory in this area.

Caution Pins P50/AD8 to P57/AD15 function as address buses when 8-bit external buses are specified.

3.2 Processor Registers

There are three groups of registers: a control register group consisting of an 8-bit register and three 16-bit registers, a general register group consisting of eight banks, each of which consists of sixteen 8-bit registers, and a special function register group consisting of registers having special functions such as I/O mode registers for the peripheral hardware.

Fig. 3-2. Register Configuration



Remark The CCW of the control register group is mapped in the special function register (SFR) area.

3.2.1 Control registers

The control register group controls the program sequence, status, and stack memory and modifies operand addressing.

This group consists of an 8-bit register and three 16-bit registers.

(1) Program counter (PC)

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

- **In normal operation**
  Automatically incremented according to the number of bytes of the instruction to be fetched.
- **When a branch instruction is executed**
  The contents of the immediate data or register are set.

Input to the $\overline{\text{RESET}}$ pin sets the data in the reset vector table at 0000H and 0001H in the PC and makes a branch.

(2) Program status word (PSW)

The program status word (PSW) is a 16-bit register consisting of flags set or reset according to the result of executing an instruction.

Read and write operations are performed by the eight high-order bits (PSWH) or eight low-order bits (PSWL). Flags are operated by the bit manipulation instructions.

When an interrupt request is issued and when a BRK instruction is executed, the contents of the PSW is automatically saved in the stack. When an RETI or RETB instruction is executed, the contents are automatically restored.

Input to the $\overline{\text{RESET}}$ pin resets all bits.

Fig. 3-3. Format of Program Status Word

Symbol	7	6	5	4	3	2	1	0
PSWH	UF	RBS2	RBS1	RBS0	0	0	0	0

	7	6	5	4	3	2	1	0
PSWL	S	Z	RSS	AC	IE	P/V	0	CY

- UF : User flag
- RBS0 to RBS2 : Register bank selection flag
- S : Sign flag (MSB after arithmetic/logical operation)
- Z : Zero flag
- RSS : Register set selection flag
- AC : Auxiliary carry flag
- IE : Interrupt request enable flag
- P/V : Parity/overflow flag
- CY : Carry flag

The flags are explained below.

(a) User flag (UF)

This flag controls the program. The flag is set or reset on the user program.

(b) Register bank selection flag (RBS0 to RBS2)

This 3-bit flag selects one of eight register banks (register bank 0 to register bank 7).

(c) Sign flag (S)

The sign flag indicates that the most significant bit after an arithmetic/logical operation is 1.

This flag is set when the most significant bit after the operation is 1. Otherwise, this flag is reset.

This flag can be tested with a conditional branch instruction.

(d) Zero flag (Z)

The zero flag indicates that the result of an arithmetic/logical operation is 0.
This flag is set when the result is 0. Otherwise, this flag is reset.
This flag can be tested with a conditional branch instruction.

(e) Register set selection flag (RSS)

This flag specifies general registers (8 bits each) functioning as X, A, C, and B and general register pairs (16 bits each) functioning as AX and BC.
The RSS flag values correspond to function names and absolute names enclosed in parentheses as follows. (See **Table 3-3**.)

- **RSS = 0**
  X (R0), A (R1), C (R2), B (R3), AX (PR0), and BC (RP1)

- **RSS = 1**
  X (R4), A (R5), C(R6), B (R7), AX (RP2), and BC (RP3)

To set or reset the RSS flag, be sure to specify an RSS pseudo instruction just before or immediately after the instruction for setting or resetting the RSS flag as shown in the example below:

<Example of a program>

- To set the RSS flag (RSS = 0)

```
RSS  0          ; RSS pseudo instruction
CLR1 PSWL.5
MOV  B, A       ; Corresponds to MOV R3, R1.
```

- To reset the RSS flag (RSS = 1)

```
RSS  1          ; RSS pseudo instruction
SET1 PSWL.5
MOV  B, A       ; Corresponds to MOV R7, R5.
```

Switching the RSS flag between the values has the same effect as using two register sets. Registers and register pairs not specified by the RSS flag can be accessed by specifying the absolute names in the program.

(f) Auxiliary carry flag (AC)

The auxiliary carry flag is used for decimal adjustment and indicates that an underflow or overflow has occurred for bit 3.

This flag is set when the result of executing an arithmetic/logical instruction generates a carry of bit 3 (overflow) or a borrow into bit 3 (underflow). Otherwise, this flag is reset.

This flag can be tested with a conditional branch instruction.

(g) Interrupt request enable flag (IE)

This flag enables or disables an interrupt request.

Executing an EI instruction sets this flag. Executing a DI instruction or receiving an interrupt resets this flag.

(h) Parity/overflow flag (P/V)

The P/V flag can be tested with a conditional branch instruction.

When an arithmetic/logical instruction is executed, this flag operates as follows:

- **Parity flag operation**

  This flag is set if the number of set bits as the result of executing a logical instruction is an even number. Otherwise, this flag is reset. The parity flag depends on only the eight low-order bits of the logical operation result regardless whether the logical operation is performed in 8-bit units or 16-bit units.

- **Overflow flag operation**

  This flag is set if the result of executing an arithmetic instruction exceeds the two's complement range. Otherwise, the flag is reset.

  For example, the two's complement range for 8-bit arithmetic operation is from 80H (–128) to 7FH (+127). The flag is set if the result exceeds this range. The flag is reset if the result is within this range.

Example: The overflow flag operates as follows while executing an 8-bit add instruction.

When 78H (+120) is added to 69H (+105), the result is E1H (+225). The P/V flag is then set because the result exceeds the upper limit of the two's complement range. E1H is represented as –31 in two's complement.

```
  78H (+120)    =      0111 1000
+) 69H (+105)   =   +) 0110 1001
                       0  1110 0001  =  –31     P/V = 1
                       ↑
                       C
```

When the following two negative numbers are added, the P/V flag is reset because the result is within the two's complement range.

```
  FBH (–5)      =      1111 1011
+) F0H (–16)    =   +) 1111 0000
                       1  1110 1011  =  –21     P/V = 0
                       ↑
                       C
```

(i) Carry flag (CY)

The carry flag indicates that an overflow or underflow has occurred in an arithmetic/logical operation. This flag is set when an arithmetic/logical operation results in a carry (overflow) or a borrow (underflow) for bit 7. In word operations, this flag is set when a carry (overflow) or borrow (underflow) occurs for bit 15. Otherwise, this flag is reset.

This flag can be tested with a conditional branch instruction. This flag also functions as a bit accumulator when a bit manipulation instruction is executed.

(3) Stack pointer (SP)

The stack pointer (SP) is a 16-bit register which holds the first address of the memory stack area (LIFO format).
The SP is manipulated by a dedicated instruction (Stack manipulation instruction).
The stack pointer is decremented before data is written (saved) into the stack memory, and is incremented after data is read (restored) from the stack memory.
As input to the $\overline{\text{RESET}}$ pin causes the SP to become undefined, the SP must be set before calling a subroutine.

Caution When executing word access to the main RAM area (FE00H to FEFFH), only even address can be specified using operand.

(4) CPU control word (CCW)

The CPU control word (CCW) is an 8-bit register which consists of flags related to CPU control.
The CCW is mapped in the special function register area (FFC1H) and can be controlled by the software.
Input to the $\overline{\text{RESET}}$ pin resets all bits.

Fig. 3-4. Format of CPU Control Word

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
CCW	0	0	0	0	0	0	TPF	0	FFC1H	00H	R/W

TPF: Table position flag

The table position flag (TPF) specifies the location of a vector table referenced by a CALLT instruction or interrupt request. The TPF flag switches the vector table location as follows:

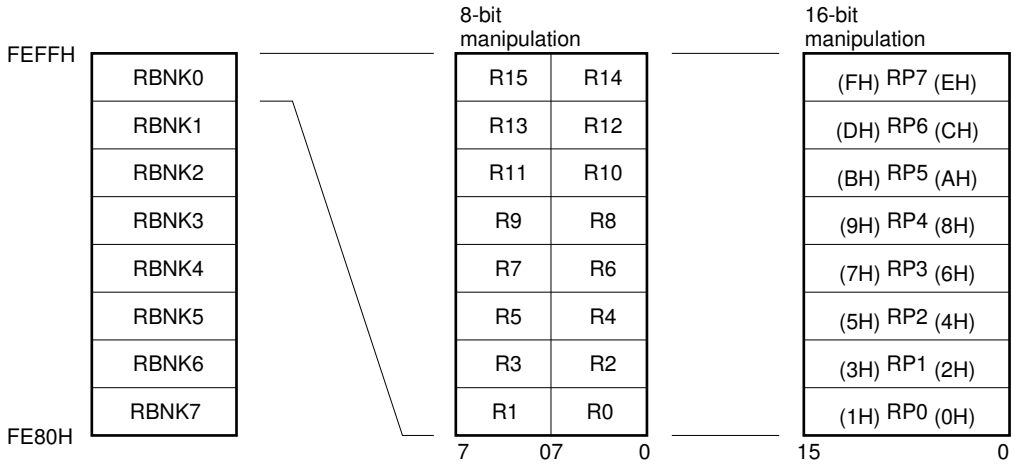
- TPF = 0 (reset)
  0000H to 007FH
- TPF = 1 (set)
  8000H to 807FH

Caution The vector tables for the $\overline{\text{RESET}}$ input, BRK instruction, and op-code trap interrupt are fixed to 0000H, 003EH, and 003CH, respectively. They are not affected by the TPF.

3.2.2 General registers

The 128-byte general register group which consists of eight banks is mapped in the specific area (FE80H to FEFFH) of the internal RAM space. Each bank consists of sixteen 8-bit general registers.

Fig. 3-5. Manipulation Bits of General Registers



A pair of 8-bit registers can function as a 16-bit register pair (RP0 to RP7).

A function name listed in Table 3-3, as well as an absolute name, is assigned to each 8-bit register (16 registers). The X register functions as the low-order bits of a 16-bit accumulator. The A register functions as an 8-bit accumulator or the high-order bits of a 16-bit accumulator. The B and C registers function as a counter. The DE, HL, VP, and UP registers, in a pair, function as an address register. The VP register functions as a base register and the UP register functions as a user stack pointer.

The value of the register set selection flag (RSS) in the program status word (PSW) changes the register having a specific function, as shown in Table 3-3.

If the program has been coded with the function names, operating the RSS flag has the same effect as using two sets of registers (X, A, B, C, AX, and BC). A register not specified by the RSS flag can be accessed by writing the absolute name in a program. When the RSS flag is 0, for example, register R4 can be accessed by specifying the absolute name, that is, R4 in the program.

The μPD78356 can implement two types of addressing: implied addressing and register addressing. Implied addressing is performed as process data addressing by a function name which places much importance on the specific function of each register. Register addressing is performed by an absolute name which aims to create a program which is easy to describe and which performs less data transfer, enabling high-speed data processing.

Table 3-3. General Register Configuration

(a) Correspondence between absolute names and function names for 8-bit registers

Absolute name	Function name	
	RSS = 0	RSS = 1
R0	X	
R1	A	
R2	C	
R3	B	
R4		X
R5		A
R6		C
R7		B
R8	VP_L	VP_L
R9	VP_H	VP_H
R10	UP_L	UP_L
R11	UP_H	UP_H
R12	E	E
R13	D	D
R14	L	L
R15	H	H

(b) Correspondence between absolute names and function names for 16-bit register pairs

Absolute name	Function name	
	RSS = 0	RSS = 1
RP0	AX	
RP1	BC	
RP2		AX
RP3		BC
RP4	VP	VP
RP5	UP	UP
RP6	DE	DE
RP7	HL	HL

3.2.3 Special function registers (SFRs)

Unlike the general registers, the special function registers (SFRs) have special functions. The SFRs are assigned to the memory space at addresses FF00H to FFFFH, namely, a 256-byte special function register area.

Short direct addressing is available for a 32-byte area at addresses FF00H to FF1FH. Data in the SFRs assigned to this area can be processed in fewer clock cycles because the word length of the SFRs in the area is less than that of the SFRs in other areas. These assigned SFRs consist of capture, compare, and port registers, and they are frequently accessed.

The 16-byte area at addresses FFD0H to FFDFH is used to access the external storage medium by SFR addressing. Instructions in short word length enable access to external memory or bit manipulation in the external device.

The SFRs can be manipulated by arithmetic/logical instructions, transfer instructions, bit manipulation instructions, or suchlike in the same way as general registers. The manipulatable bit units (1, 8, or 16 bit units) vary according to SFR. (See **Table 3-4**.)

The following describes the methods for specifying SFRs corresponding to manipulatable bit units:

- **Bit manipulation**
  Specify the abbreviation for the operand (sfr.bit) of a bit manipulation instruction. The SFR can also be addressed.
- **8-bit manipulation**
  Specify the abbreviation for the operand (sfr) of an 8-bit manipulation instruction. The SFR can also be addressed.
- **16-bit manipulation**
  Specify the abbreviation for the operand (sfrp) of a 16-bit manipulation instruction. A 16-bit SFR is assigned to a two-byte area at consecutive even and odd addresses. Specify an even address when addressing the SFR.

Table 3-4 lists the special function registers (SFRs). The items in Table 3-4 mean:

- Abbreviation ..... A symbol indicating the address of a incorporated SFR. This can be specified in the operand field of an instruction.
  Reserved word in the NEC assembler (RA78K/III).
  Can be used as sfr variables in the C compiler (CC78K/III) using the #pragma sfr instruction.
- R/W ..... Indicates whether data can be read from the special function register and/or data can be written into the register.
  R/W : Can be read and written.
  R : Can be read**Note**.
  W : Can be written.
- Manipulatable bit unit .... Indicates the unit of bits (1, 8, or 16) when manipulating the special function register (indicated by ○).
  The SFR which can be manipulated in 16-bit units can be specified in the sfrp operand.
  An even address is specified for the address specification.
  The SFR which can be manipulated in 1-bit units can be specified by a bit manipulation instruction.
- When reset ..... Indicates the status of each register when the $\overline{\text{RESET}}$ is input.

Note Read-only register. The bits of the register can be tested.

Cautions 1. Write 0 or 1 into any SFR bit correctly whenever it is predetermined to be so.

2. Do not write data into the register which is only used for data reading. Writing data into such registers may result in an error.

3. The SFR area (FF00H to FFFFH) addresses to which a special function register is not assigned cannot be accessed (except the external access area). Accessing these addresses may result in an error.

Table 3-4. Special Function Registers (1/6)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FF00H	Port 0	P0	R/W	○	○	–	Undefined
FF01H	Port 1	P1		○	○	–	
FF02H	Port 2	P2	R/W[Note 1]	○	○	–	
FF03H	Port 3	P3	R/W	○	○	–	
FF04H	Port 4	P4[Note 2]		○	○	–	
FF05H	Port 5	P5[Note 2]		○	○	–	
FF07H	Port 7	P7	R	○	○	–	
FF08H	Port 8	P8	R/W	○	○	–	
FF09H	Port 9	P9		○	○	–	
FF0AH	Port 10	P10		○	○	–	
FF10H	Capture/compare register 00	CC00		–	–	○	
FF11H							
FF12H	Capture/compare register 01	CC01		–	–	○	
FF13H							
FF14H	Capture/compare register 02	CC02		–	–	○	
FF15H							
FF16H	Capture/compare register 30	CC30		–	–	○	
FF17H							
FF18H	Capture/compare register 31	CC31		–	–	○	
FF19H							
FF1AH	Compare register 00	CM00		–	–	○	
FF1BH							
FF1CH	Compare register 01	CM01		–	–	○	
FF1DH							
FF1EH	Compare register 02	CM02		–	–	○	
FF1FH							
FF20H	Port 0 mode register	PM0		○	○	–	FFH
FF21H	Port 1 mode register	PM1		○	○	–	
FF22H	Port 2 mode register	PM2[Note 3]		○	○	–	
FF23H	Port 3 mode register	PM3		○	○	–	
FF25H	Port 5 mode register	PM5[Note 2]		○	○	–	
FF28H	Port 8 mode register	PM8		○	○	–	
FF29H	Port 9 mode register	PM9		○	○	–	0FH
FF2AH	Port 10 mode register	PM10		○	○	–	FFH

Notes **1.** Bit 0 : Read-only

2. Not provided for μPD78355.

3. Bit 0 is fixed at “1”.

Table 3-4. Special Function Registers (2/6)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FF30H	Timer register 0	TM0	R	–	–	○	0000H
FF31H							
FF32H	Timer register 1	TM1		–	–	○	
FF33H							
FF34H	Timer register 2	TM2		–	–	○	
FF35H							
FF36H	Timer register 3	TM3		–	–	○	
FF37H							
FF38H	Timer register 4	TM4		–	–	○	
FF39H							
FF3AH	Presettable up/down counter register	UDC	R/W	–	–	○	
FF3BH							
FF3CH	External interrupt mode register 0	INTM0		○	○	–	00H
FF3DH	External interrupt mode register 1	INTM1		○	○	–	
FF40H	Port 0 mode control register	PMC0		○	○	–	
FF42H	Port 2 mode control register	PMC2[Note]		○	○	–	01H
FF43H	Port 3 mode control register	PMC3		○	○	–	00H
FF44H	Pull-up resistor option register L	PUOL		○	○	–	
FF45H	Pull-up resistor option register H	PUOH		○	○	–	
FF48H	Port 8 mode control register	PMC8		○	○	–	
FF4AH	Port 10 mode control register	PMC10		○	○	–	
FF50H	Compare register 03	CM03		–	–	○	Undefined
FF51H							
FF52H	Compare register 10	CM10		–	–	○	
FF53H							
FF54H	Compare register 11	CM11		–	–	○	
FF55H							
FF56H	Compare register 20	CM20		–	–	○	
FF57H							
FF58H	Compare register 21	CM21		–	–	○	
FF59H							
FF5AH	Compare register 40	CM40		–	–	○	
FF5BH							

Note Bit 0 is fixed at “1”.

Table 3-4. Special Function Registers (3/6)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FF5CH	Up/down counter compare register 0	CMUD0	R/W	–	–	○	Undefined
FF5DH							
FF5EH	Up/down counter compare register 1	CMUD1		–	–	○	
FF5FH							
FF60H	Real-time output port register L	RTPL		○	○	–	
FF61H	Real-time output port register H	RTPH		○	○	–	
FF62H	Port read control register	PRDC		○	○	–	00H
FF63H	Real-time output port mode register	RTPM		○	○	–	
FF68H	A/D converter mode register 0	ADM0		○	○	–	
FF69H	A/D converter mode register 1	ADM1		○	○	–	07H
FF6AH	D/A conversion data setting register 0	DACS0		○	○	–	00H
FF6BH	D/A conversion data setting register 1	DACS1		○	○	–	
FF6AH	D/A conversion data setting register	DACS		–	–	○	0000H
FF6BH							
FF70H	Timer unit mode register 0	TUM0		○	○	–	00H
FF71H	Timer unit mode register 1	TUM1		○	○	–	
FF72H	Timer unit mode register 2	TUM2		○	○	–	
FF73H	Timer unit mode register 3	TUM3		○	○	–	
FF74H	Timer control register 0	TMC0		○	○	–	
FF75H	Timer control register 1	TMC1		○	○	–	
FF76H	Timer control register 2	TMC2		○	○	–	04H
FF77H	Up/down counter control register	UDCC		○	○	–	00H
FF78H	Timer output control register 0	TOC0		○	○	–	
FF79H	Timer output control register 1	TOC1		○	○	–	
FF7AH	Timer output control register 2	TOC2		○	○	–	
FF7BH	Timer overflow status register	TOVS	R/W**Note**	○	○	–	
FF7CH	Noise protection control register	NPC	R/W	○	○	–	
FF80H	Clock synchronous serial interface mode register 0	CSIM0		○	○	–	

Note Bits 7 and 6 : Always 0
Bits 5, 4, 2 and 1 : Read/write
Bits 3 and 0 : Read-only

Table 3-4. Special Function Registers (4/6)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FF82H	Serial bus interface control register	SBIC	R/W[Note]	○	○	–	00H
FF86H	Serial I/O shift register 0	SIO0	R/W	○	○	–	Undefined
FF88H	Asynchronous serial interface mode register	ASIM		○	○	–	80H
FF8AH	Asynchronous serial interface status register	ASIS	R	○	○	–	00H
FF8CH	Serial reception buffer : UART	RXB		–	○	–	Undefined
FF8EH	Serial transmission shift register : UART	TXS	W	–	○	–	
FF90H	Clock synchronous serial interface mode register 1	CSIM1	R/W	○	○	–	00H
FF96H	Serial I/O shift register 1	SIO1		○	○	–	Undefined
FFA0H	PWM control register	PWMC		○	○	–	00H
FFA2H	PWM buffer register 0L	PWM0L		○	○	–	Undefined
FFA2H	PWM buffer register 0	PWM0		–	–	○	
FFA3H							
FFA4H	PWM buffer register 1L	PWM1L		○	○	–	
FFA4H	PWM buffer register 1	PWM1		–	–	○	
FFA5H							
FFA8H	In-service priority register	ISPR	R	○	○	–	00H
FFAAH	Interrupt mode control register	IMC	R/W	○	○	–	80H
FFACH	Interrupt mask flag register	MK0L		○	○	–	FFH
FFACH	Interrupt mask flag register	MK0		–	–	○	FFFFH
FFADH							
FFADH	Interrupt mask flag register	MK0H		○	○	–	FFH
FFAEH	Interrupt mask flag register	MK1L		○	○	–	
FFAEH	Interrupt mask flag register	MK1		–	–	○	00FFH
FFAFH							
FFB0H	A/D conversion result register 0	ADCR0	R	–	–	○	Undefined
FFB1H							
FFB1H	A/D conversion result register 0H	ADCR0H		–	○	–	
FFB2H	A/D conversion result register 1	ADCR1		–	–	○	
FFB3H							
FFB3H	A/D conversion result register 1H	ADCR1H		–	○	–	

Note Bits 7 and 5 : Read/write
Bits 6, 3 and 2 : Read-only
Bits 4, 1 and 0 : Write-only

Table 3-4. Special Function Registers (5/6)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FFB4H	A/D conversion result register 2	ADCR2	R	–	–	○	Undefined
FFB5H							
FFB5H	A/D conversion result register 2H	ADCR2H		–	○	–	
FFB6H	A/D conversion result register 3	ADCR3		–	–	○	
FFB7H							
FFB7H	A/D conversion result register 3H	ADCR3H		–	○	–	
FFB8H	A/D conversion result register 4	ADCR4		–	–	○	
FFB9H							
FFB9H	A/D conversion result register 4H	ADCR4H		–	○	–	
FFBAH	A/D conversion result register 5	ADCR5		–	–	○	
FFBBH							
FFBBH	A/D conversion result register 5H	ADCR5H		–	○	–	
FFBCH	A/D conversion result register 6	ADCR6		–	–	○	
FFBDH							
FFBDH	A/D conversion result register 6H	ADCR6H		–	○	–	
FFBEH	A/D conversion result register 7	ADCR7		–	–	○	
FFBFH							
FFBFH	A/D conversion result register 7H	ADCR7H		–	○	–	
FFC0H	Standby control register	STBC[Note 1]	R/W	–	○	–	0000 x 000B
FFC1H	CPU control word	CCW		○	○	–	00H
FFC2H	Watchdog timer mode register	WDM[Note 1]		–	○	–	
FFC4H	Memory expansion mode register	MM		○	○	–	
FFC6H	Programmable wait control register	PWC		–	–	○	C0AAH[Note 2]
FFC7H							
FFD0H to FFDFH	External SFR area	–		○	○	–	Undefined
FFE0H	Interrupt control register (INTOV0)	OVIC0		○	○	–	43H
FFE1H	Interrupt control register (INTOV3)	OVIC3		○	○	–	
FFE2H	Interrupt control register (INTP0/INTCC00)	PIC0		○	○	–	
FFE3H	Interrupt control register (INTP1/INTCC01)	PIC1		○	○	–	
FFE4H	Interrupt control register (INTP2/INTCC02)	PIC2		○	○	–	
FFE5H	Interrupt control register (INTP3/INTCC30)	PIC3		○	○	–	
FFE6H	Interrupt control register (INTP4/INTCC31)	PIC4		○	○	–	
FFE7H	Interrupt control register (INTCM00)	CMIC00		○	○	–	
FFE8H	Interrupt control register (INTCM01)	CMIC01		○	○	–	
FFE9H	Interrupt control register (INTCM02)	CMIC02		○	○	–	

Notes 1. Writable only by special instructions

2. Becomes CFAAH only in the ROM-less mode of the external 16-bit bus (MODE0, 1 = HH).

Table 3-4. Special Function Registers (6/6)

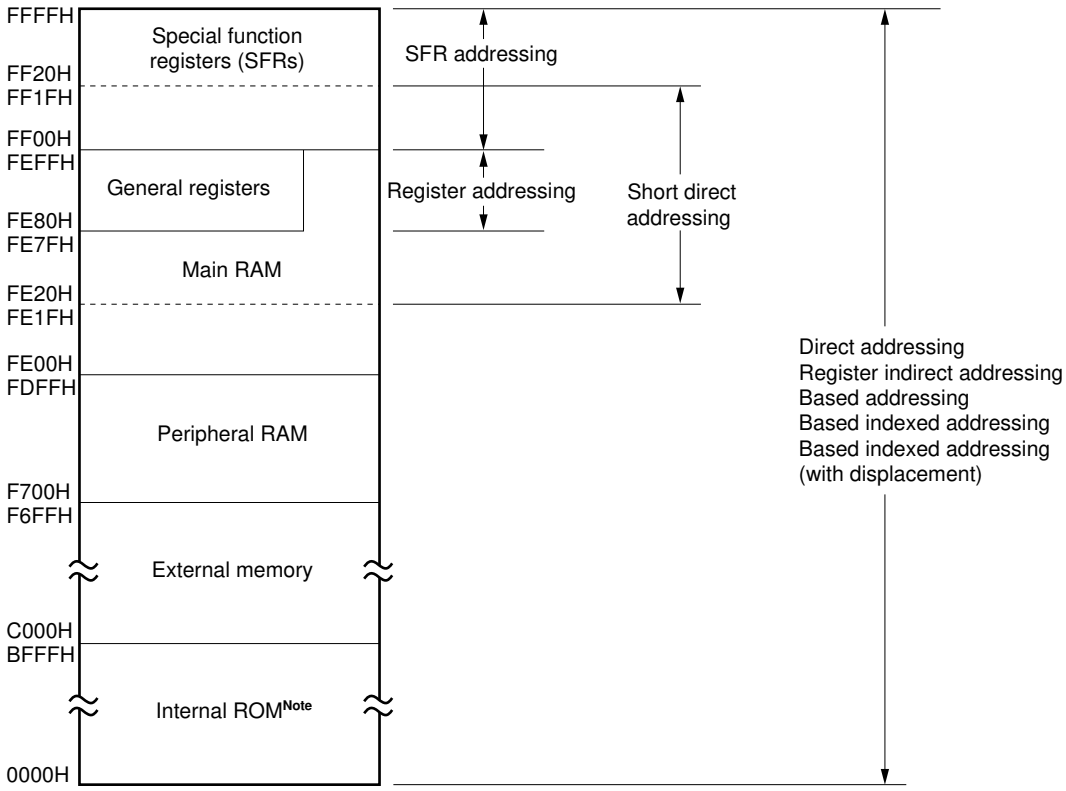
Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FFEAH	Interrupt control register (INTCM03)	CMIC03	R/W	○	○	–	43H
FFEBH	Interrupt control register (INTCM10)	CMIC10		○	○	–	
FFECH	Interrupt control register (INTCM11)	CMIC11		○	○	–	
FFEDH	Interrupt control register (INTCM20)	CMIC20		○	○	–	
FFEEH	Interrupt control register (INTCM21)	CMIC21		○	○	–	
FFEFH	Interrupt control register (INTCM40)	CMIC40		○	○	–	
FFF0H	Interrupt control register (INTCMUD0)	CMICUD0		○	○	–	
FFF1H	Interrupt control register (INTCMUD1)	CMICUD1		○	○	–	
FFF2H	Interrupt control register (INTSER)	SERIC		○	○	–	
FFF3H	Interrupt control register (INTSR)	SRIC		○	○	–	
FFF4H	Interrupt control register (INTST)	STIC		○	○	–	
FFF5H	Interrupt control register (INTCSI0)	CSIIC0		○	○	–	
FFF6H	Interrupt control register (INTCSI1)	CSIIC1		○	○	–	
FFF7H	Interrupt control register (INTAD)	ADIC		○	○	–	

3.3 Data Memory Addressing

Various addressing modes are provided for the μPD78356 to improve memory operability or to enable the use of a high-level language. Special addressing is applicable, in particular, to the space of data memory from F700H to FFFFH according to each function of the special function register (SFR) group and general register group.

Fig. 3-6 shows the addressing space of data memory.

Fig. 3-6. Addressing Space of Data Memory



Note In the ROM-less mode of the μPD78355 or μPD78356, data memory is used as external memory.

Caution When executing word access (including stack manipulation) to the main RAM area (FE00H to FEFFH), only even address can be specified using operand.

3.3.1 General-register addressing

(1) Implied addressing

The instruction automatically addresses the register that functions as an accumulator (A or AX) or loop counter (B or C) assigned to the general-register area.

Coding example: MULU r

If the value stored in register B is used as a multiplier for a multiply instruction of 8 bits by 8 bits, code the following:

MULU B; AX <- A x B

The instruction performs multiply operation between the accumulator (register A) and register B and stores the result in the 16-bit accumulator (register AX).

(2) Register addressing

The instruction directly addresses the desired registers.

Coding example: ADD r,r

To specify registers D and E storing the target values for the 8-bit add instruction, code the following:

ADD D, E; D <- D + E

3.3.2 Short direct addressing

This addressing is used for accessing the internal RAM area at addresses FE20H to FEFFH and the SFR area at addresses FF00H to FF1FH. Short direct addressing enables high-speed access to these areas by a short instruction code.

Specify an even address when manipulating 16-bit data.

Coding example: ADD A, saddr

If one target value of the 8-bit add instruction is already stored in the location at address FE80H in internal data memory, code the following:

ADD A, 0FE80H; A <- A + (FE80H)

3.3.3 Special function register (SFR) addressing

This addressing is used for manipulating SFRs mapped in the SFR area at addresses FF00H to FFFFH.

Coding example: MOV A, sfr

If a special function register is specified as a transfer source for port 0 in the SFR area, code the following 8-bit transfer instruction:

MOV A, P0; A <- P0

[MEMO]

CHAPTER 4 SUMMARY OF BLOCK FUNCTION

4.1 Execution Unit

The execution unit (EXU) controls address calculation, arithmetic/logical operations, and data transfer by a microprogram.

The EXU contains a 256-byte main RAM. Eight register banks are addressed to the main RAM in the EXU.

4.2 Bus Control Unit

The bus control unit (BCU) activates a required bus cycle according to the physical address obtained from the execution unit (EXU). When the EXU does not issue a bus cycle activation request, the BCU generates an address required for prefetching an instruction. The prefetched instruction code is fetched into the instruction queue.

8-bit or 16-bit external data bus can be used (bus sizing function).

The number of bytes held in the instruction queue depends on the area from which the instruction is fetched.

- Fetched from internal memory[Note] ..... 5 bytes
- Fetched from external memory ..... 3 bytes

Note Internal memory: Internal ROM (only in the μPD78356 and μPD78P356), peripheral RAM

4.3 Program Memory and Data Memory

The μPD78356 and μPD78P356 contain a 48-Kbyte program memory (ROM) and a 2048-byte data memory (RAM). The μPD78355 contains only a 2048-byte data memory (RAM). It does not have a ROM.

The 2048-byte data memory consists of 256-byte main RAM contained in the EXU and 1792-byte is peripheral RAM.

4.4 Ports

As indicated below, the ports have function as control pins as well as general ports.

Port name	I/O	Multiple functions available	
Port 0	8-bit I/O	General port	Real-time output port, external trigger input for the A/D converter
Port 1	8-bit I/O		–
Port 2	8-bit I/O		External interrupt input, RPU capture trigger input, clear signal input and timer output
Port 3	8-bit I/O		Serial interface I/O, RPU count pulse input, clear signal input and timer output
Port 4	8-bit I/O		Address/data bus (AD0 to AD7)
Port 5	8-bit I/O		Address/data bus (AD8 to AD15)
Port 7	8-bit input		Analog input for the A/D converter
Port 8	8-bit I/O		RPU count pulse input, clear signal input and timer output, PWM signal output
Port 9	4-bit I/O		External access control signal output
Port 10	8-bit I/O		Serial interface I/O, RPU count pulse input

Remark RPU: Real-time pulse unit

4.5 Real-Time Pulse Unit

The real-time pulse unit (RPU) consists of the following hardware.

- Five 16-bit timers
- One 10-bit timer
- Ten 16-bit compare registers
- One 10-bit compare register
- Five 16-bit capture/compare registers
- Ten Timer outputs

The RPU can measure a pulse width and frequency and output a programmable pulse.
It also can control the output timing of the real-time output port.

4.6 Real-Time Output Port

The real-time output port (RTP) can control the output timing of the port by the signal from the RPU as a trigger pulse. The port can be used in 1-bit units. This port is also used as port 0 and can output 8 real-time pulses.

4.7 A/D Converter

This converter is a 10-bit very high-speed high-resolution A/D converter having 8 analog input pins. The A/D converter uses the serial-parallel conversion system.

4.8 D/A Converter

Two D/A converters of 8-bit resolution voltage output are contained. The converter uses the resistor string system for conversion.

4.9 Serial Interfaces

Three independent serial interfaces are provided as listed below, and a baud rate generator common to the three interfaces is incorporated. The two clock synchronous serial interfaces have the operating modes indicated below.

- Asynchronous serial interface (UART)
- Synchronous serial interface
  - Serial bus interface (SBI) mode
  - Three-wire serial I/O mode
- Synchronous serial interface (with pin switching function)
  - Three-wire serial I/O mode

4.10 PWM Output Unit

The μPD78356 has two PWM signal outputs of 8/10/12-bit resolution. By externally connecting a low-pass filter, a PWM output can be used as a digital-to-analog conversion output. The PWM outputs are most suitable, for example, for a control signal for the actuator of a motor.

4.11 Watchdog Timer

The watchdog timer is a free-running counter with a nonmaskable interrupt function designed to prevent crashes or deadlocks. A program error can be detected when a watchdog timer overflow interrupt (INTWDT) is generated or when the watchdog timer output pin ($\overline{\text{WDTO}}$) goes low. By connecting this output to the $\overline{\text{RESET}}$ pin, abnormal application system operation caused by a program error can be prevented.

4.12 Interrupt Controller

The interrupt controller processes various interrupt requests (NMI and INTP0 to INTP4) issued from peripheral hardware and external device with the vectored interrupt, macro service, or context switching. The interrupt controller also allows programmable specification of the 4-level interrupt priority by software.

[MEMO]

CHAPTER 5 PORT FUNCTIONS

5.1 Hardware Configuration

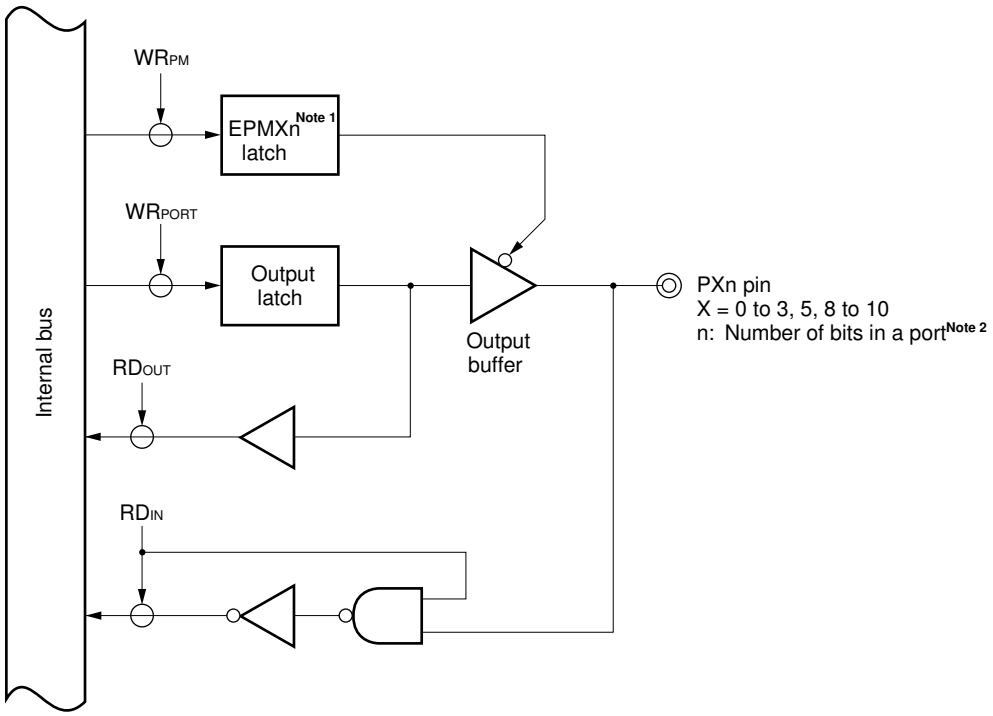
As shown in Fig. 5-1, three-state bidirectional ports are basically used for the ports of the μPD78356. (P20 of port 2 and port 7 are used only for input.)

A $\overline{\text{RESET}}$ input signal sets each bit of a port mode register to 1, specifying the port as an input port. All port pins go into the high-impedance state. The contents of the output latch become undefined by a $\overline{\text{RESET}}$ input signal.

Before specifying the port as an output port, set data in the output latch.

Caution For the μPD78355 and for the μPD78356 and μPD78P356 in the ROM-less mode, ports 4, 5, and 9 (low-order 3 bits) do not function as ports. For details, see 5.2.

Fig. 5-1. Basic I/O Port Configuration



Notes 1. PMXn latch: Bit n in the port mode register PMX (X = 0 to 3, 5, 8 to 10)

2. When X = 0, 1, 3 to 5, 8, 10, n = 0 to 7
When X = 2, n = 1 to 7
When X = 9, n = 0 to 3

Remarks 1. P20 of port 2 is used only for input. Port 7 is used only for input.

2. Port 4 can be specified as an input or output port by the memory expansion mode register (MM).

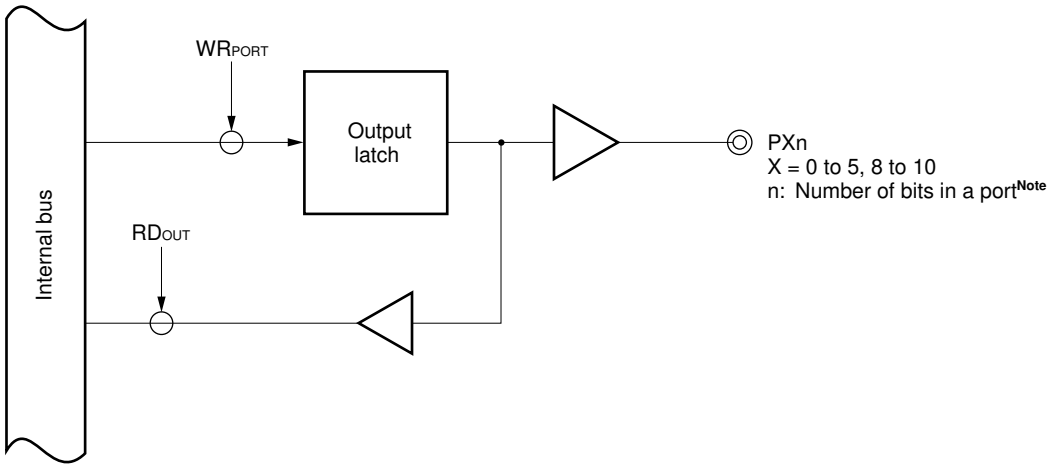
(1) When a port is specified as an output port

The output latch is enabled, and a transfer instruction can transfer data between the output latch and accumulator.

The contents of the output latch can be set or reset bit by bit. Data once written to the output latch are held until another instruction is executed to operate the port.

When a port specified as an output port is read using an instruction such as a transfer instruction, the contents of the output latch are read.

Fig. 5-2. Port Specified as an Output Port



Note When X = 0, 1, 3 to 5, 8, 10, n = 0 to 7
When X = 2, n = 1 to 7
When X = 9, n = 0 to 3

Remark P20 of port 2 is used only for input.

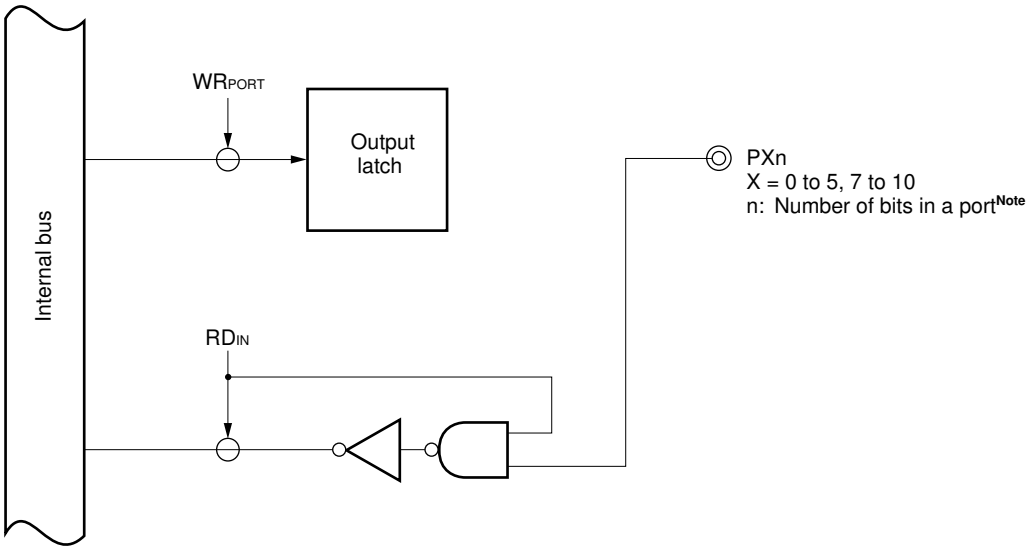
(2) When a port is specified as an input port

The levels of the port pins can be loaded into the accumulator with a transfer instruction. Even when the port is specified as an input port, writing to the output latch is possible. Data transferred from the accumulator with a transfer instruction are all stored in the output latch, regardless of whether the port is specified as an input port or output port.

However, the data is not output to the port pins because the output buffer for the bits have become high-impedance. (When the bits specified as an input are switched to an output port, the contents of the output latch are output to the port pins.)

The contents of the output latch of the bits specified as an input port cannot be loaded into the accumulator. (See **Fig. 5-3**.)

Fig. 5-3. Port Specified as an Input Port



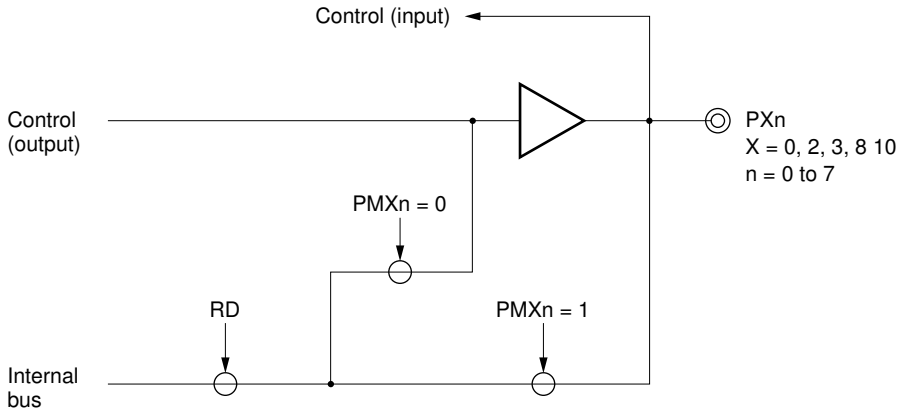
Note When X = 0 to 5, 7, 8, 10, n = 0 to 7
When X = 9, n = 0 to 3

(3) When a port is specified for control signal input or output

Ports 0, 2, 3, 8, and 10 can be used for control signal input or output on a bit-by-bit basis by setting the desired bit(s) of the port mode control registers (PMC0, PMC2, PMC3, PMC8, and PMC10) to 1. In this case, the settings of the port mode registers (PM0, PM2, PM3, PM8, and PM10) have no effect.

When a pin or pins are used for a control signal, the state of the control signal can be checked by executing a port read instruction.

Fig. 5-4. When the Control Is Specified



(a) When a port is specified for control signal output

The pin state of a control signal can be read by executing a port read instruction when the desired bit of the port mode register is set to 1.

The state of an internal control signal can be read by executing a port read instruction when the desired bit of the port mode register is reset to 0.

(b) When a port is specified for control signal input

The pin state of a control signal can be read by executing a port read instruction only when the desired bit of the port mode register is set to 1.

Caution The pin functioning as an input pin in the control mode may not operate correctly if the desired bit of the port mode control register is rewritten during the operation. Therefore writing to the port mode control register should be executed at initial settings of the system.

(4) Port output data check function

The μPD78356 has a function that enables pin state to be read in the port output mode so that application system reliability can be improved (pin access mode). With this function, output data and actual pin state can be checked as required. If a mismatch is found, an action such as replacement with another system can be taken.

Before pin state can be read, bit 0 of the port read control register (PRDC) must be set to 1.

A $\overline{\text{RESET}}$ input signal sets register PRDC to 00H.

Fig. 5-5. Port Read Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PRDC	0	0	0	0	0	0	0	PRDC0	FF62H	00H	R/W

PRDC0	Operation mode
0	Normal mode
1	Pin access mode

Example: HA sample program is given below which checks output data to port 0 (P0), port 1 (P1), and port 3 (P3) by using the pin access mode.

```
TEST:   DI              ; Disables interrupt
        MOV     A,#5AH  ; Test data = 5AH
        MOV     P0,A    ; Sets 5AH in output latch
        MOV     P1,A
        MOV     P3,A
        SET1    PRDC.0  ; Sets pin access mode (sets PRDC)
        CMP     A,P0    ; Compares pin level with output latch contents
        BNE     $ERR0   ; Performs error processing if mismatch occurs
        CMP     A,P1
        BNE     $ERR1
        CMP     A,P3
        BNE     $ERR3
        CLR1    PRDC.0  ; Normal mode (resets PRDC)
        EI              ; Enables interrupt
```

Cautions 1. In the pin access mode (PRDC0 = 1), no bit manipulation instruction for a port operates normally. After a port check is completed, be sure to reset the mode to the normal mode (PRDC0 = 0).

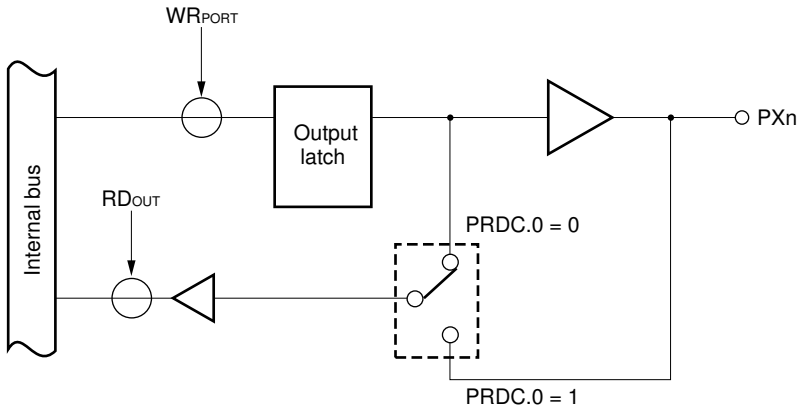
2. If an interrupt occurs in the pin access mode, a bit manipulation instruction may be executed in the same mode. This will cause an error. Before starting a check, be sure to set the DI state.
In addition, do not use macro services that manipulate ports.

3. Nonmaskable interrupts are unavoidable. So, the following provisions should be made in the program as required:

- **The nonmaskable interrupt routine is to perform no port manipulation.**
- **The level of PRDC.0 is to be saved at the start of the nonmaskable interrupt routine, then is restored when control is returned.**

When PRDC.0 is set to 1, the switch enclosed in dotted lines in the figure below is connected to the pin and the pin state is read. If a bit manipulation instruction is executed in this state, the pin state is read and a bit operation is executed. This may adversely affect the contents of the output latch.
When PRDC.0 is reset to 0, the system enters the normal mode.

Fig. 5-6. Control (Output Port Specified)



In addition, instructions (CHKL, CHKLA) dedicated to frequent port state checking are available. By EXCLUSIVE ORing, these instructions compare the pin state with the contents of the output latch (in the port mode) or the pin state with the level of the internal control output signal (in the control mode).

Example: A sample program is given below which checks pin state and the contents of an output latch using the CHKL or CHKLA instruction.

```
TEST:   SET1    P0.3        ; Sets bit 3 of port 0
        CHKL    P0          ; Checks port 0
        BNE     $ERR1       ; Branches to error processing (ERR1) if mismatch with
                            ;  output latch contents occurs
          .
          .
          .
ERR1:   CHKLA   P0          ; Check incorrect bits
        BT      A.7,$BIT07  ; Bit 7?
        BT      A.6,$BIT06  ; Bit 6?
          .
          .
          .
        BT      A.1,$BIT01  ; Bit 1?
        BR      $BIT00      ; Bit 0 is incorrect if all above bits are correct.
```

Cautions 1. Use the CHKL or CHKLA instruction only when the PRDC0 bit of the PRDC register is set to 0 (normal mode).

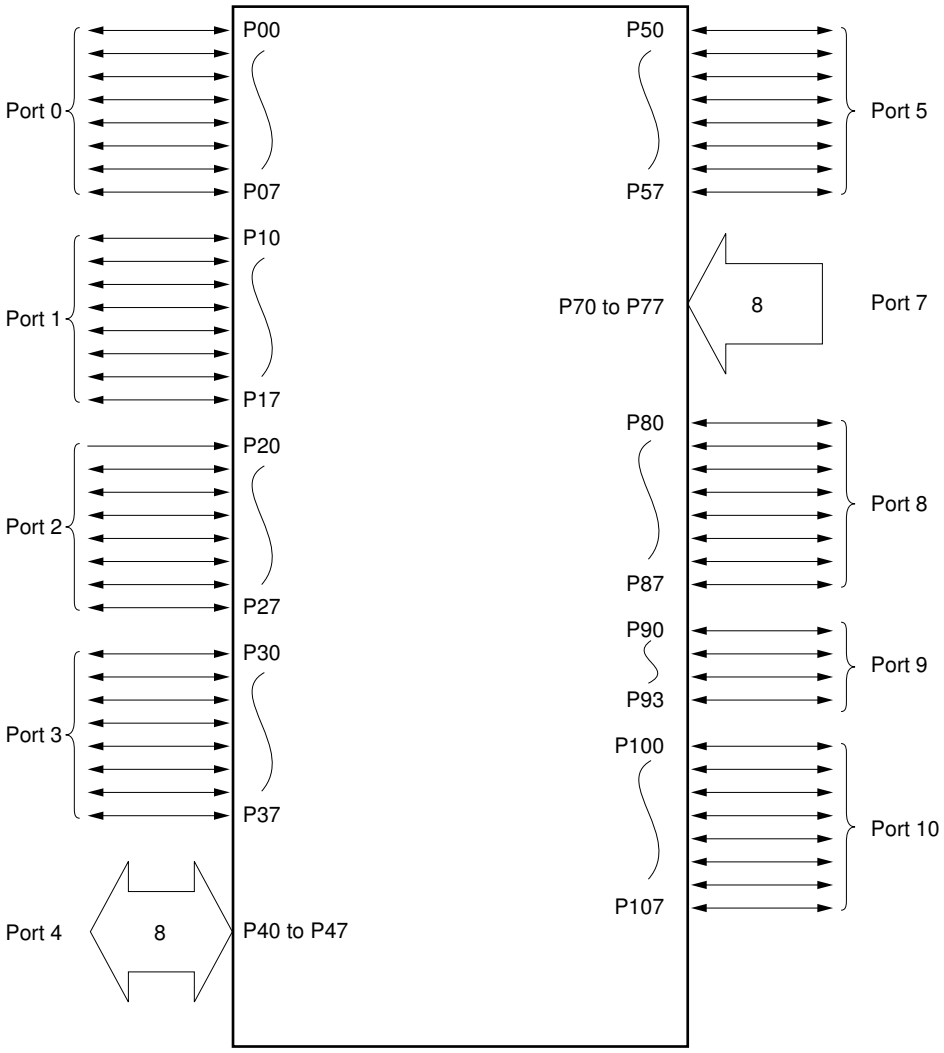
2. Concerning the I/O port pin, the check result of the pin which is set to input port mode will always be the same between the CHKL and CHKLA instructions regardless of the setting of the port mode/control mode. Concerning the input dedicated ports, the input pin level will always be read by the CHKL and CHKLA instructions because there is no latch. Therefore, for the input dedicated ports, the CHKL and CHKLA instructions are essentially ineffective.

3. Set control output pins to the input mode before executing the CHKL or CHKLA instruction to check the output level of the pin of a port where control and port output pins are used together. (The output level of a control pin cannot be checked with the CHKL or CHKLA instruction because the output level varies asynchronously.)

5.2 Functions of the Ports

The μPD78356 is provided with the ports shown in Fig. 5-7, which enable various types of control.

Fig. 5-7. Port Configuration



5.2.1 I/O port functions and alternate functions

Table 5-1 lists the function of each port. Each port functions as an I/O port and also functions as I/O pins for internal hardware.

Table 5-1. Port Functions and Alternate Functions of the Ports

Port name	Port function	Alternate function
Port 0	8-bit I/O port. Specifiable as input or output in 1-bit units.	Real-time output port (RTP) or external trigger input for A/D converter in control mode
Port 1	8-bit I/O port. Specifiable as input or output in 1-bit units.	—
Port 2	8-bit I/O port. Specifiable as input or output in 1-bit units. (P20 is excluded.)	Capture trigger input, clear signal input, or timer output for real-time pulse unit (RPU), or external interrupt input in control mode
Port 3	8-bit I/O port. Specifiable as input or output in 1-bit units.	Count pulse input, clear signal input, or timer output for real-time pulse unit (RPU), or I/O for serial interface (UART, CSI) in control mode
Port 4	8-bit I/O port. Specifiable as input or output in 8-bit units.	Address/data bus (AD0 to AD7) for memory expansion
Port 5	8-bit I/O port. Specifiable as input or output in 1-bit units.	Address/data bus (AD8 to AD15) for memory expansion
Port 7	Port used only for 8-bit input.	Analog input for A/D converter in control mode
Port 8	8-bit I/O port. Specifiable as input or output in 1-bit units.	Count pulse input, clear signal input, or timer output for real-time pulse unit (RPU), or PWM signal output in control mode
Port 9	4-bit I/O port. Specifiable as input or output in 1-bit units.	Control signal output for memory expansion
Port 10	8-bit I/O port. Specifiable as input or output in 1-bit units.	Count pulse input for real-time pulse unit (RPU) or I/O for serial interface in control mode

5.2.2 I/O mode setting

(1) Port n (n = 0 to 3, 5, 8, 9, 10)

The I/O mode of each port is set in 1-bit units with a port mode register (PM). (See **Fig. 5-8** to **5-15**.) Each pin of a port functions as an input or output port, depending on the setting of the corresponding bit of the PM register. It is an input port when the bit is one, and an output port when the bit is zero. Bit 0 of the port 2 mode register (PM2) is always one.

The contents of each PM register are specified by an 8-bit manipulation instruction.

(2) Port 7

Port 7 functions only as input port. Port 7 has no port mode register.

(3) Port 4

The I/O mode of port 4 only is set with the memory expansion mode register (MM) in 8-bit units. (See **Fig. 5-21**.)

Register MM is set with a bit or 8-bit manipulation instruction.

Fig. 5-8 to 5-15 show the format of each port mode instruction register.

Fig. 5-8. Format of the Port 0 Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00	FF20H	FFH	R/W

PM0n	I/O mode of the P0n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Fig. 5-9. Format of the Port 1 Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10	FF21H	FFH	R/W

PM1n	I/O mode of the P1n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Fig. 5-10. Format of the Port 2 Mode Register

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

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

Caution Bit 0 of the port 2 mode register is always one.

Fig. 5-11. Format of the Port 3 Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	FF23H	FFH	R/W

PM3n	I/O mode of the P3n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Fig. 5-12. Format of the Port 5 Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	FF25H	FFH	R/W

PM5n	I/O mode of the P5n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Fig. 5-13. Format of the Port 8 Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM8	PM87	PM86	PM85	PM84	PM83	PM82	PM81	PM80	FF28H	FFH	R/W

PM8n	I/O mode of the P8n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Fig. 5-14. Format of the Port 9 Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM9	0	0	0	0	PM93	PM92	PM91	PM90	FF29H	0FH	R/W

PM9n	I/O mode of the P9n pin (n = 0 to 3)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Fig. 5-15. Format of the Port 10 Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM10	PM107	PM106	PM105	PM104	PM103	PM102	PM101	PM100	FF2AH	FFH	R/W

PM10n	I/O mode of the P10n pin (n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

5.2.3 Control mode setting

(1) Port n (n = 0, 2, 3, 8, 10)

The control mode of each port is set in 1-bit units with a port mode control register (PMC). (See **Fig. 5-16** to **5-20**.)

Each pin of port n functions as a control pin when the corresponding bit of the PMC register is set to 1. In this case, the previous states of the port and the set value in the PMC register have no effect.

The contents of each PMC register are specified by an 8-bit manipulation instruction.

(2) Port 7

The pins of port 7 are always set in the control mode. Port 7 has no port mode control register.

The pin state of each port can be read by executing a port read instruction.

(3) Port 4 and port 5

(a) μPD78356

The control mode is set according to the contents of the memory expansion mode register (MM) (see **Fig. 5-21**). As shown in Table 5-2, the operations of ports 4 and 5 depend on the expansion mode of the MM register. When the MM register is in the expansion mode, port 4 functions as an address/data bus and some bits of port 5 function as address bus.

When data is written or verified for the μPD78P356 in the PROM programming mode, port 4 functions as an I/O port for the data.

The contents of the MM register are specified by a bit manipulation instruction or an 8-bit manipulation instruction.

Table 5-2. Operation of Port 4 and Port 5 (μPD78356)

<table>
<tr><td colspan="2" rowspan="2">Setting of MM register</td><td>Operation of port 4</td><td colspan="8">Operation of port 5</td></tr>
<tr><td>P40 to P47</td><td>P50</td><td>P51</td><td>P52</td><td>P53</td><td>P54</td><td>P55</td><td>P56</td><td>P57</td></tr>
<tr><td colspan="2" rowspan="2">Port mode
(single chip mode)</td><td>Input port</td><td colspan="8" rowspan="2">General I/O port</td></tr>
<tr><td>Output port</td></tr>
<tr><td rowspan="4">Expansion mode</td><td>256-byte expansion</td><td rowspan="4">Address/data bus</td><td colspan="8" rowspan="2"></td></tr>
<tr><td>4-Kbyte expansion</td><td rowspan="3">AD8</td><td rowspan="3">AD9</td><td rowspan="3">AD10</td><td rowspan="3">AD11</td><td colspan="4">General I/O port</td></tr>
<tr><td>16-Kbyte expansion</td><td>AD12</td><td>AD13</td><td colspan="2">General I/O port</td></tr>
<tr><td>Full expansion</td><td colspan="2">AD14</td><td>AD15</td></tr>
</table>

Remark ADn (n = 8 to 15): Address bus

(b) μPD78355

The ports operate as described below. One pin of the port does not function as a port.

- Port 4 : Lower address/data bus (AD0 to AD7)
- Port 5 : Higher address/data bus (AD8 to AD15)

This also applies when the μPD78356 is used in the ROM-less mode.

(4) Port 9 ★

The control mode is set according to the contents of the memory expansion mode register (MM).

(a) μPD78356

(i) External 8-bit bus

Port 9 operates as indicated in Table 5-3.

When the expansion mode is set, the P90 and P91 pins function as the $\overline{RD}$ and $\overline{LWR}$ pins, respectively, and the P92 and P93 pins function as ports. When the μPD78356 is placed in the ROM-less mode, the P90 and P91 pins function as the $\overline{RD}$ and $\overline{LWR}$ pins, respectively, and P92 and P93 pins as ports at all times. In this case, the MM0 to MM2 bits have no effect.

Table 5-3. μPD78356 Port 9 Operation (External 8-bit Bus)

Setting of MM register	Operation of port 9			
MM0 to MM2	P90	P91	P92	P93
Port mode	General-purpose port			
Expansion mode	$\overline{RD}$	$\overline{LWR}$	General-purpose port	

(ii) External 16-bit bus

Port 9 operates as indicated in Table 5-4.

When the expansion mode is set, the P90, P91, and P92 pins function as the $\overline{RD}$, $\overline{LWR}$, and $\overline{HWR}$ pins respectively, and the P93 pin function as a port. When the μPD78356 is placed in the ROM-less mode, the P90, P91, and P92 pins function as the $\overline{RD}$, $\overline{LWR}$, and $\overline{HWR}$ pins, respectively, and P93 pin as ports at all times. In this case the bits MM0 to MM2 cannot be set.

Table 5-4. μPD78356 Port 9 Operation (External 16-bit Bus)

Setting of MM register	Operation of port 9			
MM0 to MM2	P90	P91	P92	P93
Port mode	General-purpose port			
Expansion mode	$\overline{RD}$	$\overline{LWR}$	$\overline{HWR}$	General-purpose port

(b) μPD78355

(i) External 8-bit bus

The P90 and P91 pins function as the $\overline{RD}$ and $\overline{LWR}$ pins, respectively, and the P92 and P93 pins function as ports at all times. The lower three bits (MM0 to MM2) of the MM register cannot be set.

(ii) External 16-bit bus

The P90, P91, and P92 pins function as the $\overline{RD}$, $\overline{LWR}$, and $\overline{HWR}$ pins, respectively, and the P93 pin as a port. The lower three bits (MM0 to MM2) of the MM register cannot be set.

Fig. 5-16. Format of the Port 0 Mode Control Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PMC0	PMC07	PMC06	PMC05	PMC04	PMC03	PMC02	PMC01	PMC00	FF40H	00H	R/W

PMC0n	Control mode of the P0n pin (n = 0 to 3)
0	I/O port mode
1	Real-time output port mode/ADTRG input mode[Note]

PMC0n	Control mode of the P0n pin (n = 4 to 7)
0	I/O port mode
1	Real-time output port mode

Note Pins P00 to P03 function as external trigger input pins (ADTRG0 to ADTRG3) of the A/D converter when the TRG bit 2 of the A/D converter mode register 1 (ADM1) is “1”. (See **Fig. 8-4**.)

Fig. 5-17. Format of the Port 2 Mode Control Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PMC2	PMC27	PMC26	PMC25	PMC24	PMC23	PMC22	PMC21	1	FF42H	01H	R/W

PMC21	Control mode of the P21 pin
0	I/O port mode
1	INTP0 input mode/TO04 output mode[Note 1]

PMC22	Control mode of the P22 pin
0	I/O port mode
1	INTP1 input mode/TO05 input mode[Note 1]

PMC23	Control mode of the P23 pin
0	I/O port mode
1	INTP2 input mode

PMC24	Control mode of the P24 pin
0	I/O port mode
1	INTP3 input mode

PMC25	Control mode of the P25 pin
0	I/O port mode
1	INTP4 input mode

PMC26	Control mode of the P26 pin
0	I/O port mode
1	TCLR2 input mode/TO21 output mode[Note 2]

PMC27	Control mode of the P27 pin
0	I/O port mode
1	TO20 output mode

Notes 1. As external interrupt (INTP0, INTP1) will be generated even if pins P21 and P22 are used as timer outputs (TO04, TO05), mask the interrupt. (See **Fig. 15-1**.) When not using them as timer outputs, disable the timer outputs using the TOC1 register. The pins will become high-impedance. (See **Fig. 7-11**.)

2. When using pin P26 as a timer output (TO21), disable the external clear input using the TUM1 register. (See **Fig. 7-3**.)

Caution Bit 0 of the port 2 mode control register is always one.

Fig. 5-18. Format of the Port 3 Mode Control Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PMC3	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	FF43H	00H	R/W

PMC30	Control mode of the P30 pin
0	I/O port mode
1	TxD output mode

PMC31	Control mode of the P31 pin
0	I/O port mode
1	RxD input mode

PMC32	Control mode of the P32 pin
0	I/O port mode
1	SB0 I/O mode/SO00 output mode

PMC33	Control mode of the P33 pin
0	I/O port mode
1	SB1 I/O mode/SI00 input mode

PMC34	Control mode of the P34 pin
0	I/O port mode
1	$\overline{\text{SCK00}}$ I/O mode

PMC35	Control mode of the P35 pin
0	I/O port mode
1	TCLR1 input mode

PMC36	Control mode of the P36 pin
0	I/O port mode
1	TI1 input mode/TO11 output mode

PMC37	Control mode of the P37 pin
0	I/O port mode
1	TO10 output mode

Fig. 5-19. Format of the Port 8 Mode Control Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PMC8	PMC87	PMC86	PMC85	PMC84	PMC83	PMC82	PMC81	PMC80	FF48H	00H	R/W

PMC80	Control mode of the P80 pin
0	I/O port mode
1	TCLR0 input mode

PMC81	Control mode of the P81 pin
0	I/O port mode
1	TI0 input mode/TO03 output mode[Note]

PMC82	Control mode of the P82 pin
0	I/O port mode
1	TO00 output mode

PMC83	Control mode of the P83 pin
0	I/O port mode
1	TO01 output mode

PMC84	Control mode of the P84 pin
0	I/O port mode
1	TO02 output mode

PMC85	Control mode of the P85 pin
0	I/O port mode
1	TCLRUD input mode

PMC86	Control mode of the P86 pin
0	I/O port mode
1	PWM0 output mode

PMC87	Control mode of the P87 pin
0	I/O port mode
1	PWM1 output mode

Note When not using pin P81 as the timer output (TO03), disable the timer output using the TOC0 register. The pin will become high-impedance. (See **Fig. 7-10**.) When not using as the TI0 input, disable the external clock input using the TMC0 register. (See **Fig. 7-6**.)

Fig. 5-20. Format of the Port 10 Mode Control Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PMC10	PMC107	PMC106	PMC105	PMC104	PMC103	PMC102	PMC101	PMC100	FF4AH	00H	R/W

PMC100	Control mode of the P100 pin
0	I/O port mode
1	SO10 input mode

PMC101	Control mode of the P101 pin
0	I/O port mode
1	SI10 input mode

PMC102	Control mode of the P102 pin
0	I/O port mode
1	$\overline{\text{SCK10}}$ I/O mode

PMC103	Control mode of the P103 pin
0	I/O port mode
1	SO11 output mode

PMC104	Control mode of the P104 pin
0	I/O port mode
1	SI11 input mode

PMC105	Control mode of the P105 pin
0	I/O port mode
1	$\overline{\text{SCK11}}$ I/O mode

PMC106	Control mode of the P106 pin
0	I/O port mode
1	TIUD input mode

PMC107	Control mode of the P107 pin
0	I/O port mode
1	TCUD input mode

Fig. 5-21. Format of the Memory Expansion Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
MM	0	MM6	MM5	0	MM3	MM2	MM1	MM0	FFC4H	00H	R/W

MM2	MM1	MM0	Operation modes of the P40 to P47 and P50 to P57 pins		
0	0	0	Port mode	Single chip mode	P40 to 47: Input port P50 to 57: Port mode
0	0	1			P40 to 47: Output port P50 to 57: Port mode
0	1	0	Not to be set		
0	1	1	Expan-sion mode	256 bytes	P40 to 47: Expansion mode P50 to 57: Port mode
1	0	0		4 Kbytes	P40 to 47, P50 to P53: Expansion mode P54 to 57: Port mode
1	0	1		16 Kbytes	P40 to 47, P50 to P55: Expansion mode P56 to 57: Port mode
1	1	0	Not to be set		
1	1	1	Expan-sion mode	Full Expansion mode	P40 to 47, P50 to P57: Expansion mode

MM3	Internal RAM capacity
0	2 Kbytes
1	1 Kbyte

MM6	MM5	Internal ROM capacity
0	0	48 Kbytes
0	1	32 Kbytes
1	0	24 Kbytes
1	1	16 Kbytes

Cautions 1. Be sure to write 0 in bits 4 and 7. If 1 is written in these bits, operation will not be normal.

2. Invalid combinations are specified as "Not to be set" in Fig. 5-21. Never write these combinations.

5.2.4 Pull-up resistor specification

Each pin of ports 0 to 5 and ports 8 to 10 of the μPD78356 has an on-chip, pull-up resistor specifiable by software. (However, this does not apply to the P20 pin.)

(1) Ports 0 to 3, and ports 8 to 10

Whether to use an on-chip, pull-up resistor can be specified for each pin separately with the pull-up resistor option registers (PUOL, PUOH) and port mode register (PM).

When the PUO bit corresponding to a port is set to 1, the on-chip, pull-up resistor of a pin specified as an input port with the PM register is enabled.

(2) Port 4

The use of an on-chip, pull-up resistor is enabled only in the normal operation mode. The use of the on-chip, pull-up resistor of a pin is enabled when the PUO4 bit of the PUOL register is set to 1, and the pin is specified as an input port with the memory expansion mode register (MM).

(3) Port 5

The use of an on-chip, pull-up resistor is enabled only in the normal operation mode. The use of the on-chip, pull-up resistor of a pin is enabled when the PUO5 bit of the PUOL register is set to 1, and the pin is specified as an input port with the memory expansion mode register (MM) and port 5 mode register (PM5).

The use of the on-chip, pull-up resistor of a pin is also enabled when the pin is specified as a control pin with the port mode control register (PMC). So, when the on-chip, pull-up resistor of a control pin is not to be used, set the value of the corresponding bit of the PM register to 0 (output mode) or set the corresponding PUO bit to 0.

Caution In μPD78356 emulation using the IE-78350-R, the use of the on-chip, pull-up resistors of port 1, 4, 5, or 9 is disabled even if the PUO1, PUO4, PUO5, or PUO9 bit of the PUOL and PUOH registers is set to 1. When using a pull-up resistor, first set the corresponding bit to 1 to use software common to the IE-78350-R and μPD78356, then externally attach a pull-up resistor.

Fig. 5-22. Format of Pull-up Resistor Option Register L

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PUOL	0	0	PUO5	PUO4	PUO3	PUO2	PUO1	PUO0	FF44H	00H	R/W

PUOn	Pull-up resistor of port n (n = 0 to 5)
0	Not to be used on port n
1	To be used on port n

Fig. 5-23. Format of Pull-up Resistor Option Register H

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PUOH	0	0	0	0	0	PUO10	PUO9	PUO8	FF45H	00H	R/W

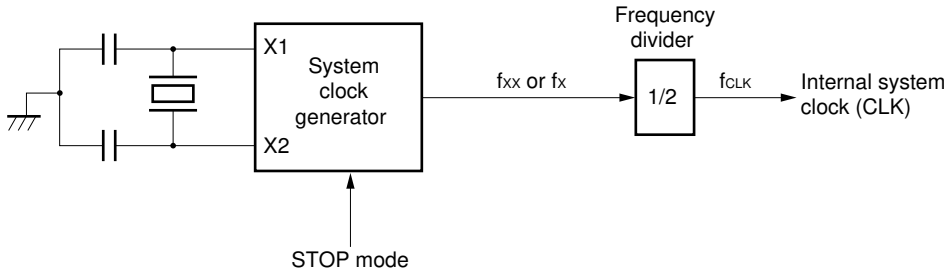
PUOn	Pull-up resistor of port n (n = 8 to 10)
0	Not to be used on port n
1	To be used on port n

[MEMO]

CHAPTER 6 CLOCK GENERATOR

The clock generator generates and controls an internal system clock (CLK) supplied to the CPU. The clock generator is configured as shown in Fig. 6-1.

Fig. 6-1. Block Diagram of the Clock Generator



Remarks 1. f_{XX} : Crystal oscillator frequency
2. f_X : External clock frequency
3. f_{CLK} : Internal system clock frequency

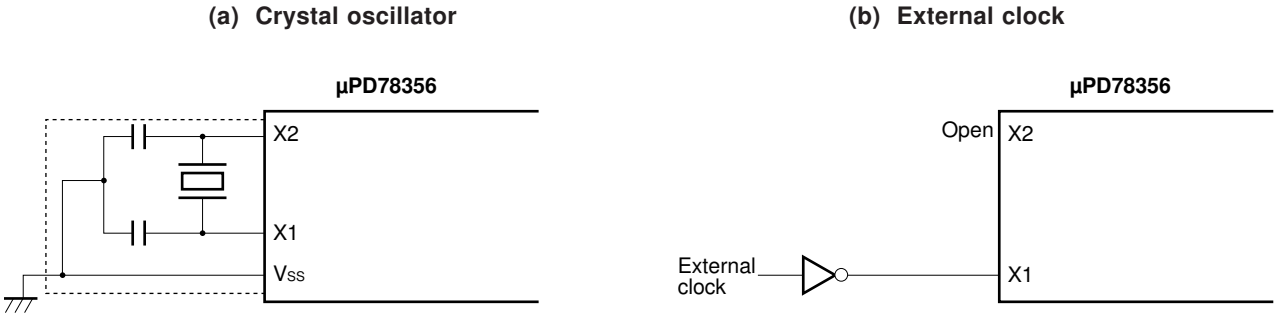
The system clock generator generates a clock signal with a crystal resonator connected to the X1 and X2 pins. The system clock generator stops oscillation when it is set to the standby mode (STOP mode). (See **CHAPTER 16**.)

An external clock can be applied. In this case, a clock signal is to be applied to the X1 pin. The X2 pin is left open.

The frequency divider divides system clock generator output (f_{XX} for the crystal oscillator or f_X for an external clock) by two to produce an internal system clock (f_{CLK}).

Caution When using an external clock, do not set the STP bit of the standby control register (STBC) to 1.

Fig. 6-2. External Circuitry of the System Clock Generator



Cautions 1. When using the system clock oscillator, run wires in areas enclosed with broken lines of Fig. 6-2 according to the following rules to avoid effects such as stray capacitance:

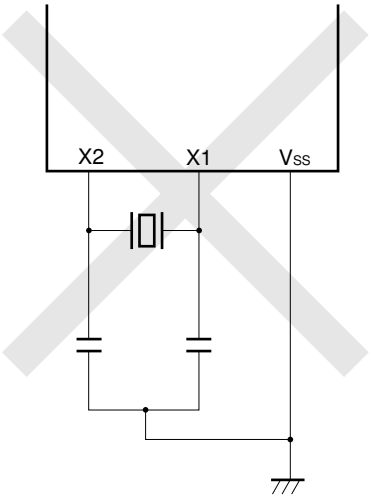
- **Minimize the wiring.**
- **Never cause the wires to cross other signal lines or run near a line carrying a large varying current.**
- **Cause the grounding point of the capacitor of the oscillator to have the same potential as V_{SS}. Never connect the capacitor to a ground pattern carrying a large current.**
- **Never extract a signal from the oscillator.**

2. When applying external clocks, do not connect a wire or a similar capacitive load to the X2 pin.

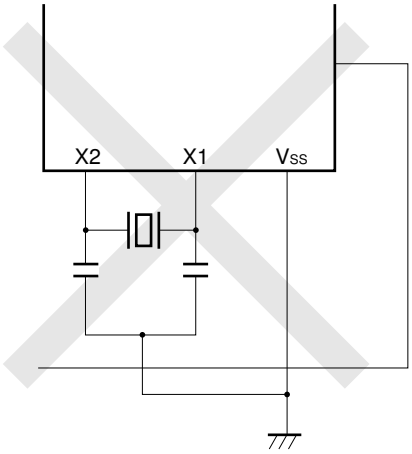
Fig. 6-3 shows examples of wrong resonator connection circuitry.

Fig. 6-3. Examples of Wrong Resonator Connection Circuitry

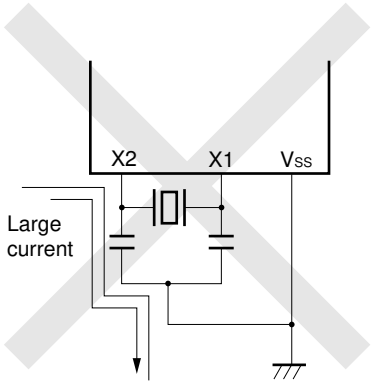
(a) Connection circuit wiring is too long.



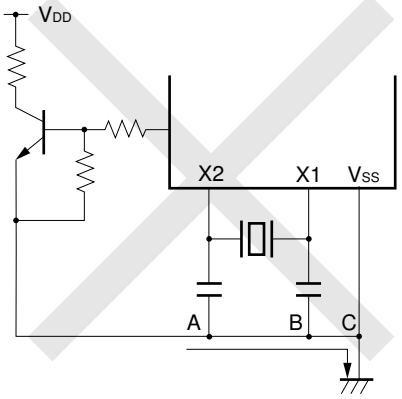
(b) There is another signal line crossing.



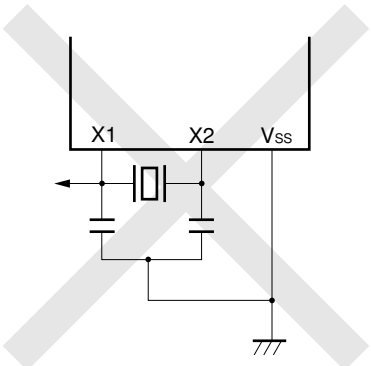
(c) A high varying current flows near a signal line.



(d) A current flows over the ground line of the generator circuit.
(The potentials of points A, B, and C change.)



(e) A signal is extracted.



[MEMO]

CHAPTER 7 REAL-TIME PULSE UNIT (RPU)

The RPU can output programmable pulses, measure pulse intervals and frequencies, generate trigger signals for the A/D converter, and function as a baud rate generator for the serial interface.

The RPU consists of four 16-bit timers (timers 0 to 3), one 10-bit timer (timer 4), and one 16-bit up/down counter.

The timers and up/down counter have the following features:

- Timer 0 ..... Generates trigger signals for the A/D converter. Up to four inputs are available for sampling and conversion timing in the A/D converter.
- Timers 0 to 2 ..... Allow various pulse outputs such as toggle output and set/reset output.
- Timers 0 and 3 and up/down counter ..... Generate overflow interrupts.
- Timers 1 and 2 ..... Control the output timing of a real-time output port.
- Timer 4 ..... Specifies the baud rate for the serial interface.
- Up/down counter ..... Software can toggle between incrementing and decrementing the counter. One of four operating modes can be selected to count external clock pulses. The counter is thus available for many applications.

Remark Toggle output and set/reset (SR) output

- Toggle output : A trigger signal inverts the state of this pulse output.
  The μPD78356 uses a match signal from a compare register as the trigger signal.
- SR output : A set trigger signal sets this pulse output to 1. A reset trigger signal resets this pulse output to 0.
  The μPD78356 uses match signals from different compare registers as the set and reset trigger signals. Set and reset timings can be separately specified.

7.1 Configuration

Fig. 7-1 shows the block diagram of the RPU. Table 7-1 lists hardware components of the timers and counter in the RPU.

Fig. 7-1. Block Diagram of the Real-Time Pulse Unit (RPU) (1/3)

Timer 0

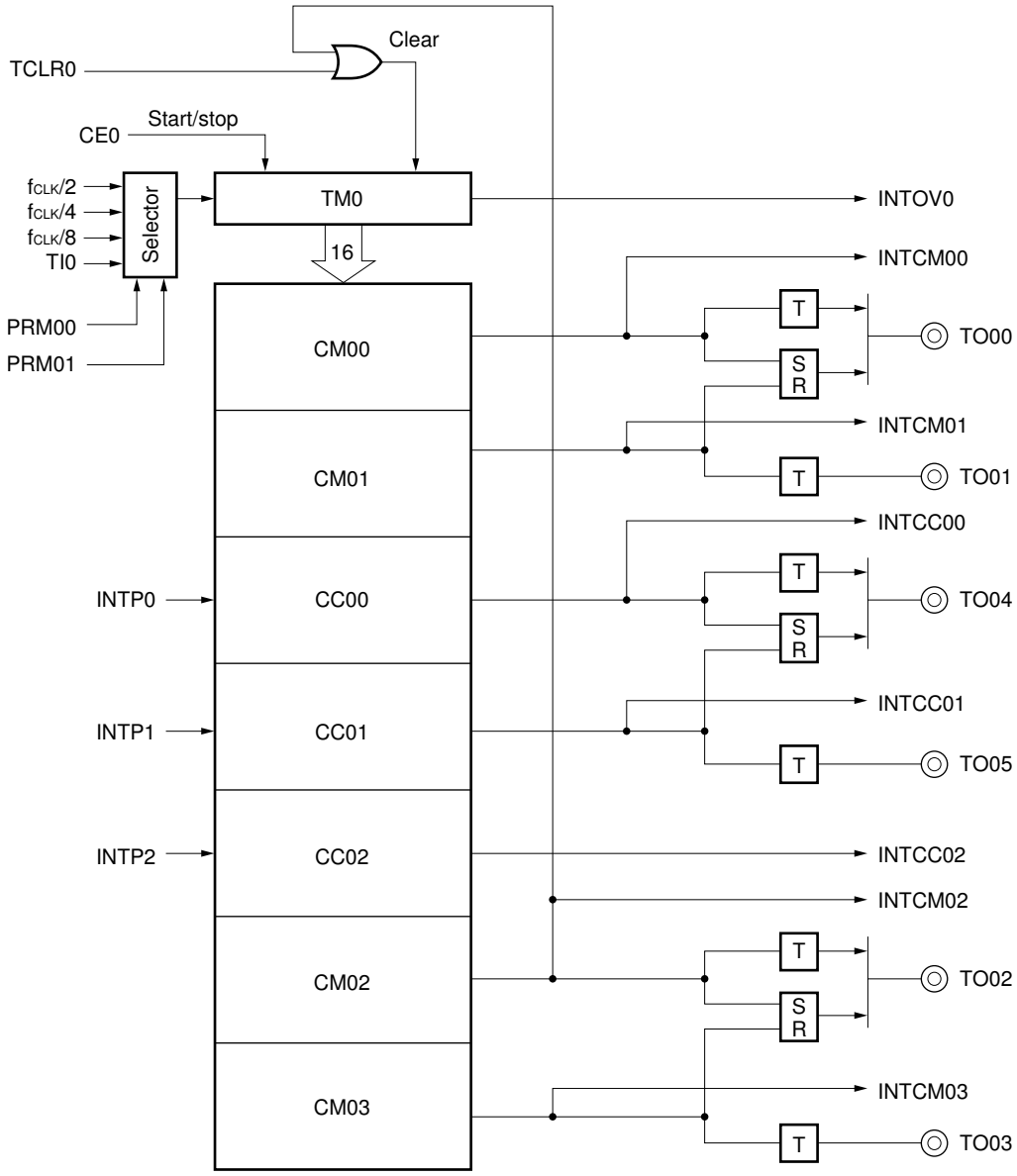
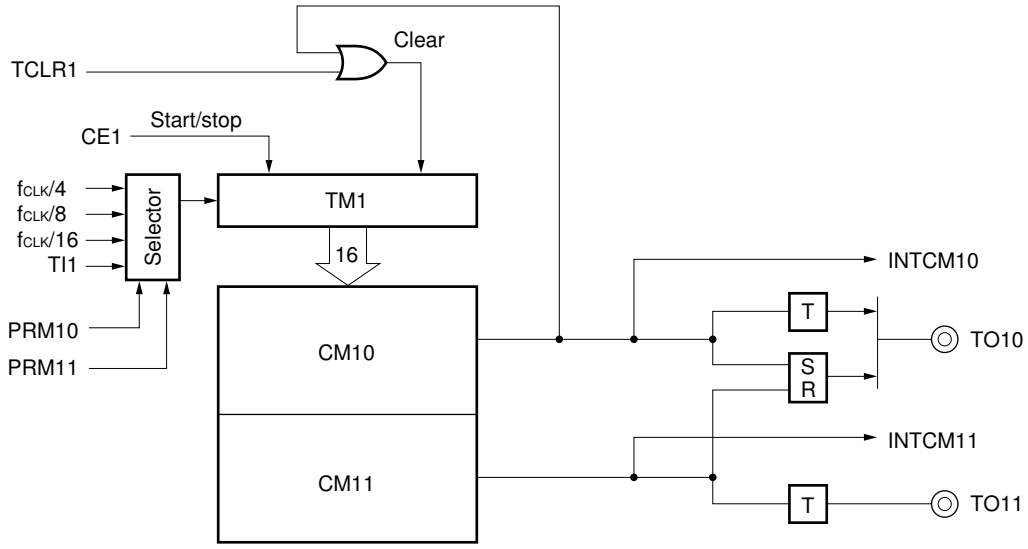


Fig. 7-1. Block Diagram of the Real-Time Pulse Unit (RPU) (2/3)

Timer 1



Timer 2

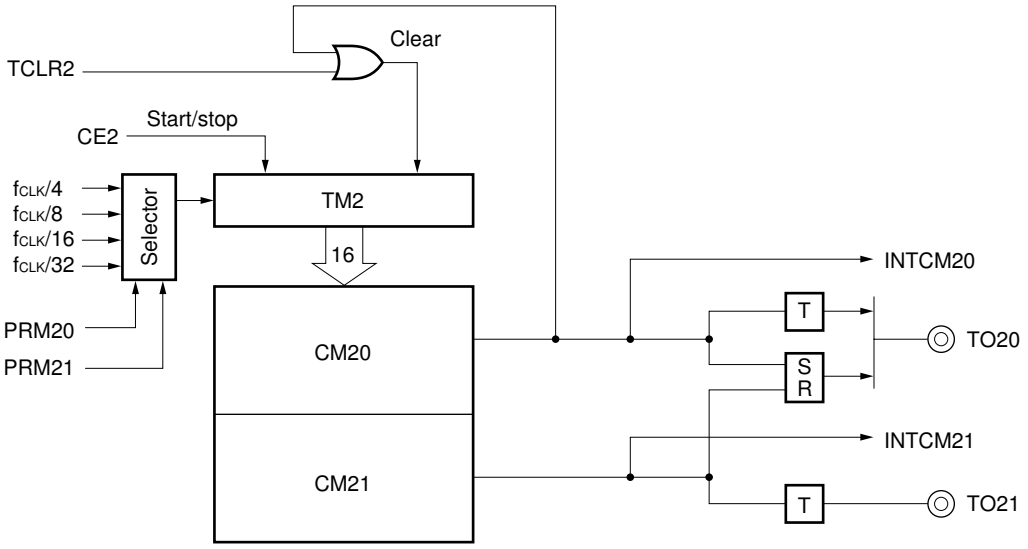
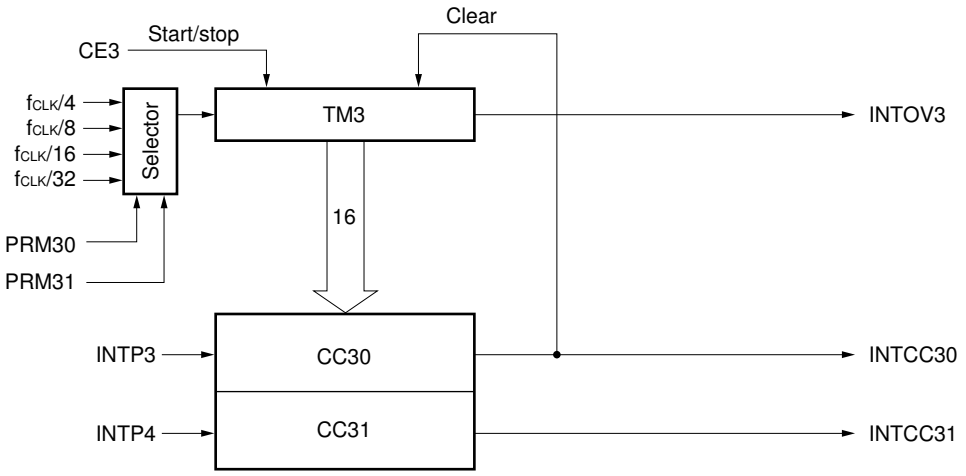
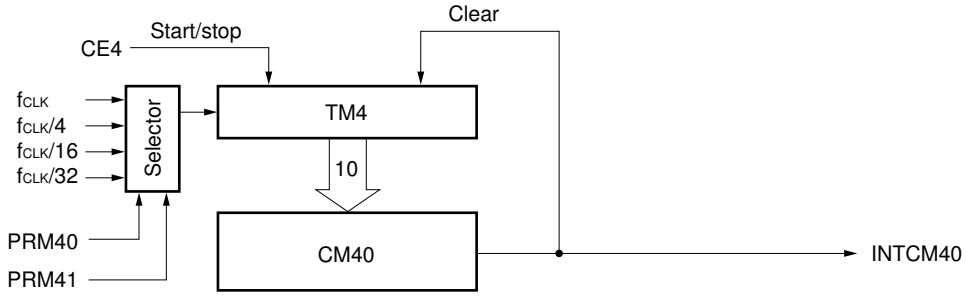


Fig. 7-1. Block Diagram of the Real-Time Pulse Unit (RPU) (3/3)

Timer 3



Timer 4



Up/down counter

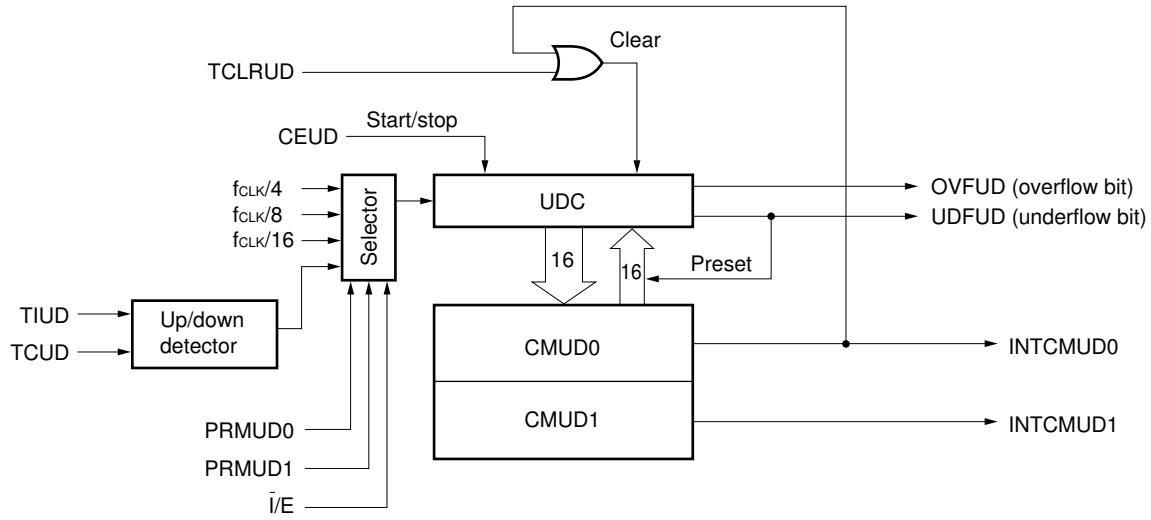


Table 7-1. Components of the Real-Time Pulse Unit (RPU)

	Timer	Count clock	Register	Match interrupt from compare register	Capture trigger	Number of timer outputs	Timer clear
Timer 0	16-bit timer (TM0)	f_{CLK}/2 f_{CLK}/4 f_{CLK}/8 Input to the TI0 pin	16-bit compare register (CM00) 16-bit compare register (CM01) 16-bit compare register (CM02) 16-bit compare register (CM03) 16-bit capture/compare register (CC00) 16-bit capture/compare register (CC01) 16-bit capture/compare register (CC02)	INTCM00Note 1 INTCM01Note 1 INTCM02 INTCM03 INTCC00Note 1 INTCC01Note 1 INTCC02	– – – – INTP0 INTP1 INTP2	6	TCLR0 INTCM02
Timer 1	16-bit timer (TM1)	f_{CLK}/4 f_{CLK}/8 f_{CLK}/16 Input to the TI1 pin	16-bit compare register (CM10) 16-bit compare register (CM11)	INTCM10 INTCM11	– –	2	TCLR1 INTCM10
Timer 2	16-bit timer (TM2)	f_{CLK}/4 f_{CLK}/8 f_{CLK}/16 f_{CLK}/32	16-bit compare register (CM20) 16-bit compare register (CM21)	INTCM20 INTCM21	– –	2	TCLR2 INTCM20
Timer 3	16-bit timer (TM3)	f_{CLK}/4 f_{CLK}/8 f_{CLK}/16 f_{CLK}/32	16-bit capture/compare register (CC30) 16-bit capture/compare register (CC31)	– –	INTP3 INTP4	–	INTCC30
Timer 4Note 2	10-bit timer (TM4)	f_{CLK} f_{CLK}/4 f_{CLK}/16 f_{CLK}/32	10-bit compare register (CM40)	INTCM40	–	–	INTCM40
Up/down counter	16-bit up/down counter (UDC)	f_{CLK}/4 f_{CLK}/8 f_{CLK}/16 Input to the TIUD pin	16-bit compare register (CMUD0) 16-bit compare register (CMUD1)	INTCMUD0 INTCMUD1	– –	–	TCLRUD INTCMUD0

Notes 1. Used also as a sampling trigger signal for the A/D converter.

2. Used also as a baud rate generator for the serial interface.

Remarks 1. f_{CLK}: Internal system clock

2. INTPn (n = 0 to 4): External interrupt

3. TM0, TM3, and the UDC have overflow interrupt functions.

The following sections describe the functions of the timers, compare registers, and capture/compare registers.

7.1.1 TM0

TM0 is a 16-bit timer/event counter. The timer can also be used as a one-shot timer.

TM0 counts internal clock pulses ($f_{CLK}/2$, $f_{CLK}/4$, or $f_{CLK}/8$) or external events applied to the TI0 pin. The PRM01 and PRM00 bits of timer control register 0 (TMC0) selects one of the three internal clock pulses or the external event.

The CE0 bit of timer control register 0 (TMC0) specifies whether the counting of TM0 is enabled or disabled.

When counting causes TM0 to overflow, an overflow interrupt (INTOV0) occurs, and the overflow bit (OVF0) is set to 1.

TM0 is cleared by an external clear input (TCLR0) or a match interrupt (INTCM02) from the compare register (CM02).

When the counter stops or a $\overline{\text{RESET}}$ signal is input, all the bits of TM0 are cleared to 0.

7.1.2 TM1

TM1 is a 16-bit timer/event counter.

TM1 counts internal clock pulses ($f_{CLK}/4$, $f_{CLK}/8$, or $f_{CLK}/16$) or external events applied to the TI1 pin. The PRM11 and PRM10 bits of timer control register 0 (TMC0) selects one of the three internal clock pulses or the external event.

The CE1 bit of timer control register 0 (TMC0) specifies whether the counting of TM1 is enabled or disabled.

When counting causes TM1 to overflow, the overflow bit (OVF1) is set to 1.

TM1 is cleared by an external clear input (TCLR1) or a match interrupt (INTCM10) from the compare register (CM10).

When the counter stops or a $\overline{\text{RESET}}$ signal is input, all the bits of TM1 are cleared to 0.

7.1.3 TM2

TM2 is a 16-bit interval timer.

TM2 counts internal clock pulses ($f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/16$, or $f_{CLK}/32$). The PRM21 and PRM20 bits of timer control register 1 (TMC1) specify which clock is counted.

The CE2 bit of timer control register 1 (TMC1) specifies whether the counting of TM2 is enabled or disabled.

When counting causes TM2 to overflow, the overflow bit (OVF2) is set to 1.

TM2 is cleared by an external clear input (TCLR2) or a match interrupt (INTCM20) from the compare register (CM20).

When the counter stops or a $\overline{\text{RESET}}$ signal is input, all the bits of TM2 are cleared to 0.

7.1.4 TM3

TM3 is a 16-bit interval timer.

TM3 counts internal clock pulses ($f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/16$, or $f_{CLK}/32$). The PRM31 and PRM30 bits of timer control register 1 (TMC1) specify which clock is counted.

The CE3 bit of timer control register 1 (TMC1) specifies whether the counting of TM3 is enabled or disabled.

When counting causes TM3 to overflow, an overflow interrupt (INTOV3) occurs, and the overflow bit (OVF3) is set to 1.

TM3 is cleared by a match interrupt (INTCC30) from the capture/compare register (CC30).

When the counter stops or a $\overline{\text{RESET}}$ signal is input, all the bits of TM3 are cleared to 0.

7.1.5 TM4

TM4 is a 10-bit interval timer. TM4 can also be used as a baud rate generator for the serial interface.

TM4 counts internal clock pulses (f_{CLK}, $f_{CLK}/4$, $f_{CLK}/16$, or $f_{CLK}/32$). The PRM41 and PRM40 bits of timer control register 2 (TMC2) specify which clock is counted.

The CE4 bit of timer control register 2 (TMC2) specifies whether the counting of TM3 is enabled or disabled.

TM4 is cleared by a match interrupt (INTCM40) from the compare register (CM40).

When the counter stops or a $\overline{\text{RESET}}$ signal is input, all the bits of TM4 are cleared to 0.

7.1.6 UDC

The UDC is a 16-bit up/down counter.

The UDC counts internal clock pulses ($f_{CLK}/4$, $f_{CLK}/8$, or $f_{CLK}/16$) or increments or decrements the number of external events input from the TIUD pin.

The PRMUD1, PRMUD0, and $\overline{\text{I}}$/E bits of the up/down counter control register (UDCC) specify whether one of the clocks is counted or the number of external events is incremented or decremented.

The CEUD bit of the UDCC specifies whether the counting of the UDC is enabled or disabled.

When counting causes the UDC to overflow or underflow, the overflow bit (OVFUD) or underflow bit (UDFUD) is set to 1.

The UDC is cleared by an external clear input (TCLRUD) or a match interrupt (INTCMUD0) from the compare register (CMUD0).

When the counter stops or a $\overline{\text{RESET}}$ signal is input, all the bits of the UDC are cleared to 0.

7.1.7 Compare registers

A compare register compares the contents of the corresponding timer with the data held in the compare register at all times, and generates an interrupt request signal when they match.

See Table 7-1 for detailed information about the configuration of the timers and compare registers and the correspondence between the compare registers and interrupt sources.

The contents of each compare register become undefined after $\overline{\text{RESET}}$ signal is input.

(1) 16-bit compare registers (CM00 to CM03, CM10, CM11, CM20, CM21, CMUD0, and CMUD1)

As shown in Fig. 7-1, 16-bit compare registers are connected to TM0 to TM2 and the UDC.

Compare registers connected to TM0 to TM2 have timer output functions. Each compare register inverts the corresponding timer output when a match interrupt occurs (toggle output operation). CM00 and CM01, CM02 and CM03, CM10 and CM11, and CM20 and CM21 can be used to set or reset output (set or reset output operation).

Table 7-2. Interrupt Request Signals from 16-Bit Compare Registers

Compare register	Interrupt request signal
CM00	INTCM00
CM01	INTCM01
CM02	INTCM02
CM03	INTCM03
CM10	INTCM10
CM11	INTCM11
CM20	INTCM20
CM21	INTCM21
CMUD0	INTCMUD0
CMUD1	INTCMUD1

(2) 10-bit compare register (CM40)

As shown in Fig. 7-1, a 10-bit compare register is connected to TM4. A match signal from CM40 can be used as a baud rate input for the serial interface.

Table 7-3. Interrupt Request Signal from the 10-Bit Compare Register

Compare register	Interrupt request signal
CM40	INTCM40

7.1.8 Capture/compare registers (CC00 to CC02, CC30, and CC31)

The capture/compare registers are 16-bit registers that function either as a capture or compare register. Control registers specify whether the capture/compare register functions as a capture or compare register.

(1) When a capture/compare register functions as a capture register

The capture/compare register takes in (captures) the contents of each timer when a capture trigger signal occurs. As a capture trigger, an external interrupt (INTPn: n = 0 to 4) can be used.

See Table 7-1 for the correspondence between the registers and capture triggers.

The occurrence of a capture trigger also means the occurrence of an interrupt. By using a capture register, the pulse width and period of externally applied pulses can be easily measured.

(2) When a capture/compare register functions as a compare register

The capture/compare register generates an interrupt based on a match between the timer and register values. When CC00 or CC01 is used as a compare register, the register either inverts the corresponding timer output when a match interrupt occurs (toggle output operation) or sets or resets output (set or reset output operation).

The interrupt source depends on whether the capture register function or compare register function is selected. When the capture register function is selected, the corresponding external interrupt capture trigger is selected. When the compare register function is selected, the corresponding match interrupt is selected.

The contents of each capture/compare register become undefined after $\overline{\text{RESET}}$ signal is input.

Caution When the compare register function is selected, interrupt generation based on the corresponding external interrupt (capture trigger signal) is impossible.

7.2 Control Registers

The following six types of control registers specify the operation of the real-time pulse unit (RPU):

- Timer unit mode registers (TUM0 to TUM3)
- Timer control registers (TMC0 to TMC2, and UDCC)
- Timer output control registers (TOC0 to TOC2)
- Timer overflow status register (TOVS)
- External interrupt mode registers (INTM0 and INTM1)
- Noise protection control register (NPC)

The following sections describe the control registers.

7.2.1 Timer unit mode registers (TUM0 to TUM3)

Timer unit mode registers (TUM0 to TUM3) are 8-bit registers. Table 7-4 lists their functions.

Table 7-4. Functions of Timer Unit Mode Registers

Register	Symbol	Function
Timer unit mode register 0	TUM0	Specifies the operating mode for capture/compare registers (CC00 to CC02, CC30, and CC31) and the output mode for timer output pins (TO00, TO02, and TO04).
Timer unit mode register 1	TUM1	Specifies whether to enable or disable clear signals (TCLR0, TCLR1, TCLR2, and TCLRUD) input to the RPU, and controls the operation of TM0 after a TM0 overflow and one-shot operation.
Timer unit mode register 2	TUM2	Specifies edges of external clock inputs (TI0, TI1, and TIUD).
Timer unit mode register 3	TUM3	Specifies edges of clear signals input to the RPU (TCLR0, TCLR1, TCLR2, and TCLRUD).

Fig. 7-2 to 7-5 show the formats of TUM0 to TUM3.

Data can be read from or written into TUM0 to TUM3 by a bit manipulation instructions or 8-bit manipulation instruction.

A $\overline{\text{RESET}}$ input signal sets TUM0 to TUM3 to 00H.

Fig. 7-2. Format of Timer Unit Mode Register 0

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
TUM0	TOM04	TOM02	TOM00	CMS31	CMS30	CMS02	CMS01	CMS00	FF70H	00H	R/W

CMSmn	Operating mode for a capture/compare register (CCmn)(m = 0, 3, n = 0 to 2)
0	Capture mode
1	Compare mode

TOM0n	Output mode for a timer output pin (TO0n) (n = 0, 2, 4)
0	Toggle output mode
1	Set/reset output mode

Caution Before changing the timer output mode (setting of the TOM00, TOM02, or TOM04 bit), disable the timer output for the corresponding timer output pin.

Fig. 7-3. Format of Timer Unit Mode Register 1

Symbol	7	6	5	4	3	2	<1>	0	Address	When reset	R/W
TUM1	0	ECLRUD	0	ECLR2	0	ECLR1	OST	ECLR0	FF71H	00H	R/W

ECLRn	A clear operation by a TCLRn pin input (n = 0 to 2, UD)
0	A clear operation by an external input is disabled.
1	A clear operation by an external input is enabled.

OST	Operation of TM0 in the free-running mode
0	TM0 continues counting after a TM0 overflow occurs. (Normal mode)
1	TM0 is cleared and stops counting after a TM0 overflow occurs. TM0 holds 0000H. Then it restarts being incremented when the CE0 bit is set to 1 or when TM0 is cleared (the specified edge of TCLR0 is input). (One-shot mode)

Caution Timer output (TO21) is impossible when a clear operation by a TCLR2 pin input is enabled.

Fig. 7-4. Format of Timer Unit Mode Register 2

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
TUM2	TESUD1	TESUD0	0	0	TES11	TES10	TES01	TES00	FF72H	00H	R/W

TESn1	TESn0	Active edge of the TIn pin (n = 0, 1, UD)
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

Caution When the operating mode for the UDC is specified as mode 4 (by the UDCC register), the specification of an edge for the TIUD pin (settings of the TESUD0 and TESUD1 bits) becomes ineffective.

Fig. 7-5. Format of Timer Unit Mode Register 3

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
TUM3	CESUD1	CESUD0	CES21	CES20	CES11	CES10	CES01	CES00	FF73H	00H	R/W

CESn1	CESn0	Active edge of the TCLRn pin (n = 0 to 2, UD)
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

7.2.2 Timer Control Registers (TMC0 to TMC2 and the UDCC)

Timer control registers are 8-bit registers. Table 7-5 lists their functions.

Table 7-5. Functions of Timer Control Registers

Register	Symbol	Function
Timer control register 0	TMC0	Controls counting of TM0 and TM1.
Timer control register 1	TMC1	Controls counting of TM2 and TM3.
Timer control register 2	TMC2	Controls counting of TM4.
Up/down counter control register	UDCC	Controls the counting of the UDC.

Fig. 7-6 to 7-9 show the formats of TMC0 to TMC2 and the UDCC.

Data can be read from or written into TMC0 to TMC2 and the UDCC by a bit manipulation instruction or 8-bit manipulation instruction.

A $\overline{\text{RESET}}$ input signal sets TMC0 to TMC2 and the UDCC to 00H.

Fig. 7-6. Format of Timer Control Register 0

Symbol	<7>	6	5	4	<3>	2	1	0	Address	When reset	R/W
TMC0	CE1	CLR1	PRM11	PRM10	CE0	CLR0	PRM01	PRM00	FF74H	00H	R/W

PRM01	PRM00	Count clock (Hz) to TM0
0	0	$f_{CLK}/2$
0	1	$f_{CLK}/4$
1	0	$f_{CLK}/8$
1	1	External clock (TI0 input)

CLR0	Operating mode for TM0
0	Free-running mode
1	Interval timer mode

CE0	Control of TM0 operation
0	TM0 is cleared and stops counting.
1	TM0 counts pulses or events.

PRM11	PRM10	Count clock (Hz) to TM1
0	0	$f_{CLK}/4$
0	1	$f_{CLK}/8$
1	0	$f_{CLK}/16$
1	1	External clock (TI1 input)

CLR1	Operating mode of TM1
0	Free-running mode
1	Interval timer mode

CE1	Control of TM1 operation
0	TM1 is cleared and stops counting.
1	TM1 counts pulses or events.

Caution Timer output (TO03) is impossible when the input of an external count clock to TM0 (TI0) is specified. Timer output (TO11) is impossible when the input of an external count clock to TM1 (TI1) is specified.

Remark f_{CLK}: Internal system clock

Fig. 7-7. Format of Timer Control Register 1

Symbol	<7>	6	5	4	<3>	2	1	0	Address	When reset	R/W
TMC1	CE3	CLR3	PRM31	PRM30	CE2	CLR2	PRM21	PRM20	FF75H	00H	R/W

PRM21	PRM20	Count clock (Hz) to TM2
0	0	$f_{CLK}/4$
0	1	$f_{CLK}/8$
1	0	$f_{CLK}/16$
1	1	$f_{CLK}/32$

CLR2	Operating mode for TM2
0	Free-running mode
1	Interval timer mode

CE2	Control of TM2 operation
0	TM2 is cleared and stops counting.
1	TM2 counts pulses.

PRM31	PRM30	Count clock (Hz) to TM3
0	0	$f_{CLK}/4$
0	1	$f_{CLK}/8$
1	0	$f_{CLK}/16$
1	1	$f_{CLK}/32$

CLR3	Operating mode for TM3
0	Free-running mode
1	Interval timer mode

CE3	Control of TM3 operation
0	TM3 is cleared and stops counting.
1	TM3 counts pulses.

Remark f_{CLK}: Internal system clock

Fig. 7-8. Format of Timer Control Register 2

Symbol	7	6	5	4	<3>	2	1	0	Address	When reset	R/W
TMC2	0	0	0	0	CE4	1	PRM41	PRM40	FF76H	04H	R/W

PRM41	PRM40	Count clock (Hz) to TM4
0	0	f_{CLK}
0	1	$f_{CLK}/4$
1	0	$f_{CLK}/16$
1	1	$f_{CLK}/32$

CE4	Control of TM4 operation
0	TM4 is cleared and stops counting.
1	TM4 counts pulses.

Remark f_{CLK}: Internal system clock

Fig. 7-9. Format of the Up/Down Counter Control Register

Symbol	<7>	6	5	4	3	2	1	0	Address	When reset	R/W
UDCC	CEUD	ENMD	PRMUD1	PRMUD0	$\overline{U}$/D	$\overline{I}$/E	0	0	FF77H	00H	R/W

$\overline{U}$/D	Whether the UDC is incremented or decremented
0	The UDC is incremented.
1	The UDC is decremented.

PRMUD1	PRMUD0	$\overline{I}$/E	Count clock	Whether the UDC incremented or decremented
0	0	0	f_{CLK}/4	Specified by setting of $\overline{U}$/D bit
0	1	0	f_{CLK}/8	
1	0	0	f_{CLK}/16	
1	1	0	Not to be set	
0	0	1	TIUD (external clock)	Mode 1
0	1	1		Mode 2
1	0	1		Mode 3
1	1	1		Mode 4

ENMD	Clearing or presetting the UDC by using the compare register (CMUD0)
0	Disabled. (Normal mode)
1	Enabled. If the contents of the CMUD0 register match the contents of the UDC, the UDC is cleared when the next count clock pulse is input. When the UDC underflows, the CMUD0 register is preset. (Up/down modulo mode)

CEUD	Control of the UDC operation
0	The UDC is cleared and stops counting.
1	The UDC counts pulses or events.

Caution When the operating mode for the UDC is specified as mode 4, the specification of an edge for the TIUD pin (by the TUM2 register) becomes ineffective.

Remarks 1. f_{CLK}: Internal system clock

2. Operating mode for the up/down counter (For details, see **7.3.6**.)

Mode 1	The UDC is decremented when the input to the TCUD pin is high. The UDC is incremented when the input to the TCUD pin is low.
Mode 2	The UDC is incremented when the specified edge of a TIUD input is detected. The UDC is decremented when the rising edge of a TCUD input is detected.
Mode 3	Whether the UDC is incremented or decremented is automatically determined according to the TCUD input state when the specified edge of the TIUD input is detected.
Mode 4	Whether the UDC is incremented or decremented is automatically determined when the rising and falling edges of the TIUD input and the rising and falling edges of the TCUD input are detected.

7.2.3 Timer output control registers (TOC0 to TOC2)

Timer output control registers (TOC0 to TOC2) are 8-bit registers. Table 7-6 lists their functions.

Table 7-6. Functions of Timer Output Control Registers

Register	Symbol	Function
Timer output control register 0	TOC0	Specifies the active level for timer output pins, TO00, TO01, TO02, and TO03, and whether the timer output is enabled or disabled for these pins.
Timer output control register 1	TOC1	Specifies the active level for timer output pins. TO04, TO05, TO10, and TO11, and whether the timer output is enabled or disabled for these pins.
Timer output control register 2	TOC2	Specifies the active level for timer output pins, TO20 and TO21, and whether the timer output is enabled or disabled for these pins. Also specifies the output mode for timer output pins, TO10 and TO20.

Fig. 7-10 to 7-12 show the formats of TOC0 to TOC2.

Data can be read from or written into TOC0 to TOC2 by a bit manipulation instruction or 8-bit manipulation instruction.

A $\overline{\text{RESET}}$ input signal sets TOC0 to TOC2 to 00H.

Table 7-7. Timer Outputs and Trigger Signals

Timer output pin	Toggle output mode	Set and reset output modes	
		Set trigger	Reset trigger
TO00	INTCM00	INTCM00	INTCM01
TO01	INTCM01	–	–
TO02	INTCM02	INTCM02	INTCM03
TO03	INTCM03	–	–
TO04	INTCC00	INTCC00	INTCC01
TO05	INTCC01	–	–
TO10	INTCM10	INTCM10	INTCM11
TO11	INTCM11	–	–
TO20	INTCM20	INTCM20	INTCM21
TO21	INTCM21	–	–

Fig. 7-10. Format of Timer Output Control Register 0

Symbol	<7>	6	<5>	4	<3>	2	<1>	0	Address	When reset	R/W
TOC0	ENTO03	ALV03	ENTO02	ALV02	ENTO01	ALV01	ENTO00	ALV00	FF78H	00H	R/W

ALV00	Active level for the TO00 pin
0	Low level
1	High level

ENTO00	Operation of the TO00 pin
0	The timer output is disabled. ($\overline{ALV00}$ is output.)
1	The timer output is enabled.

ALV01	Active level for the TO01 pin
0	Low level
1	High level

ENTO01	Operation of the TO01 pin
0	The timer output is disabled. ($\overline{ALV01}$ is output.)
1	The timer output is enabled.

ALV02	Active level for the TO02 pin
0	Low level
1	High level

ENTO02	Operation of the TO02 pin
0	The timer output is disabled. ($\overline{ALV02}$ is output.)
1	The timer output is enabled.

ALV03	Active level for the TO03 pin
0	Low level
1	High level

ENTO03	Operation of the TO03 pin
0	TI0 input mode[Note]
1	Timer output mode.

Note If the ENTO03 bit is reset to 0 when the PMC81 bit of the PMC8 register is 1, the state of TO03 pin becomes high-impedance.

Caution Before changing the active state (setting of the ALV00, ALV01, ALV02, or ALV03 bit), disable the timer output for the corresponding timer output pin.

Fig. 7-11. Format of Timer Output Control Register 1

Symbol	<7>	6	<5>	4	<3>	2	<1>	0	Address	When reset	R/W
TOC1	ENTO11	ALV11	ENTO10	ALV10	ENTO05	ALV05	ENTO04	ALV04	FF79H	00H	R/W

ALV04	Active level for the TO04 pin
0	Low level
1	High level

ENTO04	Operation of the TO04 pin
0	INTP0 input mode[Note 1]
1	Timer output mode

ALV05	Active level for the TO05 pin
0	Low level
1	High level

ENTO05	Operation of the TO05 pin
0	INTP1 input mode[Note 2]
1	Timer output mode

ALV10	Active level for the TO10 pin
0	Low level
1	High level

ENTO10	Operation of the TO10 pin
0	The timer output is disabled. ($\overline{ALV10}$ is output.)
1	The timer output is enabled.

ALV11	Active level for the TO11 pin
0	Low level
1	High level

ENTO11	Operation of the TO11 pin
0	The timer output is disabled. ($\overline{ALV11}$ is output.)
1	The timer output is enabled.

Notes 1. If the ENTO04 bit is reset to 0 when the PMC21 bit of the PMC2 register is 1, the state of TO04 pin becomes high-impedance.

2. If the ENTO05 bit is reset to 0 when the PMC22 bit of the PMC2 register is 1, the state of TO05 pin becomes high-impedance.

Cautions 1. Before changing the active level (setting of the ALV04, ALV05, ALV10, or ALV11 bit), disable the timer output for the corresponding timer output pin.

2. When the timer outputs for the TO04 and TO05 pins are disabled, external interrupt request inputs (INTP0 and INTP1) are assumed for these pins.

Fig. 7-12. Format of Timer Output Control Register 2

Symbol	7	6	5	4	<3>	2	<1>	0	Address	When reset	R/W
TOC2	TOM20	TOM10	0	0	ENTO21	ALV21	ENTO20	ALV20	FF7AH	00H	R/W

ALV20	Active level for the TO20 pin
0	Low level
1	High level

ENTO20	Operation of the TO20 pin
0	The timer output is disabled. ($\overline{\text{ALV20}}$ is output.)
1	The timer output is enabled

ALV21	Active level for the TO21 pin
0	Low level
1	High level

ENTO21	Operation of the TO21 pin
0	TCLR2 input mode[Note]
1	Timer output mode

TOM10	Output mode for the TO10 pin
0	Toggle output mode
1	Set/reset output mode

TOM20	Output mode for the TO20 pin
0	Toggle output mode
1	Set/reset output mode

Note If the ENTO21 bit is reset to 0 when the PMC26 bit of the PMC2 register is 1, the state of TO21 pin becomes high-impedance.

Caution Before changing the active level (setting of the ALV20 or ALV21 bit) and the timer output mode (setting of the TOM10 or TOM20 bit), disable the timer output for the corresponding timer output pin.

7.2.4 Timer overflow status register (TOVS)

The timer overflow status register is an 8-bit register. Table 7-8 lists its function.

Table 7-8. Function of the Timer Overflow Status Register

Register	Symbol	Function
Timer overflow status register	TOVS	Stores the overflow state of TM0 to TM3 and the overflow or underflow state of the UDC.

Fig. 7-13 shows the format of the TOVS.

Data can be read from or written into the TOVS by a bit manipulation instruction or 8-bit manipulation instruction.

A $\overline{\text{RESET}}$ input signal sets the TOVS to 00H.

Fig. 7-13. Format of the Timer Overflow Status Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
TOVS	0	0	UDFUD	OVFUD	OVF3	OVF2	OVF1	OVF0	FF7BH	00H	**Note**

R

OVF0	TM0 overflow flag
0	TM0 does not overflow.
1	TM0 overflows.

R/W

OVFn	TMn overflow flag (n = 1, 2)
0	TMn does not overflow.
1	TMn overflow.

R

OVF3	TM3 overflow flag
0	TM3 does not overflow.
1	TM3 overflows.

R/W

OVFUD	UDC overflow flag
0	The UDC does not overflow.
1	The UDC overflows.

R/W

UDFUD	UDC underflow flag
0	The UDC does not underflow.
1	The UDC underflows.

Note Each read/write bit functions as a read or write bit independently.

Caution Each flag of the TOVS is reset in a different way. Only software can reset the OVF1, OVF2, OVFUD, and UDFUD bits. Reset these flags after testing them.
The OVF0 and OVF3 bits are read-only bits. These flags are cleared to 0 when the interrupt request flags (OVIF0 and OVIF3) of the interrupt control registers (OVIC0 and OVIC3) are reset to 0 or TM0 and TM3 overflow interrupts (INTOV0 and INTOV3) are acknowledged.

7.2.5 External interrupt mode registers (INTM0 and INTM1)

External interrupt mode registers (INTM0 and INTM1) are 8-bit registers. Table 7-9 lists their functions.

Table 7-9. Functions of External Interrupt Mode Registers

Register	Symbol	Function
External interrupt mode register 0	INTM0	Specifies the edges for the NMI and INTP0 to INTP2 pins.
External interrupt mode register 1	INTM1	Specifies the edges for the INTP3 and INTP4 pins.

Fig. 7-14 and 7-15 show the formats of INTM0 and INTM1.

Data can be read from or written into INTM0 and INTM1 by a bit manipulation instruction or 8-bit manipulation instruction.

A $\overline{\text{RESET}}$ input signal sets INTM0 and INTM1 to 00H.

Fig. 7-14. Format of External Interrupt Mode Register 0

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
INTM0	ES21	ES20	ES11	ES10	ES01	ES00	0	ESN0	FF3CH	00H	R/W

ESN0	Active edge of the NMI pin
0	Falling edge
1	Rising edge

ESn1	ESn0	Active edge of the INTPn pin (n = 0 to 2)
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

Fig. 7-15. Format of External Interrupt Mode Register 1

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
INTM1	0	0	0	0	ES41	ES40	ES31	ES30	FF3DH	00H	R/W

ESn1	ESn0	Active edge of the INTPn pin (n = 3, 4)
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

7.2.6 Noise protection control register (NPC)

The noise protection control register (NPC) is an 8-bit register. Table 7-10 lists its function.

When the input time is less than the noise elimination time specified by the NPC, the input is assumed to be noise and ignored.

Sampling is performed every two clock pulses.

Table 7-10. Function of the Noise Protection Control Register

Register	Symbol	Function
Noise protection control register	NPC	Specifies the digital noise elimination time by sampling clock pulses on the pins for inputting external count clock pulses to TM0 to TM2 and the UDC (TI0, TI1, TIUD, and TCUD) and on the pins for inputting clear signals (TCLR0, TCLR1, TCLR2, and TCLRUD).

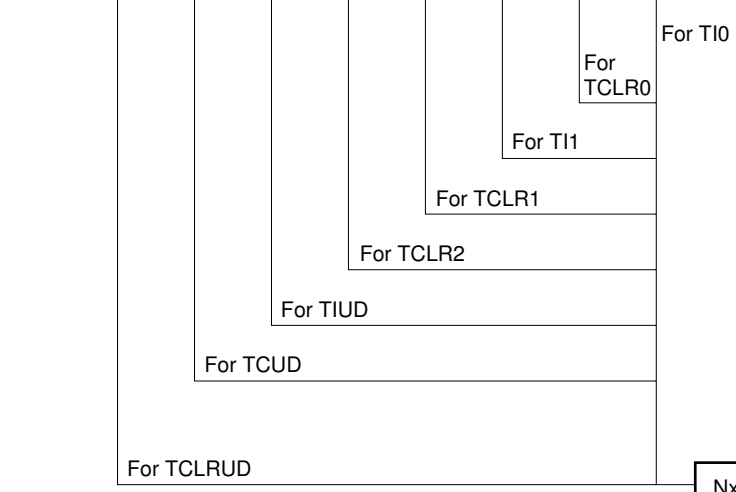
Fig. 7-16 shows the format of the NPC.

Data can be read from or written into NPC by a bit manipulation instruction or 8-bit manipulation instruction.

A $\overline{\text{RESET}}$ input signal sets the NPC to 00H.

Fig. 7-16. Format of the Noise Protection Control Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
NPC	NRUD	NCUD	NIUD	NR2	NR1	NI1	NR0	NI0	FF7CH	00H	R/W



Nxx	Noise elimination time
0	4 clock pulses (250 ns at f_{CLK} = 16 MHz)
1	16 clock pulses (1 μs at f_{CLK} = 16 MHz)

Caution Before changing the noise elimination time (the setting of the NI0, NR0, NI1, NR1, NR2, NIUD, NCUD, or NRUD bit), disable the counting of the corresponding timer.

Remark f_{CLK}: Internal system clock

7.3 Operation

7.3.1 Timer 0 (TM0)

(1) Basic operation of timer 0 (TM0)

TM0 functions as a 16-bit interval timer, free-running timer, or event counter for external signals. Timer control register 0 (TMC0) specifies the operating mode for TM0.

TM0 counts internal clock pulses ($f_{CLK}/2$, $f_{CLK}/4$, or $f_{CLK}/8$) or external clock pulses (TI0 pin input), according to the specification by the PRM01 and PRM00 bits of the TMC0 register. The specification of an external clock as a count clock enables TM0 to function as an event counter.

If TM0 overflows while counting clock pulses, an overflow interrupt (INTOV0) occurs, and the OVF0 bit of the timer overflow status register (TOVS) is set to 1.

When the preset value of the compare register (CM02) matches the value counted by TM0, the contents of TM0, which functions as an interval timer, are cleared, and a match interrupt (INTCM02) occurs. Regardless of how TM0 operates, the external clear input (TCLR0) can clear the contents of TM0.

When TM0 functions as a free-running timer, it can be used as a one-shot timer. In this case, timer unit mode register 1 (TUM1) specifies the operation of TM0.

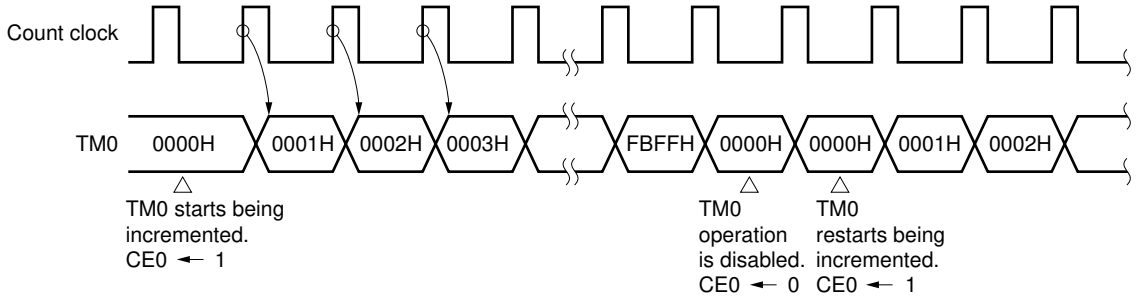
The CE0 bit of the TMC0 register specifies whether counting is enabled (see **Fig. 7-17**).

When software sets the CE0 bit to 1, TM0 starts counting. Even if software sets the CE0 bit to 1 with the bit already set to 1, TM0 continues counting.

When the CE0 bit is reset to 0, TM0 is cleared and stops counting.

A $\overline{\text{RESET}}$ input signal clears all the bits of TM0, stopping the counting of TM0.

Fig. 7-17. Basic Operation of Timer 0 (TM0)



Caution If bit 1 (PMC81) of the port 8 mode control register (PMC8) is rewritten from 0 to 1 or 1 to 0, when the count clock to timer 0 (TM0) is specified as the external clock (TI0 input) to perform count operations, TM0 may count up. Therefore, rewrite the PMC81 bit after stopping the TM0 count operations.

(2) Comparing values

The value counted by TM0 is compared with the value held in a compare register.

When the value counted by TM0 matches the value set in the compare register, a match signal is sent to an output control circuit. When the match signal is generated, the states of timer output pins (TO00 to TO05) are changed, and an interrupt request signal is generated. A match interrupt (INTCM02) from the compare register (CM02) clears TM0.

When the CLR0 bit of timer control register 0 (TMC0) is reset to 0, TM0 is set to the free-running timer mode. When the bit is set to 1, TM0 is set to the interval timer mode.

Table 7-11. Interrupt Request Signals from 16-Bit Compare Registers for Timer 0 (TM0)

Compare register	Interrupt request signal
CM00	INTCM00
CM01	INTCM01
CM02	INTCM02
CM03	INTCM03
CC00Note	INTCC00
CC01Note	INTCC01
CC02Note	INTCC02

Note These registers are also used as capture registers. Timer unit mode register 0 (TUM0) specifies whether they are used as capture or compare registers.

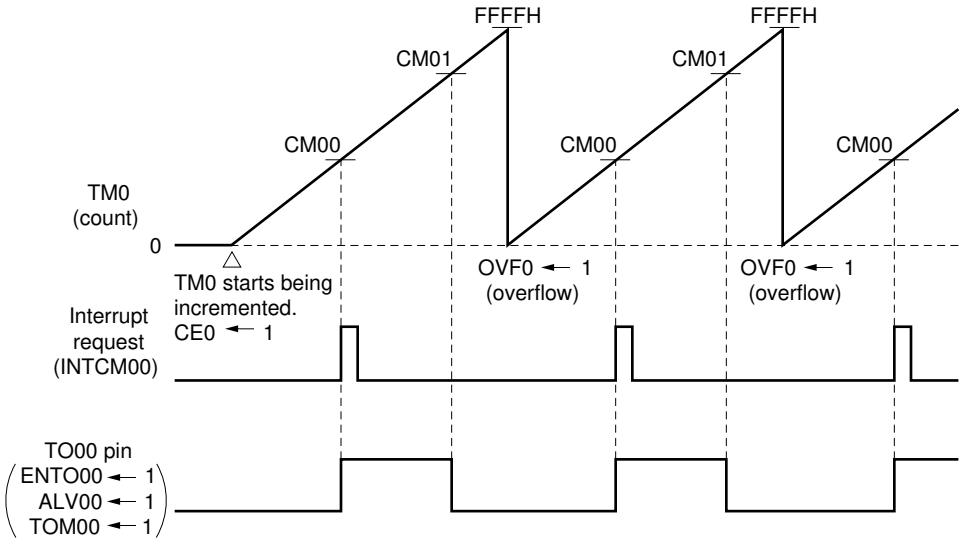
TM0 has six timer output pins (TO00 to TO05). Table 7-12 lists the operating modes for each pin. For details, see **7.4**.

Table 7-12. Operating Modes for Each Timer Output Pin of Timer 0 (TM0)

Timer output pin	Output operation mode		Operating mode specified by
TO00	Toggle	Set or reset	TOM00 bit of the TUM0 register
TO01	Toggle	–	–
TO02	Toggle	Set or reset	TOM02 bit of the TUM0 register
TO03	Toggle	–	–
TO04	Toggle	Set or reset	TOM03 bit of the TUM0 register
TO05	Toggle	–	–

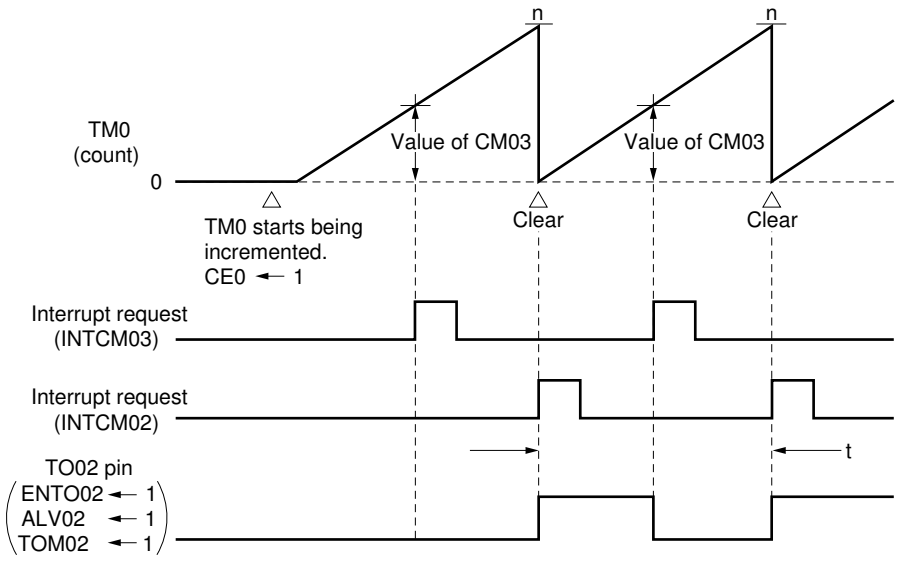
When an overflow is detected in the free-running timer mode, the contents of TM0 are cleared.

Fig. 7-18. Example of Comparison for TM0 in the Free-Running Timer Mode



When TM0 is set to the interval timer mode, the value counted by TM0 is compared with the value of the compare register (CM02). When the values match, TM0 is cleared to 0 at the next count clock. This enables TM0 to function as an interval timer having a count clock period equal to the value set in the CM02 register.

Fig. 7-19. Example of Comparison for TM0 in the Interval Timer Mode



Remark n : Value of the CM02 register

t : Interval time = (n + 1) x count clock period

Caution When 00H is set in the CM02 register in the interval timer mode, TM0 performs interval operation at each count clock input.

The interval time at the first match is different from that at the second or subsequent matches.

- **Time until the first match ..... Value of the CM02 register x count clock period**
- **Time until the second match or subsequent matches ... (Value of the CM02 register + 1) x count clock period**

(3) Capturing a value

The value counted by TM0 is sent to and held in (captured by) a capture register in synchronization with an external trigger signal. The specified edges detected at the external interrupt request input pin are used as external triggers (capture triggers). In synchronization with a capture trigger, the value counted by TM0 is captured in the capture register. The capture register holds the value until the next capture trigger is generated.

Table 7-13. Capture Trigger Signals for 16-Bit Capture Registers for Timer 0 (TM0)

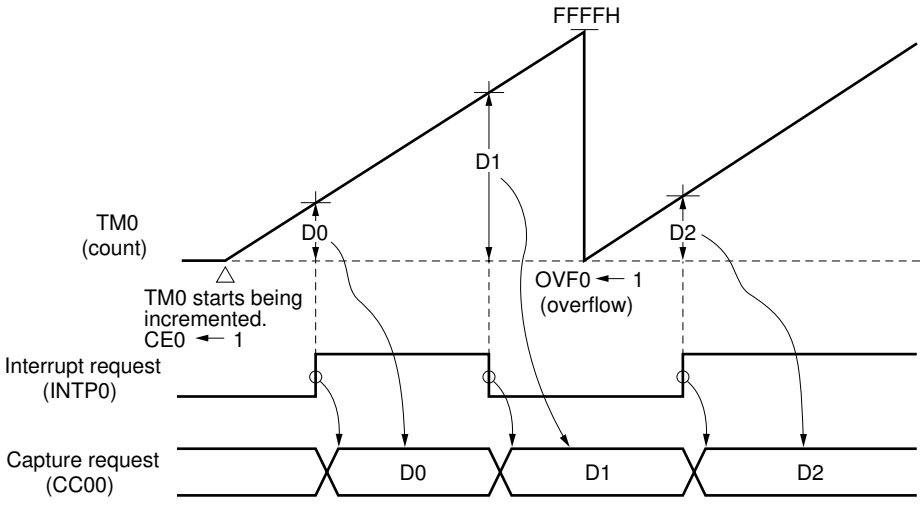
Capture register	Capture trigger
CC00[Note]	INTP0
CC01[Note]	INTP1
CC02[Note]	INTP2

Note These registers are also used as compare registers. Timer unit mode register 0 (TUM0) specifies whether they are used as capture or compare registers.

External interrupt mode register 0 (INTM0) specifies whether the rising edge, falling edge, or both are used as a capture trigger.

When both the rising edge and falling edge are used as a capture trigger, the width of an external input pulse can be measured. When either of the two edges is used as a capture trigger, the input pulse period can be measured.

Fig. 7-20. Example of Capture for Timer 0 (TM0)



Remark Dn (n = 0, 1, 2, ...): Value counted by TM0

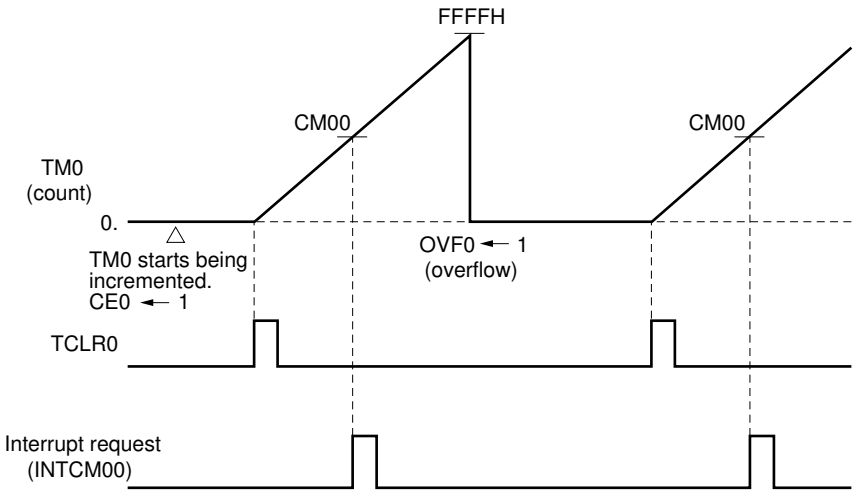
(4) One-shot operation

When the OST bit of timer unit mode register 1 (TUM1) is set to 1 in the free-running timer mode, timer 0 (TM0) is put in the one-shot operation mode.

When counting causes TM0 timer to overflow in the one-shot operation mode, an overflow interrupt (INTOV0) occurs and TM0 stops counting. In this case, TM0 is cleared to 0000H and the CE0 bit of timer control register 0 (TMC0) is set to 1.

When the CE0 bit is set to 1 again or an external clear signal (TCLR0) is input in the standby state, TM0 restarts counting. When counting causes TM0 to overflow, TM0 stops counting and enters the standby state again.

Fig. 7-21. Example of One-Shot Operation (When Timer 0 (TM0) Is Started by TCLR0 Input)



7.3.2 Timer 1 (TM1)

(1) Basic operation of timer 1 (TM1)

TM1 functions as a 16-bit interval timer, free-running timer, or event counter for external signals. Timer control register 0 (TMC0) specifies the operating mode for TM1.

TM1 counts internal clock pulses ($f_{CLK}/4$, $f_{CLK}/8$, or $f_{CLK}/16$) or external clock pulses (TI1 pin input), according to the specification by the PRM11 and PRM10 bits of the TMC0 register. The specification of an external clock as a count clock enables TM1 to function as an event counter.

If TM1 overflows while counting clock pulses, the OVF1 bit of the timer overflow status register (TOVS) is set to 1.

When the preset value of the compare register (CM10) matches the value counted by TM1, the contents of TM1, which functions as an interval timer, are cleared, and a match interrupt (INTCM10) occurs. Regardless of how TM1 operates, the external clear input (TCLR1) can clear the contents of TM1.

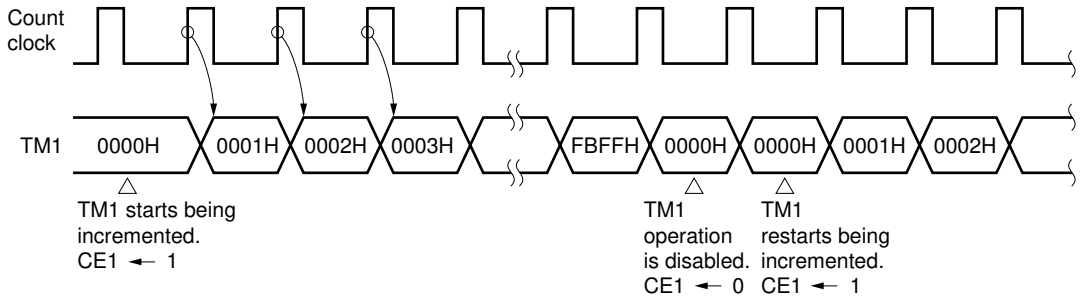
The CE1 bit of the TMC0 register specifies whether counting is enabled (see **Fig. 7-22**).

When software sets the CE1 bit to 1, TM1 starts counting. Even if software sets the CE1 bit to 1 with the bit already set to 1, TM1 continues counting.

When the CE1 bit is reset to 0, TM1 is cleared and stops counting.

A $\overline{\text{RESET}}$ input signal clears all the bits of TM1, stopping the counting of TM1.

Fig. 7-22. Basic Operation of Timer 1 (TM1)



Caution If bit 6 (PMC36) of the port 3 mode control register (PMC3) is rewritten from 0 to 1 or 1 to 0, when the count clock to timer 1 (TM1) is specified as the external clock (TI1 input) to perform count operations, TM1 may count up. Therefore, rewrite the PMC36 bit after stopping the TM1 count operations.

(2) Comparing values

The value counted by TM1 is compared with the value held in a compare register.

When the value counted by TM1 matches the value set in the compare register, a match signal is sent to an output control circuit. When the match signal is generated, the states of timer output pins (TO10 and TO11) are changed, and an interrupt request signal is generated. A match interrupt (INTCM10) from the compare register (CM10) clears TM1.

When the CLR1 bit of timer control register 0 (TMC0) is reset to 0, TM1 is set to the free-running timer mode. When CLR1 is set to 1, TM1 is set to the interval timer mode.

Table 7-14. Interrupt Request Signals from 16-Bit Compare Registers for Timer 1 (TM1)

Compare register	Interrupt request signal
CM10	INTCM10
CM11	INTCM11

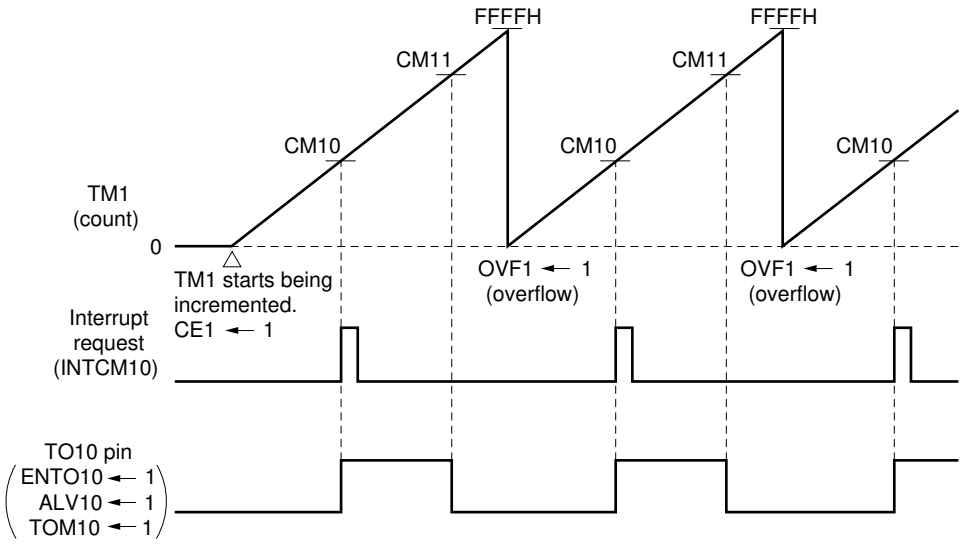
TM1 has two timer output pins (TO10 and TO11). Table 7-15 lists the operating modes for each pin. For details, see **7.4**.

Table 7-15. Operating Modes for Each Timer Output Pin of Timer 1 (TM1)

Timer output pin	Output operation mode		Operating mode specified by
TO10	Toggle	Set or reset	TOM10 bit of the TOC2 register
TO11	Toggle	–	–

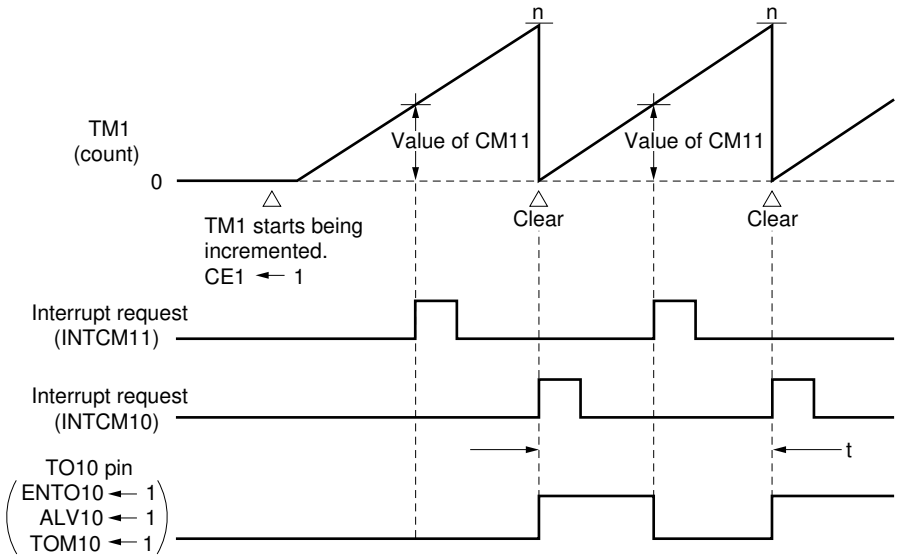
When an overflow is detected in the free-running timer mode, the contents of TM1 are cleared.

Fig. 7-23. Example of Comparison for Timer 1 (TM1) in the Free-Running Timer Mode



When TM1 is set to the interval timer mode, the value counted by TM1 is compared with the value of the compare register (CM10). When the values match, TM1 is cleared to 0 at the next count clock. This enables TM1 to function as an interval timer having a count clock period equal to the value set in the CM10 register.

Fig. 7-24. Example of Comparison for Timer 1 (TM1) in the Interval Timer Mode



Remark n : Value of the CM10 register

t : Interval time = (n + 1) x count clock period

Caution When 00H is set in the CM10 register in the interval timer mode, TM1 performs interval operation at each count clock input.
The interval time at the first match is different from that at the second or a subsequent match.

- **Time until the first match ..... Value of the CM10 register x count clock period**

- **Time until the second match or a subsequent match ... (Value of the CM10 register + 1) x count clock period**

7.3.3 Timer 2 (TM2)

(1) Basic operation of timer 2 (TM2)

TM2 functions as a 16-bit interval timer or free-running timer. Timer control register 1 (TMC1) specifies the operating mode for TM2.

TM2 counts internal clock pulses ($f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/16$, or $f_{CLK}/32$), according to the specification by the PRM21 and PRM20 bits of the TMC1 register.

If TM2 overflows while counting clock pulses, the OVF2 bit of the timer overflow status register (TOVS) is set to 1.

When the preset value of the compare register (CM20) matches the value counted by TM2, the contents of TM2, which functions as an interval timer, are cleared, and a match interrupt (INTCM20) occurs. Regardless of how TM2 operates, the external clear input (TCLR2) can clear the contents of TM2.

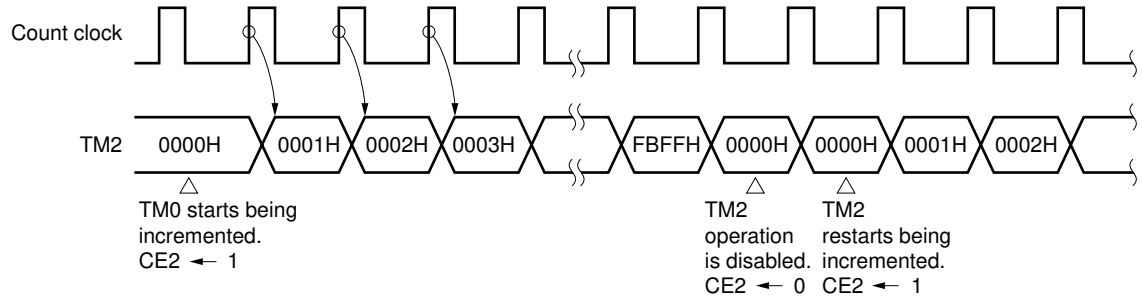
The CE2 bit of the TMC1 register specifies whether counting is enabled (see **Fig. 7-25**).

When software sets the CE2 bit to 1, TM2 starts counting. Even if software sets the CE2 bit to 1 with the bit already set to 1, TM2 continues counting.

When the CE2 bit is reset to 0, TM2 is cleared and stops counting.

A $\overline{\text{RESET}}$ input signal clears all the bits of TM2, stopping the counting of TM2.

Fig. 7-25. Basic Operation of Timer 2 (TM2)



Caution If bit 6 (PMC26) of the port 2 mode control register (PMC2) is rewritten from 0 to 1 or 1 to 0 when clear operations is enabled using pin TCLR2, TM2 may be cleared. Therefore, rewrite the PMC26 bit after stopping the TM2 count operation.

(2) Comparing values

The value counted by TM2 is compared with the value held in a compare register.

When the value counted by TM2 matches the value set in the compare register, a match signal is sent to an output control circuit. When the match signal is generated, the states of timer output pins (TO20 and TO21) are changed, and an interrupt request signal is generated. A match interrupt (INTCM20) from the compare register (CM20) clears TM2.

When the CLR2 bit of timer control register 1 (TMC1) is reset to 0, TM2 is set to the free-running timer mode. When CLR2 is set to 1, TM2 is set to the interval timer mode.

Table 7-16. Interrupt Request Signals from 16-Bit Compare Registers for Timer 2 (TM2)

Compare register	Interrupt request signal
CM20	INTCM20
CM21	INTCM21

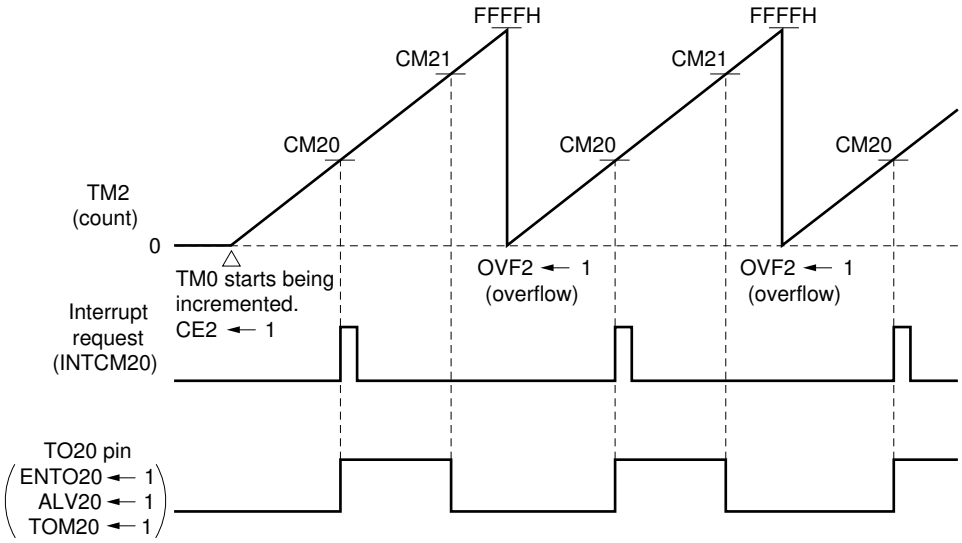
TM2 has two timer output pins (TO20 and TO21). Table 7-17 lists the operating modes for each pin. For details, see **7.4**.

Table 7-17. Operating Modes for Each Timer Output Pin of Timer 2 (TM2)

Timer output pin	Output operation mode		Operating mode specified by
TO20	Toggle	Set or reset	TOM20 bit of the TOC2 register
TO21	Toggle	–	–

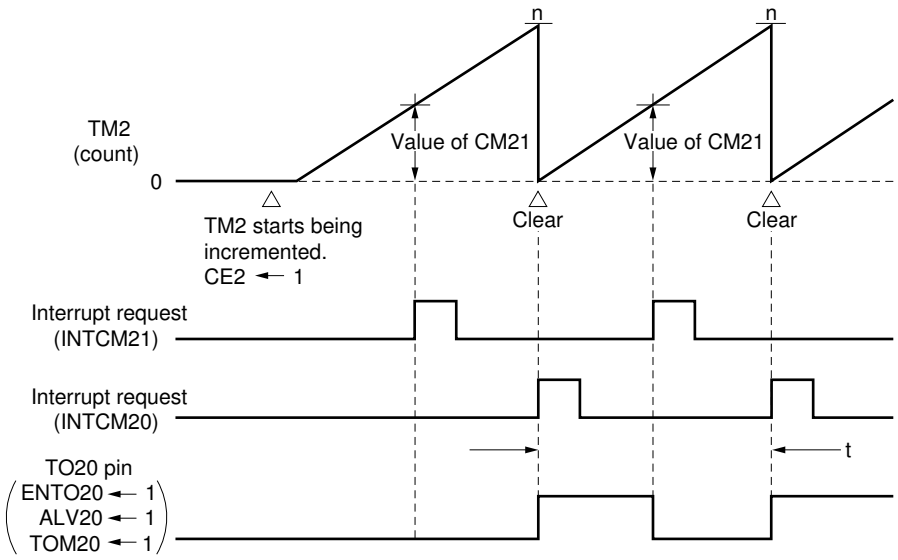
When an overflow is detected in the free-running timer mode, the contents of TM2 are cleared.

Fig. 7-26. Example of Comparison for Timer 2 (TM2) in the Free-Running Timer Mode



When TM2 is set to the interval timer mode, the value counted by TM2 is compared with the value of the compare register (CM20). When the values match, TM2 is cleared to 0 at the next count clock. This enables TM2 to function as an interval timer having a count clock period equal to the value set in the CM20 register.

Fig. 7-27. Example of Comparison for Timer 2 (TM2) in the Interval Timer Mode



Remark n : Value of the CM20 register

t : Interval time = (n + 1) x count clock period

Caution When 00H is set in the CM20 register in the interval timer mode, TM2 performs interval operation at each count clock input.

The interval time at the first match is different from that at the second or a subsequent match.

- **Time until the first match ..... Value of the CM20 register x count clock period**
- **Time until the second match or a subsequent match ... (Value of the CM20 register + 1) x count clock period**

7.3.4 Timer 3 (TM3)

(1) Basic operation of timer 3 (TM3)

TM3 functions as a 16-bit interval timer or free-running timer. Timer control register 1 (TMC1) specifies the operating mode for TM3.

TM3 counts internal clock pulses ($f_{CLK}/4$, $f_{CLK}/8$, $f_{CLK}/16$ or $f_{CLK}/32$), according to the specification by the PRM31 and PRM30 bits of the TMC1 register.

If TM3 overflows while counting clock pulses, an overflow interrupt (INTOV3) occurs, and the OVF3 bit of the timer overflow status register (TOVS) is set to 1.

When the preset value of the compare register (CC30) matches the value counted by TM3, the contents of TM3, which functions as an interval timer, are cleared, and a match interrupt (INTCM30) occurs.

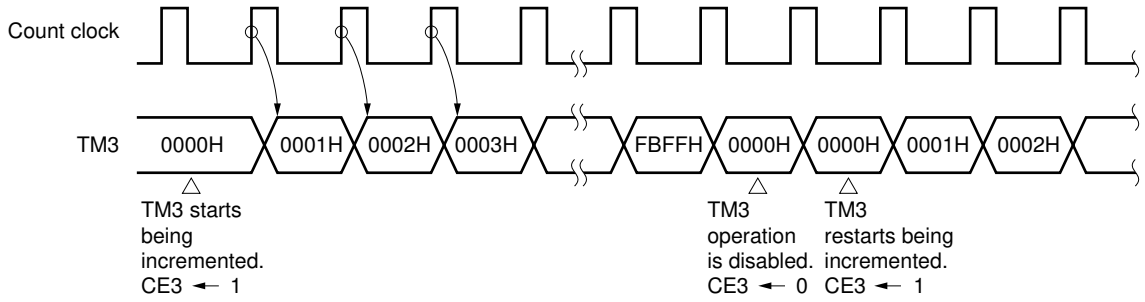
The CE3 bit of the TMC1 register specifies whether counting is enabled (see **Fig. 7-28**).

When software sets the CE3 bit to 1, TM3 starts counting. Even if software sets the CE3 bit to 1 with the bit already set to 1, TM3 continues counting.

When the CE3 bit is reset to 0, TM3 is cleared and stops counting.

A $\overline{\text{RESET}}$ input signal clears all the bits of TM3, stopping the counting of TM3.

Fig. 7-28. Basic Operation of Timer 3 (TM3)



(2) Comparing values

The value counted by TM3 is compared with the value held in a compare register.

When the value counted by TM3 matches the value set in the compare register, an interrupt request signal is generated. A match interrupt (INTCC30) from the compare register (CC30) clears TM3.

When the CLR3 bit of timer control register 1 (TMC1) is reset to 0, TM3 is set to the free-running timer mode. When CLR3 is set to 1, TM3 is set to the interval timer mode.

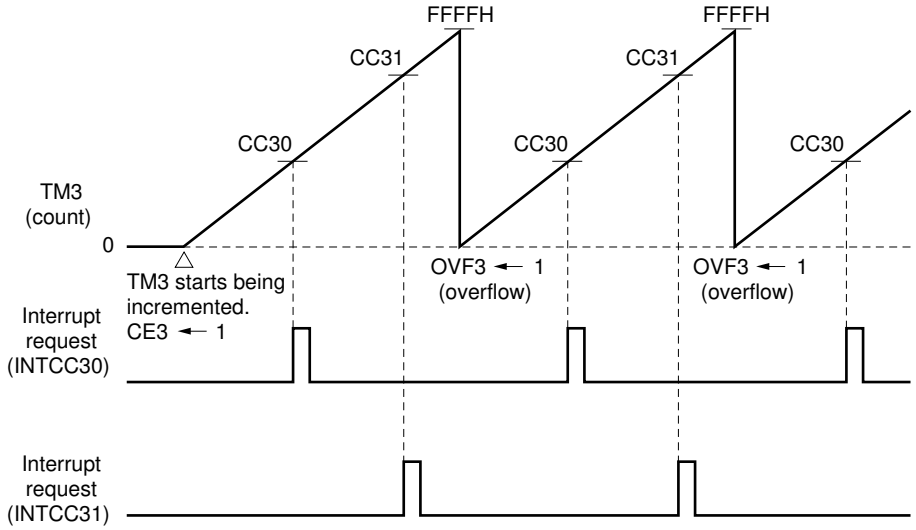
Table 7-18. Interrupt Request Signals from 16-Bit Compare Registers for Timer 3 (TM3)

Compare register	Interrupt request signal
CC30[Note]	INTCC30
CC31[Note]	INTCC31

Note These registers are also used as capture registers. Timer unit mode register 0 (TUM0) specifies whether they are used as capture or compare registers.

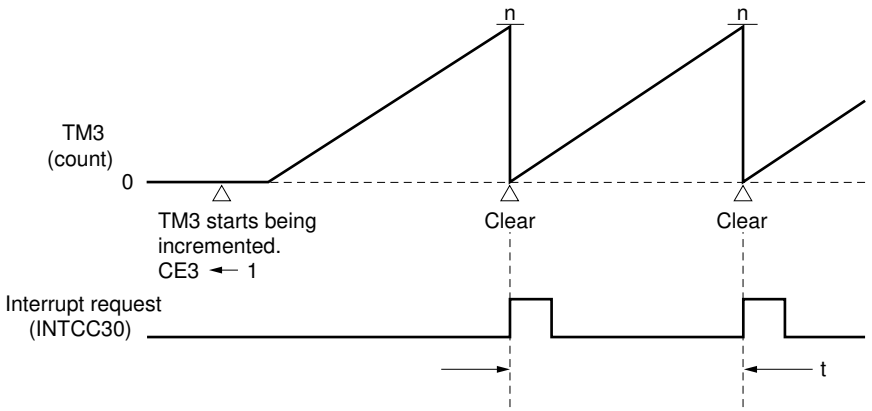
When an overflow is detected, the contents of TM3 in the free-running timer mode are cleared to 0.

Fig. 7-29. Example of Comparison for Timer 3 (TM3) in the Free-Running Timer Mode



When TM3 is set to the interval timer mode, the value counted by TM3 is compared with the value of the compare register (CC30). When the values match, TM3 is cleared to 0 at the next count clock. This enables TM3 to function as an interval timer having a count clock period equal to the value set in the CC30 register.

Fig. 7-30. Example of Comparison for Timer 3 (TM3) in the Interval Timer Mode



Remark n : Value of the CC30 register

t : Interval time = (n + 1) x count clock period

Caution When 00H is set in the CC30 register in the interval timer mode, TM3 performs interval operation at each count clock input.

The interval time at the first match is different from that at the second or a subsequent match.

- **Time until the first match ..... Value of the CC30 register x count clock period**
- **Time until the second match or a subsequent match .... (Value of the CC30 register + 1) x count clock period**

(3) Capturing a value

The value counted by TM3 is sent to and held in (captured by) a capture register in synchronization with an external trigger signal. The specified edges detected at the external interrupt request input pin are used as external triggers (capture triggers). In synchronization with a capture trigger, the value counted by TM3 is captured in the capture register. The capture register holds the value until the next capture trigger is generated.

Table 7-19. Capture Trigger Signals for 16-Bit Capture Registers for Timer 3 (TM3)

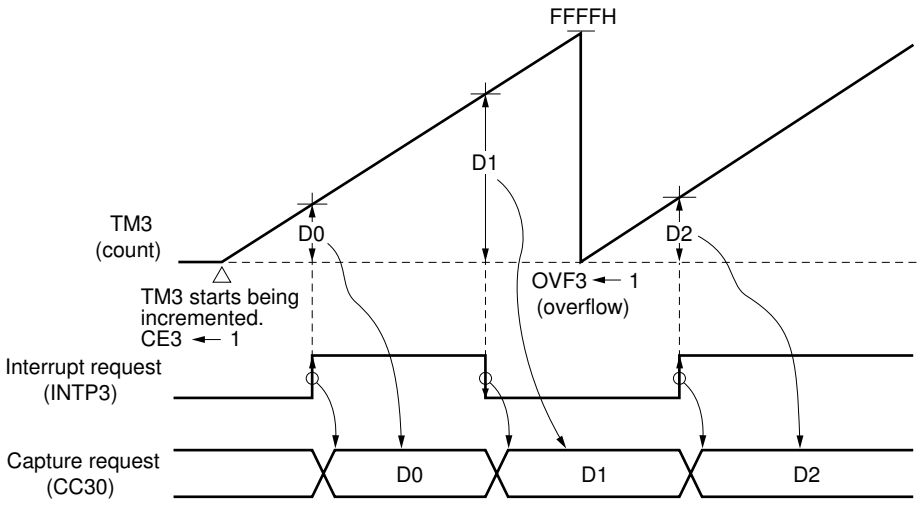
Capture register	Capture trigger signal
CC30[Note]	INTP3
CC31[Note]	INTP4

Note These registers are also used as compare registers. Timer unit mode register 0 (TUM0) specifies whether they are used as capture or compare registers.

External interrupt mode register 1 (INTM1) specifies whether the rising edge, falling edge, or both are used as a capture trigger.

When both the rising edge and falling edge are used as a capture trigger, the width of an external input pulse can be measured. When either of the two edges is used as a capture trigger, the input pulse period can be measured.

Fig. 7-31. Example of Capture for Timer 3 (TM3)



Remark Dn (n = 0, 1, 2, ...): Value counted by TM3

7.3.5 Timer 4 (TM4)

(1) Basic operation of timer 4 (TM4)

TM4 functions as a 10-bit interval timer.

TM4 counts internal clock pulses (f_{CLK}, $f_{CLK}/4$, $f_{CLK}/16$ or $f_{CLK}/32$), according to the specification by the PRM41 and PRM40 bits of the TMC2 register.

When the preset value of the compare register (CM40) matches the value counted by TM4, the contents of TM4 are cleared, and a match interrupt (INTCM40) occurs.

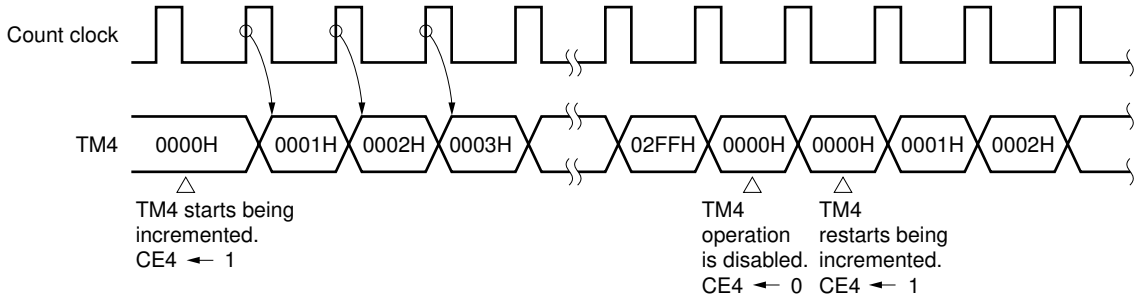
The CE4 bit of the TMC2 register determines whether counting is enabled (see **Fig. 7-32**).

When software sets the CE4 bit to 1, TM4 starts counting. Even if software sets the CE4 bit to 1 with the bit already set to 1, TM4 continues counting.

When the CE4 bit is reset to 0, TM4 is cleared and stops counting.

A $\overline{\text{RESET}}$ input signal clears all the bits of TM4, stopping the counting of TM4.

Fig. 7-32. Basic Operation of Timer 4 (TM4)



(2) Comparing values

The value counted by TM4 is compared with the value held in a compare register.

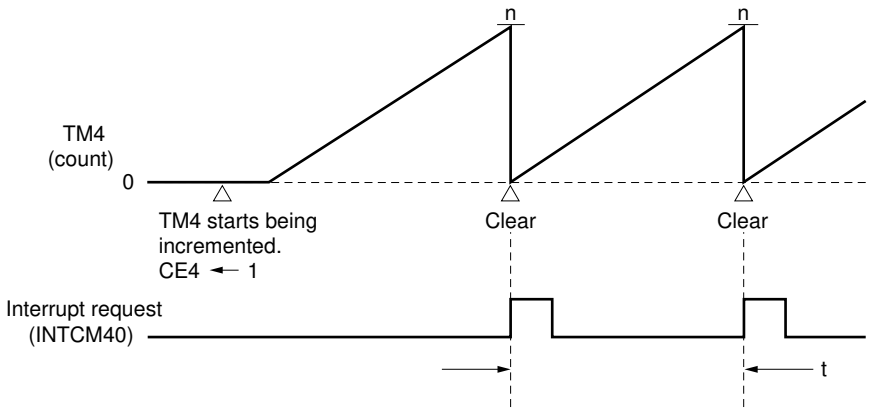
When the value counted by TM4 matches the value set in the compare register, an interrupt request signal is generated. A match interrupt (INTCM40) from the compare register (CM40) clears TM4.

Table 7-20. Interrupt Request Signals from 10-Bit Compare Registers for Timer 4 (TM4)

Compare register	Interrupt request signal
CM40	INTCM40

The value counted by TM4 is always compared with the value held in the CM40 compare register. When the values match, TM4 is cleared to 0 at the next count clock. This enables TM4 to function as an interval timer having a count clock period equal to the value set in the CM40 register.

Fig. 7-33. Example of Comparison for Timer 4 (TM4) in the Interval Timer Mode



Remark n : Value of the CM40 register

t : Interval time = (n + 1) x count clock period

Caution When 00H is set in the CM40 register in the interval timer mode, TM4 performs interval operation at each count clock input.

The interval time at the first match is different from that at the second or a subsequent match.

- **Time until the first match ..... Value of the CM40 register x count clock period**
- **Time until the second match or a subsequent match .... (Value of the CM40 register + 1) x count clock period**

7.3.6 Up/down counter (UDC)

(1) Basic operation of the up/down counter (UDC)

The UDC functions as a 16-bit interval timer, free-running timer, or event counter for external signals. The up/down counter control register (UDCC) specifies the operating mode for the UDC.

The UDC counts internal clock pulses ($f_{CLK}/4$, $f_{CLK}/8$, or $f_{CLK}/16$) or increments or decrements the number of external events input from the TIUD pin. The PRMUD1 and PRMUD0 bits of the UDCC specify whether one of the clocks is counted or the number of external events is incremented or decremented.

The $\overline{U}$/D bit of the UDCC specifies whether the UDC is incremented or decremented. However, specifying a count clock as an external clock enables external pins (the TIUD and TCUD pin inputs) to specify whether the UDC is incremented or decremented.

When counting causes the UDC to overflow, the OVFUD bit of the timer overflow status register (TOVS) is set to 1.

When counting causes the UDC to underflow, the UDFUD bit of the TOVS is set to 1.

When the UDC functions as an interval timer and the preset value of the compare register (CMUD0) matches the value counted by the UDC, the UDC is cleared, and a match interrupt (INTCMUD0) occurs. Regardless of how the UDC operates, the external clear input (TCLRUD) can clear the contents of the UDC.

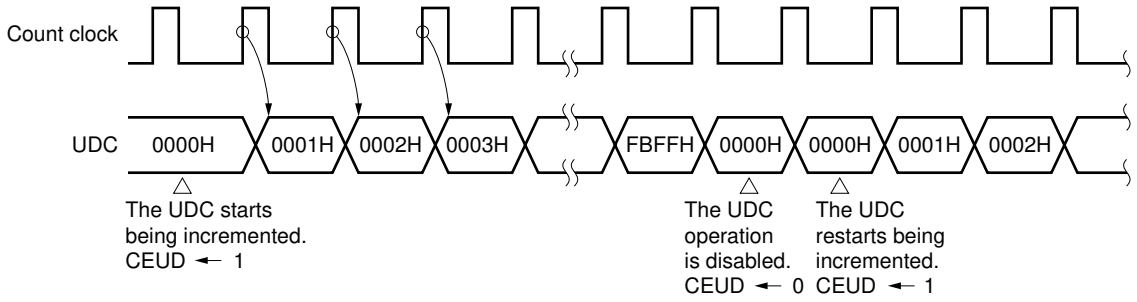
The CEUD bit of the UDCC specifies whether counting is enabled (see **Fig. 7-34**).

When software sets the CEUD bit to 1, the UDC starts counting. Even if software sets the CEUD bit to 1 with the bit already set to 1, the UDC continues counting.

When the CEUD bit is reset to 0, the UDC is cleared and stops counting.

A $\overline{RESET}$ input signal clears all the bits of the UDC, stopping the counting of the UDC.

Fig. 7-34. Basic Operation of the Up/Down Counter (UDC)



(2) Comparing values

The value counted by the UDC is compared with the value held in a compare register.

When the value counted by the UDC matches the value set in the compare register, an interrupt request signal is generated. A match interrupt (INTCMUD0) from the compare register (CMUD0) clears the UDC.

When the ENMD bit of the up/down counter control (UDCC) register is reset to 0, the UDC counts in the normal mode. When ENMD is set to 1, the UDC counts in the up/down modulo mode.

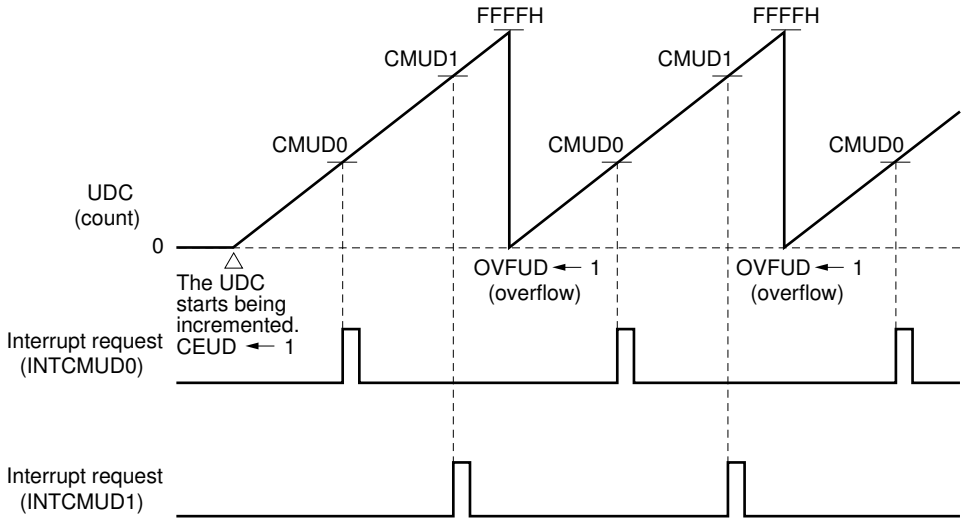
Table 7-21. Interrupt Request Signals from 16-Bit Compare Registers for the Up/Down Counter (UDC)

Compare register	Interrupt request signal
CMUD0	INTCMUD0
CMUD1	INTCMUD1

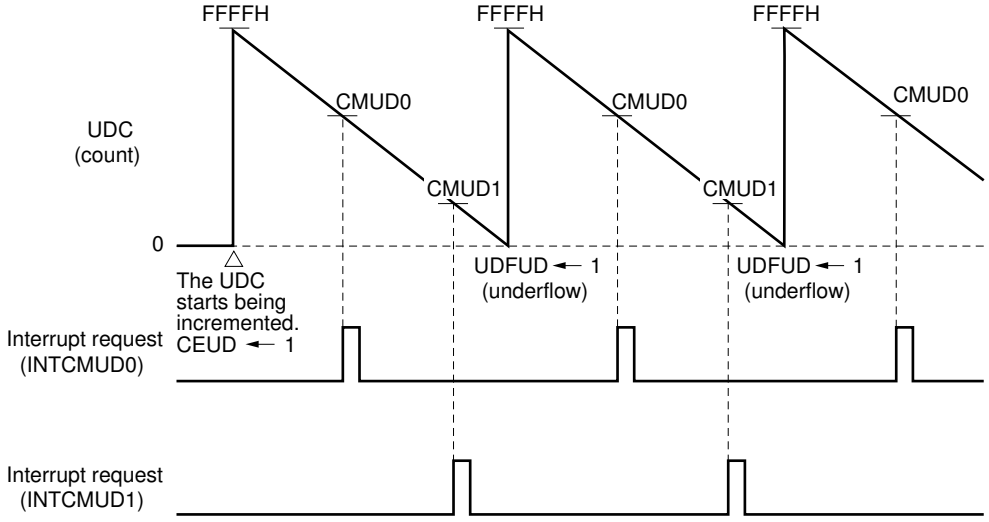
When the UDC overflows or underflows while being incremented or decremented in the normal mode, it is cleared to 0. This enables the UDC to function as a free running timer.

Fig. 7-35. Example of Comparison for the Up/Down Counter (UDC) in the Normal Mode

(1) When the UDC is incremented



(2) When the UDC is decremented

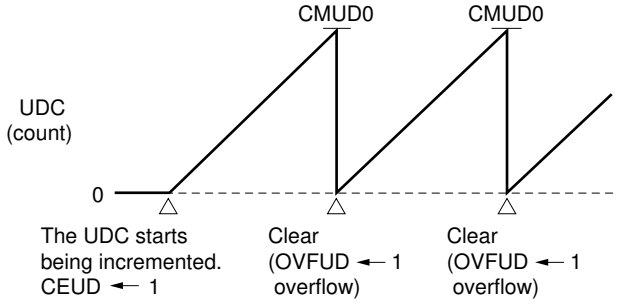


When the UDC is incremented in the up/down modulo mode, the value counted by the UDC is compared with the value held in the CMUD0 compare register. When the values match, the UDC is cleared to 0 at the next count clock. When the UDC underflows while being decremented, the UDC is preset to the value held in the CMUD0 register.

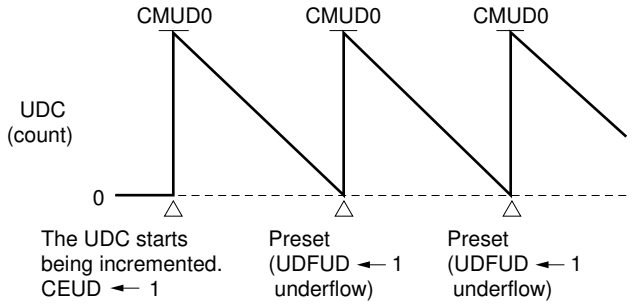
This enables the UDC to function as an interval timer or up/down counter having a count clock period equal to the value set in the CMUD0 register.

Fig. 7-36. Example of Comparison for the Up/Down Counter (UDC) in the Up/Down Modulo Mode

(1) When the UDC is incremented



(2) When the UDC is decremented



(3) Incrementing or decrementing the UDC

Specifying a count clock as an external clock for the UDC enables the external count clock input (the TIUD pin) and counting control signal input (the TCUD pin) to specify whether the UDC is incremented or decremented. The PRMUD1 and PRMUD0 bits of the up/down counter control register (UDCC) specify one of modes 1 to 4 as the operating mode for the UDC.

Table 7-22. Operating Mode for the Up/Down Counter (UDC)

UDCC register			Operating mode	Operation of the UDC
PRMUD1	PRMUD0	$\bar{I}$/E		
0	0	1	Mode 1	The UDC is decremented when the input on the TCUD pin is high. The UDC is incremented when the input on the TCUD pin is low.
0	1	1	Mode 2	The UDC is incremented when the specified edge of a TIUD input is detected. The UDC is decremented when the rising edge of a TCUD input is detected.
1	0	1	Mode 3	Whether the UDC is incremented or decremented is automatically determined according to the TCUD input level when the specified edge of the TIUD input is detected.
1	1	1	Mode 4	Whether the UDC is incremented or decremented is automatically determined when both the rising edge and falling edge of the TIUD input and both the rising edge and falling edge of the TCUD input are detected.

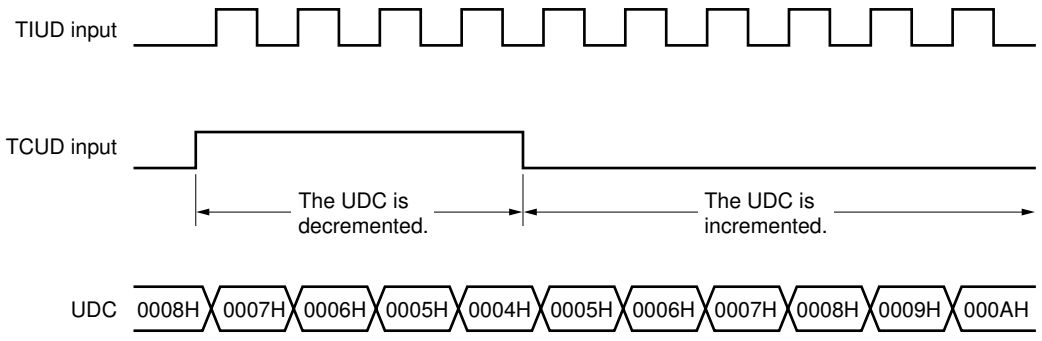
Caution Specifying mode 4 as the operating mode for the UDC invalidates the edges for specified by timer unit mode register 2 (TUM2) for the TIUD pin.

(a) Mode 1 (PRMUD1 = 0, PRMUD0 = 0)

While the TCUD pin is kept high in this mode, the UDC is decremented the number of external count clocks by one each time an external count clock is input from the TIUD pin. While the TCUD pin is kept low, the UDC is incremented the number of external count clocks.

Fig. 7-37 shows how the UDC operates when the edge for the TIUD pin is specified as a rising edge by the TUM2 register.

Fig. 7-37. Example of the Up/Down Counter (UDC) Operation in Mode 1 When the Active Edge for the TIUD Pin Is Specified to a Rising Edge

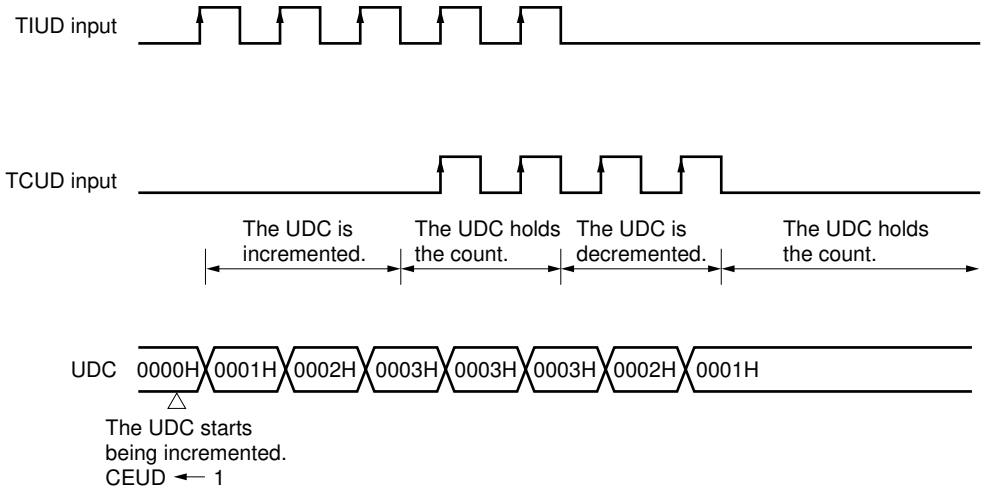


(b) Mode 2 (PRMUD1 = 0, PRMUD0 = 1)

In this mode, the UDC is incremented on the specified edge of an input to the TIUD pin (specified by timer unit mode register 2 (TUM2)). The UDC is decremented on the rising edge of an input to the TCUD pin. If count clock pulses are simultaneously input to the TIUD and TCUD pins, the UDC does not count pulses, but holds the previous value.

Fig. 7-38 shows how the UDC operates when the edge for the TIUD pin is specified as a rising edge.

Fig. 7-38. Example of the Up/Down Counter (UDC) Operation in Mode 2 (When the Active Edge for the TIUD Pin Is Specified to a Rising Edge)



(c) Mode 3 (PRMUD1 = 1, PRMUD0 = 0)

In this mode, signals 90 degrees out of phase (such as outputs of the shaft encoder for the servo motor) are input to the TIUD and TCUD pins to use them as count clock pulses.

The UDC automatically toggles between incrementing and decrementing by detecting a relative phase advance or delay between these two signals.

When the signals 90 degrees out of phase are input to the TIUD and TCUD pins, the UDC samples the level of the signal on the TCUD pin on the specified edge of an input to the TIUD pin.

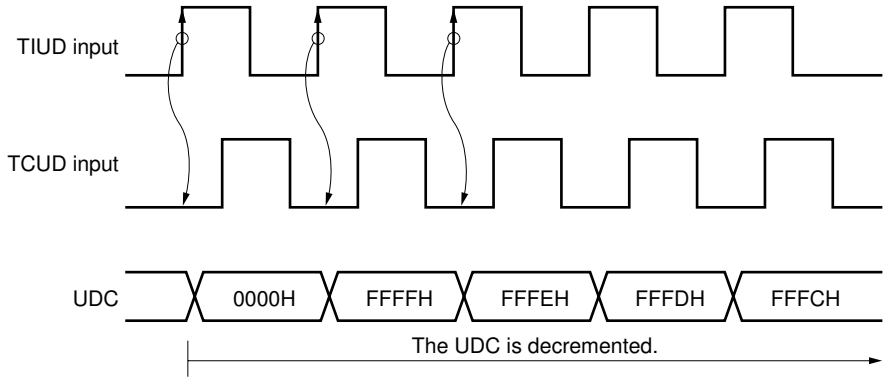
When the sampled level of the signal on the TCUD pin is low, the UDC is decremented on the specified edge of an input to the TIUD pin.

When the sampled level of the signal on the TCUD pin is high, the UDC is incremented on the specified edge of an input to the TIUD pin.

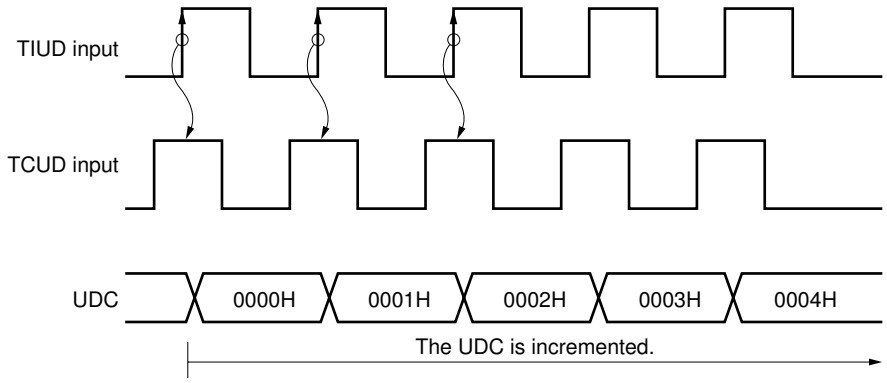
Fig. 7-39 shows the operation of the UDC when an edge of the signal on the TIUD pin is specified as a rising edge.

Fig. 7-39. Example of the Up/Down Counter (UDC) Operation in Mode 3 (When the Active Edge for the TIUD Pin Is Specified to a Rising Edge)

(1) When the UDC is decremented



(2) When the UDC is incremented



(d) Mode 4 (PRMUD1 = 1, PRMUD0 = 1)

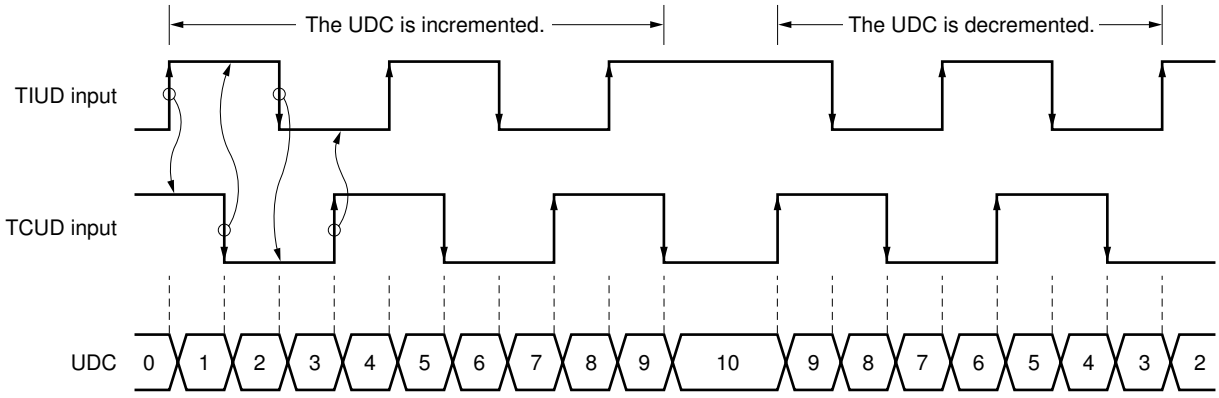
As with mode 3, in mode 4, signals 90 degrees out of phase (such as outputs of the shaft encoder for the servo motor) are input to the TIUD and TCUD pins to use them as count clock pulses.

When the signals 90 degrees out of phase are input to the TIUD and TCUD pins, the UDC automatically toggles between incrementing and decrementing at the timing as shown in Fig. 7-40.

In mode 4, the UDC is incremented or decremented on both the rising and falling edges of the out-of-phase signals. The UDC is incremented or decremented four times per cycle of input signals. (The UDC is called a four-times counter.)

Caution Specifying mode 4 as the operating mode for the UDC invalidates edges specified for the TIUD pin by timer unit mode register 2 (TUM2).

Fig. 7-40. Example of the Up/Down Counter (UDC) Operation in Mode 4



7.4 Timer Output Function

Toggle modes or set or reset modes can be specified for outputs on the TO00, TO02, TO04, TO10, and TO20 pins. Only toggle modes can be specified for outputs on the TO01, TO03, TO05, TO11, and TO21 pins.

Registers TOC0 to TOC2 specify the outputs of timers (specification of active levels, specification of whether the outputs are enabled or disabled, and selection of a toggle mode or set or reset mode).

Tables 7-23 and 7-24 list the correspondence between timer output pins and control signals. Fig. 7-41 is a block diagram of timer output for timers TM0, TM1, and TM2.

Table 7-23. Timer Output Pins and Toggle Signals

Timer output pin	Toggle output
TO00	INTCM00
TO01	INTCM01
TO02	INTCM02
TO03	INTCM03
TO04	INTCC00
TO05	INTCC01
TO10	INTCM10
TO11	INTCM11
TO20	INTCM20
TO21	INTCM21

Table 7-24. Timer Output Pins and Set and Reset Signals

Timer output pin	Set signal	Reset signal
TO00	INTCM00	INTCM01
TO02	INTCM02	INTCM03
TO04	INTCC00	INTCC01
TO10	INTCM10	INTCM11
TO20	INTCM20	INTCM21

Caution If the output level is changed when timer output has been enabled by pins TO04 and TO05, the interrupt request flag (PIC0.7, PIC1.7) will be set (1). Therefore, when using as the timer output, preset the interrupt mask flag (PIC0.6, PIC1.6) to "1" to set the interrupt mask state.

Fig. 7-41. Block Diagrams of Timer Output (TM0, TM1, TM2) (1/2)

Timer 0 (TM0)

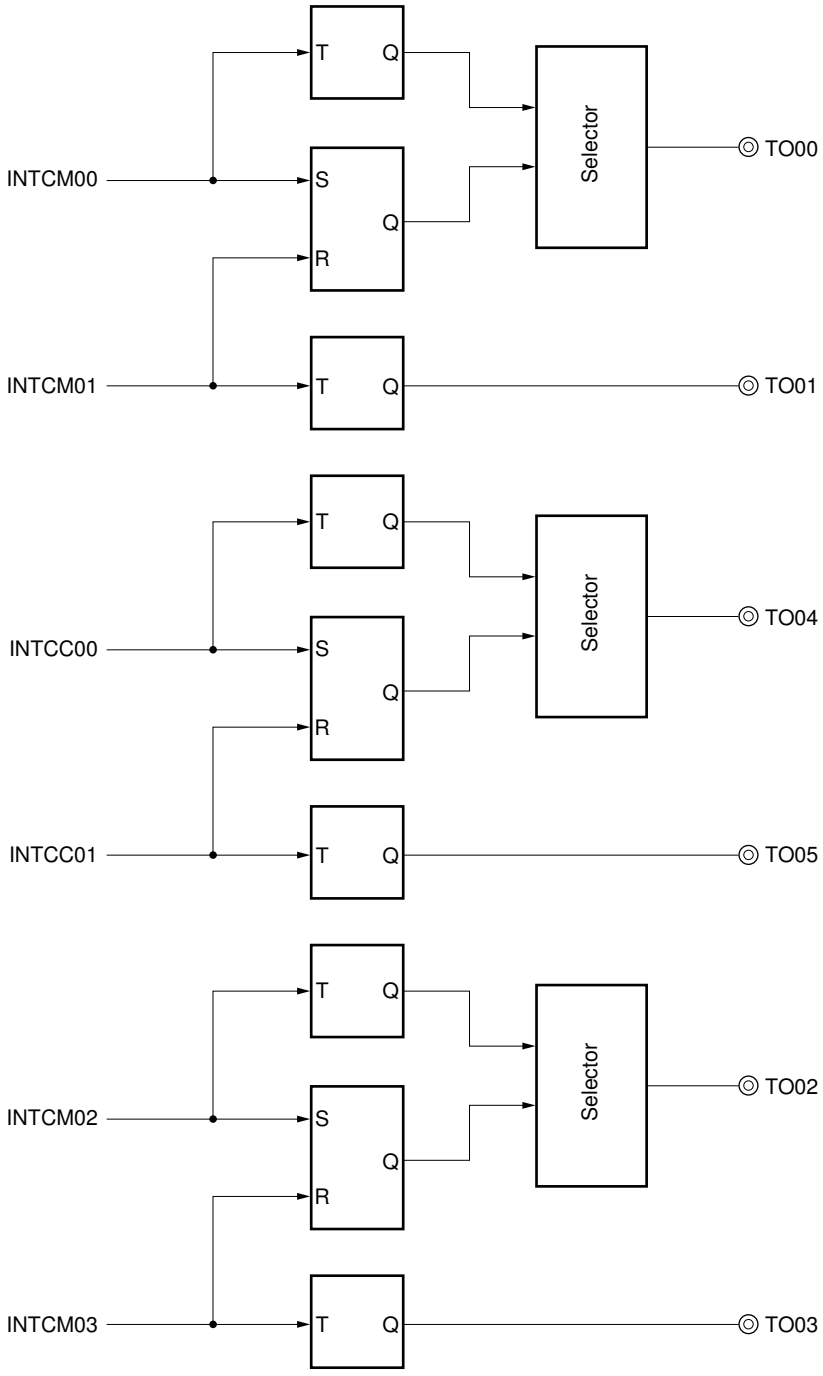
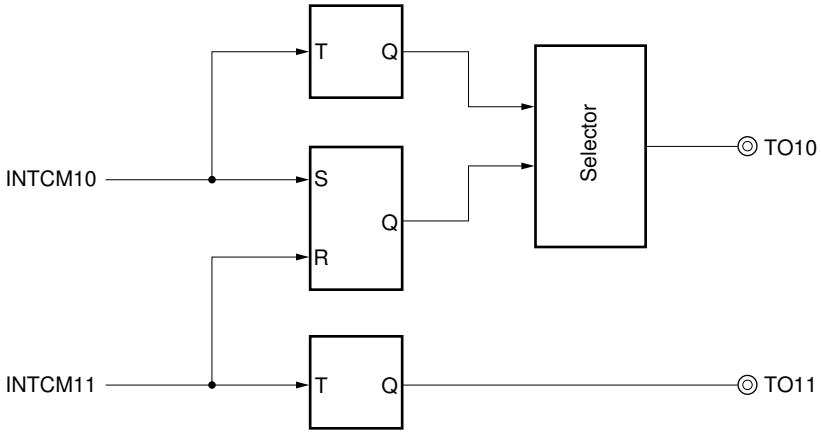
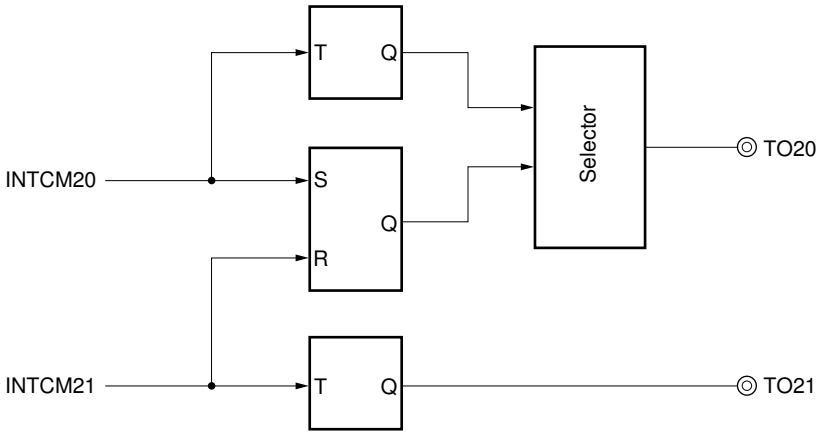


Fig. 7-41. Block Diagrams of Timer Output (TM0, TM1, TM2) (2/2)

Timer 1 (TM1)



Timer 2 (TM2)



7.5 Real-Time Output Function

The real-time output port (RTP) outputs data held in the buffer registers in phase with a trigger signal from the real-time pulse unit (RPU).

Software can specify the combination of real-time output port registers (RTPH and RTPL) and output pins. This allows multi-channel synchronous pulse output to be controlled easily.

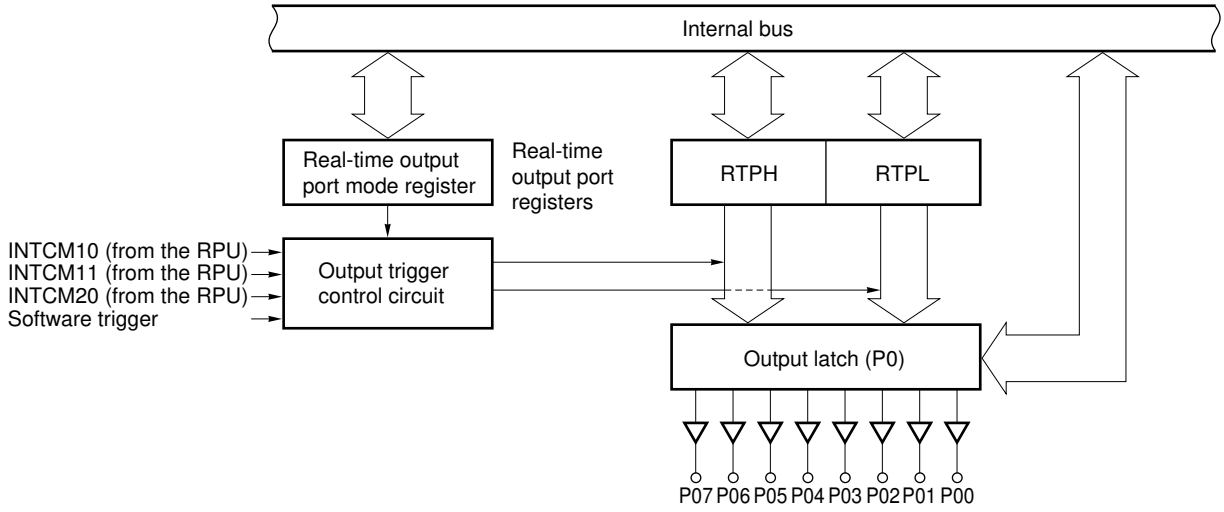
7.5.1 Configuration

The RTP multiplexes signals to port 0 (P0). The RTP has the following three registers to control the output status:

- Real-time output port mode register (RTPM)
- Real-time output port register H (RTPH)
- Real-time output port register L (RTPL)

The real-time output is changed by a trigger signal specified by the TRH1, TRH0, TRL1, and TRL0 bits of the RTPM. The signal to be output by the RPU or a software trigger can be specified as a trigger signal.

Fig. 7-42. Block Diagram of the Real-Time Output Port



7.5.2 Control registers

(1) Real-time output port mode register (RTPM)

This register is an 8-bit register that specifies the operating mode for the real-time output port.

Fig. 7-43 shows the format of the RTPM.

Data can be read from or written into the RTPM by a bit manipulation instruction or 8-bit manipulation instruction.

A $\overline{\text{RESET}}$ input signal sets the RTPM to 00H.

Fig. 7-43. Format of the Real-Time Output Port Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
RTPM	TRH1	TRH0	TRL1	TRL0	0	RCM2	RCM1	RCM0	FF63H	00H	R/W

RCM2	RCM1	RCM0	Port 0 output bits for the real-time output port registers (RTPH and RTPL)	
			RTPH	RTPL
0	0	0	P07 to P00	–
0	0	1	P07 to P01	P00
0	1	0	P07 to P02	P01, P00
0	1	1	P07 to P03	P02 to P00
1	0	0	P07 to P04	P03 to P00
1	0	1	Setting prohibited	
1	1	0		
1	1	1		

TRL1	TRL0	Trigger signal for real-time output port register L (RTPL)
0	0	INTCM10
0	1	INTCM11
1	0	INTCM20
1	1	Software trigger

TRH1	TRH0	Trigger signal for real-time output port register H (RTPH)
0	0	INTCM10
0	1	INTCM11
1	0	INTCM20
1	1	Software trigger

(2) Real-time output port register H (RTPH)

This register is an 8-bit register that stores the upper half of the output data of the real-time output port.

Write data is output to the pins connected to the RTPH according to the setting of the corresponding bit of the real-time output port mode register (RTPM).

When a signal output by the real-time pulse unit (RPU) is specified as an output trigger to the real-time output port, the data written in the RTPH is output to the pins in phase with the trigger signal. When a software trigger is specified as a trigger signal, the data written in the RTPH is output to the pins.

Data can be read from or written into the RTPH by a bit manipulation instruction or 8-bit manipulation instruction.

The contents of the RTPH become undefined after $\overline{\text{RESET}}$ signal is input.

(3) Real-time output port register L (RTPL)

This register is an 8-bit register that stores the lower half of the output data of the real-time output port.

Write data is output to the pins connected to the RTPL according to the setting of the corresponding bit of the real-time output port mode register (RTPM).

When a signal output by the real-time pulse unit (RPU) is specified as an output trigger to the real-time output port, the data written in the RTPL is output to the pins in phase with the trigger signal. When a software trigger is specified as a trigger signal, the data written in the RTPL is output to the pins.

Data can be read from or written into the RTPL by a bit manipulation instruction or 8-bit manipulation instruction.

The contents of the RTPL become undefined after $\overline{\text{RESET}}$ signal is input.

The real-time output port registers (RTPH and RTPL) are read regardless of the settings of the port 0 mode register (PM0) and port 0 mode control register (PMC0). Read data varies according to the settings of RCM0 to RCM2 as shown in Table 7-25.

Table 7-25. Data of RTPHn and RTPLn (n = 0 to 7) to Be Read

RCM2	RCM1	RCM0	RTPH contents to be read								RTPL contents to be read							
			7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	0	0	RTPH								0							
0	0	1	RTPH							0	0							RTPL
0	1	0	RTPH						0		0						RTPL	
0	1	1	RTPH					0			0					RTPL		
1	0	0	RTPH				0				0				RTPL			
Settings other than those above			Setting prohibited															

Caution 0 is always read when the RTPM register specifies data of the RTPH and RTPL registers that is not connected to any pin.

7.5.3 Operation

To specify the function of port 0 (P0) for the real-time output port, set the corresponding bit of the port 0 mode control register (PMC0) to 1 and the TRG2 bit of A/D converter mode register 1 (ADM1) to 0.

The real-time output port mode register (RTPM) is used to specify the connection of the real-time output port registers (RTPH and RTPL) with the output pins (P00 to P07) and output triggers.

When a signal output by the real-time pulse unit (RPU) is specified as an output trigger, the data written in the real-time output port registers (RTPH and RTPL) is automatically output to the port in phase with the trigger signal.

When a software trigger is specified as a trigger signal, the output of the port is changed on the timing of writing data in the RTPH and RTPL.

Caution When a software trigger is specified as an output trigger to the real-time output port, the contents of the RTPH and RTPL are output to the pins. Before specifying a software trigger, write data in the RTPH and RTPL.

CHAPTER 8 A/D CONVERTER

The μPD78356 contains an ultrahigh-speed, high-resolution 10-bit analog/digital (A/D) converter. The A/D converter has eight analog input pins (ANI0 to ANI7) and eight 10-bit A/D conversion result registers (ADCRs) that store conversion results.

The converter uses the serial-parallel conversion system and stores conversion results in ADCRs. This system allows ultrahigh-speed, high-precision conversion (conversion time: approx. 2 μs at an external clock speed of 32 MHz).

Either of the following two conversion modes can be specified for the A/D converter. Software specifies the operating mode for the A/D converter with the A/D converter mode register 0 (ADM0), enabling the A/D conversion suitable for an application system to be selected.

- **Select mode**
  The converter converts a single analog input data item.

- **Scan mode**
  The converter converts multiple analog input data items.

In each operating mode, the converter stores conversion results in ADCRs each time it terminates A/D conversion. Upon completion of A/D conversion, an A/D conversion termination interrupt (INTAD) occurs. The interrupt enables a macro service, such as automatic data transfer, to be started by hardware.

Any one of the following three conversion timing triggers can be specified for the A/D converter. Software specifies a conversion trigger with the A/D converter mode register 1 (ADM1), enabling the fine timing control suitable for an application system to be selected.

- **A/D trigger mode**
  A/D conversion is started by setting the CS bit of the ADM0 register to 1.

- **Timer trigger mode**
  A/D conversion is started by a trigger signal output by the real-time pulse unit (RPU).

- **External trigger mode**
  A/D conversion is started by the input of external triggers (ADTRG0 to ADTRG3).

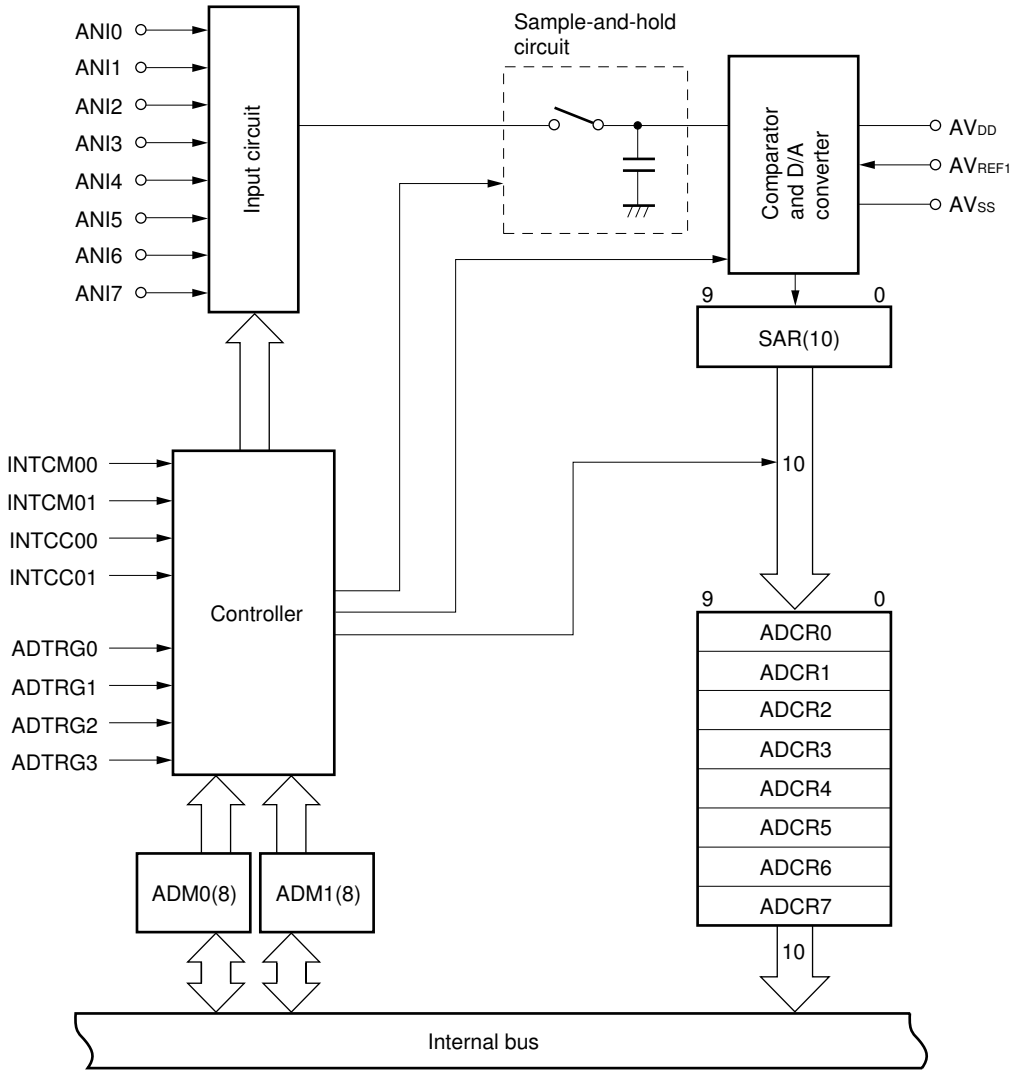
Caution The A/D trigger mode, timer trigger mode, or external trigger mode can be specified for each pin ANI0 to ANI3. However, the A/D trigger mode is always specified for the ANI4 to ANI7 pins.

8.1 Configuration

Fig. 8-1 shows the block diagram of the A/D converter. Fig. 8-2 lists the operating modes and trigger modes for the A/D converter.

The four high-order channels (ANI4 to ANI7) of the analog inputs for the A/D converter are always used in the A/D trigger mode. The A/D trigger mode, timer trigger mode, or external trigger mode, however, can be specified for each of the four low-order channels (ANI0 to ANI3). Trigger modes are selected by software.

Fig. 8-1. Block Diagram of the A/D Converter



(1) Input circuit

The input circuit selects an analog input and sends the analog input to the sample-and-hold circuit according to the operating mode and trigger mode specified by A/D converter mode registers 0 and 1 (ADM0 and ADM1).

(2) Sample-and-hold circuit

The sample-and-hold circuit samples each of the analog inputs successively received from the input circuit and sends them to the comparator. The sample-and-hold circuit, however, holds analog input being converted to digital data.

(3) Comparator

The comparator compares the analog input voltage with the output voltage of the D/A converter to determine whether the voltages are the same.

(4) D/A converter

The D/A converter outputs the voltage that matches the analog input voltage.
The successive approximation register (SAR) controls the output voltage of the D/A converter.

(5) Successive approximation register (SAR)

SAR is a 10-bit register for specifying the data to control the output voltage of the D/A converter to be compared with the analog input voltage.
Upon completion of A/D conversion, the contents of SAR (conversion result) are stored in the A/D conversion result register (ADCR). When all the specified A/D conversions are terminated, an A/D conversion termination interrupt (INTAD) occurs.

(6) Controller

The controller selects an analog input, generates the operation timing signal for the sample-and-hold circuit, and controls a conversion trigger according to the operating mode and trigger mode specified by ADM0 and ADM1.

(7) ANI0 to ANI7 pins

The ANI0 to ANI7 pins are the eight channels of the analog input pins to the A/D converter.

Caution Apply the reference voltage to the ANI0 to ANI7 pins. If a voltage in the range of V_{DD} to V_{SS} inclusive (even in the range of the absolute maximum ratings) is applied to these pins, the converted values of the channels are undefined, and the converted values of other channels may also be affected.

(8) AV_{REF1} pin

This pin is used to input the reference voltage of the A/D converter.

The signals input to the ANI0 to ANI7 pins are converted to digital signals based on the voltage applied between AV_{REF1} and AV_{SS}.

(9) AV_{SS} pin

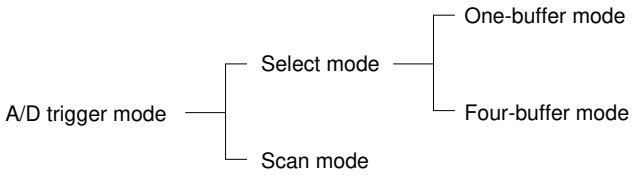
This pin is the ground pin of the A/D converter. Input the V_{SS} level to the AV_{SS} pin.

(10) AV_{DD} pin

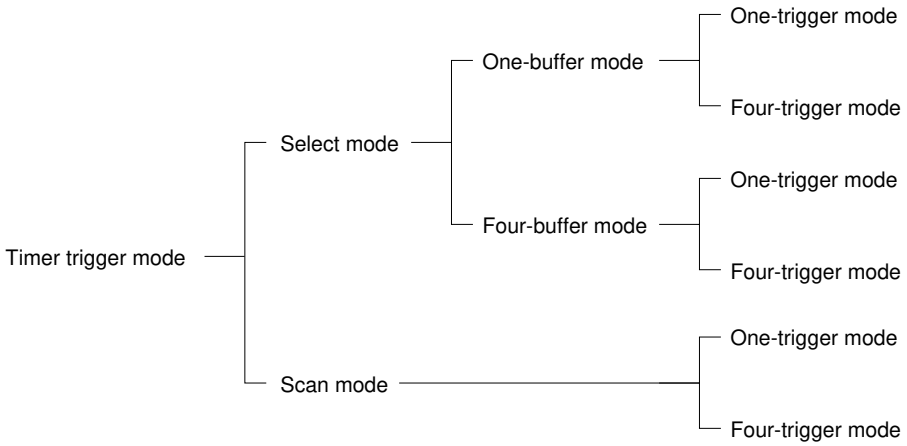
Input the V_{DD} level to the AV_{DD} pin.

Fig. 8-2. Operating Modes and Trigger Modes for the A/D Converter

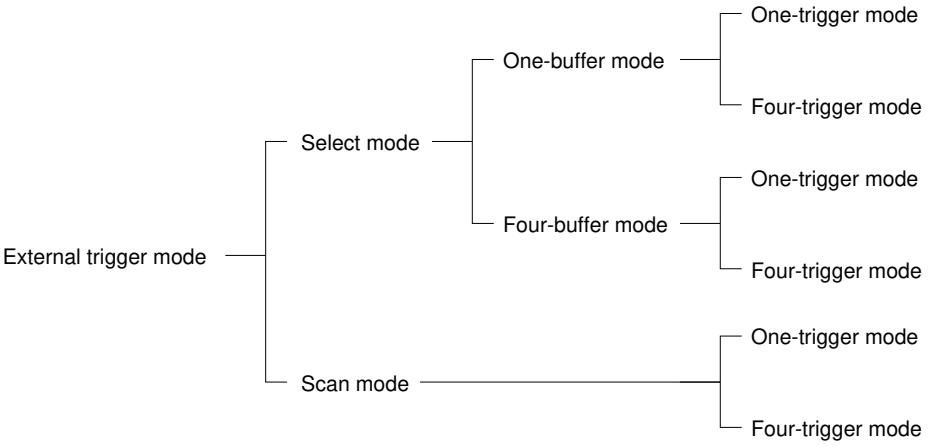
(1) A/D trigger mode



(2) Timer trigger mode



(3) External trigger mode



8.2 A/D Converter Mode Register 0 (ADM0)

A/D converter mode register 0 (ADM0) is an 8-bit register that selects analog input pins for the A/D converter, specifies the operating mode for the A/D converter, specifies the buffer mode, and controls the operation of the A/D converter.

Data can be read from or written into ADM0 with a bit manipulation instruction or 8-bit manipulation instruction. When data is written into ADM0 during A/D conversion, A/D conversion is initialized and carried out from the beginning.

When a $\overline{\text{RESET}}$ signal is input, ADM0 is set to 00H.

Fig. 8-3 shows the format of ADM0.

Fig. 8-3. Format of A/D Converter Mode Register 0 (ADM0)

Symbol	<7>	6	5	4	3	2	1	0	Address	When reset	R/W
ADM0	CS	0	BS	MS	0	ANIS2	ANIS1	ANIS0	FF68H	00H	R/W

ANIS2	ANIS1	ANIS0	Select mode	Scan mode	
				A/D trigger mode	Timer/external trigger mode
0	0	0	ANI0	ANI0	1
0	0	1	ANI1	ANI0, ANI1	2
0	1	0	ANI2	ANI0 to ANI2	3
0	1	1	ANI3	ANI0 to ANI3	4
1	0	0	ANI4	ANI0 to ANI4	ANI0 to ANI4
1	0	1	ANI5	ANI0 to ANI5	ANI0 to ANI5
1	1	0	ANI6	ANI0 to ANI6	ANI0 to ANI6
1	1	1	ANI7	ANI0 to ANI7	ANI0 to ANI7

MS	A/D conversion mode
0	Scan mode
1	Select mode

BS	Buffer mode in the select mode
0	One-buffer mode
1	Four-buffer mode

CS	Control of A/D conversion
0	A/D conversion is stopped.
1	A/D conversion is started.

The following describes each bit of ADM0.

(1) ANIS0 to ANIS2 bits (bits 0 to 2)

These bits select an analog input pin (ANI0 to ANI7) for A/D conversion.

Each of these bits specifies a pin corresponding to the operating mode for the A/D converter.

Table 8-1. Analog Input Pins to Be Selected

ANIS2	ANIS1	ANIS0	Select mode	Scan mode	
				A/D trigger mode	Timer trigger mode or external trigger mode
0	0	0	ANI0	ANI0	1
0	0	1	ANI1	ANI0, ANI1	2
0	1	0	ANI2	ANI0 to ANI2	3
0	1	1	ANI3	ANI0 to ANI3	4
1	0	0	ANI4	ANI0 to ANI4	ANI0 to ANI4
1	0	1	ANI5	ANI0 to ANI5	ANI0 to ANI5
1	1	0	ANI6	ANI0 to ANI6	ANI0 to ANI6
1	1	1	ANI7	ANI0 to ANI7	ANI0 to ANI7

Caution The order in which the ANI0 to ANI3 bits are scanned is specified according to the order in which the match signals from compare registers or external trigger signals are generated in the timer trigger mode or external trigger mode for the scan mode. As a result, the ANIS0 to ANIS2 bits specify no analog input pin, they specify the number of trigger inputs.

(2) MS bit (bit 4)

The MS bit specifies the operating mode for the A/D converter such as the scan or select mode.

- When the MS bit is set to 0, the scan mode is specified.
- When the MS bit is set to 1, the select mode is specified.

(3) BS bit (bit 5)

The BS bit specifies a buffer mode for A/D conversion in the select mode.

- When the BS bit is set to 0, the one-buffer mode is specified.
- When the BS bit is set to 1, the four-buffer mode is specified.

(4) CS bit (bit 7)

Manipulation of the CS bit varies according to the trigger mode of the A/D converter.

The CS bit enables or disables A/D conversion. Setting the CS bit to 1 starts A/D conversion.

Upon completion of A/D conversion, the CS bit is reset to 0. The CS bit is therefore used to check the state of A/D conversion.

In the timer trigger mode or external trigger mode, A/D conversion is possible regardless of the setting of the CS bit.

When the CS bit is 0, the input of a trigger signal from the real-time pulse unit (RPU) or external unit sets the CS bit to 1. Upon completion of A/D conversion for the channels specified by ADM0, the CS bit is reset to 0.

Table 8-2 shows the state of A/D conversion in the A/D trigger mode. Table 8-3 shows the state of A/D conversion in the timer trigger mode or external trigger mode.

Table 8-2. CS Bit Manipulation and A/D Conversion in the A/D Trigger Mode

CS bit manipulation	Setting of the CS bit	
	0	1
Write	A/D conversion is disabled.	A/D conversion is started.
Read	A/D conversion stop state	A/D conversion start state

Table 8-3. CS Bit Manipulation and A/D Conversion in the Timer Trigger Mode or External Trigger Mode

CS bit manipulation	Setting of the CS bit	
	0	1
Read	Trigger standby state	A/D conversion start state

8.3 A/D Converter Mode Register 1 (ADM1)

A/D converter mode register 1 (ADM1) is an 8-bit register that specifies the conversion time, the trigger mode of the A/D converter, and an edge in the external trigger mode.

Data can be read from or written into ADM1 with a bit manipulation instruction or 8-bit manipulation instruction. When data is written in ADM1 during A/D conversion, A/D conversion is initialized and carried out from the beginning.

When a $\overline{\text{RESET}}$ signal is input, ADM1 is set to 07H.

Fig. 8-4 shows the format of ADM1.

Fig. 8-4. Format of A/D Converter Mode Register 1 (ADM1)

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
ADM1	EDG	TRG2	TRG1	TRG0	0	FR2	FR1	FR0	FF69H	07H	R/W

FR2	FR1	FR0	Number of conversion clocks	Conversion time (μs)		
				f_{CLK} = 16.0 MHz	f_{CLK} = 12.5 MHz	f_{CLK} = 8.0 MHz
0	0	0	8	Setting prohibited		
0	0	1	16	Setting prohibited		2.0
0	1	0	20			2.5
0	1	1	24			3.0
1	0	0	28	Setting prohibited	2.24	3.5
1	0	1	32	2.0	2.56	4.0
1	1	0	64	4.0	5.12	8.0
1	1	1	128	8.0	10.24	16.0

TRG2	TRG1	TRG0	Trigger mode	
0	0	0	A/D trigger mode	
0	0	1		
0	1	0	Timer trigger mode	One-trigger mode
0	1	1		Four-trigger mode
1	0	0	External trigger mode	One-trigger mode
1	0	1		Four-trigger mode
Settings other than the above			Setting prohibited	

EDG	Edge of an external input signal in the external trigger mode
0	Falling edge
1	Rising edge

Remark f_{CLK}: Internal system clock

The following describes each bit of ADM1.

(1) FR0 to FR2 bits (bits 0 to 2)

These bits specify the conversion time for the A/D converter.

Each of these bits specifies an appropriate A/D conversion time when the oscillator frequency is changed.

Table 8-4. A/D Conversion Time

FR2	FR1	FR0	Number of conversion clocks	Conversion time (μs)		
				f_{CLK} = 16.0 MHz	f_{CLK} 12.5 MHz	f_{CLK} = 8.0 MHz
0	0	0	8	Setting prohibited		
0	0	1	16	Setting prohibited		2.0
0	1	0	20			2.5
0	1	1	24			3.0
1	0	0	28	Setting prohibited	2.24	3.5
1	0	1	32	2.0	2.56	4.0
1	1	0	64	4.0	5.12	8.0
1	1	1	128	8.0	10.24	16.0

(2) TRG0 to TRG2 bits (bits 4 to 6)

These bits specify a trigger mode for the four low-order channels of the analog input pins (ANI0 to ANI3).

Table 8-5. Trigger Mode for the ANI0 to ANI3 Pins

TRG2	TRG1	TRG0	Trigger mode	Number of triggers
0	0	0	A/D trigger mode	–
0	0	1		
0	1	0	Timer trigger mode	1
0	1	1		4
1	0	0	External trigger mode	1
1	0	1		4
1	1	0	Setting prohibited	–
1	1	1		

(3) EDG bit (bit 7)

This bit specifies the edge (falling or rising) of a trigger signal input from the external unit to the ADTRG0 to ADTRG3 pins in the external trigger mode.

- When the EDG bit is 0, the falling edge is specified.
- When the EDG bit is 1, the rising edge is specified.

8.4 A/D Conversion Result Registers (ADCRs)

The μPD78356 has eight 10-bit A/D conversion result registers (ADCRs).
The contents of each ADCR can only be read separately with 16-bit or 8-bit manipulation instruction.
The A/D conversion results are read from the ADCRs in either of the following two ways:

(1) Word access (16-bit manipulation instruction execution)

Effective word data is read from the low-order 10 bits of an ADCR.
0s are always read from the high-order six bits.
Fig. 8-5 shows the word access to ADCRs.

Fig. 8-5. Word Access to ADCRs

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	R/W
ADCRn	0	0	0	0	0	0	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	R

(n = 0 to 7)

Symbol	Address	When reset
ADCR0	FFB0H	Undefined
ADCR1	FFB2H	
ADCR2	FFB4H	
ADCR3	FFB6H	
ADCR4	FFB8H	
ADCR5	FFBAH	
ADCR6	FFBCH	
ADCR7	FFBEH	

Remark AD0 to AD9: A/D conversion result

(2) Byte access (8-bit manipulation instruction execution)

Data is read from the high-order eight bits of a 10-bit ADCR.

Fig. 8-6 shows the byte access to ADCRs.

Fig. 8-6. Byte Access to ADCRs

Symbol	7	6	5	4	3	2	1	0	R/W
ADCRnH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	R

(n = 0 to 7)

Symbol	Address	When reset
ADCR0H	FFB1H	Undefined
ADCR1H	FFB3H	
ADCR2H	FFB5H	
ADCR3H	FFB7H	
ADCR4H	FFB9H	
ADCR5H	FFBBH	
ADCR6H	FFBDH	
ADCR7H	FFBFH	

Remark AD2 to AD9: A/D conversion result (high-order eight bits of a 10-bit)

8.5 Operation

Only the A/D trigger mode can be specified for the higher four channels (ANI4 to ANI7) of the analog inputs for the A/D converter. However, the A/D trigger mode, timer trigger mode, or external trigger mode can be specified for each of the lower four channels (ANI0 to ANI3).

Unless otherwise specified, the explanation of the lower four channels (ANI0 to ANI3) below also applies to the higher four channels.

8.5.1 Basic operation of the A/D converter

The A/D converter operates as follows.

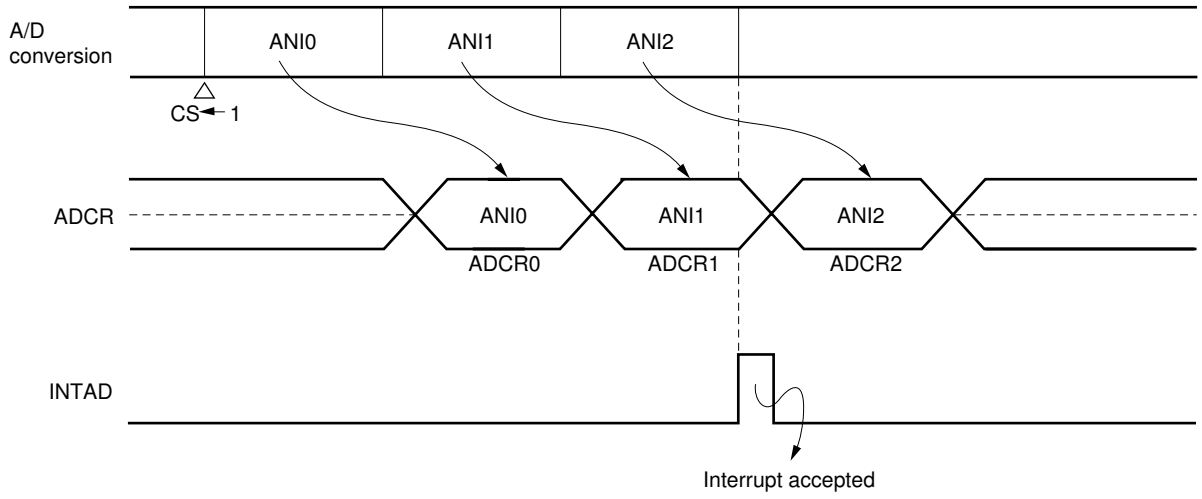
(1) An analog input is selected and an operating mode and trigger mode are specified with A/D converter mode registers 0 and 1 (ADM0 and ADM1). A/D conversion then starts when the CS bit is set to 1.

(2) The comparator compares the analog input voltage with the voltage applied by the D/A converter.

(3) Upon completion of 10-bit comparison, the effective digital value is held in the Successive Approximation Register (SAR), and the value is sent to and held in the A/D conversion result register (ADCR).
When A/D conversion is completed the number of times specified by ADM0, an A/D conversion termination interrupt (INTAD) occurs. The interrupt must be handled with a vectored interrupt or macro service function (see **15.8**).
Fig. 8-7 shows an example of the operation of the A/D converter when it converts the ANI0 to ANI2 pin inputs to digital data in the A/D trigger mode.

Caution Do not apply a voltage out of the range of AV_{SS} to AV_{DD} to the pins used as the input pins of the A/D converter.

Fig. 8-7. Basic Operation of the A/D Converter (A/D Trigger Mode: ANI0 to ANI2)



A/D conversion is started in synchronization with a trigger signal output by the real-time pulse unit (RPU) (timer trigger mode) or in synchronization with a trigger signal input from the external unit (external trigger mode).

When software specifies the timer trigger mode or external trigger mode of the scan mode in the TRG0 to TRG2 bits of ADM1, a trigger signal enters a standby state. The first trigger signal starts A/D conversion. Upon completion of the conversion operation, the next trigger signal enters the standby state. When A/D conversion is completed the number of times specified by ADM0, an A/D conversion termination interrupt (INTAD) occurs.

Fig. 8-8 shows an example of A/D conversion in the four-trigger mode of the external trigger mode.

When data is written in ADM0 and ADM1 during A/D conversion, A/D conversion is initialized and started from the beginning.

Fig. 8-9 shows an example of writing data in ADM0 during A/D conversion of the ANI0 to ANI2 pin inputs in the A/D trigger mode.

When a $\overline{\text{RESET}}$ signal is input, the contents of each ADCR become undefined.

Fig. 8-8. A/D Conversion in Synchronization with a Trigger Signal (External Trigger Mode: Four-Trigger Mode)

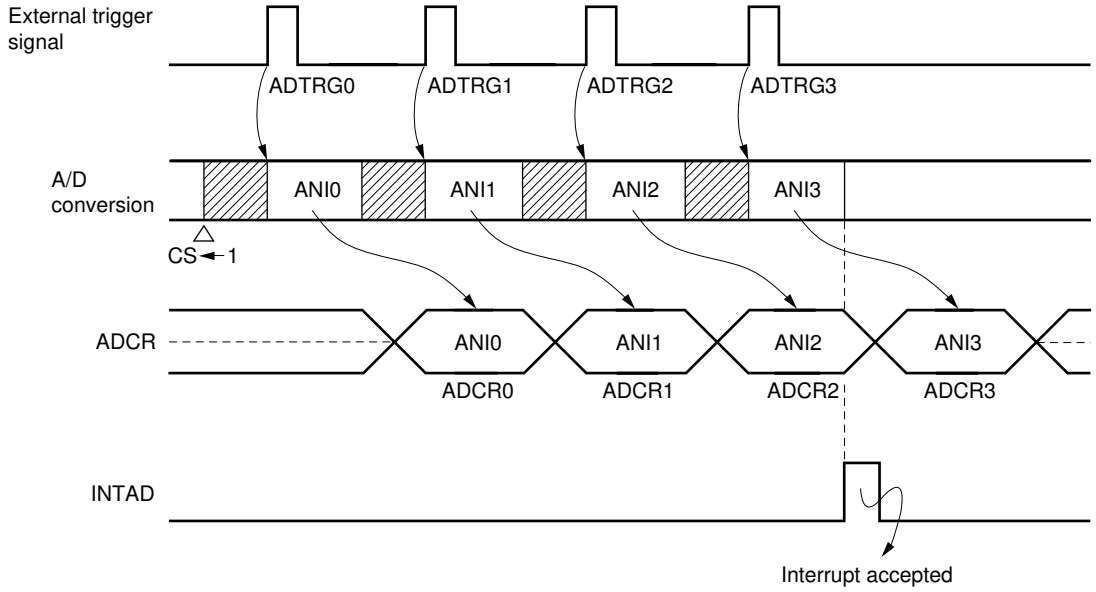
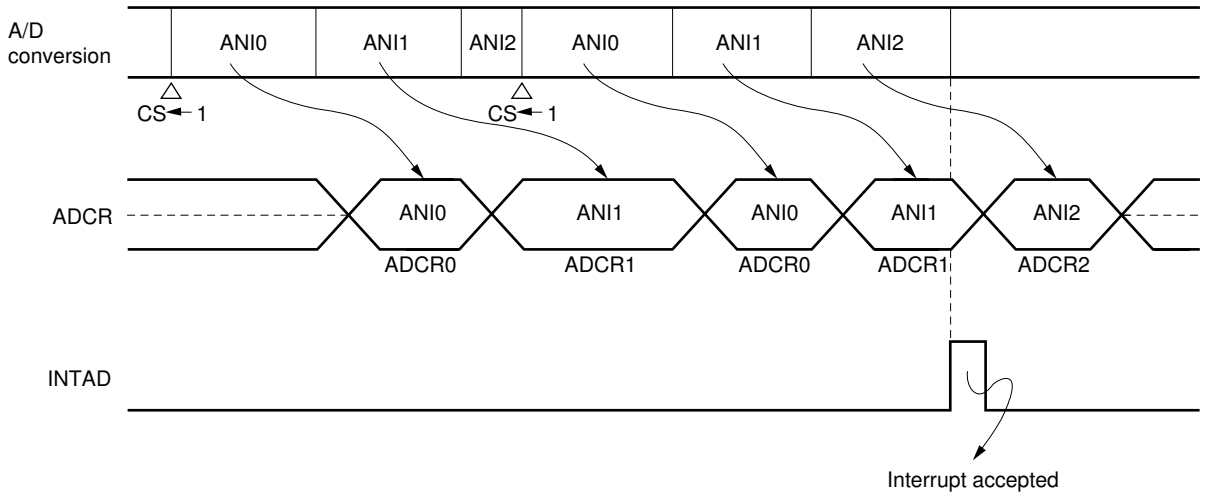


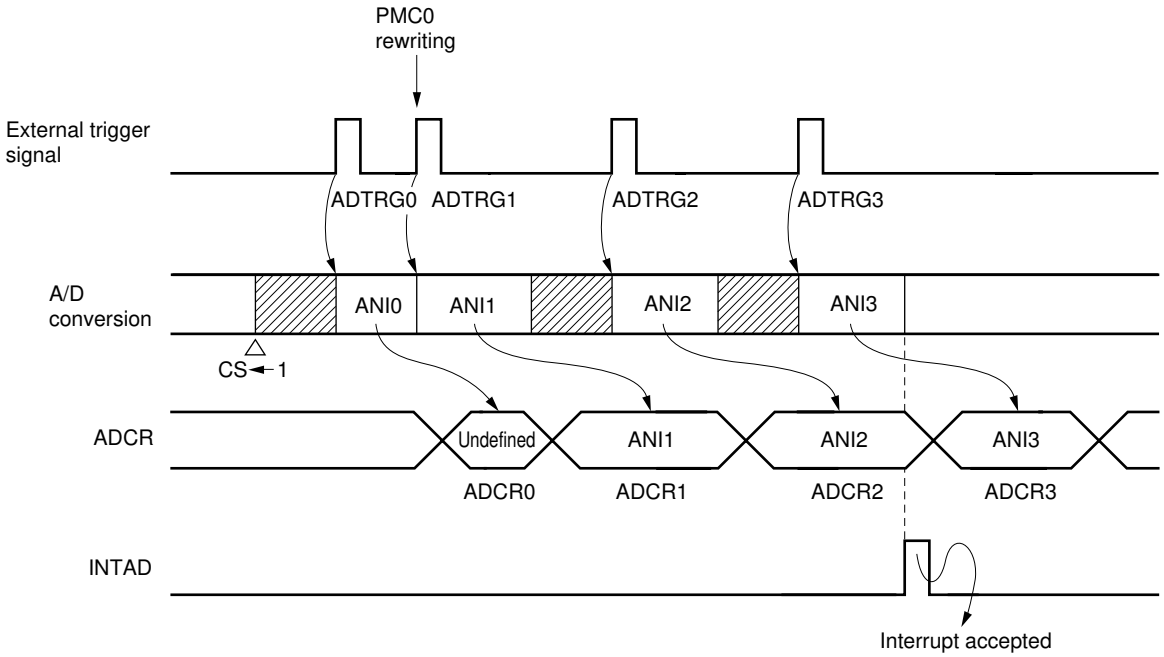
Fig. 8-9 Rewriting Data in ADM during A/D Conversion (A/D Trigger Mode: ANI0 to ANI2)



If the port 0 mode control register (PMC0) is rewritten during A/D conversion in the external trigger mode of the A/D converter, the external trigger edge is generated and the conversion operation is stopped and the result of the stopped A/D conversion will be undefined. But the number of inputs will be counted and A/D conversion will continue to be performed in accordance with the new trigger.

Fig. 8-10 shows an example of manipulation performed when data is rewritten in the PMC0 register during A/D conversion in the external trigger mode (4 trigger mode).

Fig. 8-10. Rewriting Data in PMC0 during A/D Conversion (A/D Trigger Mode: Four-Trigger Mode)



Caution If conversion results are programmed to branch off only when certain values are set in branching processes directly using the A/D conversion results, the conversion results will not become the values due to the effects of conversion differences, and may not branch to the specified routine. Therefore create programs in which conversion results branch when it ranges with the total difference width.

The following shows examples of programs where specific processes are performed when the analog input voltage input to pin ANI0 is $\frac{1}{2}$ AV_{REF1}.

<Wrong example>

```
        CMPW    ADCR0,  #01FFH
        BNE     $UNMATCH
                .
                .           } <1> Process when the A/D conversion result is 1FFH
                .
UNMATCH:
                .
                .           } <2> Process when the A/D conversion results is other than 1FFH
                .
```

If the conversion result does not become 1FFH due to conversion difference, it cannot branch to <1>.

⇩

<Correct example>

```
        CMPW    ADCR0,  #0201H
        BGT     $UNMATCH
        CMPW    ADCR0,  #01FCH
        BLT     $UNMATCH
                .
                .           } <1> Process when the A/D conversion result is 1FCH to 201H
                .
UNMATCH:
                .
                .           } <2>  Process when the A/D conversion results is outside the above range
                .
```

When the conversion result is in the 1FCH to 201H range, it is determined that the $\frac{1}{2}$ AV_{REF1} analog voltage has been input and the processing is executed.

8.5.2 Operating mode for the A/D converter

Either of the following two modes can be selected as the operating mode of the A/D converter. An operating mode is selected with A/D converter mode register 0 (ADM0). Setting the CS bit of ADM0 to 1 starts A/D conversion.

- Select mode
- Scan mode

(1) Select mode

The select mode performs an A/D conversion of one analog input (ANIn: n = 0 to 7) selected with ADM0. The conversion result is stored in the A/D conversion result register (ADCRn: n = 0 to 7) corresponding to the analog input.

The select mode has the following two modes for storing the A/D conversion result in the registers:

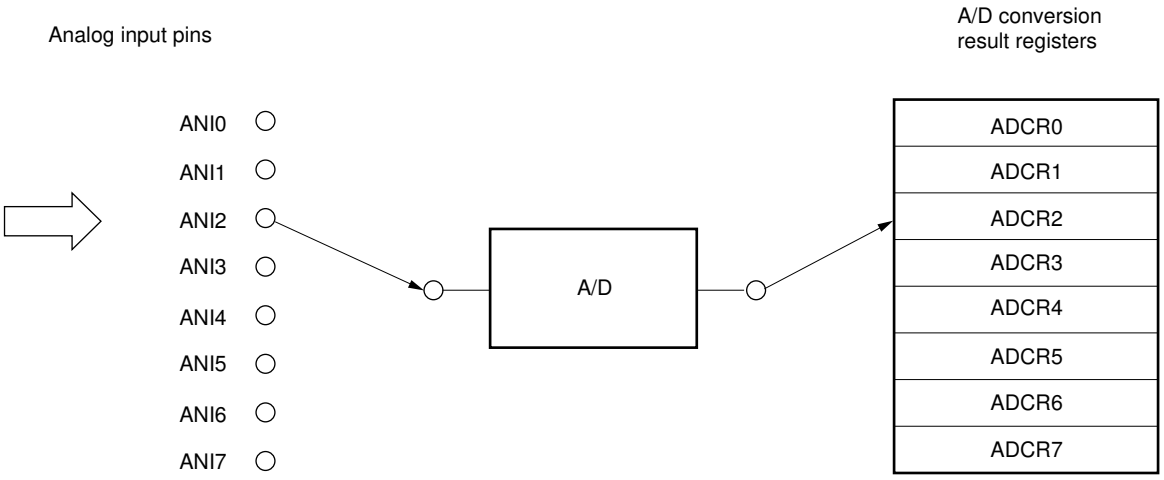
- One-buffer mode
- Four-buffer mode

(a) One-buffer mode

One A/D conversion is performed for one analog input, then the result is stored in one A/D conversion result register. In this case, an analog input corresponds to an A/D conversion result register on a one-to-one basis.

An A/D conversion termination interrupt (INTAD) occurs each time an A/D conversion is performed to indicate that the A/D conversion has been completed.

Fig. 8-11. A/D Conversion in the Select Mode (One-Buffer Mode)



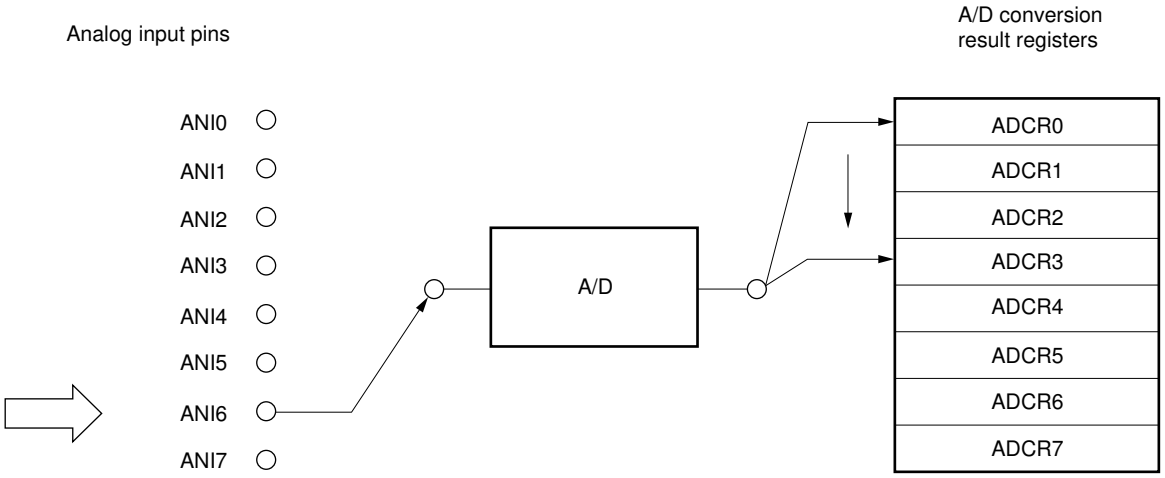
(b) Four-buffer mode

Four A/D conversions are performed for one analog input, then the results are stored in four A/D conversion result registers (ADCR0 to ADCR3).

An A/D conversion termination interrupt (INTAD) occurs upon completion of four A/D conversions.

This mode is useful for finding the average of A/D conversions.

Fig. 8-12. A/D Conversion in the Select Mode (Four-Buffer Mode)

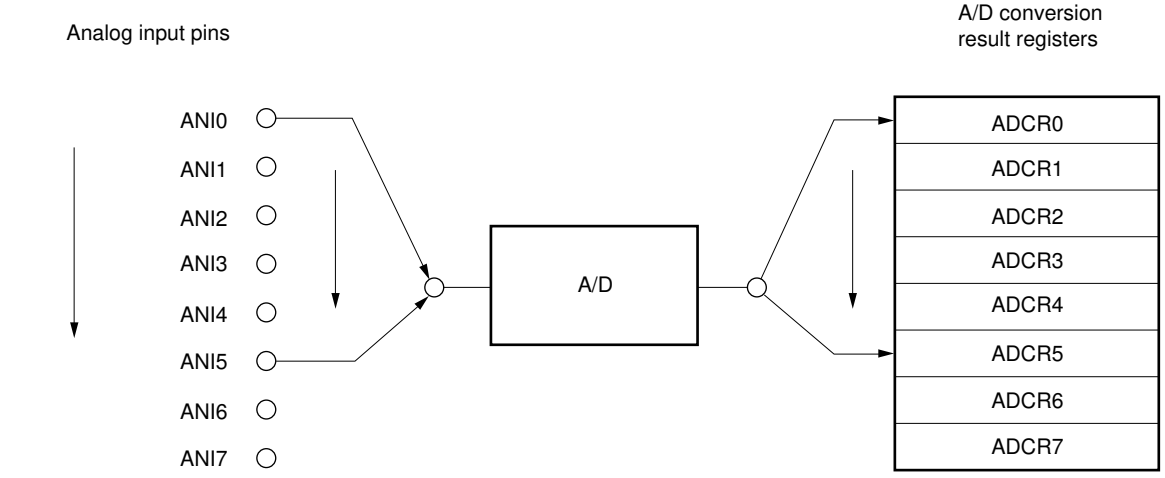


(2) Scan mode

To perform A/D conversion, the scan mode sequentially selects analog inputs on the ANI0 pin for the analog input pin specified by ADM0. The results of A/D conversions are stored in the A/D conversion result registers corresponding to the analog inputs on a one-to-one basis. An A/D conversion termination interrupt (INTAD) occurs upon completion of A/D conversion of the specified analog inputs.

This mode is useful for monitoring multiple analog inputs.

Fig. 8-13. A/D Conversion in the Scan Mode



8.5.3 Trigger modes for the A/D converter

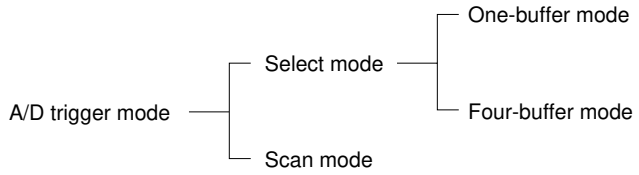
The trigger modes are classified into the following three modes. Each mode is used as a timing trigger to start A/D conversion. One of these trigger modes and an A/D conversion time are specified with A/D converter mode register 1 (ADM1).

The A/D trigger mode is always specified for the higher four channels (ANI4 to ANI7) of the analog inputs for the A/D converter. However, the A/D trigger mode, timer trigger mode, or external trigger mode can be specified for each of the lower four channels (ANI0 to ANI3).

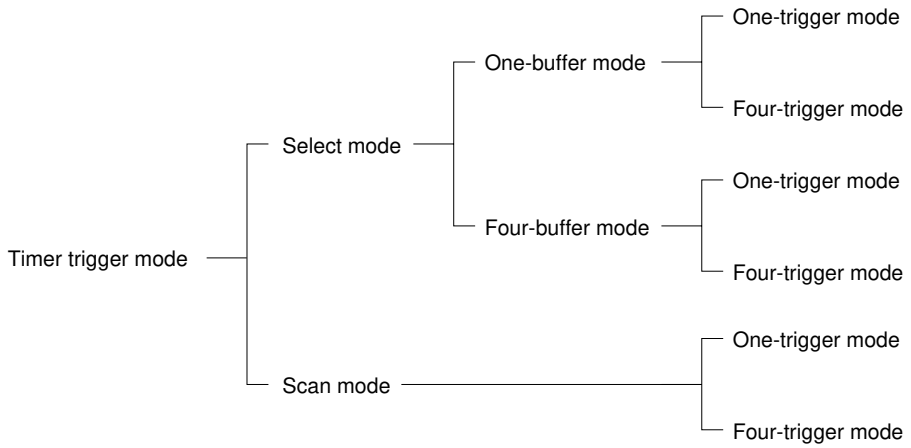
- A/D trigger mode
- Timer trigger mode
- External trigger mode

Fig. 8-14. Operating Modes and Trigger Modes for the A/D Converter

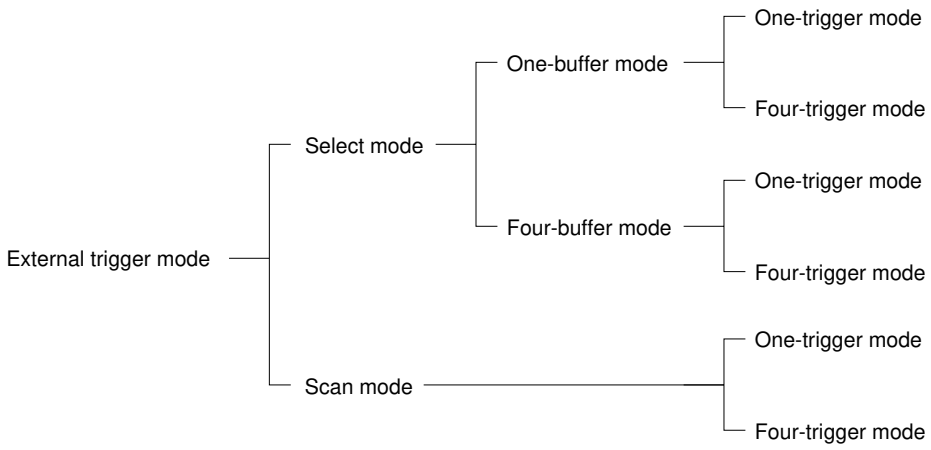
(1) A/D trigger mode



(2) Timer trigger mode



(3) External trigger mode



(1) A/D trigger mode

Writing data in A/D converter mode register 0 (ADM0) starts A/D conversion. A/D conversion can be synchronized using software.

(a) Select mode

The select mode performs the A/D conversion of one analog input (ANIn: n = 0 to 7) selected with ADM0. The conversion result is stored in the A/D conversion result register (ADCRn: n = 0 to 7) corresponding to the analog input.

The select mode has the following two modes (one-buffer mode and four-buffer mode) for storing the A/D conversion result in the registers.

(i) One-buffer mode

One A/D conversion is performed for one analog input, then the result is stored in one A/D conversion result register. In this case, an analog input corresponds to an A/D conversion result register on a one-to-one basis.

An INTAD interrupt occurs each time an A/D conversion is performed, then the A/D conversion is terminated (CS = 0).

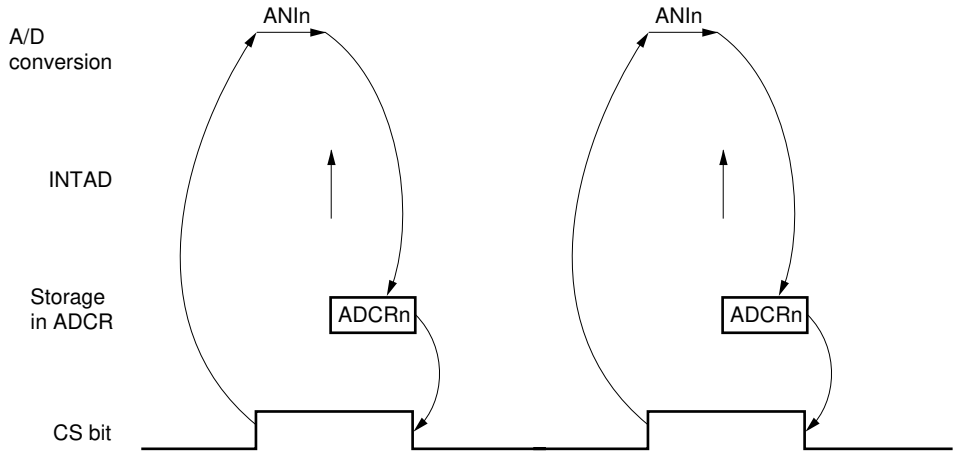
Writing data in ADM0 restarts A/D conversion.

This mode is useful for reading the A/D conversion result each time an A/D conversion is performed.

Analog input		A/D conversion result register
ANIn	->	ADCRn

Remark n = 0 to 7

Fig. 8-15. Example 1 of A/D Conversion in the A/D Trigger Mode (Select Mode, One-Buffer Mode, ANIn)



Remark n = 0 to 7

(ii) Four-buffer mode

Four A/D conversions are performed for one analog input, then the results are stored in four A/D conversion result registers (ADCR0 to ADCR3).

An A/D conversion termination interrupt (INTAD) occurs upon completion of four A/D conversions, then the A/D conversion is terminated (CS = 0).

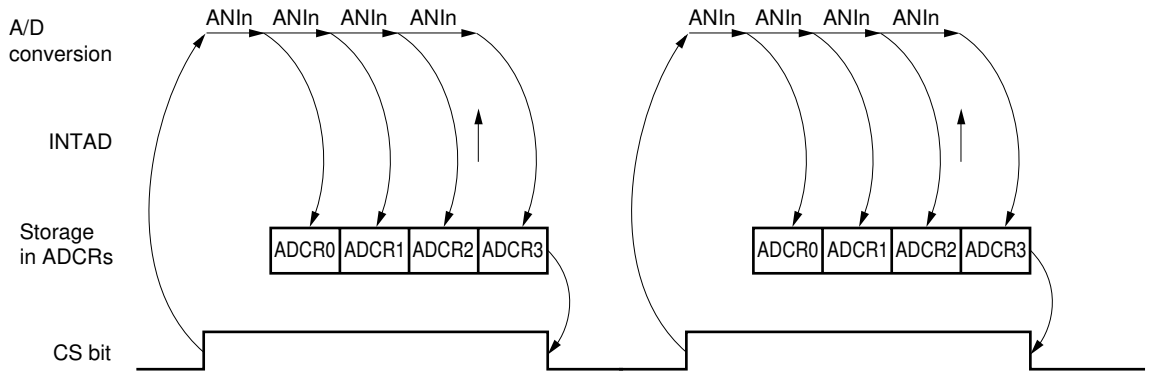
Writing data in ADM0 restarts A/D conversion.

This mode is useful for finding the average of A/D conversion results.

Analog input		A/D conversion result register
ANIn	->	ADCR0
ANIn	->	ADCR1
ANIn	->	ADCR2
ANIn	->	ADCR3

Remark n = 0 to 7

Fig. 8-16. Example 2 of A/D Conversion in the A/D Trigger Mode (Select Mode, Four-Buffer Mode, ANIn)



Remark n = 0 to 7

(b) Scan mode

To perform A/D conversion, the scan mode sequentially selects analog inputs on the ANI0 pin to the analog input pin (ANIn: n = 0 to 7) specified by ADM0. The results of A/D conversions are stored in the A/D conversion result registers corresponding to the analog inputs (ADCRn: n = 0 to 7).

An A/D conversion termination interrupt (INTAD) occurs upon completion of A/D conversion of the specified analog inputs (CS = 0).

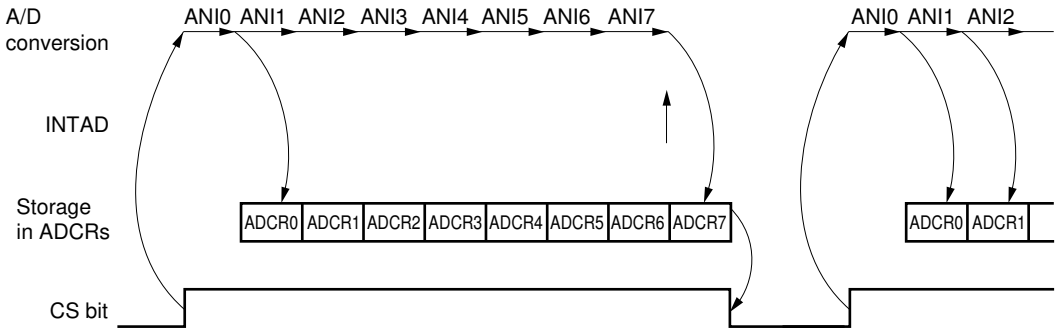
Writing data in ADM0 restarts A/D conversion.

This mode is useful for monitoring multiple analog inputs.

Analog input		A/D conversion result register
ANI0	->	ADCR0
.	.	.
.	.	.
.	.	.
ANIn	->	ADCRn

Remark n = 0 to 7

Fig. 8-17. Example 3 of A/D Conversion in the A/D Trigger Mode (Scan Mode, ANI0 to ANI7)



(2) Timer trigger mode

A/D conversion of analog inputs (ANI0 to ANI3) is performed at the timing specified by compare registers of the real-time pulse unit (RPU).

The occurrence of match interrupts from four compare registers (CM00, CM01, CC00, and CC01) connected to timer 0 (TM0) outputs analog input conversion start signals.

The setting of timer unit mode register 1 (TUM1) specifies whether A/D conversion is terminated after a single A/D conversion (one-shot mode) or is repeated (loop mode).

Cautions 1. Before specifying the timer trigger mode, specify that the capture/compare registers (CC00 and CC01) function as a compare register.

2. The timer trigger mode is effective for ANI0 to ANI3 analog inputs only.

(a) Select mode

An analog input (ANIn: n = 0 to 3) specified by A/D converter mode register 0 (ADM0) is converted to digital data. The conversion result is stored in an A/D conversion result register (ADCRn: n = 0 to 3) corresponding to the analog input.

The select mode has the following two modes (one-buffer mode and four-buffer mode) for storing the A/D conversion result in the registers.

(i) One-buffer mode

One A/D conversion is performed for one analog input, then the result is stored in one A/D conversion result register. In this case, an analog input corresponds to an A/D conversion result register on a one-to-one basis.

The one-buffer mode has the following two modes (one-trigger mode and four-trigger mode) corresponding to the number of triggers.

This mode is useful for reading the A/D conversion result each time an A/D conversion is performed.

<1> One-trigger mode

When a trigger signal (match interrupt (INTCM00)) is input, one A/D conversion is performed for one analog input, then the result is stored in one A/D conversion result register.

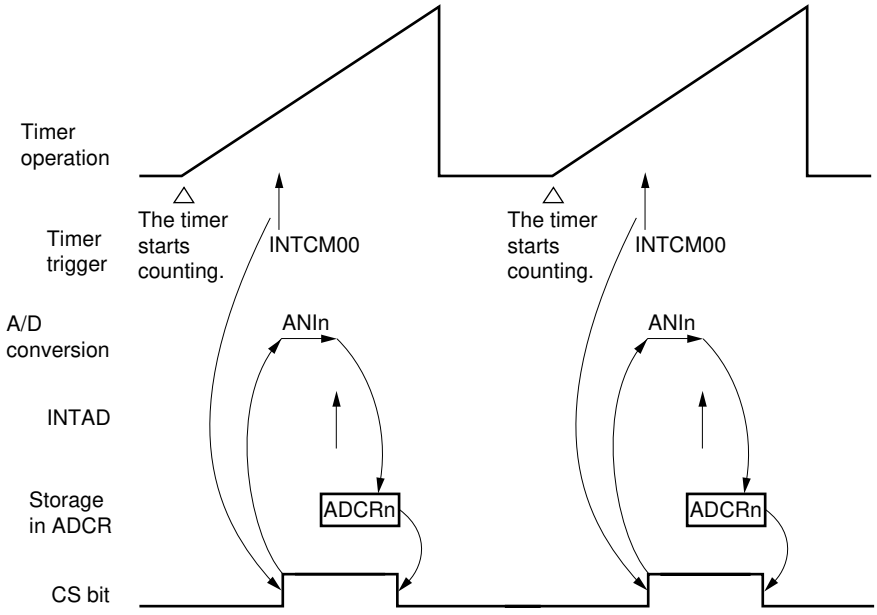
An INTAD interrupt occurs each time an A/D conversion is performed, then the A/D conversion is terminated (CS = 0).

When timer 0 is in the one-shot mode, A/D conversion is terminated after a single A/D conversion. To restart A/D conversion, timer 0 must be restarted. When timer 0 is in the loop mode, A/D conversion is repeated each time an INTCM00 interrupt occurs.

Trigger signal	Analog input		A/D conversion result register
INTCM00	ANIn	->	ADCRn

Remark n = 0 to 3

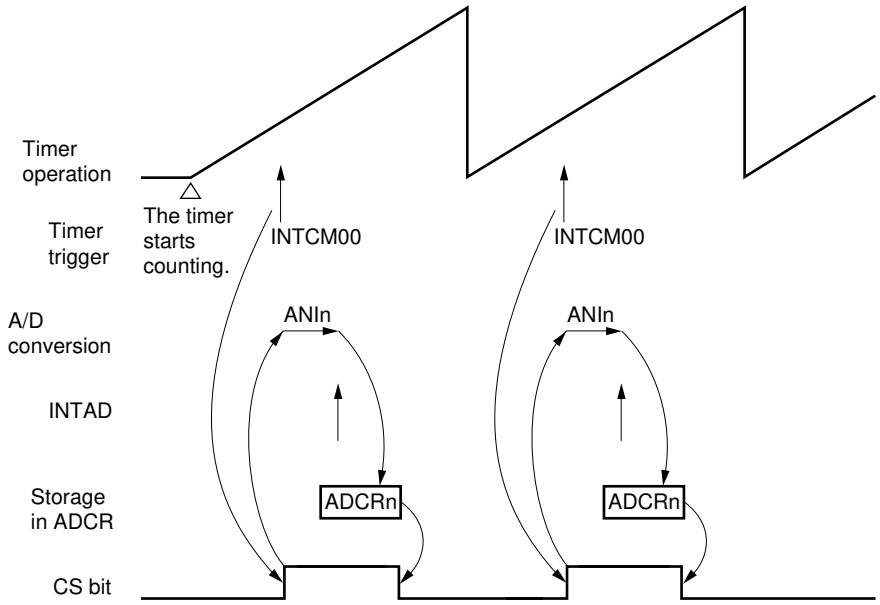
Fig. 8-18. Example 1 of A/D Conversion in the Timer Trigger Mode (Select Mode, One-Buffer Mode, One-Trigger Mode, Timer One-Shot Mode, ANIn)



Remark n = 0 to 3

Caution When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

Fig. 8-19. Example 2 of A/D Conversion in the Timer Trigger Mode (Select Mode, One-Buffer Mode, One-Trigger Mode, Timer Loop Mode, ANIn)



Remark n = 0 to 3

Caution When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

<2> Four-trigger mode

When four trigger signals (four match interrupts (INTCM00, INTCM01, INTCC00, and INTCC01)) are input, an analog input is converted to digital data four times, and the result is stored in one A/D conversion result register.

Each time an A/D conversion is performed, an INTAD interrupt occurs, and the CS bit is reset to 0.

The result of a single A/D conversion is held in the A/D conversion result register until the next A/D conversion is terminated. The conversion result must be loaded into memory when an INTAD interrupt occurs upon completion of the single A/D conversion.

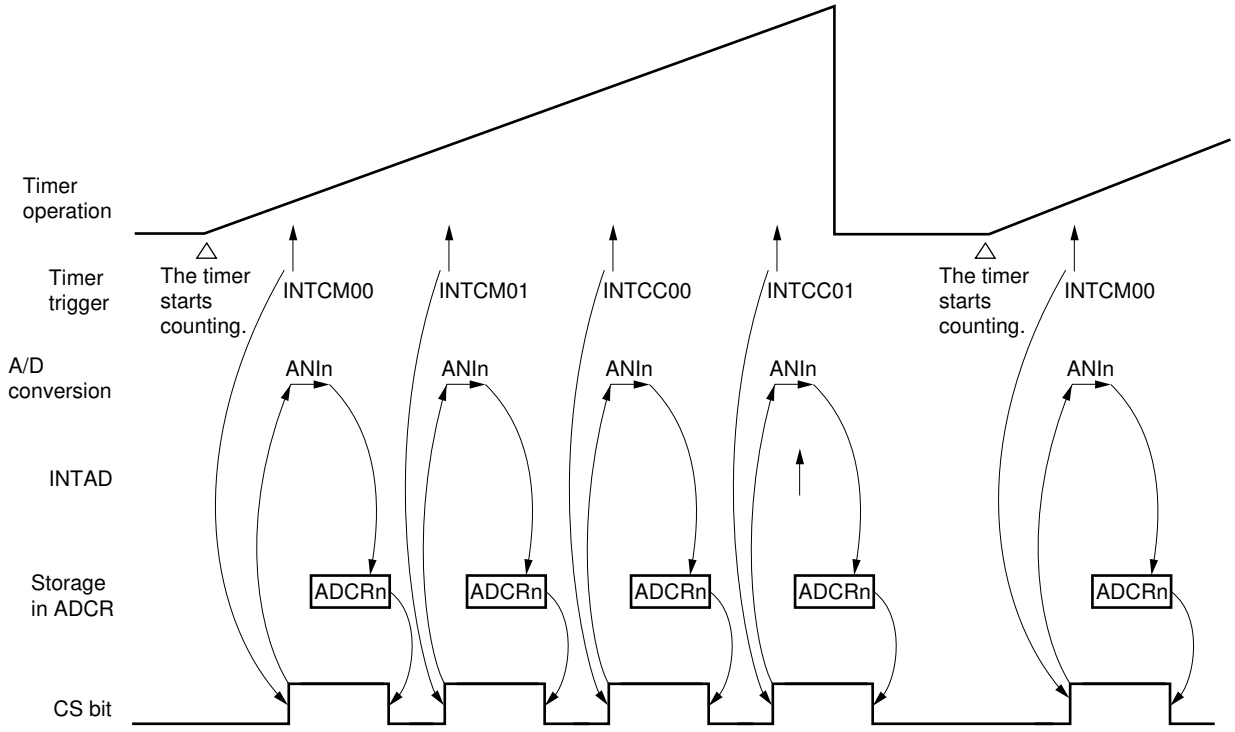
When timer 0 is in the one-shot mode and A/D conversion is performed four times, A/D conversion is terminated. Timer 0 need be restarted to restart A/D conversion. When the first match interrupt occurs after timer 0 is started, the CS bit is set to 1 to start A/D conversion.

When timer 0 is in the loop mode, A/D conversion is repeated each time a match interrupt occurs. Match interrupts may occur in any order.

Trigger signal	Analog input		A/D conversion result register
INTCM00	ANIn	->	ADCRn
INTCM01	ANIn	->	ADCRn
INTCC00	ANIn	->	ADCRn
INTCC01	ANIn	->	ADCRn

Remark n = 0 to 3

Fig. 8-20. Example 3 of A/D Conversion in the Timer Trigger Mode (Select Mode, One-Buffer Mode, Four-Trigger Mode, Timer One-Shot Mode, ANIn)

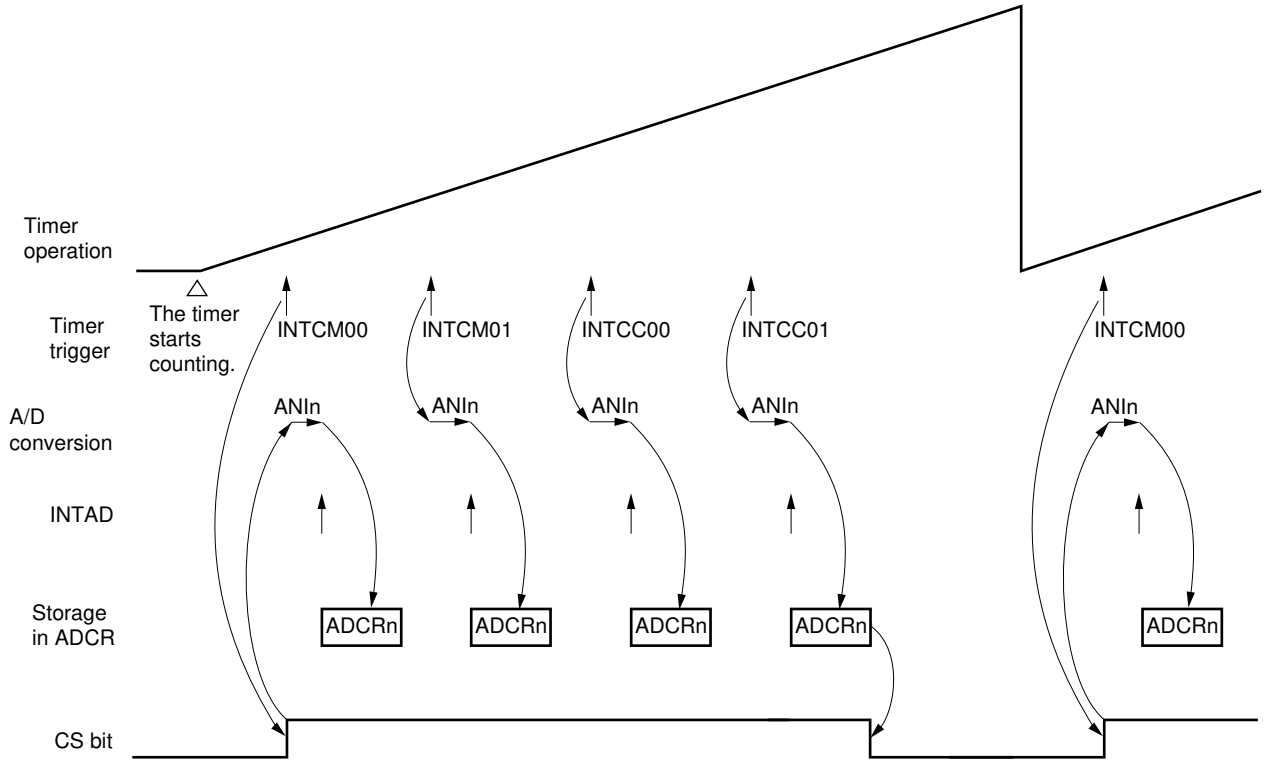


Remark n = 0 to 3

Cautions 1. When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

2. When several triggers are input simultaneously, the analog input of the analog input pin (ANIn) with the smallest number is converted. The other input triggers are ignored and the number of triggers is not counted. Therefore, the INTAD interrupt is not generated and A/D conversion results will be undefined.

Fig. 8-21. Example 4 of A/D Conversion in the Timer Trigger Mode (Select Mode, One-Buffer Mode, Four-Trigger Mode, Timer Loop Mode, ANIn)



Remark n = 0 to 3

Cautions 1. When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

2. When several triggers are input simultaneously, the analog input of the analog input pin (ANIn) with the smallest number is converted. The other input triggers are ignored and the number of triggers is not counted. Therefore, the INTAD interrupt is not generated and A/D conversion results will be undefined.

(ii) Four-buffer mode

Four A/D conversions are performed for one analog input, then the results are stored in four A/D conversion result registers (ADCR0 to ADCR3).

The four-buffer mode has the following two modes (one-trigger mode and four-trigger mode) corresponding to the number of triggers.

This mode is useful for finding the average of A/D conversion results.

<1> One-trigger mode

When a trigger signal (match interrupt (INTCM00)) is input, four A/D conversions are performed for one analog input, then the results are stored in four A/D conversion result registers.

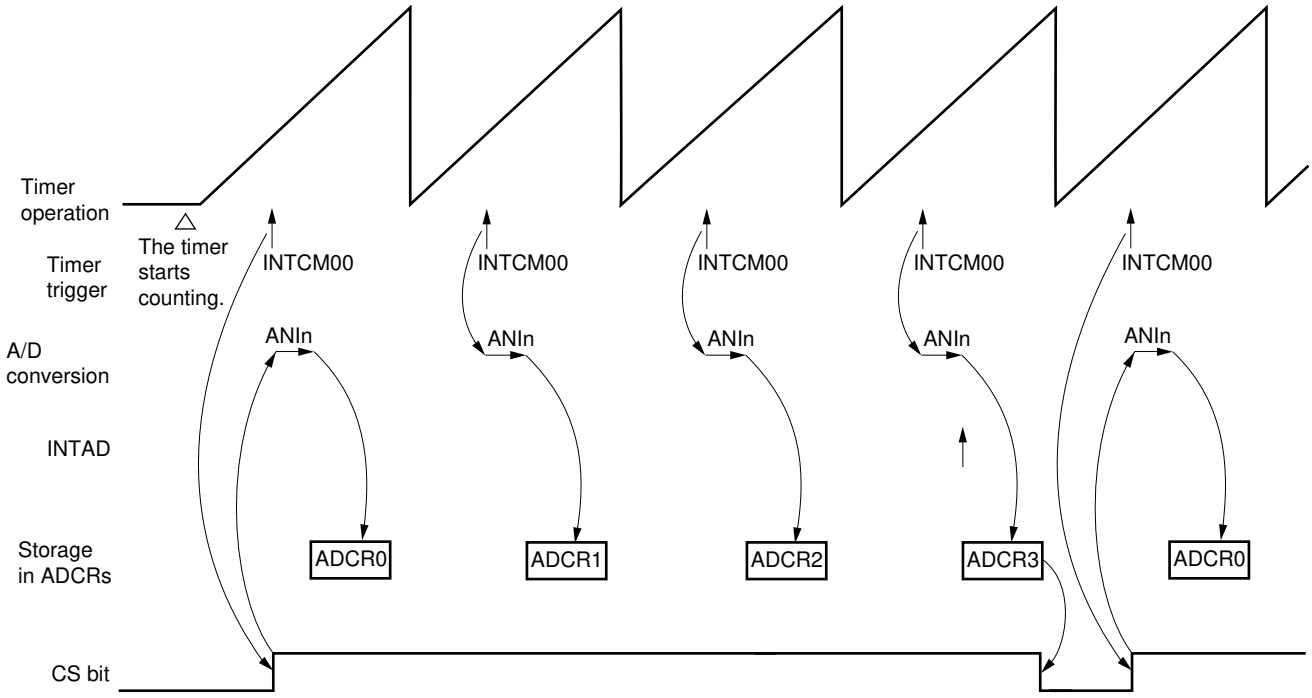
Upon completion of the four A/D conversions, an INTAD interrupt occurs and the A/D conversions are terminated (CS = 0).

A/D conversion is repeated each time an INTCM00 interrupt occurs.

Trigger signal	Analog input		A/D conversion result register
INTCM00	ANIn	->	ADCR0
INTCM00	ANIn	->	ADCR1
INTCM00	ANIn	->	ADCR2
INTCM00	ANIn	->	ADCR3

Remark n = 0 to 3

Fig. 8-22. Example 5 of A/D Conversion in the Timer Trigger Mode (Select Mode, Four-Buffer Mode, One-Trigger Mode, Timer Loop Mode, ANIn)



Remark n = 0 to 3

Caution In the one-shot mode for timer 0, only a single INTCM00 interrupt occurs. No A/D conversion is therefore terminated. To prevent this, be sure to put timer 0 in the loop mode.

<2> Four-trigger mode

When four trigger signals (four match interrupts (INTCM00, INTCM01, INTCC00, and INTCC01)) are input, four A/D conversions are performed for one analog input, and the results are stored in four A/D conversion result registers.

Upon completion of the four A/D conversions, an INTAD interrupt occurs, and A/D conversion is terminated (CS = 0).

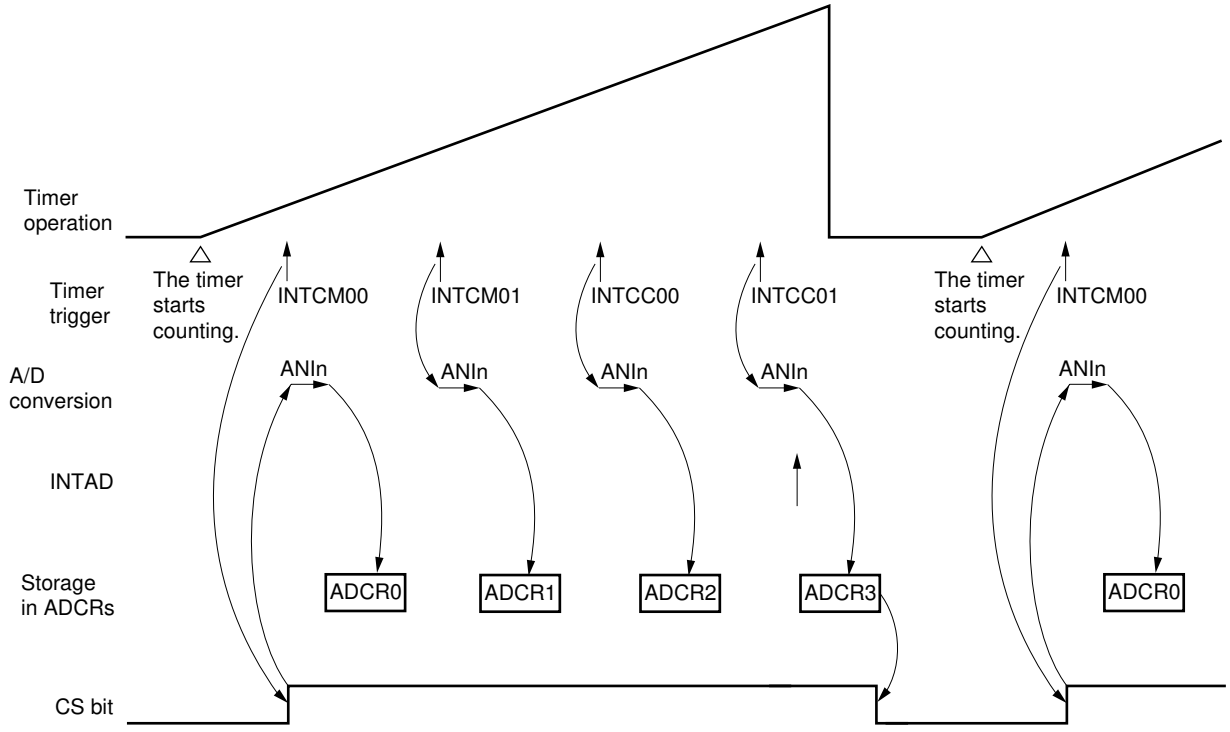
When timer 0 is in the one-shot mode and four A/D conversions are performed, A/D conversion is terminated. Timer 0 need be restarted to restart A/D conversion. When timer 0 is in the loop mode, A/D conversion is repeated each time a match interrupt occurs.

Match interrupts may occur in any order.

Trigger signal	Analog input		A/D conversion result register
INTCM00	ANIn	->	ADCR0
INTCM01	ANIn	->	ADCR1
INTCC00	ANIn	->	ADCR2
INTCC01	ANIn	->	ADCR3

Remark n = 0 to 3

Fig. 8-23. Example 6 of A/D Conversion in the Timer Trigger Mode (Select Mode, Four-Buffer Mode, Four-Trigger Mode, Timer One-Shot Mode, ANIn)

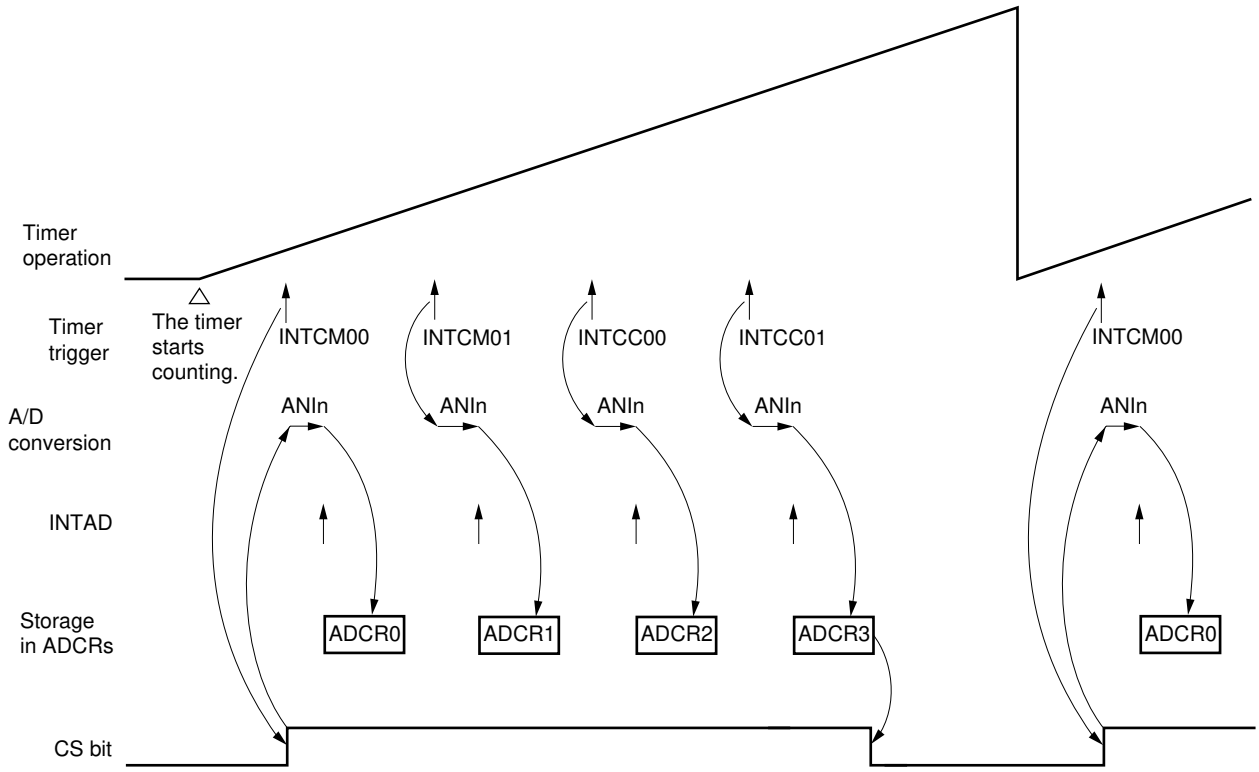


Remark n = 0 to 3

Cautions 1. When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

2. When several triggers are input simultaneously, the analog input of the analog input pin (ANIn) with the smallest number is converted. The other input triggers are ignored and the number of triggers is not counted. Therefore, the INTAD interrupt is not generated and A/D conversion results will be undefined.

Fig. 8-24. Example 7 of A/D Conversion in the Timer Trigger Mode (Select Mode, Four-Buffer Mode, Four-Trigger Mode, Timer Loop Mode, ANIn)



Remark n = 0 to 3

Cautions 1. When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

2. When several triggers are input simultaneously, the analog input of the analog input pin (ANIn) with the smallest number is converted. The other input triggers are ignored and the number of triggers is not counted. Therefore, the INTAD interrupt is not generated and A/D conversion results will be undefined.

(b) Scan mode

To perform A/D conversion, the scan mode sequentially selects analog inputs on the ANI0 pin to the analog input pin (ANIn: n = 0 to 7) specified by ADM0.

A/D conversion is performed for the lower four channels of analog inputs (ANI0 to ANI3) the specified number of times. When the real-time pulse unit (RPU) outputs a trigger signal, A/D conversion is performed for all four analog inputs starting with the first analog input on the ANI0 pin.

When ADM0 specifies that the lower four channels of analog inputs (ANI0 to ANI3) are scanned and A/D conversion is performed the specified number of times, an INTAD interrupt occurs and A/D conversion is terminated.

When ADM0 specifies that the higher four and lower four channels of analog inputs (ANI4 to ANI7) are scanned, the A/D converter enters the A/D trigger mode upon completion of A/D conversion for the lower four channels, then A/D conversion is performed for the remaining higher four channels.

The conversion results are stored in the A/D conversion result registers (ADCRn: n = 0 to 7) corresponding to the analog inputs.

Upon completion of A/D conversion for all the specified analog inputs, an INTAD interrupt occurs and A/D conversion is terminated (CS = 0).

The scan mode has the following two modes (one-trigger mode and four-trigger mode) corresponding to the number of triggers.

This mode is useful for monitoring multiple analog inputs at all times.

(i) One-trigger mode

When a trigger signal (match interrupt (INTCM00)) is input, four A/D conversions are performed for the lower analog inputs (ANI0 to ANI3), then the A/D converter enters the A/D trigger mode to continue A/D conversion.

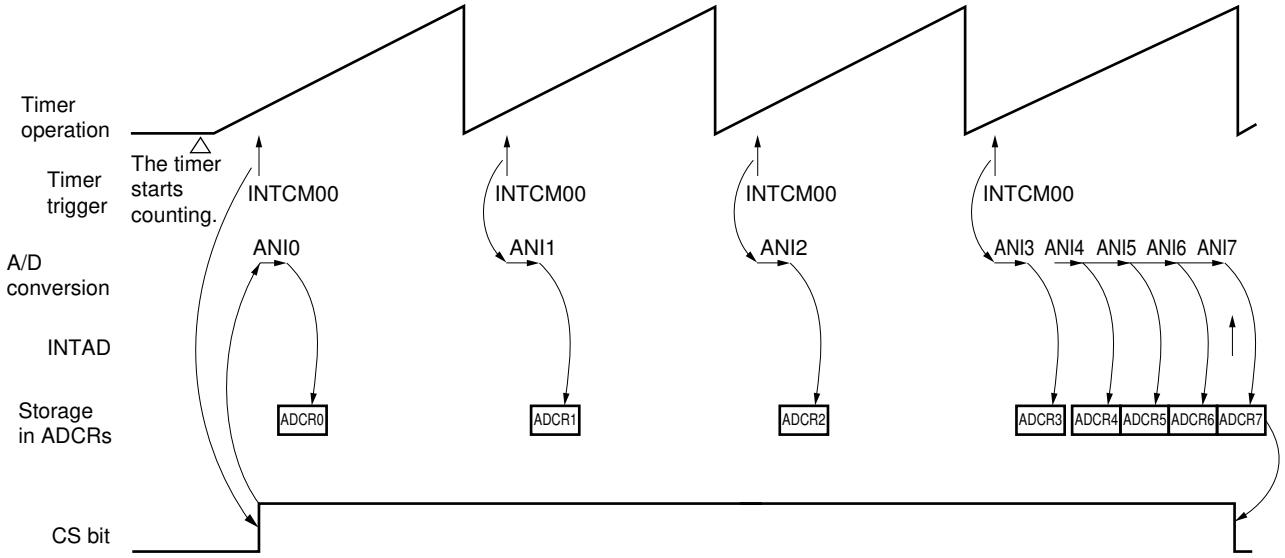
The analog inputs correspond to the A/D conversion result registers on a one-to-one basis.

Upon completion of all the specified A/D conversions, an INTAD interrupt occurs to terminate A/D conversion (CS = 0).

When an INTCM00 interrupt occurs upon completion of all the specified A/D conversions, however, A/D conversion is restarted.

Trigger signal	Analog input		A/D conversion result register
INTCM00	ANI0	->	ADCR0
INTCM00	ANI1	->	ADCR1
INTCM00	ANI2	->	ADCR2
INTCM00	ANI3	->	ADCR3
(A/D trigger mode)	ANI4	->	ADCR4
(A/D trigger mode)	ANI5	->	ADCR5
(A/D trigger mode)	ANI6	->	ADCR6
(A/D trigger mode)	ANI7	->	ADCR7

Fig. 8-25. Example 8 of A/D Conversion in the Timer Trigger Mode (Scan Mode, One-Trigger Mode, Timer Loop Mode, ANI0 to ANI7)



Cautions 1. In the one-shot mode for timer 0, only a single INTCM00 interrupt occurs. No A/D conversion is therefore terminated. To prevent this, be sure to put timer 0 in the loop mode.

2. When an INTCM00 interrupt occurs during A/D conversion of the analog inputs in the higher four channels (ANI4 to ANI7), the conversion is stopped and the ANI0 channel and subsequent channels are scanned again. The results of the stopped A/D conversions are undefined.

(ii) Four-trigger mode

When four trigger signals (four match interrupts (INTCM00, INTCM01, INTCC00, and INTCC01)) are input, four A/D conversions are performed for the low-order analog inputs (ANI0 to ANI3), then the A/D converter enters the A/D trigger mode to continue A/D conversion.

The analog inputs correspond to the A/D conversion result registers on a one-to-one basis.

Upon completion of all the specified A/D conversions, an INTAD interrupt occurs and A/D conversion is terminated (CS = 0).

When timer 0 is in the one-shot mode and all the specified A/D conversions are performed, A/D conversion is terminated. Timer 0 need be restarted to restart A/D conversion. When timer 0 is in the loop mode and all the specified A/D conversions are performed, a match interrupt occurs and A/D conversion is restarted.

Match interrupts may occur in any order. Because trigger signals correspond to analog inputs on a one-to-one basis, however, the order in which the compare registers generate match signals determines the order in which the analog inputs are scanned.

Trigger signal	Analog input		A/D conversion result register
INTCM00	ANI0	->	ADCR0
INTCM01	ANI1	->	ADCR1
INTCC00	ANI2	->	ADCR2
INTCC01	ANI3	->	ADCR3
(A/D trigger mode)	ANI4	->	ADCR4
(A/D trigger mode)	ANI5	->	ADCR5
(A/D trigger mode)	ANI6	->	ADCR6
(A/D trigger mode)	ANI7	->	ADCR7

Fig. 8-26. Example 9 of A/D Conversion in the Timer Trigger Mode (Scan Mode, Four-Trigger Mode, Timer One-Shot Mode, ANI0 to ANI7)

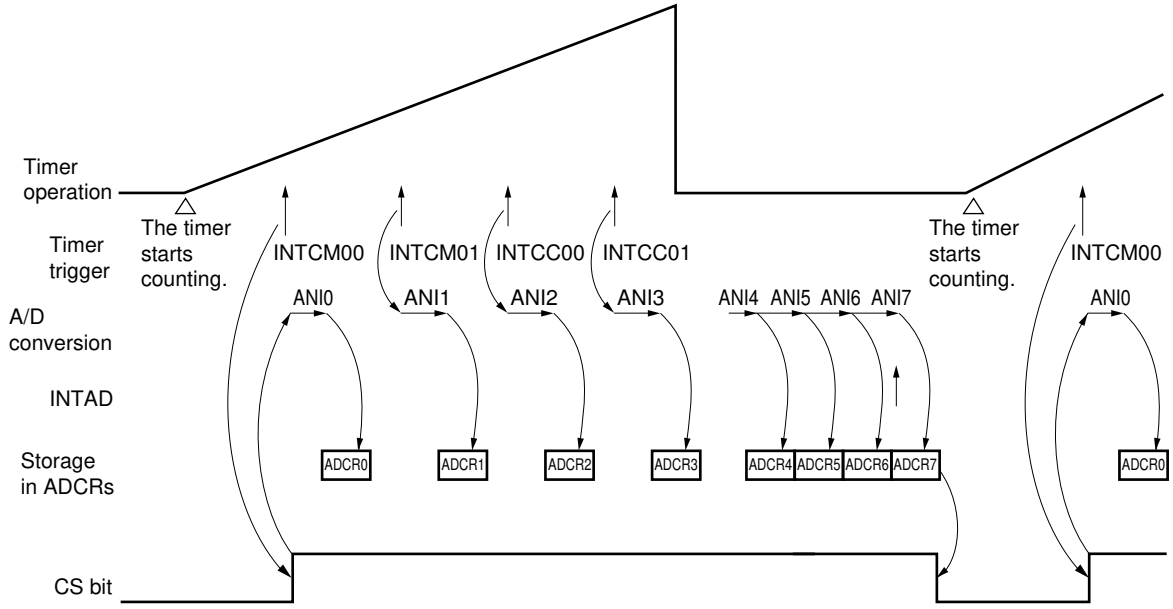
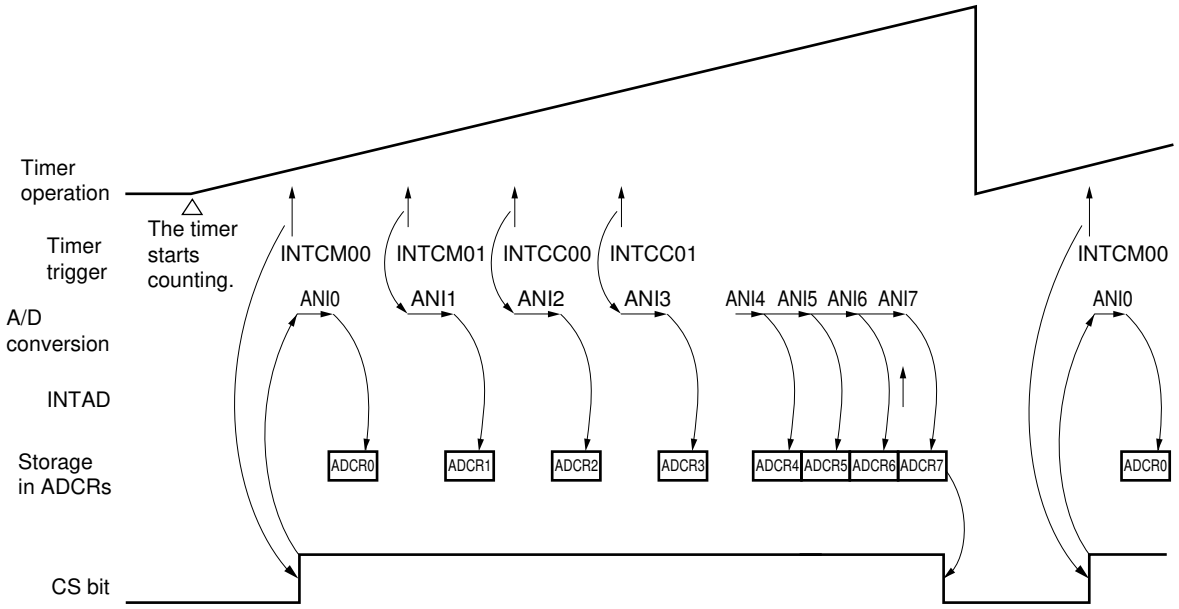


Fig. 8-27. Example 10 of A/D Conversion in the Timer Trigger Mode (Scan Mode, Four-Trigger Mode, Timer Loop Mode, ANI0 to ANI7)



Cautions 1. When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

2. When several triggers are input simultaneously, the analog input of the analog input pin (ANIn) with the smallest number is converted. The other input triggers are ignored and the number of triggers is not counted. Therefore, the INTAD interrupt is not generated and A/D conversion results will be undefined.

(3) External trigger mode

A/D conversion is performed for analog inputs (ANI0 to ANI3) at the timing of external trigger inputs (ADTRG0 to ADTRG3).

The external trigger inputs (ADTRG0 to ADTRG3) are also used as port 0 (P00 to P03). To set port 0 to the external trigger input mode, set the corresponding bit of the port 0 mode control register (PMC0) to set (1).

A/D converter mode register 1 (ADM1) specifies either a four-trigger mode using the ADTRG0 to ADTRG3 pins or a one-trigger mode using the ADTRG0 pin only.

Cautions 1. The external trigger mode is effective for analog inputs ANI0 to ANI3 only.

2. External triggers may be generated when bits 0 to 3 (PMC00 to PMC03) of the PMC0 register are rewritten from 0 to 1 or 1 to 0 during A/D conversion in the external trigger mode. Therefore, rewrite these bits after stopping the operations of the A/D converter.

(a) Select mode

A/D conversion is performed for a single analog input (ANIn: n = 0 to 3) specified by A/D converter mode register 0 (ADM0). The conversion result is stored in the A/D conversion result register (ADCRn: n = 0 to 3) corresponding to the analog input.

The select mode has the following two modes (one-buffer mode and four-buffer mode) for storing the A/D conversion result in the registers.

(i) One-buffer mode

One A/D conversion is performed for one analog input, then the result is stored in one A/D conversion result register. In this case, an analog input corresponds to an A/D conversion result register on a one-to-one basis.

The one-buffer mode has the following two modes (one-trigger mode and four-trigger mode) corresponding to the number of triggers.

This mode is useful for reading the A/D conversion result each time an A/D conversion is performed.

<1> One-trigger mode

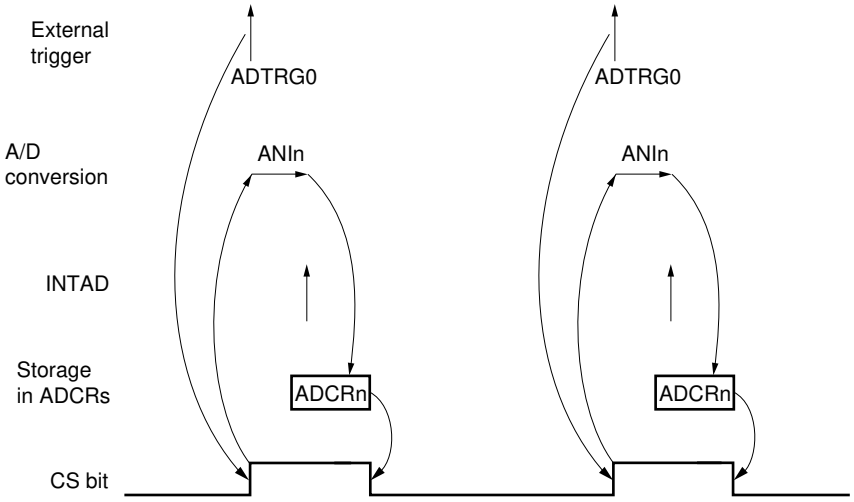
When a trigger signal (external trigger input (ADTRG0)) is input, one A/D conversion is performed for one analog input, then the result is stored in one A/D conversion result register. An INTAD interrupt occurs each time an A/D conversion is performed, then the A/D conversion is terminated (CS = 0).

A/D conversion is repeated each time the ADTRG0 signal is input.

Trigger signal	Analog input		A/D conversion result register
ADTRG0	ANIn	->	ADCRn

Remark n = 0 to 3

Fig. 8-28. Example 1 of A/D Conversion in the External Trigger Mode (Select Mode, One-Buffer Mode, One-Trigger Mode, ANIn)



Remark n = 0 to 3

Caution When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

<2> Four-trigger mode

When four external trigger signals (ADTRG0 to ADTRG3) are input, four A/D conversions are performed for one analog input, and the result is stored in one A/D conversion result register. Each time an A/D conversion is performed, an INTAD interrupt occurs, and the CS bit is reset to 0. A/D conversion is terminated when four A/D conversions have been completed.

The result of a single A/D conversion is held in the A/D conversion result register until the next A/D conversion is terminated. The conversion result must be loaded into memory when an INTAD interrupt occurs upon completion of the single A/D conversion.

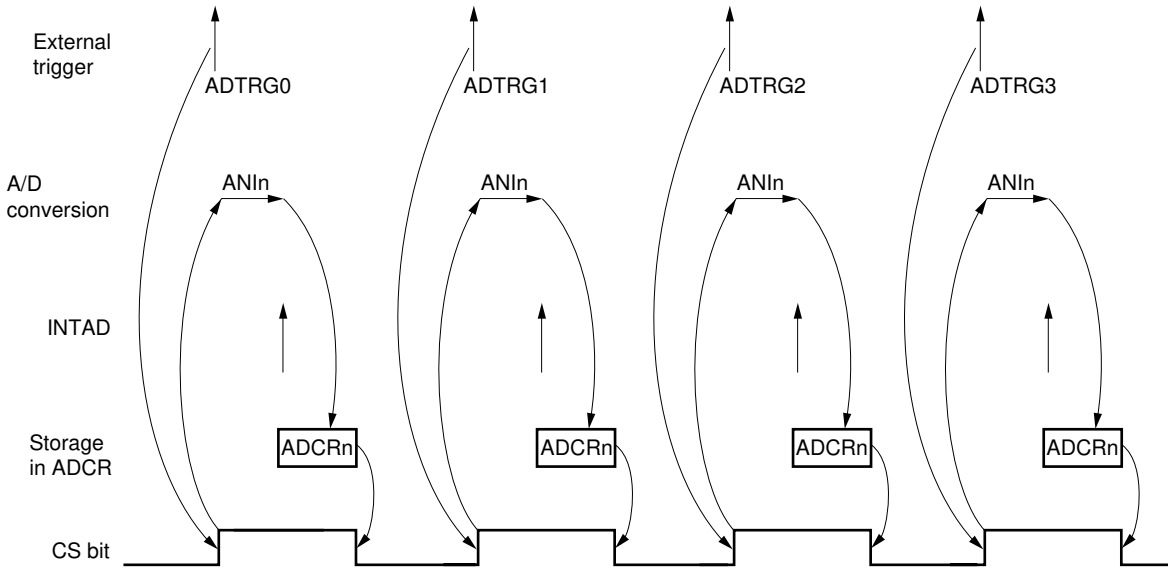
A/D conversion is repeated each time an external trigger signal is input.

External trigger signals may be generated in any order.

Trigger signal	Analog input		A/D conversion result register
ADTRG0	ANIn	->	ADCRn
ADTRG1	ANIn	->	ADCRn
ADTRG2	ANIn	->	ADCRn
ADTRG3	ANIn	->	ADCRn

Remark n = 0 to 3

Fig. 8-29. Example 2 of A/D Conversion in the External Trigger Mode (Select Mode, One-Buffer Mode, Four-Trigger Mode, ANIn)



Remark n = 0 to 3

Cautions 1. When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

2. When several triggers are input simultaneously, the analog input of the analog input pin (ANIn) with the smallest number is converted. The other input triggers are ignored and the number of triggers is not counted. Therefore, the INTAD interrupt is not generated and A/D conversion results will be undefined.

(ii) Four-buffer mode

Four A/D conversions are performed for one analog input, then the results are stored in four A/D conversion result registers (ADCR0 to ADCR3).

The four-buffer mode has the following two modes (one-trigger mode and four-trigger mode) corresponding to the number of triggers.

This mode is useful for finding the average of A/D conversion results.

<1> One-trigger mode

When an external trigger signal (ADTRG0) is input, four A/D conversions are performed for one analog input, then the results are stored in four A/D conversion result registers.

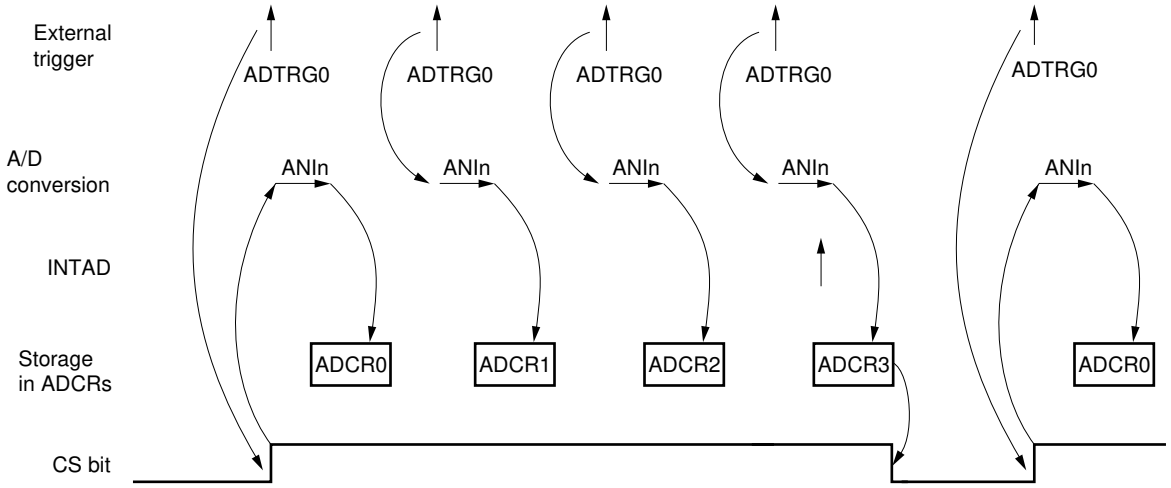
Upon completion of the four A/D conversions, an INTAD interrupt occurs and the A/D conversions are terminated (CS = 0).

A/D conversion is repeated each time an ADTRG0 signal is input.

Trigger signal	Analog input		A/D conversion result register
ADTRG0	ANIn	->	ADCR0
ADTRG0	ANIn	->	ADCR1
ADTRG0	ANIn	->	ADCR2
ADTRG0	ANIn	->	ADCR3

Remark n = 0 to 3

Fig. 8-30. Example 3 of A/D Conversion in the External Trigger Mode (Select Mode, Four-Buffer Mode, One-Trigger Mode, ANIn)



Remark n = 0 to 3

<2> Four-trigger mode

When four trigger signals (ADTRG0 to ADTRG3)) are input, four A/D conversions are performed for one analog input, and the results are stored in four A/D conversion result registers.

Upon completion of the four A/D conversions, an INTAD interrupt occurs, and A/D conversion is terminated (CS = 0).

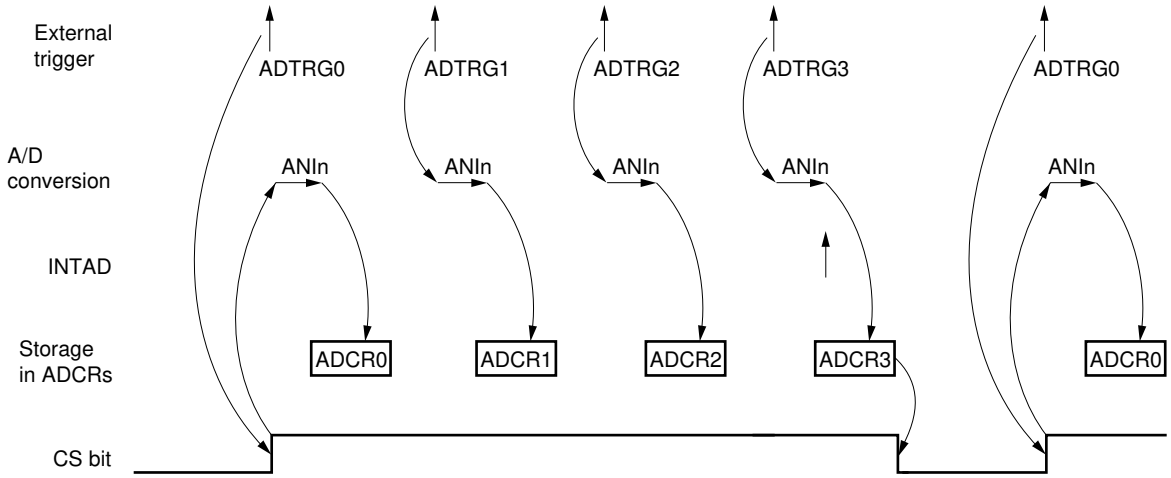
A/D conversion is repeated each time an external trigger signal is input.

External trigger signals may be generated in any order.

Trigger signal	Analog input		A/D conversion result register
ADTRG0	ANIn	->	ADCR0
ADTRG1	ANIn	->	ADCR1
ADTRG2	ANIn	->	ADCR2
ADTRG3	ANIn	->	ADCR3

Remark n = 0 to 3

Fig. 8-31. Example 4 of A/D Conversion in the External Trigger Mode (Select Mode, Four-Buffer Mode, Four-Trigger Mode, ANIn)



Remark n = 0 to 3

Cautions 1. When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

2. When several triggers are input simultaneously, the analog input of the analog input pin (ANIn) with the smallest number is converted. The other input triggers are ignored and the number of triggers is not counted. Therefore, the INTAD interrupt is not generated and A/D conversion results will be undefined.

(b) Scan mode

To perform A/D conversion, the scan mode sequentially selects analog inputs on the ANI0 pin to the analog input pin (ANIn: n = 0 to 7) specified by ADM0.

A/D conversion is performed for the lower four channels of analog inputs (ANI0 to ANI3) the specified number of times.

When ADM0 specifies that the lower four channels of analog inputs (ANI0 to ANI3) are scanned and A/D conversion is performed the specified number of times, an INTAD interrupt occurs and A/D conversion is terminated.

When ADM0 specifies that the higher four and lower four channels of analog inputs (ANI0 to ANI7) are scanned, the A/D converter enters the A/D trigger mode on completion of conversion for the lower four channels, then A/D conversion is performed for the remaining higher four channels.

The conversion results are stored in the A/D conversion result registers (ADCRn: n = 0 to 7) corresponding to the analog inputs.

Upon completion of A/D conversion of all the specified analog inputs, an INTAD interrupt occurs and A/D conversion is terminated (CS = 0).

The scan mode has the following two modes (one-trigger mode and four-trigger mode) corresponding to the number of triggers.

This mode is useful for monitoring multiple analog inputs.

(i) One-trigger mode

When an external trigger signal (ADTRG0) is input, four A/D conversions are performed for the low-order analog inputs (ANI0 to ANI3), then the A/D converter enters the A/D trigger mode to continue A/D conversion.

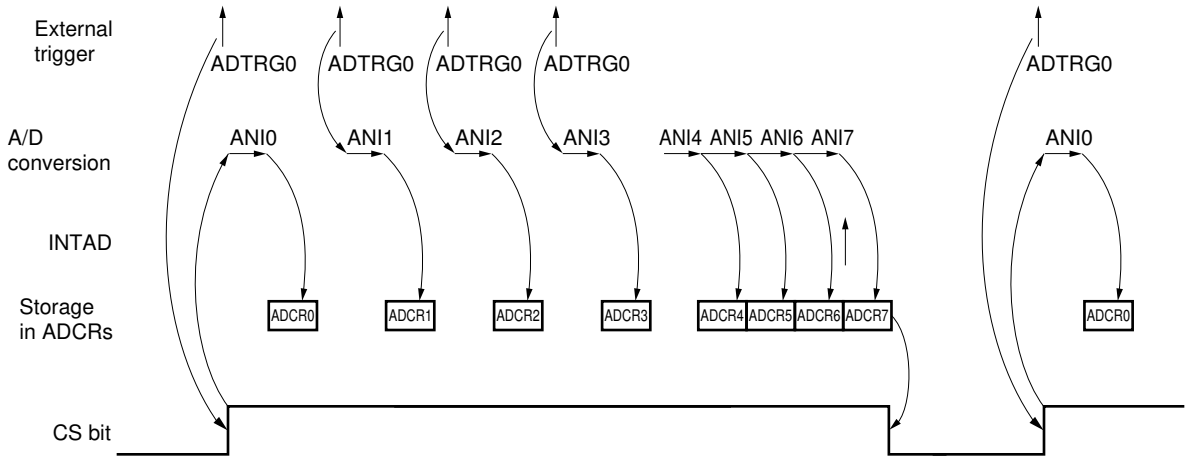
The analog inputs correspond to the A/D conversion result registers on a one-to-one basis.

Upon completion of all the specified A/D conversions, an INTAD interrupt occurs and A/D conversion is terminated (CS = 0).

When an ADTRG0 signal is input upon completion of all the specified A/D conversions, A/D conversion is restarted.

Trigger signal	Analog input		A/D conversion result register
ADTRG0	ANI0	->	ADCR0
ADTRG0	ANI1	->	ADCR1
ADTRG0	ANI2	->	ADCR2
ADTRG0	ANI3	->	ADCR3
(A/D trigger mode)	ANI4	->	ADCR4
(A/D trigger mode)	ANI5	->	ADCR5
(A/D trigger mode)	ANI6	->	ADCR6
(A/D trigger mode)	ANI7	->	ADCR7

Fig. 8-32. Example 5 of A/D Conversion in the External Trigger Mode (Scan Mode, One-Trigger Mode, ANI0 to ANI7)



Cautions 1. When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

2. When an ADTRG0 signal is input during A/D conversion of the analog inputs in the higher four channels (ANI4 to ANI7), the conversion is stopped and the ANI0 channel and subsequent channels are scanned again. The results of the stopped A/D conversions are undefined.

(ii) Four-trigger mode

When four trigger signals (ADTRG0 to ADTRG3) are input, four A/D conversions are performed for the low-order analog inputs (ANI0 to ANI3), then the A/D converter enters the A/D trigger mode to continue A/D conversion.

The analog inputs correspond to the A/D conversion result registers on a one-to-one basis.

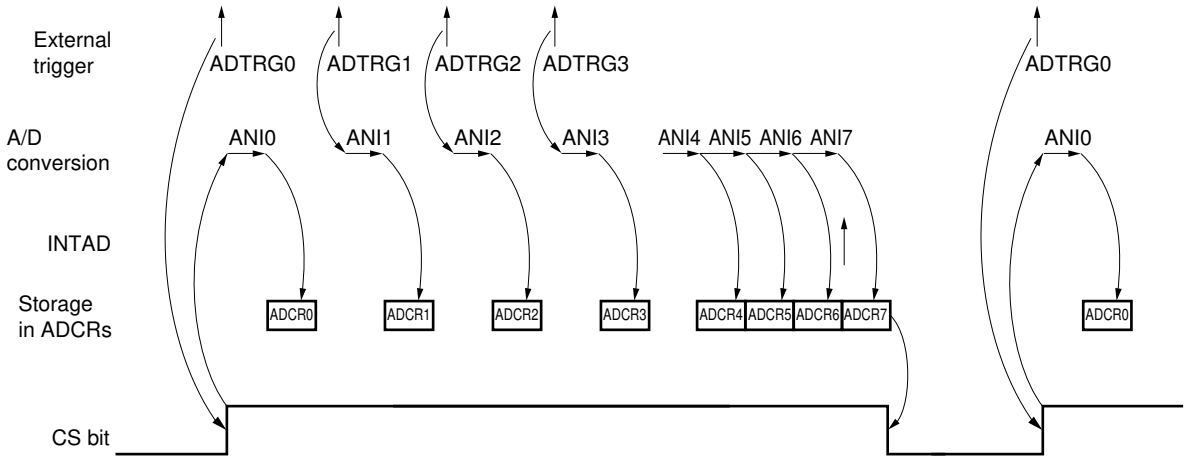
Upon completion of all the specified A/D conversions, an INTAD interrupt occurs and A/D conversion is terminated (CS = 0).

When an external trigger signal is input after all the specified A/D conversions are performed, A/D conversion is restarted.

External trigger signals may be generated in any order. Because external trigger signals correspond to analog inputs on a one-to-one basis, however, the order in which the external trigger signals are input determines the order in which the analog inputs are scanned.

Trigger signal	Analog input		A/D conversion result register
ADTRG0	ANI0	->	ADCR0
ADTRG1	ANI1	->	ADCR1
ADTRG2	ANI2	->	ADCR2
ADTRG3	ANI3	->	ADCR3
(A/D trigger mode)	ANI4	->	ADCR4
(A/D trigger mode)	ANI5	->	ADCR5
(A/D trigger mode)	ANI6	->	ADCR6
(A/D trigger mode)	ANI7	->	ADCR7

Fig. 8-33. Example 6 of A/D Conversion in the External Trigger Mode (Scan Mode, Four-Trigger Mode, ANI0 to ANI7)



Cautions 1. When the next trigger signal is input during A/D conversion, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined.

2. When several triggers are input simultaneously, the analog input of the analog input pin (ANIn) with the smallest number is converted. The other input triggers are ignored and the number of triggers is not counted. Therefore, the INTAD interrupt is not generated and A/D conversion results will be undefined.

3. When the ADTRG0 to ADTRG3 signals are input during A/D conversion of analog inputs in the higher four channels (ANI4 to ANI7), the conversion is stopped and the ANI0 channel and subsequent channels are scanned again. The results of the stopped A/D conversions are undefined.

8.6 Glossary for A/D Converter Characteristics Graphs

This glossary contains definition of terms used in explaining A/D converter characteristics graphs.

(1) Resolution

A ratio of an acceptable minimum analog input voltage, that is, an analog input to a 1-bit digital output is called 1 LSB (least significant bit). The ratio of one LSB to the full scale is represented in %FSR (full-scale range).

For 10 bits,

$1LSB = 1/2^{10} = 1/1024$
$= 0.0098$ %FSR

Accuracy is not related to resolution, but depends on a combined error.

(2) Combined error

The combined error represents the maximum difference between an actual value and a theoretical value. The combined error represents the error caused by combining any two of these errors: a zero-scale error, full-scale error, and non-linearity error.
The combined error in the characteristics graph contains no quantization error.

(3) Quantization error

This is an error generated during analog-to-digital conversion. It occurs because an analog-to-digital converter can be no more accurate than ±1/2 LSB because of its resolution.
The combined error, zero-scale error, full-scale error, and non-linearity error in the characteristics graphs contain no quantization error.

Fig. 8-34. Combined Error

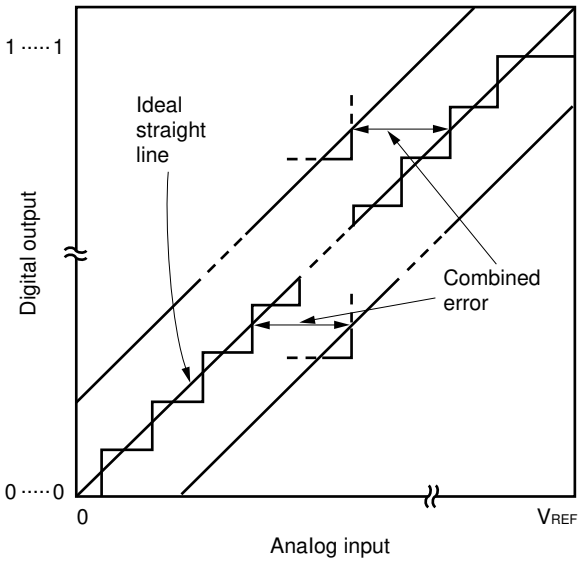
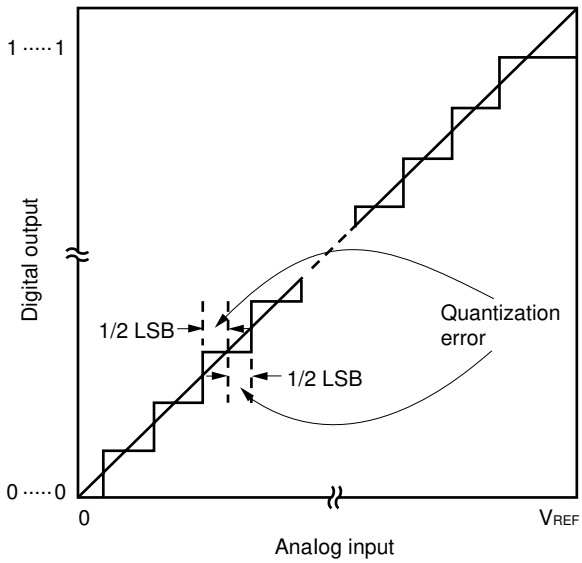


Fig. 8-35. Quantization Error



(4) Zero-scale error

The zero-scale error represents the difference between the actual analog input voltage and theoretical analog input voltage (1/2 LSB) when the digital output varies from 0·····000 to 0·····001. When the actual value is greater than the theoretical value, the zero-scale error represents the difference between the actual analog input voltage and theoretical analog input voltage (3/2 LSB) when the digital output varies from 0·····001 to 0·····010.

(5) Full-scale error

The full-scale error represents the difference between the actual analog input voltage and theoretical analog input voltage (full-scale minus 3/2 LSB) when the digital output varies from 1·····110 to 1·····111.

(6) Non-linearity error

The non-linearity error indicates that the conversion characteristics does not form an ideal straight line. The non-linearity error represents the maximum difference between the actual and ideal values when the zero-scale and full-scale errors are 0.

Fig. 8-36. Zero-Scale Error

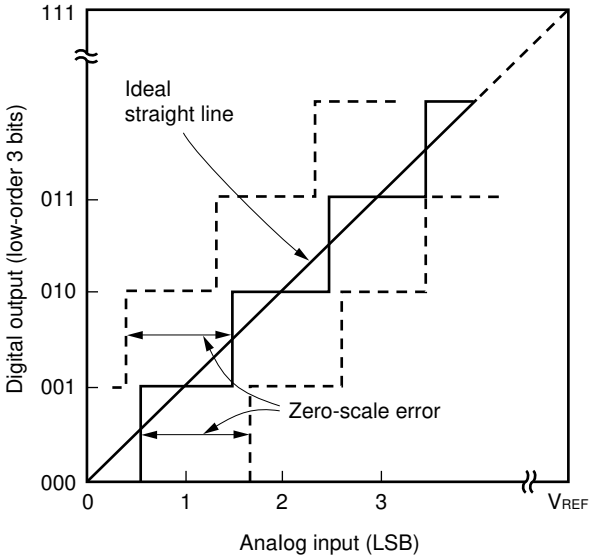


Fig. 8-37. Full-Scale Error

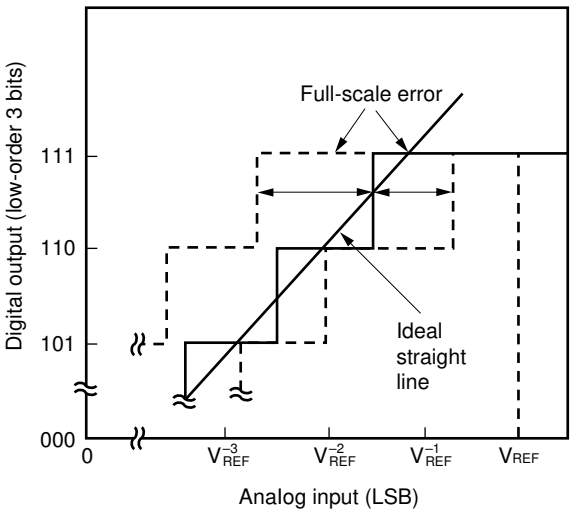
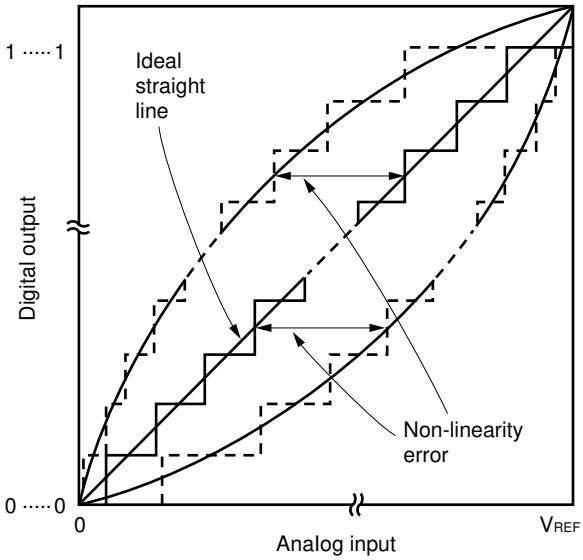


Fig. 8-38. Non-linearity Error



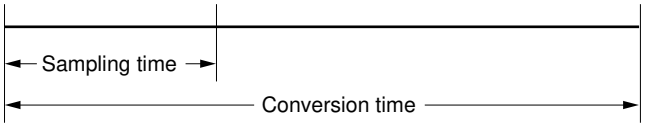
(7) Conversion time

Represents the time required until digital output is obtained after analog input voltage is applied.

The conversion time in the characteristics tables includes sampling time.

(8) Sampling time

Time required for the sample-and-hold circuit to accept the analog voltage while an analog switch is on.



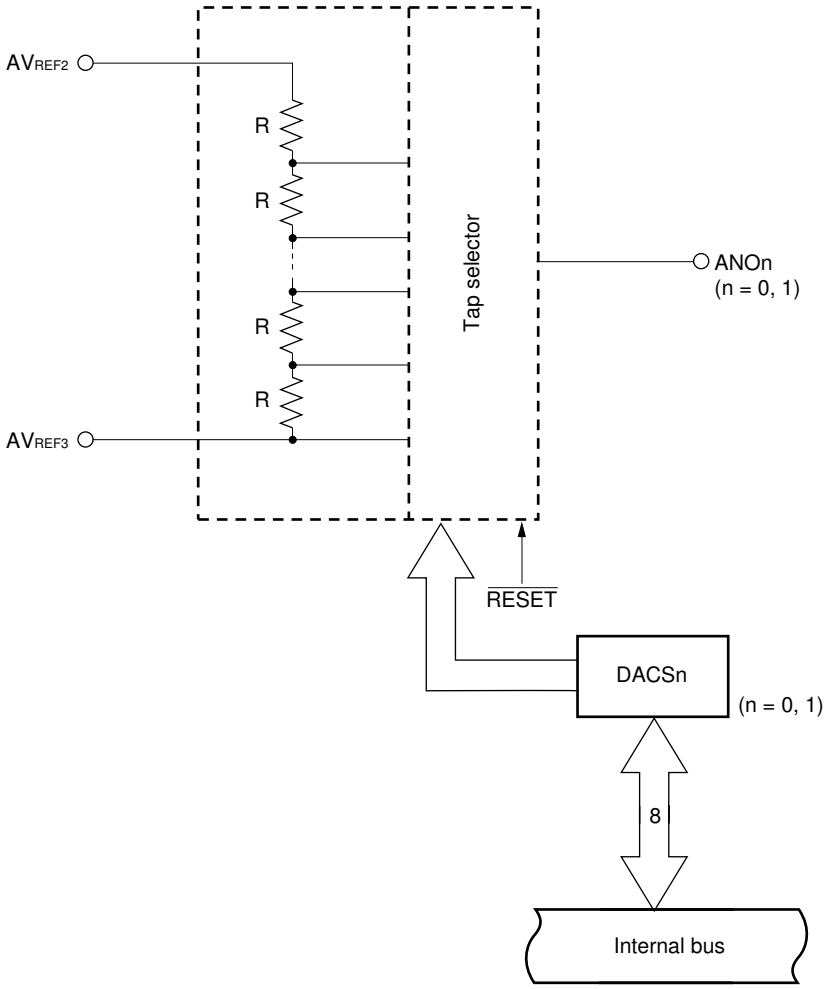
[MEMO]

The μPD78356 contains two 8-bit voltage output digital/analog (D/A) converters. The converters convert digital inputs using a resistor string system.

9.1 Configuration

Fig. 9-1 shows the configuration of a D/A converter.

Fig. 9-1. Block Diagram of a D/A Converter (n = 0, 1)



(1) D/A converted data coefficient register (DACSn: n = 0 or 1)

DACSn sets the voltage output to the ANOn pin (n = 0 or 1). Byte access or word access to the register is possible. The voltage output to the ANOn pin is calculated by the following formula:

$$ANOn = \frac{AV_{REF2} - AV_{REF3}}{256} \times DACSn + AV_{REF3} \text{ [V]}$$

When a $\overline{\text{RESET}}$ signal is input, DACSn is set to 00H.

(2) Resistor string

In the D/A converter, 256 resistors with the same resistance are serially connected. The resistors at the ends of the resistor string are connected to the AV_{REF2} and AV_{REF3} pins, respectively. The μPD78356 contains two resistor strings one for the ANO0 pin, and one for the ANO1 pin.

(3) Tap selector

The tap selector selects one of 256 taps of the resistor string according to the value of DACSn and connects the selected tap with the ANOn pin.

When the $\overline{\text{RESET}}$ signal input is low (0), no tap is connected to the ANOn pin, namely the output is set to high-impedance state.

9.2 Operation of the D/A Converters

As soon as a value is written in the D/A converted data coefficient register (DACSn: n = 0 or 1), an analog voltage corresponding to the written value is output to the ANOn pin (n = 0 or 1). The output voltage is held until a new value is written in DACSn (n = 0 or 1).

The voltage output to the ANOn pin is calculated by the following formula:

$$ANOn = \frac{AV_{REF2} - AV_{REF3}}{256} \times DACSn + AV_{REF3} \text{ [V]}$$

While the $\overline{\text{RESET}}$ signal input is low, the ANOn output is in high-impedance state and the contents of the DACSn is 00H. After the $\overline{\text{RESET}}$ signal input is released, the ANOn pin outputs the voltage whose level is the same as the AV_{REF3} pin.

9.3 Cautions

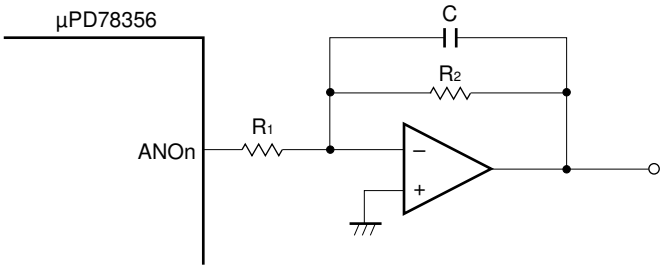
(1) The ANOn pin (n = 0 or 1) cannot source a current due to the high output impedance of the D/A converters. When connecting a load with a low input impedance, insert a buffer amplifier between the load and ANOn pin. Keep the wire to the buffer amplifier or load as short as possible (because the output impedance is high). If this is impossible, surround the wire with the ground pattern.

(2) The output voltage of each D/A converter changes step by step. In general, use the signal output from a D/A converter after passing it through a low-pass filter.

(3) The D/A converters contained in the μPD78356 have high-impedance output while the $\overline{\text{RESET}}$ signal input is low. Arrange the load circuit so that it can accept high-impedance input.

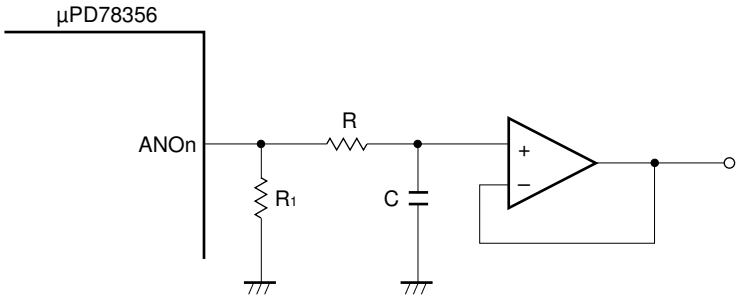
Fig. 9-2. Examples of Buffer Amplifier Connection

(a) Inverting amplifier



- The input impedance of the buffer amplifier is equal to $R_1 + R_2$.

(b) Voltage follower



- The input impedance of the buffer amplifier is equal to R_1.
- Be sure to connect R_1. Otherwise, the output becomes undefined while the $\overline{\text{RESET}}$ signal is asserted.

(4) After a reset is released, the D/A converter outputs the voltage whose level is the same as the AV_{REF3} pin. Arrange the D/A converter output so that the converter outputs the same voltage as the voltage on the AV_{REF3} pin.

[MEMO]

CHAPTER 10 ASYNCHRONOUS SERIAL INTERFACE

The μPD78356 contains the universal asynchronous receiver transmitter (UART) that serves as an asynchronous serial interface. The interface transmits and receives 1-byte serial data following a start bit. Full-duplex operation is possible. In addition, the baud rate generator provided in the μPD78356 allows communication at a wide range of baud rates.

The interface operates independently of clock synchronous serial interfaces with and without the pin switching function.

10.1 Configuration of Asynchronous Serial Interface

The asynchronous serial interface is controlled by the asynchronous serial interface mode register (ASIM) and asynchronous serial interface status register (ASIS). Received data is retained in the receive buffer (RXB), and data to be transmitted is written in the shift register for transmission (TXS).

Fig. 10-1 shows the configuration of the asynchronous serial interface.

(1) Asynchronous serial interface mode register (ASIM)

The ASIM register is an 8-bit register that controls the asynchronous serial interface operation.
Both data reading and writing are possible with 8-bit and single-bit manipulation instructions.
When the $\overline{\text{RESET}}$ signal is input, the ASIM register is set to 80H.

(2) Asynchronous Serial Interface Status Register (ASIS)

The ASIS register is an aggregate of flags that indicate the presence of an error in received data. If an error is detected in one portion of the received data, the flag associated with that portion is set to 1. The flag is reset to 0 when data is read from the receive buffer (RXB) or the next data portion is received. (If the newly received data has an error, then the flag associated with that portion is set to 1.)
Only data writing is possible with 8-bit and single-bit manipulation instructions.
When the $\overline{\text{RESET}}$ signal is input, the ASIS register is set to 00H.

(3) Parity check for reception control

Reception control is executed according to the contents of the ASIM register. Received data is checked for parity errors. If an error is detected, the value corresponding to the error is set in the ASIS register.

(4) Shift register for reception

The shift register for reception converts the serial data received at the RxD pin to parallel data. Data is transferred to the receive buffer each time one byte of data has been received by the register.
Note that the shift register for reception is not subject to direct control by the CPU.

(5) Receive buffer (RXB)

The receive buffer (RXB) retains data received from the shift register in 1-byte blocks.

When the data length is seven bits, received data is transferred to bits 0 to 6 of the RXB and 0 is always set in the MSB section.

Only data writing is possible with 8-bit manipulation instructions. When the $\overline{\text{RESET}}$ signal is input, the contents of the buffer become undefined.

(6) Shift register for transmission (TXS)

The shift register for transmission (TXS) sets the data to be transmitted. The data written in this register is transmitted as serial data.

When the data length is seven bits, bits 0 to 6 of the TXS are to be transmitted. When data is written in the TXS, transmission begins. Data must not be written in the TXS during transmission.

Only data writing is possible with 8-bit manipulation instructions. When the $\overline{\text{RESET}}$ signal is input, the contents of the TXS become undefined.

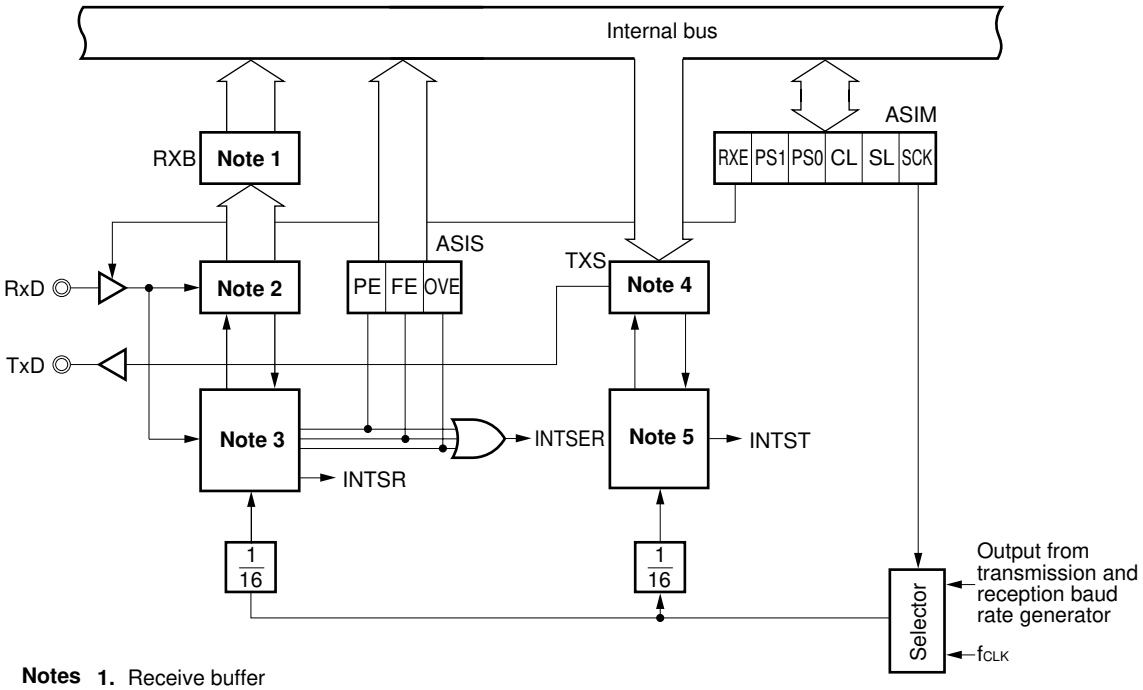
(7) Parity bit addition for transmission control

Transmission control is executed by automatically adding a start, parity or stop bit to the data written in the TXS register. Which bit is added depends on the contents of the ASIM register.

(8) Selector

The selector selects the clock source for baud rate generation.

Fig. 10-1. Block Diagram of the Asynchronous Serial Interface



Notes
1. Receive buffer
2. Shift register for reception
3. Parity check for reception control
4. Shift register for transmission
5. Parity bit addition for transmission control

10.2 Setting Pins for Serial Transmission

The RxD and TxD pins, which are also used as general purpose ports, must be set to the control mode before beginning serial transmission.

(1) Setting pins for serial transmission

In the asynchronous serial interface, the TxD pin is used for data transmission and the RxD pin for data reception. These pins also serve as general purpose ports P30 and P31, respectively. Therefore, before beginning serial transmission, these pins must be set to the control mode in the port 3 mode control register (PMC3).

Fig. 10-2. Port 3 Mode Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PMC3	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	FF43H	00H	R/W

PMC30	Control mode setting for P30 pin
0	I/O port mode
1	TxD output mode

PMC31	Control mode setting for P31 pin
0	I/O port mode
1	RxD input mode

PMC32	Control mode setting for P32 pin
0	I/O port mode
1	SB0 input and output mode/SO00 output mode

PMC33	Control mode setting for P33 pin
0	I/O port mode
1	SB1 input and output mode/SI00 input mode

PMC34	Control mode setting for P34 pin
0	I/O port mode
1	$\overline{\text{SCK00}}$ input and output mode

PMC35	Control mode setting for P35 pin
0	I/O port mode
1	TCLR1 input mode

PMC36	Control mode setting for P36 pin
0	I/O port mode
1	TI1 input mode/TO11 output mode

PMC37	Control mode setting for P37 pin
0	I/O port mode
1	TO10 output mode

(2) Reading pin levels

When port 3 (P3) has been set to the control mode using the port 3 mode control register (PMC3), the read instructions of P3 can read the following information:

(a) TxD/P30 pin

- When bit 0 of the port 3 mode register (PM3) is set to 1, the TxD pin level can be read.
- When bit 0 of the port 3 mode register (PM3) is set to 0 (reset), the level of the internal transmitted data can be read.

(b) RxD/P31 pin

- The RxD pin level can be read only when bit 1 of the PM3 is set to 1.

Reading pin levels allows pins checks, including the TxD pin check for collision.
Writing data in P3 does not change the TxD or RxD pin level. (Data is written in the P3 output buffer.)

Fig. 10-3. Port 3 Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	FF23H	FFH	R/W

PM3n	I/O mode setting for P3n pin (n = 0 to 7)
0	Output mode (Output buffer is on.)
1	Input mode (Output buffer is off.)

10.3 Setting Data Format

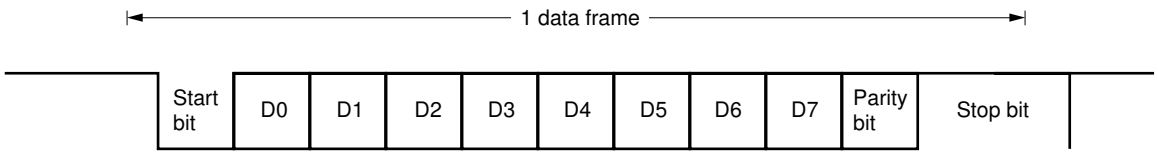
The character bit length, parity bit selection, and stop bit length can be set using the asynchronous serial interface mode register (ASIM).

(1) Setting data format

The transmit and receive data consists of a start bit, character bits, a parity bit, and a stop bit or bits per data frame, as shown in Fig. 10-4.

The character bit length, parity bit selection, and stop bit length can be set using the ASIM register (see **Fig. 10-5**).

Fig. 10-4. Data Format of the Asynchronous Serial Interface



- Start bit ... 1 bit
- Characters bits ... 7 or 8 bits
- Parity bit ... Even parity/odd parity/zero parity/no parity
- Stop bits ... 1 or 2 bits

(2) Zero parity - useful option for starting up the system

The μPD78356 includes zero parity in the parity options it offers for the asynchronous serial interface. When zero parity is selected, a 0 is added to the serial data to be transmitted. When receiving, data is accepted regardless of the status of the parity bit, and parity errors never occur.

Zero parity is useful in performing serial transmission in a situation, like system start-up, where a specific data format is yet to be defined.

Fig. 10-5. Setting of ASIM Register (Data Format)

Symbol	7	<6>	5	4	3	2	1	0	Address	When reset	R/W
ASIM	1	RXE	PS1	PS0	CL	SL	0	SCK	FF88H	80H	R/W

SCK	Setting of serial clock
0	Internal baud rate generator output
1	Internal clock f_{CLK}

SL	Setting of stop bit length
0	1 bit
1	2 bits

CL	Setting of character length
0	7 bits
1	8 bits

PS1	PS0	Setting of parity bit
0	0	No parity
0	1	Transmission: A 0 will be added as a parity bit. Reception : A parity error will not be detected.
1	0	Odd parity
1	1	Even parity

RXE	Reception status
0	Reception disabled
1	Reception enable

Remark f_{CLK}: Internal system clock

10.4 Setting Baud Rate

Either the baud rate generator output or the internal clock f_{CLK} can be selected as the serial clock. The baud rate generator output can be set at any desired baud rate, independently of the operation clock frequency.

(1) Baud rate equal to serial clock/16

In the asynchronous serial interface, the signals on the RxD pin are sampled by the clock whose frequency is 1/16 of the serial clock specified in the asynchronous serial interface mode register (ASIM).

(2) Selecting serial clock

The SCK bit of the ASIM register is used for serial clock selection. (See **Fig. 10-6**.)

Selecting the internal system clock f_{CLK} sets the baud rate at $f_{CLK}/16$. Therefore, when the internal system clock is 16 MHz, the baud rate is 1 Mbps.

Remark The baud rate generator is also used for clock synchronous serial interfaces with and without the pin switching function. (See **11.3** and **12.4**.)

Fig. 10-6. Setting of ASIM Register (Serial Clock)

Symbol	7	<6>	5	4	3	2	1	0	Address	When reset	R/W
ASIM	1	RXE	PS1	PS0	CL	SL	0	SCK	FF88H	80H	R/W

SCK	Setting of serial clock
0	Internal baud rate generator output
1	Internal clock f_{CLK}

SL	Setting of stop bit length
0	1 bit
1	2 bits

CL	Setting of character length
0	7 bits
1	8 bits

PS1	PS0	Setting of parity bit
0	0	No parity
0	1	Transmission: A 0 will be added as a parity bit. Reception : A parity error will not be detected.
1	0	Odd parity
1	1	Even parity

RXE	Reception status
0	Reception disabled
1	Reception enabled

Remark f_{CLK}: Internal system clock

10.4.1 Configuration of baud rate generator

The baud rate generator uses timer 4 (TM4) in the real-time pulse unit (RPU). The TMC2 register and the 10-bit compare register (CM40) control this generator.

Fig. 10-7 shows the configuration of the baud rate generator.

(1) Timer control register 2 (TMC2)

TMC2 is an 8-bit register used for TM4 count clock selection and baud rate generator operation control.
Both data reading and writing are possible with 8-bit and single-bit manipulation instructions.
When the $\overline{\text{RESET}}$ signal is input, TMC2 is set to 00H.
Fig. 10-8 shows the TMC2 register format.

(2) Selector

The selector selects the count clock of the 10-bit timer (TM4) according to the contents of the TMC2 register.

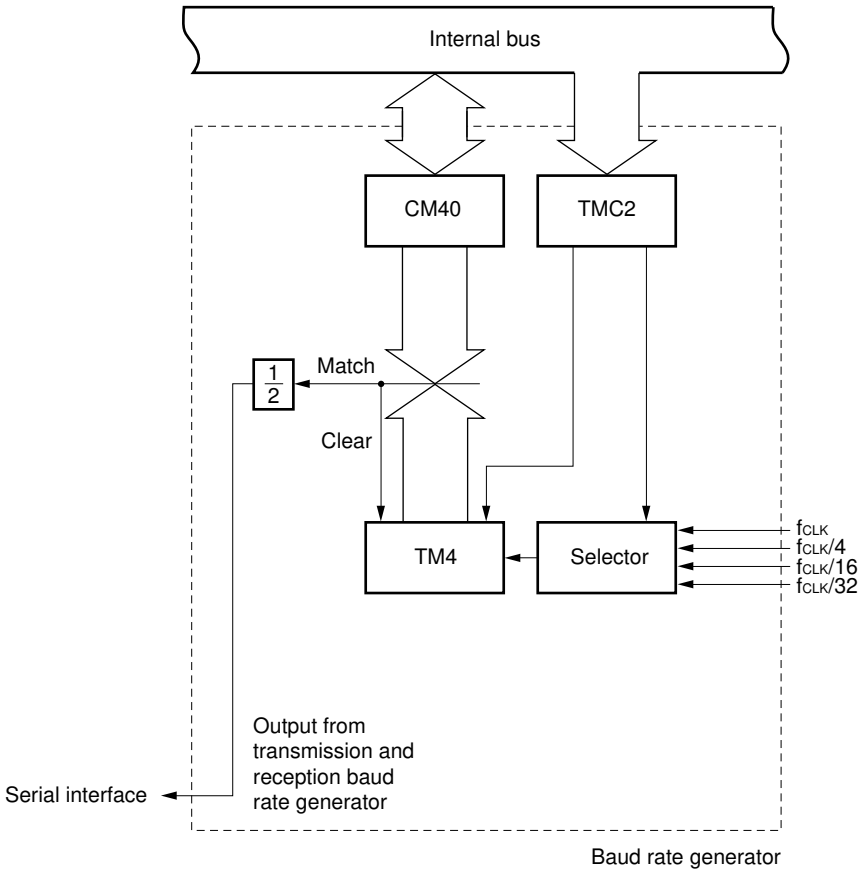
(3) 10-bit timer (TM4)

TM4 is a 10-bit timer that counts the count clocks selected by the selector.
When a matching signal is issued by the CM40 register, this timer is cleared to zero at the next count clock.
The TMC2 register controls the start and end of counting.
Only data reading is possible with 16-bit manipulation instructions.

(4) 10-bit compare register (CM40)

CM40 compares its data against the contents of TM4 and, when they match, issues a matching signal. TM4 is cleared to zero in response to the matching signal.
The signal output by the baud rate generator has a frequency 1/2 of the matching signal frequency.
Both data reading and writing are possible with 16-bit manipulation instructions.
When the $\overline{\text{RESET}}$ signal is input, the contents of CM40 become undefined.

Fig. 10-7. Block Diagram of Baud Rate Generator



10.4.2 Setting a desired baud rate

Any desired baud rate can be selected for the serial clock by setting the timer control register 2 (TMC2) and 10-bit compare register (CM40).

(1) Baud rate equal to serial clock/16

In the asynchronous serial interface, the signals on the RxD pin are sampled by the clock whose frequency is 1/16 of the serial clock specified in the asynchronous serial interface mode register (ASIM).

(2) Setting a desired baud rate

The baud rate to be set can be obtained by the equation below. Set the CM40 register and TM4 counter values for the desired baud rate, then start the operation of the baud rate generator.

Equation to obtain baud rate

$$\text{Baud rate (bps)} = \frac{f_{CLK}}{2^n} \times \frac{1}{(m+1)} \times \frac{1}{2} \times \frac{1}{16}$$

f_{CLK} : Internal system clock (external oscillator frequency $f_{OSC}/2$)
m : Value set in CM40 (0 to 1023)
n : Value corresponding to TMC2 value (0, 2, 4, 5)

Table 10-1 shows baud rate setting examples.

(3) Baud rate difference

In the asynchronous serial interface, the RxD pin signals are sampled in synchronization with the start bit. If the difference between the receiving and transmission baud rates becomes greater than the value shown below, the start and stop bits are forced out of synchronization by more than 0.5 bits, degrading transmission quality.

$$\text{Maximum tolerance} = \frac{0.5 \text{ bits}}{\text{Length of 1 data frame}} = \frac{0.5 \text{ bits}}{12 \text{ bits maximum}} = 4.1\%$$

Fig. 10-8. Timer Control Register 2 Format

Symbol	7	6	5	4	<3>	2	1	0	Address	When reset	R/W
TMC2	0	0	0	0	CE4	1	PRM41	PRM40	FF76H	04H	R/W

PRM41	PRM40	Setting of count clock (Hz) in TM4
0	0	f_{CLK}
0	1	$f_{CLK}/4$
1	0	$f_{CLK}/16$
1	1	$f_{CLK}/32$

CE4	TM4 operation control
0	Clear and halt
1	Count

Remark f_{CLK}: Internal system clock

Table 10-1. Baud Rate Setting Examples (Asynchronous Serial Interface)

External oscillator frequency f_{OSC} (MHz)	32			28			25			20		
Internal system clock f_{CLK} (MHz)	16			14			12.5			10		
Baud rate (bps)	m	Count clock	Baud rate difference (%)	m	Count clock	Baud rate difference (%)	m	Count clock	Baud rate difference (%)	m	Count clock	Baud rate difference (%)
150	832	$f_{CLK}/4$	0.04	728	$f_{CLK}/4$	0.02	650	$f_{CLK}/4$	0.01	520	$f_{CLK}/4$	0.03
300	416	$f_{CLK}/4$	0.08	364	$f_{CLK}/4$	0.11	325	$f_{CLK}/4$	0.15	259	$f_{CLK}/4$	0.16
600	832	f_{CLK}	0.04	728	f_{CLK}	0.02	650	f_{CLK}	0.01	520	f_{CLK}	0.03
1200	416	f_{CLK}	0.08	364	f_{CLK}	0.11	325	f_{CLK}	0.15	259	f_{CLK}	0.16
2400	207	f_{CLK}	0.16	181	f_{CLK}	0.16	162	f_{CLK}	0.15	129	f_{CLK}	0.16
4800	103	f_{CLK}	0.16	90	f_{CLK}	0.16	80	f_{CLK}	0.47	64	f_{CLK}	1.37
9600	51	f_{CLK}	0.16	45	f_{CLK}	0.93	40	f_{CLK}	0.76	32	f_{CLK}	1.36
19200	25	f_{CLK}	0.16	22	f_{CLK}	0.93	19	f_{CLK}	1.7	15	f_{CLK}	1.73
38400	12	f_{CLK}	0.16	10	f_{CLK}	3.6	9	f_{CLK}	1.7	7	f_{CLK}	1.73
76800	6	f_{CLK}	7.0 Note	5	f_{CLK}	5.1 Note	4	f_{CLK}	1.7	3	f_{CLK}	1.73
153600	–	–	–	–	–	–	2	f_{CLK}	15.2 Note	1	f_{CLK}	1.73

Note The difference is too great for proper operation.

Remark m: Value set in compare register (CM40)

10.5 Transmitting Data

Writing data to the shift register for transmission (TXS) activates transmission. The next block of data is written to the register in response to the transmission completion interrupt (INTST).

(1) Transmitting data

The asynchronous serial interface of the μPD78356 always permits data transmission. Transmission is activated by writing data to the shift register for transmission (TXS). The start, parity and stop bits are automatically added.

As transmission proceeds, data is shifted in the TXS register. When the register becomes empty, the transmission completion interrupt (INTST) occurs.

Transmission is halted unless the next block of data is written to the register.

(2) Writing data to the TXS register

The transmission rate can be affected if the next block of data is not written to the TXS register immediately after transmitting the current register data.

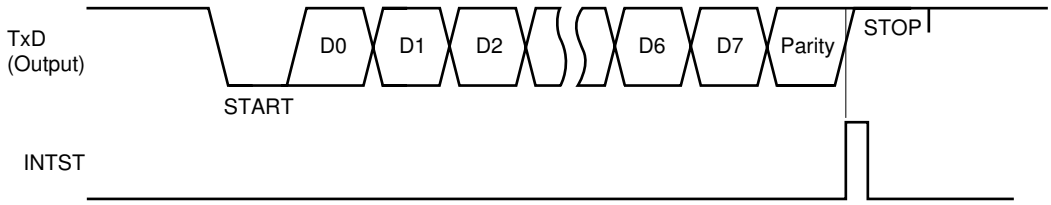
The use of the block transfer (BLKTRS) mode of the macro service is recommended in writing transmission data to the TXS register. The macro service is immediately activated whenever INTST occurs, regardless of the priority. This ensures a higher transmission rate.

Cautions 1. Usually, the transmission completion interrupt (INTST) occurs whenever the TXS register becomes empty. However, even when the $\overline{\text{RESET}}$ signal is input and the TXS becomes empty, INTST does not occur.

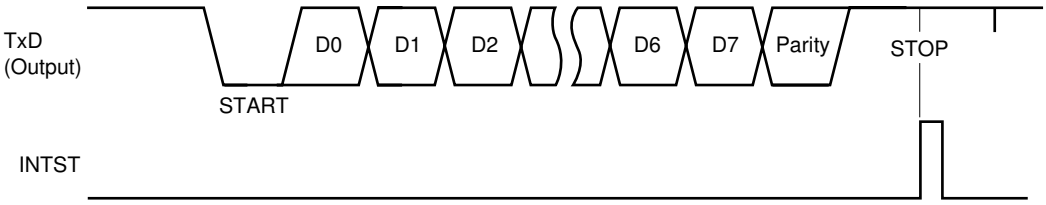
2. Data written to the TXS register during transmission (before INTST occurs) is regarded as invalid.

Fig. 10-9. Transmission Completion Interrupt Timing for Asynchronous Serial Interface

(a) Stop bit length: 1



(b) Stop bit length: 2



Remark INTST... Vector table address: 002EH (TPF = 0), 802EH (TPF = 1)
Macro service control word address: FE2EH

Caution Usually, the transmission completion interrupt (INTST) occurs whenever the TXS register becomes empty. However, even when the $\overline{\text{RESET}}$ signal is input and the TXS becomes empty, INTST does not occur.

10.6 Receiving Data

When reception is enabled, sampling starts on the RxD pin. Upon detection of a start bit, data reception begins. The reception completion interrupt (INTSR) occurs each time one frame of data has been received. Usually, the receive buffer (RXB) transfers the received data to the memory in response to this interrupt.

(1) Receiving data

Reception is enabled by setting the RXE bit of the asynchronous serial interface mode register (ASIM) to 1. Once reception is enabled, signal sampling starts on the RxD pin using the serial clock specified in the ASIM register.

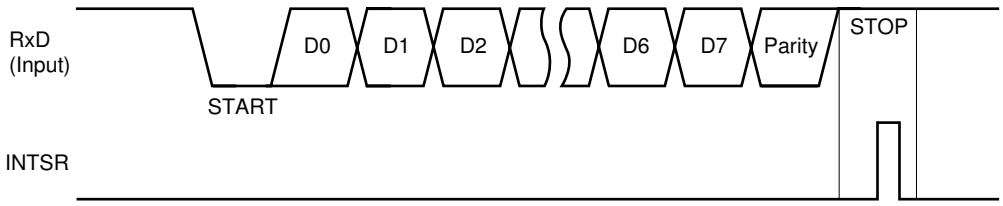
When the input to the RxD pin becomes low, the divide-by-16 counter starts counting. When the counter has counted to eight, the start timing signal for data sampling is issued. The start timing signal is sampled at the RxD pin. When the start timing signal is low, it is recognized as the start bit and the counter is initialized and resumes counting for data sampling. When character data and parity and stop bits are detected following the start bit, the process of receiving one frame of data is completed.

Upon completion of the process of receiving one frame of data, the shift register for reception transfers the received data to the receive buffer (RXB). As a result, a reception completion interrupt (INTSR) occurs.

If an error occurs during data reception, the data associated with that error is transferred to the RXB. Then, INTSR and a reception error interrupt (INTSER) are issued at the same time.

Resetting the RXE bit to 0 during data reception immediately halts the reception process. However, the contents of the RXB and the asynchronous serial interface status register (ASIS) remain unchanged, and INTSR and INTSER do not occur.

Fig. 10-10. Reception Completion Interrupt Timing for Asynchronous Serial Interface



Remark INTSR ... Vector table address: 002CH (TPF = 0), 802CH (TPF = 1)
Macro service control word address: FE2CH

Fig. 10-11. Setting of ASIM Register (Reception Enabled)

Symbol	7	<6>	5	4	3	2	1	0	Address	When reset	R/W
ASIM	1	RXE	PS1	PS0	CL	SL	0	SCK	FF88H	80H	R/W

SCK	Setting of serial clock
0	Internal baud rate generator output
1	Internal clock f_{CLK}

SL	Setting of stop bit length
0	1 bit
1	2 bits

CL	Setting of character length
0	7 bits
1	8 bits

PS1	PS0	Setting of parity bit
0	0	No parity
0	1	Transmission: A 0 will be added as a parity bit. Reception : A parity error will not be detected.
1	0	Odd parity
1	1	Even parity

RXE	Reception status
0	Reception disabled
1	Reception enabled

Remark f_{CLK}: Internal system clock

(2) Transferring receive buffer (RXB) data to memory

If the data stored in the receive buffer (RXB) is not transferred to memory before the reception of the next data block is completed, an overrun error occurs.

The use of the block transfer mode (BLKTRS) of the macro service is recommended in transferring received data to memory. The macro service is immediately activated whenever INTSR occurs, regardless of the priority. This ensures that the RXB data is transferred to memory before the reception of the next data block is completed.

Caution In the event of a reception error, be sure to read the RXB data. Otherwise, an overrun error occurs when the next data block is received, leading to an endless error condition.

10.7 Transmitting/Receiving Data Using the Macro Service ★

When transmitting data using the macro service, two vectored interrupt requests occur while only one vectored interrupt request occurs during reception.

- **Transmitting/receiving using the macro service**

  Transmission is started by writing data to the shift register for transmission (TXS). If transmission is executed using the macro service, data is written to TXS for the set number of times and then transmitted. The macro service processing to write the following data is executed by the transmission completion interrupt (INTST) occurring after the completion of the transmission. When the last data is written to TXS, the macro service is completed (MSC = 0), and a vectored interrupt request occurs (**Fig. 10-12** <1>).

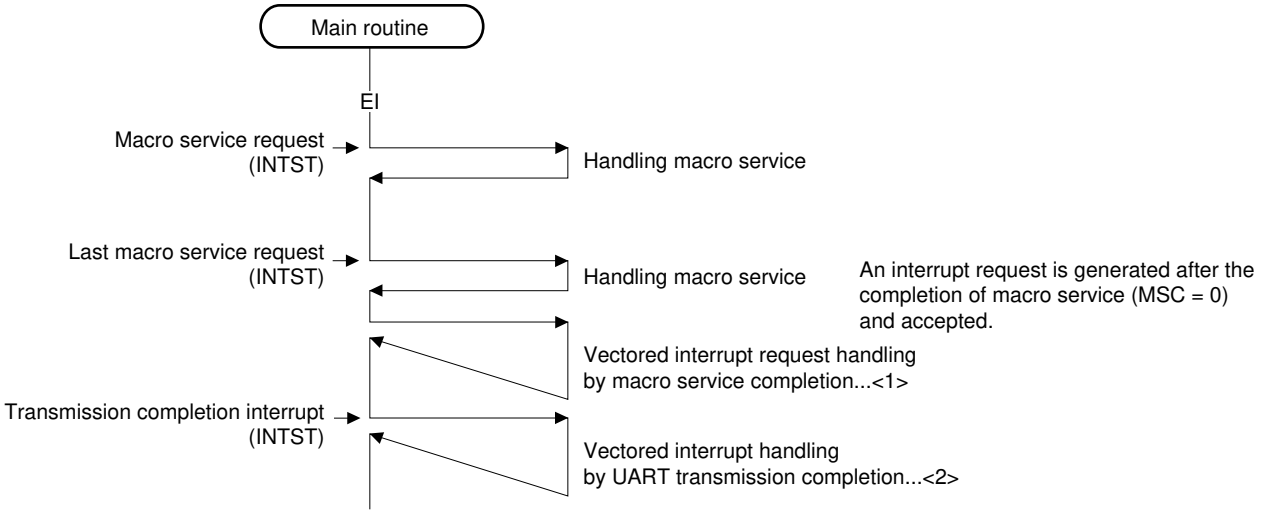
  Upon the completion of the process of transferring one frame of data, INTST occurs and a vectored interrupt request occurs again (**Fig. 10-12** <2>).

  Therefore, if the macro service is started by the INTST, two vectored interrupts may occur by the single interrupt request (INTST in this case).

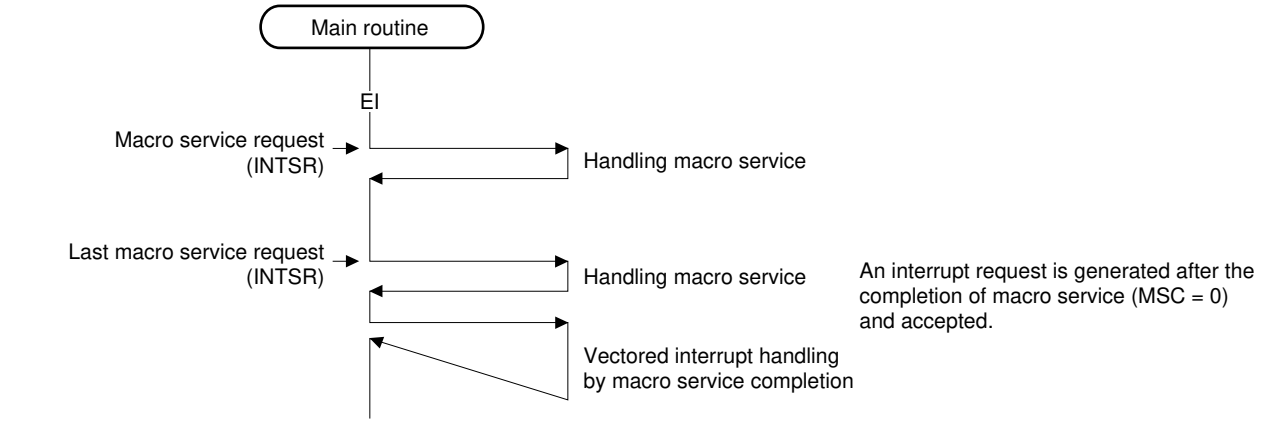
  On the other hand, during reception no more than one vectored interrupt requests occurs. In case of reception, only one vectored interrupt request occurs upon the completion of the macro service because the macro service processing is excuted to transfer the received data to memory by reception completion interrupt (INTSR) occurring after the completion of reception.

Fig. 10-12. Transmitting/Receiving Data Using the Macro Service

(a) Transmission operation



(b) Reception operation



10.8 Handling Reception Error

When a reception error has occurred, the user can find out what type of error it was by reading the data stored in the asynchronous serial interface status register (ASIS).

- **Three types of reception errors**

  Three types of reception errors can occur during data reception: a parity error, framing error, and overrun error. If a reception error occurs, an error flag is set in the ASIS register and, at the same time, a reception error interrupt (INTSER) occurs. Table 10-2 lists the cause of each reception error.

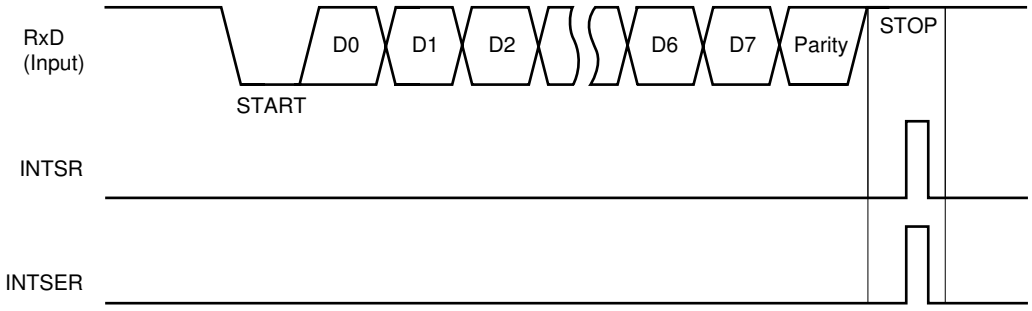
  By reading the ASIS register data during the INTSER process, the user can find out what type of reception error occurred. (See **Fig. 10-14**.)

  The ASIS register is reset to 0 either by reading the RXB data or by transmitting the next data block. (If the next data block is erroneous, the error flag corresponding to that data block is set.)

Table 10-2. Causes of Reception Errors

Reception error	Cause
Parity error	The parity bit specified before transmission does not match the parity bit detected in received data.
Framing error	The stop bit is not detected.
Overrun error	The reception of the next data block is completed before reading the RXB data.

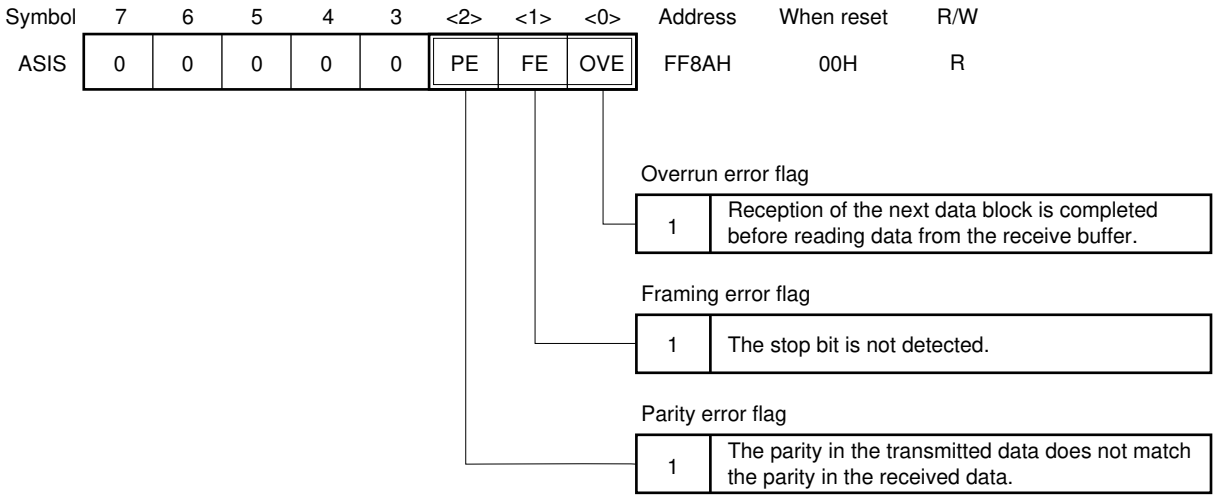
Fig. 10-13. Reception Error Timing



Remarks 1. INTSR ... Vector table address: 002CH (TPF = 0), 802CH (TPF = 1)
Macro service control word address: FE2CH

2. INTSER .. Vector table address: 002AH (TPF = 0), 802AH (TPF = 1)
Macro service control word address: FE2AH

Fig. 10-14. ASIS Register Format



Cautions 1. The ASIS register is reset to 0 when the RXB buffer is read or the next data block is received. To find out about the error, be sure to read the ASIS register before reading the RXB buffer.

In the case that the received data is transmitted to the memory by macro service, the reception buffer (RxB) is incidentally read out when receiving the serial data, and the ASIS register is reset to 0. Therefore, before using macro service, make sure this will not affect performance.

The error occurrence can be known when the reception error interrupt request flag (SERIF) is set to 1, or when the reception error interrupt (INTSER) is acknowledged.

2. In the event of a reception error, be sure to read the RXB data. Otherwise, an overrun error occurs when the next data block is received, leading to an endless error condition.

CHAPTER 11 CLOCK SYNCHRONOUS SERIAL INTERFACE

The μPD78356 provides a clock synchronous serial interface. This interface operates in two modes: 3-wire serial I/O mode and serial bus interface (SBI) mode. This makes the interface capable of responding to a wide range of applications.

This interface operates independently of the asynchronous serial interface and the clock synchronous serial interface with the pin switching function.

11.1 Configuration of Clock Synchronous Serial Interface

The clock synchronous serial interface is controlled by the clock synchronous serial interface mode register 0 (CSIM0) and serial bus interface control register (SBIC). Data transmitted and received can be written to and read from the shift register (SIO0) register.

(1) Clock synchronous serial interface mode register 0 (CSIM0)

The CSIM0 register is an 8-bit register that controls the operation of the clock synchronous serial interface. Both data reading and writing are possible with 8-bit and single-bit manipulation instructions.

When the $\overline{\text{RESET}}$ signal is input, CSIM0 is set to 00H.

(2) Serial bus interface control register (SBIC)

The SBIC register contains eight bits that control the serial bus status and indicate the status of data input from the serial bus. This register can be used in the SBI mode only (it cannot be used in the 3-wire serial I/O mode).

Both 8-bit and single-bit manipulation instructions are used to operate the register. Data reading and writing may or may not be possible, depending on the bit. When a bit for which only data write is allowed is read, 0 is returned.

When the $\overline{\text{RESET}}$ signal is input, the SBIC register is set to 00H.

The ACKD, CMDD and RELD flags are cleared to zero when data transmission and reception is prohibited (the CTXE0 and CRXE0 bits are both set to 0).

(3) Shift register (SIO0)

The SIO0 register is an 8-bit register that converts serial data to parallel data and vice versa. The register is used for both transmitting and receiving data.

Data is shifted in (received) or shifted out (transmitted) from the MSB or LSB side.

The transmitting and receiving of data is actually performed by writing data to and reading data from the SIO0 register.

For data writing and reading, 8-bit manipulation instructions are used.

When the $\overline{\text{RESET}}$ signal is input, the contents of SIO0 become undefined.

(4) SO latch

The SO latch maintains the level of the signal output on the SO00/SB0 pin. In the SBI mode, this can be directly controlled using software.

(5) Serial clock selector

This selector selects the serial clock to be used.

(6) Serial clock control circuit

This control circuit controls the supply of the serial clock to the shift register. When an internal clock is being used, it also controls the clock output to the $\overline{\text{SCK00}}$ pin.

(7) Serial clock counter

This counter counts serial clocks being output and input during data transmission and reception to check whether 8-bit data has been transmitted and received.

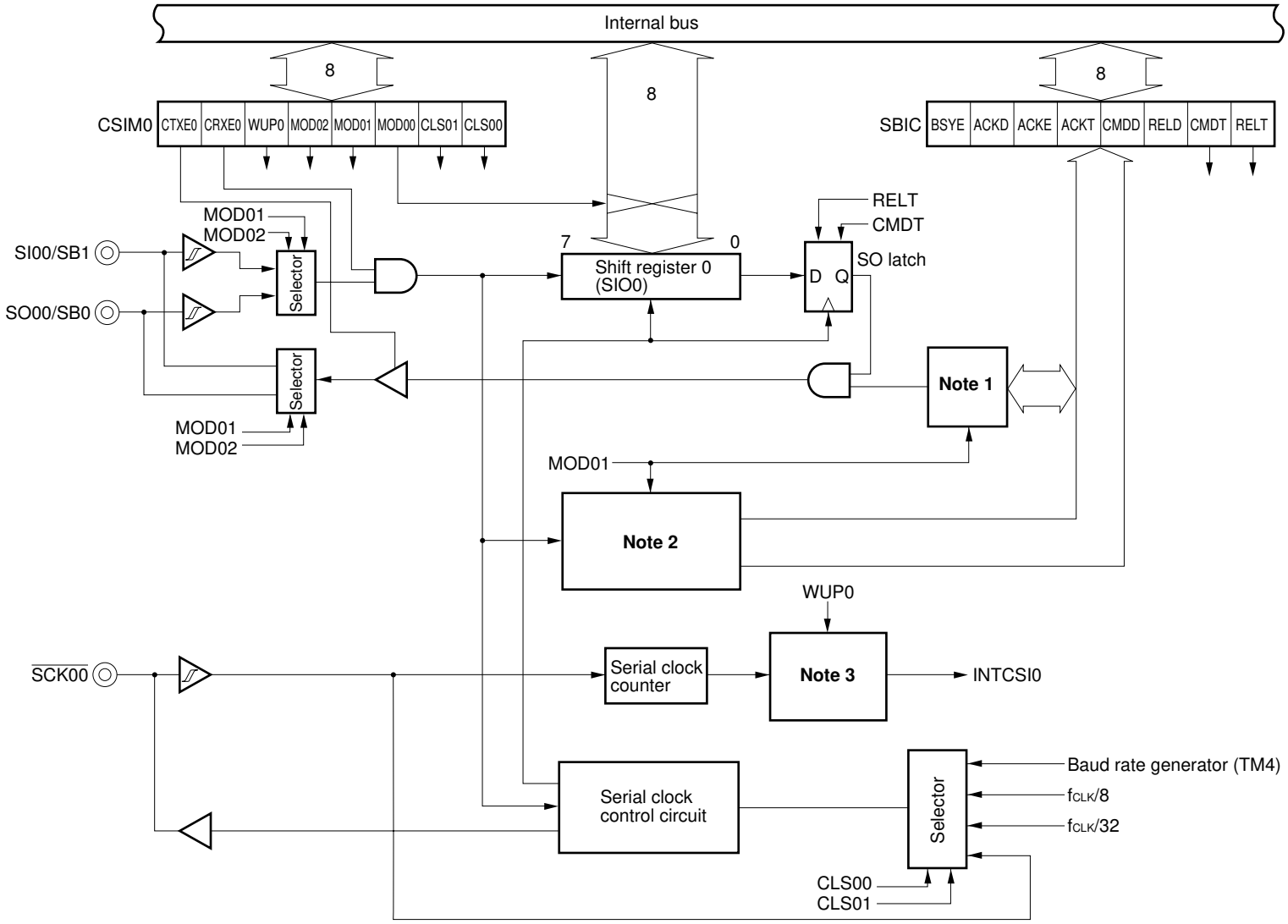
(8) Interrupt signal generation control circuit

This control circuit controls whether to make an interrupt request when the serial clock counter has counted eight serial clocks. In the 3-wire serial I/O mode, an interrupt request is made each time eight serial clocks have been counted. In the SBI mode, the request is made if the preset conditions are met.

(9) Busy/acknowledge signal detection circuit and bus release/command/acknowledge signal detection circuit

These circuits output and detect signals in the SBI mode. In the 3-wire serial I/O mode, these circuits are not used.

Fig. 11-1. Block Diagram of Clock Synchronous Serial Interface



Notes 1. Busy/acknowledge signal detection circuit
2. Bus release/command/acknowledge signal detection circuit
3. Interrupt signal generation control circuit

11.2 Setting Pins for Serial Transmission

The SO00/SB0, SI00/SB1, and $\overline{\text{SCK00}}$ pins, which are also used as general purpose ports, must be set to the control mode before beginning serial transmission.

(1) Setting pins for serial transmission

The clock synchronous serial interface uses the SO00/SB0, SI00/SB1, and $\overline{\text{SCK00}}$ pins. These pins also serve as general purpose ports P32, P33 and P34, respectively. Therefore, before beginning serial transmission, these pins must be set to the control mode in the port 3 mode control register (PMC3).

Fig. 11-2. Port 3 Mode Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PMC3	PMC37	PMC36	PMC35	PMC34	PMC33	PMC32	PMC31	PMC30	FF43H	00H	R/W

PMC30	Control mode setting for P30 pin
0	I/O port mode
1	TxD output mode

PMC31	Control mode setting for P31 pin
0	I/O port mode
1	RxD input mode

PMC32	Control mode setting for P32 pin
0	I/O port mode
1	SB0 input and output mode/SO00 output mode

PMC33	Control mode setting for P33 pin
0	I/O port mode
1	SB1 input and output mode/SI00 input mode

PMC34	Control mode setting for P34 pin
0	I/O port mode
1	$\overline{\text{SCK00}}$ I/O mode

PMC35	Control mode setting for P35 pin
0	I/O port mode
1	TCLR1 input mode

PMC36	Control mode setting for P36 pin
0	I/O port mode
1	TI1 input mode/TO11 output mode

PMC37	Control mode setting for P37 pin
0	I/O port mode
1	TO10 output mode

(2) Reading pin levels

When port 3 (P3) has been set to the control mode in the port 3 mode control register (PMC3), the read instructions of P3 can read the following information:

(a) When a bit of the port 3 mode register (PM3) is set to 1
- The corresponding pin level can be read.

(b) When a bit of the PM3 is set to 0 (reset)
- The level of the internal signal can be read.

Reading pin levels allows pins checks, including the check for serial bus contention.
Writing data in P3 does not change any pin level. (Data is written in the P3 output buffer.)

Fig. 11-3. Port 3 Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	FF23H	FFH	R/W

PM3n	I/O mode setting for P3n pin (n = 0 to 7)
0	Output mode (Output buffer is on.)
1	Input mode (Output buffer is off.)

11.3 Setting Baud Rate

The baud rate generator output, internal clock $f_{CLK}/8$ or $f_{CLK}/32$, or the external clock can be selected as the serial clock. The baud rate generator output can be set at any desired baud rate, independently of the operation clock frequency.

(1) Baud rate equal to serial clock

In the clock synchronous serial interface, received data is sampled at the rising edge of the serial clock, which means the frequency of the serial clock is equal to the baud rate.

(2) Selecting serial clock

The serial clock can be selected using the CLS01 and CLS00 bits of the clock synchronous serial interface mode register 0 (CSIM0).

When an external clock is selected, the $\overline{SCK00}$ pin is used for the input of the serial clock from the device with which communication is being performed.

With the 16-MHz internal system clock, $f_{CLK}/8$ provides a baud rate of 2 Mbps and $f_{CLK}/32$ provides a baud rate of 500 Kbps.

Remark The baud rate generator is also used for the asynchronous serial interface and the clock synchronous serial interface with the pin switching function. (See **10.4** and **12.4**.)

Fig. 11-4. Setting of CSIM0 Register (Serial Clock)

Symbol	<7>	<6>	<5>	<4>	3	2	1	0	Address	When reset	R/W
CSIM0	CTXE0	CRXE0	WUP0	MOD02	MOD01	MOD00	CLS01	CLS00	FF80H	00H	R/W

CLS01	CLS00	Setting of serial clock		$\overline{SCK}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD01	MOD00	Operation mode	Start bit	Number of pins	Pin name
0	0	3-wire serial I/O mode	MSB	3	SO00/SB0, SI00/SB1, $\overline{SCK00}$
0	1		LSB		
1	0	SBI mode	MSB	2	SO00/SB0, (or SI00/ SB1), $\overline{SCK00}$
1	1	Setting prohibited			

MOD02	Setting of pin for transmission and reception
0	SO00/SB0 pin is used. (SI00/SB1 pin goes to high-impedance state.)
1	SI00/SB1 pin is used. (SO00/SB0 pin goes to high-impedance state.)

WUP0	Wake-up function control
0	Interrupt request signal is generated for each serial transfer in every mode.
1	Interrupt request signal is generated only when an address is received in SBI mode. The serial output pin goes to high-impedance state.

CRXE0	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE0	Data transmission
0	Disabled (Output buffers of SO00/SB0 and SI00/SB1 pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

Caution Do not switch serial clocks during data transmission. Since the switching is performed asynchronously with the serial clock being supplied, doing this during data transmission can produces a serial clock of undefined frequency.

11.3.1 Configuration of baud rate generator

The baud rate generator uses timer 4 (TM4) in the real-time pulse unit (RPU), and is controlled by the timer control register 2 (TMC2) and 10-bit compare register (CM40).

Fig. 11-5 shows the configuration of the baud rate generator.

(1) Timer control register 2 (TMC2)

TMC2 is an 8-bit register used for TM4 count clock selection and baud rate generator operation control.
Both data reading and writing are possible with 8-bit and single-bit manipulation instructions.
When the $\overline{\text{RESET}}$ signal is input, TMC2 is set to 00H.
Fig. 11-6 shows the TMC2 register format.

(2) Selector

The selector selects the count clock of the 10-bit timer (TM4) according to the contents of the TMC2 register.

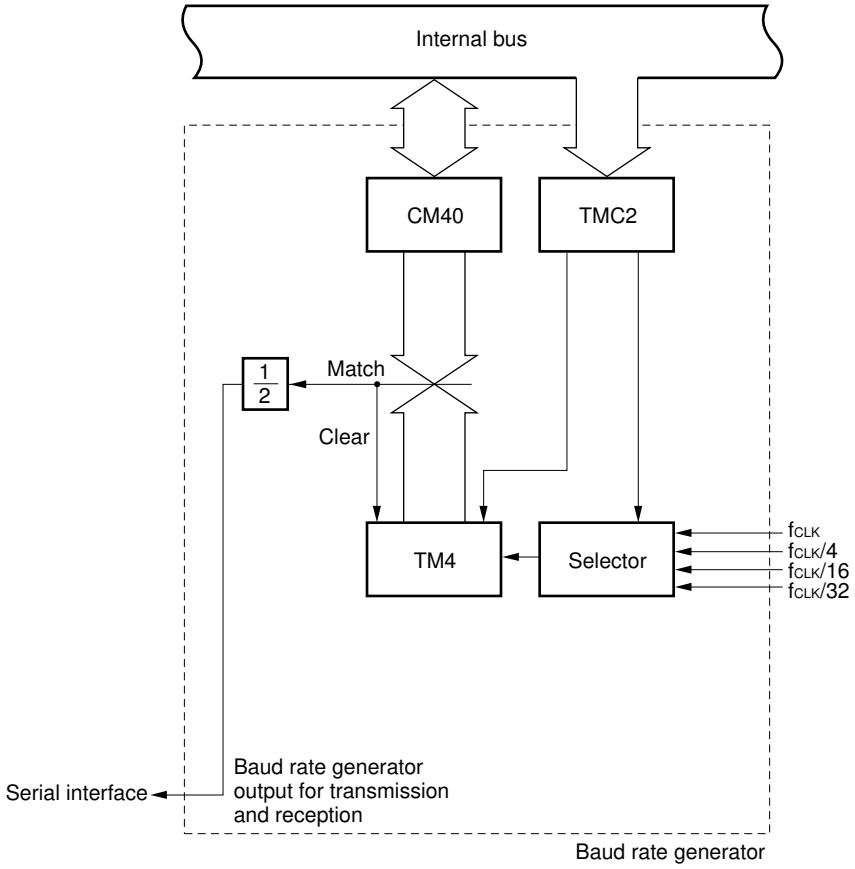
(3) 10-bit timer (TM4)

TM4 is a 10-bit timer that counts the count clocks selected by the selector.
When a matching signal is issued by the CM40 register, this timer is cleared to zero at the next count clock.
The TMC2 register controls the start and end of counting.
Only data reading is possible with 16-bit manipulation instructions.

(4) 10-bit compare register (CM40)

CM40 compares its data against the contents of TM4 and, when they match, issues a matching signal. TM4 is cleared to zero in response to the matching signal.
The signal output by the baud rate generator, has a frequency 1/2 of the matching signal frequency.
Both data reading and writing are possible with 16-bit manipulation instructions.
When the $\overline{\text{RESET}}$ signal is input, the contents of CM40 become undefined.

Fig. 11-5. Block Diagram of Baud Rate Generator



11.3.2 Setting a desired baud rate

Any desired baud rate can be selected for the serial clock by setting the timer control register 2 (TMC2) and 10-bit compare register (CM40).

(1) Baud rate equal to serial clock

In the clock synchronous serial interface, received data is sampled at the rising edge of the serial clock, which means the frequency of the serial clock is equal to the baud rate.

(2) Setting a desired baud rate

The baud rate to be set can be obtained by the equation below. Set the CM40 register and TM4 counter values for the desired baud rate, then start the operation of the baud rate generator.

Equation to obtain baud rate

$$\text{Baud rate (bps)} = \frac{f_{CLK}}{2^n} \times \frac{1}{(m + 1)} \times \frac{1}{2}$$

f_{CLK} : Internal system clock (external oscillator frequency $f_{OSC}/2$)

m : Value set in CM40 (0 to 1023)

n : Value corresponding to TMC2 value (0, 2, 4, 5)

Fig. 11-6. Timer Control Register 2 Format

Symbol	7	6	5	4	<3>	2	1	0	Address	When reset	R/W
TMC2	0	0	0	0	CE4	1	PRM41	PRM40	FF76H	04H	R/W

PRM41	PRM40	Setting of count clock (Hz) in TM4
0	0	f_{CLK}
0	1	$f_{CLK}/4$
1	0	$f_{CLK}/16$
1	1	$f_{CLK}/32$

CE4	TM4 operation control
0	Clear and halt
1	Count

Remark f_{CLK}: Internal system clock

11.4 Two Operation Modes of Clock Synchronous Serial Interface

Using the 3-wire serial I/O mode is advantageous when communicating with a device having a conventional clock synchronous serial interface.

The serial bus interface (SBI) mode is an innovation of NEC that allows communication with multiple devices via two lines.

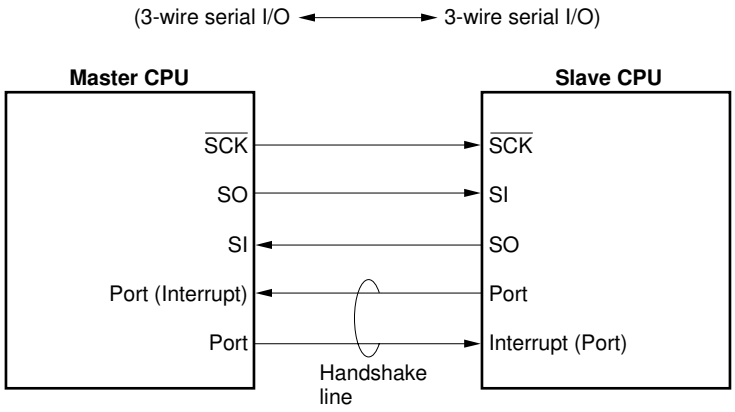
The μPD78356 operates in the following two modes:

(1) Three-wire serial I/O mode

In the 3-wire serial I/O mode, 8-bit data communication is carried out using three lines: serial clock ($\overline{SCK}$), serial input (SI), and serial output (SO). Using this mode is advantageous when connecting to a peripheral I/O device having a conventional clock synchronous serial interface or a display controller.

To connect more than one device, another line is required for handshaking.

Fig. 11-7. Example of System Configuration in the 3-Wire Serial I/O Mode



(2) Serial bus interface (SBI) mode

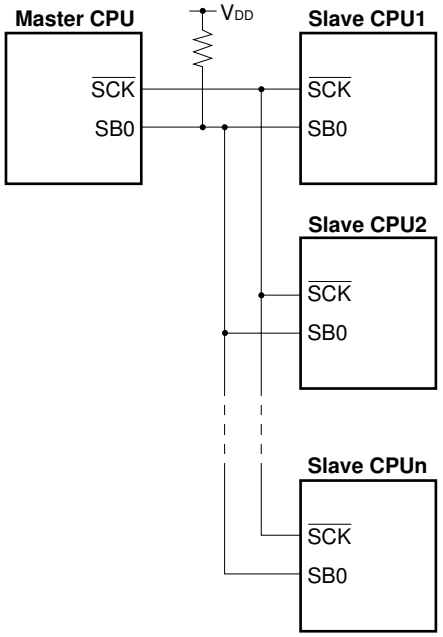
The SBI mode, which conforms to NEC's serial bus format, allows communication with multiple devices via two lines: serial clock ($\overline{SCK}$) and serial data bus (SB0 or SB1).

In the SBI mode, information called "address," "command," and "data" is output on the serial data bus. Address is used to select devices for serial communication; command gives instructions to the devices; and data is the actual data to be transmitted.

The use of the SBI mode eliminates the need for the handshake line that is necessary when connecting multiple devices through the conventional clock synchronous serial interface, thus ensuring more effective use of I/O ports and reduced software load.

Since the serial data bus pins (SB0 and SB1) employ an open drain configuration for output in the SBI mode, the serial data bus line is provided in a wired OR state and therefore requires a pull-up resistor.

Fig. 11-8. Example of System Configuration in the SBI Mode



11.5 Setting 3-Wire Serial I/O Mode

The 3-wire serial I/O mode is set using the clock synchronous serial interface mode register 0 (CSIM0). Since this mode allows the selection of the start bit (MSB or LSB), communication can be performed with a variety of devices.

(1) Setting 3-wire serial I/O mode

The 3-wire serial I/O mode is set using the MOD01 and MOD00 bits of the CSIM0 register (see **Fig. 11-10**). Since this mode allows the selection of the start bit (MSB or LSB), communication can be performed with a variety of devices.

(2) Operation timing in 3-wire serial I/O mode

In the 3-wire serial I/O mode, data is transmitted and received in blocks of eight bits, with either the MSB or LSB as the start bit (specified in the CSIM0 register). The eight-bit data block is transferred bit by bit in synchronization with the serial clock.

Transmitted data is output in synchronization with the falling edge of the $\overline{SCK00}$ signal. Received data is sampled at the rising edge of the $\overline{SCK00}$ signal. At the rising edge of the 8th $\overline{SCK00}$ signal, interrupt request INTCSI0 is generated.

When the internal clock is used as the $\overline{SCK00}$ signal, the output is halted at the rising edge of the 8th $\overline{SCK00}$ signal. The $\overline{SCK00}$ pin maintains the high level until the transmission or reception of the next data block is activated.

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

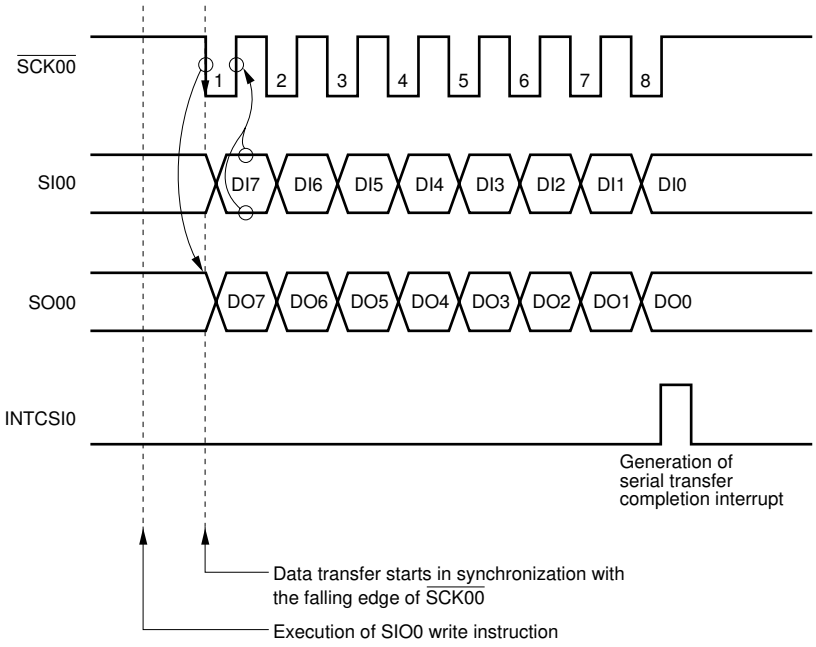


Fig. 11-10. Setting of CSIM0 Register (3-Wire Serial I/O Mode)

Symbol	<7>	<6>	<5>	<4>	3	2	1	0	Address	When reset	R/W
CSIM0	CTXE0	CRXE0	WUP0	MOD02	MOD01	MOD00	CLS01	CLS00	FF80H	00H	R/W

CLS01	CLS00	Setting of serial clock		$\overline{SCK}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD01	MOD00	Operation mode	Start bit	Number of pins	Pin name
0	0	3-wire serial I/O mode	MSB	3	SO00/SB0, SI00/SB1, $\overline{SCK00}$
0	1		LSB		
1	0	SBI mode	MSB	2	SO00/SB0, (or SI00/ $\overline{SB1}$), $\overline{SCK00}$
1	1	Setting prohibited			

MOD02	Setting of pin for transmission and reception
0	SO00/SB0 pin is used. (SI00/SB1 pin goes to high-impedance state.)
1	SI00/SB1 pin is used. (SO00/SB0 pin goes to high-impedance state.)

WUP0	Wake-up function control
0	Interrupt request signal is generated for each serial transfer in every mode.
1	Interrupt request signal is generated only when an address is received in SBI mode. The serial output pin goes to high-impedance state.

CRXE0	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE0	Data transmission
0	Disabled (Output buffers of SO00/SB0 and SI00/SB1 pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

11.5.1 Transmitting data in 3-wire serial I/O mode

Writing data in the shift register (SIO0) after enabling transmission in the clock synchronous serial interface mode register 0 (CSIM0) activates transmission.

(1) Activating transmission

Transmission is activated by setting the CTXE0 bit of the CSIM0 register to 1 (set the CRXE0 bit to 0) and then writing transmission data to the SIO0 register. The block transfer (BLKTRS) mode of the macro service is useful in writing data to the SIO0 register.

While the CTXE0 bit is reset to 0, the output of the SO00 pin remains in the high-impedance state.

(2) Transmitting data in synchronization with the serial clock

(a) When an internal clock is selected as the serial clock

Once transmission is activated, the $\overline{\text{SCK00}}$ pin starts the output of the serial clock. The SIO0 register then sequentially transfers data to the SO00 pin in synchronization with the falling edge of the serial clock.

(b) When an external clock is selected as the serial clock

Upon the activation of transmission, data is sequentially transmitted from SIO0 to the SO00 pin in synchronization with the falling edge of the serial clock that is input to the $\overline{\text{SCK00}}$ pin. When transmission is not activated, sending the serial clock to the $\overline{\text{SCK00}}$ pin does not cause the SIO0 register to transfer data and the output level of the SO00 pin does not change.

Fig. 11-11. Timing of 3-Wire Serial I/O Mode (Transmission)

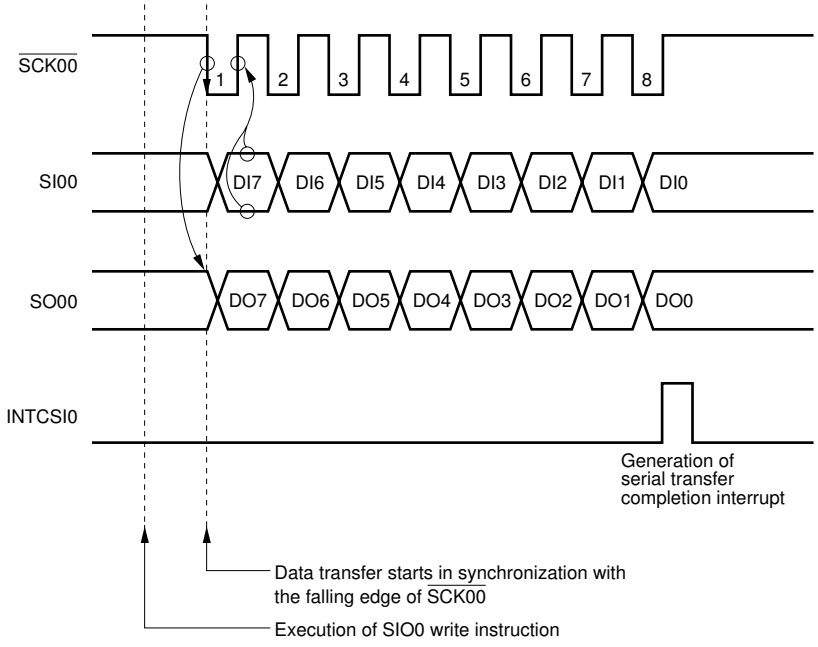


Fig. 11-12. Setting of CSIM0 Register (Transmission Enabled)

Symbol	<7>	<6>	<5>	<4>	3	2	1	0	Address	When reset	R/W
CSIM0	CTXE0	CRXE0	WUP0	MOD02	MOD01	MOD00	CLS01	CLS00	FF80H	00H	R/W

CLS01	CLS00	Setting of serial clock		$\overline{\text{SCK}}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD01	MOD00	Operation mode	Start bit	Number of pins	Pin name
0	0	3-wire serial I/O mode	MSB	3	SO00/SB0, SI00/SB1, $\overline{\text{SCK00}}$
0	1		LSB		
1	0	SBI mode	MSB	2	SO00/SB0, (or SI00/ SB1), $\overline{\text{SCK00}}$
1	1	Setting prohibited			

MOD02	Setting of pin for transmission and reception
0	SO00/SB0 pin is used. (SI00/SB1 pin goes to high-impedance state.)
1	SI00/SB1 pin is used. (SO00/SB0 pin goes to high-impedance state.)

WUP0	Wake-up function control
0	Interrupt request signal is generated for each serial transfer in every mode.
1	Interrupt request signal is generated only when an address is received in SBI mode. The serial output pin goes to high-impedance state.

CRXE0	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE0	Data transmission
0	Disabled (Output buffers of SO00/SB0 and SI00/SB1 pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

11.5.2 Receiving data in 3-wire serial I/O mode

Reception is activated by setting the CRXE0 bit of the clock synchronous serial interface mode register 0 (CSIM0) to 1 (reception enabled) (when CTXE0 = 0) or, when reception is already enabled, by reading data from the shift register (SIO0).

(1) Activating reception

Reception can be activated in the following two ways:

<1> When CTXE0 bit is 0, change the setting of the CRXE0 bit of the CSIM0 register from 0 (reception disabled) to 1 (reception enabled).

<2> When the CRXE0 bit is already set to 1, read data from the SIO0 register.

The block transfer (BLKTRS) mode of the macro service is useful in reading received data from the SIO0 register. Attempting to set the CRXE0 bit to 1 when it is already set to 1 does not activate reception. Furthermore, attempting to set the CRXE0 bit to 1 from 0 when CTXE0 bit is 1 does not activate reception.

(2) Receiving data in synchronization with the serial clock

(a) When an internal clock is selected as the serial clock

Once reception is activated, the $\overline{SCK00}$ pin starts the output of the serial clock. The data is sequentially read from the SI00 pin to SIO0 in synchronization with the rising edge of the serial clock.

(b) When an external clock is selected as the serial clock

Once reception is activated, the data is sequentially read from the SI00 pin to SIO0 in synchronization with the rising edge of the serial clock input to the $\overline{SCK00}$ pin. When reception is not activated, inputting the serial clock to the $\overline{SCK00}$ pin does not cause the SIO0 register to transfer the data.

Fig. 11-13. Timing of 3-Wire Serial I/O Mode (Reception)

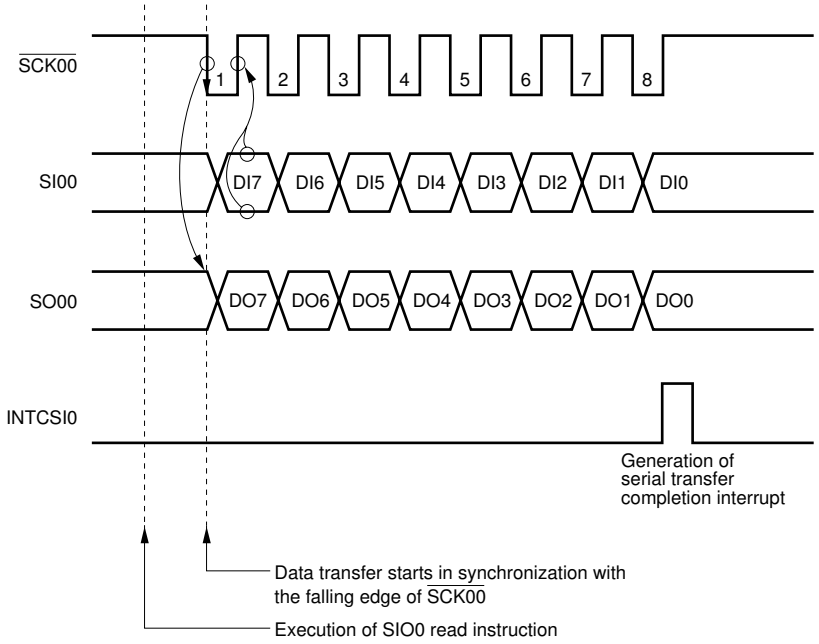


Fig. 11-14. Setting of CSIM0 Register (Reception Enabled)

Symbol	<7>	<6>	<5>	<4>	3	2	1	0	Address	When reset	R/W
CSIM0	CTXE0	CRXE0	WUP0	MOD02	MOD01	MOD00	CLS01	CLS00	FF80H	00H	R/W

CLS01	CLS00	Setting of serial clock		$\overline{SCK}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD01	MOD00	Operation mode	Start bit	Number of pins	Pin name
0	0	3-wire serial I/O mode	MSB	3	SO00/SB0, $\overline{SI00}$/SB1, $\overline{SCK00}$
0	1		LSB		
1	0	SBI mode	MSB	2	SO00/SB0, (or $\overline{SI00}$/SB1), $\overline{SCK00}$
1	1	Setting prohibited			

MOD02	Setting of pin for transmission and reception
0	SO00/SB0 pin is used. ($\overline{SI00}$/SB1 pin goes to high-impedance state.)
1	$\overline{SI00}$/SB1 pin is used. (SO00/SB0 pin goes to high-impedance state.)

WUP0	Wake-up function control
0	Interrupt request signal is generated for each serial transfer in every mode.
1	Interrupt request signal is generated only when an address is received in SBI mode. The serial output pin goes to high-impedance state.

CRXE0	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE0	Data transmission
0	Disabled (Output buffers of SO00/SB0 and $\overline{SI00}$/SB1 pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

11.5.3 Transmitting and receiving data in 3-wire serial I/O mode

Enabling both transmission and reception using the clock synchronous serial interface mode register 0 (CSIM0) allows the two operations to be performed simultaneously.

(1) Activating transmission and reception

Transmission and reception can be performed simultaneously when the CTXE0 and CRXE0 bits of the CSIM0 register are both set to 1.

The transmission and reception can be activated by writing the transmitted data to the shift register 0 (SIO0) when both the CTXE0 and CRXE0 bits of the CSIM0 register become 1 (transmission/reception enable state). Attempting to set the CRXE0 bit to 1 when it is already set to 1 does not activate transmission or reception.

(2) Transmitting and receiving data in synchronization with the serial clock

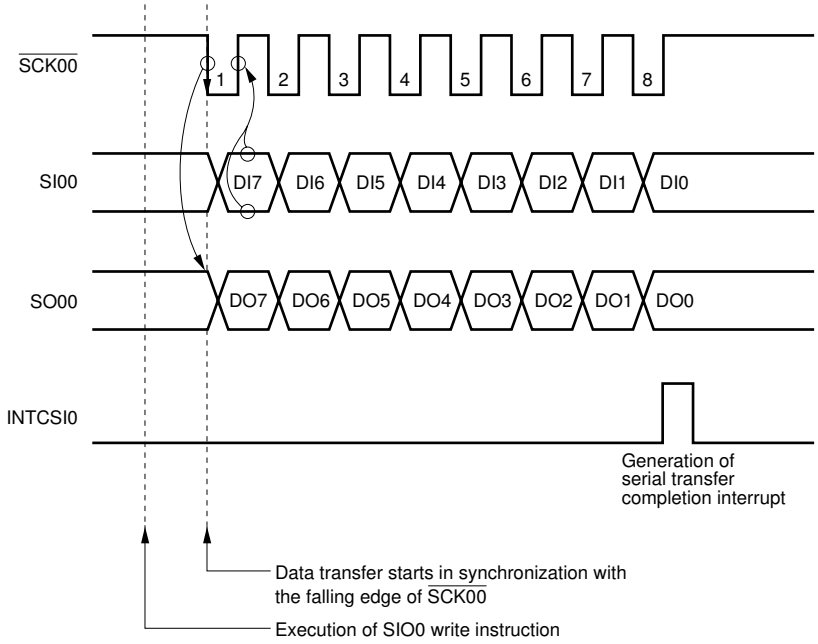
(a) When an internal clock is selected as the serial clock

Once transmission and reception are activated, the $\overline{\text{SCK00}}$ pin starts the output of the serial clock. The SIO0 register then sequentially transfers data to the SO00 pin in synchronization with the falling edge of the serial clock. At the same time, data is sequentially read from the SI00 pin to SIO0 in synchronization with the rising edge of the serial clock.

(b) When an external clock is selected as the serial clock

Upon the activation of transmission and reception, data is sequentially transmitted from SIO0 to the SO00 pin in synchronization with the falling edge of the serial clock that is input to the $\overline{\text{SCK00}}$ pin. At the same time, data is sequentially read from the SI00 pin to SIO0 in synchronization with the rising edge of the serial clock. When transmission and reception are not activated, sending the serial clock to the $\overline{\text{SCK00}}$ pin does not cause the SIO0 register to transfer data and the output level of the SO00 pin does not change.

Fig. 11-15. Timing of 3-Wire Serial I/O Mode (Transmission and Reception)



Remark INTCSI0 ... Vector table address: 0030H (TPF = 0), 8030H (TPF = 1)
Macro service control word address: FE30H

Fig. 11-16. Setting of CSIM0 Register (Transmission and Reception Enabled)

Symbol	<7>	<6>	<5>	<4>	3	2	1	0	Address	When reset	R/W
CSIM0	CTXE0	CRXE0	WUP0	MOD02	MOD01	MOD00	CLS01	CLS00	FF80H	00H	R/W

CLS01	CLS00	Setting of serial clock		$\overline{\text{SCK}}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD01	MOD00	Operation mode	Start bit	Number of pins	Pin name
0	0	3-wire serial I/O mode	MSB	3	SO00/SB0, SI00/SB1, SCK00
0	1		LSB		
1	0	SBI mode	MSB	2	SO00/SB0, (or SI00/ SB1), SCK00
1	1	Setting prohibited			

MOD02	Setting of pin for transmission and reception
0	SO00/SB0 pin is used. (SI00/SB1 pin goes to high-impedance state.)
1	SI00/SB1 pin is used. (SO00/SB0 pin goes to high-impedance state.)

WUP0	Wake-up function control
0	Interrupt request signal is generated for each serial transfer in every mode.
1	Interrupt request signal is generated only when an address is received in SBI mode. The serial output pin goes to high-impedance state.

CRXE0	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE0	Data transmission
0	Disabled (Output buffers of SO00/SB0 and SI00/SB1 pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

11.5.4 Action taken when shift operation is not in phase with serial clock

The synchronization of the shift operation with the serial clock can be restored by disabling both transmission and reception.

- **Action taken when shift operation is not in phase with serial clock**
  When an external clock is being used as the serial clock, a mismatch between the number of serial clock pulses and the shift operation can occur (for example because of noise). When this happens, disable both transmission and reception by setting the CTXE0 and CRXE0 bits to 0 (reset). This initializes the serial clock counter. Then, enable transmission or reception, and the first pulse input will be treated as the first serial clock. Based on this first clock, the synchronization between the shift operation and serial clock is restored.

11.6 Setting SBI Mode

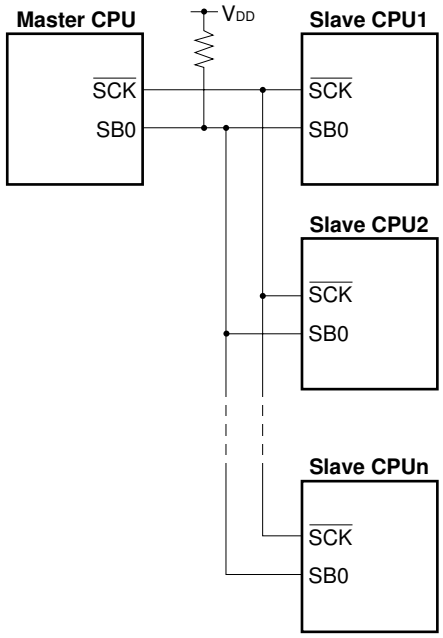
In the serial bus interface (SBI) mode, a serial bus is configured with two lines: $\overline{SCK}$ and SB0 (or SB1). This type of configuration reduces the number of ports and the software load.

- **Setting SBI mode**

  The SBI mode can be set using the MOD02, MOD01, and MOD00 bits of the clock synchronous serial interface mode register 0 (CSIM0).

  In the SBI mode, either the SB0 or SB1 pin can be selected as the I/O pin for serial data. As a result, one of two different serial bus configurations can be selected.

Fig. 11-17. Example of System Configuration in SBI Mode



Cautions 1. Since the serial data bus pins (SB0 and SB1) employ an open drain configuration for output in the SBI mode, the serial data bus line is provided in a wired OR state and therefore requires a pull-up resistor.

2. To switch between the master and slave CPUs, a pull-up resistor is also needed for the $\overline{SCK}$ line. This is because the $\overline{SCK}$ input/output switching is performed in the master and slave CPUs asynchronously.

Fig. 11-18. Setting of CSIM0 Register (SBI Mode)

Symbol	<7>	<6>	<5>	<4>	3	2	1	0	Address	When reset	R/W
CSIM0	CTXE0	CRXE0	WUP0	MOD02	MOD01	MOD00	CLS01	CLS00	FF80H	00H	R/W

CLS01	CLS00	Setting of serial clock		$\overline{\text{SCK}}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD01	MOD00	Operation mode	Start bit	Number of pins	Pin name
0	0	3-wire serial I/O mode	MSB	3	SO00/SB0, SI00/SB1, $\overline{\text{SCK00}}$
0	1		LSB		
1	0	SBI mode	MSB	2	SO00/SB0, (or SI00/SB1), $\overline{\text{SCK00}}$
1	1	Setting prohibited			

MOD02	Setting of pin for transmission and reception
0	SO00/SB0 pin is used. (SI00/SB1 pin goes to high-impedance state.)
1	SI00/SB1 pin is used. (SO00/SB0 pin goes to high-impedance state.)

WUP0	Wake-up function control
0	Interrupt request signal is generated for each serial transfer in every mode.
1	Interrupt request signal is generated only when an address is received in SBI mode. The serial output pin goes to high-impedance state.

CRXE0	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE0	Data transmission
0	Disabled (Output buffers of SO00/SB0 and SI00/SB1 pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

11.6.1 SBI data format

In the SBI mode, various control signals are defined using different combinations of the serial clock line ($\overline{SCK}$) and the serial data bus (SB0 or SB1).

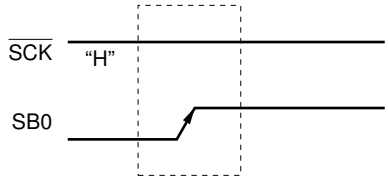
(1) Control signals used in SBI mode

In the SBI mode, the following eight control signals are defined using different combinations of two lines ($\overline{SCK}$ and SB0 or SB1):

<1> Bus release signal (REL)

The master CPU sends this signal to the slave CPU to notify that an address is to be transmitted from master to slave. The slave CPU has a built-in hardware component to detect the REL signal.

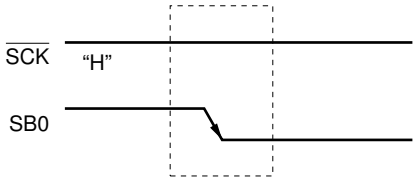
Fig. 11-19. Bus Release Signal



<2> Command signal (CMD)

The master CPU sends this signal to the slave CPU to notify that an address or command is to be sent from master to slave. The slave CPU has a built-in hardware component to detect the CMD signal.

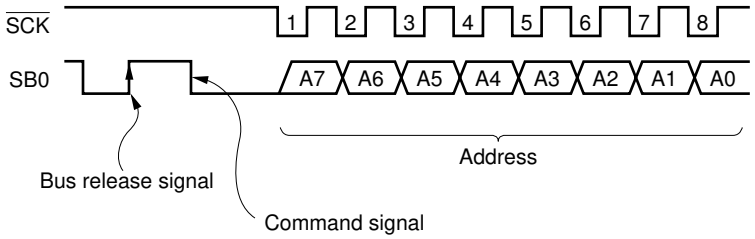
Fig. 11-20. Command Signal



<3> Address

This is an 8-bit data section that follows the REL and CMD signals. The master CPU selects the slave signal using this address. The slave CPU compares the received address with its own address and, when they match, performs communication.

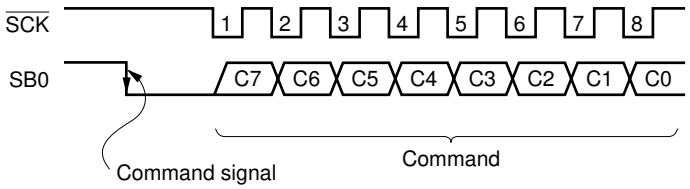
Fig. 11-21. Address



<4> Command

This 8-bit data section is transferred after the CMD signal when the REL signal is not output. The use of the command can be defined freely.

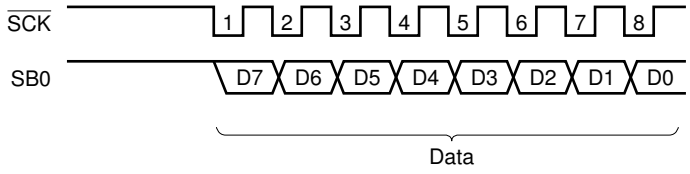
Fig. 11-22. Command



<5> Data

The master CPU outputs this 8-bit data section without the REL and CMD signals.

Fig. 11-23. Data

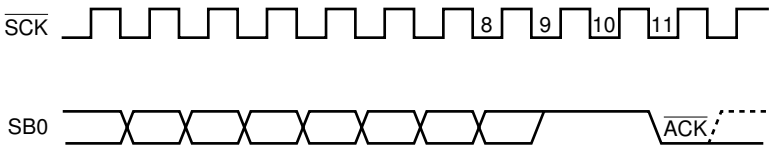


<6> Acknowledge signal ($\overline{\text{ACK}}$)

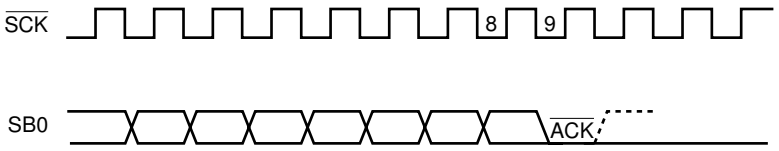
This signal indicates the reception of data. The master CPU has a built-in hardware component to detect the ACK signal. If the ACK signal is not returned, the master CPU judges that the transmitted data has not been properly received by the slave CPU.

Fig. 11-24. Acknowledge Signal

(When output synchronously with the 11th clock of $\overline{\text{SCK}}$)



(When output synchronously with the 9th clock of $\overline{\text{SCK}}$)



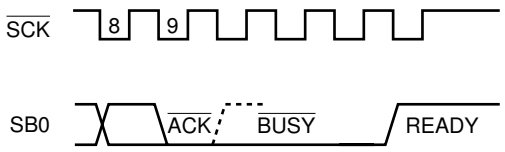
<7> Busy signal ($\overline{\text{BUSY}}$)

The slave CPU sends this signal to the master CPU to notify that it is in the process of preparing to transmit and receive data. While the master CPU is sending the BUSY signal, the master CPU continues with the serial clock output but does not transfer data.

<8> Ready signal (READY)

The slave CPU sends this signal to the master CPU to indicate that it is ready to transmit and receive data.

Fig. 11-25. Busy Signal and Ready Signal



(2) Data frame structure in SBI mode

In the SBI mode, one frame of serial data is configured as follows:

(REL) + (CMD) + 8-bit data section + $\overline{\text{ACK}}$ + ($\overline{\text{BUSY}}$)

Table 11-1. Signals Used in SBI Mode (1/2)

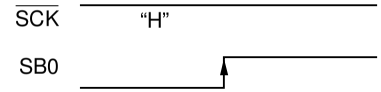
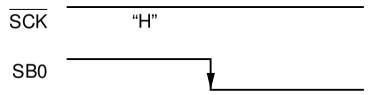
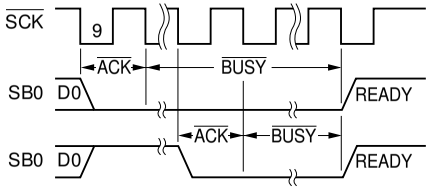
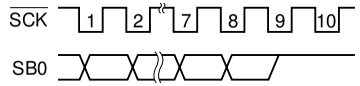
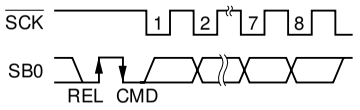
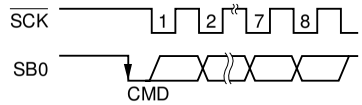
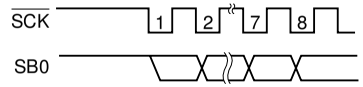
Signal name	Output device	Definition	Timing chart	Output conditions	Influence on flags	Meaning
Bus release signal (REL)	Master	Rising edge of the signal on SB0 when $\overline{SCK}$ = 1		• RELT is set.	• RELD is set. • CMDD is cleared.	Indicates the completion of transfer to the slave CPU.
Command signal (CMD)	Master	Falling edge of the signal on SB0 when $\overline{SCK}$ = 1		• CMDT is set.	• CMDD is set.	i) Address is transmitted after the REL signal is output. ii) Command is transmitted when the REL signal is not output.
Acknowledge signal ($\overline{ACK}$)	Slave	Low-level signal on SB0 during one $\overline{SCK}$ clock after completion of serial data reception.	(Output of synchronous busy signal) 	• ACKE = 1 • ACKT is set.	• ACKD is set.	Indicates the completion of data reception.
Busy signal ($\overline{BUSY}$)	Slave	[Synchronous busy signal] Low-level signal applied on SB0 following the $\overline{ACK}$ signal.		• BSYE = 1	–	Indicates that the slave CPU cannot receive serial data because it is busy.
Ready signal (READY)	Slave	High-level applied on SB0 before and after serial data transfer.		• BSYE = 0 • SIO data write instruction (transfer start instruction) is executed.	–	Indicates that the slave CPU is ready to receive data.

Table 11-1. Signals Used in SBI Mode (2/2)

Signal name	Output device	Definition	Timing chart	Output conditions	Influence on flags	Meaning
Serial clock ($\overline{SCK}$)	Master	Synchronizing clock for output of address, command, data, and $\overline{ACK}$ and BUSY signals. The address, command, and data sections are transferred within the first 8 clock pulses.		SIO data write instruction (serial data transfer start instruction) is executed when CTXE is 1.Note 3	CSIIFNote 1 is set at the rotating edge of the 8th clock pulse.Note 2	Provides timing for signal output to the serial data bus.
Address (A0 to A7)	Master	8-bit data section that follows the REL and CMD signals. Transferred in synchronization with $\overline{SCK}$.			Note 2	Slave CPU address on the serial data bus.
Command (D0 to D7)	Master	8-bit data section output after the CMD signal when the REL signal is not output. Transferred in synchronization with $\overline{SCK}$.			None	Instruction messages for the slave CPU.
Data (D0 to D7)	Master/ slave	8-bit data output when the REL and CMD signals are not output. Transferred in synchronization with $\overline{SCK}$.			None	Actual data processed by master and slave CPUs.

Notes 1. Interrupt request flag that is set in response to serial data transmission/reception interrupt (INTCSI)

2. When WUP is 0, CSIIF is always set at the rising edge of the 8th clock pulse of the $\overline{SCK}$ signal.
When WUP is 1, CSIIF is set only in response to address reception.

3. If the slave CPU is $\overline{BUSY}$, data transfer starts when it is READY.

11.6.2 Controlling and detecting the serial bus status

The status of the serial bus can be controlled and detected using the serial bus interface control register (SBIC).

- **Controlling and detecting the serial bus status**

  In the SBI mode, the SBIC register is used to control and detect the serial bus status.

  The SBIC register is an 8-bit register that consists of bits to control the bus status and flags to indicate the status of input data from the serial bus.

  Both read and write operations are possible, however, with 8-bit and single-bit manipulation instructions. Notice that some of the bits are read-only or write-only. Reading a write-only bit results in the return of a 0.

  When the $\overline{\text{RESET}}$ signal is input, the SBIC register is set to 00H.

  Fig. 11-26 shows the format of the SBIC register.

  Fig. 11-27 to 11-31 show the operation of each bit.

  Although the explanations given in the following figures assume the use of the SB0 pin for serial data input and output, the same bit operations are performed when using the SB1 pin.

Fig. 11-26. Format of Serial Bus Interface Control Register

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	When reset
SBIC	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	FF82H	00H

RELT	Bus release signal (REL) of trigger output control
W	
0	Not output
1	Output

CMDT	Command signal (CMD) of trigger output control
W	
0	Not output
1	Output

RELD	Bus release signal (REL) of detection
R	
0	Not detected
1	Detected (The serial bus changes its level from low to high while SCK remains at high level.)

CMDD	Command signal (CMD) of detection
R	
0	Not detected
1	Detected (The serial bus changes its level from high to low while SCK remains at high level.)

ACKT	Acknowledge signal ($\overline{ACK}$) of trigger output control
W	
0	Not output
1	Output

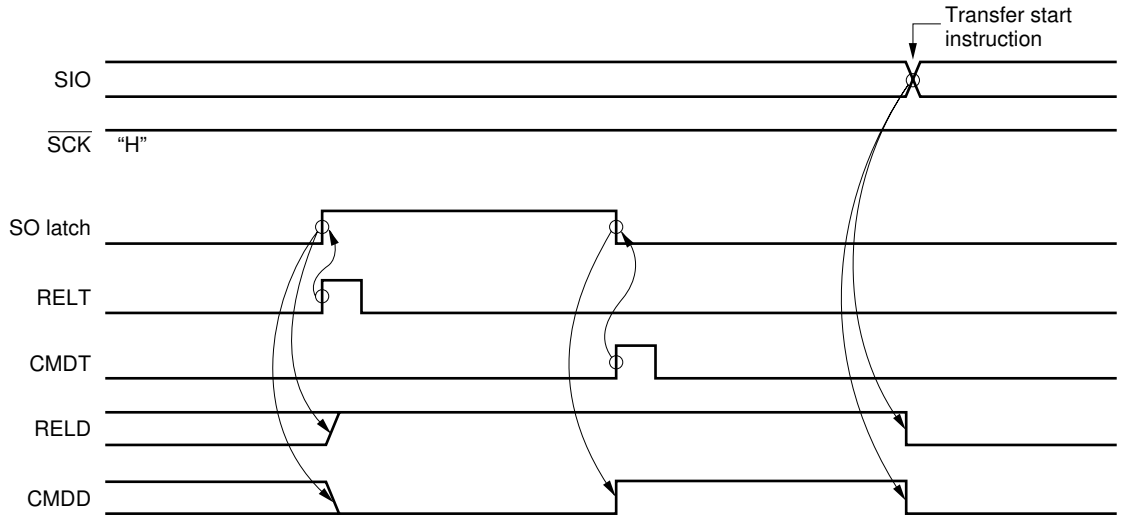
ACKE	Acknowledge signal ($\overline{ACK}$) of automatic output control
R/W	
0	Disabled
1	Enabled

ACKD	Acknowledge signal ($\overline{ACK}$) of detection
R	
0	Not detected
1	Detected

BSYE	Synchronous busy signal ($\overline{BUSY}$) of automatic output control
R/W	
0	Disabled
1	Enabled

Remark R/W : Both read and write operations are enabled.
R : Read only
W : Write only

Fig. 11-27. Operations of RELT, CMDT, RELD and CMDD



Caution Do not operate the RELT or CMDT bit during data transmission or reception.

Fig. 11-28. Operation of ACKT

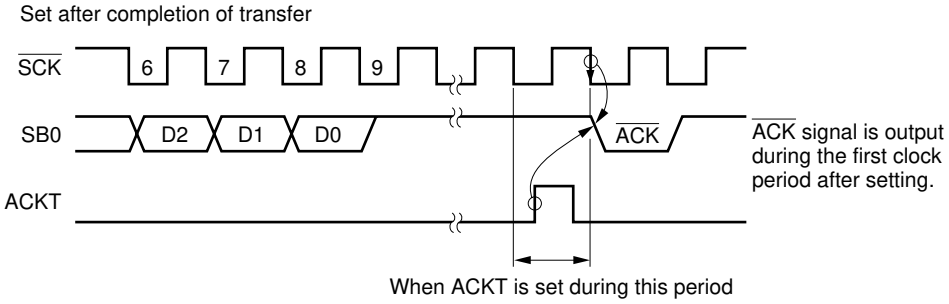
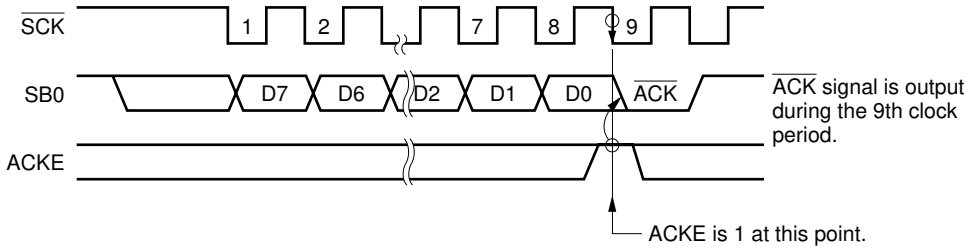
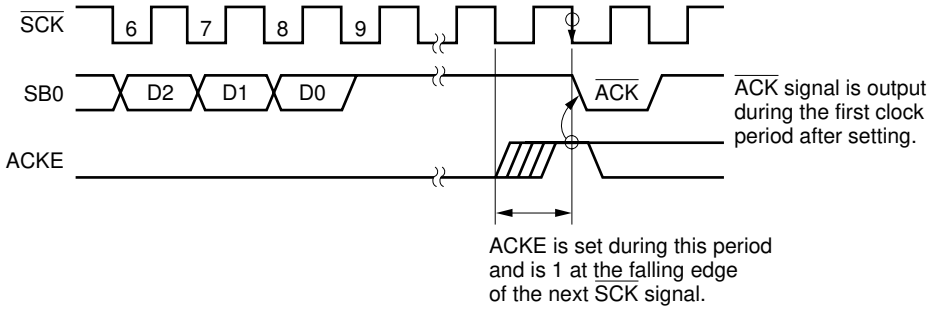


Fig. 11-29. Operation of ACKE

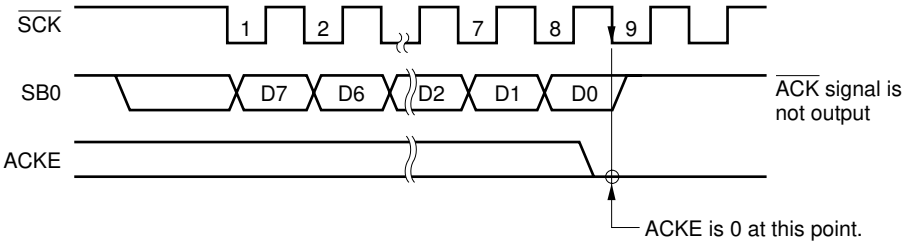
a. When ACKE is 1 at the time of data transfer



b. When ACKE is set after completion of transfer



c. When ACKE is 0 at transfer completion



d. When ACKE remains 1 for too short a period

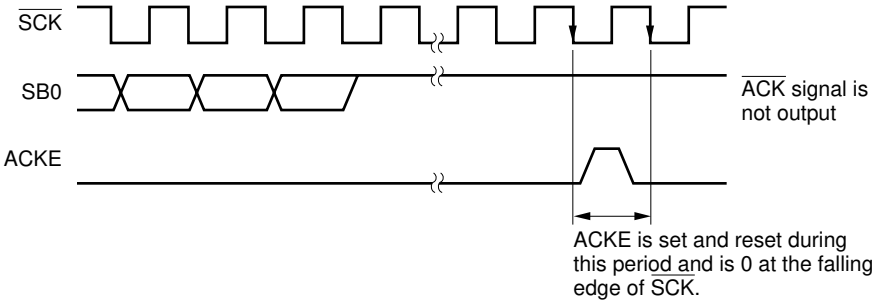
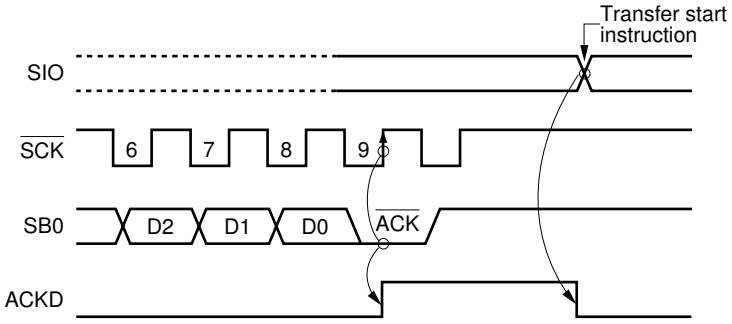
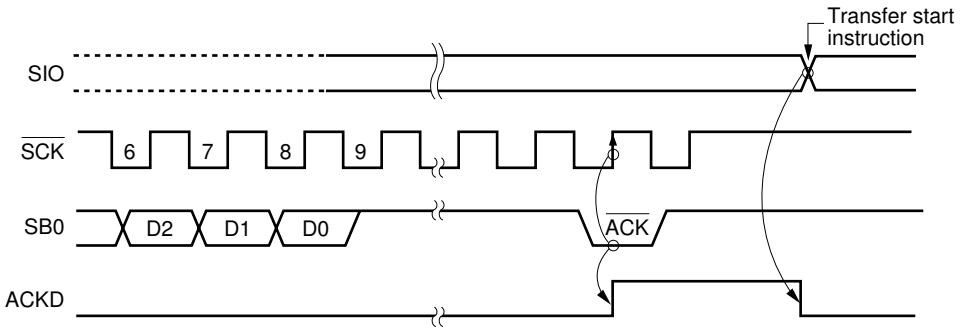


Fig. 11-30. Operation of ACKD

a. When the $\overline{ACK}$ signal is output during the 9th clock period of $\overline{SCK}$



b. When the $\overline{ACK}$ signal is output after the 9th clock period of $\overline{SCK}$



c. Timing for clearing when a transfer start instruction is issued during BUSY state

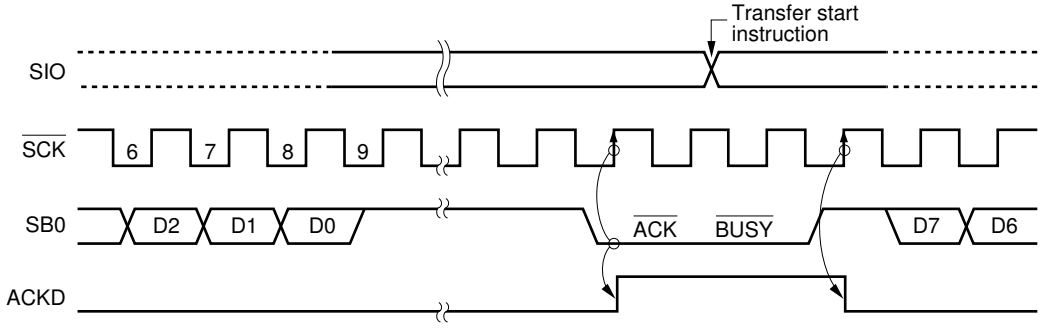
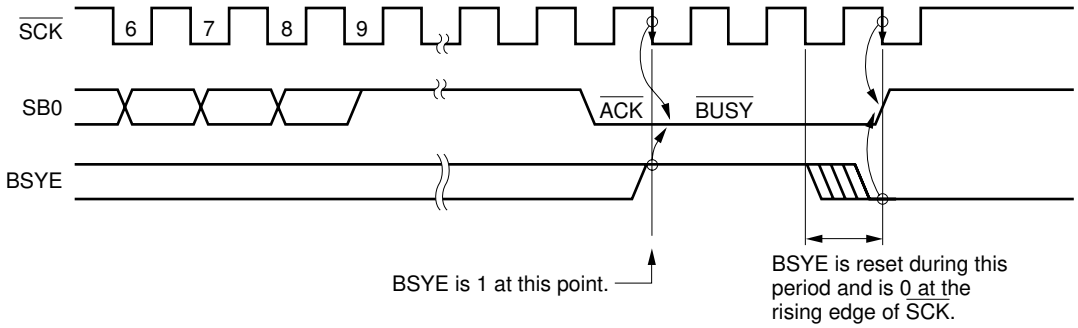


Fig. 11-31. Operation of BSYE



11.6.3 Transmitting and receiving data in SBI mode

After controlling the serial bus status with the serial bus interface control register (SBIC), transmission and reception can be performed by writing data to the shift register (SIO).

Although the following text assumes the use of the SB0 pin, the same operation can be performed using the SB1 pin.

(1) Activating transmission

When the CTXE0 bit of the clock synchronous serial interface mode register 0 (CSIM0) is set to 1, writing data to the SIO0 register activates transmission.

The SIO0 register transfers its data in synchronization with the falling edge of the serial clock ($\overline{SCK}$). Data is output from the SB0 pin, with the MSB as the start bit. After transmission, a 0 is written in the SIO0 register, and interrupt request INTCSI0 is generated.

(2) Activating reception

Reception can be activated in the following two ways:

<1> When CTXE0 bit is 0, change the status of the CRXE0 bit of the CSIM0 register from 0 (reception disabled) to 1 (reception enabled).

<2> When the CRXE0 bit is set to 1, read received data from the SIO0 register.

Attempting to set the CRXE0 bit to 1 when it is already set to 1 does not activate reception. Furthermore, attempting to set the CRXE0 bit to 1 from 0 when CTXE0 bit is 1 does not activate reception.

The 8-bit data that is input to the SB0 pin at the rising edge of the serial clock ($\overline{SCK}$) is latched by the SIO0 register. Then, interrupt request INTCSI0 is generated.

(3) Activating both transmission and reception

To activate both transmission and reception, write data to the SIO0 register when the CTXE0 and CRXE0 bits of the CSIM0 register are both set to 1.

The data on the serial bus is written to the SIO0 register unchanged. The bus status (e.g. bus contention) can be checked by comparing the data written in the SIO0 register against the data input from the serial bus.

Remark INTCSI0 ... Vector table address: 0030H (TPF = 0), 8030H (TPF = 1)
Macro service control word address: FE30H

Caution When transferring data in the SBI mode, be sure to specify the pin and serial clock before setting the CTXE0 and CRXE0 bits.

Fig. 11-32. Setting of CSIM0 Register (Transmission/Reception Enabled)

Symbol	<7>	<6>	<5>	<4>	3	2	1	0	Address	When reset	R/W
CSIM0	CTXE0	CRXE0	WUP0	MOD02	MOD01	MOD00	CLS01	CLS00	FF80H	00H	R/W

CLS01	CLS00	Setting of serial clock		$\overline{SCK}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD01	MOD00	Operation mode	Start bit	Number of pins	Pin name
0	0	3-wire serial I/O mode	MSB	3	SO00/SB0, $\overline{SI00/SB1}$, $\overline{SCK00}$
0	1		LSB		
1	0	SBI mode	MSB	2	SO00/SB0, (or $\overline{SI00/SB1}$), $\overline{SCK00}$
1	1	Setting prohibited			

MOD02	Setting of pin for transmission and reception
0	SO00/SB0 pin is used. (SI00/SB1 pin goes to high-impedance state.)
1	SI00/SB1 pin is used. (SO00/SB0 pin goes to high-impedance state.)

WUP0	Wake-up function control
0	Interrupt request signal is generated for each serial transfer in every mode.
1	Interrupt request signal is generated only when an address is received in SBI mode. The serial output pin goes to high-impedance state.

CRXE0	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE0	Data transmission
0	Disabled (Output buffers of SO00/SB0 and SI00/SB1 pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

Fig. 11-33. Address Transfer from Master to Slave Device

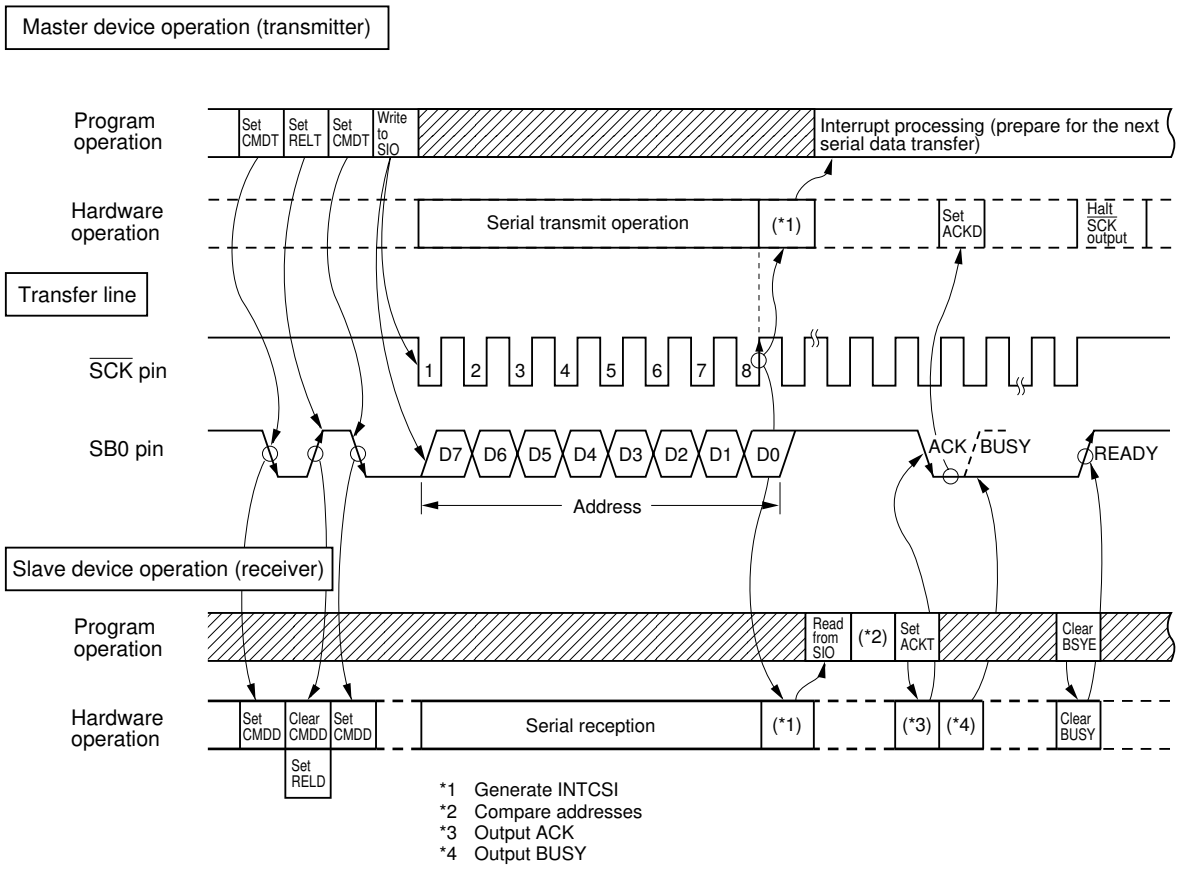


Fig. 11-34. Command Transfer from Master to Slave Device

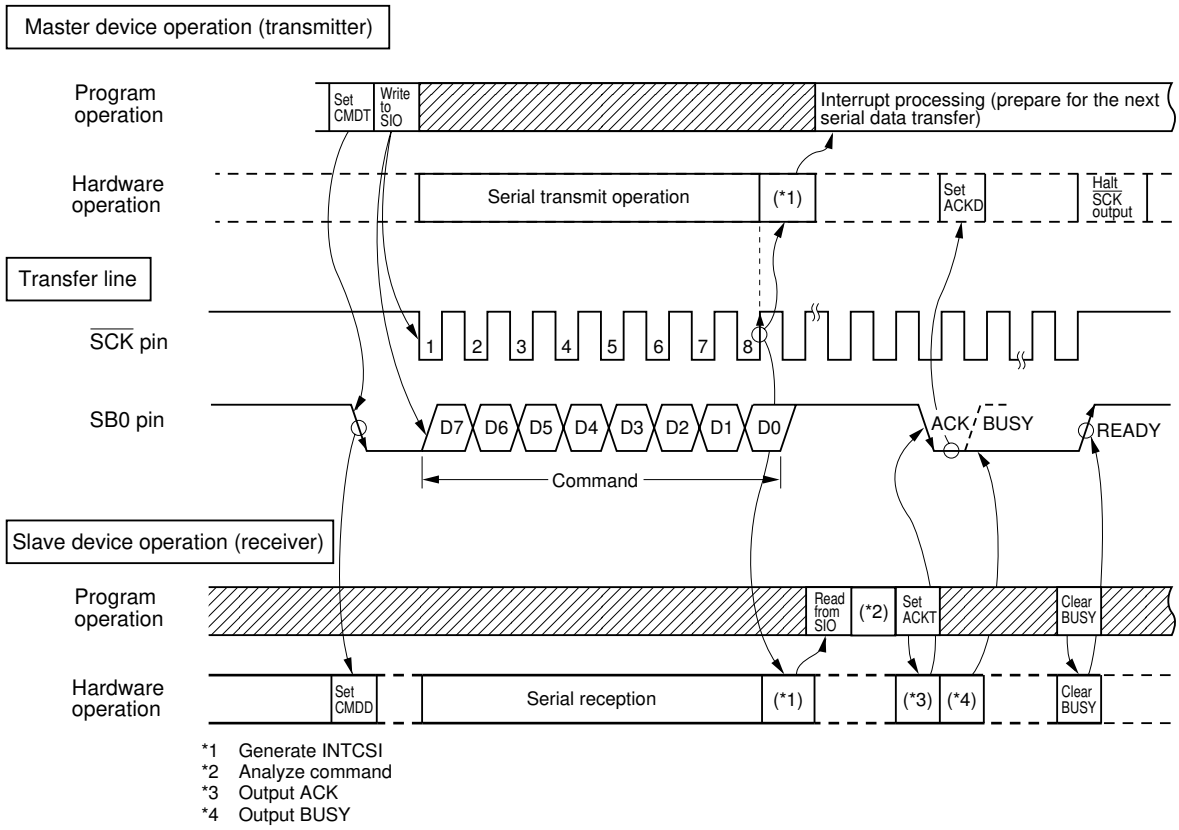


Fig. 11-35. Data Transfer from Master to Slave Device

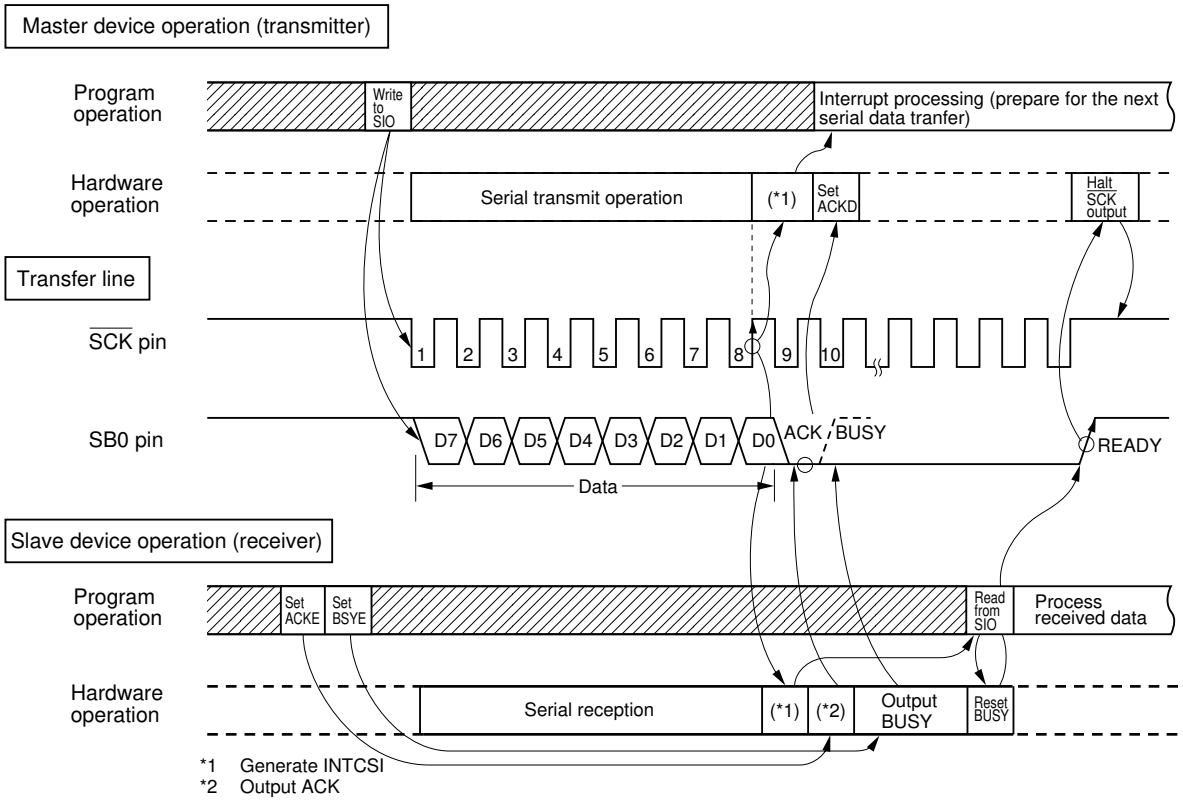
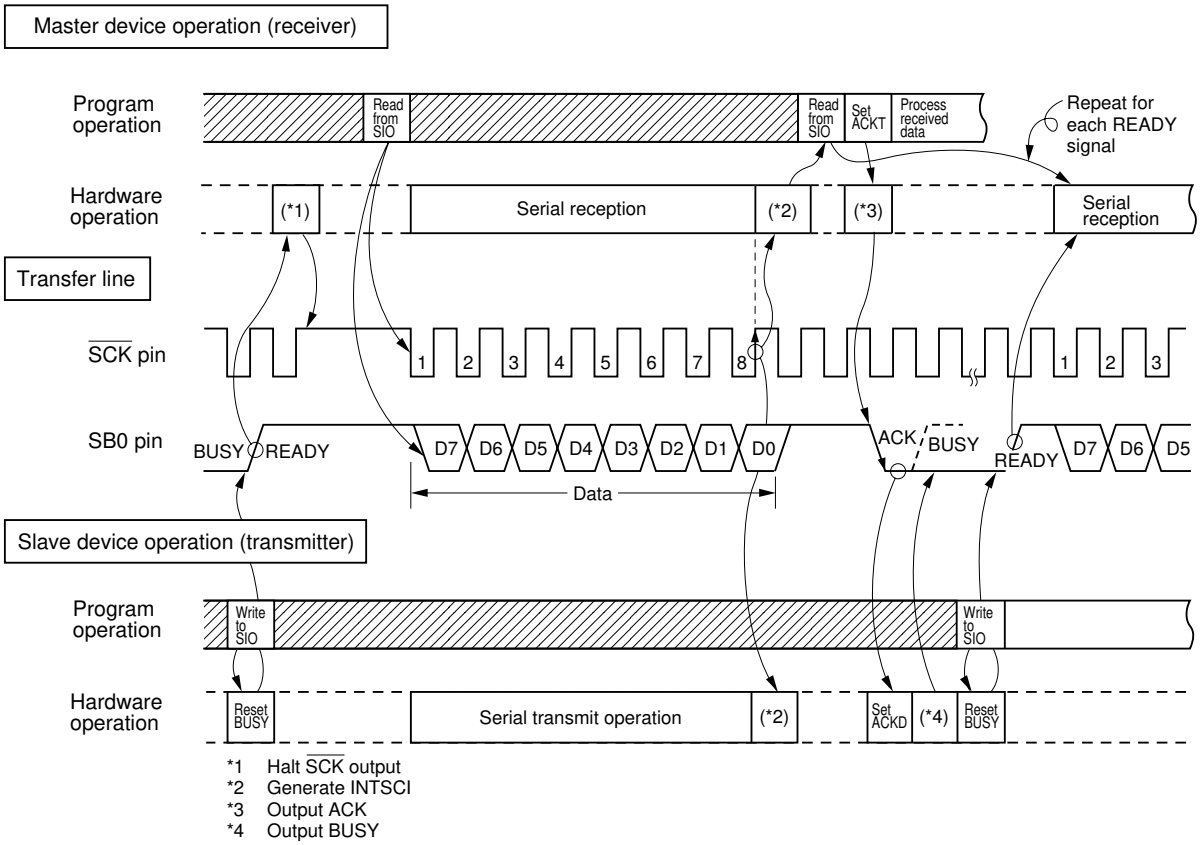


Fig. 11-36. Data Transfer from Slave to Master Device



11.6.4 Activating operation only when an address is received

The wake-up function, which activates the operation only in response to the reception of an address, ensures efficient use of the processing power of the slave CPUs.

- **Use the wake-up function to improve the efficiency of the slave CPU processing**

  The SBI mode allows the use of the wake-up function, which generates interrupt request INTCSI0 only when an address is received.

  In a system having a serial interface configuration as shown in Fig. 11-37, while the master CPU is engaged in serial data communication with one of the slave CPUs, the other slaves can perform their own operations. Without the wake-up function, an interrupt occurs each time data is received, affecting the operations of all the slave CPUs. The wake-up function eliminates this inconvenience and improves the efficiency of the slave CPU processing.

  During the interrupt occurring in response to the reception of an address, the CPU compares the received address against its own address and, when they match, releases the wake-up function to proceed with data transfer. If the addresses fail to match, the wake-up function continues to be valid.

Remark INTCSI0 ... Vector table address: 0030H (TPF = 0), 8030H (TPF = 1)
Macro service control word address: FE30H

Fig. 11-37. Example of System Configuration in SBI Mode

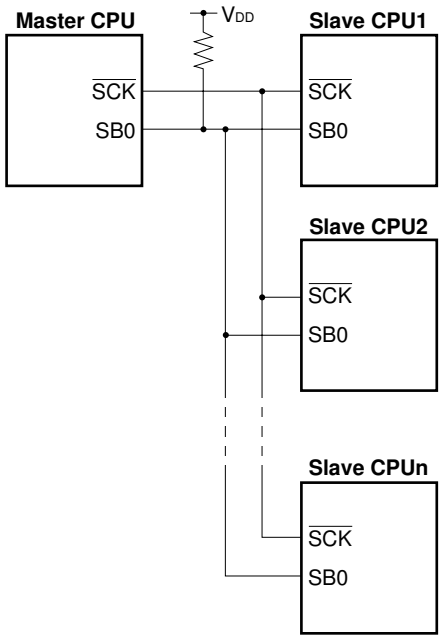


Fig. 11-38. Setting of CSIM0 Register (Wake-Up Function)

Symbol	<7>	<6>	<5>	<4>	3	2	1	0	Address	When reset	R/W
CSIM0	CTXE0	CRXE0	WUP0	MOD02	MOD01	MOD00	CLS01	CLS00	FF80H	00H	R/W

CLS01	CLS00	Setting of serial clock		$\overline{\text{SCK}}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD01	MOD00	Operation mode	Start bit	Number of pins	Pin name
0	0	3-wire serial I/O mode	MSB	3	SO00/SB0, SI00/SB1, $\overline{\text{SCK00}}$
0	1		LSB		
1	0	SBI mode	MSB	2	SO00/SB0, (or SI00/ SB1), $\overline{\text{SCK00}}$
1	1	Setting prohibited			

MOD02	Setting of pin for transmission and reception
0	SO00/SB0 pin is used. (SI00/SB1 pin goes to high-impedance state.)
1	SI00/SB1 pin is used. (SO00/SB0 pin goes to high-impedance state.)

WUP0	Wake-up function control
0	Interrupt request signal is generated for each serial transfer in every mode.
1	Interrupt request signal is generated only when an address is received in SBI mode. The serial output pin goes to high-impedance state.

CRXE0	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE0	Data transmission
0	Disabled (Output buffers of SO00/SB0 and SI00/SB1 pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

[MEMO]

CHAPTER 12 CLOCK SYNCHRONOUS SERIAL INTERFACE (WITH PIN SWITCHING FUNCTION)

The μPD78356 offers the pin switching function for the clock synchronous serial interface that operates in the 3-wire serial I/O mode. In the clock synchronous serial interface with the pin switching function, the pins used in the 3-wire serial I/O mode can be switched using software.

This interface operates independently of the asynchronous serial interface and the clock synchronous serial interface without the pin switching function.

12.1 Configuration of Clock Synchronous Serial Interface (with Pin Switching Function)

The clock synchronous serial interface is controlled by clock synchronous serial interface mode register 1 (CSIM1). Data transmitted and received can be written to and read from the shift register (SIO1).

(1) Clock synchronous serial interface mode register 1 (CSIM1)

The CSIM1 register is an 8-bit register that controls the operation of the clock synchronous serial interface with the pin switching function.

Both data reading and writing are possible with 8-bit and single-bit manipulation instructions. When the $\overline{\text{RESET}}$ signal is input, CSIM1 is set to 00H.

(2) Shift register (SIO1)

The SIO1 register is an 8-bit register that converts serial data to parallel data and vice versa. The register is used for both transmitting and receiving data.

Data is shifted in (received) or shifted out (transmitted) from the MSB or LSB side.

The transmitting and receiving of data is actually performed by writing data to and reading data from the SIO1 register.

For data writing and reading, 8-bit manipulation instructions are used.

When the $\overline{\text{RESET}}$ signal is input, the contents of SIO1 become undefined.

(3) Serial clock selector

This selector selects the serial clock to be used.

(4) Serial clock control circuit

This control circuit controls the supply of serial clock to the shift register. When an internal clock is being used, it also controls the clock output to the $\overline{\text{SCK10}}$ or $\overline{\text{SCK11}}$ pin.

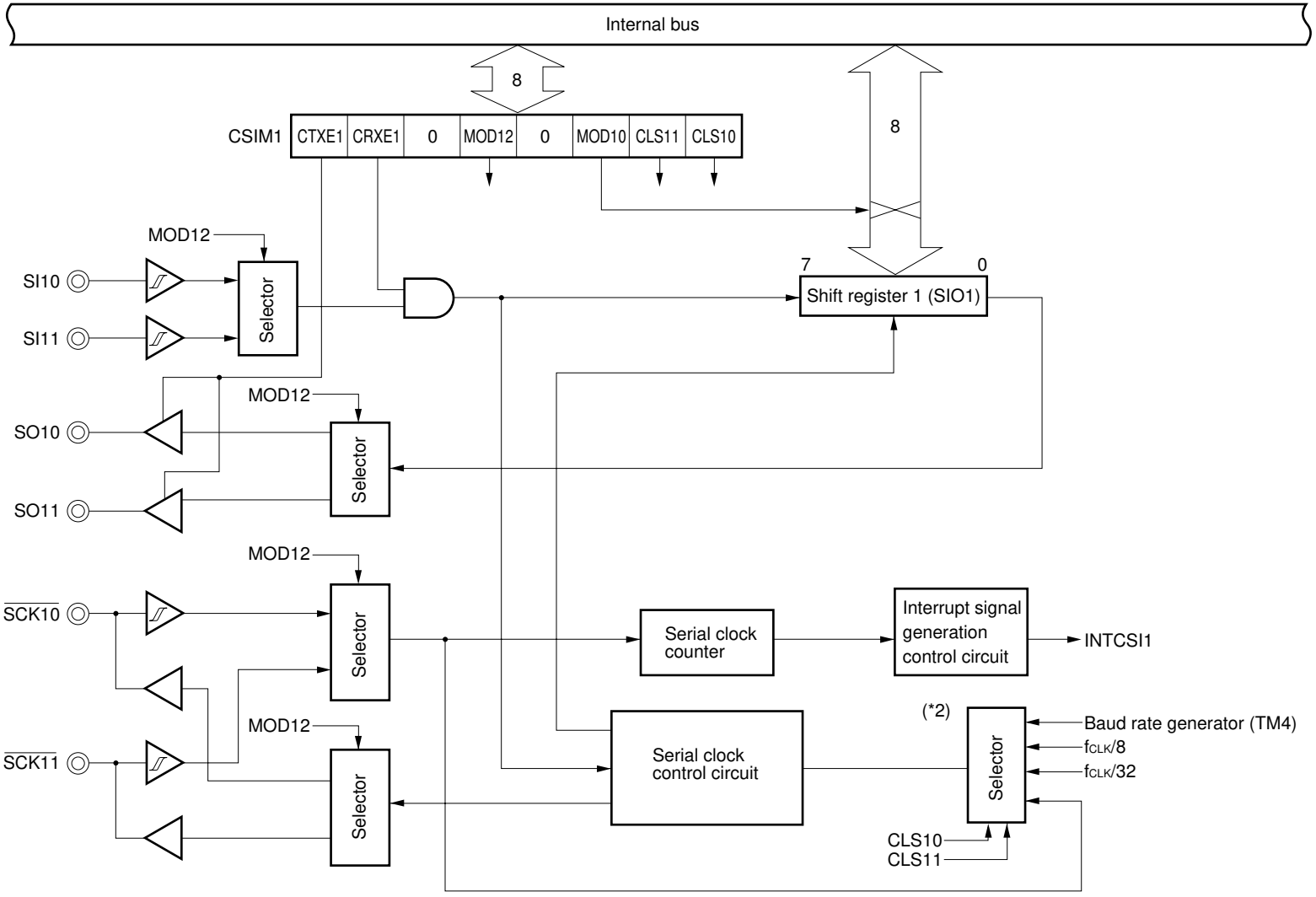
(5) Serial clock counter

This counter counts serial clocks being output and input during data transmission and reception to check whether 8-bit data has been transmitted and received.

(6) Interrupt signal generation control circuit

This control circuit controls whether to make an interrupt request when the serial clock counter has counted eight serial clocks.

Fig. 12-1. Block Diagram of Clock Synchronous Serial Interface (with Pin Switching Function)



12.2 Setting Pins for Serial Transmission

The SO10, SI10, $\overline{\text{SCK10}}$ pins or SO11, SI11, $\overline{\text{SCK11}}$ pins, which are also used as general purpose ports, must be set to the control mode before beginning serial transmission.

(1) Setting pins for serial transmission

The clock synchronous serial interface with the pin switching function uses two different sets of pins (SO10, SI10, $\overline{\text{SCK10}}$ and SO11, SI11, $\overline{\text{SCK11}}$). These pins also serve as general purpose ports P100, P101, P102 and P103, P104, P105, respectively. Therefore, before beginning serial transmission, these pins must be set to the control mode in the port 10 mode control register (PMC10).

Fig. 12-2. Port 10 Mode Control Register Format

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PMC10	PMC107	PMC106	PMC105	PMC104	PMC103	PMC102	PMC101	PMC100	FF4AH	00H	R/W

PMC100	Control mode setting for P100 pin
0	I/O port mode
1	SO10 output mode

PMC101	Control mode setting for P101 pin
0	I/O port mode
1	SI10 input mode

PMC102	Control mode setting for P102 pin
0	I/O port mode
1	$\overline{\text{SCK10}}$ input and output mode

PMC103	Control mode setting for P103 pin
0	I/O port mode
1	SO11 input and output mode

PMC104	Control mode setting for P104 pin
0	I/O port mode
1	SI11 input mode

PMC105	Control mode setting for P105 pin
0	I/O port mode
1	$\overline{\text{SCK11}}$ input and output mode

PMC106	Control mode setting for P106 pin
0	I/O port mode
1	TIUD input mode

PMC107	Control mode setting for P107 pin
0	I/O port mode
1	TCUD input mode

(2) Reading pin levels

When port 10 (P10) has been set to the control mode using the port 10 mode control register (PMC10), the read instructions of P10 can read the following information:

(a) When a bit of the port 10 mode register (PM10) is set to 1

- The corresponding pin level can be read.

(b) When a bit of the PM10 is set to 0 (reset)

- The level of the internal signal can be read.

Reading pin levels allows pins checks, including the check for serial bus contention.
Writing data in P10 does not change any pin level. (Data is written in the P10 output buffer.)

Fig. 12-3. Port 10 Mode Register Format

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
PM10	PM107	PM106	PM105	PM104	PM103	PM102	PM101	PM100	FF2AH	FFH	R/W

PM10n	I/O mode setting for P10n pin (n = 0 to 7)
0	Output mode (Output buffer is on)
1	Input mode (Output buffer is off)

12.3 Switching Pins in 3-Wire Serial I/O Mode

In the clock synchronous serial interface operating in the 3-wire serial I/O mode, the pin switching function switches the pins using software.

The clock synchronous serial interface mode register 1 (CSIM1) can be used to switch between two sets of three pins.

- **Switching pins used for serial data transfer in 3-wire serial I/O mode**

  In the clock synchronous serial interface with the pin switching function, the pins used for serial data transfer in the 3-wire serial I/O mode can be switched using software.

  The switching of the pins is performed using the MOD12 bit of the CSIM1.

  The pins not being used are placed in a high-impedance state.

  (a) When $\overline{\text{SCK10}}$, SO10 and SI10 pins are specified

  To specify the $\overline{\text{SCK10}}$, SO10 and SI10 pins for the 3-wire serial I/O mode transmission, set the MOD12 bit of the CSIM1 register to 0 (reset).

  When not specified, the pins are placed in a high-impedance state.

  (b) When $\overline{\text{SCK11}}$, SO11 and SI11 pins are specified

  To specify the $\overline{\text{SCK11}}$, SO11 and SI11 pins for the 3-wire serial I/O mode transmission, set the MOD12 bit of the CSIM1 register to 1.

  When not specified, the pins are placed in a high-impedance state.

Fig. 12-4. Setting of CSIM1 Register (Pin Switching)

Symbol	<7>	<6>	5	<4>	3	2	1	0	Address	When reset	R/W
CSIM1	CTXE1	CRXE1	0	MOD12	0	MOD10	CLS11	CLS10	FF90H	00H	R/W

CLS11	CLS10	Setting of serial clock		$\overline{\text{SCK}}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD10	Operation mode	Start bit	Number of pins	Pin name
0	3-wire serial I/O mode	MSB	3	SO10, SI10, $\overline{\text{SCK10}}$ (or SO11,SI11, $\overline{\text{SCK11}}$)
1		LSB		

MOD12	Setting of pin for transmission and reception
0	SO10, SI10 and $\overline{\text{SCK10}}$ pins are used. (SO11, SI11 and $\overline{\text{SCK11}}$ pins go to high-impedance state.)
1	SO11, SI11 and $\overline{\text{SCK11}}$ pins are used. (SO10, SI10 and $\overline{\text{SCK10}}$ pins go to high-impedance state.)

CRXE1	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE1	Data transmission
0	Disabled (Output buffers of SO10 and SI10 (or SO11 and SI11) pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

12.4 Setting Baud Rate

The baud rate generator output, internal clock f_{CLK}/8 or f_{CLK}/32, or external clock can be selected as the serial clock. The baud rate generator output can be set at any desired baud rate, independently of the operation clock frequency.

(1) Baud rate equal to serial clock

In the clock synchronous serial interface with the pin switching function, received data is sampled at the rising edge of the serial clock, which means the frequency of the serial clock is equal to the baud rate.

(2) Selecting serial clock

The serial clock can be selected using the CLS11 and CLS10 bits of the clock synchronous serial interface mode register 1 (CSIM1).

When the external clock is selected, the $\overline{SCK10}$ or $\overline{SCK11}$ pin is used for the input of the serial clock from the device with which communication is being performed.

With the 16-MHz internal system clock, f_{CLK}/8 provides a baud rate of 2 Mbps and f_{CLK}/32 provides a baud rate of 500 Kbps.

Remark The baud rate generator is also used for the asynchronous serial interface and the clock synchronous serial interface with the pin switching function. (See **10.4** and **11.3**.)

Fig. 12-5. Setting of CSIM1 Register (Serial Clock)

Symbol	<7>	<6>	5	<4>	3	2	1	0	Address	When reset	R/W
CSIM1	CTXE1	CRXE1	0	MOD12	0	MOD10	CLS11	CLS10	FF90H	00H	R/W

CLS11	CLS10	Setting of serial clock		$\overline{\text{SCK}}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD10	Operation mode	Start bit	Number of pins	Pin name
0	3-wire serial I/O mode	MSB	3	SO10, SI10, $\overline{\text{SCK10}}$ (or SO11,SI11, $\overline{\text{SCK11}}$)
1		LSB		

MOD12	Setting of pin for transmission and reception
0	SO10, SI10 and $\overline{\text{SCK10}}$ pins are used. (SO11, SI11 and $\overline{\text{SCK11}}$ pins go to high-impedance state.)
1	SO11, SI11 and $\overline{\text{SCK11}}$ pins are used. (SO10, SI10 and $\overline{\text{SCK10}}$ pins go to high-impedance state.)

CRXE1	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE1	Data transmission
0	Disabled (Output buffers of SO10 and SI10 (or SO11 and SI11) pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

Caution Do not switch serial clocks during data transmission. Since the switching is performed asynchronously with the serial clock being supplied, doing this during data transmission can produce a serial clock of undefined frequency.

12.4.1 Configuration of baud rate generator

The baud rate generator uses timer 4 (TM4) in the real-time pulse unit (RPU), and is controlled by the timer control register 2 (TMC2) and 10-bit compare register (CM40).

Fig. 12-6 shows the configuration of the baud rate generator.

(1) Timer control register 2 (TMC2)

TMC2 is an 8-bit register used for TM4 count clock selection and baud rate generator operation control.
Both data reading and writing are possible with 8-bit and single-bit manipulation instructions.
When the $\overline{\text{RESET}}$ signal is input, TMC2 is set to 00H.
Fig. 12-7 shows the TMC2 register format.

(2) Selector

The selector selects the count clock of the 10-bit timer (TM4) according to the contents of the TMC2 register.

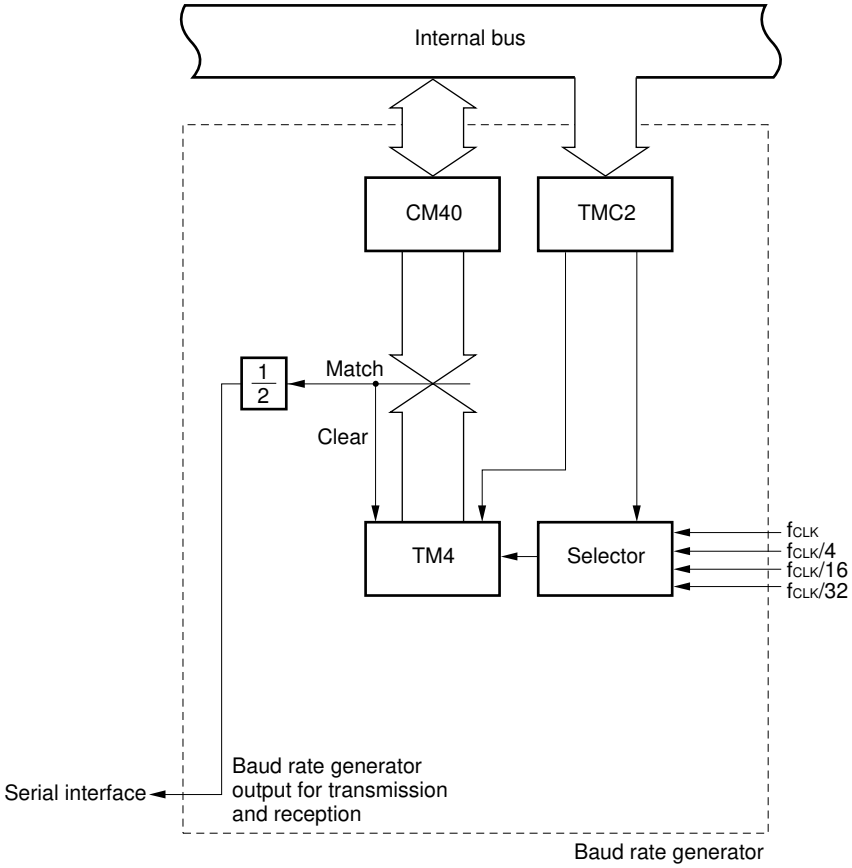
(3) 10-bit timer (TM4)

TM4 is a 10-bit timer that counts the count clocks selected by the selector.
When a matching signal is issued by the CM40 register, this timer is cleared to zero at the next count clock.
The TMC2 register controls the start and end of counting.
Only data reading is possible with 16-bit manipulation instructions.

(4) 10-bit compare register (CM40)

CM40 compares its data against the contents of TM4 and, when they match, issues a matching signal. TM4 is cleared to zero in response to the matching signal.
The signal output by the baud rate generator has a frequency 1/2 of the matching signal frequency.
Both data reading and writing are possible with 16-bit manipulation instructions.
When the $\overline{\text{RESET}}$ signal is input, the contents of CM40 become undefined.

Fig. 12-6. Block Diagram of Baud Rate Generator



12.4.2 Setting a desired baud rate

Any desired baud rate can be selected for the serial clock by setting the timer control register 2 (TMC2) and 10-bit compare register (CM40).

(1) Baud rate equal to serial clock

In the clock synchronous serial interface, received data is sampled at the rising edge of the serial clock, which means the frequency of the serial clock is equal to the baud rate.

(2) Setting a desired baud rate

The baud rate to be set can be obtained by the equation below. Set the CM40 register and TM4 counter values for the desired baud rate, then start the operation of the baud rate generator.

Equation to obtain baud rate

$$\text{Baud rate (bps)} = \frac{f_{CLK}}{2^n} \times \frac{1}{(m+1)} \times \frac{1}{2}$$

f_{CLK} : Internal system clock (external oscillator frequency $f_{OSC}/2$)
m : Value set in CM40 (0 to 1023)
n : Value corresponding to TMC2 value (0, 2, 4, 5)

Fig. 12-7. Timer Control Register 2 Format

Symbol	7	6	5	4	<3>	2	1	0	Address	When reset	R/W
TMC2	0	0	0	0	CE4	1	PRM41	PRM40	FF76H	04H	R/W

PRM41	PRM40	Setting of count clock (Hz) in TM4
0	0	f_{CLK}
0	1	$f_{CLK}/4$
1	0	$f_{CLK}/16$
1	1	$f_{CLK}/32$

CE4	TM4 operation control
0	Clear and halt
1	Count

Remark f_{CLK}: Internal system clock

12.5 Operation Mode of Clock Synchronous Serial Interface (with Pin Switching Function)

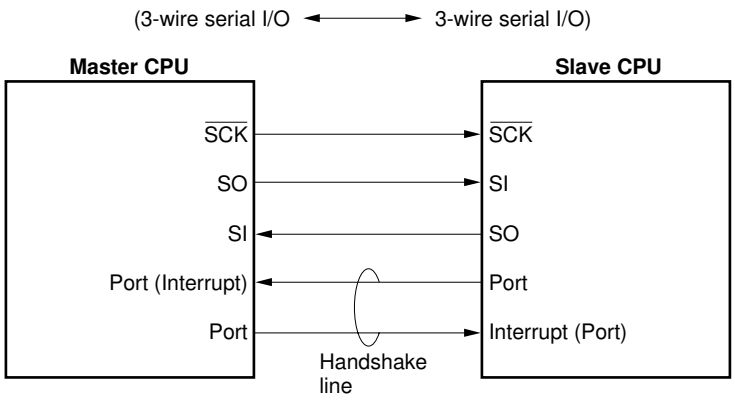
The clock synchronous serial interface with the pin switching function operates in the 3-wire serial I/O mode only.

Using the 3-wire serial I/O mode is advantageous when communicating with a device having a conventional clock synchronous serial interface.

In the μPD78356, the clock synchronous serial interface with the pin switching function operates in the 3-wire serial I/O mode only.

- **Three-wire serial I/O mode**
  In the 3-wire serial I/O mode, 8-bit data communication is carried out using three lines: serial clock ($\overline{SCK}$), serial input (SI), and serial output (SO). Using this mode is advantageous when connecting to a peripheral I/O device having a conventional clock synchronous serial interface or a display controller.
  To connect more than one device, another line is required for handshaking.

Fig. 12-8. Example of System Configuration in the 3-Wire Serial I/O Mode



12.6 Selecting Start Bit for 3-Wire Serial I/O Mode Transmission

The start bit for the 3-wire serial I/O mode transmission can be selected using the clock synchronous serial interface mode register 1 (CSIM1). Since the MSB or LSB can be selected as the start bit, communication can be performed with a variety of devices.

(1) Selecting start bit for 3-wire serial I/O mode transmission

The start bit for the 3-wire serial I/O mode transmission is selected using the MOD10 bit of the CSIM1 register. (See **Fig. 12-10**.) Since the MSB or LSB can be selected as the start bit, communication can be performed with a variety of devices.

(2) Operation timing in 3-wire serial I/O mode

In the 3-wire serial I/O mode, data is transmitted and received in blocks of eight bits, with either the MSB or LSB as the start bit (specified in the CSIM1 register). The eight-bit data block is transferred bit by bit in synchronization with the serial clock.

Transmitted data is output in synchronization with the falling edge of the $\overline{SCK10}$ or $\overline{SCK11}$ signal. Received data is sampled at the rising edge of the $\overline{SCK10}$ or $\overline{SCK11}$ signal. At the rising edge of the 8th $\overline{SCK10}$ or $\overline{SCK11}$ signal, interrupt request INTCSI1 is generated.

When the internal clock is used as the $\overline{SCK10}$ or $\overline{SCK11}$ signal, the output is halted at the rising edge of the 8th $\overline{SCK10}$ or $\overline{SCK11}$ signal. The $\overline{SCK10}$ or $\overline{SCK11}$ pin maintains the high level until the transmission or reception of the next data block is activated.

Fig. 12-9. Timing of 3-Wire Serial I/O Mode

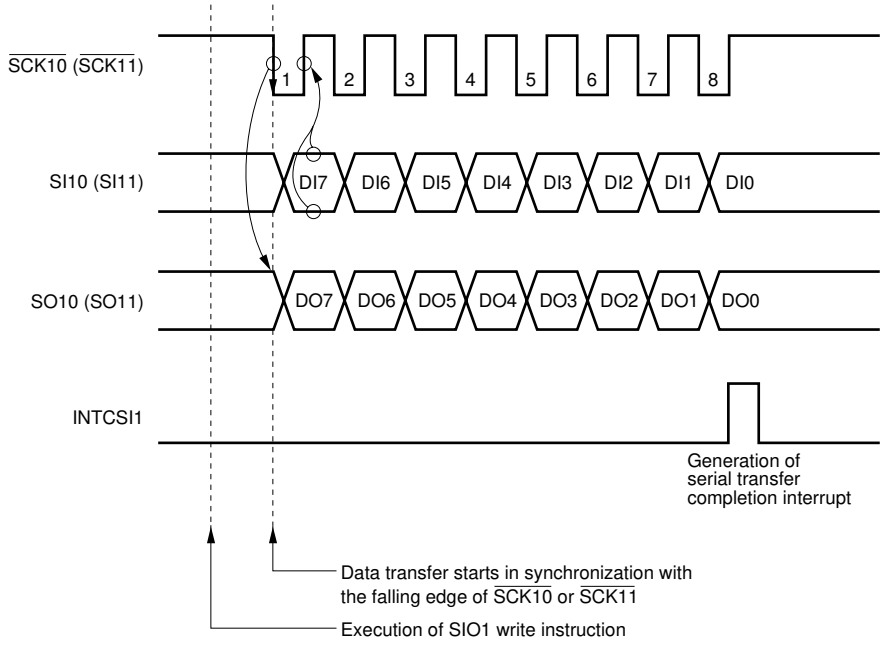


Fig. 12-10. Setting of CSIM1 Register (Start Bit for 3-Wire Serial I/O Mode)

Symbol	<7>	<6>	5	<4>	3	2	1	0	Address	When reset	R/W
CSIM1	CTXE1	CRXE1	0	MOD12	0	MOD10	CLS11	CLS10	FF90H	00H	R/W

CLS11	CLS10	Setting of serial clock		$\overline{SCK}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD10	Operation mode	Start bit	Number of pins	Pin name
0	3-wire serial I/O mode	MSB	3	SO10, SI10, $\overline{SCK10}$ (or SO11,SI11, $\overline{SCK11}$)
1		LSB		

MOD12	Setting of pin for transmission and reception
0	SO10, SI10 and $\overline{SCK10}$ pins are used. (SO11, SI11 and $\overline{SCK11}$ pins go to high-impedance state.)
1	SO11, SI11 and $\overline{SCK11}$ pins are used. (SO10, SI10 and $\overline{SCK10}$ pins go to high-impedance state.)

CRXE1	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE1	Data transmission
0	Disabled (Output buffers of SO10 and SI10 (or SO11 and SI11) pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

12.6.1 Transmitting data in 3-wire serial I/O mode

Writing data in the shift register (SIO1) after enabling transmission in the clock synchronous serial interface mode register 1 (CSIM1) activates transmission.

(1) Activating transmission

Transmission is activated by setting the CTXE1 bit of the CSIM1 register to 1 (set the CRXE1 bit to 0) and then writing transmission data to the SIO1 register. The block transfer (BLKTRS) mode of the macro service is useful in writing data to the SIO1 register.

While the CTXE1 bit is reset to 0, the output of the SO10 or SO11 pin remains in the high-impedance state.

(2) Transmitting data in synchronization with the serial clock

(a) When an internal clock is selected as the serial clock

Once transmission is activated, the $\overline{SCK10}$ or $\overline{SCK11}$ pin starts the output of the serial clock. The SIO1 register then sequentially transfers data to the SO10 or SO11 pin in synchronization with the falling edge of the serial clock.

(b) When an external clock is selected as the serial clock

Upon the activation of transmission, data is sequentially transmitted from SIO1 to the SO10 or SO11 pin in synchronization with the falling edge of the serial clock that is input to the $\overline{SCK10}$ or $\overline{SCK11}$ pin. When transmission is not activated, sending the serial clock to the $\overline{SCK10}$ or $\overline{SCK11}$ pin does not cause the SIO1 register to transfer data and the output level of the SO10 or SO11 pin does not change.

Fig. 12-11. Timing of 3-Wire Serial I/O Mode (Transmission)

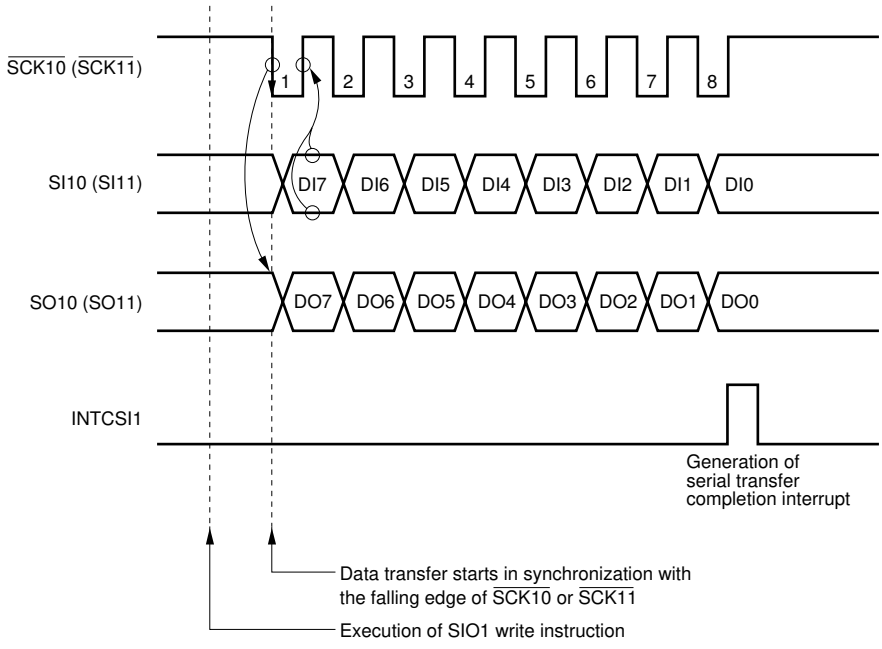


Fig. 12-12. Setting of CSIM1 Register (Transmission Enabled)

Symbol	<7>	<6>	5	<4>	3	2	1	0	Address	When reset	R/W
CSIM1	CTXE1	CRXE1	0	MOD12	0	MOD10	CLS11	CLS10	FF90H	00H	R/W

CLS11	CLS10	Setting of serial clock		$\overline{SCK}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD10	Operation mode	Start bit	Number of pins	Pin name
0	3-wire serial I/O mode	MSB	3	$\overline{SO10}$, SI10, $\overline{SCK10}$ (or $\overline{SO11}$,SI11, $\overline{SCK11}$)
1		LSB		

MOD12	Setting of pin for transmission and reception
0	SO10, SI10 and $\overline{SCK10}$ pins are used. (SO11, SI11 and $\overline{SCK11}$ pins go to high-impedance state.)
1	SO11, SI11 and $\overline{SCK11}$ pins are used. (SO10, SI10 and $\overline{SCK10}$ pins go to high-impedance state.)

CRXE1	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE1	Data transmission
0	Disabled (Output buffers of SO10 and SI10 (or SO11 and SI11) pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

12.6.2 Receiving data in 3-wire serial I/O mode

The reception is activated by setting the CRXE1 bit of the clock synchronous serial interface mode register 1 (CSIM1) to 1 (reception enabled) (when CTXE1 = 0) or, when reception is already enabled, by reading data from the shift register (SIO1).

(1) Activating reception

Reception can be activated in the following two ways:

<1> When CTXE1 bit is 0, change the setting of the CRXE1 bit of the CSIM1 register from 0 (reception disabled) to 1 (reception enabled).

<2> When the CRXE1 bit is already set to 1, read data from the SIO1 register.

The block transfer (BLKTRS) mode of the macro service is useful in reading received data from the SIO1 register. Attempting to set the CRXE1 bit to 1 when it is already set to 1 does not activate reception. Furthermore, attempting to set the CRXE1 bit to 1 from 0 when CTXE1 bit is 1 does not activate reception.

(2) Receiving data in synchronization with the serial clock

(a) When an internal clock is selected as the serial clock

Once reception is activated, the $\overline{SCK10}$ or $\overline{SCK11}$ pin starts the output of the serial clock. The data is sequentially read from the SI10 or SI11 pin to SIO1 in synchronization with the rising edge of the serial clock.

(b) When an external clock is selected as the serial clock

Once reception is activated, the data is sequentially read from the SI10 or SI11 pin to SIO1 in synchronization with the rising edge of the serial clock input to the $\overline{SCK10}$ or $\overline{SCK11}$ pin. When reception is not activated, inputting the serial clock to the $\overline{SCK10}$ or $\overline{SCK11}$ pin does not cause the SIO1 register to transfer the data.

Fig. 12-13. Timing of 3-Wire Serial I/O Mode (Reception)

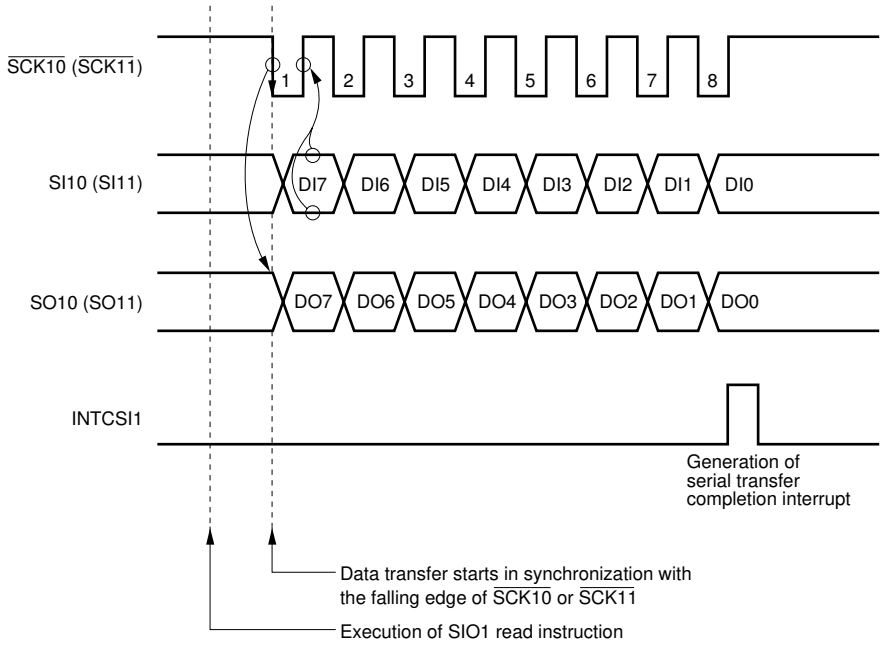


Fig. 12-14. Setting of CSIM1 Register (Reception Enabled)

Symbol	<7>	<6>	5	<4>	3	2	1	0	Address	When reset	R/W
CSIM1	CTXE1	CRXE1	0	MOD12	0	MOD10	CLS11	CLS10	FF90H	00H	R/W

CLS11	CLS10	Setting of serial clock		$\overline{SCK}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD10	Operation mode	Start bit	Number of pins	Pin name
0	3-wire serial I/O mode	MSB	3	SO10, SI10, $\overline{SCK10}$ (or SO11,SI11, $\overline{SCK11}$)
1		LSB		

MOD12	Setting of pin for transmission and reception
0	SO10, SI10 and $\overline{SCK10}$ pins are used. (SO11, SI11 and $\overline{SCK11}$ pins go to high-impedance state.)
1	SO11, SI11 and $\overline{SCK11}$ pins are used. (SO10, SI10 and $\overline{SCK10}$ pins go to high-impedance state.)

CRXE1	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE1	Data transmission
0	Disabled (Output buffers of SO10 and SI10 (or SO11 and SI11) pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

12.6.3 Transmitting and receiving data in 3-wire serial I/O mode

Enabling both transmission and reception using the clock synchronous serial interface mode register 1 (CSIM1) allows the two operations to be performed simultaneously.

(1) Activating transmission and reception

Transmission and reception can be performed simultaneously when the CTXE1 and CRXE1 bits of the CSIM1 register are both set to 1.

The transmission and reception can be activated by writing the transmitted data to the shift register 1 (SIO1) when both the CTXE1 and CRXE1 bits of the CSIM1 register become 1 (transmission/reception enable state). Attempting to set the CRXE1 bit to 1 when it is already set to 1 does not activate transmission or reception.

(2) Transmitting and receiving data in synchronization with the serial clock

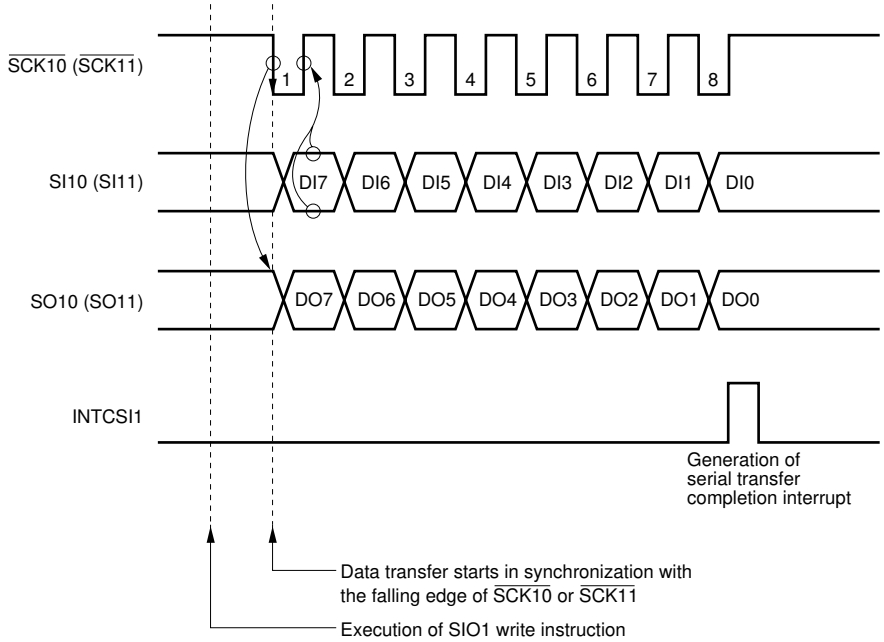
(a) When an internal clock is selected as the serial clock

Once transmission and reception are activated, the $\overline{\text{SCK10}}$ or $\overline{\text{SCK11}}$ pin starts the output of the serial clock. The SIO1 register then sequentially transfers data to the SO10 or SO11 pin in synchronization with the falling edge of the serial clock. At the same time, data is sequentially read from the SI10 or SI11 pin to SIO1 in synchronization with the rising edge of the serial clock.

(b) When an external clock is selected as the serial clock

Upon the activation of transmission and reception, data are sequentially transmitted from SIO1 to the SO10 or SO11 pin in synchronization with the falling edge of the serial clock that is input to the $\overline{\text{SCK10}}$ or $\overline{\text{SCK11}}$ pin. At the same time, data is sequentially read from the SI10 or SI11 pin to SIO1 in synchronization with the rising edge of the serial clock. When transmission and reception are not activated, sending the serial clock to the $\overline{\text{SCK10}}$ or $\overline{\text{SCK11}}$ pin does not cause the SIO1 register to transfer data and the output level of the SO10 or SO11 pin does not change.

Fig. 12-15. Timing of 3-Wire Serial I/O Mode (Transmission and Reception)



Remark INTCSI1: Vector table address: 0032H (TFP = 0), 8032H (TFP = 1)
Macro service control word address: FE32H

Fig. 12-16. Setting of CSIM1 Register (Transmission and Reception Enabled)

Symbol	<7>	<6>	5	<4>	3	2	1	0	Address	When reset	R/W
CSIM1	CTXE1	CRXE1	0	MOD12	0	MOD10	CLS11	CLS10	FF90H	00H	R/W

CLS11	CLS10	Setting of serial clock		$\overline{\text{SCK}}$ pin
0	0	External clock		Input
0	1	Internal clock	Baud rate generator/2	Output
1	0		f_{CLK}/32	
1	1		f_{CLK}/8	

MOD10	Operation mode	Start bit	Number of pins	Pin name
0	3-wire serial I/O mode	MSB	3	SO10, SI10, $\overline{\text{SCK10}}$ (or SO11,SI11, $\overline{\text{SCK11}}$)
1		LSB		

MOD12	Setting of pin for transmission and reception
0	SO10, SI10 and $\overline{\text{SCK10}}$ pins are used. (SO11, SI11 and $\overline{\text{SCK11}}$ pins go to high-impedance state.)
1	SO11, SI11 and $\overline{\text{SCK11}}$ pins are used. (SO10, SI10 and $\overline{\text{SCK10}}$ pins go to high-impedance state.)

CRXE1	Data reception
0	Disabled (When serial clock is input, a 0 is input in the shift register.)
1	Enabled

CTXE1	Data transmission
0	Disabled (Output buffers of SO10 and SI10 (or SO11 and SI11) pins go to high-impedance state.)
1	Enabled

Remark f_{CLK}: Internal system clock

12.6.4 Action taken when shift operation is not in phase with serial clock

The synchronization of the shift operation with the serial clock can be restored by disabling both transmission and reception.

- **Action taken when shift operation is not in phase with serial clock**
  When an external clock is being used as the serial clock, a mismatch between the number of serial clock pulses and the shift operation can occur (for example because of noise). When this happens, disable both transmission and reception by setting the CTXE1 and CRXE1 bits to 0 (reset). This initializes the serial clock counter. Then, enable transmission and reception, and the first pulse input will be treated as the first serial clock. Based on this first clock, the synchronization between the shift operation and serial clock is restored.

13.1 Configuration

The μPD78356 has two PWM signal outputs of 8-/10-/12-bit variable resolution. By externally connecting a low-pass filter, a PWM output can be used as a digital-to-analog conversion output. The PWM outputs are most suitable, for example, for a control signal for the actuator of a motor.

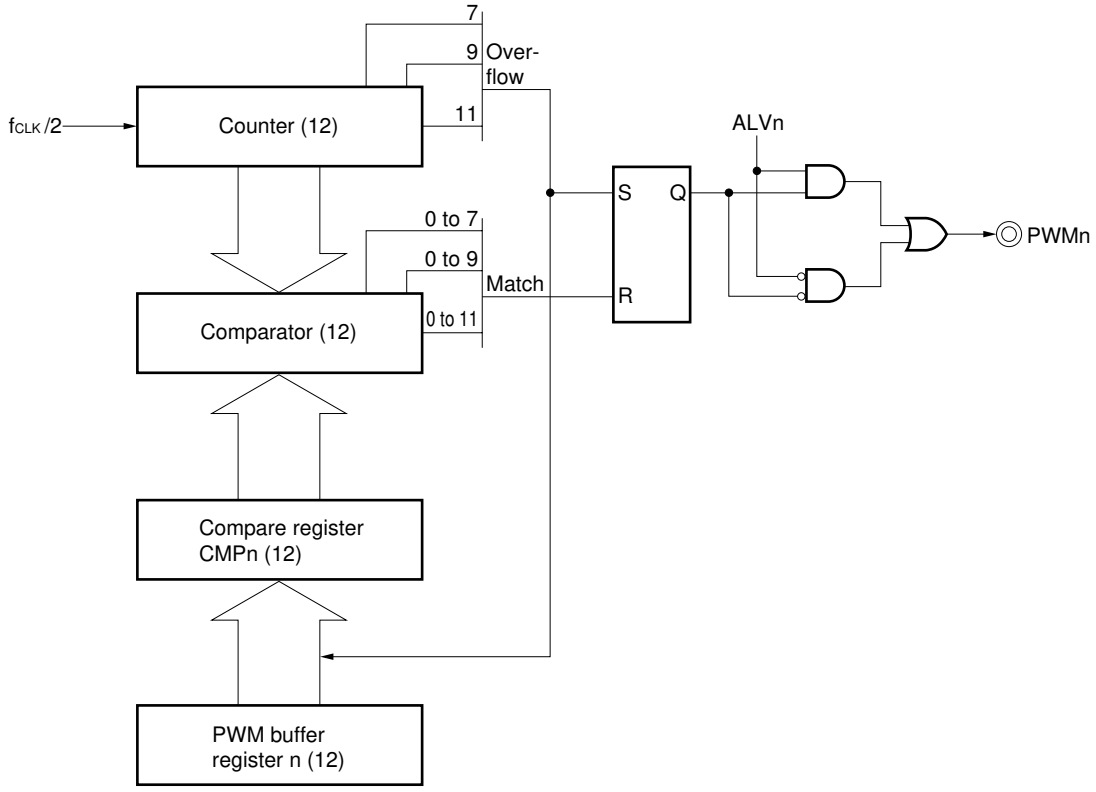
Table 13-1 lists PWM signal output repetition frequencies. Fig. 13-1 shows the block diagram of the PWM unit.

Table 13-1. PWM Signal Repetition Frequencies

Resolution per bit	Bit length for PWM output	Repetition frequency
2/f_{CLK} (125 ns)	8 bits	$f_{CLK}/2^9$ (31.2 kHz)
	10 bits	$f_{CLK}/2^{11}$ (7.8 kHz)
	12 bits	$f_{CLK}/2^{13}$ (1.9 kHz)

Remark The values in parentheses are for f_{CLK} = 16 MHz.

Fig. 13-1. Block Diagram of the PWM Unit



Remark n = 0 or 1

13.2 Control Register

13.2.1 PWM control register (PWMC)

The PWM control register controls PWM output. Fig. 13-2 shows the format of the PWMC register.

When a $\overline{\text{RESET}}$ signal is input, the PWMC register is located at 00H.

Fig. 13-2. Format of PWM Control Register

Symbol	<7>	6	5	4	<3>	2	1	0	Address	When reset	R/W
PWMC	PWME1	ALV1	PRM11	PRM10	PWME0	ALV0	PRM01	PRM00	FFA0H	00H	R/W

PRMn1	PRMn0	Bit length of counter or CMP register n (n = 0 or 1)
0	0	8 bits
0	1	10 bits
1	0	12 bits
1	1	Setting prohibited

ALVn	Specification of active level for PWMn signal output (n = 0 or 1)
0	Low level
1	High level

PWMEn	Operation control of PWMn signal output (n = 0 or 1)
0	Operation stopped
1	Operation allowed

13.2.2 PWM buffer registers (PWM0 and PWM1)

The PWM0 and PWM1 registers are 12-bit registers that set the data for controlling the active signal width of PWM output. Data can be read from or written into the registers in units of bytes or words by an instruction.

When an overflow occurs in the counter for PWM output control, the contents of the PWM0 or PWM1 register are transferred to compare register CMP0 or CMP1.

A $\overline{\text{RESET}}$ input signal makes the contents of the PWM buffer registers undefined.

Caution Byte access/bit access is possible for the low-order of the PWM0/PWM1 register, but not for its high-order.

13.2.3 Compare registers (CMP0 and CMP1)

The CMP0 and CMP1 compare registers are 12-bit registers used to detect coincidence between the values of the register and counter for PWM output control.

The registers cannot be directly manipulated by an instruction.

13.3 Operation

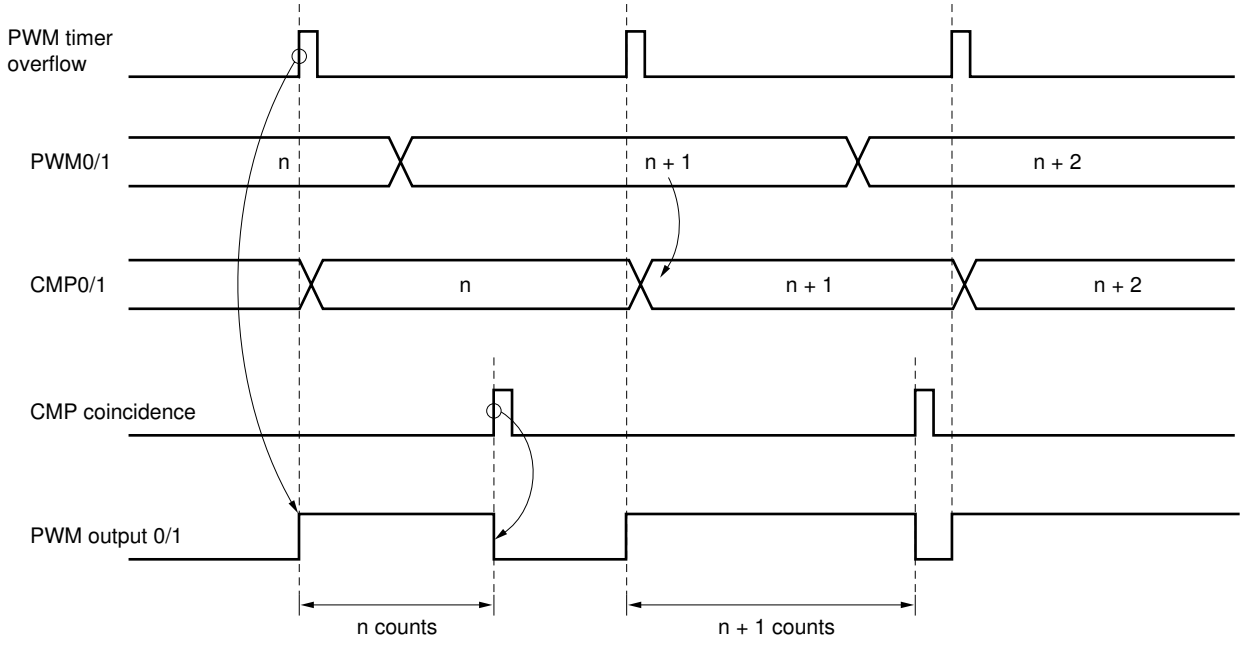
The PWM output function is used in the following procedure:

- The PWM output signal period is specified by bit ALV0 or ALV1 of the PWMC register.
- The data for controlling the active signal width of PWM output is set in the PWM0 or PWM1 register.
- Bit PWME0 or PWME1 of the PWMC register is set to 1.

After these steps are completed, a PWM signal is output from port P86 or P87. The active signal width of PWM output can be changed by rewriting the contents of the PWM0 or PWM1 register.

The PWM output function sends the contents of the PWM0 or PWM1 register to the CMP0 or CMP1 register when the counter for PWM output control overflows. The PWM output signal goes inactive when the contents of the CMP0 or CMP1 register coincide with the value of the counter for PWM output control. The PWM output signal goes active when the counter for PWM output control overflows.

Fig. 13-3. Operation of PWM Output Function (High-Active Setting)



[MEMO]

14.1 Configuration

The watchdog timer prevents crashes or deadlocks.

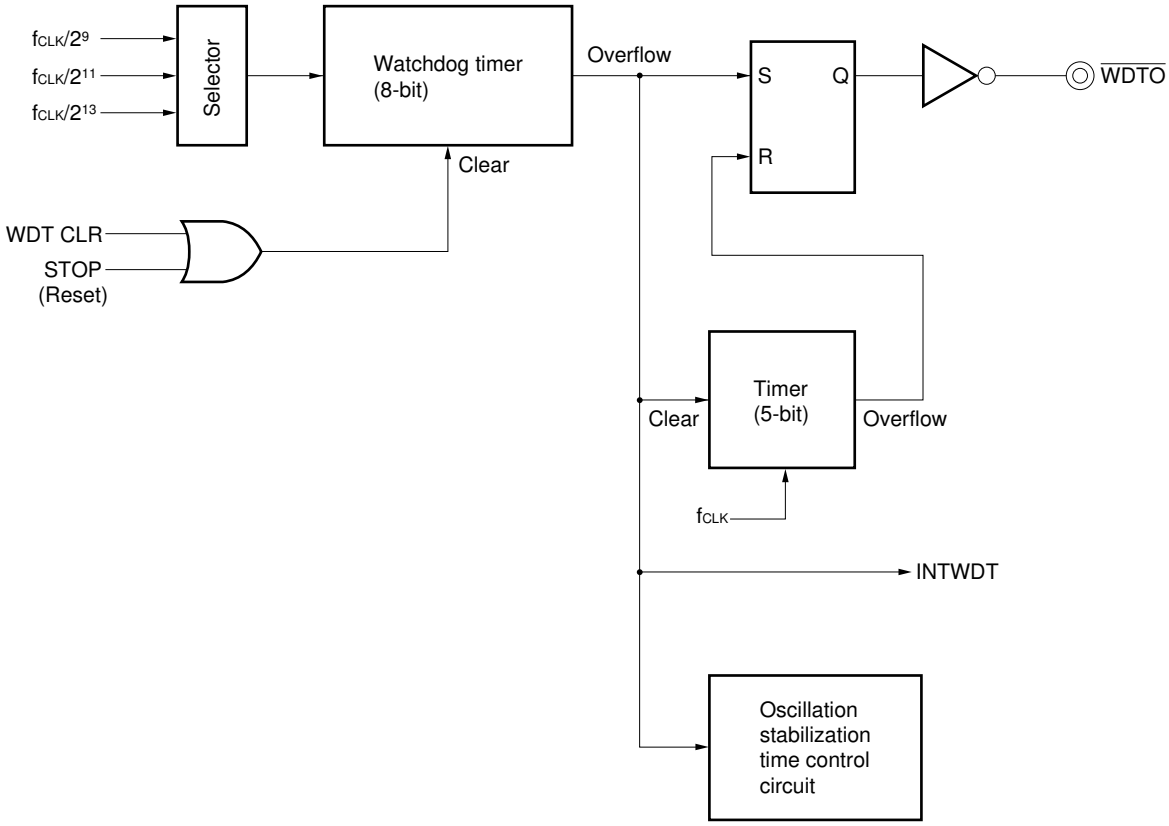
If no watchdog timer interrupt occurs, the program or system is running normally. Each module of a program must have an instruction to clear the watchdog timer and to start counting.

If the instruction to clear the watchdog timer is not executed within a specified time period, an overflow occurs in the watchdog timer and a watchdog timer interrupt (INTWDT) is generated. At the same time, the $\overline{\text{WDTO}}$ pin goes low to indicate that an error has occurred in the program.

The watchdog timer can also be used to guarantee a time required for the oscillator to perform stable operation when the STOP mode is released. (See **16.3.2**.)

Fig. 14-1 shows the block diagram of the watchdog timer.

Fig. 14-1. Block Diagram of the Watchdog Timer



14.2 Watchdog Timer Mode Register (WDM)

The watchdog timer mode (WDM) register is an 8-bit register which controls the operation of the watchdog timer.

Data can be written into the WDM register only by a special instruction. This prevents the contents of the WDM register from being rewritten accidentally if the program crashes. The specialized instruction is MOV WDM, #byte instruction, consisting of special codes (4 bytes). Data is written only when the op-codes of bytes 3 and 4 complement each other.

Unless the op-codes of bytes 3 and 4 complement each other, data is not written and an op-code trap interrupt occurs. The address of the instruction causing the trap is saved in the stack area. When an RETB instruction is executed, the program can be restarted from the address of the instruction causing the trap.

If the RETB instruction is executed before a hardware error or other cause of the op-code trap is eliminated, the program enters an infinite loop.

If the watchdog timer is started after a system reset signal ($\overline{\text{RESET}}$) is entered, the contents of the WDM register cannot be rewritten. Only the system reset signal can stop the watchdog timer. The watchdog timer can be cleared at any time by a special instruction.

The contents of the WDM register can be read at any time by a data transfer instruction.

When a $\overline{\text{RESET}}$ signal is input, the WDM register is located at 00H.

Fig. 14-2 shows the format of the WDM register.

Fig. 14-2. Format of the Watchdog Timer Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
WDM	RUN	0	0	PRC	0	WDI2	WDI1	0	FFC2H	00H	R/W

WDI2	WDI1	Count clock	Overflow time [ms]	
			f_{CLK} = 12.5 MHz	f_{CLK} = 16.0 MHz
0	0	$f_{CLK}/2^9$	10.5	8.2
0	1	$f_{CLK}/2^{11}$	41.9	32.8
1	0	$f_{CLK}/2^{13}$	167.8	131.1
1	1	Setting prohibited		

Remark f_{CLK} is an internal system clock frequency.

PRC	Priority of watchdog timer interrupt request		
0	Watchdog timer interrupt request	<	NMI pin input interrupt request
1	Watchdog timer interrupt request	>	NMI pin input interrupt request

RUN	Watchdog timer operation specification
0	Stop watchdog timer
1	Clear watchdog timer, and start counting

Cautions 1. Data can be written into the WDM register only by a dedicated instruction (MOV WDM, #byte).

2. Do not change the priority for interrupt requests dynamically, that is, when a program is being executed.

3. The RUN bit cannot be reset to 0 by software.

4. The count clock is not reset even when the watchdog timer is cleared by setting the RUN bit to 1.

14.3 Watchdog Timer Output Pin

The watchdog timer output pin ($\overline{\text{WDTO}}$) is an open-drain output pin which can report a system error to the outside without software intervention. When the watchdog timer overflows, the watchdog timer output goes low for a period of 32 system clocks.

As the $\overline{\text{RESET}}$ pin can be directly connected, the watchdog timer output pin outputs a low-level signal for a period of 32 clocks even when a system reset occurs.

Caution Immediately after the power is turned on, the watchdog timer output pin may go low for a period of up to 32 clocks.

14.4 Example of Application

At an early stage in program development, a program is designed without the watchdog timer. After rough debugging is completed, the program is debugged with the watchdog timer.

The watchdog timer must be set in the mode of short overflow time if the response time after a system error is critical.

When the watchdog timer is not required, it can be used for nonmaskable time base interrupt.

CHAPTER 15 INTERRUPT FUNCTION

The μPD78356 is provided with the following three modes for handling interrupt requests. (See **Table 15-1**). These three modes can be set freely in the program. Interrupt processes by macro service however can be selected only for the interrupt requests sources provided with the macro service processing modes shown in Table 15-2. Context switching cannot be selected for nonmaskable interrupt requests and op-code trap interrupt requests.

Table 15-1. Interrupt Request Handling

Interrupt request handling mode	Handled by	Contents of PC and PSW	Handling
Vectored interrupt	Software	Saving/restoring to stack	Branches into the addresses specified in the vector table and executes the interrupt processing routine.
Context switching		Saving/restoring to the fixed area in register bank	Automatically switched to the register bank specified in the vector table, branches to the addresses specified in the fixed area in the register bank, and executes the interrupt processing routine.
Macro service	Hardware (Firmware)	Retained	Executes preset processes such as data transfer between the memory and I/O.

For maskable vectored interrupts, multiple processing control with a 4-level priority order can be performed easily.

Table 15-2. Interrupt Sources (1/2)

Interrupt request type	Default priority	Interrupt source				Macro service	Macro service control word address	Vector table address	
		Interrupt request signal	Interrupt request flag	Source	Unit requesting interrupt			TPF = 0	TPF = 1
Software	—	—		Op-code trap	—	Not provided	—	003CH	
				Execution of BRK instruction				003EH	
Non-maskable		NMI	–	Input to NMI pin	External			0002H	8002H
		INTWDT	–	Watchdog timer overflow	WDT			0004H	8004H
Maskable	0	INTOV0	OVIF0	Timer 0 overflow	RPU	Provided	FE06H	0006H	8006H
	1	INTOV3	OVIF3	Timer 3 overflow			FE08H	0008H	8008H
	2	INTP0/INTCC00	PIF0	INTP0 pin input/CC00 match signal	External/RPU		FE0AH	000AH	800AH
	3	INTP1/INTCC01	PIF1	INTP1 pin input/CC01 match signal			FE0CH	000CH	800CH
	4	INTP2/INTCC02	PIF2	INTP2 pin input/CC02 match signal			FE0EH	000EH	800EH
	5	INTP3/INTCC30	PIF3	INTP3 pin input/CC30 match signal			FE10H	0010H	8010H
	6	INTP4/INTCC31	PIF4	INTP4 pin input/CC31 match signal			FE12H	0012H	8012H
	7	INTCM00	CMIF00	CM00 match signal	RPU		FE14H	0014H	8014H
	8	INTCM01	CMIF01	CM01 match signal			FE16H	0016H	8016H
	9	INTCM02	CMIF02	CM02 match signal			FE18H	0018H	8018H
	10	INTCM03	CMIF03	CM03 match signal			FE1AH	001AH	801AH
	11	INTCM10	CMIF10	CM10 match signal			FE1CH	001CH	801CH
	12	INTCM11	CMIF11	CM11 match signal			FE1EH	001EH	801EH
	13	INTCM20	CMIF20	CM20 match signal			FE20H	0020H	8020H
	14	INTCM21	CMIF21	CM21 match signal			FE22H	0022H	8022H
	15	INTCM40	CMIF40	CM40 match signal			FE24H	0024H	8024H

Remark Default priority: priority fixed by hardware

Table 15-2. Interrupt Sources (2/2)

Interrupt request type	Default priority	Interrupt source				Macro service	Macro service control word address	Vector table address	
		Interrupt request signal	Interrupt request flag	Source	Unit requesting interrupt			TPF = 0	TPF = 1
Maskable	16	INTCMUD0	CMIFUD0	CMUD0 match signal	RPU	Provided	FE26H	0026H	8026H
	17	INTCMUD1	CMIFUD1	CMUD1 match signal			FE28H	0028H	8028H
	18	INTSER	SERIF	Serial receive error	UART		FE2AH	002AH	802AH
	19	INTSR	SRIF	Serial reception completion			FE2CH	002CH	802CH
	20	INTST	STIF	Serial transmission completion			FE2EH	002EH	802EH
	21	INTCSI0	CSIIF0	Serial transmission/ reception completion	CSI0		FE30H	0030H	8030H
	22	INTCSI1	CAIIF1	Serial transmission/ reception completion	CSI1		FE32H	0032H	8032H
	23	INTAD	ADIF	A/D conversion completion	A/D		FE34H	0034H	8034H
Reset	—	$\overline{\text{RESET}}$	—	Input to $\overline{\text{RESET}}$ pin	—	Not provided	—	0000H	

Remark Default priority: priority fixed by hardware

15.1 Interrupt Requests

With the μPD78356, four types of interrupt requests are used:

- Nonmaskable interrupt
- Maskable interrupt
- Software interrupt
- Op-code trap interrupt

Each type of interrupt request is explained below.

15.1.1 Nonmaskable interrupt

The nonmaskable interrupt is generated by the NMI pin input or watchdog timer. The nonmaskable interrupt is accepted unconditionally[Note] even in the interrupt disabled state. It is also not subjected to the interrupt priority level control and has priority over all the other interrupts.

Note Except while the same nonmaskable interrupt handling is being executed and while nonmaskable interrupt handlings with higher priority are being executed.

15.1.2 Maskable interrupt

The maskable interrupt is an interrupt which is mask controlled according to the setting of the interrupt mask flags. Acceptance able/disable can be specified for all the maskable interrupts using the PSW IE flag.

The maskable interrupt can also be accepted by context switching and macro service in addition to the normal vectored interrupt. (See **Table 15-2**).

The top priority of maskable interrupt is determined when several interrupt requests with the same priority as shown in Table 15-2 are generated together (Default priority level). The interrupt priority level can be divided into 4 groups of different levels and multiple handling control can be performed. The macro service however can be accepted regardless of the priority level control and IE flag.

15.1.3 Software interrupt

The software interrupts consist of the BRK instruction which generates the vectored interrupt and BRKCS instruction which performs context switching.

The software interrupt can be accepted even in the interrupt disabled state. It is not subject to the interrupt priority level control.

15.1.4 Op-code trap interrupt

If a write to the watchdog timer mode register (WDM) or standby control register (STBC) is not executed normally, an op-code trap interrupt request is generated. (See **14.2** and **16.2**.)

The op-code trap interrupt can also be accepted in the DI state. It is not subject to the interrupt priority level control.

15.2 Interrupt Servicing Modes

With the μPD78356, three interrupt servicing modes are available:

- Vectored interrupt servicing
- Macro service
- Context switching

15.2.1 Vectored interrupt service

When an interrupt is acknowledged, the contents of program counter (PC) and program status word (PSW) are saved in the stack memory automatically. Then a branch is made to the address indicated by the data contained in the vector address table to execute the interrupt service routine.

RETI instruction is executed to the return from the interrupt service routine.

15.2.2 Macro service

When an interrupt is accepted, CPU execution is terminated temporarily to transfer data by hardware rather than by software. The macro service is performed without CPU involvement, so that the CPU statuses such as PC and PSW need not be saved or restored. Thus the macro service much increases CPU service time. (See **15.8**)

15.2.3 Context switching

When an interrupt is accepted, a specified register bank is selected by hardware. Then a branch is made to the already selected vector address in the register bank, and the current contents of PC and PSW are saved in the register bank at the same time. (See **15.5.2** and **15.6.2**)

Remark The context means CPU registers that can be accessed from a program being executed. The registers include general registers, PC, PSW, and stack pointer (SP).

15.3 Interrupt Control Registers

Responses to interrupts in the μPD78356 are controlled according to the interrupt requests. This control is done using control registers that specify how interrupts are handled. Table 15-3 lists interrupt control registers.

Table 15-3. Interrupt Control Registers

Register	Symbol	Function
Interrupt control register	OVIC0 OVIC3 PIC0 PIC1 PIC2 PIC3 PIC4 CMIC00 CMIC01 CMIC02 CMIC03 CMIC10 CMIC11 CMIC20 CMIC21 CMIC40 CMICUD0 CMICUD1 SERIC SRIC STIC CSIIC0 CSIIC1 ADIC	Each register indicates interrupt request generation, performs mask control, specified vectored interrupt handling or macro service processing, enables or disables the context switching function, and specifies priority.
Interrupt mask flag register	MK0 MK1	Each register performs mask control for maskable interrupt requests and is set in correspondence with the mask control flag of the interrupt control register. Each register can be accessed in words or in bytes.
In-service priority register	ISPR	Indicates priority of currently accepted interrupt request.
Interrupt mode control register	IMC	Controls nesting of maskable interrupt for which lowest priority (level 3) is specified.

A control register is assigned to each interrupt source. The flags of each register are used to specify various types of control which are determined by the bit positions in the register.

Table 15-4 lists the flags in the interrupt control registers for the interrupt request signals.

Table 15-4. Interrupt Control Register Flags for the Interrupt Request Signals

Default priority	Interrupt request signal	Interrupt control register				
		Interrupt request flag	Interrupt mask flag	Macro service enable flag	Priority specification flag	Context switching enable flag
0	INTOV0	OVIF0	OVMK0	OVISM0	OVPR00 OVPR01	OVCSE0
1	INTOV3	OVIF3	OVMK3	OVISM3	OVPR30 OVPR31	OVCSE3
2	INTP0/ INTCC00	PIF0	PMK0	PISM0	PPR00 PPR01	PCSE0
3	INTP1/ INTCC01	PIF1	PMK1	PISM1	PPR10 PPR11	PCSE1
4	INTP2/ INTCC02	PIF2	PMK2	PISM2	PPR20 PPR21	PCSE2
5	INTP3/ INTCC30	PIF3	PMK3	PISM3	PPR30 PPR31	PCSE3
6	INTP4/ INTCC31	PIF4	PMK4	PISM4	PPR40 PPR41	PCSE4
7	INTCM00	CMIF00	CMMK00	CMISM00	CMPR000 CMPR001	CMCSE00
8	INTCM01	CMIF01	CMMK01	CMISM01	CMPR010 CMPR011	CMCSE01
9	INTCM02	CMIF02	CMMK02	CMISM02	CMPR020 CMPR021	CMCSE02
10	INTCM03	CMIF03	CMMK03	CMISM03	CMPR030 CMPR031	CMCSE03
11	INTCM10	CMIF10	CMMK10	CMISM10	CMPR100 CMPR101	CMCSE10
12	INTCM11	CMIF11	CMMK11	CMISM11	CMPR110 CMPR111	CMCSE11
13	INTCM20	CMIF20	CMMK20	CMISM20	CMPR200 CMPR201	CMCSE20
14	INTCM21	CMIF21	CMMK21	CMISM21	CMPR210 CMPR211	CMCSE21
15	INTCM40	CMIF40	CMMK40	CMISM40	CMPR400 CMPR401	CMCSE40
16	INTCMUD0	CMIFUD0	CMMKUD0	CMISMUD0	CMPRUD00 CMPRUD01	CMCSEUD0
17	INTCMUD1	CMIFUD1	CMMKUD1	CMISMUD1	CMPRUD10 CMPRUD11	CMCSEUD1
18	INTSER	SERIF	SERMK	SERISM	SERPR0 SERPR1	SERCSE
19	INTSR	SRIF	SRMK	SRISM	SRPR0 SRPR1	SRCSE
20	INTST	STIF	STMK	STISM	STPR0 STPR1	STCSE
21	INTCSI0	CSIIF0	CSIMK0	CSIISM0	CSIPR00 CSIPR01	CSICSE0
22	INTCSI1	CSIIF1	CSIMK1	CSIISM1	CSIPR10 CSIPR11	CSICSE1
23	INTAD	ADIF	ADMK	ADISM	ADPR0 ADPR1	ADCSE

15.3.1 Interrupt control registers

The interrupt control registers are assigned to different interrupt sources. Each register is used to control priority and masking for its associated interrupt source. Fig. 15-1 shows the format of the interrupt control registers.

(1) Priority specification flag (xxPR1, xxPR0)

The priority specification flags specify the priority of the associated interrupt source for the twenty-four maskable interrupts.

Priority level 0 to 4 can be specified. More than one interrupt source can have the same priority level. Maskable interrupt sources with level 0 have the highest priority.

If more than one interrupt request is generated and the interrupt sources have the same priority level, the requests are accepted according to the default priority.

The flags are set or reset bit by bit by software.

$\overline{\text{RESET}}$ input sets all bits to 1.

(2) Context switching enable flag (xxCSE)

The context switching enable flag specifies whether to respond to a maskable interrupt request with context switching.

The context switching function selects the previously specified register bank on a hardware basis, makes a branch to the vector address stored in the register bank, and also saves the contents of the current program counter (PC) and program status word (PSW) in the register bank.

Context switching can start interrupt service at a higher speed than normal vectored interrupt handling. Context switching is therefore appropriate for real-time processing.

The flag is set or reset bit by bit by software.

$\overline{\text{RESET}}$ input sets all bits to 0.

(3) Macro service enable flag (xxISM)

The macro service enable flag specifies whether to respond to the associated interrupt request with vectored interrupt handling or macro service processing.

If macro service processing is selected, when macro service terminates (when the macro service counter overflows) and a vectored interrupt is generated, the flag is automatically reset to 0 by hardware (vectored interrupt handling).

The flag can be set or reset bit by bit by software.

$\overline{\text{RESET}}$ input sets all bits to 0.

(4) Interrupt mask flag (xxMK)

The interrupt mask flag enables or disables vectored interrupt processing or macro service processing for the associated interrupt request.

The interrupt mask flag is not changed by the activation of interrupt handling. The setting of the interrupt mask flag matches the contents of the interrupt mask flag register. (See **15.3.2.**)

As the macro service request is subject to mask control, the macro service request can also be masked by this flag.

This flag can be set or reset bit by bit by software.

$\overline{\text{RESET}}$ input sets all bits to 1.

The interrupt handling is determined by the combination of the interrupt mask flag (xxMK) and macro service enable flag (xxISM).

xxMK	xxISM	Interrupt handling
0	0	Handled by vectored interrupt
0	1	Handled by vectored interrupt after handling by macro service
1	x	Maskable interrupt requests are not accepted

(5) Interrupt request flag (xxIF)

The interrupt request flag is set to 1 when the associated interrupt request is generated. When the interrupt is accepted, the flag is automatically reset to 0 by hardware.

This flag can be set or reset bit by bit by software.

$\overline{\text{RESET}}$ input sets all bits to 0.

Fig. 15-1. Format of the Interrupt Control Registers (1/3)

Symbol	<7>	<6>	<5>	<4>	3	2	<1>	<0>	Address	When reset	R/W
OVIC0	OVIF0	OVMK0	OVISM0	OVCSE0	0	0	OVPR01	OVPR00	FFE0H	43H	R/W
OVIC3	OVIF3	OVMK3	OVISM3	OVCSE3	0	0	OVPR31	OVPR30	FFE1H	43H	R/W
PIC0	PIF0	PMK0	PISM0	PCSE0	0	0	PPR01	PPR00	FFE2H	43H	R/W
PIC1	PIF1	PMK1	PISM1	PCSE1	0	0	PPR11	PPR10	FFE3H	43H	R/W
PIC2	PIF2	PMK2	PISM2	PCSE2	0	0	PPR21	PPR20	FFE4H	43H	R/W
PIC3	PIF3	PMK3	PISM3	PCSE3	0	0	PPR31	PPR30	FFE5H	43H	R/W
PIC4	PIF4	PMK4	PISM4	PCSE4	0	0	PPR41	PPR40	FFE6H	43H	R/W
CMIC00	CMIF00	CMMK 00	CMISM 00	CMCSE 00	0	0	CMPR 001	CMPR 000	FFF7H	43H	R/W

xxPR1	xxPR0	Priority specification
0	0	Priority 0 (highest priority)
0	1	Priority 1
1	0	Priority 2
1	1	Priority 3

xxCSE	Context switching enable flag
0	Handled by vectored interrupt
1	Handled by context switching

xxISM	Macro service enable flag
0	Handled by vectored interrupt
1	Handled by macro service

xxMK	Interrupt request enable/disable specification
0	Enables interrupt handling
1	Disables interrupt handling

xxIF	Interrupt request flag
0	No interrupt request. The interrupt request signal is not generated.
1	The interrupt request signal is generated, and an interrupt is requested.

Fig. 15-1. Format of the Interrupt Control Registers (2/3)

Symbol	<7>	<6>	<5>	<4>	3	2	<1>	<0>	Address	When reset	R/W
CMIC01	CMIF01	CMMK01	CMISM01	CMCSE01	0	0	CMPR011	CMPR010	FFE8H	43H	R/W
CMIC02	CMIF02	CMMK02	CMISM02	CMCSE02	0	0	CMPR021	CMPR020	FFE9H	43H	R/W
CMIC03	CMIF03	CMMK03	CMISM03	CMCSE03	0	0	CMPR031	CMPR030	FFEAH	43H	R/W
CMIC10	CMIF10	CMMK10	CMISM10	CMCSE10	0	0	CMPR101	CMPR100	FFEBH	43H	R/W
CMIC11	CMIF11	CMMK11	CMISM11	CMCSE11	0	0	CMPR111	CMPR110	FFECH	43H	R/W
CMIC20	CMIF20	CMMK20	CMISM20	CMCSE20	0	0	CMPR201	CMPR200	FFEDH	43H	R/W
CMIC21	CMIF21	CMMK21	CMISM21	CMCSE21	0	0	CMPR211	CMPR210	FFEEH	43H	R/W
CMIC40	CMIF40	CMMK40	CMISM40	CMCSE40	0	0	CMPR401	CMPR400	FFEFH	43H	R/W

xxPR1	xxPR0	Priority specification
0	0	Priority 0 (highest priority)
0	1	Priority 1
1	0	Priority 2
1	1	Priority 3

xxCSE	Context switching enable flag
0	Handled by vectored interrupt
1	Handled by context switching

xxISM	Macro service enable flag
0	Handled by vectored interrupt
1	Handled by macro service

xxMK	Interrupt request enable/disable specification
0	Enables interrupt handling
1	Disables interrupt handling

xxIF	Interrupt request flag
0	No interrupt request. The interrupt request signal is not generated.
1	The interrupt request signal is generated, and an interrupt is requested.

Fig. 15-1. Format of the Interrupt Control Registers (3/3)

Symbol	<7>	<6>	<5>	<4>	3	2	<1>	<0>	Address	When reset	R/W
CMICUD0	CMIF UD0	CMMK UD0	CMISM UD0	CMCSE UD0	0	0	CMPR UD01	CMPR UD00	FFF0H	43H	R/W
CMICUD1	CMIF UD1	CMMK UD1	CMISM UD1	CMCSE UD1	0	0	CMPR UD11	CMPR UD10	FFF1H	43H	R/W
SERIC	SERIF	SERMK	SERISM	SERCSE	0	0	SERPR1	SERPR0	FFF2H	43H	R/W
SRIC	SRIF	SRMK	SRISM	SRCSE	0	0	SRPR1	SRPR0	FFF3H	43H	R/W
STIC	STIF	STMK	STISM	STCSE	0	0	STPR1	STPR0	FFF4H	43H	R/W
CSIIC0	CSIIF0	CSIMK0	CSI ISM0	CSI CSE0	0	0	CSI PR01	CSI PR00	FFF5H	43H	R/W
CSIIC1	CSIIF1	CSIMK1	CSI ISM1	CSI CSE1	0	0	CSI PR11	CSI PR10	FFF6H	43H	R/W
ADIC	ADIF	ADMK	ADISM	ADCSE	0	0	ADPR1	ADPR0	FFF7H	43H	R/W

xxPR1	xxPR0	Priority specification
0	0	Priority 0 (highest priority)
0	1	Priority 1
1	0	Priority 2
1	1	Priority 3

xxCSE	Context switching enable flag
0	Handled by vectored interrupt
1	Handled by context switching

xxISM	Macro service enable flag
0	Handled by vectored interrupt
1	Handled by macro service

xxMK	Interrupt request enable/disable specification
0	Enables interrupt handling
1	Disables interrupt handling

xxIF	Interrupt request flag
0	No interrupt request. The interrupt request signal is not generated.
1	The interrupt request signal is generated, and an interrupt is requested.

15.3.2 Interrupt mask flag register (MK0 and MK1)

The interrupt mask flag registers (MK0 and MK1) are made up of interrupt mask flags for the associated twenty-four maskable interrupt requests.

The MK0 register is a 16-bit register which can be manipulated in 16 bits. It can also be manipulated in 8 bits as MK0L and MK0H.

The MK1 register is a 16-bit register which can be manipulated in 16 bits. In addition, it can also be manipulated in 8 bits as MK1L. Each bit of the MK0 and MK1 registers can be manipulated in 1 bit by the bit manipulation instruction. Each interrupt mask flag controls the enable/disable of the corresponding interrupt request.

When the interrupt mask flag is set (1), the acceptance of the corresponding interrupt request is disabled.

When the interrupt mask flag is set (0), the acceptance of the corresponding interrupt request is enabled, as the vectored interrupt or macro service.

The interrupt mask flags of the MK0 and MK1 registers are the same as those of the interrupt control registers. The MK0 and MK1 registers are provided so that interrupt masks can be controlled together.

The $\overline{\text{RESET}}$ input sets the interrupt mask flags to “1” and disables all maskable interrupts.

Fig. 15-2 shows the format of the interrupt mask flag registers.

Fig. 15-2. Format of the Interrupt Mask Flag Registers (1/2)

(When accessed in bytes)

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	When reset	R/W
MK0L	CMMK00	PMK4	PMK3	PMK2	PMK1	PMK0	OVMK3	OVMK0	FFACH	FFH	R/W
MK0H	CMMK40	CMMK21	CMMK20	CMMK11	CMMK10	CMMK03	CMMK02	CMMK01	FFADH	FFH	R/W
MK1L	ADMK	CSIMK1	CSIMK0	STMK	SRMK	SERMK	CMMKUD1	CMMKUD0	FFAEH	FFH	R/W

xxMK	Interrupt request enable/disable specification
0	Enables interrupt handling
1	Disables interrupt handling

Fig. 15-2. Format of the Interrupt Mask Flag Registers (2/2)

(When accessed in words)

Symbol	<15>	<14>	<13>	<12>	<11>	<10>	<9>	<8>	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>	Address	When reset	R/W
MK0	CMMK40	CMMK21	CMMK20	CMMK11	CMMK10	CMMK03	CMMK02	CMMK01	CMMK00	PMK4	PMK3	PMK2	PMK1	PMK0	OVMK3	OVMK0	FFACH	FFFFH	R/W
MK1	0	0	0	0	0	0	0	0	ADMK	CSIMK1	CSIMK0	STMK	SRMK	SERMK	CMMKUD1	CMMKUD0	FFAEH	00FFH	R/W

xxMK	Interrupt request enable/disable specification
0	Enables interrupt handling
1	Disables interrupt handling

Relationship between bits and interrupt requests (MK0 register)

Bit	Bit name	Interrupt request source
Bit 0	OVMK0	Timer 0 overflow (INTOV0)
Bit 1	OVMK3	Timer 3 overflow (INTOV3)
Bit 2	PMK0	INTP0 pin input/CC00 match (INTP0/INTCC00)
Bit 3	PMK1	INTP1 pin input/CC01 match (INTP1/INTCC01)
Bit 4	PMK2	INTP2 pin input/CC02 match (INTP2/INTCC02)
Bit 5	PMK3	INTP3 pin input/CC30 match (INTP3/INTCC30)
Bit 6	PMK4	INTP4 pin input/CC31 match (INTP4/INTCC31)
Bit 7	CMMK00	CM00 match (INTCM00)
Bit 8	CMMK01	CM01 match (INTCM01)
Bit 9	CMMK02	CM02 match (INTCM02)
Bit 10	CMMK03	CM03 match (INTCM03)
Bit 11	CMMK10	CM10 match (INTCM10)
Bit 12	CMMK11	CM11 match (INTCM11)
Bit 13	CMMK20	CM20 match (INTCM20)
Bit 14	CMMK21	CM21 match (INTCM21)
Bit 15	CMMK40	CM40 match (INTCM40)

Relationship between bits and interrupt requests (MK1 register)

Bit	Bit name	Interrupt request source
Bit 0	CMMKUD0	CMUD0 match (INTCMUD0)
Bit 1	CMMKUD1	CMUD1 match (INTCMUD1)
Bit 2	SERMK	Serial receive error (INTSER)
Bit 3	SRMK	Serial reception completion (INTSR)
Bit 4	STMK	Serial transmission completion (INTST)
Bit 5	CSIMK0	Serial transmission/reception completion (INTCSI0)
Bit 6	CSIMK1	Serial transmission/reception completion (INTCSI1)
Bit 7	ADMK	A/D conversion completion (INTAD)

15.3.3 Interrupt mode control register (IMC)

The interrupt mode control register (IMC) has a PRSL flag. The PRSL flag enables or disables nesting of maskable interrupts for which the lowest priority (level 3) is specified.

For levels 0, 1, and 2, interrupts of the same levels cannot be nested regardless of the setting of the IMC register.

Before manipulating the IMC register, enter the interrupt disable status (DI state) to prevent an error.

The register can be set or reset by software.

$\overline{\text{RESET}}$ input sets the PRSL flag to 1.

Fig. 15-3 shows the format of the IMC register.

Fig. 15-3. Format of the Interrupt Mode Control Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
IMC	PRSL	0	0	0	0	0	0	0	FFAAH	80H	R/W

PRSL	Nesting control
0	Enables nesting between interrupts for which level 3 (lowest level) is set.
1	Disables nesting between interrupts for which level 3 (lowest level) is set.

15.3.4 In-service priority register (ISPR)

The in-service priority register (ISPR) holds the priority level of the interrupt request being serviced. When the interrupt request is accepted, the bit corresponding to the priority level for the request is set to (1), and the level is held during service.

When the RETI or RETCS instruction is executed, the bit for an interrupt request with the highest priority among the bits that are set to (1) in the ISPR register is reset to (0) automatically by hardware.

The ISPR register contents are not changed by the execution of the RETB or RETCSB instruction.

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

Fig. 15-4 shows the format of the ISPR register.

Fig. 15-4. Format of the In-service Priority Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
ISPR	0	0	0	0	ISPR3	ISPR2	ISPR1	ISPR0	FFA8H	00H	R

ISPRn	Priority level
0	Interrupt request with priority n is not accepted.
1	Interrupt request with priority n is accepted.

Remark n = 0 to 3

Caution The ISPR register can be accessed for read only. Write to the register may cause an error.

15.3.5 Program status word (PSW)

The PSW is a register which maintains the current state for the execution results of instructions and interrupt requests. The IE flag which sets the enable/disable of the maskable interrupt is mapped to the lower 8 bits (PSWL) of the PSW.

The PSWL can be read/written in 8 bits and can be manipulated by the bit manipulation instruction and dedicated instruction (EI, DI).

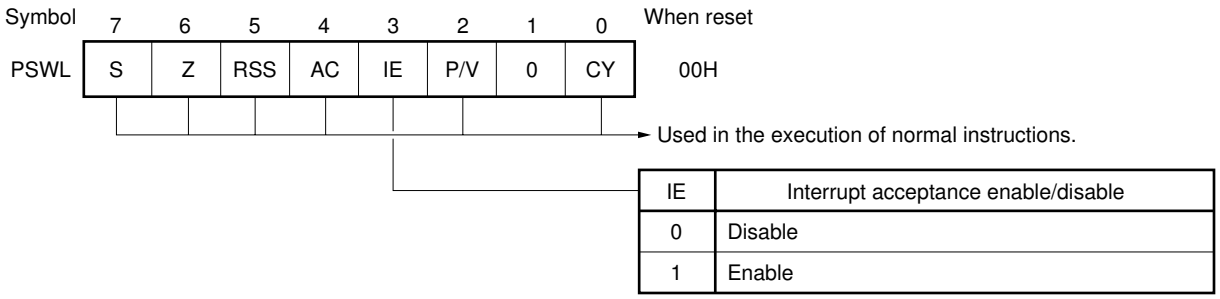
When the vectored interrupt is acknowledged or during the execution of the BRK instruction, it is saved in the stack area and the IE flag is reset (0). It is also saved in the stack area by the PUSH PSW instruction and restored from the stack area by the RETI, RETB, POP PSW instructions.

During context switching and BRKCS instruction execution, it is saved in the register bank fixed area and the IE flag is reset (0). It is restored from the fixed area in the register bank by the RETCS and RETCSB instructions.

An interrupt request for a nonmaskable interrupt or macro service is acknowledged irrespective of the IE flag. The IE flag setting is not changed by macro service.

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

Fig. 15-5. Format of the Program Status Word (PSWL)



IE	Interrupt acceptance enable/disable
0	Disable
1	Enable

15.4 Nonmaskable Interrupt Acknowledgement

The nonmaskable interrupt is acknowledged even in the interrupt disable state. The nonmaskable interrupt is always acknowledged except when the same nonmaskable interrupt or a high priority nonmaskable interrupt servicing is executed.

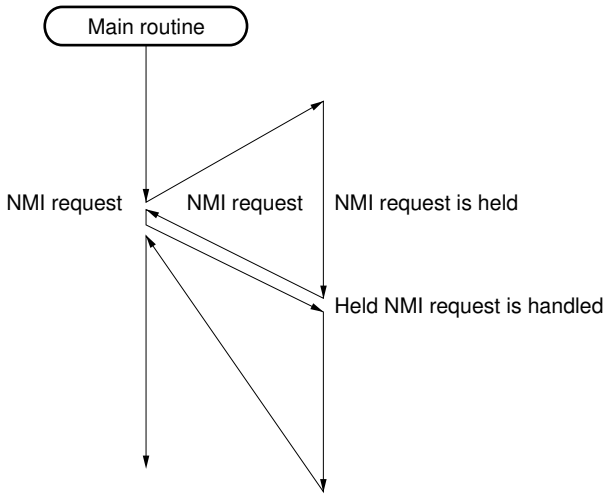
The priority level of the nonmaskable interrupts is set by the PRC bit of the watchdog timer mode register (WDM). (See **14.2**.)

The nonmaskable interrupt request is acknowledged promptly if the state is not the state described in section “15.9 When Interrupt Request and Macro Service are Held Temporarily”. When the nonmaskable interrupt request is acknowledged, it is saved in the stack area in the order of PSW and PC. The PSW IE flag is reset (0), and the contents of the vector table are loaded to PC and branched.

While the nonmaskable interrupt servicing is executed, requests for nonmaskable interrupts which are the same as the one currently executed and nonmaskable interrupt requests with lower priority than the one executed currently are put on hold. The held nonmaskable interrupt is acknowledged after the servicing of the currently executed nonmaskable interrupt ends (RETI instruction execution ends). However, if the same nonmaskable interrupt request is generated more than twice while a nonmaskable interrupt is being handled, only one nonmaskable interrupt will be acknowledged after the completion of the nonmaskable interrupt servicing.

Fig. 15-6. Nonmaskable Interrupt Request Acknowledgement (1/2)

(a) When a new NMI request is generated during NMI interrupt servicing



(b) When the watchdog timer interrupt request is generated during the NMI interrupt servicing (when the watchdog timer interrupt priority is high, the PRC bit of the WDM register = 1)

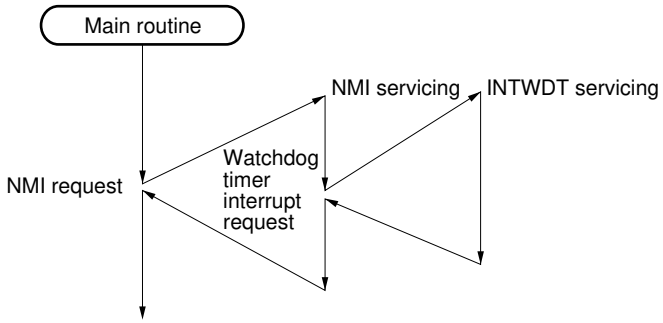
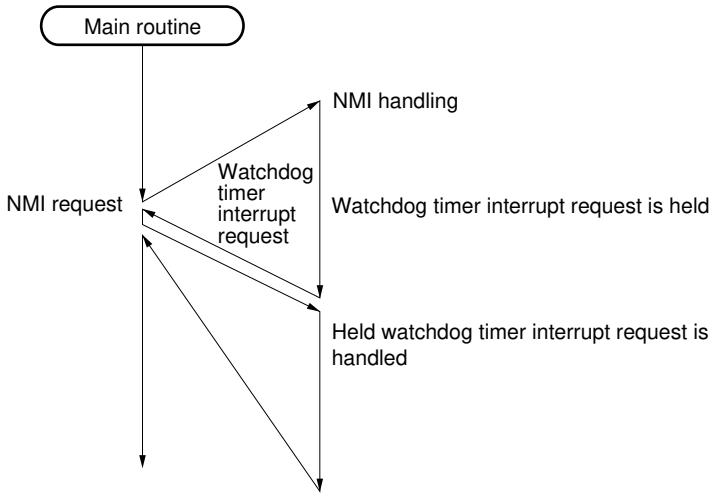
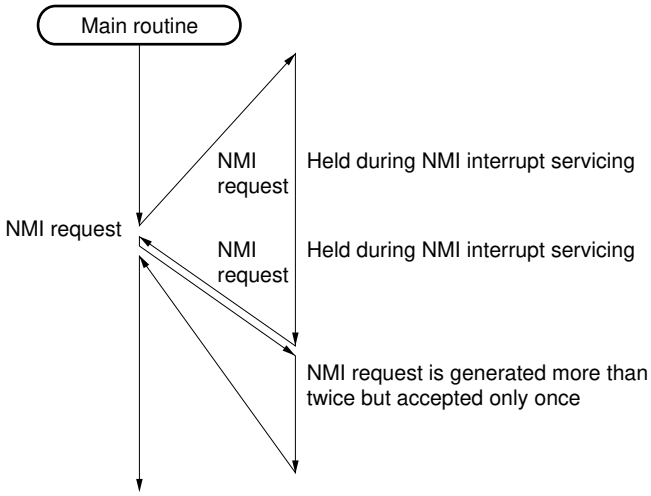


Fig. 15-6. Nonmaskable Interrupt Request Acknowledgement (2/2)

(c) When the watchdog timer interrupt request is generated during the NMI interrupt servicing (when the NMI interrupt priority is high, the PRC bit of the WDM register = 0)



(d) When two new NMI requests are generated during NMI interrupt servicing



Cautions 1. The macro service request is acknowledged and handled even during the nonmaskable interrupt servicing routine. If the macro service servicing is not to be performed during the nonmaskable interrupt servicing routine, manipulate the interrupt mask flag register during the nonmaskable interrupt servicing routine so that the macro service is not generated.

2. Always use the RETI instruction to restore from the nonmaskable interrupt. Interrupts will not be acknowledged properly hereafter if other instructions are used.

3. The nonmaskable interrupt is always acknowledged except for while the nonmaskable interrupt servicing is being executed (excludes when a high priority nonmaskable interrupt request is generated during the execution of the low priority nonmaskable interrupt servicing) and during a certain period of time after the execution of special instructions shown in section 15.9. Therefore, nonmaskable interrupt is acknowledged even if stack pointer values after reset release, etc. are undefined. In such cases, depending on the stack pointer value, program counter (PC) and program status words (PSW) may be written in addresses (See Table 3-4 in 3.2.3.) which disable the writing of special function registers and cause the CPU to deadlock and unexpected signals to be output from pins, or PC and PSW may be written in addresses not mounted with the RAM, and as a result, the main routine cannot be returned normally from the nonmaskable interrupt servicing routine and the CPU overruns.
Therefore, always set programs after the $\overline{\text{RESET}}$ release to as follows.

```
        CSEG AT 0
        DW   STRT
STRT:
        MOVW SP, #imm16
```

15.5 Maskable Interrupt Acknowledgement

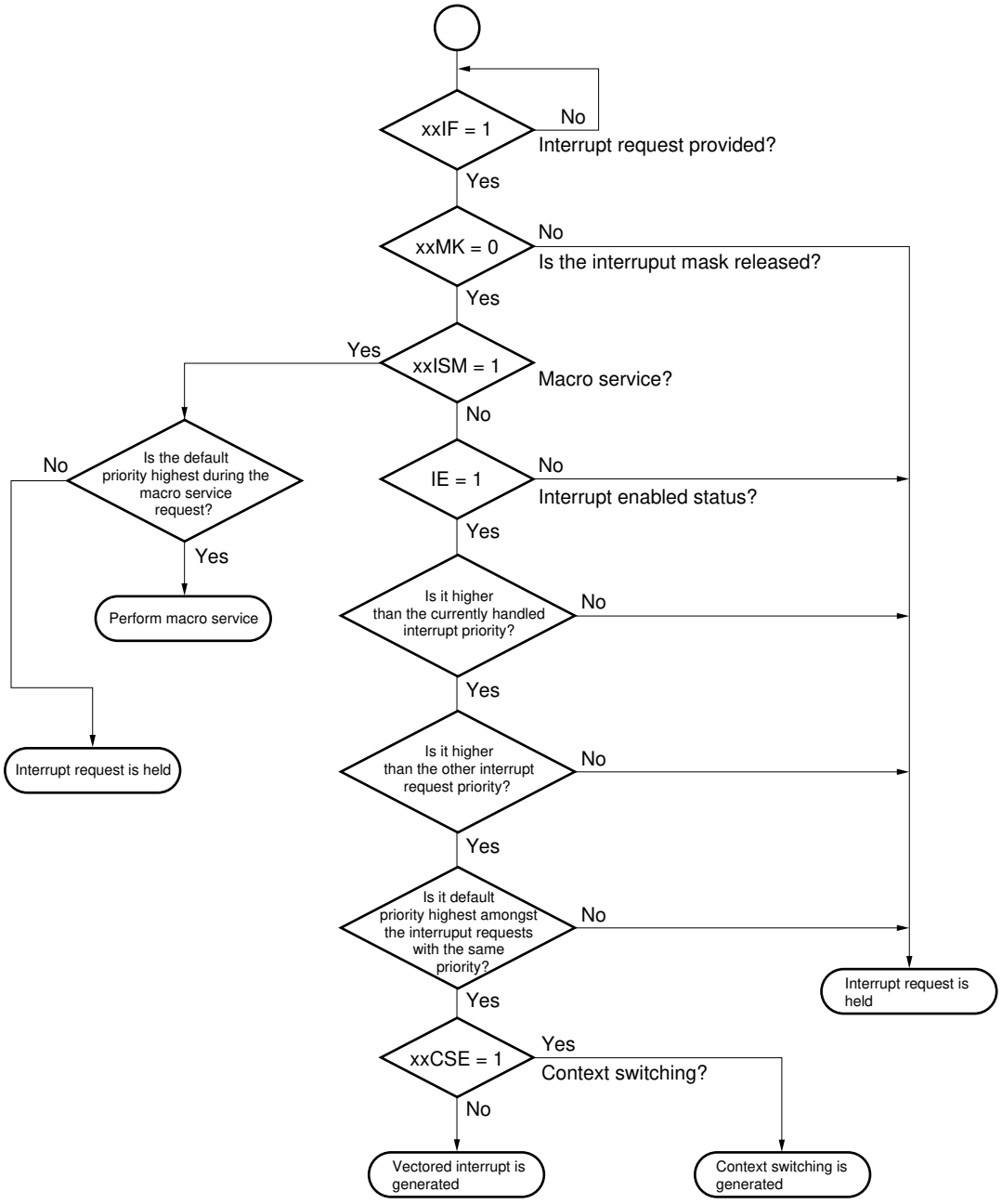
The maskable interrupt can be acceptable when the interrupt request flag is set (1) and the interrupt mask flag is reset (0). When handled by macro service, it is acknowledged promptly and handled. For vectored interrupt or context switching, it is acknowledged if the interrupt enabled state (IE flag is set (1)) and the interrupt priority is acceptable.

If maskable interrupt request are generated simultaneously, these are acknowledged from those specified with the highest priority by the priority level specification flag. If interrupts have the same priority, the default priority level is followed.

The held interrupt is acknowledged when it can be acknowledged.

The interrupt acceptance algorithm is shown in Fig. 15-7.

Fig. 15-7. Interrupt Acknowledgement Algorithm



15.5.1 Vectored interrupt

When the maskable interrupt request is acknowledged by vectored interrupt, it is saved in the stack area in the order of PSW and PC, the IE flag is reset (0) (interrupt disabled state), and the bit of the ISPR register corresponding to the priority of the acknowledged interrupt is set (1). In addition, the data in the vector table determined for each interrupt request is loaded in the PC and branched. Restoring from the vectored interrupt is performed by the RETI instruction.

Caution When the maskable interrupt is acknowledged by the vectored interrupt, always restore using the RETI instruction. If other instructions are used, interrupt operations hereafter will not be performed properly.

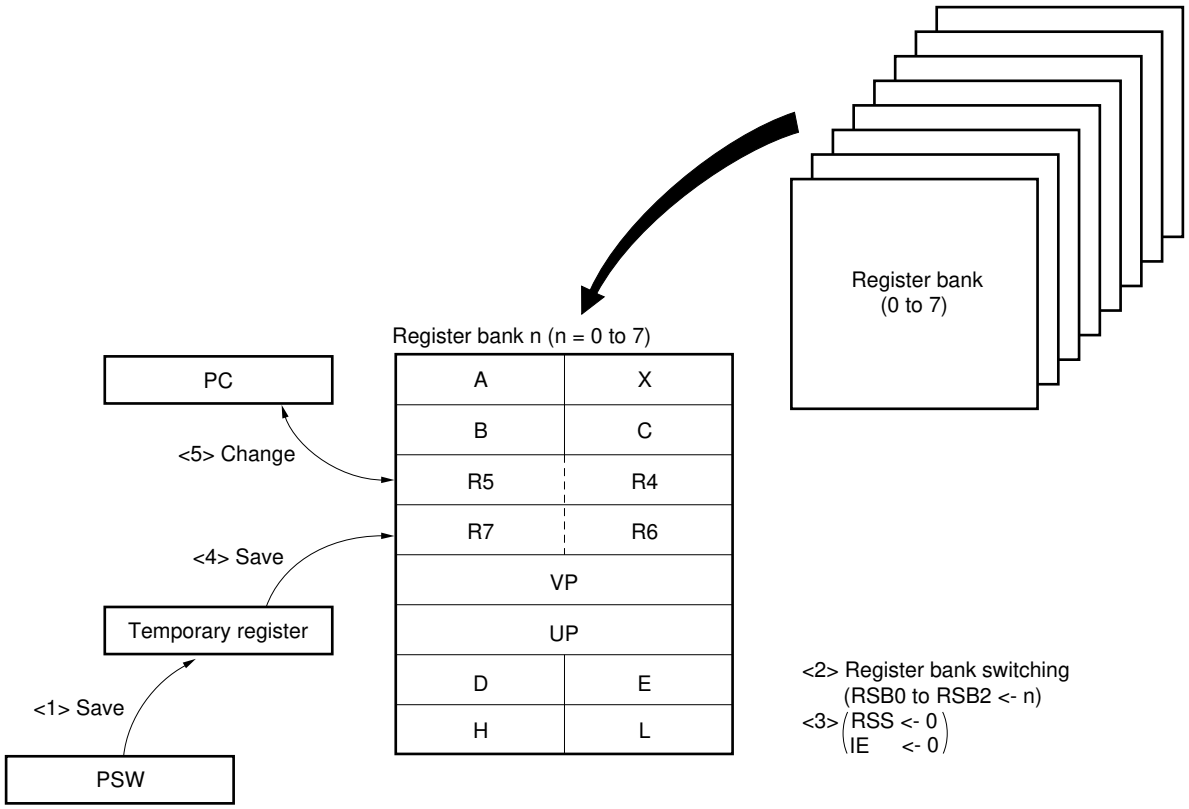
15.5.2 Context switching

Setting the context switching enable flag of the interrupt control register (see **Table 15-4** and **Fig. 15-1**) to (1) enables the context switching function to be activated.

When an interrupt request which is not masked and for which the context switching function is enabled is generated in the EI status, the register bank specified by the low-order 3 bits of the lower address (even address) of the corresponding vector table addresses is selected.

The vector address already stored in the selected register bank is transferred to the PC, and at the same time the PC and PSW contents present immediately before the transfer are saved in the register bank. Then, the function branches to the interrupt service routine.

Fig. 15-8. Context Switching Operation in Response to Interrupt Request

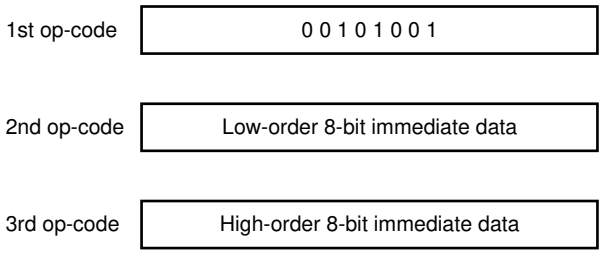


After an interrupt is made by context switching, control is returned by executing the RETCS or RETCSB instruction.

When the RETCS instruction is executed, the contents of the R4 and R5 registers in the selected register bank are transferred to the PC, and the contents of the R6 and R7 registers are transferred to the PSW. Then, the 16-bit immediate data specified by the second and third op-codes of the RETCS instruction is stored in the R4 and R5 registers of the register bank.

When the same register bank is selected again by the context switching function, the 16-bit immediate data specified by the second and third op-codes of the RETCS instruction is used as the branch address.

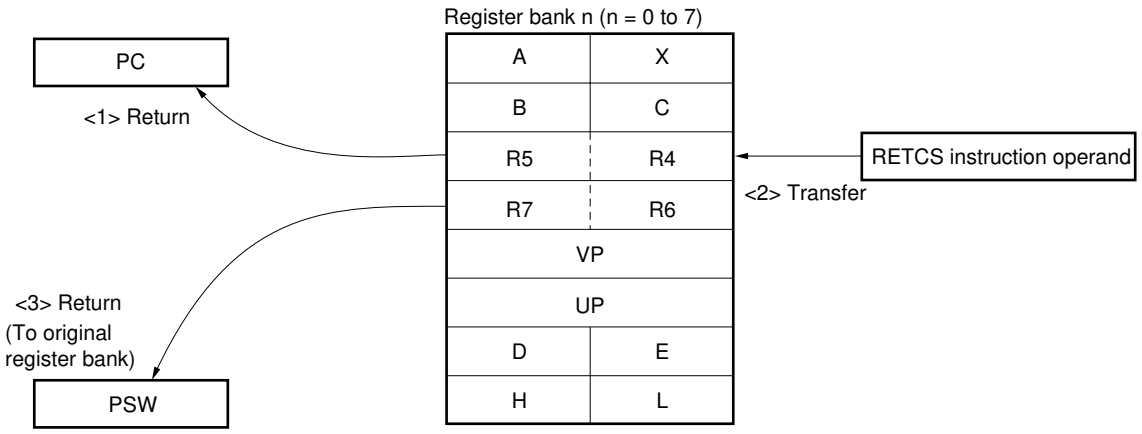
Fig. 15-9. Format of the RETCS Instruction



When control is returned by executing the RETCS instruction after branch operation, the bit for the highest priority among the bits set to (1) in the inservice priority register (ISPR) is reset to 0.

Caution Always use the RETCS instruction when restoring from interrupt by context switching. If other instructions are used, interrupt operations hereafter will not be performed properly.

Fig. 15-10. Return Operation from Interrupt Using Context Switching Function by RETCS Instruction



15.5.3 Maskable interrupt priority

The μPD78356 performs multiple-interrupt servicing which can acknowledge different interrupts during an interrupt servicing.

Multiple interrupts can be handled by the priority level control. The priority order control consists of control according to default priority level and programmable priority level control by setting the priority level specification flag. For priority level control by default priority level, when several interrupts are generated simultaneously, they are handled according to the priority level (default priority level) preassigned to each interrupt request. (See **Table 15-2**). For the programmable priority level control, each interrupt request is divided into four levels by setting the priority level specification flag. Multiple-interrupt service requests are shown in Table 15-5.

As the IE flag is automatically reset (0) when the interrupt is acknowledged, to use Multiple-interrupt service, execute the EI instruction during the interrupt service routine, set the IE flag to (1) and set the interrupt enable state.

Table 15-5. Multiple-interrupt Servicing

Priority of interrupt currently accepted	Value of ISPR	IE flag of PSW	PRSL flag of IMC register	Acceptable maskable interrupt
Interrupt accepted is not provided	00000000	0	x	• All macro service only
		1	x	• All maskable interrupt
3	00001000	0	x	• All macro service only
		1	0	• All maskable interrupt
		1	1	• All macro service • Maskable interrupts whose priority has been specified as 0, 1, and 2
2	0000x100	0	x	• All macro service only
		1	x	• All macro service • Maskable interrupts whose priority has been specified as 0, and 1
1	0000xx10	0	x	• All macro service only
		1	x	• All macro service • Maskable interrupts whose priority has been specified as 0
0	0000xxx1	x	x	• All macro service only

Fig. 15-11. Operation Example When Other Interrupt Requests are Generated during Interrupt Servicing (1/3)

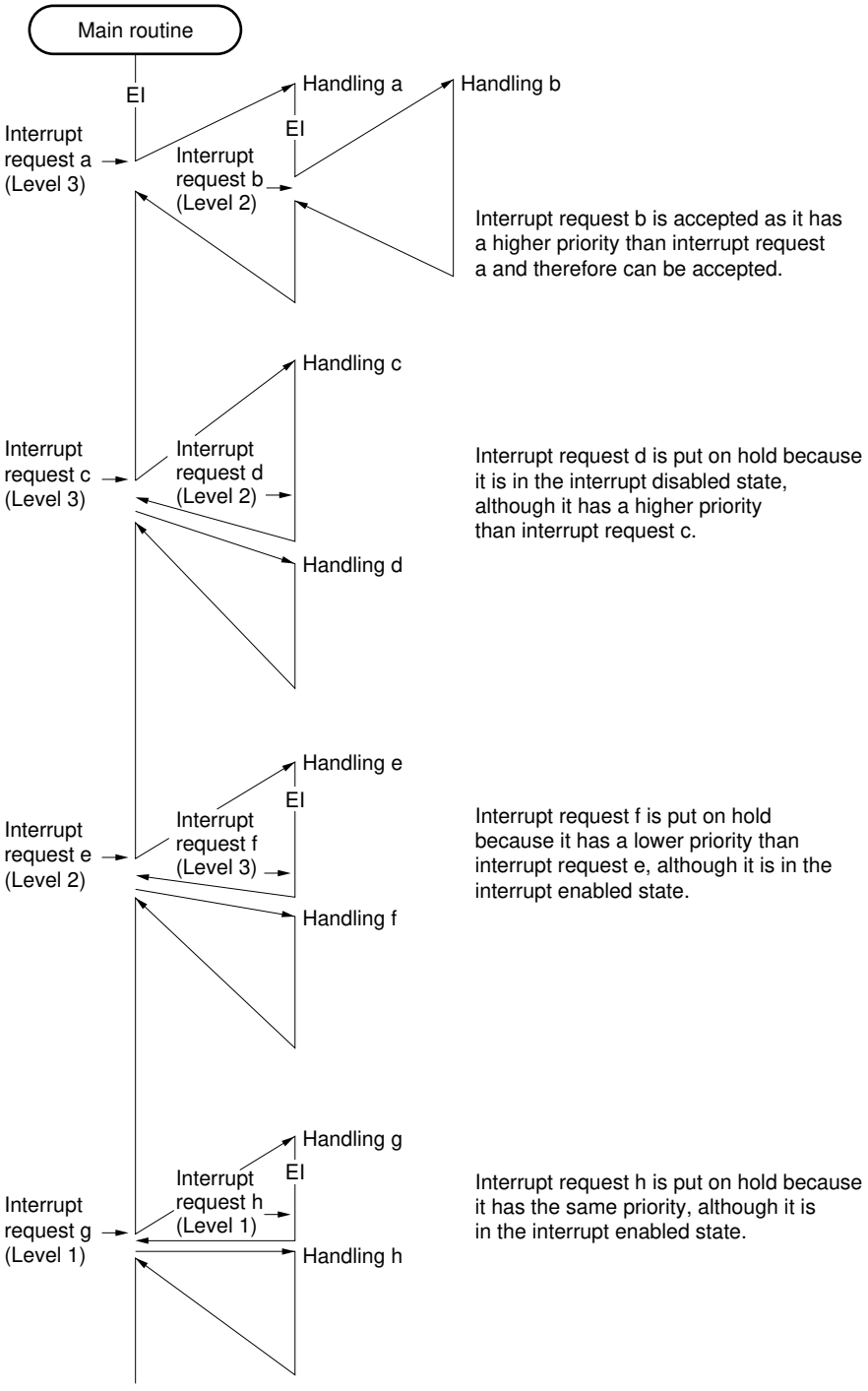


Fig. 15-11. Operation Example When Other Interrupt Requests are Generated during Interrupt Servicing (2/3)

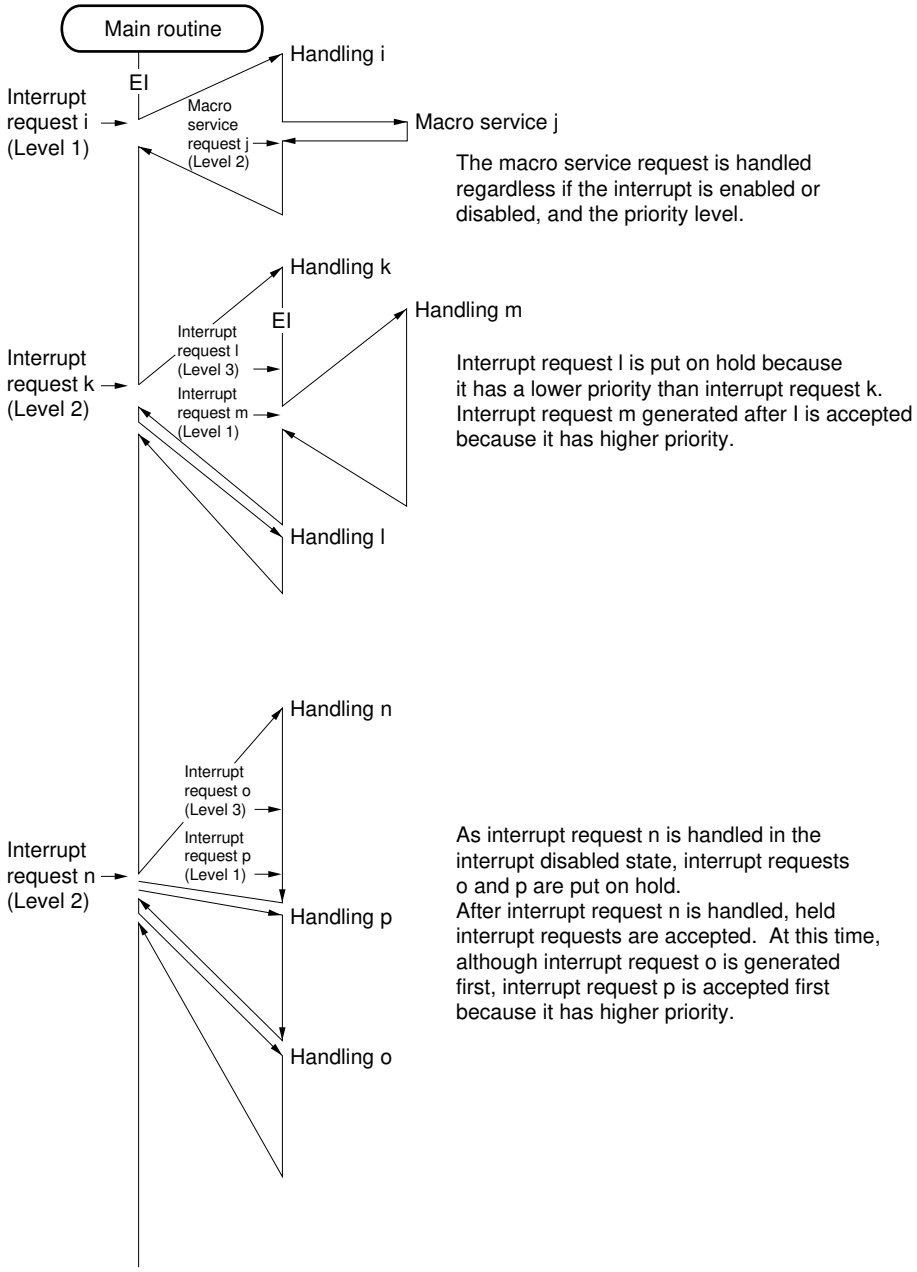
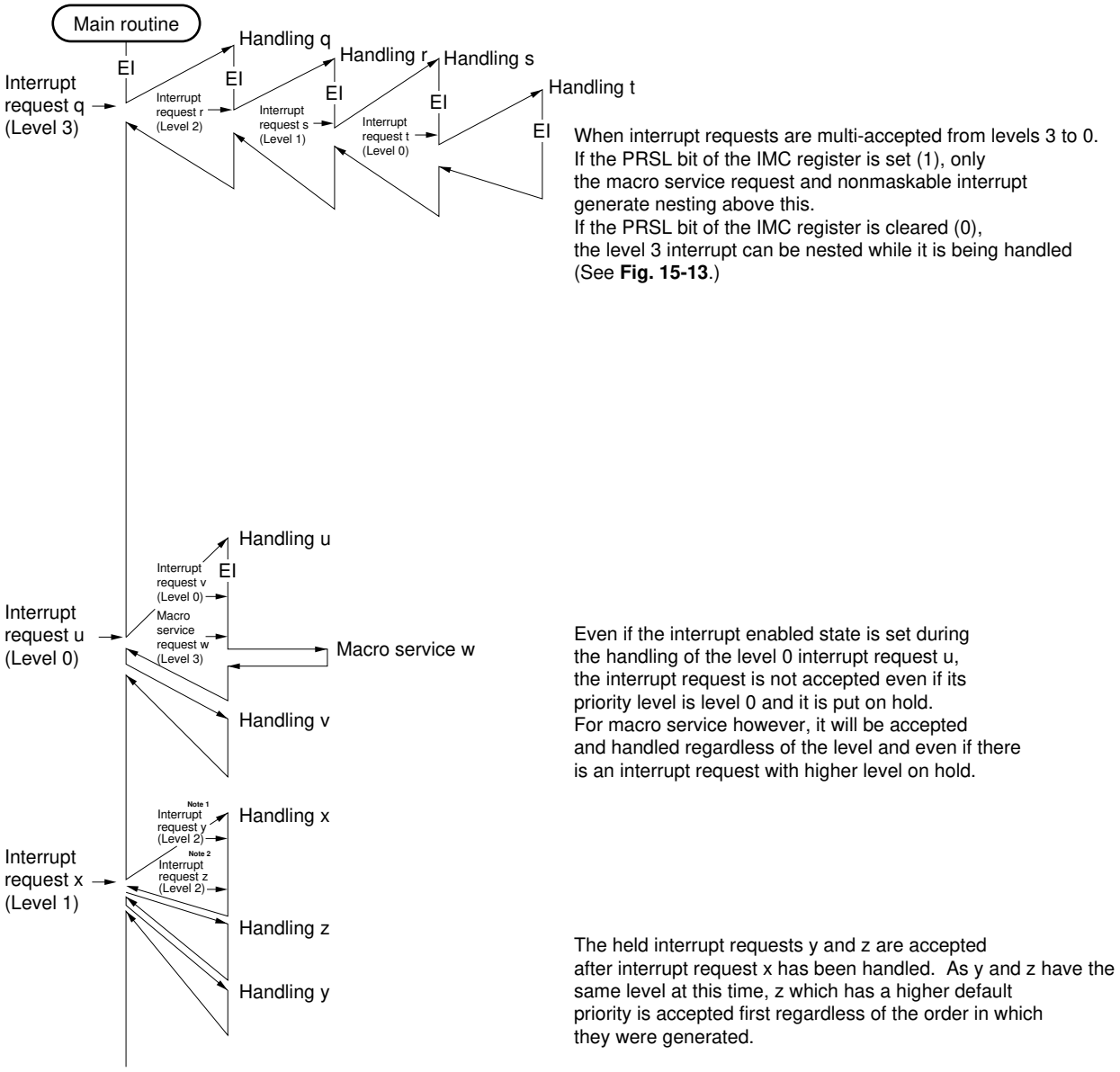


Fig. 15-11. Operation Example When Other Interrupt Requests are Generated during Interrupt Servicing (3/3)



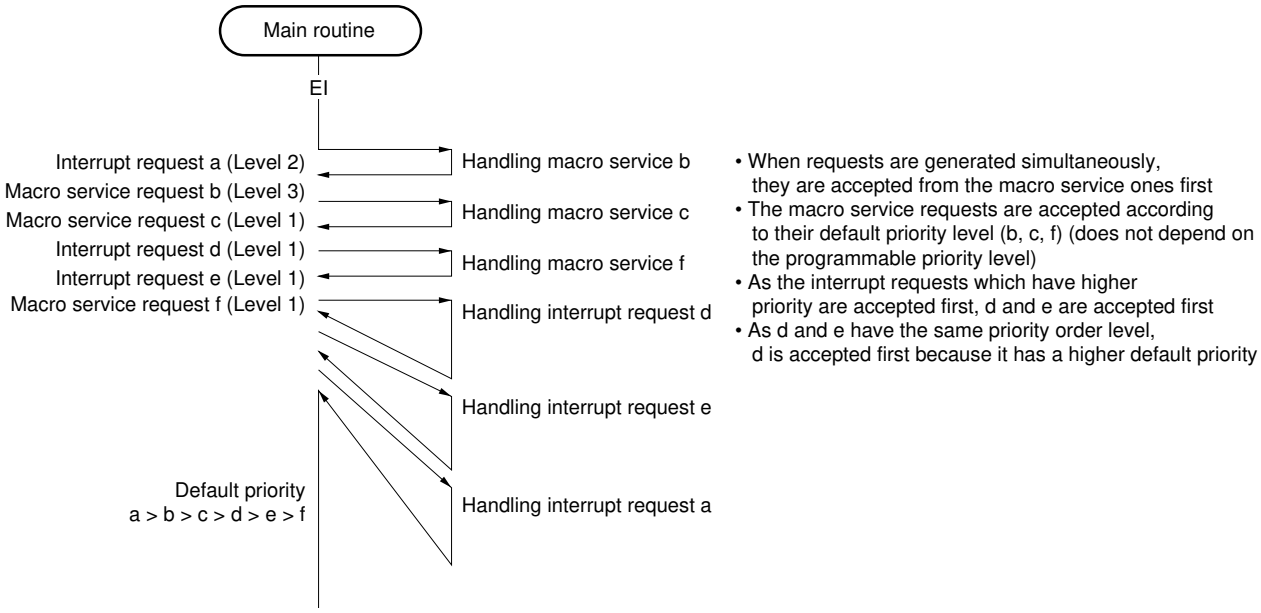
Notes 1. Default priority is low.

2. Default priority is high.

Remarks 1. a to z in the figure is the temporary name given to classify the interrupt requests and the macro service requests.

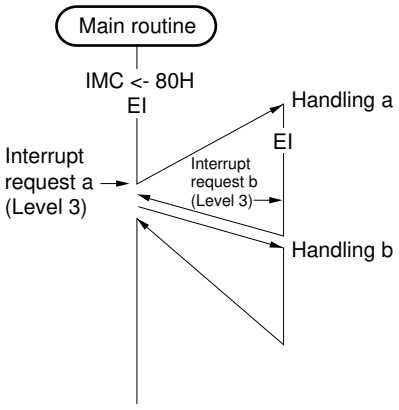
2. The high/low default priority in the figure indicates the relative priority level of two interrupt requests.

Fig. 15-12. Operation Example of Simultaneously Generated Interrupt Requests



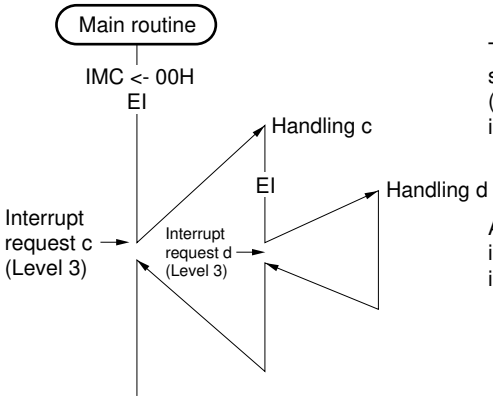
Remark a to f in the figure is the temporary name given to classify the interrupt requests and the macro service requests.

Fig. 15-13. Level 3 Interrupt Acknowledgement Due to IMC Register Setting



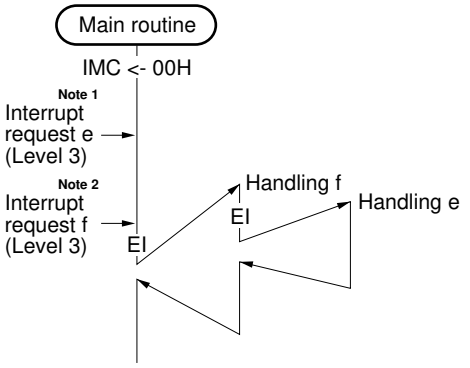
Disable nesting between level 3 interrupts, with the IMC register PRSL as 1.

Interrupt request b is held even if in the interrupt enable state because it has the same priority.



Taking the IMC register PRSL as 0, set so that the level 3 interrupt is accepted (can be nested) even if the level 3 interrupt is being handled.

As the level 3 interrupt request c is handled in the interrupt enabled state and PRSL = 0, interrupt request d of the same level 3 is accepted.



As interrupt request e and f are the same level, interrupt request f which has a higher default priority is accepted first.
If the interrupt enabled state is set while interrupt request f is being handled, the held interrupt request e is accepted because PRSL = 0.

Notes 1. Default priority is low.

2. Default priority is high.

Remarks 1. a to f in the figure is the temporary name given to classify the interrupt requests.

2. The high/low default priority in the figure indicates the relative priority level of two interrupt requests.

15.6 Software Interrupt Acknowledgement

The software interrupt is acknowledged when the BRK and BRKCS instructions are executed. The software interrupt cannot be disabled.

15.6.1 Software interrupt acknowledgement by the BRK instruction

When the BRK instruction is executed, PSW and PC are saved in the stack area in order, the IE flag is reset (0), and the contents of the vector table (003EH, 003FH) are loaded to the PC and branched.

To restore from a software interrupt by the BRK instruction, use the RETB instruction.

Caution To restore from a software interrupt by the BRK instruction, do not use the RETI instruction.

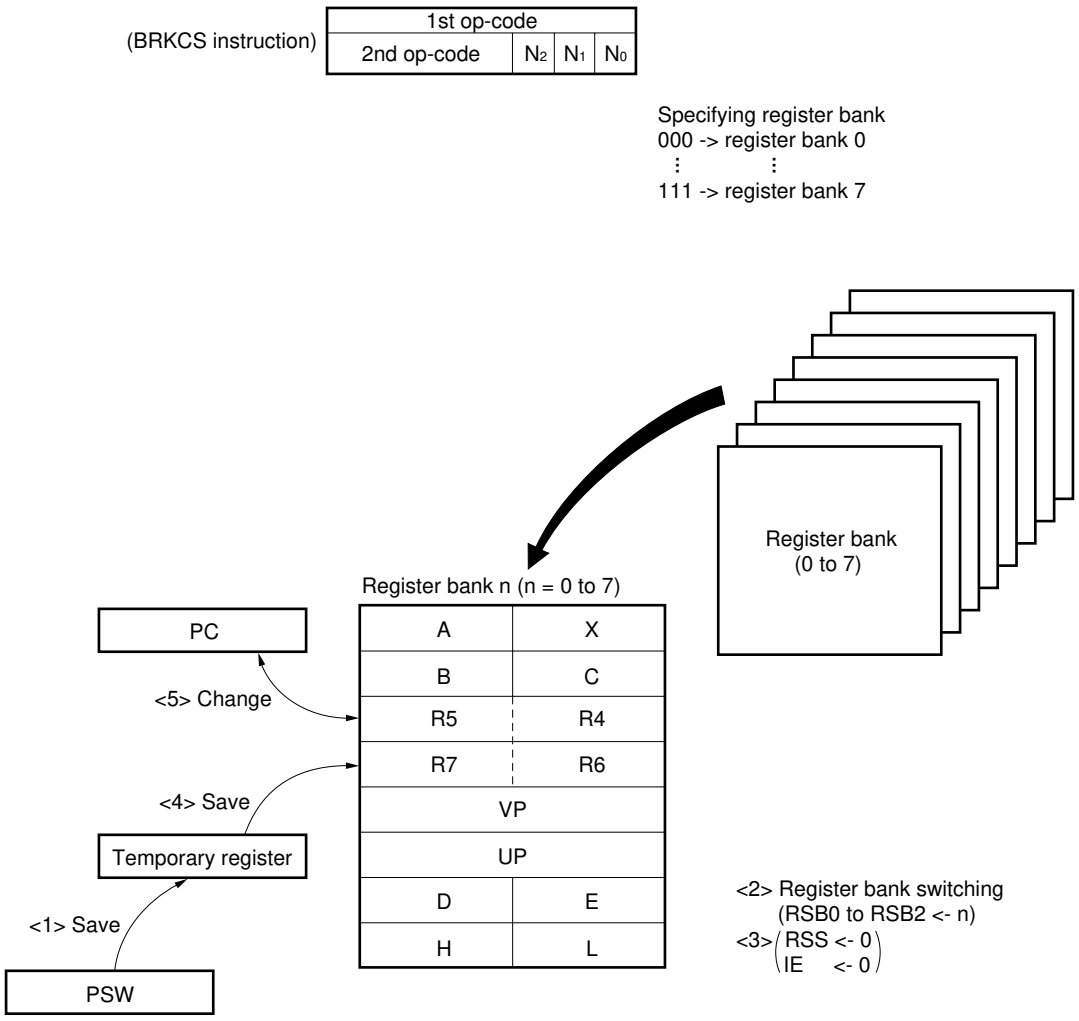
15.6.2 Software interrupt (context switching) acknowledgement by the BRKCS instruction

The context switching function can be activated by executing the BRKCS instruction.

The register bank used after context switching is specified by the immediate data (N_2 to N_0) in the low-order 3 bits of the second op-code of the BRKCS instruction.

When the BRKCS instruction is executed, a branch is made to the vector address stored in the register bank. At the same time the PC and PSW contents present immediately before the branch are saved in the register bank.

Fig. 15-14. Context Switching Operation by the BRKCS Instruction Execution

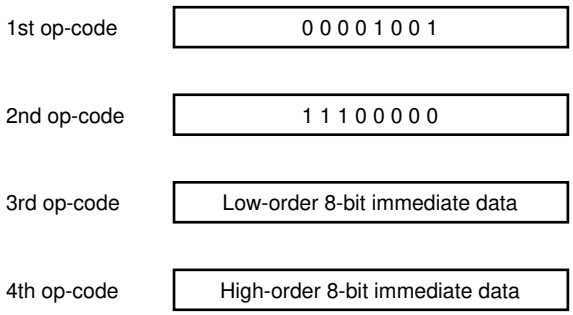


Execute the RETCSB instruction to restore from the software interrupt by the BRKCS instruction.

When the RETCSB instruction is executed, the contents of the R4 and R5 registers in the selected register bank are transferred to the PC, and the contents of the R6 and R7 registers are transferred to the PSW. Then, the 16-bit immediate data specified by the third and fourth op-codes of the RETCSB instruction is stored in the R4 and R5 registers of the register bank.

When the same register bank is selected again by the context switching function, the 16-bit immediate data specified by the third and fourth op-codes of the RETCSB instruction is used as the branch address.

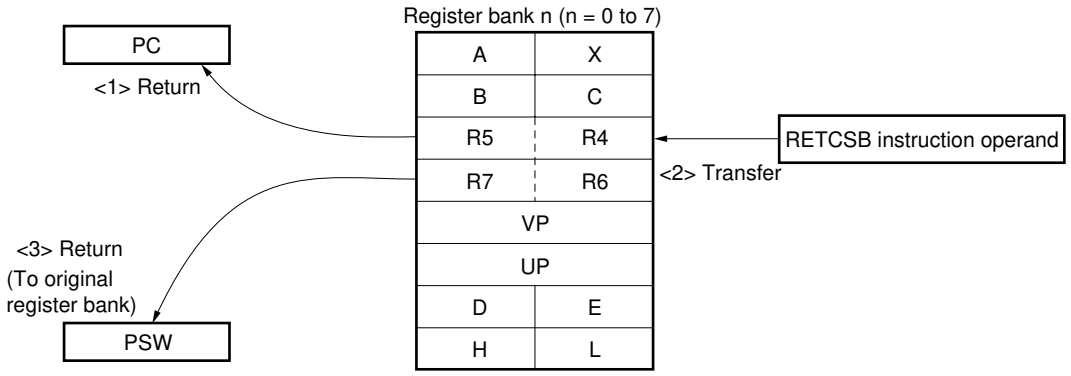
Fig. 15-15. Format of the RETCSB Instruction



Caution When the context switching function is activated by executing the BRKCS instruction, and if the RETCS instruction is executed and a bit of the ISPR register is reset to 0, interrupt nesting control is destroyed.

When returning from processing activated by the BRKCS instruction, be sure to use the RETCSB instruction.

Fig. 15-16. Restoring from Software Interrupt by BRKCS Instruction (RETCSB Instruction Operation)



15.7 Op-code Trap Interrupt Acknowledgement

The op-code trap interrupt is generated when the data obtained by reversing all the bits of the 3 byte operand of the MOV STBC, #byte instruction and MOV WDM, #byte instruction does not match the 4th byte of the operand. The op-code trap interrupt cannot be disabled.

When the op-code trap interrupt is generated, the head address of the instruction generating PSW and errors is saved in the stack, the IE flag is reset (0), the vector table values are loaded in the PC and branched.

As the address saved in the stack area is the head address of the instruction generating errors, if the RETB instruction is executed at the end of the op-code trap interrupt servicing routine, the op-code trap interrupt will be generated again.

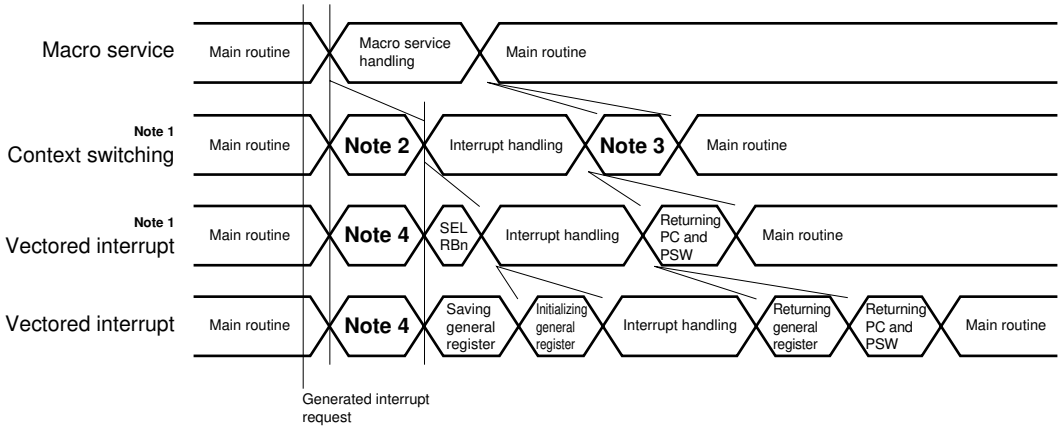
15.8 Macro Service Function

15.8.1 Macro service overview

The macro service is a method of servicing interrupts. For normal interrupts, the program counter (PC) and program status word (PSW) are saved and the head address of the interrupt service program to PC is loaded. In macro service, a different servicing method is performed instead (mainly data transfer). Therefore, responses to interrupt requests can be given quickly. Furthermore, the servicing time is also reduced because high speed transfer servicing can be performed than the use of a program.

The vectored interrupt program can also be simplified because vectored interrupts are generated after the specified number of servicings have been performed.

Fig. 15-17. Difference between Vectored Interrupt and Macro Service Servicing



Notes 1. When register bank switching is used and initial values are preset in the register

2. Register bank switching by context switching, save to PC and PSW

3. Register bank by context switching, return to PC and PSW

4. Save PC and PSW to stack, load vector address to PC

Table 15-6. Interrupt for Which Macro Service Can be Used

Default priority	Interrupt request	Interrupt source	Generation unit	Macro service control word address
0	INTOV0	Timer 0 overflow	Real-time pulse unit	FE06H
1	INTOV3	Timer 3 overflow		FE08H
2	INTP0/INTCC00	INTP0 pin input/CC00 match signal	External/Real-time pulse unit	FE0AH
3	INTP1/INTCC01	INTP1 pin input/CC01 match signal		FE0CH
4	INTP2/INTCC02	INTP2 pin input/CC02 match signal		FE0EH
5	INTP3/INTCC30	INTP3 pin input/CC30 match signal		FE10H
6	INTP4/INTCC31	INTP4 pin input/CC31 match signal		FE12H
7	INTCM00	CM00 match signal	Real-time pulse unit	FE14H
8	INTCM01	CM01 match signal		FE16H
9	INTCM02	CM02 match signal		FE18H
10	INTCM03	CM03 match signal		FE1AH
11	INTCM10	CM10 match signal		FE1CH
12	INTCM11	CM11 match signal		FE1EH
13	INTCM20	CM20 match signal		FE20H
14	INTCM21	CM21 match signal		FE22H
15	INTCM40	CM40 match signal		FE24H
16	INTCMUD0	CMUD0 match signal		FE26H
17	INTCMUD1	CMUD1 match signal		FE28H
18	INTSER	Serial receiver error	Asynchronous serial interface	FE2AH
19	INTSR	Serial reception completion		FE2CH
20	INTST	Serial transmission completion		FE2EH
21	INTCSI0	Serial transmission/reception completion	Clock synchronous serial interface	FE30H
22	INTCSI1	Serial transmission/reception completion	Clock synchronous serial interface (with pin switching function)	FE32H
23	INTAD	A/D conversion completion	A/D converter	FE34H

Remark The default priority level is a fixed value. It is the priority level when macro service requests are generated simultaneously.

The macro service operation has the following five modes.

(1) Counter mode: EVTCNT

The macro service counter (MSC) is increased (+1) or decreased (–1) every time an interrupt request is generated, and a vectored interrupt request is generated when MSC becomes 00H.

This mode is used for dividing the number of interrupt request generated.

(2) Block transfer mode: BLKTRS

A 1-byte or 1-word data is transferred between the special function register (SFR) specified by the SFR pointer (SFRP) and buffer every time an interrupt request is generated, and a vectored interrupt request is generated when data transfer is performed for the number of times specified.

The buffer used in the transfer is limited to the FE00H to FEFFH main RAM.

The specification method is simple and this mode is used for the high speed data transfer of small volume data.

(3) Block transfer mode (with memory pointer): BLKTRS-P

Like the block transfer mode, a 1-byte or 1-word data is transferred between the SFR specified by the SFRP and buffer every time an interrupt request is generated, and a vectored interrupt request is generated when data transfer is performed for the number of times specified.

The buffer used in the transfer is specified by the memory pointer (MEM.PTR) (the memory used is the total 64 Kbyte space).

This mode is used generally for transferring large volume data.

(4) Data difference mode: DTADIF

The difference between the current value of the SFR specified by the SFRP and “the value immediately before” stored in the memory is written in the buffer every time an interrupt request is generated, and the current value is taken as “the value immediately before”.

A vectored interrupt request is generated when data transfer is performed for the number of times specified.

The buffer used in the transfer is limited to the FE00H to FEFFH main RAM.

This mode is used for measuring the period of input pulses and pulse width by the capture register.

(5) Data difference mode (with memory pointer): DTADIF-P

Like the data difference mode, the difference between the current value of the SFR specified by the SFRP and “the value immediately before” stored in the memory is written in the buffer every time an interrupt request is generated, and the current value is taken as “the value immediately before”.

Vectored interrupt request is generated when data transfer is performed for the number of times specified.

The buffer used in the transfer is specified by the memory pointer (MEM. PTR) (the memory used is the total 64 Kbyte space).

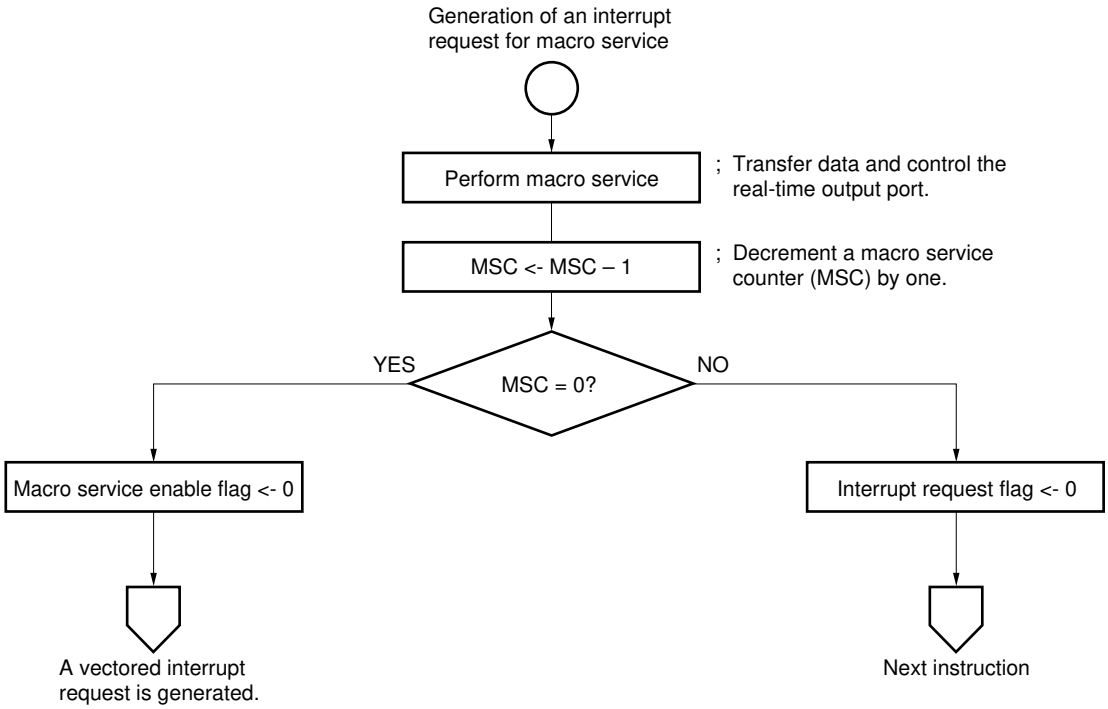
This mode is used generally for transferring large volume data.

15.8.2 Basic operation of macro service

The macro service function is used to transfer data on a hardware basis between the special function register area and memory space in response to an interrupt request.

When a macro service request is generated, the CPU temporarily stops program execution, and one- or two-byte data is transferred automatically between special function registers (SFRs) and memory. When data transfer ends, the interrupt request flag is reset to 0, then the CPU restarts program execution. After data transfer is performed as many times as the value set in the macro service counter (MSC), a vectored interrupt request is generated.

Fig. 15-18. Sample Sequence of Macro Service Processing



Unlike other interrupt servicing that is accompanied by the activation of an interrupt service routine, macro service is performed automatically. This means that macro service does not require a series of the operations of making a branch to an interrupt service routine, saving and restoring register contents, and returning from the interrupt service routine. It is therefore possible to improve the CPU service time and to reduce the number of program steps.

During macro service processing, the general-purpose registers and instruction queue in the CPU hold the data present immediately before macro service processing is activated.

Interrupt requests for which macro service processing is specified are not influenced by the status of the IE flag in the program status word (PSW). Macro service can be performed even in the interrupt disable status or while the interrupt service program is being executed. Macro service is disabled only when the corresponding bit of the interrupt mask flag registers (MK0 and MK1) is set to (1).

If more than one macro service request is present, the service order is determined by the default priority. Instruction execution is suspended until all of these macro service requests are processed.

The μPD78356 supports macro service for all included interrupt requests.

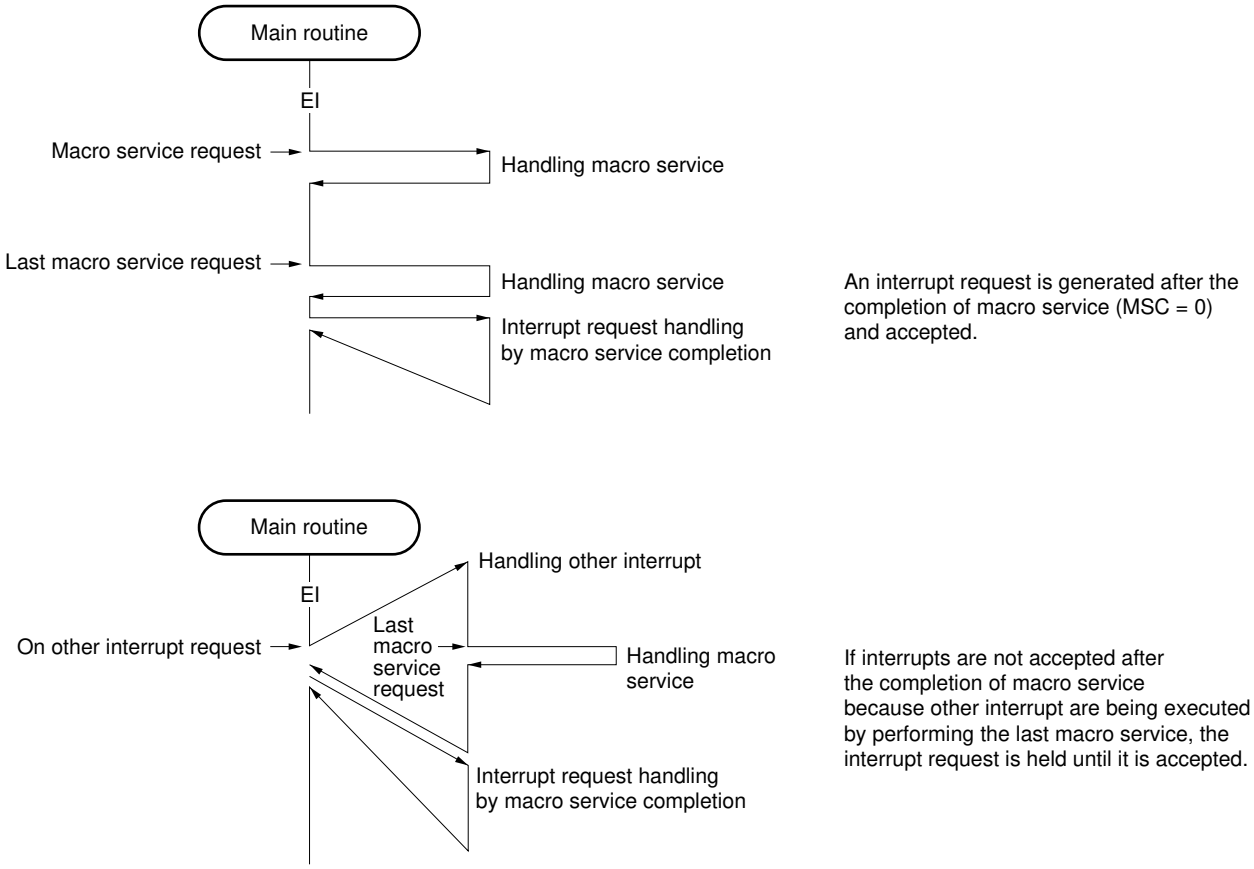
The followings are two basic operations of macro service processing.

- Data transfer from memory to special function registers (SFR)
- Data transfer from special function registers (SFR) to memory

15.8.3 Macro service completion

After the macro service performs for the number of times specified during the execution of other programs, the macro service completes (when the macro service counter (MSC) becomes 0).

Fig. 15-19. Operation when Macro Service Completes



★ **Caution When transmitting data by the UART using macro service, two vectored interrupt requests occur. (See 10.7)**

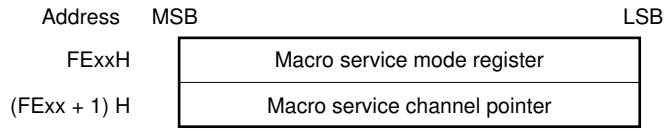
15.8.4 Macro service control register

(1) Macro service control word

A macro service control word is made up of a macro service mode register which controls the macro service function and a macro service channel pointer. These control words are located in addresses FE06H to FE35H in the main RAM area. (See **Fig. 15-21.**)

Fig. 15-20 shows the basic structure of a macro service control word.

Fig. 15-20. Basic Structure of a Macro Service Control Word



The macro service mode register sets the macro service processing mode, and the macro service channel pointer specifies the address of a macro service channel.

Before macro service processing can be performed, values must be loaded into the macro service mode register and channel pointer corresponding to the interrupt request for which macro service processing can be specified.

Fig. 15-21. Macro Service Control Word Format

Address		Factor
FE06H	Mode register	INTOV0
FE07H	Channel pointer	
FE08H	Mode register	INTOV3
FE09H	Channel pointer	
FE0AH	Mode register	INTP0/INTCC00
FE0BH	Channel pointer	
FE0CH	Mode register	INTP1/INTCC01
FE0DH	Channel pointer	
FE0EH	Mode register	INTP2/INTCC02
FE0FH	Channel pointer	
FE10H	Mode register	INTP3/INTCC30
FE11H	Channel pointer	
FE12H	Mode register	INTP4/INTCC31
FE13H	Channel pointer	
FE14H	Mode register	INTCM00
FE15H	Channel pointer	
FE16H	Mode register	INTCM01
FE17H	Channel pointer	
FE18H	Mode register	INTCM02
FE19H	Channel pointer	
FE1AH	Mode register	INTCM03
FE1BH	Channel pointer	
FE1CH	Mode register	INTCM10
FE1DH	Channel pointer	
FE1EH	Mode register	INTCM11
FE1FH	Channel pointer	
FE20H	Mode register	INTCM20
FE21H	Channel pointer	
FE22H	Mode register	INTCM21
FE23H	Channel pointer	
FE24H	Mode register	INTCM40
FE25H	Channel pointer	
FE26H	Mode register	INTCMUD0
FE27H	Channel pointer	
FE28H	Mode register	INTCMUD1
FE29H	Channel pointer	
FE2AH	Mode register	INTSER
FE2BH	Channel pointer	
FE2CH	Mode register	INTSR
FE2DH	Channel pointer	
FE2EH	Mode register	INTST
FE2FH	Channel pointer	
FE30H	Mode register	INTCSI0
FE31H	Channel pointer	
FE32H	Mode register	INTCSI1
FE33H	Channel pointer	
FE34H	Mode register	INTAD
FE35H	Channel pointer	

(2) Macro service mode register

The macro service mode registers are 8-bit registers that specify macro service operation. They are mapped as part of the macro service control words in the main RAM area (see **Fig. 15-20**).

15.8.5 Macro service mode

Operation of the macro service is specified by setting a macro service mode register. The macro service mode is determined by the low-order 6 bits of the macro service mode register. The modes are classified into two groups: Group 0 and group 1.

- Group 0: Control word only; no channel provided
- Group 1: Control word and channel

The high-order 2 bits of the macro service mode register function as a subcommand (see **Table 15-7**).

15.8.6 Macro service operation

There are five modes of macro service operation.

Table 15-7. Classification of Macro Service Modes

Group	Macro service mode register	Function	
Group 0	CC000001	Counter mode	EVTCNT
Group 1	CC010011	Block transfer mode	BLKTRS
	CC010100	Block transfer mode (with memory pointer)	BLKTRS-P
	10011001	Data difference mode	DTADIF
	10011010	Data difference mode (with memory pointer)	DTADIF-P

C in the most significant bit (MSB) of a macro service mode register indicates the length of the data to be operated on except for EVTCNT.

- C = 0: Byte data
- C = 1: Word data

As to BLKTRS and BLKTRS-P, byte buffer representation is always used. If word specification is made, byte buffers must be replaced by word buffers.

Caution Word buffers must be placed at even addresses.

(1) Counter mode: EVTCNT

[Macro service control word]

	MSB LSB
Lower address	CC000001
Higher address	MSC

[Operation]

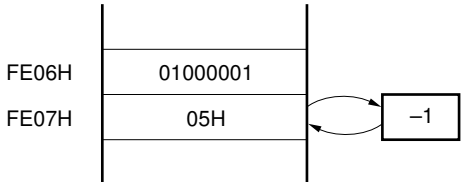
Every time the macro service is generated, the macro service counter (MSC) is incremented or decremented by one. When the MSC reaches 00H (overflow), a vectored interrupt request is generated.

Table 15-8. Counter Mode Operation Specification

CC	Operation
00	Increment
01	Decrement
10	Setting prohibited
11	

In the counter mode, the macro service function operates as a counter to divide the number of interrupt requests.

Example: Dividing the number of INTOV interrupt requests by five using macro service

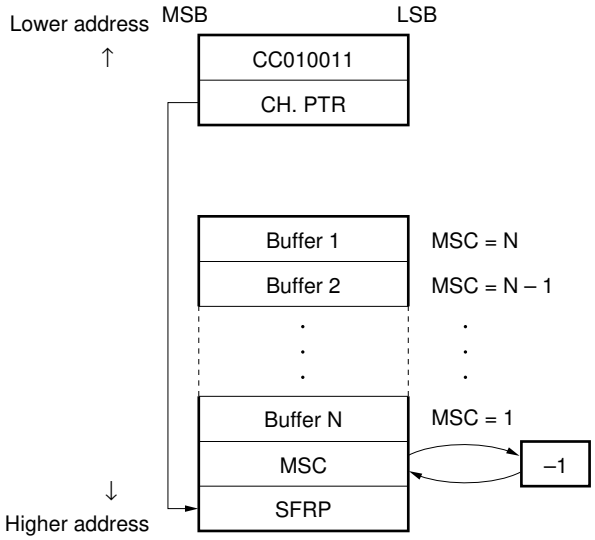


[Application]

Event counter, or measurement of the number of capture times

(2) Block transfer mode: BLKTRS

[Macro service control word]



[Operation]

The channel pointer (CH.PTR) specifies the SFR pointer (SFRP). A buffer is addressed by the channel pointer (CH.PTR) and macro service counter (MSC).

Data is transferred between the SFR pointed to by the SFRP and buffers. Data transfer starts from buffer 1.

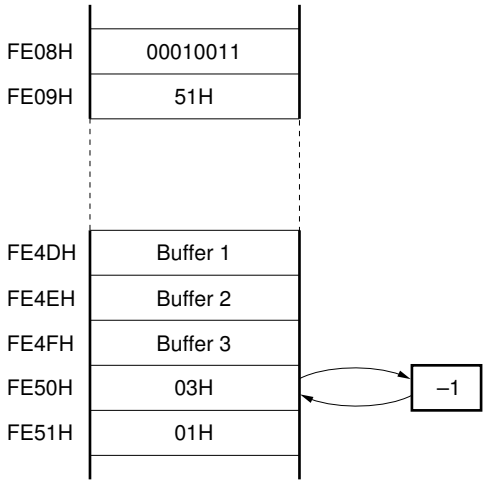
Every time transfer terminates, the MSC is decremented by one.

When the MSC reaches 0, a vectored interrupt request is generated.

Table 15-9. Specification of Block Transfer Mode Operation

CC	Operation	Transferred data	Buffer address
00	Buffer <- SFR	Byte	(CH.PTR content) – (MSC content) – 1
01	SFR <- buffer		
10	Buffer <- SFR	Word	(CH.PTR content) – (MSC content x 2) – 1
11	SFR <- buffer		

Example: Transferring the contents of port 1 (FF01H) to buffers in response to the INTP0 interrupt request



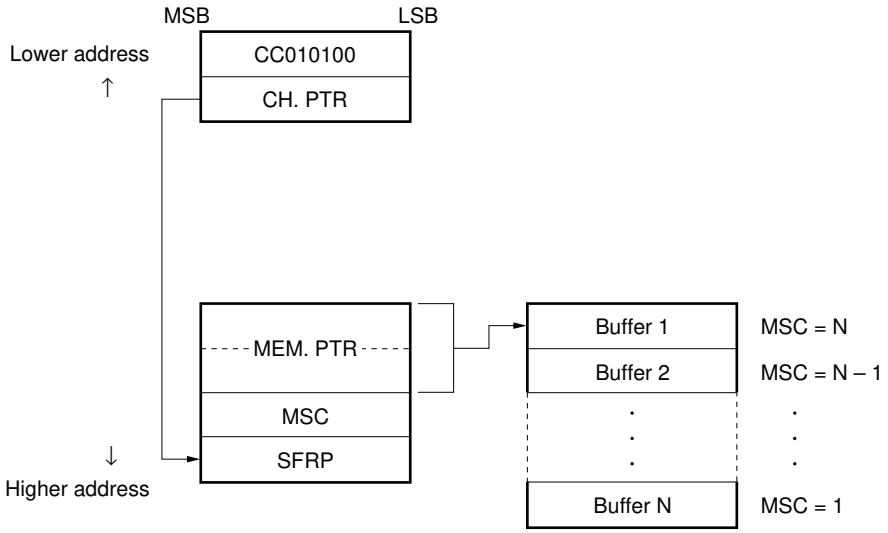
[Application]

This mode can be used for data transfer over a serial interface.

Caution Word buffers must be placed at even addresses.

(3) Block transfer mode (with a memory pointer): BLKTRS-P

[Macro service control word]



[Operation]

The channel pointer (CH.PTR) specifies the SFR pointer (SFRP). Data is transferred between the SFR pointed to by the SFRP and the buffer addressed by MEM.PTR. Data transfer starts from buffer 1.

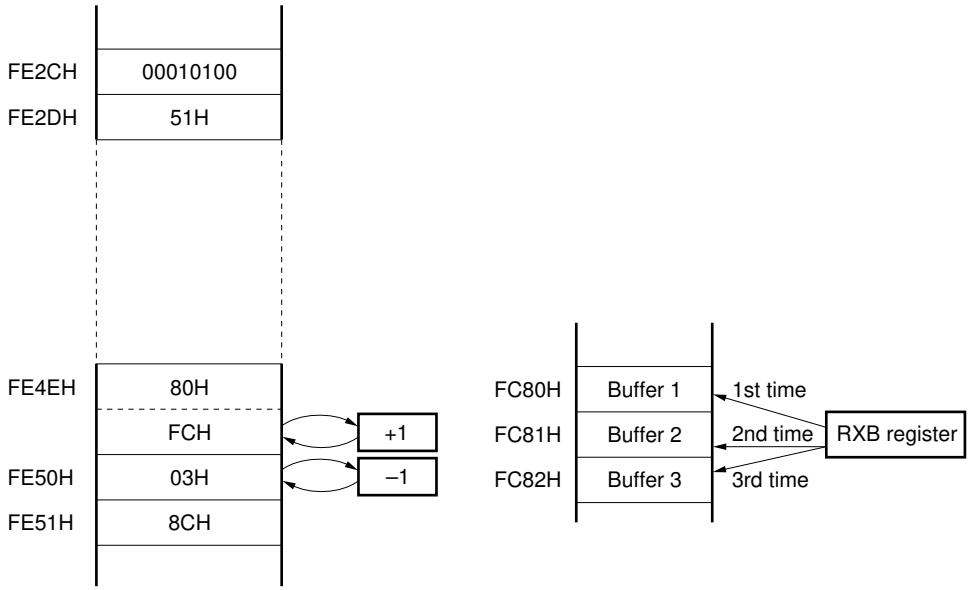
When byte data transfer terminates, the MEM.PTR is incremented by one. When word data transfer terminates, the MEM.PTR is incremented by two.

Every time transfer terminates, the macro service counter (MSC) is decremented by one. When the MSC reaches 0, a vectored interrupt request is generated.

Table 15-10. Specification of Block Transfer Mode (with a Memory Pointer) Operation

CC	Operation	Transferred data
00	Buffer <- SFR	Byte
01	SFR <- buffer	Byte
10	Buffer <- SFR	Word
11	SFR <- buffer	Word

Example: Transferring the contents of serial reception buffer RXB (FF8CH) to buffers in response to the INTSR interrupt request



[Application]

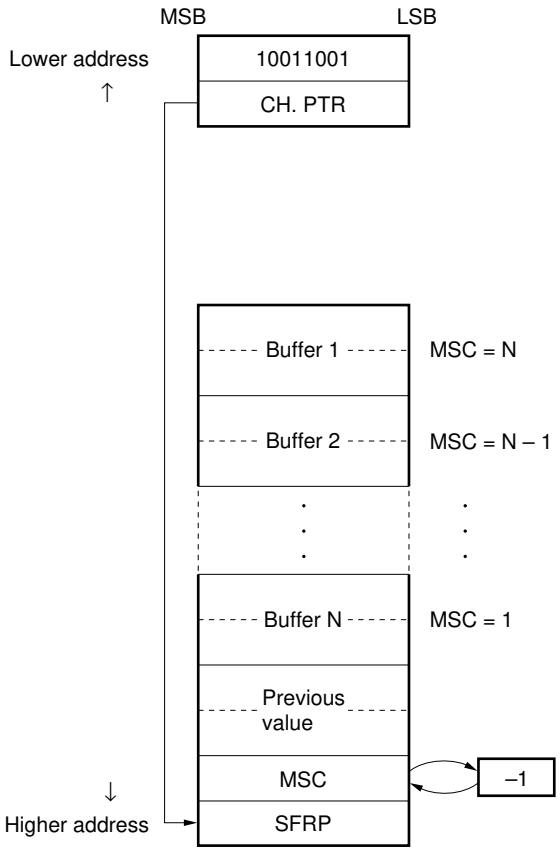
This mode is used for data transfer over a serial interface.

Cautions 1. Word buffers must be placed at even addresses.

2. MEM.PTR must be placed at an even address.

(4) Data difference mode: DTADIF

[Macro service control word]



[Operation]

The channel pointer (CH.PTR) specifies the SFR pointer (SFRP). A buffer is addressed by the channel pointer (CH.PTR) and macro service counter (MSC).

The difference between the current value of the SFR (especially capture register) pointed to by the SFRP and the previous value is written to the buffer. The current value of the SFR is then regarded as the new "previous value." Data write starts from buffer 1.

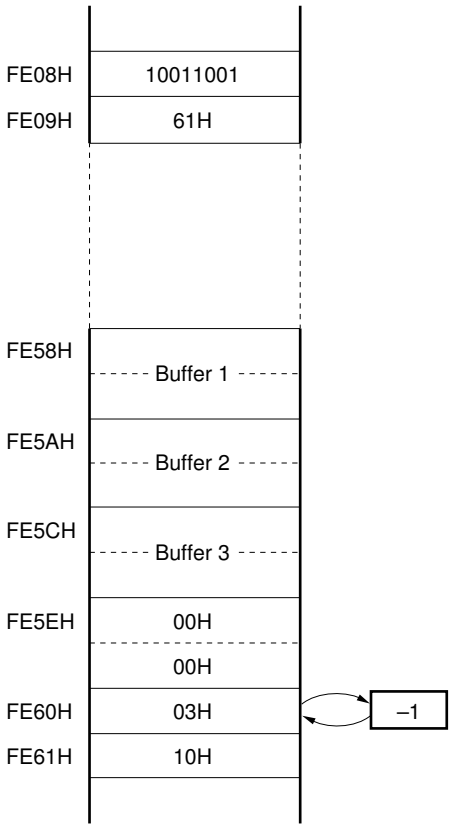
Every time data is written, the MSC is decremented by one.

When the MSC reaches 0, a vectored interrupt request is generated.

The buffer address is obtained as follows:

(Buffer address) = (value of CH.PTR) – (value of MSC x 2) – 3

Example: The difference between the current value and previous value of capture/compare register CC00 (FF10H) is written into a buffer with the INTP0 input signal used as the trigger signal. The period of the INTP0 input signal is then measured using the difference by the vectored interrupt service routine.



[Application]

This mode can be used to measure periods or pulse widths using a capture register.

Cautions 1. 00H must not be set in the MSC.

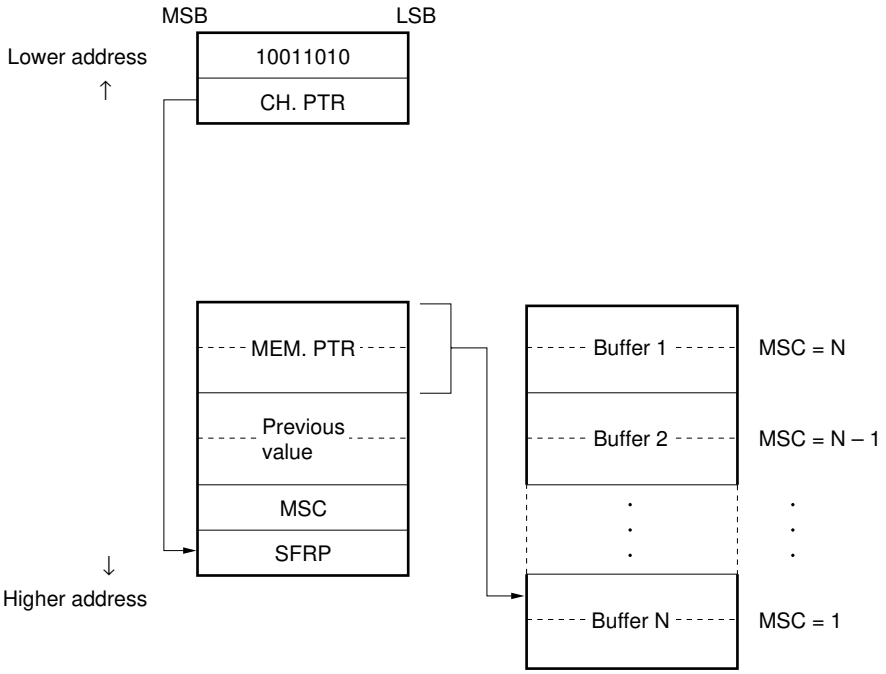
2. Buffers must be placed at even addresses.

3. The "previous value" must be initialized to dummy data in advance.

4. The SFRP can specify 16-bit SFRs only.

(5) Data difference mode (with a memory pointer): DTADIF-P

[Macro service control word]



[Operation]

The channel pointer (CH.PTR) specifies the SFR pointer (SFRP). A buffer is addressed by the MEM.PTR and macro service counter (MSC).

The difference between the current value of the SFR (especially capture register) specified by the SFRP and the previous value is written to the buffer. The current value of the SFR is then regarded as the new "previous value." Data write starts from buffer 1.

Every time data is written, the MSC is decremented by one. When the MSC reaches 0, a vectored interrupt request is generated.

The MEM.PTR remains unchanged.

The buffer address is obtained as follows:

(Buffer address) = (value of MEM.PTR) – (value of MSC x 2) + 2

[Application]

This mode is used to measure periods and pulse widths using a capture register.

Cautions 1. 00H must not be set in the MSC.

2. Buffers must be placed at even addresses.

3. The MEM.PTR must be placed at an even address.

4. The "previous value" must be initialized to dummy data in advance.

5. The SFRP can specify 16-bit SFRs only.

15.9 When Interrupt Requests and Macro Service are Held Temporarily

(1) Instructions which the interrupt request is not acknowledged

When the following instructions are executed, the acceptance of interrupts and servicing of macro service are held temporarily. The software interrupt is however not held.

MOV1	PSWL. bit, CY	RETB	
MOV1	PSWH. bit, CY	RETI	
SET1	PSWL. bit	RETCS	!addr16
CLR1	PSWL. bit	RETCSB	!addr16
NOT1	PSWL. bit	POP	PSW
BFSET	PSWL. bit, $addr16	POPU	post
BFSET	PSWH. bit, $addr16	EI	
BTCLR	PSWL. bit, $addr16	DI	
BTCLR	PSWH. bit, $addr16		
BRK			
BRKCS	RBn		

(2) Instructions with which interrupt requests may not be acknowledged

The instructions shown in Table 15-11 mainly perform writing in various interrupt servicing control registers (INTCreg). When they are executed, the acceptance of interrupts and servicing of macro service may be held temporarily according to the CPU state.

Remark INTCreg: Registers MK0H, MK0L, MK1L, ISPR and IMC and interrupt control registers (OVIC0, OVIC3, PIC0 to PIC4, CMIC00 to CMIC03, CMIC10, CMIC11, CMIC20, CMIC21, CMIC40, CMICUD0, CMICUD1, SERIC, SRIC, STIC, CSIIC0, CSIIC1, ADIC)

Table 15-11. List of Instructions with which Interrupt Requests may not be Acknowledged

Instrucitons	Mnemonic	Operand	Condition
8-bit data transfer	MOV	INTCreg, #byte	
		INTCreg, A	
		mem, A	To set mem to INTCreg address
		[saddrp], A	To set saddrp to INTCreg address
		!addr16, A	To set addr16 to INTCreg address
		PSWL, #byte	
		PSWH, #byte	
		PSWL, A	
		PSWH, A	
	XCH	A, mem	To set mem to INTCreg address
		A, [saddrp]	To set saddrp to INTCreg address
16-bit data transfer	MOVW	INTCreg, #word	
		INTCreg, AX	
		!INTCreg, rp1	
		mem, AX	To set mem to INTCreg address
	XCHW	AX, INTCreg	
		AX, mem	To set mem to INTCreg address
8-bit arithmetic/ logical operation	ALU	INTCreg, #byte	ALU: ADD, ADDC, SUB, SUBC, AND, OR, XOR
		mem, A	To set mem to INTCreg address ALU: ADD, ADDC, SUB, SUBC, AND, OR, XOR
16-bit arithmetic/ logical operation	ALUW	INTCreg, #word	ALUW: ADDW, SUBW
Bit manipulation	MOV1	INTCreg. bit, CY	
	BIT	INTCreg. bit	BIT: SET1, CLR1, NOT1
Stack manipulation	POP	INTCreg	
Branch with conditions	BTCLR BFSET	INTCreg. bit, $addr16 INTCreg. bit, $addr16	
String	STRING	[DE+], A [DE–], A [DE+], [HL+] [DE–], [HL–]	To set INTCreg address to destination (DE register) STRING: MOVM, MOVBK, XCHM, XCHBK, CMPME, CMPBKE, CMPMNE, CMPBKNE, CMPMC, CMPBKC, CMPMNC, CMPBKNC

Cautions 1. When polling interrupt related registers using BTCLR instruction, etc., make sure one instruction is not branched to the same instruction. If programs in which instructions are branched to themselves are written, all interrupts and macro service are held until conditions with which the instructions will not branch to themselves are established.

```
        Wrong example
          ⋮
LOOP:   BTCLR PIC0.7, $LOOP     All interrupts and macro services are held until PIC0.7 becomes 1.
        xxx                  <- The interrupts and macro services are handled first after the instruction
          ⋮                     after the BTCLR instruction is executed.

        Correct example (1)
          ⋮
LOOP:   NOP
        BTCLR PIC0.7, $LOOP  <- As the interrupts and macro services are handled after the execution
          ⋮                     of the NOP instruction, they will not be held for a long period of time.

        Correct example (2)
          ⋮
LOOP:   BTCLR PIC0.7, $NEXT
        BR $LOOP             <- As the interrupts and macro services are handled after the execution
NEXT:     ⋮                     of the BR instruction, they will not be held for a long period of time.
```

Cautions 2. If the above instructions are to be used continuously due to the same reasons, and the interrupts and macro services are going to be held for a long time, insert NOP instruction, etc. halfway and create timings with which the interrupts and macro services can be acknowledged.

15.10 Instructions Which are Stopped Temporarily in Interrupts and Macro Services

The execution of the following instructions are stopped temporarily when acceptable interrupts and macro services are requested and these can be acknowledged. The stopped instructions are restarted after the completion of the interrupt servicing or macro service servicing.

<Temporarily stopped instructions>

MOVM, XCHM, MOVBK, XCHBK
CMPME, CMPMNE, CMPMC, CMPMNC
CMPBKE, CMPBKNE, CMPBKC, CMPBKNC
SACW

[MEMO]

CHAPTER 16 STANDBY FUNCTION

16.1 Function Overview

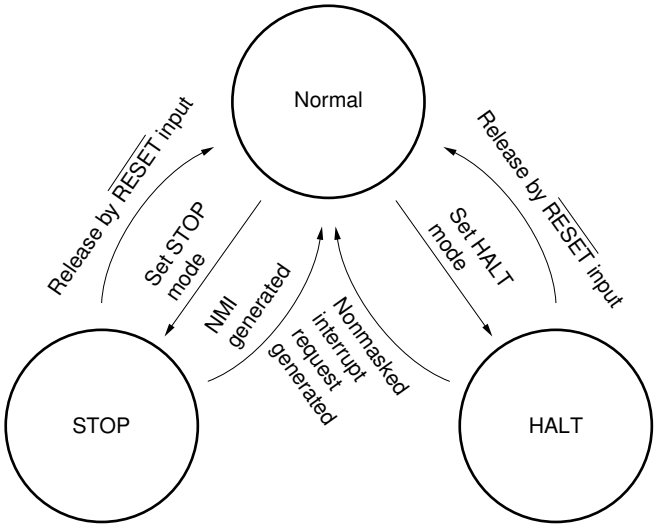
The μPD78356 has a standby function to reduce power consumption of the system. With the standby function, two modes are available:

- HALT mode ..... In this mode, the CPU operation clock is stopped. Intermittent operation, when combined with the normal operation mode, can reduce overall system power consumption.
- STOP mode ..... In this mode, the oscillator is stopped to stop the entire system. Since only leakage currents may flow in this mode, system power consumption can be minimized.

Each mode is set by software.

Fig. 16-1 is the transition diagram of the standby modes (STOP and HALT modes).

Fig. 16-1. Transition Diagram of the Standby Modes



16.2 Standby Control Register (STBC)

The standby control register (STBC) is an 8-bit register which controls the standby mode.

As the contents of the STBC register can only be changed by a dedicated instruction, it will not be changed by a program crash. The dedicated instruction MOV STBC, #byte is shown below.

Fig. 16-2. STBC Register Write Instruction

	B1	B2	B3	B4
MOV STBC, #byte	09H	C0H	data0	data1

This instruction has a data check function to prevent the application system stopping unintentionally by a program crash. Data can be written in the STBC register only when the last two bytes of the instruction are the complement of each other, i.e., when $\overline{\text{data0}}$ = data1.

If they are not complementary, the STBC resister is not written to and an op-code trap interrupt occurs. In this case, the address of the instruction which causes the trap is saved in the stack area as return address. Therefore, the program can be restarted from the return address by checking the trap address or executing the RETB instruction. (See sections **15.1.4** and **15.7**.)

However, if the cause of an op-code trap cannot be removed due to a hardware error, the RETB instruction may cause an infinite loop.

The SBF bit of the STBC register is a standby flag. It is used to determine whether to release the STOP mode by the $\overline{\text{RESET}}$ input. This bit is reset to 0 only when the supply voltage (V_{DD}) is supplied (power-on reset). It cannot be reset by software. It is not affected simply by supplying the $\overline{\text{RESET}}$ signal.

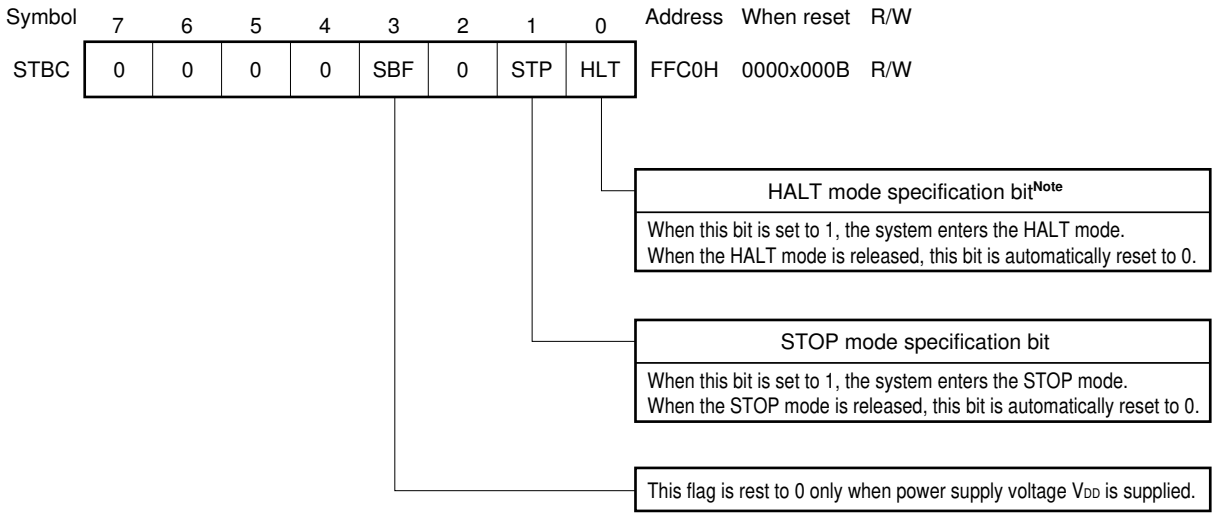
The contents of the STBC register can be read by the data transfer instruction at any time.

$\overline{\text{RESET}}$ input sets the STBC register to 0000x000B.

Fig. 16-3. shows the format of the STBC register.

Caution After the SBF flag is read, set it to 1. Software can then discriminate a power-on reset from release of the STOP or HALT mode.

Fig. 16-3. Format of the Standby Control Register



Note If the HALT mode is released by macro service activation, this bit is reset to 0 when a vectored interrupt request is issued at the end of the macro service.

Caution When an external clock is used, do not set the STP bit of the standby control register (STBC) to 1.

16.3 Operation

16.3.1 HALT mode

(1) Setting the HALT mode and operation states in the HALT mode

In the HALT mode, the CPU clock is stopped.

The total power consumption of the system can be reduced by setting the system to the HALT mode during CPU dead time. The system is put in the HALT state by setting the HLT bit in the STBC register to 1.

In the HALT mode, the CPU clock and program execution are stopped. Nevertheless, the contents of all registers and internal RAM immediately before the HALT mode is set are retained. On-chip peripheral hardware can operate. Hardware functions enter the states shown in Table 16-1.

Caution If the interrupt request flag (xxIF) is set to 1 and the interrupt is not masked (xxMK = 0), the system does not enter the HALT mode. When macro service processing (xxISM = 1) is performed, the system enters the HALT mode after the macro service terminates.

Table 16-1. Operation States in the HALT Mode

Function		Operation state
Clock generator		Operating
Internal system clock		
CPU		Stopped
I/O line		Retained
On-chip peripheral hardware		Operating
Internal data		Internal data such as the CPU status, data, and contents of the internal RAM is retained as it was before setting the HALT mode.
When external devices are expanded	AD0 to AD7	High-impedance
	AD8 to AD15	
	ASTB	0
	$\overline{RD}$	1
	$\overline{LWR}$, $\overline{HWR}$	
	CLKOUT	Operating

(2) Releasing the HALT mode

The HALT mode can be released by a nonmaskable interrupt request, nonmasked maskable interrupt request, nonmasked macro service request, or $\overline{\text{RESET}}$ input.

(a) Releasing the HALT mode by an interrupt request

The HALT mode is released by a nonmaskable interrupt request, a maskable interrupt request which is not masked, or macro service request, regardless of the priority of the request. If the HALT mode is set by the interrupt service routine, however, the following operation takes place (see **Table 16-2**):

(i) When an interrupt request, having higher priority than the interrupt request being handled, is issued, the HALT mode is released, and this new interrupt request is accepted. Such interrupt requests include a non-maskable interrupt request. When the PRSL bit of an interrupt mode control register (IMC) is set to 0, however, the system only accepts multiple interrupt requests having the lowest priority.

(ii) When an interrupt request, having the same priority as or lower priority than the interrupt request being handled, is issued, the HALT mode is only released. In this case, the new interrupt request is not accepted, and the next instruction is executed. The interrupt request is retained.

Table 16-2. Acceptance of Interrupts Generated during Interrupt Handling

Priority of interrupt service routines in the HALT mode	Priority and state of generated interrupts	
	Accepted	Retained
0	–	0 to 3
1	0	1 to 3
2	0, 1	2, 3
3	0 to 2, 3 (PRSL = 0)	3 (PRSL = 1)

Table 16-3. Operations after the HALT Mode is Released by an Interrupt Request

Releasing request	EI state	DI state
Nonmaskable interrupt	Branches to the vector address.	
Maskable interrupt	Branches to the vector address or executes the next instruction.	Executes the next instruction.

Table 16-4. Releasing the HALT Mode by a Macro Service Request

Conditions for a vectored interrupt at the end of macro service	HALT mode	Operations after releasing the HALT mode	
		EI state	DI state
Satisfied	Released	Branches to the vector address or executes the next instruction[Note]	Executes the next instruction.
Not satisfied	The system returns to the HALT mode.	—	

Note If the HALT mode is set by an interrupt service routine then an interrupt request which is not masked and has higher priority is generated, processing branches to the vector address.

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

Same as the normal reset operation except that data in the internal RAM before setting the HALT mode is retained.

16.3.2 STOP mode

(1) Setting the STOP mode and operation states in the STOP mode

In the STOP mode, the oscillator is stopped.

Power consumption can be substantially reduced when the entire application system stops. The system is placed in the STOP state by setting the STP bit in the STBC register to 1.

In the STOP mode, generation of the internal clock is stopped when the oscillator stops operating. At the same time, the watchdog timer for reserving the oscillation stabilization time is automatically cleared by the hardware.

Although program execution is stopped, the contents of all registers and internal RAM immediately before the STOP mode is set are retained. Hardware functions enter the states shown in Table 16-5.

Table 16-5. Operation States in the STOP Mode

Function		Operation state
Clock generator		Stopped
Internal system clock		
CPU		
I/O line		Retained when V_{DD} is within the operable range
On-chip peripheral hardware		Stopped
Internal data		Internal data such as the CPU status, data, and contents of the internal RAM is retained as it was before setting the STOP mode when V_{DD} is within the operating range[Note].
When external devices are expanded	AD0 to AD7	High-impedance
	AD8 to AD15	
	ASTB	0
	$\overline{RD}$	1
	$\overline{LWR}$, $\overline{HWR}$	
	CLKOUT	0

Note Only the contents of internal RAM are retained by keeping the data retention voltage when V_{DD} is lower than the minimum operable voltage.

(2) Releasing the STOP mode

Operations after the STOP mode is released depend on the releasing conditions and the status of the system when it entered the STOP mode.

The STOP mode can be released by NMI or $\overline{\text{RESET}}$ input.

(a) Releasing the STOP mode by NMI input

The oscillator restarts when a valid edge is applied to the NMI input. While the NMI input is active, the watchdog timer remains cleared and does not start counting. When the NMI input level returns to the inactive level, the watchdog timer starts counting. When the watchdog timer overflows, generation of the internal system clock is started.

Therefore, the μPD78356 is in the wait state during the following period:

(Active level duration after detecting a valid edge in NMI input)
+
(Watchdog timer overflow time)

The overflow time for the watchdog timer is specified using the WDM register.

Operations after the STOP mode is released depend on the status of the system when it entered the STOP mode, and the priority level of the watchdog timer interrupt request and NMI interrupt request.

Priority is specified for a watchdog timer interrupt request or interrupt request using the WDM register (see **Fig. 16-4**).

Fig. 16-4. Format of the Watchdog Timer Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
WDM	RUN	0	0	PRC	0	WDI2	WDI1	0	FFC2H	00H	R/W

WDI2	WDI1	Count clock	Overflow time [ms]	
			f_{CLK} = 12.5 MHz	f_{CLK} = 16.0 MHz
0	0	$f_{CLK}/2^9$	10.5	8.2
0	1	$f_{CLK}/2^{11}$	41.9	32.8
1	0	$f_{CLK}/2^{13}$	167.8	131.1
1	1	Not to be set		

Remark f_{CLK} is an internal system clock frequency.

PRC	Priority of watchdog timer interrupt request		
0	Watchdog timer interrupt request	<	NMI pin input interrupt request
1	Watchdog timer interrupt request	>	NMI pin input interrupt request

RUN	Watchdog timer operation specification
0	Stop watchdog timer
1	Clear watchdog timer, and start counting

Cautions 1. Data can be written into the WDM register only by a dedicated instruction (MOV WDM, #byte).

2. Do not change the priority for interrupt requests dynamically, that is, when a program is being executed.

3. The RUN bit cannot be reset to 0 by software.

4. The count clock is not reset even when the watchdog timer is cleared by setting the RUN bit to 1.

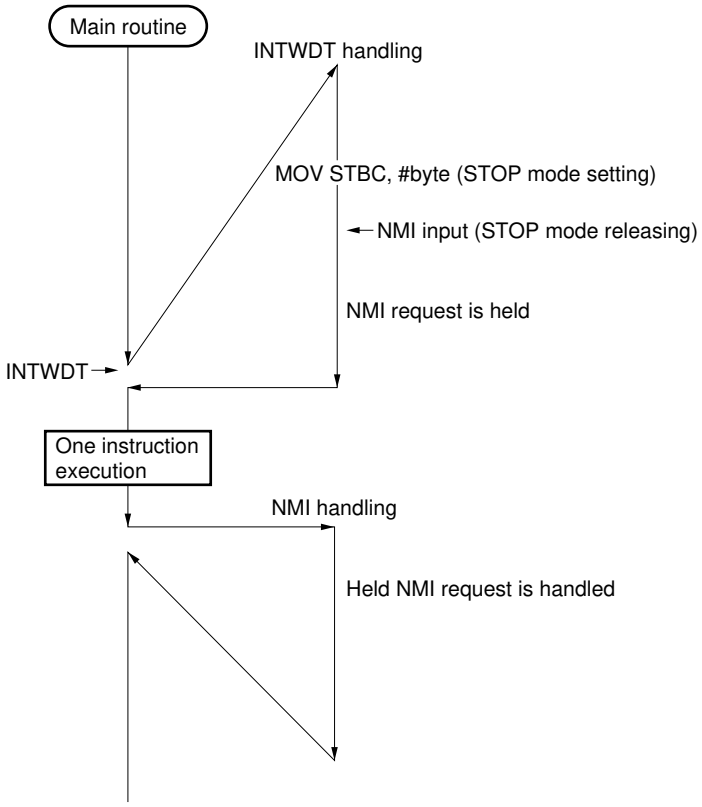
(i) When the STOP mode is set during the watchdog timer interrupt handling routine

<1> When the priority level is set as watchdog timer interrupt request > NMI interrupt request (WDM register PRC bit = 1)

After the STOP mode is released by NMI input, operations are started from the instruction after the instruction setting the STOP mode (The NMI interrupt request is held.)

When the RETI instruction is executed in the watchdog timer interrupt handling routine, operations restore from the watchdog timer interrupt handling routine. After this, when one instruction is executed, operations branch to the NMI interrupt handling routine.

Fig. 16-5. Operations after STOP Mode is Released (1)



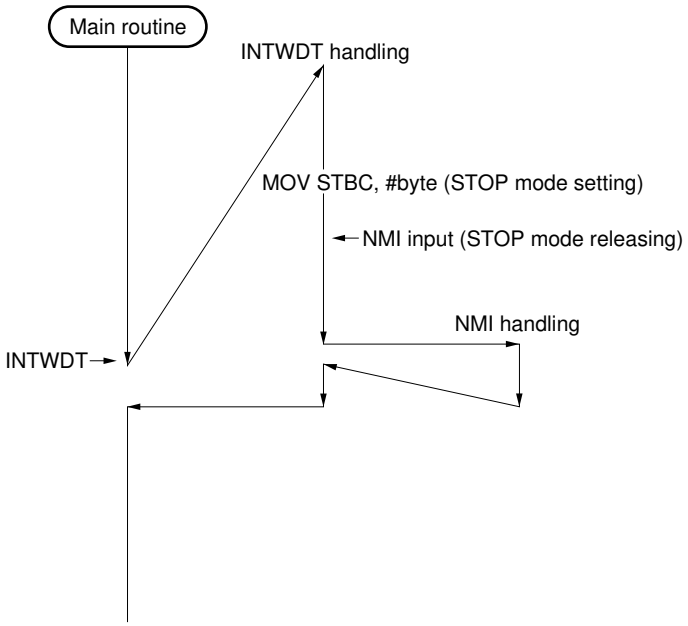
<2> When the priority level is set as watchdog timer interrupt request < NMI interrupt request (WDM register PRC bit = 0)

After the STOP mode is released by NMI input, operations branch into the NMI interrupt handling routine immediately.

When the RETI instruction is executed in the NMI interrupt handling routine, operations restore from the instruction after the instruction setting the STOP mode during the watchdog timer interrupt handling routine.

After this, when the RETI instruction is executed, operations restore from the watchdog timer interrupt handling routine.

Fig. 16-6. Operations after STOP Mode is Released (2)



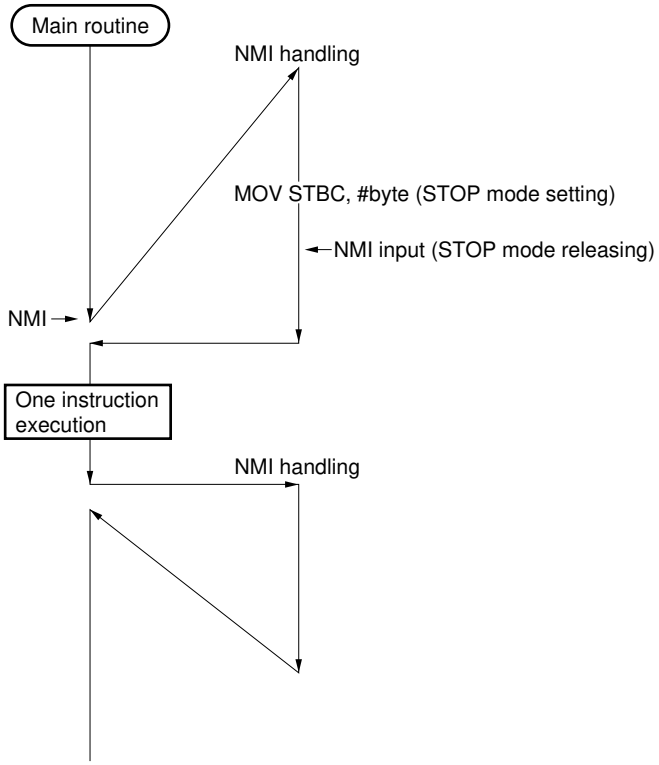
(ii) When the STOP mode is set during the NMI interrupt handling routine

After the STOP mode is released by NMI input, operations start from the instruction after the instruction setting the STOP mode.

When the RETI instruction is executed in the NMI interrupt handling routine, operations restore from the NMI interrupt handling routine.

After this, when one instruction is executed, operations branch to the NMI interrupt handling routine again.

Fig. 16-7. Operations after STOP Mode is Released (3)



(iii) When the STOP mode is set in other than nonmaskable interrupt handling routine

After the STOP mode is released by NMI input, operations branch into the NMI interrupt handling routine immediately.

When the RETI instruction is executed in the NMI interrupt handling routine, operations restore from the instruction after the instruction setting the STOP mode.

Fig. 16-8. Operations after STOP Mode is Released (4)

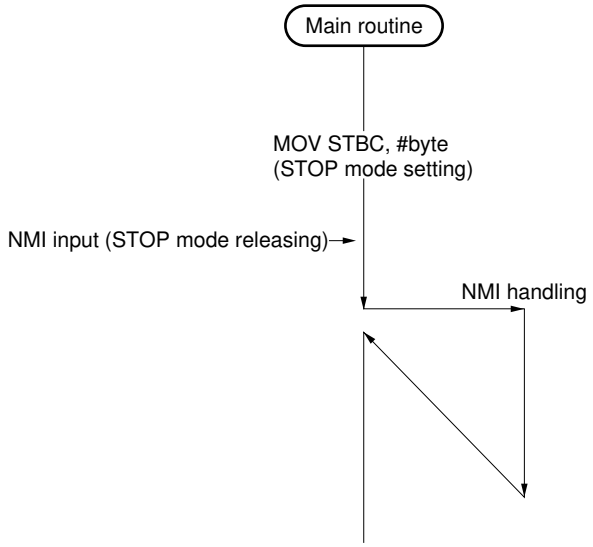


Table 16-6. Release of STOP Mode and Operations after Release

Release source	STOP mode set status	PRC[Note]	Operations after STOP Mode is Released
$\overline{\text{RESET}}$ input	–	x	Operations are started from the reset address.
NMI input	INTWDT routine	1	Operations start from the instruction after the MOV STBC #byte instruction (The NMI interrupt request is held). The NMI interrupt request is accepted after the watchdog timer interrupt handling setting the STOP mode completes (See **Fig. 16-5**).
		0	The NMI interrupt request is accepted (See **Fig. 16-6**).
	NMI routine	x	Operations start from the instruction after the MOV STBC #byte instruction (The NMI interrupt request is held). The NMI interrupt request is accepted again after the NMI interrupt handling setting the STOP mode completes (See **Fig. 16-7**).
	Other than nonmaskable interrupt routine	x	The NMI interrupt request is accepted (See **Fig. 16-8**).

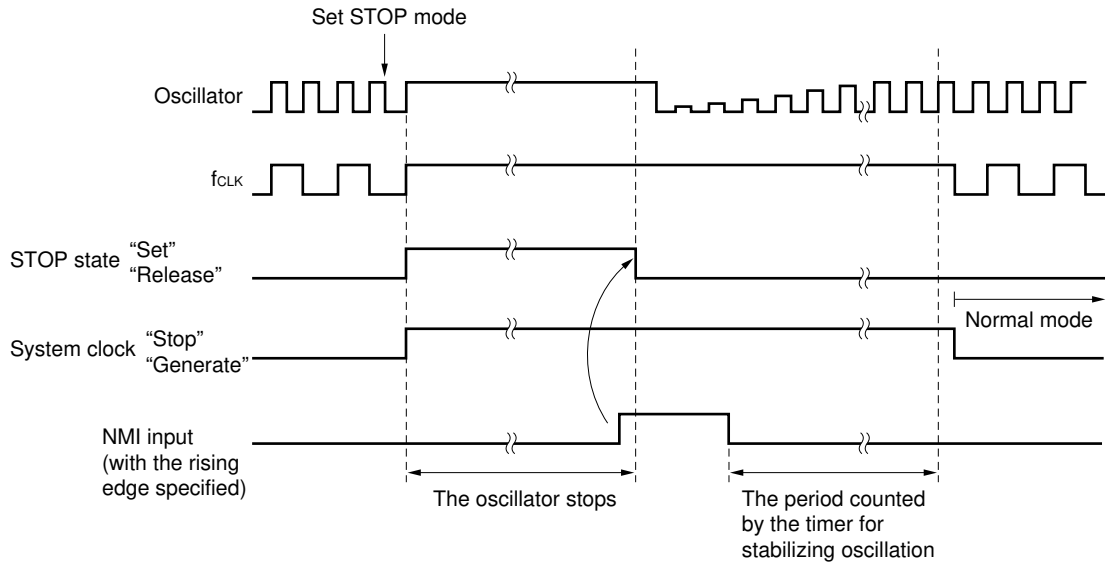
Note A priority level specification flag in the watchdog timer mode register (WDM).

PRC = 1 ... Watchdog timer interrupt request > NMI interrupt request

PRC = 0 ... Watchdog timer interrupt request < NMI interrupt request

PRC = x ... don't care

Fig. 16-9. Releasing the STOP Mode by NMI Input



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

The oscillator restarts at the same time $\overline{\text{RESET}}$ input level is changed from high to low to put the system in the reset state.

Apply the $\overline{\text{RESET}}$ active period for the oscillation stabilization time. When $\overline{\text{RESET}}$ starts up, operations start from the addresses stored in the reset vector.

[MEMO]

CHAPTER 17 RESET FUNCTION

When the signal applied to the $\overline{\text{RESET}}$ pin is low, the system is reset, and each hardware component is placed in the status indicated in Table 17-1.

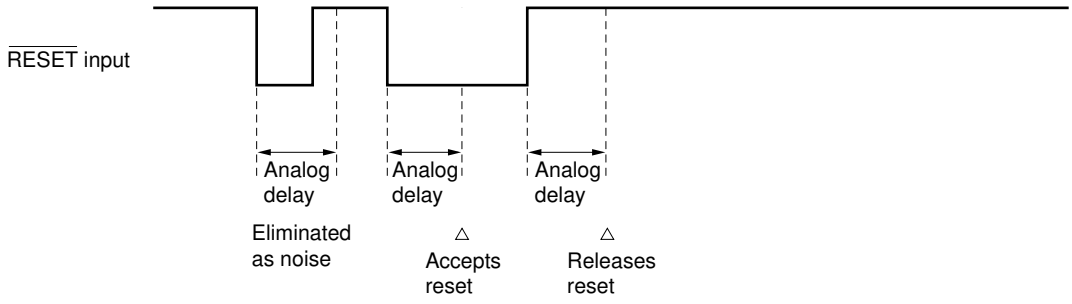
When the signal applied to the $\overline{\text{RESET}}$ input port goes high, the reset status is released, and program execution starts. The contents of registers must be initialized in the program as required. In particular, the number of cycles specified in the programmable wait control register (PWC) must be changed as required. After resetting, the initial setting in the PWC register is effective: Three wait cycles are added to the bus cycle, and the fetch cycle mode is normal (see **Fig. 18-5**).

The $\overline{\text{RESET}}$ pin contains a noise eliminator based on analog delays to prevent abnormal operation due to noise.

Cautions 1. When $\overline{\text{RESET}}$ is active, all pins except $\overline{\text{WDTO}}$, CLKOUT, AV_{REF}, AV_{DD}, AV_{SS}, V_{DD}, V_{SS}, X1, and X2 go into the high-impedance state.

2. When RAM is expanded externally, attach a pull-up resistor to the P90/$\overline{\text{RD}}$ pin, P91/$\overline{\text{LWR}}$ pin, and P92/$\overline{\text{HWR}}$ pin. Otherwise, these pins may go into the high-impedance state, and the contents of the external RAM may be lost. Or signal collision on the address/data bus may damage the input/output circuit.

Fig. 17-1. Acceptance of the $\overline{\text{RESET}}$ Signal



In reset operation at power-on, a time for stabilized operation between power-on to reset acceptance is required as shown in Fig. 17-2.

Fig. 17-2. Reset Operation at Power-on

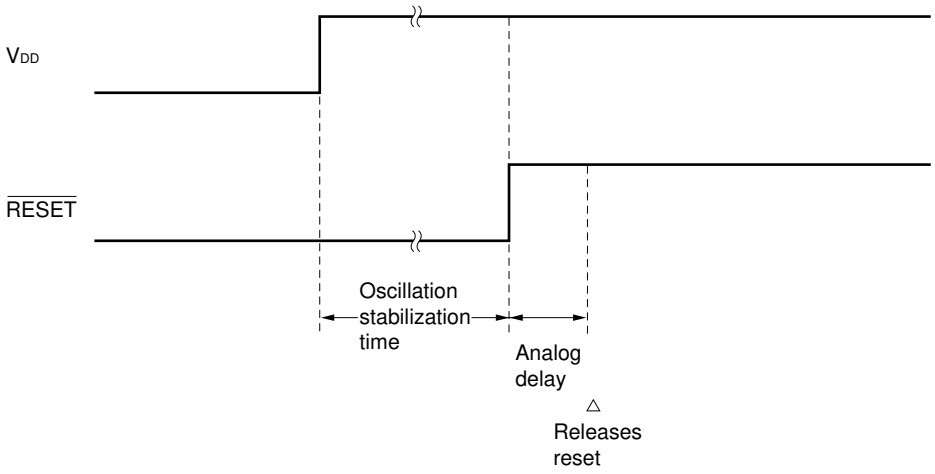


Table 17-1. Hardware Statuses after Reset (1/2)

Hardware			Status after reset
Control registers	Program counter (PC)		The contents of a reset vector table (0000H, 0001H) are set.
	Stack pointer (SP)		Undefined[Note]
	Program status word (PSW)		0000H
	CPU control word (CCW)		00H
Internal RAM	Data memory		Undefined[Note]
	General registers (R0 to R15)		
Ports	Output latches (P0 to P5, P8 to P10)		Undefined
	Mode registers	(PM0 to PM3, PM5, PM8, PM10)	FFH
		(PM9)	0FH
	Mode control registers	(PMC0, PMC3, PMC8, PMC10)	00H
		(PMC2)	01H
	Pull-up registor option registers (PUOL, PUOH)		00H
	Port read control register (PRDC)		
Real-time output ports (RTP)	Real-time output prot registers (RTPL, RTPH)		Undefined
	Real-time output port mode register (RTPM)		00H
Real-time pulse unit (RPU)	Timers (TM0 to TM4)		0000H
	Up/down counter (UDC)		
	Timer unit mode registers (TUM0 to TUM3)		00H
	Timer control registers	(TMC0, TMC1)	00H
		(TMC2)	04H
	Up/down counter control register (UDCC)		00H
	Timer output control registers (TOC0 to TOC2)		
	Timer overflow status register (TOVS)		
	Noise protection control register (NPC)		
	Compare registers (CM00 to CM03, CM10, CM11, CM20, CM21, CM40, CMUD0, CMUD1)		Undefined
	Capture/compare registers (CC00 to CC02, CC30, CC31)		
A/D converter	A/D converter mode registers	(ADM0)	00H
		(ADM1)	07H
	A/D conversion result registers (ADCR0 to ADCR7, ADCR0H to ADCR7H)		Undefined
D/A converter	D/A converted data coefficient registers	(DACS0, DACS1)	00H
		(DACS)	0000H

Note When the STOP mode is released by $\overline{\text{RESET}}$ input, the values before the STOP mode was set are restored.

Table 17-1. Hardware Statuses after Reset (2/2)

Hardware			Status after reset
Serial interface	Asynchronous serial interface mode register (ASIM)		80H
	Asynchronous serial interface status register (ASIS)		00H
	Clock synchronous serial interface mode registers (CSIM0, CSIM1)		
	Serial bus interface control register (SBIC)		
	Serial I/O shift registers (SIO0, SIO1)		Undefined
	Serial reception buffer (RXB)		
	Serial transmission shift register (TXS)		
PWM output function	PWM control register (PWMC)		00H
	PWM buffer registers (PWM0, PWM0L, PWM1, PWM1L)		Undefined
Watchdog timer	Watchdog timer mode register (WDM)		00H
Interrupt function	External interrupt mode registers (INTM0, INTM1)		00H
	Interrupt mode control register (IMC)		80H
	Interrupt mask flag registers	MK0L, MK0H, MK1L	FFH
		MK0	FFFFH
		MK1	00FFH
	Interrupt control registers (OVIC0, OVIC3, PIC0 to PIC4, CMIC00 to CMIC03, CMIC10, CMIC11, CMIC20, CMIC21, CMIC40, CMICUD0, CMICUD1, SERIC, SRIC, STIC, CSIIC0, CSIIC1, ADIC)		43H
	In-service priority register (ISPR)		00H
External expansion function	Memory expansion mode register (MM)		00H
	Programmable wait control register (PWC)		C0AAH[Note]
CPU control	Standby control register (STBC)		0000x000B

Note Becomes CFAAH only in the external 16-bit bus ROM-less mode (MODE0, 1 = HH).

CHAPTER 18 BUS INTERFACE FUNCTION

The μPD78356 has bus interface functions to control external memory (ROM, RAM) and I/O devices.

18.1 External Device Expansion Function for the μPD78356

For the μPD78356, external devices (data memory, program memory, and peripheral devices) can be assigned to the external memory area (C000H to F6FFH).

An external 8- or 16-bit data bus can be specified by setting the programmable wait control register (PWC).

(1) When an external 8-bit bus is specified

When an external device is connected, port 4 (P40 to P47) is used as the multiplexed address/data bus (AD0 to AD7) and port 5 (P50 to P57) is used as the address bus (AD8 to AD15) by setting the memory expansion mode register (MM). The $\overline{RD}$, $\overline{LWR}$, and ASTB signals are used to access the external device.

Table 18-1 lists the pins used for access to an external device and indicate how to assign functions to the pins.

Table 18-1. Assigning Functions to Pins (for μPD78356 When an External 8-Bit Bus is Specified)

Memory expansion mode register	Pin funciton			
MM0 to MM2	P40 to P47	P50 to P57	P90	P91
Port mode	General port			
Expansion mode	AD0 to AD7	Set to AD8 to AD15 according to the expanded memory size	$\overline{RD}$	$\overline{LWR}$

Remark AD8 to AD15 are used as the address bus.

The number of port 5 pins which operate as the address bus can be changed according to the size of memory expanded externally. This feature allows memory to be expanded gradually as shown in the following table. Pins not used as the address bus can be used as general I/O ports. (See **Table 18-2**.)

Table 18-2. Operation of Port 5 (Expansion Mode)

<table>
<tr><th>P50</th><th>P51</th><th>P52</th><th>P53</th><th>P54</th><th>P55</th><th>P56</th><th>P57</th><th>External address space</th></tr>
<tr><td colspan="8">General I/O port</td><td>Within 256 bytes</td></tr>
<tr><td rowspan="3">AD8</td><td rowspan="3">AD9</td><td rowspan="3">AD10</td><td rowspan="3">AD11</td><td colspan="4"></td><td>Within 4 Kbytes</td></tr>
<tr><td rowspan="2">AD12</td><td rowspan="2">AD13</td><td colspan="2"></td><td>Within about 16 Kbytes</td></tr>
<tr><td>AD14</td><td>AD15</td><td>Full expansion mode</td></tr>
</table>

When an external device reference instruction is executed in the 256-, 4K-, or 16K-byte expansion mode, the system operates as follows. Fig. 18-1 shows the memory map for expansion mode.

(a) 256-byte expansion mode

Masks the 8 high-order bits of the 16 external reference address bits and outputs 00H to FFH as address information from pins AD0 to AD7.

(b) 4K-byte expansion mode

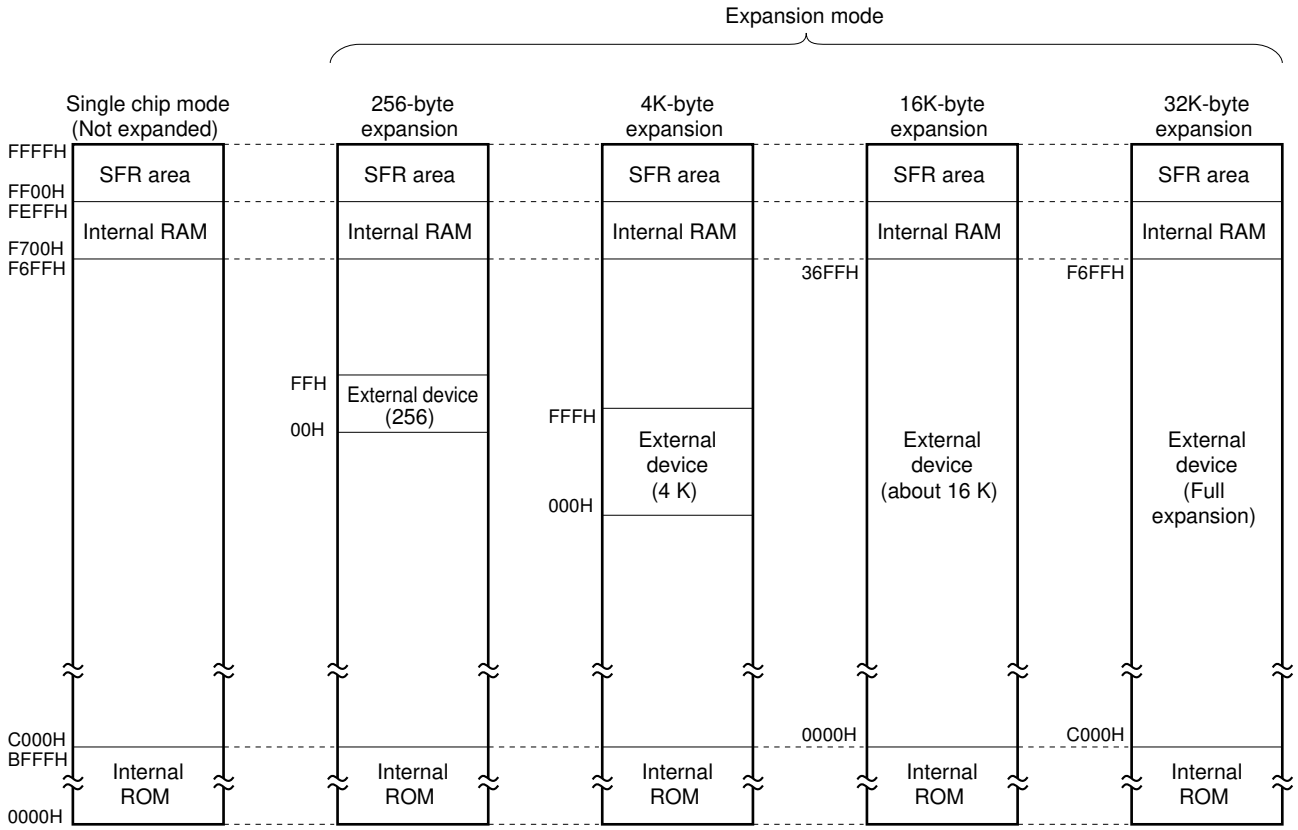
Masks the 4 high-order bits among 16 external reference address bits and outputs 000H to FFFH as address information from pins A8 to A11 and AD0 to AD7.

(c) 16K-byte expansion mode

Masks the 2 high-order bits of the 16 external reference address bits and output 0000H to 36FFH as address information from pins A8 to A13 and AD0 to AD7.

As described above, the 8, 4, and 2 high-order bits of the 16-bit address are masked in the 256-, 4K-, and 16K-byte expansion modes, respectively.

Fig. 18-1. Memory Map for Expansion Mode (for μPD78356 When an External 8-bit bus is Specified)



Caution The internal memory (ROM, RAM) capacity of the μPD78356 can be changed by setting the memory expansion mode register (MM). (See Fig. 18-4.)

(2) When an external 16-bit bus is specified

When an external device is connected, port 4 (P40 to P47) is used as the lower half (AD0 to AD7) of the multiplexed address/data bus and port 5 (P50 to P57) is used as the upper half (AD8 to AD15) of the multiplexed address/data bus by setting the memory expansion mode register (MM). The $\overline{RD}$, $\overline{LWR}$, $\overline{HWR}$, and ASTB signals are used to access the external device.

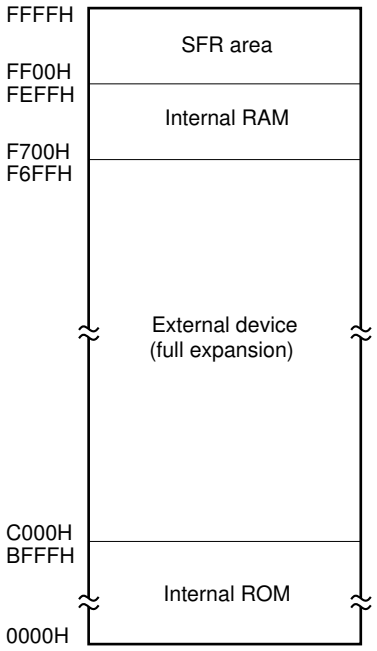
Table 18-3 lists the pins used for access to an external device and indicates how to assign functions to the pins.

Table 18-3. Assigning Functions to Pins (for μPD78356 When an External 16-Bit Bus is Specified)

Memory expansion mode register	Pin function				
MM0 to MM2	P40 to P47	P50 to P57	P90	P91	P92
Port mode	General port				
Expansion mode	AD0 to AD7	AD8 to AD15	$\overline{RD}$	$\overline{LWR}$	$\overline{HWR}$

Fig. 18-2. Memory Map for Expansion Mode (for μPD78356 When an External 16-Bit Bus is Specified)

Full expansion mode



Caution The internal memory (ROM, RAM) capacity of the μPD78356 can be changed by setting the memory expansion mode register (MM). (See Fig. 18-4.)

18.2 Access to External Devices by the μPD78355

The μPD78355 does not contain ROM. External devices (data memory, program memory, and peripheral devices) can be assigned to the area (0000H to F6FFH) other than that used for the internal RAM.

An external 8- or 16-bit data bus can be specified by setting the programmable wait control register (PWC).

(1) When an external 8-bit bus is specified

When an external device is connected, AD0 to AD7 are used as the multiplexed address/data bus, and AD8 to AD15 are used as the address bus. To access the external device, $\overline{RD}$, $\overline{LWR}$, and ASTB signals are used.

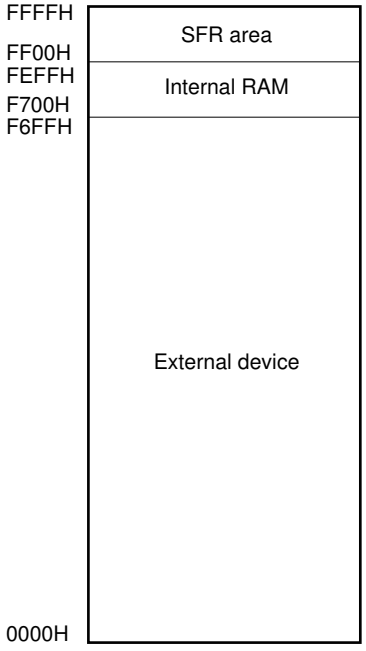
(2) When an external 16-bit bus is specified

When an external device is connected, AD0 to AD7 are used as the lower half of the multiplexed address/data bus, and AD8 to AD15 are used as the upper half of the multiplexed address/data bus. The $\overline{RD}$, $\overline{LWR}$, $\overline{HWR}$, and ASTB signals are used to access the external device.

In the μPD78355, specification in the MM0 to MM2 bits of the memory expansion mode register (MM) is invalidated.

Unlike the μPD78356, the memory of μPD78355 cannot be expanded gradually.

Fig. 18-3. Memory Map of the μPD78355



18.3 Control Registers

18.3.1 Memory expansion mode register

The memory expansion mode register (MM) is an 8-bit register which controls the address/data bus and signals such as $\overline{RD}$, $\overline{LWR}$, and $\overline{HWR}$ when memory or I/O is externally expanded.

Bits MM0 to MM2 are valid only when MODE0 and MODE1 are LL. These bits are used to specify the functions of the pins of ports 4 and 5.

When the expansion mode is specified using bits MM0 to MM2, the P90 and P91 pins function as the $\overline{RD}$ and $\overline{LWR}$ pins automatically. When the 16-bit bus is specified for at least one memory space by the programmable wait control register (PWC), the P92 pin functions as the $\overline{HWR}$ pin. (The P92 pin functions as the $\overline{HWR}$ pin even when the 16-bit bus is specified for the internal memory space.)

In the expansion mode, when an external 16-bit bus is used for a memory configuration consisting of internal ROM and external memory, only the full expansion mode can be specified.

The MM register is set to 00H by $\overline{RESET}$ input.

Fig. 18-4 shows the format of the MM register.

Fig. 18-4. Format of the Memory Expansion Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	When reset	R/W
MM	0	MM6	MM5	0	MM3	MM2	MM1	MM0	FFC4H	00H	R/W

MM2	MM1	MM0	Operation modes of the P40 to P47 and P50 to P57 pins		
0	0	0	Port mode	Single chip mode	P40 to 47: Input port P50 to 57: Port mode
0	0	1			P40 to 47: Output port P50 to 57: Port mode
0	1	0	Not to be set		
0	1	1	Expansion mode	256 bytes	P40 to 47: Expansion mode P50 to 57: Port mode
1	0	0		4 Kbytes	P40 to 47, P50 to P53: Expansion mode P54 to 57: Port mode
1	0	1		16 Kbytes	P40 to 47, P50 to P55: Expansion mode P56 to 57: Port mode
1	1	0	Not to be set		
1	1	1	Expansion mode	Full Expansion mode	P40 to 47, P50 to P57: Expansion mode

MM3	Internal RAM capacity
0	2 Kbytes
1	1 Kbyte

MM6	MM5	Internal ROM capacity
0	0	48 Kbytes
0	1	32 Kbytes
1	0	24 Kbytes
1	1	16 Kbytes

Cautions 1. Be sure to write 0 in bits 4 and 7. If 1 is written in these bits, operation will not be normal.

2. Invalid combinations are specified as "Not to be set" in Fig. 18-4. Never write these combinations.

18.3.2 Programmable wait control register

The programmable wait control register (PWC) is a 16-bit register for programmable wait control to the bus cycle (internal[Note] and external memory access) generated by the μPD78356. This register also specifies the bus width when an external device is connected. This register enables low-speed memory and peripheral devices to be connected externally.

The PWC register cannot be accessed in 8-bit mode. Use a 16-bit data transfer instruction to access the PWC register.

The PWC register value after $\overline{\text{RESET}}$ input varies according to the operation mode specified with the MODE0 and MODE1 pins. (See **Table 18-4**.)

Fig. 18-5 shows the format of the PWC register.

Note The internal memory refers to the internal ROM area (only for the μPD78356 and μPD78P356) and internal RAM area (F700H to FEFFH).

Table 18-4. Status after PWC Register Reset

MODE0	MODE1	μPD78355	μPD78356	μPD78P356	PWC register value after reset
L	L	Not to be set	Single chip mode		C0AAH
L	H	Not to be set			–
H	L	Romless mode (External 8-bit bus)		PROM mode	C0AAH
H	H	Romless mode (External 16-bit bus)			CFAAH

Fig. 18-5. Format of the Programmable Wait Control Register

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	R/W
PWC	AW1	AW0	0	0	BW3	BW2	BW1	BW0	PWC7	PWC6	PWC5	PWC4	PWC3	PWC2	PWC1	PWC0	FFC6H	R/W

PWC1	PWC0	Number of clocks specified for the memory space (0000H to 3FFFH)			
		Wait cycle	Data access cycle	Fetch cycle mode	Fetch cycle
0	0	0	3	Normal fetch	3
0	1	1	4		4
1	0	2	5		5
1	1	0	3	High-speed fetch	2

PWC3	PWC2	Number of clocks specified for the memory space (4000H to 7FFFH)			
		Wait cycle	Data access cycle	Fetch cycle mode	Fetch cycle
0	0	0	3	Normal fetch	3
0	1	1	4		4
1	0	2	5		5
1	1	0	3	High-speed fetch	2

PWC5	PWC4	Number of clocks specified for the memory space (8000H to BFFFH)			
		Wait cycle	Data access cycle	Fetch cycle mode	Fetch cycle
0	0	0	3	Normal fetch	3
0	1	1	4		4
1	0	2	5		5
1	1	0	3	High-speed fetch	2

PWC7	PWC6	Number of clocks specified for the memory space (C000H to F6FFH, FFD0H to FFDFH)			
		Wait cycle	Data access cycle	Fetch cycle mode	Fetch cycle
0	0	0	3	Normal fetch	3
0	1	1	4		4
1	0	2	5		5
1	1	0	3	High-speed fetch	2

BW0	Bus width (valid address: 0000H to 3FFFH)
0	8 bits
1	16 bits

BW1	Bus width (valid address: 4000H to 7FFFH)
0	8 bits
1	16 bits

BW2	Bus width (valid address: 8000H to BFFFH)
0	8 bits
1	16 bits

BW3	Bus width (valid address: C000H to F6FFH, FFD0H to FFDFH)
0	8 bits
1	16 bits

AW0	Address wait control (valid address: 0000H to 7FFFH)
0	Address wait is not added
1	Address wait is added

AW1	Address wait control (valid address: 8000H to F6FFH, FFD0H to FFDFH)
0	Address wait is not added
1	Address wait is added

Cautions 1. Cycle counts indicated in Fig. 18-5 do not contain any address wait. When an address wait is added, one cycle must be added to the indicated cycle count.

2. Although instruction fetch and data access are allowed for the peripheral RAM area (F700H to FDFFH), the bus width specification and wait specification made by the PWC register are invalid for that area, and 16-bit bus operation is always performed. Every instruction fetch is performed in the high-speed fetch mode.

3. Instruction fetches cannot be performed for the main RAM area (FE00H to FEFFH). Data is always accessed in 16-bit units; the bus width specification and wait specification made by the PWC register are invalid at the time of data access. (In this case, a special bus cycle made up of two clock cycles is activated.)

4. The wait specification for the external SFR area (FFD0H to FFDFH) is made by the PWC7 and PWC6 bits.

5. For the internal ROM area, 16-bit bus operation is performed regardless of the PWC register setting. The wait specification for this area can be made using the PWC register. Note that, however, specifying a 16-bit bus for the internal ROM area makes the P92 pin function as the $\overline{\text{HWR}}$ pin even when only an 8-bit external memory is connected. The port function is then disabled. So, never specify the 16-bit bus for the internal ROM area.

18.4 Bus Timing

Fig. 18-6 to 18-11 show the timing charts for a read or write to the μPD78356 memory and external I/O.

When a 16-bit bus is set for a memory area by the MODE0 and MODE1 pins, and the programmable wait control register (PWC), an access to an odd address of that area is performed in two bus cycles. In other cases, an access to memory or I/O is performed in one bus cycle. One bus cycle is made up of three clocks. When wait cycles are inserted according to the PWC register setting, the inserted wait cycles are added to the bus cycle. One clock period is 62.5 ns when an external 32-MHz clock is used.

In the μPD78356, an instruction code is fetched on the same timing as data read.

Fig. 18-6. Read Operation (with an 8-Bit Bus)

Condition

- Bus size : 8 bits
- Bus cycle : No wait

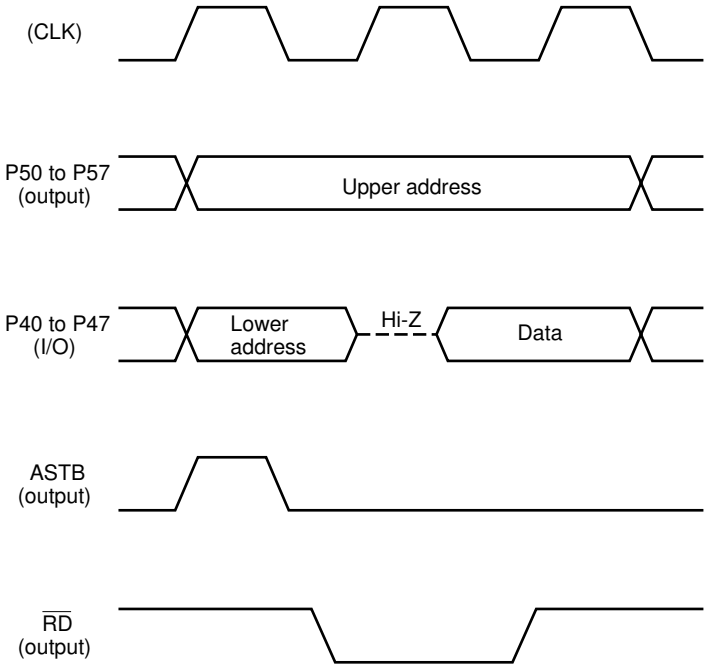


Fig. 18-7. Write Operation (with an 8-Bit Bus)

Condition

- Bus size : 8 bits
- Bus cycle : No wait

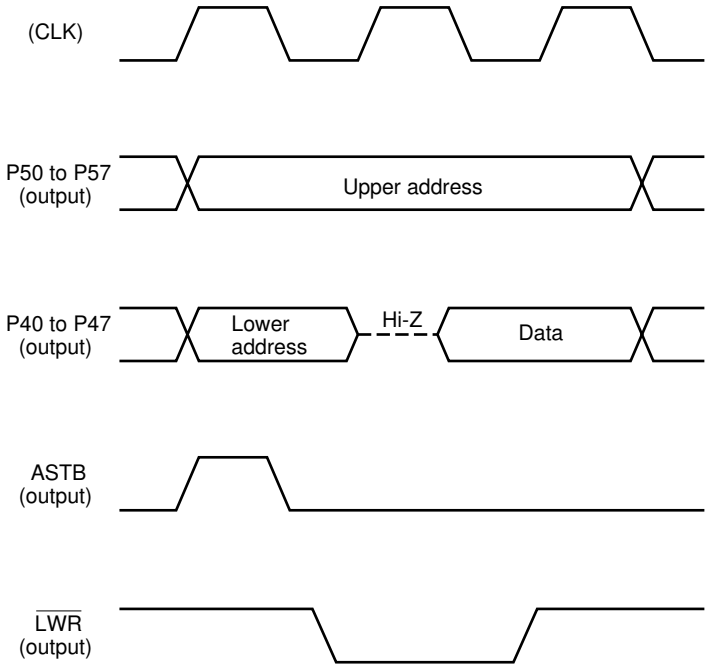


Fig. 18-8. Read Operation (When an Even Address is Accessed with a 16-Bit Bus)

Condition

- Bus size : 16 bits
- Bus cycle : No wait
- Lower 8-bit data : Even address
- Higher 8-bit data : Odd address

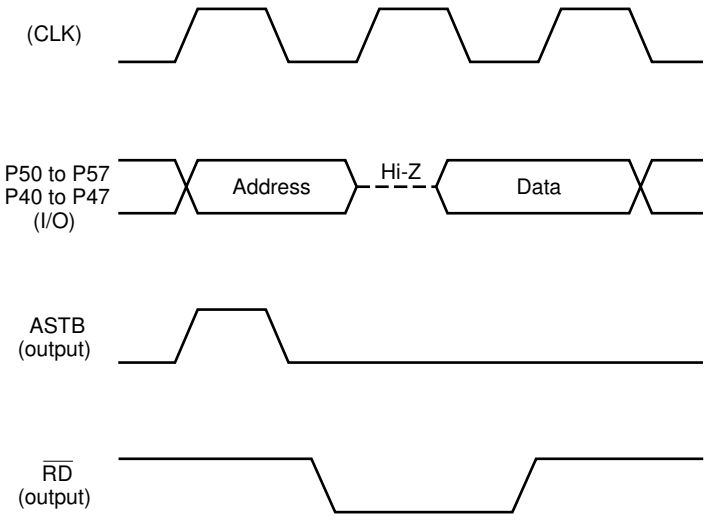
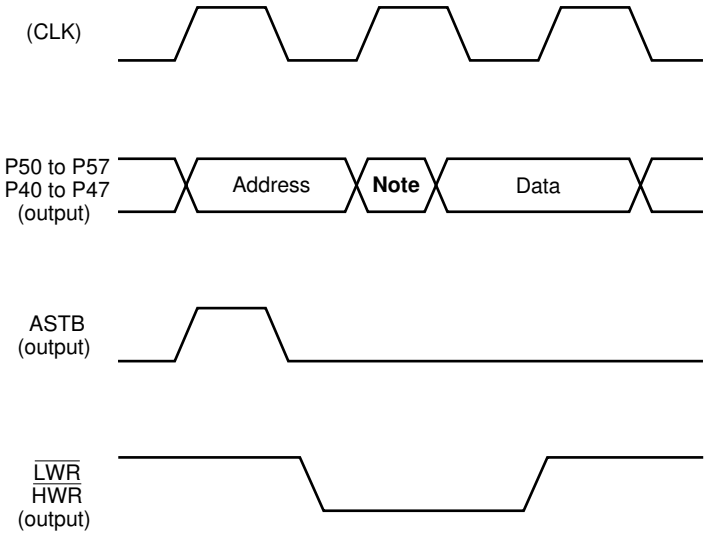


Fig. 18-9. Write Operation (When an Even Address is Accessed with a 16-Bit Bus)

Condition

- Bus size : 16 bits
- Bus cycle : No wait
- Lower 8-bit data : Even address
- Higher 8-bit data : Odd address

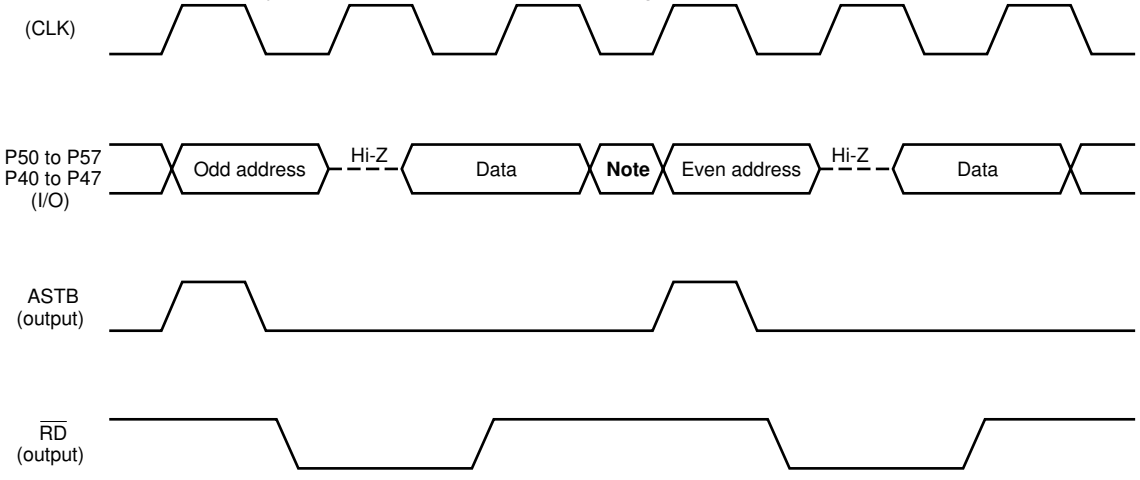


Note Undefined

Fig. 18-10. Read Operation (When an Odd Address is Accessed with a 16-Bit Bus)

Condition

- Bus size : 16 bits
- Bus cycle : No wait
- Lower 8-bit data : Odd address
- Higher 8-bit data : Even address

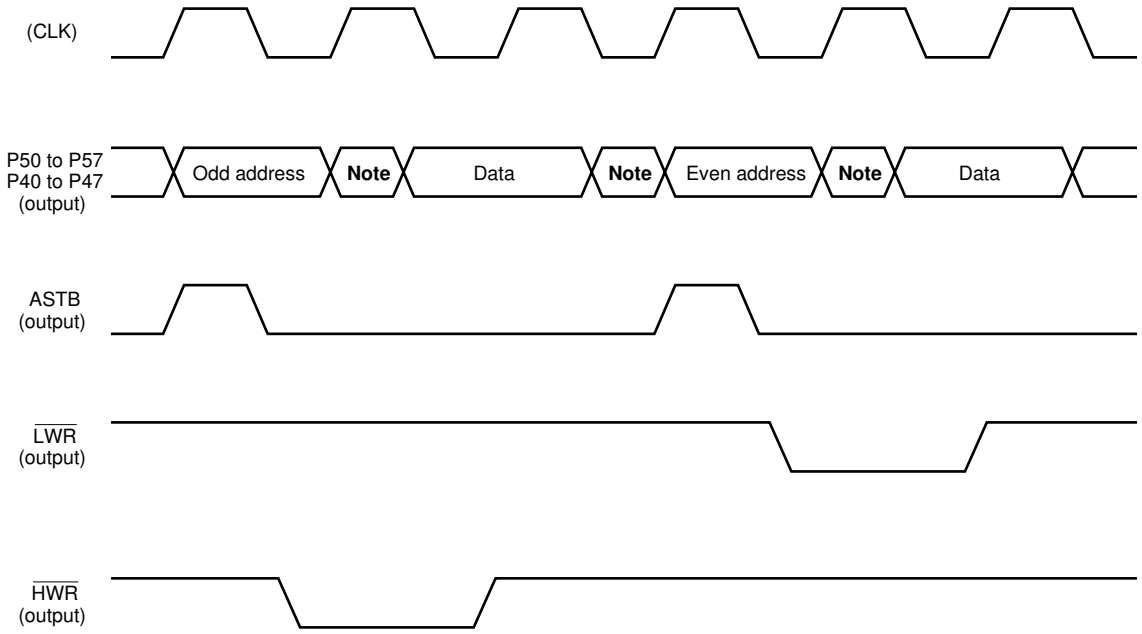


Note Undefined

Fig. 18-11. Write Operation (When an Odd Address is Accessed with a 16-Bit Bus)

Condition

- Bus size : 16 bits
- Bus cycle : No wait
- Lower 8-bit data : Odd address
- Higher 8-bit data : Even address

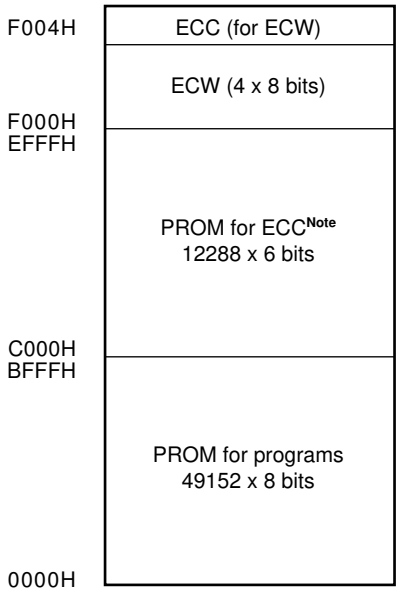


Note Undefined

CHAPTER 19 PROGRAMMING FOR THE μPD78P356

The μPD78P356 contains electrically writable PROM (49152 x 8 bits for programs and 12288 x 6 bits for ECC). Fig. 19-1 shows a memory map for the programming mode.

Fig. 19-1. Memory Map (in the μPD78P356 Programming Mode)



Note For the PROM for ECC, the low-order 6 bits are valid.

An error correction code (ECC) can be written to the space from C000H to EFFFH to improve the reliability of an electrically written program (0000H to BFFFH).

The ECC control word (ECW) is a 4-byte register to control the addition of an ECC circuit. When the ECC circuit is used, the least significant bit of the lowest byte (F000.0) of ECW must be reset. ECC and ECW are generated by ECCGEN, which is supplied with the RA78K3 assembler package. (ECC is generated in the least significant 6 bits, and the most significant 2 bits are fixed to 1.

Use the MODE0/V_{PP}, MODE1, P21, and $\overline{\text{RESET}}$ pins to set the μPD78P356 to the PROM programming mode when programming on the PROM.

The μPD78P356 provides programming characteristics compatibility with the μPD27C1001A.

Table 19-1. Functions of Pins in the Programming Mode

Function	Normal operating mode	Programming mode
Address input	P00 to P07, P50, P20, P51 to P57	A0 to A16
Data input	P40 to P47	D0 to D7
Program pulse	P12	$\overline{PGM}$
Chip enable	P10	$\overline{CE}$
Output enable	P11	$\overline{OE}$
Program voltage	MODE0/V_{PP}	
Mode voltage	MODE1, P21, $\overline{RESET}$	

19.1 Operating Mode

When the MODE0/V_{PP} pin is set to H, the MODE1 pin is set to L, the P21 pin is set to L, and the $\overline{RESET}$ pin is set to L, the μPD78P356 enters the programming write/verify mode. This mode varies to each operating mode shown in Table 19-2 according to how to set the $\overline{CE}$, $\overline{OE}$, and $\overline{PGM}$ pins.

Setting the μPD78P356 to the read mode enables it to read the contents of PROM.

Connect unused pins as shown in **1.4 Pin Configuration (Top View)**.

Table 19-2. Operating Modes for Programming on PROM

Mode	MODE1	P21	$\overline{RESET}$	$\overline{CE}$	$\overline{OE}$	$\overline{PGM}$	V_{PP}/ MODE0	V_{DD}	D0 to D7
Page data latch	L	L	L	H	L	H	+12.5 V	+6.5 V	Data input
Page program				H	H	L			High-impedance
Byte program				L	H	L			Data input
Program verify				L	L	H			Data output
Program inhibit				x x	L H	L H			High-impedance
Read				L	L	H	+5 V	+5 V	Data output
Output disable				L	H	x			High-impedance
Standby				H	x	x			High-impedance

Remark L : Directly connected to the V_{SS} pin.
H : Directly connected to the V_{DD} pin.
x : Indicates L or H.

19.2 Procedure for Writing on PROM (Page Program Mode)

The following is a procedure for writing on PROM. (See **Fig. 19-2**.)

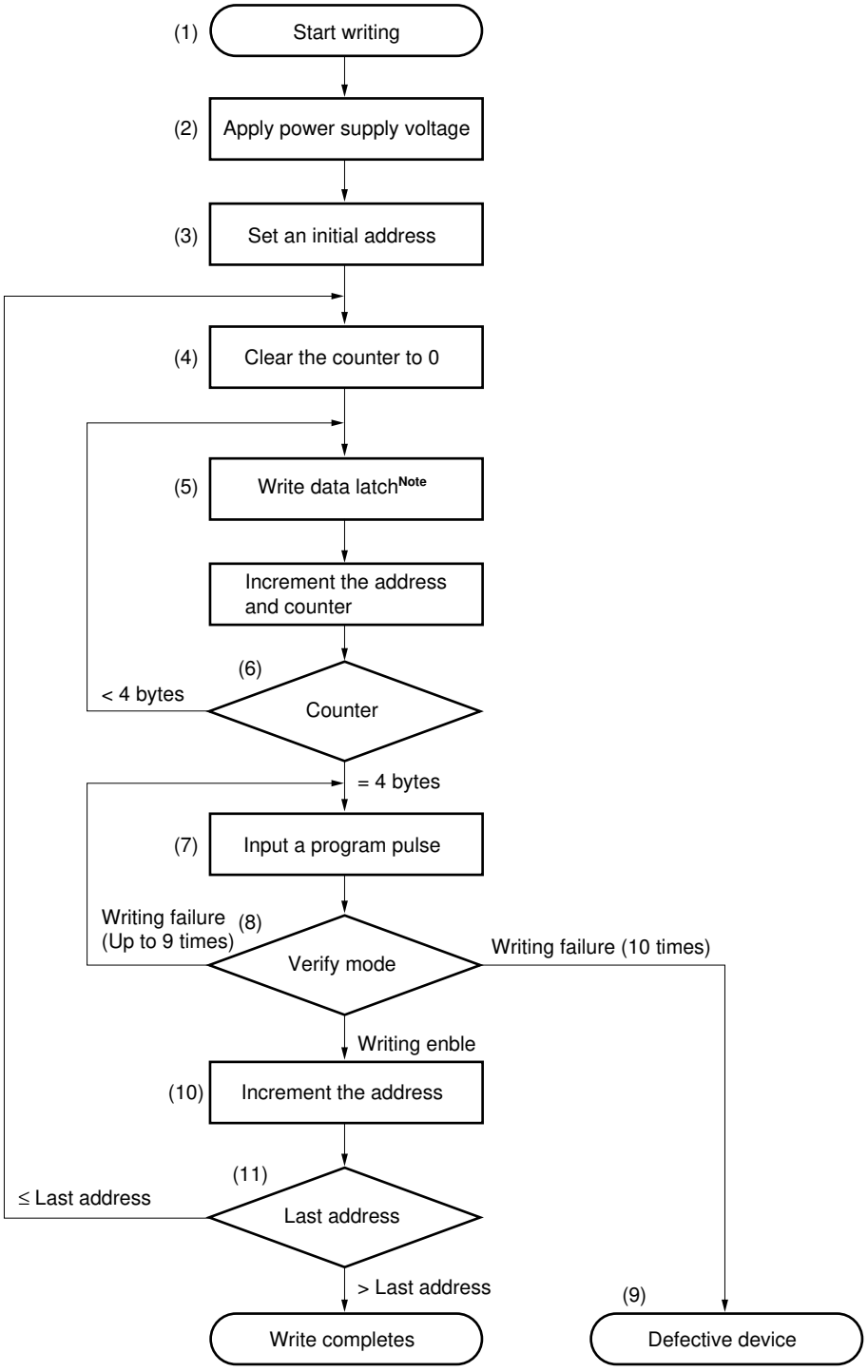
In the page program mode, data is written in units of pages (four bytes). When write data completes midway of a page, latch FFH after the data so that the data fits into pages.

(1) Always set each pin as follows: MODE0/V_{PP} = H, MODE1 = L, P21 = L, and $\overline{\text{RESET}}$ = L. Connect unused pins as shown in **1.4 Pin Configuration (Top View)**.
(2) Apply +6.5 V to the V_{DD} pin and +12.5 V to the MODE0/V_{PP} pin.
(3) Input an initial address to the A0 to A16 pins.
(4) Clear the page counter.
(5) Data latch mode. Input write data to the D0 to D7 pins and input an active-low pulse to the $\overline{\text{OE}}$ pin. Increment the address and the page counter.
(6) Repeat step (5) for a page (four bytes).
(7) Input a 0.1 ms program pulse (active low) to the $\overline{\text{PGM}}$ pin.
(8) Verify mode. Checks if data has been written in PROM.
Apply a low level to the $\overline{\text{CE}}$ pin, input an active-low pulse to the $\overline{\text{OE}}$ pin, and then read the write data from the D0 to D7 pins. Repeat this for a page (four bytes). When verification completes, apply a high level to the $\overline{\text{CE}}$ pin.
  - If data has been written, go to step (10).
  - If not, repeat steps (7) and (8). If no data is written yet after the steps have been repeated 10 times, go to step (9).

(9) Assume the device to be defective and stop write operation.
(10) Increment the address.
(11) Repeat steps (4) to (10) until the address exceeds the last address.

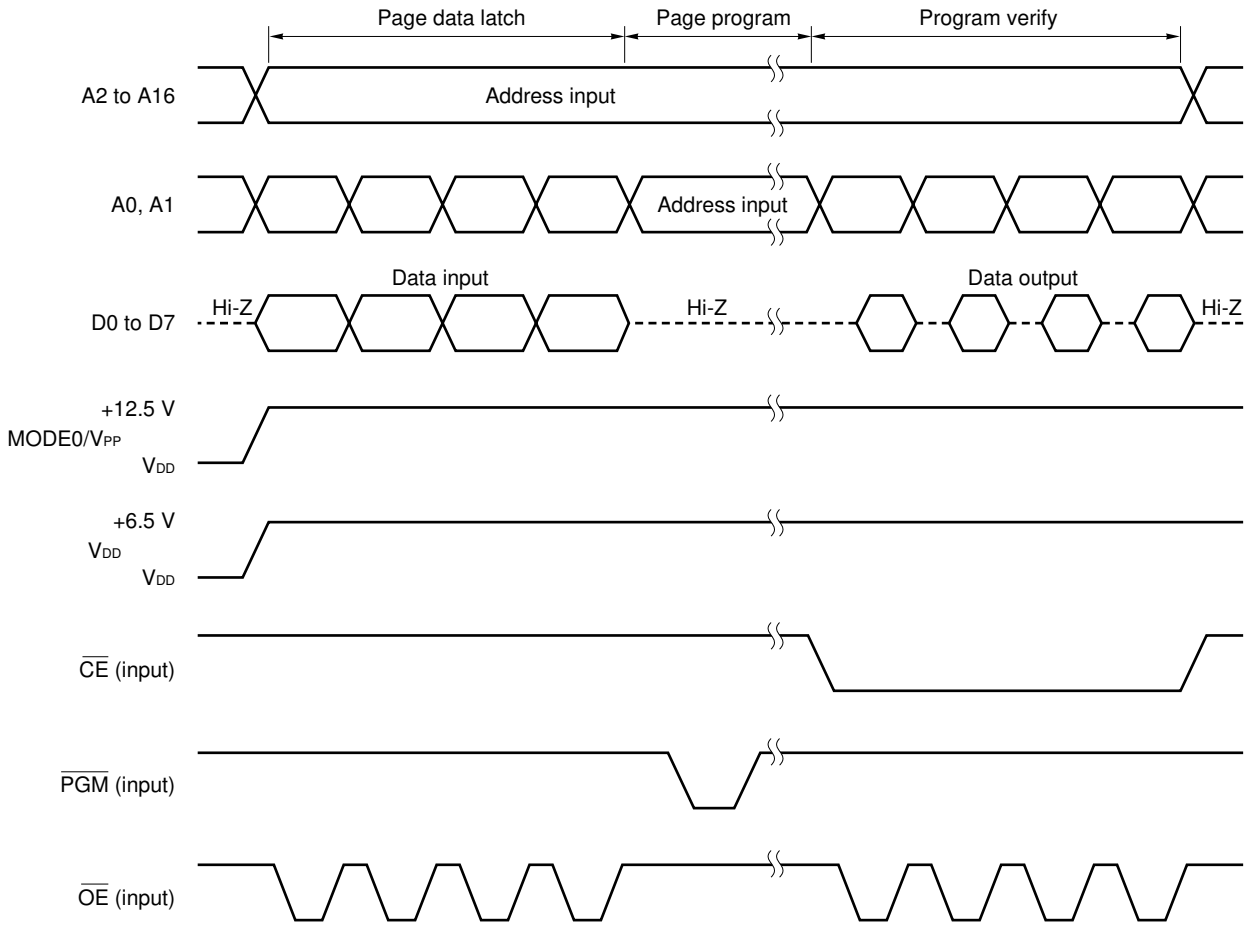
Fig. 19-3 is a timing chart of these steps (2) to (9).

Fig. 19-2. Flowchart of Procedure for Writing (Page Program Mode)



Note When write data completes midway of a page, latch FFH after the data so that the data fits into pages.

Fig. 19-3. Timing Chart of PROM Write and Verify Operation (Page Program Mode)



19.3 Procedure for Writing Data into PROM (Byte Program Mode)

The procedure for writing data into PROM is as follows (see **Fig. 19-4**).

(1) Always set each pin as follows: MODE0/V_{PP} = H, MODE1 = L, P21 = L, and $\overline{\text{RESET}}$ = L. Connect unused pins as shown in **1.4 Pin Configuration (Top View)**.
(2) Apply +6.5 V to the V_{DD} pin and +12.5 V to the MODE0/V_{PP} pin. Apply a low-level signal to the $\overline{\text{CE}}$ pin.
(3) Input an initial address to the A0 to A16 pins.
(4) Input write data to the D0 to D7 pins.
(5) Input a program pulse (active low) which has a period of 0.1 ms to the $\overline{\text{PGM}}$ pin.
(6) Verify mode. Check that data has been written in PROM.
Input an active-low pulse to the $\overline{\text{OE}}$ pin and read the written data from the D0 to D7 pin.
• When data has been written, go to step (8).
• If not, repeat steps (4) to (6). If no data is written after the steps are repeated ten times, go to step (7).
(7) Assume the device to be defective and stop write operation.
(8) Increment the address.
(9) Repeat steps (4) to (8) until the address exceeds the last address.

Fig. 19-5 is a timing chart of steps (2) to (7) above.

Fig. 19-4. Write Operation Flowchart (Byte Program Mode)

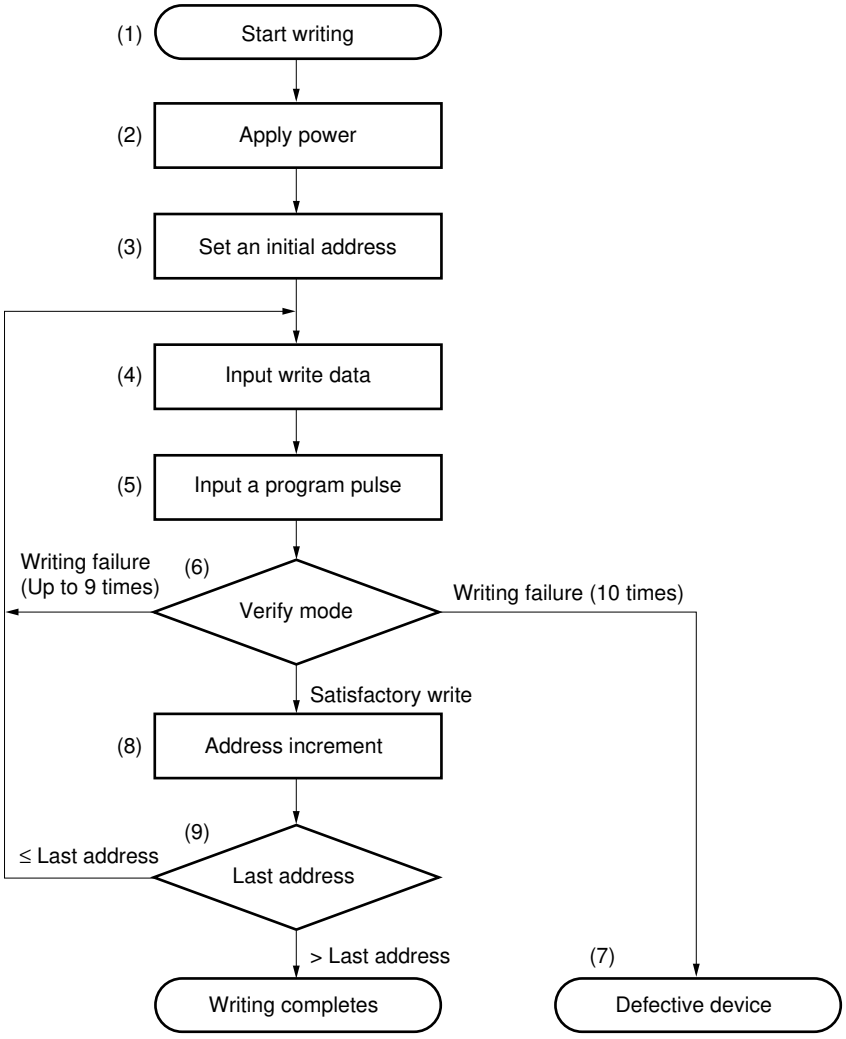
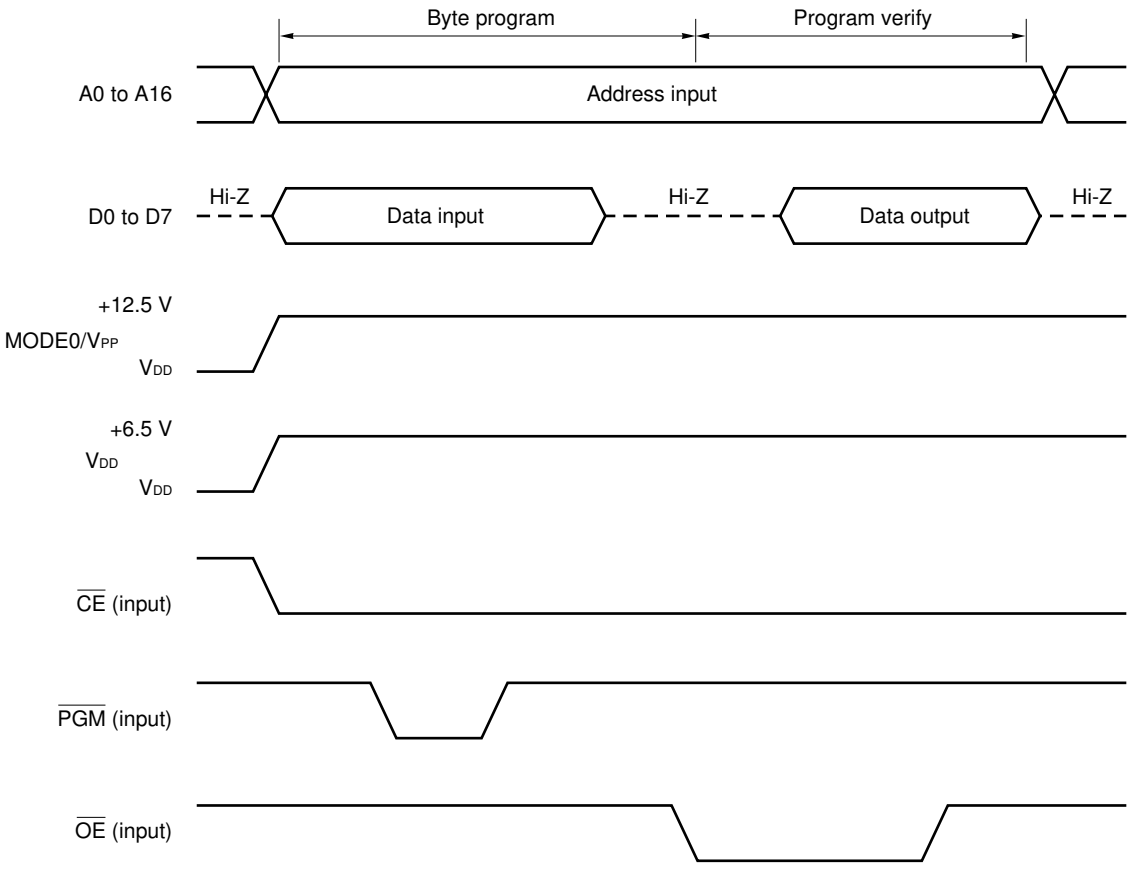


Fig. 19-5. Timing Chart of PROM Write and Verify Operation (Byte Program Mode)



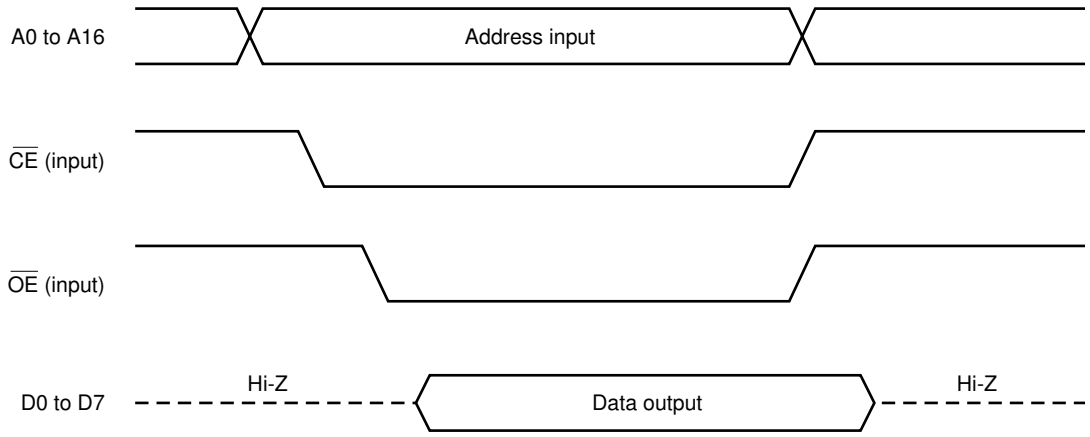
19.4 Procedure for Reading from PROM

The contents of PROM can be read out to the external data bus (D0 to D7) in the following steps:

(1) Always set the MODE0/V_{PP} pin to H, the MODE1 pin to L, the P21 pin to L, and the $\overline{\text{RESET}}$ pin to L. Connect unused pins as shown in **1.4 Pin Configuration (Top View)**.
(2) Apply +5 V to the V_{DD} and MODE0/V_{PP} pins.
(3) Input the address of data to be read into the A0 to A16 pins.
(4) Read mode ($\overline{\text{CE}}$ = L, $\overline{\text{OE}}$ = L)
(5) Output the data on the D0 to D7 pins.

Fig. 19-6 is a timing chart of these steps (2) to (5).

Fig. 19-6. PROM Read Timing Chart



19.5 Erasure Characteristics (μPD78P356KP Only)

Data written in the μPD78P356KP program memory can be erased (FFH); therefore users can write other data in the memory.

To erase the written data, expose the erasure window to light with a wavelength shorter than approx. 400 nm. Normally, ultraviolet light with a wavelength of 254 nm is employed. The amount of light required to completely erase the data is as follows:

- Intensity of ultraviolet light x erasing time: 15 W • s/cm^2 min.
- Erasing time: 15 to 20 minutes (When using a 12000 μW/cm^2 ultraviolet lamp. It may, however, take more time due to lamp deterioration, dirt on the erasure window, or the like.)

The ultraviolet lamp should be placed within 2.5 cm from the erasure window during erasure. In addition, if a filter is attached to the ultraviolet lamp, remove the filter before erasure.

19.6 Protective Film Covering the Erasure Window (μPD78P356KP Only)

After the erasure window of the μPD78P356KP has been exposed to sunlight or a fluorescent lamp for a long time, data in EPROM may be erased and the internal circuits may malfunction. To prevent these failures, the erasure window should be covered with a protective film when it is not used for erasure.

EPROM version devices with window are masked with an NEC-guaranteed protective film when they are delivered.

19.7 Screening of One-Time PROM Version Devices

The one-time PROM version (μPD78P356GC-7EA, 78P356GD-5BB) cannot be fully tested before shipping due to its structure. It is therefore recommended that, after writing data, screening be performed to verify the PROM contents after storage in high temperature in the following conditions.

Storage temperature	Storage time
125 °C	24 hours

NEC provides services from one-time PROM writing under the name QTOP™ microprocessor, to printing, screening, and verification at a charge. For details, contact your NEC dealer.

CHAPTER 20 INSTRUCTION SET

This chapter lists the operations of the μPD78356 instructions.

For details of operations of the instructions, instruction codes and clock number, refer to the μPD78356 User's Manual Instruction Volume.

20.1 Operand Identifier and Description

Operands are coded in the operand field of each instruction as listed in the description column of Table 20-1. For details of the operand format, refer to the relevant assembler specifications. When several coding forms are presented, any one of them is selected. Uppercase letters and the symbols, +, –, #, $, !, and [ ], are keywords and must be written as they are.

For immediate data, an appropriate numeric or label must be written. The symbols #, $, !, and [ ] must not be omitted when describing labels.

Table 20-1. Operand Identifier and Description

Identifier	Description
r r1 r2	R0, R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13, R14, R15 R0, R1, R2, R3, R4, R5, R6, R7 C, B
rp rp1 rp2	RP0, RP1, RP2, RP3, RP4, RP5, RP6, RP7 RP0, RP1, RP2, RP3, RP4, RP5, RP6, RP7 DE, HL, VP, UP
sfr sfrp	Special function register name (See **Table 3-4**.) Special function register name (16-bit manipulation register. See **Table 3-4**.)
post	RP0, RP1, RP2, RP3, RP4, RP5/PSW, RP6, RP7 (Can be coded more than once. However, RP5 can only be used in a PUSH or POP instruction and PSW can only be used in a PUSHU or POPU instruction.)
mem	[DE], [HL], [DE+], [HL+], [DE–], [HL–], [VP], [UP]: Register indirect mode [DE + A], [HL + A], [DE + B], [HL + B], [VP + DE], [VP + HL]: Based indexed mode [DE + byte], [HL + byte], [VP + byte], [UP + byte], [SP + byte]: Based mode word[A], word[B], word[DE], word[HL]: Indexed mode
saddr saddrp	FE20H to FF1FH Immediate data or label FE20H to FF1EH Immediate data (bit 0 = 0, however) or label (for 16-bit manipulation)
$addr16 !addr16 addr11 addr5	0000H to FDFFH Immediate data or label: Relative addressing 0000H to FDFFH Immediate data or label: Immediate addressing (Data at an address up to FFFFH can be coded in an MOV instruction. Data at an address from FE00H to FEFFH can be coded in an MOVTBLW instruction.) 800H to FFFH Immediate data or label 40H to 7EH Immediate data (bit 0 = 0, however)[Note] or label
word byte bit n	16-bit immediate data or label 8-bit immediate data or label 3-bit immediate data or label 3-bit immediate data (0 to 7)

Note Do not attempt to access word data at an odd-numbered address (bit 0 = 1).

Remarks 1. The same register name can be specified in rp and rp1, but different codes are generated.

2. Immediate addressing is effective for entire address spaces. Relative addressing is effective for the locations within a displacement range of –128 to +127 from the starting address of the next instruction.

The 8-bit registers (r, r1) and 16-bit register pairs (rp, rp1, post) can be represented by either absolute names (R0 to R15, RP0 to RP7) or function names. Table 20-2 lists the absolute names and corresponding function names of an 8-bit register. Table 20-3 lists those of a 16-bit register.

Table 20-2. Absolute Names and Their Corresponding Function Names of an 8-bit Register

Absolute name	Function name		Absolute name	Function name	
	RSS = 0	RSS = 1		RSS = 0	RSS = 1
R0	X		R8	VP_L	VP_L
R1	A		R9	VP_H	VP_H
R2	C		R10	UP_L	UP_L
R3	B		R11	UP_H	UP_H
R4		X	R12	E	E
R5		A	R13	D	D
R6		C	R14	L	L
R7		B	R15	H	H

Table 20-3. Absolute Names and Their Corresponding Function Names of a 16-bit Register

Absolute name	Function name	
	RSS = 0	RSS = 1
RP0	AX	
RP1	BC	
RP2		AX
RP3		BC
RP4	VP	VP
RP5	UP	UP
RP6	DE	DE
RP7	HL	HL

RSS stands for the register set selection flag (bit 5 of PSW). Setting or resetting RSS switches function names for correspondence with an absolute name.

20.2 Legend

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
R0 to R15	:	Register 0 to register 15 (absolute name)
AX	:	Register pair (AX); 16-bit accumulator
BC	:	Register pair (BC)
DE	:	Register pair (DE)
HL	:	Register pair (HL)
RP0 to RP7	:	Register pair 0 to register pair 7 (absolute name)
PC	:	Program counter
SP	:	Stack pointer
UP	:	User stack pointer
PSW	:	Program status word
CY	:	Carry flag
AC	:	Auxiliary carry flag
Z	:	Zero flag
P/V	:	Parity/overflow flag
S	:	Sign flag
TPF	:	Table position flag
RBS	:	Register bank selecting flag
RSS	:	Register set selecting flag
IE	:	Interrupt request enable flag
STBC	:	Standby control register
WDM	:	Watchdog timer mode register
jdisp8	:	Signed 8-bit data (displacement value: –128 to +127)
( )	:	Contents at an address enclosed in parentheses or at an address indicated in a register indicated in parentheses. (+) and (–) indicate that an address or the contents of a register indicated in parentheses are incremented and decremented by one after execution of the instruction, respectively.
(( ))	:	Contents at an address indicated by the contents at an address indicated in parentheses (( )).
xxH	:	Hexadecimal number
x_H, x_L	:	High-order 8 bits and low-order 8 bits of 16-bit register

20.3 Notational Symbols in Flag Operation Field

Table 20-4. Notational Symbols in Flag Operation Field

Symbol	Explanation
(Blank)	No change
0	Cleared to zero.
1	Set to 1.
x	Set or reset according to the result.
P	P/V flag operates as a parity flag.
V	P/V flag operates as an overflow flag.
R	Saved values are restored.

20.4 Differences between the μPD78356 and μPD78352A Instruction Sets

The μPD78356 has the following two additional instructions comparing to the μPD78352A. Except these instructions, the μPD78356 and μPD78352A instruction sets are identical.

- Sum-of-products instruction with overflow and underflow indication function
- Correlation instruction

20.5 Operations of Basic Instructions

(1) 8-bit data transfer instructions: MOV, XCH

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
MOV	r1, #byte	2	r1 <- byte					
	saddr, #byte	3	(saddr) <- byte					
	sfr[Note], #byte	3	sfr <- byte					
	r, r1	2	r <- r1					
	A, r1	1	A <- r1					
	A, saddr	2	A <- (saddr)					
	saddr, A	2	(saddr) <- A					
	saddr, saddr	3	(saddr) <- (saddr)					
	A, sfr	2	A <- sfr					
	sfr, A	2	sfr <- A					
	A, mem	1 to 4	A <- (mem)					
	mem, A	1 to 4	(mem) <- A					
	A, [saddrp]	2	A <- ((saddrp))					
	[saddrp], A	2	((saddrp)) <- A					
	A, !addr16	4	A <- (addr16)					
	!addr16, A	4	(addr16) <- A					
	PSWL, #byte	3	PSW_L <- byte	x	x	x	x	x
	PSWH, #byte	3	PSW_H <- byte					
	PSWL, A	2	PSW_L <- A	x	x	x	x	x
	PSWH, A	2	PSW_H <- A					
	A, PSWL	2	A <- PSW_L					
	A, PSWH	2	A <- PSW_H					
XCH	A, r1	1	A <–> r1					
	r, r1	2	r <–> r1					
	A, mem	2 to 4	A <–> (mem)					
	A, saddr	2	A <–> (saddr)					
	A, sfr	3	A <–> sfr					
	A, [saddrp]	2	A <–> ((saddrp))					
	saddr, saddr	3	(saddr) <–> (saddr)					

Note If STBC or WDM is coded in sfr, a different instruction having the different byte count is generated.

(2) 16-bit data transfer instructions: MOVW, XCHW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
MOVW	rp1, #word	3	rp1 <- word					
	saddrp, #word	4	(saddrp) <- word					
	sfrp, #word	4	sfrp <- word					
	rp, rp1	2	rp <- rp1					
	AX, saddrp	2	AX <- (saddrp)					
	saddrp, AX	2	(saddrp) <- AX					
	saddrp, saddrp	3	(saddrp) <- (saddrp)					
	AX, sfrp	2	AX <- sfrp					
	sfrp, AX	2	sfrp <- AX					
	rp1, !addr16	4	rp1 <- (addr16)					
	!addr16, rp1	4	(addr16) <- rp1					
	AX, mem	2 to 4	AX <- (mem)					
	mem, AX	2 to 4	(mem) <- AX					
XCHW	AX, saddrp	2	AX <–> (saddrp)					
	AX, sfrp	3	AX <–> sfrp					
	saddrp, saddrp	3	(saddrp) <–> (saddrp)					
	rp, rp1	2	rp <–> rp1					
	AX, mem	2 to 4	AX <–> (mem)					

(3) 8-bit arithmetic/logical instructions: ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP

(1/2)

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
ADD	A, #byte	2	A, CY <- A + byte	x	x	x	V	x
	saddr, #byte	3	(saddr), CY <- (saddr) + byte	x	x	x	V	x
	sfr, #byte	4	sfr, CY <- sfr + byte	x	x	x	V	x
	r, r1	2	r, CY <- r + r1	x	x	x	V	x
	A, saddr	2	A, CY <- A + (saddr)	x	x	x	V	x
	A, sfr	3	A, CY <- A + sfr	x	x	x	V	x
	saddr, saddr	3	(saddr), CY <- (saddr) + (saddr)	x	x	x	V	x
	A, mem	2 to 4	A, CY <- A + (mem)	x	x	x	V	x
	mem, A	2 to 4	(mem), CY <- (mem) + A	x	x	x	V	x
ADDC	A, #byte	2	A, CY <- A + byte + CY	x	x	x	V	x
	saddr, #byte	3	(saddr), CY <- (saddr) + byte + CY	x	x	x	V	x
	sfr, #byte	4	sfr, CY <- sfr + byte + CY	x	x	x	V	x
	r, r1	2	r, CY <- r + r1 + CY	x	x	x	V	x
	A, saddr	2	A, CY <- A + (saddr) + CY	x	x	x	V	x
	A, sfr	3	A, CY <- A + sfr + CY	x	x	x	V	x
	saddr, saddr	3	(saddr), CY <- (saddr) + (saddr) + CY	x	x	x	V	x
	A, mem	2 to 4	A, CY <- A + (mem) + CY	x	x	x	V	x
	mem, A	2 to 4	(mem), CY <- (mem) + A + CY	x	x	x	V	x
SUB	A, #byte	2	A, CY <- A – byte	x	x	x	V	x
	saddr, #byte	3	(saddr), CY <- (saddr) – byte	x	x	x	V	x
	sfr, #byte	4	sfr, CY <- sfr – byte	x	x	x	V	x
	r, r1	2	r, CY <- r – r1	x	x	x	V	x
	A, saddr	2	A, CY <- A – (saddr)	x	x	x	V	x
	A, sfr	3	A, CY <- A – sfr	x	x	x	V	x
	saddr, saddr	3	(saddr), CY <- (saddr) – (saddr)	x	x	x	V	x
	A, mem	2 to 4	A, CY <- A – (mem)	x	x	x	V	x
	mem, A	2 to 4	(mem), CY <- (mem) – A	x	x	x	V	x
SUBC	A, #byte	2	A, CY <- A – byte – CY	x	x	x	V	x
	saddr, #byte	3	(saddr), CY <- (saddr) – byte – CY	x	x	x	V	x
	sfr, #byte	4	sfr, CY <- sfr – byte – CY	x	x	x	V	x
	r, r1	2	r, CY <- r – r1 – CY	x	x	x	V	x
	A, saddr	2	A, CY <- A – (saddr) – CY	x	x	x	V	x
	A, sfr	3	A, CY <- A – sf – CY	x	x	x	V	x
	saddr, saddr	3	(saddr), CY <- (saddr) – (saddr) – CY	x	x	x	V	x
	A, mem	2 to 4	A, CY <- A – (mem) – CY	x	x	x	V	x
	mem, A	2 to 4	(mem), CY <- (mem) – A – CY	x	x	x	V	x

(2/2)

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
AND	A, #byte	2	A <- A ∧ byte	x	x		P	
	saddr, #byte	3	(saddr) <- (saddr) ∧ byte	x	x		P	
	sfr, #byte	4	sfr <- sfr ∧ byte	x	x		P	
	r, r1	2	r <- r ∧ r1	x	x		P	
	A, saddr	2	A <- A ∧ (saddr)	x	x		P	
	A, sfr	3	A <- A ∧ sfr	x	x		P	
	saddr, saddr	3	(saddr) <- (saddr) ∧ (saddr)	x	x		P	
	A, mem	2 to 4	A <- A ∧ (mem)	x	x		P	
	mem, A	2 to 4	(mem) <- (mem) ∧ A	x	x		P	
OR	A, #byte	2	A <- A ∨ byte	x	x		P	
	saddr, #byte	3	(saddr) <- (saddr) ∨ byte	x	x		P	
	sfr, #byte	4	sfr <- sfr ∨ byte	x	x		P	
	r, r1	2	r <- r ∨ r1	x	x		P	
	A, saddr	2	A <- A ∨ (saddr)	x	x		P	
	A, sfr	3	A <- A ∨ sfr	x	x		P	
	saddr, saddr	3	(saddr) <- (saddr) ∨ (saddr)	x	x		P	
	A, mem	2 to 4	A <- A ∨ (mem)	x	x		P	
	mem, A	2 to 4	(mem) <- (mem) ∨ A	x	x		P	
XOR	A, #byte	2	A <- A ⩒ byte	x	x		P	
	saddr, #byte	3	(saddr) <- (saddr) ⩒ byte	x	x		P	
	sfr, #byte	4	sfr <- sfr ⩒ byte	x	x		P	
	r, r1	2	r <- r ⩒ r1	x	x		P	
	A, saddr	2	A <- A ⩒ (saddr)	x	x		P	
	A, sfr	3	A <- A ⩒ sfr	x	x		P	
	saddr, saddr	3	(saddr) <- (saddr) ⩒ (saddr)	x	x		P	
	A, mem	2 to 4	A <- A ⩒ (mem)	x	x		P	
	mem, A	2 to 4	(mem) <- (mem) ⩒ A	x	x		P	
CMP	A, #byte	2	A – byte	x	x	x	V	x
	saddr, #byte	3	(saddr) – byte	x	x	x	V	x
	sfr, #byte	4	sfr – byte	x	x	x	V	x
	r, r1	2	r – r1	x	x	x	V	x
	A, saddr	2	A – (saddr)	x	x	x	V	x
	A, sfr	3	A – sfr	x	x	x	V	x
	saddr, saddr	3	(saddr) – (saddr)	x	x	x	V	x
	A, mem	2 to 4	A – (mem)	x	x	x	V	x
	mem, A	2 to 4	(mem) – A	x	x	x	V	x

(4) 16-bit arithmetic/logical instructions: ADDW, SUBW, CMPW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
ADDW	AX, #word	3	AX, CY <- AX + word	x	x	x	V	x
	saddrp, #word	4	(saddrp), CY <- (saddrp) + word	x	x	x	V	x
	sfrp, #word	5	sfrp, CY <- sfrp + word	x	x	x	V	x
	rp, rp1	2	rp, CY <- rp + rp1	x	x	x	V	x
	AX, saddrp	2	AX, CY <- AX + (saddrp)	x	x	x	V	x
	AX, sfrp	3	AX, CY <- AX + sfrp	x	x	x	V	x
	saddrp, saddrp	3	(saddrp), CY <- (saddrp) + (saddrp)	x	x	x	V	x
SUBW	AX, #word	3	AX, CY <- AX – word	x	x	x	V	x
	saddrp, #word	4	(saddrp), CY <- (saddrp) – word	x	x	x	V	x
	sfrp, #word	5	sfrp, CY <- sfrp – word	x	x	x	V	x
	rp, rp1	2	rp, CY <- rp – rp1	x	x	x	V	x
	AX, saddrp	2	AX, CY <- AX – (saddrp)	x	x	x	V	x
	AX, sfrp	3	AX, CY <- AX – sfrp	x	x	x	V	x
	saddrp, saddrp	3	(saddrp), CY <- (saddrp) – (saddrp)	x	x	x	V	x
CMPW	AX, #word	3	AX – word	x	x	x	V	x
	saddrp, #word	4	(saddrp) – word	x	x	x	V	x
	sfrp, #word	5	sfrp – word	x	x	x	V	x
	rp, rp1	2	rp – rp1	x	x	x	V	x
	AX, saddrp	2	AX – (saddrp)	x	x	x	V	x
	AX, sfrp	3	AX – sfrp	x	x	x	V	x
	saddrp, saddrp	3	(saddrp) – (saddrp)	x	x	x	V	x

(5) Multiply/divide instructions: MULU, DIVUW, MULUW, DIVUX

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
MULU	r1	2	AX <- A x r1					
DIVUW	r1	2	AX (quotient) <- r1 (remainder) <- AX ÷ r1					
MULUW	rp1	2	AX (high-order 16 bits), rp1 (low-order 16 bits) <- AX x rp1					
DIVUX	rp1	2	AXDE (quotient), rp1 (remainder) <- AXDE ÷ rp1					

(6) Signed multiply instruction: MULW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
MULW	rp1	2	AX (high-order 16 bits), rp1 (low-order 16 bits) <- AX x rp1					

(7) Sum-of-products instruction: MACW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
MACW	n	3	AXDE <- (B) x (C) + AXDE B <- B + 2, C <- C + 2, n <- n – 1 End if n = 0 or P/V = 1	x	x	x	V	x

(8) Sum-of-products instruction with overflow and underflow indication function: MACSW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY

MACSW	n	3	AXDE <- (B) x (C) + AXDE
			B <- B + 2, C <- C + 2, n <- n – 1
			If overflow (P/V = 1) then
			AXDE <- 7FFFFFFFH
			if underflow (P/V = 1) then
			AXDE <- 80000000H
			end if n = 0 or P/V = 1
				x	x	x	V	x

(9) Correlation instruction: SACW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
SACW	[DE +], [HL +]	4	AX <- AX + | (DE) – (HL) | DE <- DE + 2, HL <- HL + 2, C <- C – 1 end if C = 0 or CY = 1	x	x	x	V	x

(10) Table shift instruction: MOVTBLW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
MOVTBLW	!addr16, n	4	(addr16 + 2) <- (addr16), n <- n – 1 addr16 <- addr16 – 2, End if n = 0					

Remark The addressing range of the table shift instruction is FE00H to FEFFH.

(11) Increment/decrement instructions: INC, DEC, INCW, DECW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
INC	r1	1	r1 <- r1 + 1	x	x	x	V	
	saddr	2	(saddr) <- (saddr) + 1	x	x	x	V	
DEC	r1	1	r1 <- r1 – 1	x	x	x	V	
	saddr	2	(saddr) <- (saddr) – 1	x	x	x	V	
INCW	rp2	1	rp2 <- rp2 + 1					
	saddrp	3	(saddrp) <- (saddrp) + 1					
DECW	rp2	1	rp2 <- rp2 – 1					
	saddrp	3	(saddrp) <- (saddrp) – 1					

(12) Shift/rotate instructions: ROR, ROL, RORC, ROLC, SHR, SHL, SHRW, SHLW, ROR4, ROL4

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
ROR	r1, n	2	(CY, $r1_7$ <- $r1_0$, $r1_{m-1}$ <- $r1_m$) x n times (n = 0 to 7)				P	x
ROL	r1, n	2	(CY, $r1_0$ <- $r1_7$, $r1_{m+1}$ <- $r1_m$) x n times (n = 0 to 7)				P	x
RORC	r1, n	2	(CY <- $r1_0$, $r1_7$, <- CY, $r1_{m-1}$ <- $r1_m$) x n times (n = 0 to 7)				P	x
ROLC	r1, n	2	(CY <- $r1_7$, $r1_0$, <- CY, $r1_{m+1}$ <- $r1_m$) x n times (n = 0 to 7)				P	x
SHR	r1, n	2	(CY <- $r1_0$, $r1_7$, <- 0, $r1_{m-1}$ <- $r1_m$) x n times (n = 0 to 7)	x	x	0	P	x
SHL	r1, n	2	(CY <- $r1_7$, $r1_0$, <- 0, $r1_{m+1}$ <- $r1_m$) x n times (n = 0 to 7)	x	x	0	P	x
SHRW	rp1, n	2	(CY <- $rp1_0$, $rp1_{15}$, <- 0, $rp1_{m-1}$ <- $rp1_m$) x n times (n = 0 to 7)	x	x	0	P	x
SHLW	rp1, n	2	(CY <- $rp1_{15}$, $rp1_0$, <- 0, $rp1_{m+1}$ <- $rp1_m$) x n times (n = 0 to 7)	x	x	0	P	x
ROR4	[rp1]	2	A_{3-0} <- $(rp1)_{3-0}$, $(rp1)_{7-4}$ <- A_{3-0}, $(rp1)_{3-0}$ <- $(rp1)_{7-4}$					
ROL4	[rp1]	2	A_{3-0} <- $(rp1)_{7-4}$, $(rp1)_{3-0}$ <- A_{3-0}, $(rp1)_{7-4}$ <- $(rp1)_{3-0}$					

Remark n indicates the number of shifts or rotations.

(13) BCD adjust instructions: ADJBA, ADJBS

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
ADJBA		2	Decimal Adjust Accumulator	x	x	x	P	x
ADJBS								

(14) Data conversion instruction: CVTBW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
CVTBW		1	When A_7 = 0, X <- A, A <- 00H When A_7 = 1, X <- A, A <- FFH					

(15) Bit manipulation instructions: MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1

(1/2)

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
MOV1	CY, saddr.bit	3	CY <- (saddr.bit)					x
	CY, sfr.bit	3	CY <- sfr.bit					x
	CY, A.bit	2	CY <- A.bit					x
	CY, X.bit	2	CY <- X.bit					x
	CY, PSWH.bit	2	CY <- PSW_H.bit					x
	CY, PSWL.bit	2	CY <- PSW_L.bit					x
	saddr.bit, CY	3	(saddr.bit) <- CY					
	sfr.bit, CY	3	sfr.bit <- CY					
	A.bit, CY	2	A.bit <- CY					
	X.bit, CY	2	X.bit <- CY					
	PSWH.bit, CY	2	PSW_H.bit <- CY					
	PSWL.bit, CY	2	PSW_L.bit <- CY					
AND1	CY, saddr.bit	3	CY <- CY $\wedge$ (saddr.bit)					x
	CY, /saddr.bit	3	CY <- CY $\wedge$ ($\overline{\text{saddr.bit}}$)					x
	CY, sfr.bit	3	CY <- CY $\wedge$ sfr.bit					x
	CY, /sfr.bit	3	CY <- CY $\wedge$ $\overline{\text{sfr.bit}}$					x
	CY, A.bit	2	CY <- CY $\wedge$ A.bit					x
	CY, /A.bit	2	CY <- CY $\wedge$ $\overline{\text{A.bit}}$					x
	CY, X.bit	2	CY <- CY $\wedge$ X.bit					x
	CY, /X.bit	2	CY <- CY $\wedge$ $\overline{\text{X.bit}}$					x
	CY, PSWH.bit	2	CY <- CY $\wedge$ PSW_H.bit					x
	CY, /PSWH.bit	2	CY <- CY $\wedge$ $\overline{PSW_H.bit}$					x
	CY, PSWL.bit	2	CY <- CY $\wedge$ PSW_L.bit					x
	CY, /PSWL.bit	2	CY <- CY $\wedge$ $\overline{PSW_L.bit}$					x
OR1	CY, saddr.bit	3	CY <- CY $\vee$ (saddr.bit)					x
	CY, /saddr.bit	3	CY <- CY $\vee$ ($\overline{\text{saddr.bit}}$)					x
	CY, sfr.bit	3	CY <- CY $\vee$ sfr.bit					x
	CY, /sfr.bit	3	CY <- CY $\vee$ $\overline{\text{sfr.bit}}$					x
	CY, A.bit	2	CY <- CY $\vee$ A.bit					x
	CY, /A.bit	2	CY <- CY $\vee$ $\overline{\text{A.bit}}$					x
	CY, X.bit	2	CY <- CY $\vee$ X.bit					x
	CY, /X.bit	2	CY <- CY $\vee$ $\overline{\text{X.bit}}$					x
	CY, PSWH.bit	2	CY <- CY $\vee$ PSW_H.bit					x
	CY, /PSWH.bit	2	CY <- CY $\vee$ $\overline{PSW_H.bit}$					x
	CY, PSWL.bit	2	CY <- CY $\vee$ PSW_L.bit					x
	CY, /PSWL.bit	2	CY <- CY $\vee$ $\overline{PSW_L.bit}$					x

(2/2)

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
XOR1	CY, saddr.bit	3	CY <- CY ∀ (saddr.bit)					x
	CY, sfr.bit	3	CY <- CY ∀ sfr.bit					x
	CY, A.bit	2	CY <- CY ∀ A.bit					x
	CY, X.bit	2	CY <- CY ∀ X.bit					x
	CY, PSWH.bit	2	CY <- CY ∀ PSW_H.bit					x
	CY, PSWL.bit	2	CY <- CY ∀ PSW_L.bit					x
SET1	saddr.bit	2	(saddr.bit) <- 1					
	sfr.bit	3	sfr.bit <- 1					
	A.bit	2	A.bit <- 1					
	X.bit	2	X.bit <- 1					
	PSWH.bit	2	PSW_H.bit <- 1					
	PSWL.bit	2	PSW_L.bit <- 1	x	x	x	x	x
	CY	1	CY <- 1					1
CLR1	saddr.bit	2	(saddr.bit) <- 0					
	sfr.bit	3	sfr.bit <- 0					
	A.bit	2	A.bit <- 0					
	X.bit	2	X.bit <- 0					
	PSWH.bit	2	PSW_H.bit <- 0					
	PSWL.bit	2	PSW_L.bit <- 0	x	x	x	x	x
	CY	1	CY <- 0					0
NOT1	saddr.bit	3	(saddr.bit) <- $\overline{(saddr.bit)}$					
	sfr.bit	3	sfr.bit <- $\overline{sfr.bit}$					
	A.bit	2	A.bit <- $\overline{A.bit}$					
	X.bit	2	X.bit <- $\overline{X.bit}$					
	PSWH.bit	2	PSW_H.bit <- $\overline{PSW_H.bit}$					
	PSWL.bit	2	PSW_L.bit <- $\overline{PSW_L.bit}$	x	x	x	x	x
	CY	1	CY <- CY					x

(16) Call/return instructions: CALL, CALLF, CALLT, BRK, RET, RETB, RETI

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
CALL	!addr16	3	$(SP - 1) \leftarrow (PC + 3)_H$, $(SP - 2) \leftarrow (PC + 3)_L$, $PC \leftarrow$ addr16, $SP \leftarrow SP - 2$					
	rp1	2	$(SP - 1) \leftarrow (PC + 2)_H$, $(SP - 2) \leftarrow (PC + 2)_L$, $PC_H \leftarrow rp1_H$, $PC_L \leftarrow rp1_L$, $SP \leftarrow SP - 2$					
	[rp1]	2	$(SP - 1) \leftarrow (PC + 2)_H$, $(SP - 2) \leftarrow (PC + 2)_L$, $PC_H \leftarrow (rp1 + 1)$, $PC_L \leftarrow (rp1)$, $SP \leftarrow SP - 2$					
CALLF	!addr11	2	$(SP - 1) \leftarrow (PC + 2)_H$, $(SP - 2) \leftarrow (PC + 2)_L$, $PC_{15-11} \leftarrow 00001$, $PC_{10-0} \leftarrow$ addr11, $SP \leftarrow SP - 2$					
CALLT	[addr5]	1	$(SP - 1) \leftarrow (PC + 1)_H$, $(SP - 2) \leftarrow (PC + 1)_L$, $PC_H \leftarrow$ (TPF, 00000000, addr5 + 1), $PC_L \leftarrow$ (TPF, 00000000, addr5), $SP \leftarrow SP - 2$					
BRK		1	$(SP - 1) \leftarrow PSW_H$, $(SP - 2) \leftarrow PSW_L$, $(SP - 3) \leftarrow (PC + 1)_H$, $(SP - 4) \leftarrow (PC + 1)_L$, $PC_L \leftarrow$ (003EH), $PC_H \leftarrow$ (003FH) $SP \leftarrow SP - 4$, IE <- 0					
RET		1	$PC_L \leftarrow$ (SP), $PC_H \leftarrow$ (SP + 1), $SP \leftarrow SP + 2$					
RETB		1	$PC_L \leftarrow$ (SP), $PC_H \leftarrow$ (SP + 1), $PSW_L \leftarrow$ (SP + 2), $PSW_H \leftarrow$ (SP + 3), $SP \leftarrow SP + 4$	R	R	R	R	R
RETI		1	$PC_L \leftarrow$ (SP), $PC_H \leftarrow$ (SP + 1), $PSW_L \leftarrow$ (SP + 2), $PSW_H \leftarrow$ (SP + 3), $SP \leftarrow SP + 4$, ISPRn <- 0**Note**	R	R	R	R	R

Note The bit corresponding to the interrupt request with the highest priority amongst the bits (n = 0 to 3) set to (1) in the ISPR register during RETI instruction is reset (0).

(17) Stack manipulation instructions: PUSH, PUSHU, POP, POPU, MOVW, INCW, DECW

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
PUSH	sfrp	3	(SP – 1) <- sfr_H, (SP – 2) <- sfr_L, SP <- SP – 2					
	post	2	{(SP – 1) <- $post_H$, (SP – 2) <- $post_L$, SP <- SP – 2} x n times					
	PSW	1	(SP – 1) <- PSW_H, (SP – 2) <- PSW_L, SP <- SP – 2					
PUSHU	post	2	{(UP – 1) <- $post_H$, (UP – 2) <- $post_L$, UP <- UP – 2} x n times					
POP	sfrp	3	sfr_L <- (SP), sfr_H <- (SP + 1), SP <- SP + 2					
	post	2	{$post_L$ <- (SP), $post_H$ <- (SP + 1), SP <- SP + 2} x n times					
	PSW	1	PSW_L <- (SP), PSW_H <- (SP + 1), SP <- SP + 2	R	R	R	R	R
POPU	post	2	{$post_L$ <- (UP), $post_H$ <- (UP + 1), UP <- UP + 2} x n times					
MOVW	SP, #word	4	SP <- word					
	SP, AX	2	SP <- AX					
	AX, SP	2	AX <- SP					
INCW	SP	2	SP <- SP + 1					
DECW	SP	2	SP <- SP – 1					

Remark n indicates the number of registers specified

(18) Special instructions: CHKL, CHKLA

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
CHKL	sfr	3	(Pin level) ∀ (Signal level before output buffer)	x	x		P	
CHKLA	sfr	3	A <- {(Pin level) ∀ (Signal level before output buffer)}	x	x		P	

(19) Unconditional branch instruction: BR

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
BR	!addr16	3	PC <- addr16					
	rp1	2	PC_H <- $rp1_H$, PC_L <- $rp1_L$					
	[rp1]	2	PC_H <- (rp1 + 1), PC_L <- (rp1)					
	$addr16	2	PC <- PC + 2 + jdisp8					

(20) Conditional branch instructions: BC, BL, BNC, BNL, BZ, BE, BNZ, BNE, BV, BPE, BNV, BPO, BN, BP, BGT, BGE, BLT, BLE, BH, BNH, BT, BF, BTCLR, BFSET, DBNZ

(1/2)

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
BC	$addr16	2	PC <- PC + 2 + jdisp8 if CY = 1					
BL								
BNC	$addr16	2	PC <- PC + 2 + jdisp8 if CY = 0					
BNL								
BZ	$addr16	2	PC <- PC + 2 + jdisp8 if Z = 1					
BE								
BNZ	$addr16	2	PC <- PC + 2 + jdisp8 if Z = 0					
BNE								
BV	$addr16	2	PC <- PC + 2 + jdisp8 if P/V = 1					
BPE								
BNV	$addr16	2	PC <- PC + 2 + jdisp8 if P/V = 1					
BPO								
BN	$addr16	2	PC <- PC + 2 + jdisp8 if S = 1					
BP	$addr16	2	PC <- PC + 2 + jdisp8 if S = 0					
BGT	$addr16	3	PC <- PC + 3 + jdisp8 if (P/V $\forall$ S) $\vee$ Z = 0					
BGE	$addr16	3	PC <- PC + 3 + jdisp8 if P/V $\forall$ S = 0					
BLT	$addr16	3	PC <- PC + 3 + jdisp8 if P/V $\forall$ S = 1					
BLE	$addr16	3	PC <- PC + 3 + jdisp8 if (P/V $\forall$ S) $\vee$ Z = 1					
BH	$addr16	3	PC <- PC + 3 + jdisp8 if Z $\vee$ CY = 0					
BNH	$addr16	3	PC <- PC + 3 + jdisp8 if Z $\vee$ CY = 1					
BT	saddr.bit, $addr16	3	PC <- PC + 3 + jdisp8 if (saddr.bit) = 1					
	sfr.bit, $addr16	4	PC <- PC + 4 + jdisp8 if sfr.bit = 1					
	A.bit, $addr16	3	PC <- PC + 3 + jdisp8 if A.bit = 1					
	X.bit, $addr16	3	PC <- PC + 3 + jdisp8 if X.bit = 1					
	PSWH.bit, $addr16	3	PC <- PC + 3 + jdisp8 if PSW_H.bit = 1					
	PSWL.bit, $addr16	3	PC <- PC + 3 + jdisp8 if PSW_L.bit = 1					
BF	saddr.bit, $addr16	4	PC <- PC + 4 + jdisp8 if (saddr.bit) = 0					
	sfr.bit, $addr16	4	PC <- PC + 4 + jdisp8 if sfr.bit = 0					
	A.bit, $addr16	3	PC <- PC + 3 + jdisp8 if A.bit = 0					
	X.bit, $addr16	3	PC <- PC + 3 + jdisp8 if X.bit = 0					
	PSWH.bit, $addr16	3	PC <- PC + 3 + jdisp8 if PSW_H.bit = 0					
	PSWL.bit, $addr16	3	PC <- PC + 3 + jdisp8 if PSW_L.bit = 0					

(2/2)

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
BTCLR	saddr.bit, $addr16	4	PC <- PC + 4 + jdisp8 if (saddr.bit) = 1 then reset (saddr.bit)					
	sfr.bit, $addr16	4	PC <- PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit					
	A.bit, $addr16	3	PC <- PC + 3 + jdisp8 if A.bit = 1 then reset A.bit					
	X.bit, $addr16	3	PC <- PC + 3 + jdisp8 if X.bit = 1 then reset X.bit					
	PSWH.bit, $addr16	3	PC <- PC + 3 + jdisp8 if PSW_H.bit = 1 then reset PSW_H.bit					
	PSWL.bit, $addr16	3	PC <- PC + 3 + jdisp8 if PSW_L.bit = 1 then reset PSW_L.bit	x	x	x	x	x
BFSET	saddr.bit, $addr16	4	PC <- PC + 4 + jdisp8 if (saddr.bit) = 0 then set (saddr.bit)					
	sfr.bit, $addr16	4	PC <- PC + 4 + jdisp8 if sfr.bit = 0 then set sfr.bit					
	A.bit, $addr16	3	PC <- PC + 3 + jdisp8 if A.bit = 0 then set A.bit					
	X.bit, $addr16	3	PC <- PC + 3 + jdisp8 if X.bit = 0 then set X.bit					
	PSWH.bit, $addr16	3	PC <- PC + 3 + jdisp8 if PSW_H.bit = 0 then set PSW_H.bit					
	PSWL.bit, $addr16	3	PC <- PC + 3 + jdisp8 if PSW_L.bit = 0 then set PSW_L.bit	x	x	x	x	x
DBNZ	r2, $addr16	2	r2 <- r2 – 1, then PC <- PC + 2 + jdisp8 if r2 ≠ 0					
	saddr, $addr16	3	(saddr) <- (saddr) – 1, then PC <- PC + 3 + jdisp8 if (saddr) ≠ 0					

(21) Context switching instructions: BRKCS, RETCS, RETCSB

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
BRKCS	RBn	2	RBS2 – 0 <- n, PC_H <–> R5, PC_L <–> R4, R7 <- PSW_H, R6 <- PSW_L, RSS <- 0, IE <- 0					
RETCS	!addr16	3	PC_H <- R5, PC_L <- R4, R5 <- $addr16_H$, R4 <- $addr16_L$, PSW_H <- R7, PSW_L <- R6	R	R	R	R	R
RETCSB	!addr16	4	PC_H <- R5, PC_L <- R4, R5 <- $addr16_H$, R4 <- $addr16_L$, PSW_H <- R7, PSW_L <- R6	R	R	R	R	R

(22) String instructions: MOVM, MOVBK, XCHM, XCHBK, CMPME, CMPBKE, CMPMNE, CMPBKNE,CMPMC, CMPBKC, CMPMNC, CMPBKNC

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
MOVM	[DE+], A	2	(DE+) <- A, C <- C – 1, End if C = 0					
	[DE–], A	2	(DE–) <- A, C <- C – 1, End if C = 0					
MOVBK	[DE+], [HL+]	2	(DE+) <- (HL+), C <- C – 1, End if C = 0					
	[DE–], [HL–]	2	(DE–) <- (HL–), C <- C – 1, End if C = 0					
XCHM	[DE+], A	2	(DE+) <–> A, C <- C – 1, End if C = 0					
	[DE–], A	2	(DE–) <–> A, C <- C – 1, End if C = 0					
XCHBK	[DE+], [HL+]	2	(DE+) <–> (HL+), C <- C – 1, End if C = 0					
	[DE–], [HL–]	2	(DE–) <–> (HL–), C <- C – 1, End if C = 0					
CMPME	[DE+], A	2	(DE+) – A, C <- C – 1, End if C = 0 or Z = 0	x	x	x	V	x
	[DE–], A	2	(DE–) – A, C <- C – 1, End if C = 0 or Z = 0	x	x	x	V	x
CMPBKE	[DE+], [HL+]	2	(DE+) – (HL+), C <- C – 1, End if C = 0 or Z = 0	x	x	x	V	x
	[DE–], [HL–]	2	(DE–) – (HL–), C <- C – 1, End if C = 0 or Z = 0	x	x	x	V	x
CMPMNE	[DE+], A	2	(DE+) – A, C <- C – 1, End if C = 0 or Z = 1	x	x	x	V	x
	[DE–], A	2	(DE–) – A, C <- C – 1, End if C = 0 or Z = 1	x	x	x	V	x
CMPBKNE	[DE+], [HL+]	2	(DE+) – (HL+), C <- C – 1, End if C = 0 or Z = 1	x	x	x	V	x
	[DE–], [HL–]	2	(DE–) – (HL–), C <- C – 1, End if C = 0 or Z = 1	x	x	x	V	x
CMPMC	[DE+], A	2	(DE+) – A, C <- C – 1, End if C = 0 or CY = 0	x	x	x	V	x
	[DE–], A	2	(DE–) – A, C <- C – 1, End if C = 0 or CY = 0	x	x	x	V	x
CMPBKC	[DE+], [HL+]	2	(DE+) – (HL+), C <- C – 1, End if C = 0 or CY = 0	x	x	x	V	x
	[DE–], [HL–]	2	(DE–) – (HL–), C <- C – 1, End if C = 0 or CY = 0	x	x	x	V	x
CMPMNC	[DE+], A	2	(DE+) – A, C <- C – 1, End if C = 0 or CY = 1	x	x	x	V	x
	[DE–], A	2	(DE–) – A, C <- C – 1, End if C = 0 or CY = 1	x	x	x	V	x
CMPBKNC	[DE+], [HL+]	2	(DE+) – (HL+), C <- C – 1, End if C = 0 or CY = 1	x	x	x	V	x
	[DE–], [HL–]	2	(DE–) – (HL–), C <- C – 1, End if C = 0 or CY = 1	x	x	x	V	x

(23) CPU control instructions: MOV, SWRS, SEL, NOP, EI, DI

Mnemonic	Operand	Byte	Operation	Flag				
				S	Z	AC	P/V	CY
MOV	STBC, #byte	4	STBC <- byte[Note]					
	WDM, #byte	4	WDM <- byte[Note]					
SWRS		1	RSS <- $\overline{RSS}$					
SEL	RBn	2	RSS2 – 0 <- n, RSS <- 0					
	RBn, ALT	2	RSS2 – 0 <- n, RSS <- 1					
NOP		1	No Operation					
EI		1	IE <- 1 (Enable Interrupt)					
DI		1	IE <- 1 (Disable Interrupt)					

Note An op-code trap interrupt occurs if an invalid op-code is specified in an STBC or WDM register manipulation instruction.

Trap operation: $(SP - 1) \leftarrow PSW_H$, $(SP - 2) \leftarrow PSW_L$,
$(SP - 3) \leftarrow (PC - 4)_H$, $(SP - 4) \leftarrow (PC - 4)_L$,
$PC_L \leftarrow (003CH)$, $PC_H \leftarrow (003DH)$,
$SP \leftarrow SP - 4$, $IE \leftarrow 0$

[MEMO]

CHAPTER 21 INSTRUCTION EXECUTION RATE

21.1 Memory Space and Access Speed

21.1.1 Main RAM and peripheral RAM

The μPD78356 has 2048-byte internal RAM in the area from F700H to FEFFH. The internal RAM is divided into two sections, main RAM and peripheral RAM, according to the access speed.

Main RAM is accessed at the highest speed, because it is located in the execution unit (EXU).

- Main RAM (FE00H to FEFFH) ..... 1 clock per word

- Peripheral RAM (F700H to FDFFH) ..... 3 + n clocks per word (when word access to even address)

Caution When executing word access (including stack manipulation) in the main RAM area (FE00H to FEFFH), only even addresses can be specified with the operand.

Remark n is a wait count specified by the programmable wait control register (PWC).

21.1.2 Memory access

(1) Op-code fetch

(a) Access range

The μPD78356 allows op-code fetch in the area from 0000H to FDFFH (see **Fig. 21-1**).

Op-code fetch is not allowed in the area from FE00H to FFFFH.

(b) Number of clocks required for access

The number of clocks required for op-code fetch can be specified by the PWC register in units of 16 Kbytes.

The number of clocks vary according to the area accessed.

Only an area in internal memory can be specified in the high-speed fetch cycle mode.

Table 21-1. The Number of Clocks Registered for Op-code Fetch

Access area		Access cycle
Peripheral RAM	Normal fetch	3 + n clocks per word
	High-speed fetch	2 clocks per word
External memory		3 + n clocks per word
Internal ROM[Note]	Normal fetch	3 + n clocks per word
	High-speed fetch	2 clocks per word

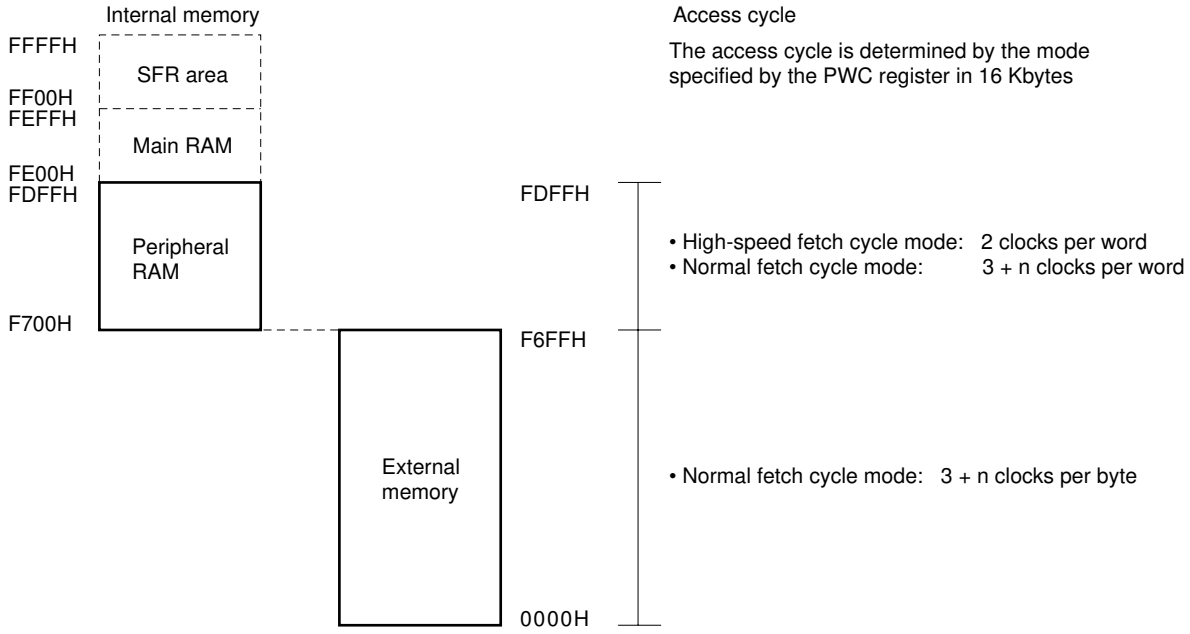
Note μPD78356 and 78P356 only

Remark n is a wait count specified by the PWC register.

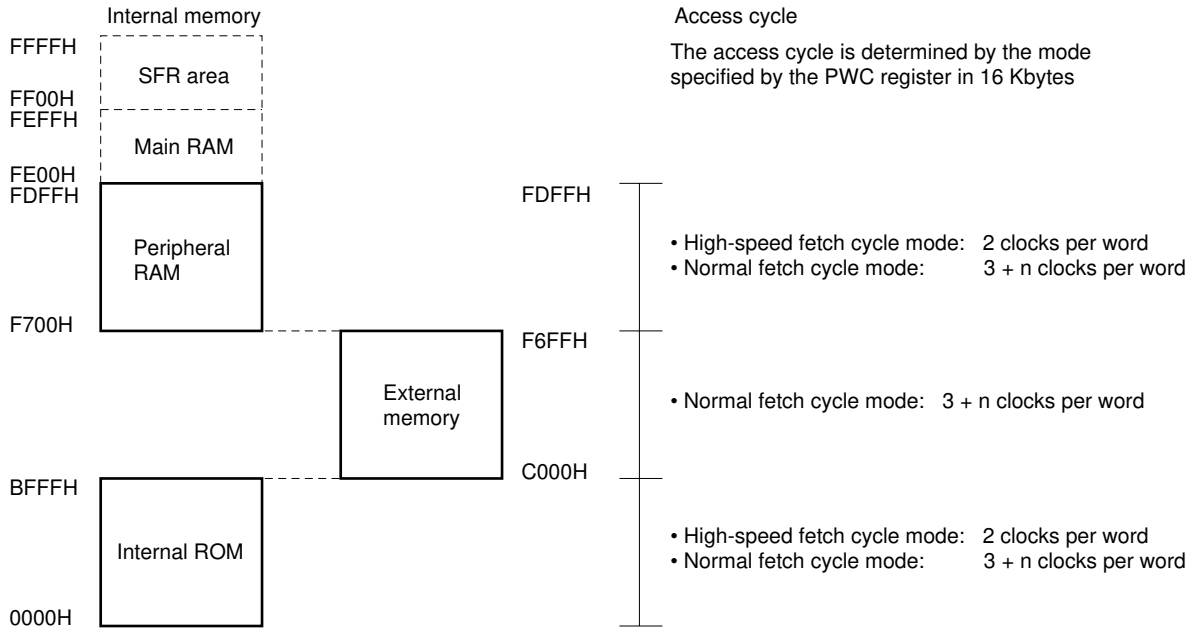
Caution The PWC register specifies the mode of an area, irrespective of whether the area is in internal memory or external memory. After reset, the whole space is set in the normal fetch cycle mode.

Fig. 21-1. Concept of Memory Access in Op-Code Fetch

(1) μPD78355



(2) μPD78356 and μPD78P356



Caution When executing word access (including stack manipulation) in the main RAM area (FE00H to FEFFH), only even addresses can be specified with the operand.

Remark n is the wait count specified by the PWC register.

(c) Bus control signals for memory access (ASTB, $\overline{RD}$, $\overline{LWR}$ and $\overline{HWR}$)

When an op-code is fetched, the μPD78356 outputs the ASTB, $\overline{RD}$, $\overline{LWR}$ and/or $\overline{HWR}$ bus control signals. The signals to be output vary according to the area accessed. (See **Table 21-2**.)

Table 21-2. Bus Control Signals in Op-Code Fetch

(1) μPD78355

Access area		Address	ASTB	$\overline{RD}$	$\overline{LWR}$	$\overline{HWR}$
Peripheral RAM	Normal fetch	F700H to FDFFH	○	–	–	–
	High-speed fetch		–Note	–	–	–
External memory		0000H to F6FFH	○	○	–	–

Remark ○ : Outputs the signal.
– : Does not output the signal (outputs the inactive level).

Note The ASTB signal is output only when the BR instruction is issued.

(2) μPD78356 and μPD78P356

Access area		Address	ASTB	$\overline{RD}$	$\overline{LWR}$	$\overline{HWR}$
Peripheral RAM	Normal fetch	F700H to FDFFH	○	–	–	–
	High-speed fetch		–Note	–	–	–
External memory		C000H to F6FFH	○	○	–	–
Internal ROM	Normal fetch	0000H to BFFFH	○	–	–	–
	High-speed fetch		–Note	–	–	–

Remark ○ : Outputs the signal.
– : Does not output the signal (outputs the inactive level).

Note The ASTB signal is output only when the BR instruction is issued.

(3) Data access

While instructions such as MOV A, [HL] and SUB [DE+], A are being executed, data is read from or written into memory.

(a) Access range

The μPD78356 allows data access to the whole 64K-byte range. In the address range from FE00H to FFFFH, access to internal memory has precedence. In the 16-byte external SFR area from FFD0H to FFDFH, however, external memory is accessed. (See **Fig. 21-2**.)

(b) Number of clocks required for access

The number of clocks required for data access depends on the area to be accessed.

Table 21-3. Number of Clocks Required for Data Access

Area to be accessed	Address	Read access	Write access
Main RAM	FE00H to FEFFH	1 Clock per word	1 Clock per word
SFR	FF00H to FFFFH	4 Clocks per word	4 Clocks per word
Internal ROM[Note 1]	0000H to BFFFH	3 + n clocks per word	—
Peripheral RAM	F700H to FDFFH		1 Clock per word[Note 2]
External SFR, external memory	FFD0H to FFDFH —	3 + n clocks per byte	1 Clock per byte[Note 2]

Remark n is the wait count specified by the PWC register.

Notes 1. μPD78356 and μPD78P356 only

2. One clock is required to execute data write access to peripheral RAM, external SFR or external memory, in which only the address and data are passed to the bus control unit (BCU). However, the bus is occupied for the same time period as in data read access (3 + n clocks). When data access to this area is executed after data write access, more clocks than indicated in the table may be required.

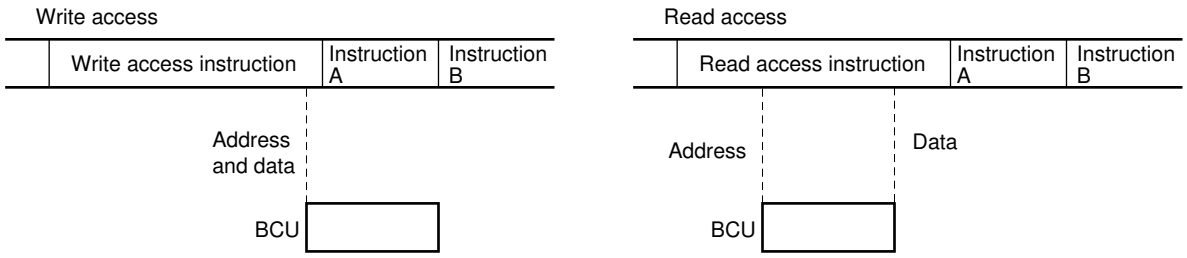
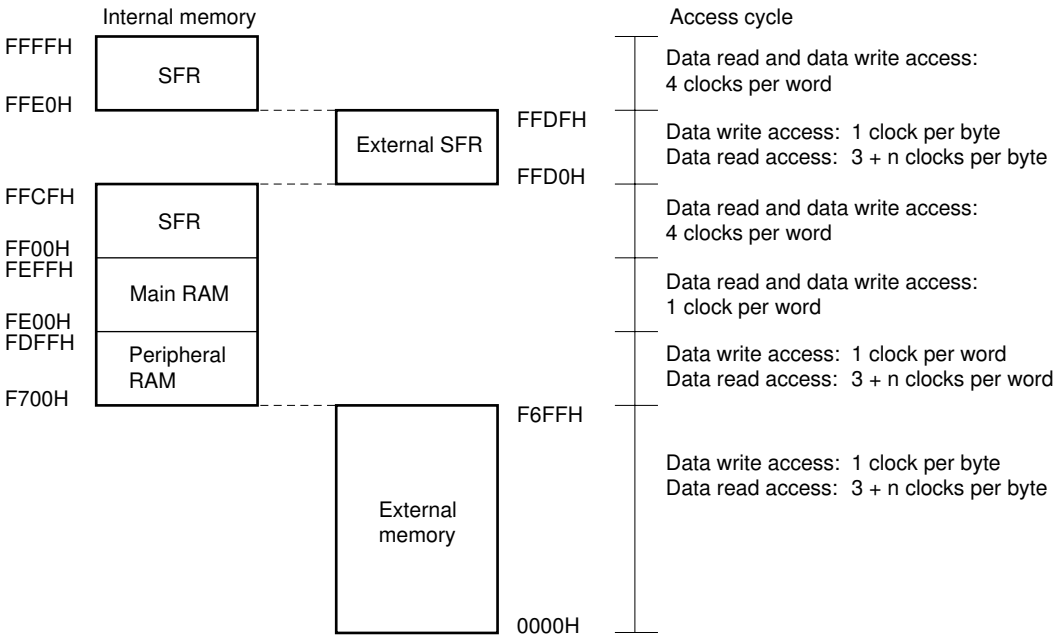
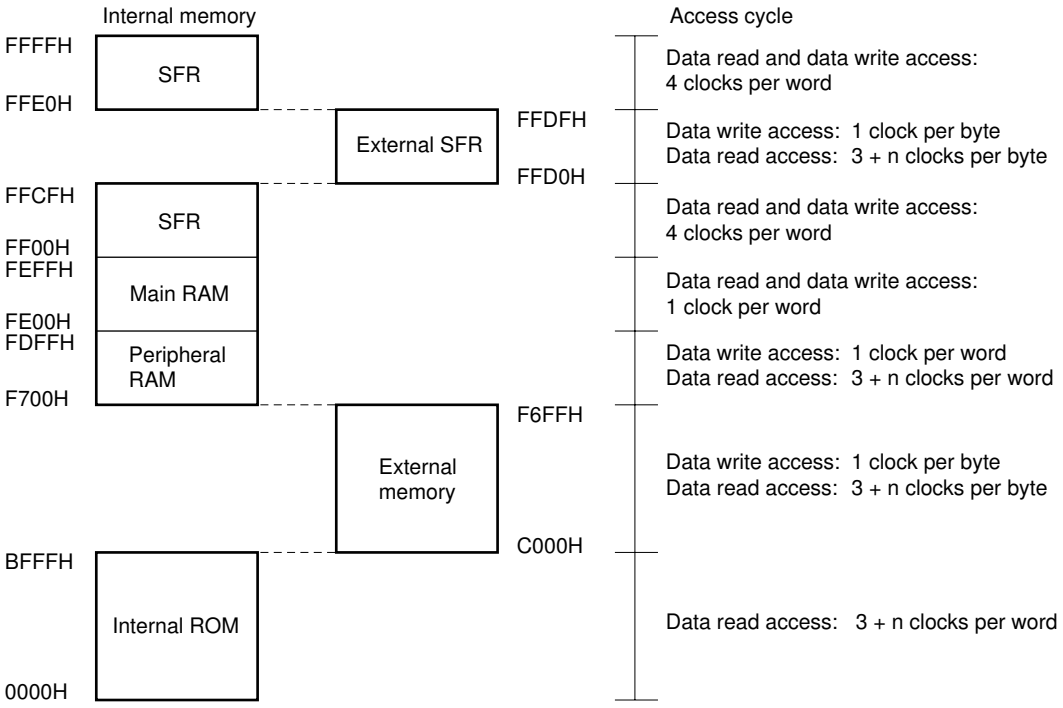


Fig. 21-2. Concept of Memory Access in Data Access

(1) μPD78355



(2) μPD78356 and μPD78P356



Caution When executing word access (including stack manipulation) in the main RAM area (FE00H to FEFFH), only even addresses can be specified with the operand.

Remark n is the wait count specified by the PWC register.

(c) Bus control signals for memory access (ASTB, $\overline{RD}$, $\overline{LWR}$, $\overline{HWR}$)

When data is accessed, the μPD78356 outputs the ASTB, $\overline{RD}$, $\overline{LWR}$, and/or $\overline{HWR}$ bus control signals. The signals to be output vary according to the area accessed. (See **Tables 21-4**, **21-5**.)

Table 21-4. Bus Control Signals in Data Access (When Specified External 8-bit Bus)

(1) μPD78355

Access area	Address	ASTB	$\overline{RD}$	$\overline{LWR}$	AD0 to AD7	A8 to A15
SFR	FFE0H to FFFFH FF00H to FFCFH	○	–	–	Address/ data	Address
External SFR	FFD0H to FFDFH	○	○	○	Address/ data	Address
Main RAM	FE00H to FEFFH	–	–	–	Undefined	Undefined
Peripheral RAM	F700H to FDFFH	○	–	–	Undefined	Undefined
External memory	0000H to F6FFH	○	○	○	Address/ data	Address

Remark ○ : Outputs the signal.
– : Does not output the signal (outputs the inactive level).

(2) μPD78356 and μPD78P356

Access area	Address	ASTB	$\overline{RD}$	$\overline{LWR}$	AD0 to AD7	A8 to A15
SFR	FFE0H to FFFFH FF00H to FFCFH	○	–	–	Address/ data	Address
External SFR	FFD0H to FFDFH	○	○	○	Address/ data	Address
Main RAM	FE00H to FEFFH	–	–	–	Undefined	Undefined
Peripheral RAM	F700H to FDFFH	○	–	–	Undefined	Undefined
External memory	C000H to F6FFH	○	○	○	Address/ data	Address
Internal ROM	0000H to BFFFH	○	–	–	Undefined	Undefined

Remark ○ : Outputs the signal.
– : Does not output the signal (outputs the inactive level).

Table 21-5. Bus Control Signals in Data Access (When Specified External 16-bit Bus)

(1) μPD78355

Access area	Address	ASTB	$\overline{RD}$	$\overline{LWR}$	$\overline{HWR}$	AD0 to AD7	AD8 to AD15
SFR	FFE0H to FFFFH FF00H to FFCFH	○	–	–	–	Address/ data	Address/ data
External SFR	FFD0H to FFDFH	○	○	○	○	Address/ data	Address/ undefined
Main RAM	FE00H to FEFFH	–	–	–	–	Undefined	Undefined
Peripheral RAM	F700H to FDFFH	○	–	–	–	Undefined	Undefined
External memory	0000H to F6FFH	○	○	○	○	Address/ data	Address/ data

Remark ○ : Outputs the signal.
– : Does not output the signal (outputs the inactive level).

(2) μPD78356 and μPD78P356

Access area	Address	ASTB	$\overline{RD}$	$\overline{LWR}$	$\overline{HWR}$	AD0 to AD7	AD8 to AD15
SFR	FFE0H to FFFFH FF00H to FFCFH	○	–	–	–	Address/ data	Address/ data
External SFR	FFD0H to FFDFH	○	○	○	○	Address/ data	Address/ data
Main RAM	FE00H to FEFFH	–	–	–	–	Undefined	Undefined
Peripheral RAM	F700H to FDFFH	○	–	–	–	Undefined	Undefined
External memory	C000H to F6FFH	○	○	○	○	Address/ data	Address/ data
Internal ROM	0000H to BFFFH	○	–	–	–	Undefined	Undefined

Remark ○ : Outputs the signal.
– : Does not output the signal (outputs the inactive level).

21.2 Interrupt Execution Rate

The following tables list interrupt execution rates. The time required to determine the priority is ignored in the tables. The priority is determined every two clocks. The time required to determine the priority ranges from zero to two clocks, depending on when the interrupt occurs.

n is the wait count specified by the PWC register.

(1) Vectored interrupt handling

Stack	Number of clocks
Main RAM (FE00H to FEFFH)	21 + 2n
Peripheral RAM (F700H to FDFFH)	25 + 2n/33 + 2n[Note]
External memory	33 + 6n

Note Values before slash are for even addresses.
Values after slash are for odd addresses.

(2) Context switching

Number of clocks: 17 + 2n clocks

(3) Macro service

Macro service		Number of clocks	
		Byte operation	Word operation
EVTCNT		12	
BLKTRS mem -> SFR		18	19
BLKTRS SFR -> mem		17	18
BLKTRS-P mem -> SFR	(IRAM)	20	21
	(PRAM)	22	23/27[Note]
	(EMEM)	22 + n	27 + 2n
BLKTRS-P SFR -> mem	(IRAM)	20	21
	(PRAM)	22	23/27[Note]
	(EMEM)	22 + n	27 + 2n
DTADIF		–	22
DTADIF-P	(IRAM)	–	26
	(PRAM)		28/32[Note]
	(EMEM)		32 + 2n

Note Values before slash are for even addresses.
Values after slash are for odd addresses.

21.3 Calculating the Number of Execution Clocks

This section describes the procedure for calculating the number of instruction execution clocks.

(1) Calculating the number of basic clocks

(a) When a program is held in internal ROM or peripheral RAM (F700H to FDFFH) in the high-speed fetch mode
The total number of clocks is the number of basic clocks.

(b) When a program is usually held in external memory in the normal fetch mode
The larger value of the following two is taken as the number of basic clocks:
- Total number of clocks
- Total number of bytes of executed instructions x (3 + n)
  n is the wait count specified by the PWC register.

When the high-speed fetch mode is not specified, the number of basic clocks is calculated according to procedure (b) above even if the program is in internal ROM or peripheral RAM.

Example of calculation: μPD78356 (Internal clock of 16 MHz, wait 0, main RAM specified by [HL])

		Number of bytes	Number of clocks
MOV	A, [HL]	1	6
ADD	A, B	2	3
MOV	[HL], A	1	5
		4	14

The time to execute this program is calculated as follows: However, when the program is in the external memory, the calculation is when the external 8-bit is specified.

(a) When the program is held in internal ROM
Number of execution clocks: 14
14 x 0.0625 μs = 0.875 μs

(b) When the program is held in external memory
Number of execution clocks: 4 x 3 = 12 < 14
Taking the number of execution clocks as 14, the calculation shown in (a) above is performed. The calculated execution time is 0.875 μs.

(2) Adding a correction factor to the number of basic clocks

(a) Accessing the SFR area (FF00H to FF1FH) by an instruction having saddr or saddrp as an operand

Each time an access is made, four clocks are added.

Table 21-6. Number of saddr Accesses by Instruction (1/2)

Instruction		Number of accesses
MOV	saddr, #byte	1
	A, saddr	
	saddr, A	
	saddr, saddr	1/2[Note 2]
	A, [saddrp]	1
	[saddrp], A	
XCH	A, saddr	2
	A, [saddrp]	1
	saddr, saddr	2/4[Note 2]
MOVW	saddr, #word	1
	AX, saddrp	
	saddrp, AX	
	saddrp, saddrp	1/2[Note 2]
XCHW	AX, saddrp	2
	saddrp, saddrp	2/4[Note 2]
ALU[Note 1]	saddr, #byte	2
	A, saddr	1
	saddr-D, saddr-S	1/2/3[Note 3]
CMP	saddr, #byte	1
	A, saddr	
	saddr, saddr	1/2[Note 2]

Notes 1. ALU: ADD, ADDC, SUB, SUBC, AND, OR, XOR

2. When either or both are in the SFR area

3. When only the source (saddr-S), only the destination (saddr-D), or both are in the SFR area

Table 21-6. Number of saddr Accesses by Instruction (2/2)

Instruction		Number of accesses
ADDW, SUBW	saddrp, #word	2
	AX, saddrp	1
	saddrp-D, saddrp-S	1/2/3[Note 1]
CMPW	saddrp, #word	1
	AX, saddrp	
	saddrp-D, saddrp-S	1/2[Note 2]
INC, DEC	saddr	2
INCW, DECW	saddrp	2
MOV1	CY, saddr.bit	1
	saddr.bit, CY	2
AND1, OR1	CY, saddr.bit	1
	CY, /saddr.bit	
XOR1	CY, saddr.bit	1
SET1, CLR1, NOT1	saddr.bit	2
BT, BF	saddr.bit, $addr16	1
BTCLR, BFSET	saddr.bit, $addr16	2
DBNZ	saddr.bit, $addr16	2

Notes 1. When only the source (saddr-S), only the destination (saddr-D), or both are in the SFR area

2. When either or both are in the SFR area

(b) Accessing the following SFR area by an instruction having sfr or sfrp as an operand

Correction factor = Number of clocks to be increased per access x Number of accesses

SFR to be accessed	Number of clocks to be increased per access	
	Read access	Write access
TM0	1 to 2	–
TM1, TM2	1 to 4	–
TM3	1 to 8	–
UDC	1 to 8	0 to 7
CM00, CM01, CM02, CM03, CM10, CM11, CM20, CM21, CC00, CC01, CC02	1 to 2	0 to 1
CMUD0, CMUD1, CC30, CC31	1 to 6	0 to 5
TMC0, TMC1	0	0 to 1
UDCC	0	0 to 3
OVIC0, OVIC3, PIC0, PIC1, PIC2, PIC3, PIC4, CMIC00, CMIC01, CMIC02, CMIC03, CMIC10, CMIC11, CMIC20, CMIC21, CMIC40, CMICUD0, CMICUD1, SERIC, SRIC, STIC, CSIIC0, CSIIC1, ADIC, IMC, MK0L MK0, MK0H, MK1L, MK1, ISPR	2	1
External SFR (FFD0H to FFDFH)	n/0[Note]	
PWC	0	1

Note Normal fetch/high-speed fetch

Remark n is the wait count specified by the PWC register.

The number of clocks to be increased depends on the state of the peripheral hardware at the time of access and varies within the range shown in the table above.

Table 21-7. Number of sfr Accesses by an Instruction

Instruction		Number of accesses	
		Read access	Write access
MOV	sfr, #byte	0	1
	A, sfr	1	0
	sfr, A	0	1
XCH	A, sfr	1	1
MOVW	sfrp, #word	0	1
	AX, sfrp	1	0
	sfrp, AX	0	1
XCHW	AX, sfrp	1	1
ALU[Note]	sfr, #byte	1	1
	A, sfr	1	0
CMP	sfr, #byte	1	0
	A, sfr		
ADDW, SUBW	sfrp, #word	1	1
	AX, sfrp	1	0
CMPW	sfrp, #word	1	0
	AX, sfrp		
MOV1	CY, sfr.bit	1	0
	sfr.bit, CY	1	1
AND1, OR1	CY, sfr.bit	1	0
	CY, /sfr.bit		
XOR1	CY, sfr.bit	1	0
SET1, CRL1, NOT1	sfr.bit	1	1
BT, BF	sfr.bit, $addr16	1	0
BTCLR, BFSET	sfr.bit, $addr16	1	1

Note ALU: ADD, ADDC, SUB, SUBC, AND, OR, XOR

(c) Branch instruction and CALL instruction

A branch or CALL instruction clears the instruction queue. Before the instruction following the branch instruction is executed, the additional clocks shown below are required:

In the high-speed fetch mode : 7 to 8 clocks
In the normal fetch mode : 7 to 10 + n clocks

The number of clocks to be added depends on when op-code fetch is executed and varies in the range shown above.

CHAPTER 22 CAUTIONS

This chapter collects the cautions described in each chapter. Read this chapter before developing application products. The number enclosed in parentheses represents the page on which the caution is initially described.

22.1 Cautions for Chapter 2

(1) When the $\overline{\text{RESET}}$ signal is input, all the pins of P00 to P07 (port 0), P10 to P17 (port 1), P20 to P27 (port 2), P30 to P37 (port 3), P80 to P87 (port 8), and P100 to 107 (port 10) are set to the input mode (output high to impedance). The contents of the output latch becomes undefined. (p. 21-23, 26, 28)

(2) When the $\overline{\text{RESET}}$ signal is input, pins P40 to P47 (port 4), P50 to P57 (port 5), and P90 to P93 (port 9) are set to the input mode (output high-impedance) in either port or external memory expansion mode. The contents of the output latch becomes undefined. (p. 24, 25, 27)

(3) Connect the MODE0 and MODE1 pins directly to V_{DD} or V_{SS}. (p. 29)

22.2 Cautions for Chapter 3

(1) When executing word access (including stack manipulation) in the main RAM area (FE00H to FEFFH), only even addresses can be specified with the operand. (p. 36, 50, 61)

(2) When executing word access (including stack manipulation) in the main RAM area (FE00H to FEFFH), access operations will differ according to if the reference address is even or odd. If both even and odd addresses are accessed together, operations may not be performed correctly. Therefore, specify only even addresses for the reference address. When executing the 16-bit data transfer instruction, specify even addresses with the operand. If odd addresses are specified, errors will generate in the assembler package (RA78K/III). (p. 39)

(3) Do not perform word access covering both the peripheral RAM area and main RAM area. (p. 39)

(4) Unmapped addresses of the special function register cannot be accessed (except the external access area.) (p. 43)

(5) Pins P50/AD8 to P57/AD15 function as address buses when 8-bit external buses are specified. (p. 43)

(6) The vector tables for the reset input, BRK instruction, and op-code trap interrupt are fixed to 0000H, 003EH, and 003CH, respectively. They are not affected by the TPF. (p. 50)

(7) Write 0 or 1 into any bit of the special function register (SFR) correctly whenever it is predetermined to be so. (p. 54)

(8) Do not write data into the register which is only used for data reading. Writing data into such registers may result in an error. (p. 54)

(9) The SFR area (FF00H to FFFFH) addresses to which a special function register is not assigned cannot be accessed (except the external access area). Accessing these addresses may result in an error. (p 54)

22.3 Cautions for Chapter 5

(1) For the μPD78355 and for the μPD78356 and μPD78P356 in the ROM-less mode, ports 4, 5, and 9 (low-order 3 bits) do not function as ports. (p. 69)

(2) Pins functioning as input pins in the control mode may not operate correctly if corresponding bits of the port mode control register are rewritten during operations. therefore, write in the port mode control register during the initial settings of the system. (p. 73)

(3) If the port read control register (PRDC) is in the pin access mode (PRDC0 = 1), no bit manipulation instruction for a port operates normally. After a port check is completed, be sure to reset the mode to the normal mode (PRDC0 = 0). (p. 75)

(4) If an interrupt occurs when the PRDC register is in the pin access mode, a bit manipulation instruction may be executed in the same mode. This will cause an error. Before starting a check, be sure to set the DI state. In addition, do not use macro services that manipulate ports. (p. 75)

(5) Nonmaskable interrupts are unavoidable if the PRDC register is in the pin access mode. So, the following provisions should be made in the program as required: (p. 75)

- The nonmaskable interrupt routine is to perform no port manipulation.
- The level of PRDC.0 is to be saved at the start of the nonmaskable interrupt routine, then is restored when control is returned.

(6) Use the CHKL or CHKLA instruction only when the PRDC0 bit of the PRDC register is set to 0 (normal mode). (p. 76)

(7) For input/output port pins set to the input port mode, the results of the CHKL or CHKLA instruction is always found to agree with the control output level, regardless of the setting of the port mode/control mode.
For the input dedicated port, as there is not output latch, the input pin level is read when the CHKL or CHKLA instruction is executed. Therefore, as the CHKL and CHKLA instructions for the input dedicated port are essentially ineffective, do not use them. (p. 76)

(8) Set control output pins to the input mode before executing the CHKL or CHKLA instruction to check the output level of the pin of a port where control and port output pins are used together. (The output level of a control pin cannot be checked with the CHKL or CHKLA instruction because the output level varies asynchronously.) (p. 76)

(9) Bit 0 of the port 2 mode register (PM2) is always one. (p. 80)

(10) Bit 0 of the port 2 mode control register (PMC2) is always one. (p. 85)

(11) Be sure to write 0 in bits 4 and 7 of the memory expansion mode register (MM). If 1 is written in these bits, operation will not be normal. (p. 89)

(12) Invalid combinations are specified as “Not to be set” in Fig. 5-21. Never write these combinations. (p. 89)

(13) In μPD78356 emulation using the IE-78350-R, the use of the built-in, pull-up resistors of port 1, 4, 5, or 9 is disabled even if the PUO1, PUO4, PUO5, or PUO9 bit of the PUOL and PUOH registers is set to 1. When using a pull-up resistor, first set the corresponding bit to 1 to use software common to the IE-78350-R and μPD78356, then externally attach a pull-up resistor. (p. 90)

22.4 Cautions for Chapter 6

(1) When using an external clock, do not set the STP bit of the standby control register (STBC) to 1. (p. 93)

(2) When using the system clock oscillator, run wires according to the following rules to avoid effects such as stray capacitance: (p. 94)

- Minimize the wiring.
- Never cause the wires to cross other signal lines or run near a line carrying a large varying current.
- Cause the grounding point of the capacitor of the oscillator circuit to have the same potential as V_{SS}. Never connect the capacitor to a ground pattern carrying a large current.
- Never extract a signal from the oscillator.

(3) When applying external clocks, do not connect a wire or a similar capacitive load to the X2 pin. (p. 94)

22.5 Cautions for Chapter 7

(1) When the compare register function is selected, the capture/compare registers (CC00 to CC02, CC30, and CC31) cannot generate interrupts based on the corresponding external interrupt (capture trigger signal). (p. 105)

(2) Before changing the timer output mode (setting of the TOM00, TOM02, or TOM04 bit) with the TUM0 register, disable the timer output for the corresponding timer output pin. (p. 107)

(3) Timer output (TO21) is impossible when a clear operation by a TCLR2 pin input is enabled by the TUM1 register. (p. 107)

(4) When the operating mode for the UDC is specified as mode 4 by the UDCC register, the specification of an edge for the TIUD pin (by the TUM2 register) becomes ineffective. (p. 108, 113, 148, 151)

(5) Timer output (TO03) is impossible when the input of an external count clock to TM0 (TI0) is specified by the TMC0 register. Timer output (TO11) is impossible when the input of an external count clock to TM1 (TI1) is specified by the TMC0 register. (p. 110)

(6) Before changing the active state (setting of the ALV00, ALV01, ALV02, or ALV03 bit) with the TOC0 register, disable the timer output for the corresponding timer output pin. (p. 115)

(7) Before changing the active state (setting of the ALV04, ALV05, ALV10, or ALV11 bit) with the TOC1 register, disable the timer output for the corresponding timer output pin. (p. 117)

(8) When the timer outputs for the TO04 and TO05 pins are disabled by the TOC1 register, external interrupt request inputs (INTP0 and INTP1) are assumed for these pins. (p. 117)

(9) Before changing the active state (setting of the ALV10 or ALV20 bit) with the TOC2 register, disable the timer output for the corresponding timer output pin. (p. 118)

(10) Each flag of TOVS is reset in a different way. Only software can reset the OVF1, OVF2, OVFUD, and UDFUD bits. Reset these flags after testing them.
The OVF0 and OVF3 bits are read-only bits. These flags are cleared to 0 when the interrupt request flags (OVIF0 and OVIF3) of the interrupt control registers (OVIC0 and OVIC3) are reset to 0 or TM0 and TM3 overflow interrupts (INTOV0 and INTOV3) are received. (p. 120)

(11) Before changing the noise elimination time (the setting of the NI0, NR0, NI1, NR1, NR2, NIUD, NCUD, or NRUD bit) with the NPC register, disable the counting of the corresponding timer. (p. 123)

(12) If the count clock for timer 0 (TM0) is specified as the external clock (TI0 input) and count operations are performed, when bit 1 (PMC81) of the port 8 mode control register (PMC8) is rewritten from 0 to 1 or vice versa, the TM0 may start counting up. Therefore, perform PMC81 bit rewriting after stopping the TM0 count operations first. (p. 124)

(13) When 00H is set in compare register CM02, CM10, CM20, or CM40 in the interval timer mode, TM0 performs interval operation at each count clock input.
The interval time at the first match is different from that at the second or subsequent matches. (p. 127, 133, 137, 143)

- Time until the first match : Value of the compare register x count clock period
- Time until the second match or subsequent matches : (Value of the compare register + 1) x count clock period

(14) If the count clock for timer 1 (TM1) is specified as the external clock (TI1 input) and count operations are performed, when bit 6 (PMC36) of the port 3 mode control register (PMC3) is rewritten from 0 to 1 or vice versa, the TM1 may start counting up. Therefore, perform PMC36 bit rewriting after stopping the TM1 count operations first. (p. 130)

(15) If clear operations by the TCLR2 pin input are enabled, when bit 6 (PMC26) of the port 2 mode control register (PMC2) is rewritten from 0 to 1 or vice versa, TM2 may be cleared. Therefore, rewrite the PMC26 bit after stopping the TM2 count operations first. (p. 134)

(16) When 00H is set in capture/compare register CC30 in the interval timer mode, TM3 performs interval operation at each count clock input.
The interval time at the first match is different from that at the second or subsequent matches. (p. 140)

- Time until the first match : Value of the capture/compare register x count clock period
- Time until the second match or subsequent matches : (Value of the capture/compare register + 1) x count clock period

(17) If timer output by the TO04 and TO05 pins are enabled, when the output level is changed, the interrupt request flag (PIC0.7, PIC1.7) is set (1). Therefore, when using as the timer output, preset the interrupt mask flag (PIC0.6, PIC1.6) to "1" and set the interrupt mask state. (p. 152)

(18) 0 is always read when the RTPM register specifies data of the RTPH and RTPL registers that is not connected to any pin. (p. 158)

(19) When a software trigger is specified as an output trigger to the real-time output port, the contents of RTPH and RTPL are output to the pins. Before specifying a software trigger, write data in RTPH and RTPL. (p. 158)

22.6 Cautions for Chapter 8

(1) The A/D trigger mode, timer trigger mode, or external trigger mode can be specified for each pin ANI0 to ANI3. However, the A/D trigger mode is always specified for the ANI4 to ANI7 pins. (p. 159)

(2) Apply the reference voltage to the ANI0 to ANI7 pins. If a voltage in the range of V_{DD} to V_{SS} inclusive (even in the range of the absolute maximum ratings) is applied to these pins, the converted values of the channels are undefined, and the converted values of other channels may also be affected. (p. 161)

(3) The order in which the ANI0 to ANI3 bits are scanned is specified according to the order in which the match signals from compare registers or external trigger signals are generated in the timer trigger mode or external trigger mode for the scan mode. As a result, the ANIS2 to ANIS0 bits specify no analog input pin, they specify the number of trigger inputs. (p. 165)

(4) Apply a voltage in the range of AV_{SS} to AV_{DD} to the pins used as the input pins of the A/D converter. (p. 171)

(5) When directly using the A/D conversion results to perform branching, if programs which branch only when the conversion results become the specified value are created, the conversion results will not become the specified value due to the effects of conversion difference and operations will not branch to the specified routine. Therefore, develop programs that will branch when the conversion results are in ranges that include the width for overall error difference. (p. 174)

(6) Before specifying the timer trigger mode, specify that the capture/compare registers (CC00 and CC01) function as a compare register. (p. 181)

(7) The timer trigger mode is effective for ANI0 to ANI3 analog inputs only. (p. 181)

(8) When the next trigger signal is input during A/D conversion in the timer trigger mode or external trigger mode, the conversion is stopped and an A/D conversion corresponding to the new trigger signal is performed. The result of the stopped A/D conversion is undefined. (p. 182, 183, 185, 186, 190, 191, 195, 197, 198, 200, 202, 204)

(9) When several triggers are input simultaneously, the smallest analog input of the analog input pin (ANIn) number is converted. Other trigger signals input at the same time are ignored and the number of triggers input is not counted. Therefore, the INTAD interrupt is not generated and the A/D conversion results become undefined. (p. 185, 186, 190, 191, 195, 198, 200, 204)

(10) When the following modes are selected, only a single INTCM00 interrupt occurs in the one-shot mode for timer 0. No A/D conversion is therefore terminated. To prevent this, be sure to put timer 0 in the loop mode. (p. 188, 193)

- Select mode + four-buffer mode + one-trigger mode
- Scan mode + one-trigger mode

(11) When an INTCM00 interrupt occurs during A/D conversion of the analog inputs in the higher four channels (ANI4 to ANI7), the conversion is stopped and the AN0 channel and subsequent channels are scanned again. The results of the stopped A/D conversions are undefined. (p. 193)

(12) The external trigger mode is effective for analog inputs ANI0 to ANI3 only. (p. 196)

(13) When bits 0 to 3 (PMC00 to PMC03) of the port 0 mode control register (PMC0) are rewritten from 0 to 1 or vice versa during A/D conversion in the external trigger mode, external triggers may be generated. Therefore, rewrite the PMC00 to PMC03 bits after stopping the A/D converter operations first. (p. 196)

(14) When an ADTRG0 signal is input during A/D conversion of the analog inputs in the higher four channels (ANI4 to ANI7), the conversion is stopped and the AN0 channel and subsequent channels are scanned again. The results of the stopped A/D conversions are undefined. (p. 202)

(15) When ADTRG0 to ADTRG3 signals are input during A/D conversion of analog inputs in the higher four channels (ANI4 to ANI7), the conversion is stopped and the ANI0 channel and subsequent channels are scanned again. The results of the stopped A/D conversions are undefined. (p. 204)

22.7 Cautions for Chapter 9

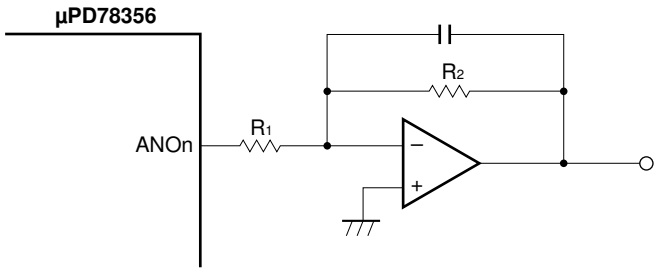
(1) The ANOn pin (n = 0 or 1) cannot source a current due to the high output impedance of the D/A converters. When connecting a load with a low input impedance, insert a buffer amplifier between the load and ANOn pin. Keep the wire to the buffer amplifier or load as short as possible (because the output impedance is high). If this is impossible, surround the wire with the ground pattern. (p. 211)

(2) The output voltage of each D/A converter changes step by step. In general, use the signal output from a D/A converter after passing it through a low-pass filter. (p. 211)

(3) The D/A converters contained in the μPD78356 have high-impedance output while the $\overline{\text{RESET}}$ signal input is low. Arrange the load circuit so that it can accept high-impedance input. (p. 211)

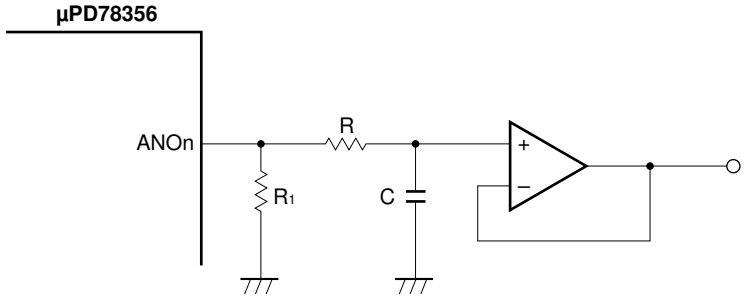
Fig. 22-1. Examples of Buffer Amplifier Connection

(a) Inverting amplifier



• Input impedance of the buffer amplifier is $R_1 + R_2$.

(b) Voltage follower



• Input impedance of the buffer amplifier is R_1.
• Be sure to connect R_1. Otherwise, the output when $\overline{\text{RESET}}$ is low level becomes undefined.

(4) After a reset is released, the D/A converter outputs the voltage whose level is the same as the AV_{REF3} pin. Arrange the D/A converter output so that the converter outputs the same voltage as the voltage on the AV_{REF3} pin. (p. 211)

22.8 Cautions for Chapter 10

(1) Usually, the transmission completion interrupt (INTST) occurs whenever the TXS register becomes empty. However, even when the $\overline{\text{RESET}}$ signal is input and TXS becomes empty, INTST does not occur. (p. 227, 228)

(2) Data written to the TXS register during transmission (before INTST occurs) is regarded as invalid. (p. 227)

(3) In the event of a reception error, be sure to read the RXB data. Otherwise, an overrun error occurs when the next data block is received, leading to an endless error condition. (p. 230, 234)

(4) The ASIS register is reset to 0 when the RXB buffer is read or the next data block is received. To find out about the error, be sure to read the ASIS register before reading the RXB buffer.
When received data is being transferred to the memory using macro service, the receive buffer (RXB) may be read during serial data reception and as a result, the ASIS register may be reset (0). So, it is only detected that an error occurs. Before use, make sure that this will not cause any serious problems.
An error can be detected by the setting condition (1) of the reception error interrupt request flag (SERIF) or the reception of the reception error interrupt (INTSER). (p. 234)

22.9 Cautions for Chapter 11

(1) Do not switch serial clocks during data transmission. Since the switching is performed asynchronously with the serial clock being supplied, doing this during data transmission can produce a serial clock of undefined frequency. (p. 242)

(2) Since the serial data bus pins (SB0 and SB1) employ an open drain configuration for output in the SBI mode, the serial data bus line is provided in a wired OR state and therefore requires a pull-up resistor. (p. 259)

(3) To switch between the master and slave CPUs in the SBI mode, a pull-up resistor is also needed for the $\overline{SCK}$ line. This is because the $\overline{SCK}$ input/output switching is performed in the master and slave CPUs asynchronously. (p. 259)

(4) Do not operate the RELT or CMDT bit of the serial bus interface control register (SBIC) during data transmission or reception. (p. 268)

(5) When transferring data in the SBI mode, be sure to specify the pin and serial clock before setting the CTXE0 and CRXE0 bits. (p. 272)

22.10 Cautions for Chapter 12

Do not switch serial clocks during data transmission. Since the switching is performed asynchronously with the serial clock being supplied, doing this during data transmission can produce a serial clock of undefined frequency. (p. 287)

22.11 Cautions for Chapter 13

Byte access/bit access can be performed for the lower side of the PWM0/PWM1 register but not for the higher side. (p. 304)

22.12 Cautions for Chapter 14

(1) Data can be written into the WDM register only by a dedicated instruction (MOV WDM, #byte). (p. 309)

(2) Do not change the priority for interrupt requests specified in the WDM register dynamically, that is, when a program is being executed. (p. 309)

(3) The RUN bit of the watchdog timer mode register (WDM) cannot be reset to 0 by software. (p. 309)

(4) The count clock is not reset even when the watchdog timer is cleared by setting the RUN bit of the WDM register to 1. (p. 309)

(5) Immediately after the power is turned on, the watchdog timer output pin ($\overline{WDTO}$) may go low for a period of up to 32 clocks. (p. 310)

22.13 Cautions for Chapter 15

(1) The in-service priority register (ISPR) can be accessed for read only. Write to the register may cause an error. (p. 327)

(2) Macro service requests are accepted and handled even during the nonmaskable interrupt handling routine. If macro service handling is not to be performed during this routine, manipulate the interrupt mask flag register during the routine so that the macro service is not generated. (p. 332)

(3) Always use the RETI instruction to restore from the nonmaskable interrupt. Interrupts will not be accpeted properly hereafter of other instructions are used. (p. 332)

(4) The nonmaskable interrupt is always accepted except for while the nonmaskable interrupt handling is being executed (excludes when a high priority nonmaskable interrupt request is generated during the execution of the low priority nonmaskable interrupt handling) and during a certain period of time after the execution of special instructions shown in **15.9**. Therefore, nonmaskable interrupt is accepted even if stack pointer values after reset release, etc. are undefined. In such cases, depending on the stack pointer value, program counter (PC) and program status words (PSW) may be written in addresses (See **Table 3-4** in **3.2.3**.) which disable the writing of special function registers and cause the CPU to deadlock and unexpected signals to be output from terminals, or PC and PSW may be written in addresses not mounted with the RAM, and as a result, the main routine cannot be returned to normally from the nonmaskable interrupt handling routine and the CPU overruns.
Therefore, always set programs after the $\overline{\text{RESET}}$ release to as follows. (p. 332)

```
        CSEG  AT 0
        DW    STRT
STRT:
        MOVW SP, #imm16
```

(5) When the maskable interrupt is accepted using the vectored interrupt, always restore using the RETI instruction. Interrupt operations will not be performed properly hereafter if other instructions are used. (p. 335)

(6) Always use the RETCS instruction to restore from interrupts by context switching. Interrupt operations will not be performed properly hereafter is other instructions are used. (p. 336)

(7) Do not use the RETI instruction to restore from the software interrupt by the BRK instruction. (p. 343)

(8) When the context switching function is started up by executing the BRKCS instruction, if the bit of the ISPR register is reset (0) following the execution of the RETCS instruction, the interrupt nesting control may destroyed.
Therefore, always use the RETCSB instruction to restore from the handling started up by the BRKCS instruction. (p. 345)

(9) When transmitting data by the UART using macro service, two vectored interrupt requests occur. (p. 352) ★

(10) Word buffers must be placed at even addresses in the block transfer mode. (p. 358)

(11) Word buffers must be placed at even addresses in the block transfer mode (with a memory pointer). (p. 360)

(12) MEM.PTR must be placed at an even address in the block transfer mode (with a memory pointer). (p. 360)

(13) 00H must not be set in the MSC in the data difference mode. (p. 362)

(14) Buffers must be placed at even addresses in the data difference mode. (p. 362)

(15) The “previous value” must be initialized to dummy data in advance in the data difference mode. (p. 362)

(16) The SFRP can specify 16-bit SFRs only in the data difference mode. (p. 362)

(17) 00H must not be set in the MSC in the data difference mode (with a memory pointer). (p. 363)

(18) Buffers must be placed at even addresses in the data difference mode (with a memory pointer). (p. 363)

(19) The MEM.PTR must be placed at an even address in the data difference mode (with a memory pointer). (p. 363)

(20) The “previous value” must be initialized to dummy data in advance in the data difference mode (with a memory pointer). (p. 363)

(21) The SFRP can specify only 16-bit SFRs in the data difference mode (with a memory pointer). (p. 363)

(22) When polling interrupt related registers using BRCLR instructions, make sure one is not branched to the same instruction. If programs in which instructions are branched to themselves are written, all interrupts and macro services are held until conditions with which the instructions will not branch to themselves are established. (p. 365)

Wrong example

```
        ⋮
LOOP:  BTCLR PIC0.7, $LOOP     All interrupts and macro services are held until PIC0.7 becomes 1.
       x x x                   <- The interrupts and macro services are handled first after the instruction after the BTCLR instruction is executed.
        ⋮
```

Correct example (1)

```
        ⋮
LOOP:  NOP
       BTCLR PIC0.7, $LOOP     <- As the interrupts and macro services are handled after the execution of the NOP instruction, they will not be held for a long period of time.
        ⋮
```

Correct example (2)

```
        ⋮
LOOP:  BTCLR PIC0.7, $NEXT
       BR $LOOP                <- As the interrupts and macro services are handled after the execution of the BR instruction, they will not be held for a long period of time.
NEXT:   ⋮
```

(23) If the above instructions are to be used continuously due to the same reasons as (22), and the interrupts and macro services are going to be held for a long time, insert NOP instructions, etc. halfway and create timings with which the interrupts and macro services can be accepted. (p. 366)

22.14 Cautions for Chapter 16

(1) After the SBF flag in the standby control register (STBC) is read, set it to 1. Software can then discriminate a power-on reset from release of the STOP or HALT mode. (p. 370)

(2) When an external clock is used, do not set the STP bit of the standby control register (STBC) to 1. (p. 371)

(3) If the interrupt request flag (xxIF) is set to 1 and the interrupt is not masked (xxMK = 0), the system does not enter the HALT mode. When macro service processing (xxISM = 1) is performed, the system enters the HALT mode after the macro service terminates. (p. 372)

(4) Data can be written into the WDM register only by a dedicated instruction (MOV WDM, #byte). (p. 377)

(5) Do not change the priority for interrupt requests specified in the WDM register dynamically, that is, when a program is being executed. (p. 377)

(6) The RUN bit of the watchdog timer mode register (WDM) cannot be reset to 0 by software. (p. 377)

(7) The count clock is not reset even when the watchdog timer is cleared by setting the RUN bit of the WDM register to 1. (p. 377)

22.15 Cautions for Chapter 17

(1) When $\overline{\text{RESET}}$ is active, all pins except $\overline{\text{WDTO}}$, CLKOUT, AV_{REF}, AV_{DD}, AV_{SS}, V_{DD}, V_{SS}, X1, and X2 go into the high-impedance state. (p. 385)

(2) When RAM is expanded externally, attach a pull-up resistor to the P90/$\overline{\text{RD}}$ pin, P91/$\overline{\text{LWR}}$ pin, and P92/$\overline{\text{HWR}}$ pin. Otherwise, these pins may go into the high-impedance state, and the contents of the external RAM may be lost. Or signal collision on the address/data bus may damage the input/output circuit. (p. 385)

22.16 Cautions for Chapter 18

(1) The internal memory (ROM, RAM) capacity of the μPD78356 can be changed by setting the memory expansion mode register (MM). (p. 391, 392)

(2) Be sure to write 0 in bits 4 and 7 of the memory expansion mode register (MM). If 1 is written in these bits, operation will not be normal. (p. 395)

(3) Invalid combinations are specified as “Not to be set” in Fig. 18-4. Never write these combinations. (p. 395)

(4) Cycle counts indicated in Fig. 18-5 do not contain any address wait. When an address wait is added, one cycle must be added to the indicated cycle count. (p. 398)

(5) Although instruction fetch and data access are allowed for the peripheral RAM area (F700H to FDFFH), the bus width specification and wait specification made by the PWC register are invalid for that area, and 16-bit bus operation is always performed. Every instruction fetch is performed in the high-speed fetch mode. (p. 398)

(6) Instruction fetches cannot be performed for the main RAM area (FE00H to FEFFH). Data is always accessed in 16-bit units; the bus width specification and wait specification made by the PWC register are invalid at the time of data access. (In this case, a special bus cycle made up of two clock cycles is activated.) (p. 398)

(7) The wait specification for the external SFR area (FFD0H to FFDFH) is made by the PWC7 and PWC6 bits. (p. 398)

(8) For the internal ROM area, 16-bit bus operation is performed regardless of the PWC register setting. The wait specification for this area can be made using the PWC register. Note that, however, specifying a 16-bit bus for the internal ROM area makes the P92 pin function as the $\overline{\text{HWR}}$ pin even when only an 8-bit external memory is connected. The port function is then disabled. So, never specify the 16-bit bus for the internal ROM area. (p. 398)

22.17 Cautions for Chapter 21

(1) When executing word access (including stack manipulation) in the main RAM area (FE00H to FEFFH), only even addresses can be specified with the operand. (p. 435, 437, 440)

(2) The PWC register specifies the mode of an area, irrespective of whether the area is in internal memory or external memory. After reset, the whole space is set in the normal fetch cycle mode. (p. 436)

APPENDIX A DIFFERENCES BETWEEN THE μPD78356 AND μPD78334

The following pages list the functions of the μPD78356 sub-series (μPD78355, 78356, 78P356) and μPD78334 sub-series (μPD78330, 78334, 78P334).

Functions (1/2)

Item \ Product		μPD78355	μPD78356	μPD78P356
Minimum instruciton execution		125 ns (Internal clock: 16 MHz operation, External clock: 32 MHz operation		
Internal memory	ROM	—	48 Kbytes	—
	PROM	—	—	48 Kbytes
	RAM	2 Kbytes		
Memory space		64 Kbytes (can externally be extended)		
General register		8 bits x 16 x 8 banks		
Number of basic instructions		115		
Instruction set		• 16-bit transfer or arithmetic/logical operation • Multiply/divide operation (16 bits x 16 bits, 32 bits ÷ 16 bits) • Bit manipulation • String • Sum-of-products operation (16 bits x 16 bits + 32 bits) • Correlation operation		
I/O line	Input	9 (8 ports for analog input)		
	I/O	48	67	
Real-time pulse unit		• 16-bit timers : 5 • 10-bit timer : 1 • 16-bit compare registers : 10 • 10-bit compare register : 1 • 16-bit capture/compare registers : 5 • Timer outputs : 10		
Real-time output port		Pulse outputs synchronized with real-time pulse unit: 8		
PWM unit		8/10/12-bit variable PWM outputs: 2 channels		
A/D converter		10-bit resolution: 8 channels		
D/A converter		8-bit resolution: 2 channels		
Serial interface		Serial interface with a dedicated baud rate generator UART: 1 Clock synchronous serial interface/SBI: 1 Clock synchronous serial interface (with pin switching function): 1		
Interrupt function		• External: 5, internal: 25 (5 pins are also used for external interrupts.) • Four levels of priorities can be specified according to software types. • One interrupt processing mode can be selected out of three types: vectored interrupt function, macro service function, and context switching function.		
Test factor		Not provided		
Bus sizing function		8-bit/16-bit external data bus width can be selected		
ECC circuit		Not provided		Provided
Package	Without a window	• 100-pin plastic QFP (14 x 14 mm) • 120-pin plastic QFP (28 x 28 mm)		
	With a window	—		• 120-pin ceramic WQFN
Others		• Watchdog timer function • Standby function (HALT/STOP)		

Functions (2/2)

Item \ Product		μPD78330	μPD78334	μPD78P334
Minimum instruciton execution		250 ns (Internal clock: 8 MHz operation, External clock: 16 MHz operation		
Internal memory	ROM	—	32 Kbytes	—
	PROM	—	—	32 Kbytes
	RAM	1 Kbytes		
Memory space		64 Kbytes (can externally be extended)		
General register		8 bits x 16 x 8 banks		
Number of basic instructions		111		
Instruction set		• 16-bit transfer or arithmetic/logical operation • Multiply/divide operation (16 bits x 16 bits, 32 bits ÷ 16 bits) • Bit manipulation • String		
I/O line	Input	24 (16 ports for analog input)		
	I/O	28	46	
Real-time pulse unit		• 18/16-bit timer/counter : 1 • 18/16-bit compare registers : 5 • 18/16-bit capture registers : 3 • 18/16-bit capture/compare registers : 2 • 16-bit timers/counters : 3 • 16-bit compare registers : 5 • 16-bit capture registers : 1 • Timer outputs : 5 • Programmanble pulse outputs : 6		
Real-time output port		Pulse outputs synchronized with real-time pulse unit: 8		
PWM unit		8-bit PWM outputs: 2 channels		
A/D converter		10-bit resolution: 16 channels		
D/A converter		—		
Serial interface		Serial interface with a dedicated baud rate generator UART: 1 Clock synchronous serial interface/SBI: 1		
Interrupt function		• External: 8, internal: 15 (2 pins are also used for external interrupts.) • Three levels of priorities can be specified according to software types. • One interrupt processing mode can be selected out of three types: vectored interrupt function, macro service function, and context switching function.		
Test factor		Internal 1		
Bus sizing function		Not provided		
ECC circuit		Not provided		Provided
Package	Without a window	• 84-pin PLCC (1150 mil) • 94-pin plastic QFP (20 x 20 mm)		
	With a window	—		• 84-pin ceramic WQFN • 94-pin ceramic WQFN
Others		• Watchdog timer function • Standby function (HALT/STOP)		

[MEMO]

APPENDIX B TOOLS

B.1 Development Tools

The following tools are provided for developing a system that uses the μPD78356:

Language Processor

78K/III series relocatable assembler (RA78K/III)	This relocatable program can be used for all 78K/III series emulators, With its macro functions, it allows the user to improve program development efficiency. A structured-programming assembler is also provided, which enables explicit description of program control structures. This assembler could improve productivity in program production and maintenance.			
	Host machine	OS	Distribution media	Part number
	PC-9800 series	MS-DOS™	3.5-inch 2HD	μS5A13RA78K3
			5-inch 2HD	μS5A10RA78K3
	IBM PC/AT and compatible units	PC DOS™	3.5-inch 2HC	μS7B13RA78K3
			5-inch 2HC	μS7B10RA78K3
	HP9000 series 700™	HP-UX™	DAT	μS3P16RA78K3 ★
	SPARCstation™	SunOS™	Cartridge tape (QIC-24)	μS3K15RA78K3 ★
	NEWS™	NEWS-OS™		μS3R15RA78K3 ★
78K/III series C compiler (CC78K/III)	This C compiler can be used for all 78K/III series emulators. The compiler converts programs written in C language into object codes executable on the microcontroller. When the compiler is used, the 78K/III series relocatable assembler package (RA78K/III) is needed.			
	Host machine	OS	Distribution media	Part number
	PC-9800 series	MS-DOS	3.5-inch 2HD	μS5A13CC78K3
			5-inch 2HD	μS5A10CC78K3
	IBM PC/AT and compatible units	PC DOS	3.5-inch 2HC	μS7B13CC78K3
			5-inch 2HC	μS7B10CC78K3
	HP9000 series 700	HP-UX	DAT	μS3P16CC78K3 ★
	SPARCstation	SunOS	Cartridge tape (QIC-24)	μS3K15CC78K3 ★
	NEWS	NEWS-OS		μS3R15CC78K3 ★

Remark The operations of the relocatable assembler and C compiler are assured only on the OS with the above host machines.

Phase-out/Discontinued

Connections between Development Tools and Target Devices

Development tool / Target device	In-circuit emulator	Emulaiton probe and EPROM product	Conversion adapter	Conversion socket or conversion adapter
GC package (100-pin QFP)	IE-78350-R + IE-78355-R-EM1	EP-78355GC-R	—	EV-9500GC-100
		EP-78355GD-R	EV-9501GC-100	
	—	μPD78P356KP (120-pin WQFN)		
GD package (120-pin QFP)	IE-78350-R + IE-78355-R-EM1	EP-78355GD-R	—	EV-9200GD-120
	—	μPD78P356KP (120-pin WQFN)		

PROM Writing Tools

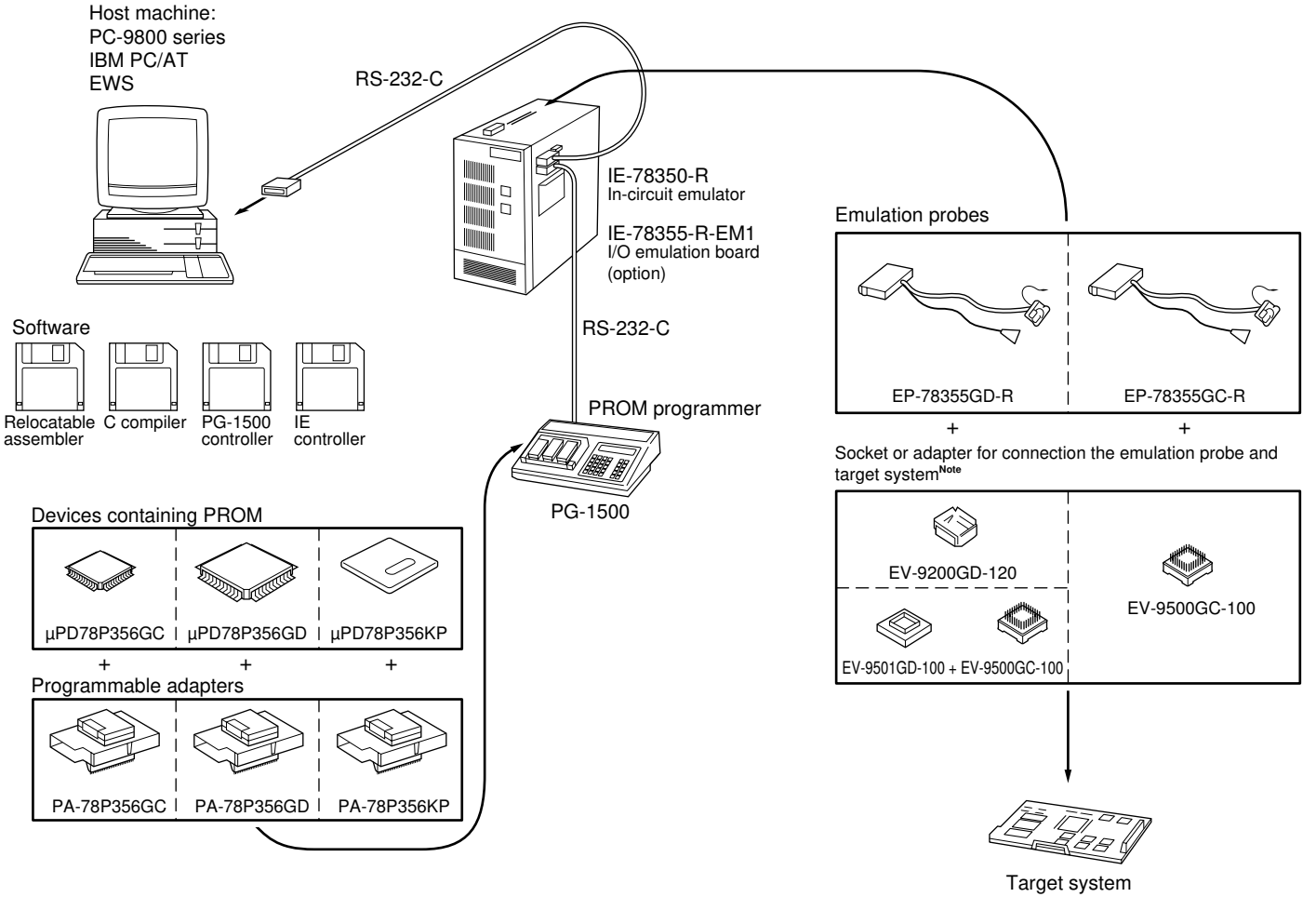
Hardware	PG-1500	By connecting the provided board and optional programmer adapter, can be used as a PROM programmer which can program the single-chip microcontroller incorporated in the PROM by stand-alone or operations through the host machine. Can also program typical PROMs from 256 Kbits to 4 Mbits.			
	PA-78P356GC PA-78P356GD PA-78P356KP	PROM programmer adapter for writing programs in the μPD78P356 on general-use PROM programmers such as the PG-1500. PA-78P356GC ..... μPD78P356GC PA-78P356GD ..... μPD78P356GD PA-78P356KP μPD78P356KP			
Software	PG-1500 controller	Controls the PG-1500 on the host machine when the PG-1500 and host machine are connected with the serial interface and parallel interface.			
		Host machine	OS	Distribution media	Part number
		PC-9800 series	MS-DOS	3.5 inch 2HD	μS5A13PG1500
				5 inch 2HD	μS5A10PG1500
★		IBM PC/AT and compatible units	PC DOS	3.5 inch 2HD	μS7B10PG1500
				5 inch 2HC	μS7B10PG1500

Remark The operations of the PG-1500 controller are assured only on the above host machines and the OS.

Debugging Tools (When Using the IE Controller)

<table>
<tr><td rowspan="5">Hardware</td><td>IE-78350-R</td><td colspan="4">In-circuit emulator for developing and debugging application systems. For debugging, connect the emulator to the host machine.</td></tr>
<tr><td>IE-78355-R-EM1</td><td colspan="4">I/O emulation board for emulating the peripheral functions (the I/O port, etc.) of the μPD78356.</td></tr>
<tr><td>EP-78355GC-R
EV-9500GC-100</td><td colspan="4">μPD78356 100-pin QFP emulation probe for connecting the IE-78350-R and target system. One EV-9500GC-100 conversion adapter is provided for connection to the target system.</td></tr>
<tr><td>EP-78355GD-R
EV-9200GD-120
EV-9501GC-100
+
EV-9500GC-100</td><td colspan="4">μPD78356 120-pin QFP emulation probe for connecting the IE-78350-R and target system. One EV-9200GD-120 conversion socket is provided for connection to the target system. By connecting the 100-pin QFP conversion adapter EV-9501GC-100 (optional), this emulation probe can also be used to develop the 100-pin QFP of the μPD78356. However, for connection to the target system, use the EV-9500GC-100 (optional) conversion adapter.</td></tr>
<tr><td>Software</td><td>IE-78350-R
Control program
(IE controller)</td><td colspan="4">This program controls the IE-78350-R on the host machine. Its automatic command execution function ensures more efficient debugging.</td></tr>
<tr><td colspan="2"></td><td rowspan="2">Host machine</td><td colspan="2" rowspan="2">OS</td><td rowspan="2">Distribution media</td><td rowspan="2">Part number</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td colspan="2"></td><td>PC-9800 series</td><td colspan="2">MS-DOS</td><td>3.5 inch 2HD</td><td>μS5A13IE78355</td></tr>
<tr><td colspan="2"></td><td></td><td colspan="2"></td><td>5 inch 2HD</td><td>μS5A10IE78355</td></tr>
<tr><td colspan="2"></td><td>IBM PC/AT and compatible units</td><td colspan="2">PC DOS</td><td>3.5 inch 2HC</td><td>μS7B13IE78355</td></tr>
<tr><td colspan="2"></td><td></td><td colspan="2"></td><td>5 inch 2HC</td><td>μS7B10IE78355</td></tr>
</table>

Remark The operations of the IE controller are assured only on the above host machines and the OS.



Note The conversion socket or the conversion adapter is supplied with the emulation probe.

Remarks 1. The PG-1500 can be directly connected to the host machine via the RS-232-C interface.

2. In this figure, the distribution media for software is represented by 3.5-inch floppy disk.

Phase-out/Discontinued

Debugging Tools (When Using the Integrated Debugger) (1/2) ★

Hardware	IE-784000-R	In-circuit emulator for developing and debugging application systems. For debugging, connect the emulator to the host machine.
	IE-78350-R-EM-A[Note]	Emulation board for emulating the peripheral functions (the I/O port, etc.) of the target device.
	IE-78355-R-EM1	I/O emulation board for emulating the peripheral functions (the I/O port, etc.) of the target device.
	EP-78355GC-R EV-9500GC-100	100-pin QFP emulation probe for connecting the IE-784000-R and target system. One EV-9500GC-100 conversion adapter is provided for connection to the target system.
	EP-78355GD-R EV-9200GD-120 EV-9501GC-100 + EV-9500GC-100	120-pin QFP emulation probe for connecting the IE-784000-R and target system. One EV-9200GD-120 conversion socket is provided for connection to the target system. By connecting to the EV-9501GC-100 100-pin QFP conversion adapter (option), this emulation probe can be used for developing the μPD78356 100-pin QFP. Use EV-9500GC-100 conversion adapter (option) for connection to the target system.
	IE-70000-98-IF-B	Interface adapter to use the PC-9800 series (except notebook personal computers) as the host machines.
	IE-70000-98N-IF	Interface adapter and cable to use the PC-9800 series notebook personal computers as the host machines.
	IE-70000-PC-IF-B	Interface adapter to use the IBM PC/AT as the host machine.
	IE-78000-R-SV3	Interface adapter and cable to use the EWS as the host machine.

Note Under development

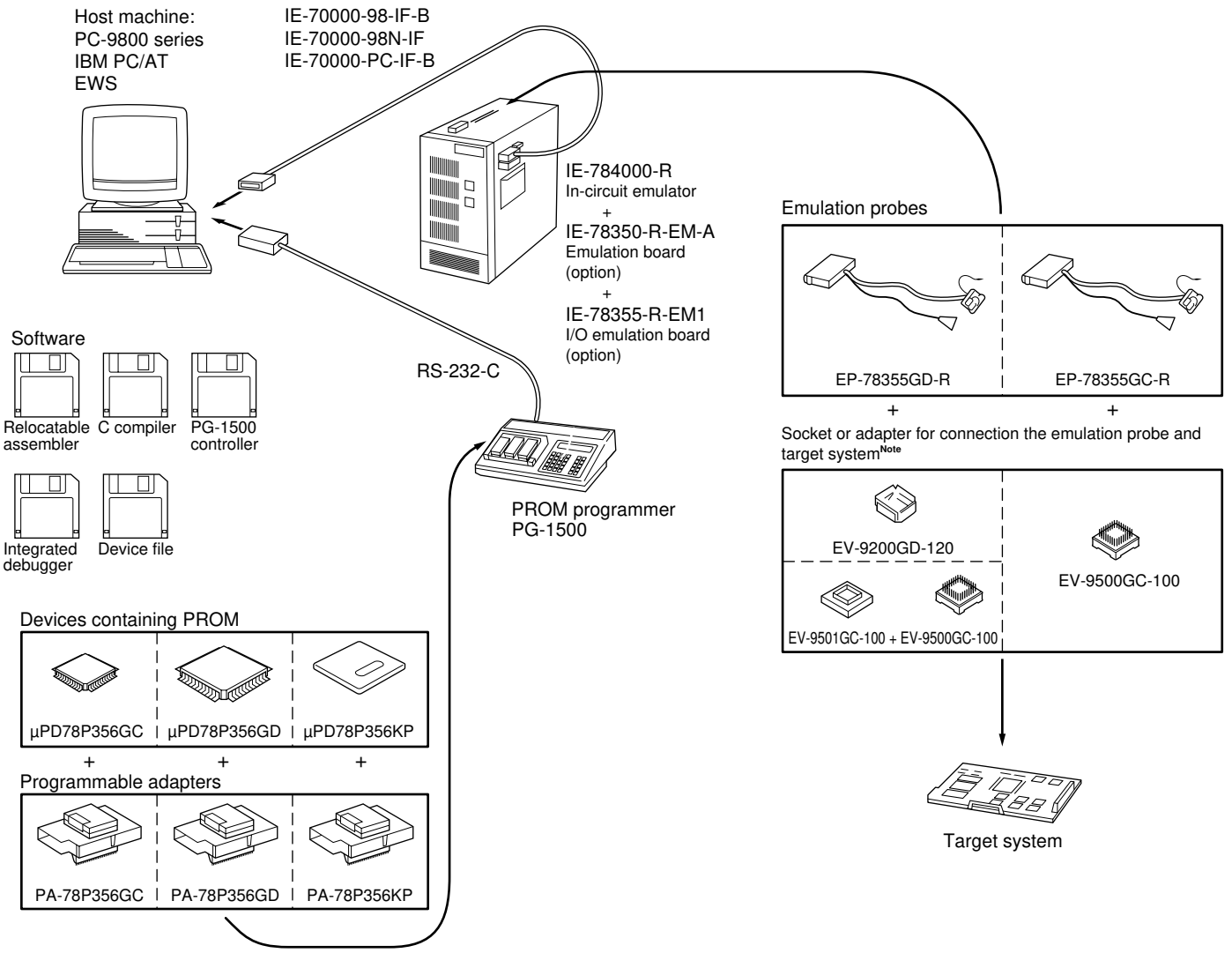
Phase-out/Discontinued

★ **Debugging Tools (When Using the Integrated Debugger) (2/2)**

<table>
<tr><td rowspan="11">Software</td><td rowspan="6">Integrated Debugger (ID78K/III)Note</td><td colspan="4">Program to control the 78K/III series in-circuit emulator. Used in combination with the device file (DF78355).
This debugger enables debugging in the source program level written in the C language, structured assembly language, and assembly language. As various information can be displayed simultaneously by dividing the screen of the host machine, very efficient debugging is possible.</td></tr>
<tr><td rowspan="2">Host machine</td><td rowspan="2">OS</td><td rowspan="2">Distribution media</td><td rowspan="2">Part number</td></tr>
<tr></tr>
<tr><td>PC-9800 series</td><td rowspan="2">MS-DOS
+
Windows™</td><td>3.5-inch 2HD</td><td>μSAA13ID78K3</td></tr>
<tr><td></td><td>5-inch 2HD</td><td>μSAA10ID78K3</td></tr>
<tr><td>IBM PC/AT and compatible units (Japanese Windows)</td><td rowspan="3">PC DOS
+
Windows</td><td>3.5-inch 2HC</td><td>μSAB13ID78K3</td></tr>
<tr><td></td><td>5-inch 2HC</td><td>μSAB10ID78K3</td></tr>
<tr><td>IBM PC/AT and compatible units (English Windows)</td><td>3.5-inch 2HC</td><td>μSBB13ID78K3</td></tr>
<tr><td></td><td>5-inch 2HC</td><td>μSBB10ID78K3</td></tr>
<tr><td rowspan="5">Device file (DF78355)Note</td><td colspan="4">File containing the information inherent to the device. Used in combination with the assembler (RA78K/III), C compiler (CC78K/III), and integrated debugger (ID78K/III).</td></tr>
<tr><td rowspan="2">Host machine</td><td rowspan="2">OS</td><td rowspan="2">Distribution media</td><td rowspan="2">Part number</td></tr>
<tr></tr>
<tr><td>PC-9800 series</td><td rowspan="2">MS-DOS</td><td>3.5-inch 2HD</td><td>μS5A13DF78355</td></tr>
<tr><td></td><td>5-inch 2HD</td><td>μS5A10DF78355</td></tr>
<tr><td>IBM PC/AT and compatible units</td><td rowspan="2">PC DOS</td><td>3.5-inch 2HC</td><td>μS7B13DF78355</td></tr>
<tr><td></td><td>5-inch 2HC</td><td>μS7B10DF78355</td></tr>
</table>

Note Under development

Remark The operations of the integrated debugger and device file are assured only on the above host machines and the OS.



Note The conversion socket or the conversion adapter is supplied with the emulation probe.

Remarks 1. In this figure, the distribution media for software is represented by 3.5-inch floppy disks.

2. In this figure, the host machines are represented by desk-top personal computers.

★

B.2 Embedded Software

The following software are available for programs development and maintenance to be performed more effectively.

Real-time OS

<table>
<tr><td rowspan="6">Real-time OS
(RX78K/III)Note</td><td colspan="3">The aim of RX78K/III to realize multi-task environments targeting real-time control fields. The CPU idle time can be allotted to other processes to improve the overall performance of the system.
The RX78K/III provides system call conforming to the ITRON specifications.
The RX78K/III package provides the tools (configuration) for creating the RX78K/III nucleus and several information tables.</td></tr>
<tr><td rowspan="2">Host machine</td><td rowspan="2">OS</td><td>Distribution media</td><td rowspan="2">Part number</td></tr>
<tr><td></td></tr>
<tr><td>PC-9800 series</td><td>MS-DOS</td><td>3.5-inch 2HD</td><td>Undefined</td></tr>
<tr><td></td><td></td><td>5-inch 2HD</td><td>Undefined</td></tr>
<tr><td>IBM PC/AT and compatible units</td><td>PC DOS</td><td>3.5-inch 2HC</td><td>Undefined</td></tr>
<tr><td></td><td></td><td>5-inch 2HC</td><td>Undefined</td></tr>
</table>

Note Under development

Caution To purchase, the purchasing application must be filled in advance and a using conditions agreement signed.

Remark The RA78K/III assembler package (optional) is required when using the RX78K/III real-time OS.

Phase-out/Discontinued

Fuzzy Inference Development Support System

Fuzzy knowledge data compilation tool (FE9000, FE9200)	This program supports the input/editing and simulation of fuzzy knowledge data (fuzzy rules and membership functions).			
	Host machine	OS	Distribution media	Part number
	PC-9800 series	MS-DOS	3.5-inch 2HD	μS5A13FE9000
			5-inch 2HD	μS5A10FE9000
	IBM PC/AT and compatible units	PC DOS + Windows	3.5-inch 2HC	μS7B13FE9200
			5-inch 2HC	μS7B10FE9200
Translator (FT78K3)Note	This program converts fuzzy knowledge data obtained using the fuzzy knowledge data compilation tool to RA78K/III assembler source programs.			
	Host machine	OS	Distribution media	Part number
	PC-9800 series	MS-DOS	3.5-inch 2HD	μS5A13FT78K3
			5-inch 2HD	μS5A10FT78K3
	IBM PC/AT and compatible units	PC DOS	3.5-inch 2HC	μS7B13FT78K3
			5-inch 2HC	μS7B10FT78K3
Fuzzy inference module (FI78K/III)Note	This program executes fuzzy inference by linking fuzzy knowledge data converted by the translator.			
	Host machine	OS	Distribution media	Part number
	PC-9800 series	MS-DOS	3.5-inch 2HD	μS5A13FI78K3
			5-inch 2HD	μS5A10FI78K3
	IBM PC/AT and compatible units	PC DOS	3.5-inch 2HC	μS7B13FI78K3
			5-inch 2HC	μS7B10FI78K3
Fuzzy inference debugger (FD78K/III)	This support software simulates and adjust fuzzy knowledge data in the hardware level using the in-circuit emulator.			
	Host machine	OS	Distribution media	Part number
	PC-9800 series	MS-DOS	3.5-inch 2HD	μS5A13FD78K3
			5-inch 2HD	μS5A10FD78K3
	IBM PC/AT and compatible units	PC DOS	3.5-inch 2HC	μS7B13FD78K3
			5-inch 2HC	μS7B10FD78K3

Note Under development

[MEMO]

APPENDIX C REGISTER INDEX

C.1 Register Name Index (Alphabetic Order)

[U]

[W]

C.2 Register Symbol Index (Alphabetic Order)

APPENDIX D FUNCTION INDEX

APPENDIX E REVISION HISTORY

The history of revisions made up to now is shown in the following. The Application column shows in which chapters these revisions were made in.

Edition No.	Revisions	Application
2nd Edition	Addition of μPD78355 (A), 78355 (A1), 78355 (A2), 78356 (A), 78356 (A1), 78356 (A2), 78P356 (A), 78P356 (A1) and 78P356 (A2) to applied models.	Throughout
	Addition of cautions for word access to the main RAM area	**CHAPTER 3 CPU ARCHITECTURE**
	Addition of cautions and program examples for branch processes using A/D conversion results.	**CHAPTER 8 A/D CONVERTER**
	Addition of cautions when several triggers are input simultaneously in timer trigger mode and external trigger mode.	
	Change of explanation on reception operation start up in 3-wire serial I/O mode, and SBI mode and delete cautions.	**CHAPTER 11 CLOCK SYNCHRONOUS SERIAL INTERFACE**
	Change of explanation on reception operation start up in 3-wire serial I/O mode, and delete cautions.	**CHAPTER 12 CLOCK SYNCHRONOUS SERIAL INTERFACE (WITH PIN SWITCHING FUNCTION)**
	Addition of **15.4 Nonmaskable Interrupt Acknowledgement, 15.5 Maskable Interrupt Acknowledgement, 15.9 When Interrupt Requests and Macro Service are Held Temporarily** and **15.10 Instructions Which are Stopped Temporarily in Interrupts and Macro Services**	**CHAPTER 15 INTERRUPT FUNCTION**
	Deletion of **19.2 Handling Unused Pins**	**CHAPTER 19 PROGRAMMING FOR THE μPD78P356**
	Addition of **19.7 Screening of One-Time PROM Version Devices**	
	Deletion of **20.2 Instruction Codes, 20.3 Number of Clocks of the Instructions, 20.4 Instruction Addressing, 20.5 Operand Addressing** and **20.6 Explanation of Instructions**	**CHAPTER 20 INSTRUCTION SET**
	Deletion of **21.1 Instruction Execution Rate**	**CHAPTER 21 INSTRUCTION EXECUTION RATE**
	Addition of **B.2 Embedded Software**	**APPENDIX B TOOLS**
3rd Edition	"The following products are under development" -> changed to "already developed" μPD78355, 78356, 78P356, 78356 (A), 78P356 (A)	Throughout
	Addition of description of the external 8-bit bus to **5.2.3 (4) Port 9**	**CHAPTER 5 PORT FUNCTIONS**
	Addition of **10.7 Transmitting/Receiving Data Using the Macro Service**	**CHAPTER 10 ASYNCHRONOUS SERIAL INTERFACE**
	Addition of description when using the integrated debugger	**APPENDIX B TOOLS**

[MEMO]

Phase-out/Discontinued

NEC

Facsimile Message

From:

Name

Company

Tel. FAX

Address

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

Thank you for your kind support.

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

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

South America
NEC do Brasil S.A.
Fax: +55-11-889-1689

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

Korea
NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: +82-2-551-0451

Taiwan
NEC Electronics Taiwan Ltd.
Fax: +886-2-719-5951

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

Japan
NEC Corporation
Semiconductor Solution Engineering Division
Technical Information Support Dept.
Fax: +81-44-548-7900

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

Document title: _____

Document number: _____ Page number: _____

If possible, please fax the referenced page or drawing.

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