

To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: <http://www.renesas.com>

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (<http://www.renesas.com>)

Send any inquiries to <http://www.renesas.com/inquiry>.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
 - “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
 - “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
 - “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

8-BIT SINGLE-CHIP MICROCOMPUTER (WITH A/D CONVERTER)

DESCRIPTION

The μ PD78C14A is a CMOS 8-bit microprocessor which can integrate 16-bit ALU, ROM, RAM, an A/D converter, a multi-function timer/event counter, and a general-purpose serial interface into a single chip, then expand the memory (ROM/RAM) up to 48 K bytes externally. The μ PD78C14A is operated at low power consumption, because it has a CMOS construction. Also, it can hold data with low power consumption by using standby function.

On-chip PROM products, μ PD78CP14 and μ PD78CP18 which are ideal for evaluation or preproduction use during system development, early start-up and short-run multiple-device production of application sets, are available.

Detailed functions are described in the following user's manual. Be sure to read this manual when you design your system.

87AD Series μ PD78C18 User's Manual : IEU-1314

FEATURES

- Abundant 159 types of instructions : 87AD series instruction set, multiplication/division instructions, 16-bit operation instructions
- Instruction cycle : 0.8 μ s (at 15 MHz operation)
- On-chip ROM : 16384W \times 8
- On-chip RAM : 256W \times 8
- Memory (ROM/RAM) can be addressed up to 64 Kbytes
- High-precision 8-bit A/D converter : 8 analog inputs
- General-purpose serial interface : Asynchronous, synchronous, I/O interface mode
- Multi-function 16-bit timer/event counter
- Two 8-bit timers
- I/O lines
I/O port : 40 (PA, PB, and PC can be connected with a pull-up resistor through mask option).
Edge detection input : 4
- Interrupt function (external - 3, internal - 8) : Non-maskable interrupt \times 1, maskable interrupt \times 10
- Zero-cross detection function : (2 inputs)
- Standby function : HALT mode, hardware/software STOP mode
- CMOS
- Single power supply

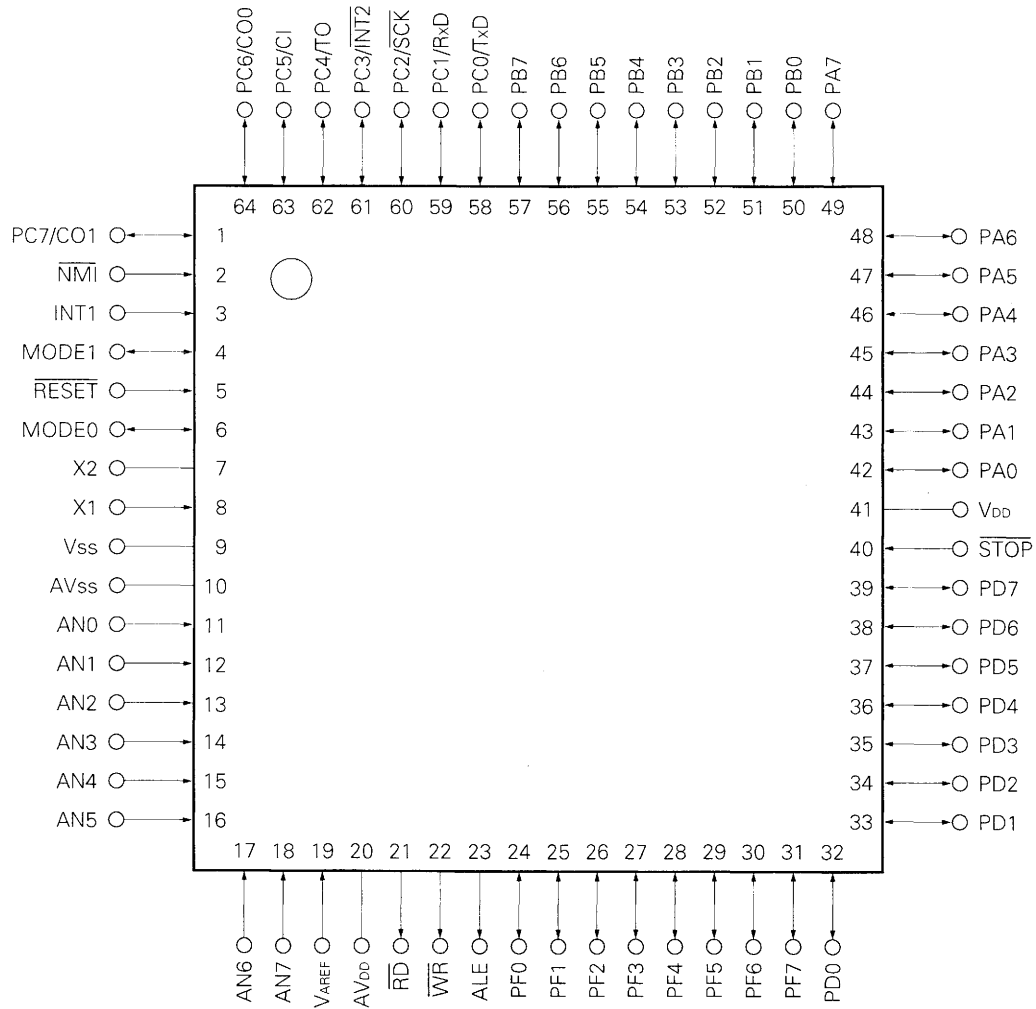
ORDERING INFORMATION

Part Number	Package
μ PD78C14AG-xxx-AB8	64-pin plastic QFP (14 \times 14 mm)

Remark xxx indicates a ROM code number.

The information in this document is subject to change without notice.

PIN CONFIGURATION (TOP VIEW)



CONTENTS

- 1. PIN FUNCTIONS 5
 - 1.1 LIST OF PIN FUNCTION 5
 - 1.2 PIN INPUT/OUTPUT CIRCUITS 7
 - 1.3 PIN MASK OPTIONS 12
 - 1.4 RECOMMENDED CONNECTION OF UNUSED PINS 12

- 2. INTERNAL BLOCK FUNCTIONS 13
 - 2.1 REGISTERS 13
 - 2.2 ARITHMETIC AND LOGICAL UNIT (ALU): 16 BITS 14
 - 2.3 PROGRAM STATUS WORD (PSW) 14
 - 2.4 MEMORY 16
 - 2.5 PORT FUNCTIONS 18
 - 2.6 TIMERS 26
 - 2.7 TIMER/EVENT COUNTER 29
 - 2.8 SERIAL INTERFACE 35
 - 2.9 ANALOG/DIGITAL CONVERTER 45
 - 2.10 ZERO-CROSS DETECTION CIRCUIT 48

- 3. INTERRUPT FUNCTION 50
 - 3.1 INTERRUPT CONTROL CIRCUIT 50
 - 3.2 OPERATION OF NON-MASKABLE INTERRUPTS 54
 - 3.3 MASKABLE INTERRUPT OPERATION 56
 - 3.4 INTERRUPT OPERATION BY SOFTI INSTRUCTION 57

- 4. STANDBY FUNCTION 58
 - 4.1 HALT MODE 58
 - 4.2 CANCELLATION OF HALT MODE 58
 - 4.3 SOFTWARE STOP MODE 59
 - 4.4 CANCELLATION OF SOFTWARE STOP MODE 60
 - 4.5 HARDWARE STOP MODE 61
 - 4.6 CANCELLATION OF HARDWARE STOP MODE 61
 - 4.7 LOW-SUPPLY-VOLTAGE DATA RETENTION MODE 62

- 5. RESET OPERATIONS 63

- 6. INSTRUCTION SET 64
 - 6.1 IDENTIFIER/DESCRIPTION OF OPERAND 64
 - 6.2 SYMBOL DESCRIPTION OF OPERATION CODE 65
 - 6.3 INSTRUCTION EXECUTION TIME 66

- 7. LIST OF MODE REGISTERS 78

- 8. ELECTRICAL SPECIFICATIONS 79

- 9. CHARACTERISTIC CURVES (REFERENCE VALUES) 91

- 10. PACKAGE DRAWINGS 92

- 11. RECOMMENDED SOLDERING CONDITIONS 93

- 12. DIFFERENCES BETWEEN μPD78C14A and 78C14 94

- APPENDIX DEVELOPMENT TOOLS 95

1. PIN FUNCTIONS

1.1 LIST OF PIN FUNCTION (1/2)

Pin Name	I/O	Function	
PA7 to PA0 (Port A)	Input/Output	8-bit input-output port, which can specify input/output bit-wise.	
PB7 to PB0 (Port B)	Input/Output	8-bit input-output port, which can specify input/output bit-wise.	
PC0/TxD	Input-output/ Output	Port C 8-bit input-output port, which can specify input/ output bit-wise.	Transmit Data Output pin for serial data.
PC1/RxD	Input-output/ Input		Receive Data Input pin for serial data.
PC2/SCK	Input-output/ Input-output		Serial Clock Input-output pin for serial clock. It becomes output clock for the internal clock use, and input for the external.
PC3/INT2/TI	Input-output/ Input/Input		Interrupt Request/Timer Input Maskable interrupt input pin of the edge trigger (falling edge), or an external clock input pin for a timer. Also, it can be used as a zero-cross detection pin for AC input.
PC4/TO	Input-output/ Output		Timer Output Square wave defining one cycle of internal clock or timer counter time as half cycle is output.
PC5/CI	Input-output/ Input		Counter Input External pulse input pin to timer/event counter.
PC6/CO0 PC7/CO1	Input-output/ Output		Counter Output 0, 1 Programmable rectangle wave output by timer/event counter.
PD7 to PD0/ AD7 to AD0	Input-output/ Input-output		Port D 8-bit input-output port, which can specify input-output in byte units.
PF7 to PF0/ AB15 to AB8	Input-output/ Output	Port F 8-bit input-output port, which can specify input-output bit-wise.	Address Bus When external memory is used, it be- comes address bus.
\overline{WR} (Write Strobe)	Output	Strobe signal which is output for write operation of external memory. It becomes high in any cycle other than the data write machine cycle of external memory. When \overline{RESET} signal is either low or in the hardware STOP mode, this signal becomes output high-impedance.	
\overline{RD} (Read Strobe)	Output	Strobe signal which is output for read operation of external memory. It becomes high in any cycle other than the read machine cycle of external memory. When \overline{RESET} signal is either low or in the hardware STOP mode, this signal becomes output high-impedance.	
ALE (Address Latch Enable)	Output	Strobe signal to latch externally the lower address information which is output to PD7 to PD0 pins to access external memory. When \overline{RESET} signal is either low or in the hardware STOP mode, this signal becomes output high-impedance.	

1.1 LIST OF PIN FUNCTION (2/2)

Pin Name	I/O	Function
MODE0 MODE1 (Mode)	Input-output	μPD78C14A sets MODE0 pin to "0" (low level), and MODE1 pin to "1" (high level) ^{Note} Also, when each of MODE0 and MODE1 pins is set to "1" ^{Note} , it is synchronized to ALE to output a control signal.
$\overline{\text{NMI}}$ (Non-Maskable Interrupt)	Input	Non-maskable interrupt input pin of the edge trigger (falling edge)
INT1 (Interrupt Request)	Input	A maskable interrupt input pin of the edge trigger (rising edge). Also, it can be used as a zero-cross detection pin for AC input.
AN7 to AN0 (Analog Input)	Input	8 pins of analog input to A/D converter. AN7 to AN4 can be used as edge detection (falling edge) input.
V _{AREF} (Reference Voltage)	Input	A common pin serving both as a standard voltage input pin for A/D converter and as a control pin for A/D converter operation.
AV _{DD} (Analog V _{DD})		Power supply pin for A/D converter.
AV _{SS} (Analog V _{SS})		GND pin for A/D converter.
X1, X2 (Crystal)		Crystal connection pins for system clock oscillation. X1 should be input when a clock is supplied from outside. Input the clock of the reverse phase of X1 to X2.
$\overline{\text{RESET}}$ (Reset)	Input	Low-level active system reset input.
$\overline{\text{STOP}}$ (Stop)		Control signal input pin in hardware STOP mode. The oscillation stops when a clock is supplied from outside.
V _{DD}		Positive power supply pin.
V _{SS}		GND pin.

Note Pull-up. Pull-up resistor R is $4 \text{ [k}\Omega\text{]} \leq R \leq 0.4 \text{ tcy} \text{ [k}\Omega\text{]}$ (tcyc is ns unit).

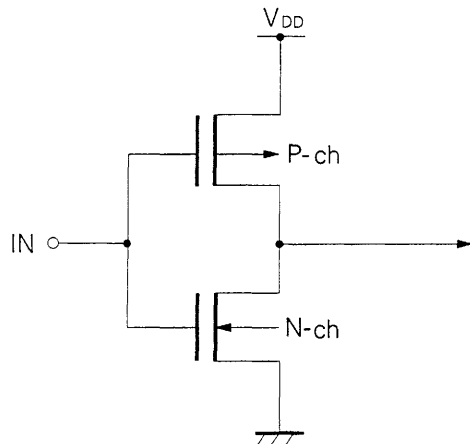
1.2 PIN INPUT/OUTPUT CIRCUITS

Table 1-1 and figures (1) to (15) show input- output circuits of each pin in a partially simplified form.

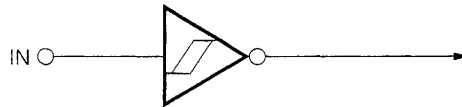
Table 1-1. Pin Type No.

Pin Name	Type No.	Pin Name	Type No.
PA0-7	5-A	$\overline{\text{RESET}}$	2
PB0-7	5-A	$\overline{\text{RD}}$	4
PC0-1	5-A	$\overline{\text{WR}}$	4
PC2/ $\overline{\text{SCK}}$	8-A	ALE	4
PC3/ $\overline{\text{INT2}}$	10-A	$\overline{\text{STOP}}$	2
PC4-7	5-A	MODE0	11
PD0-7	5	MODE1	11
PF0-7	5	AN0-3	7
$\overline{\text{NMI}}$	5	AN4-7	12
INT1	2	V _{AREF}	13

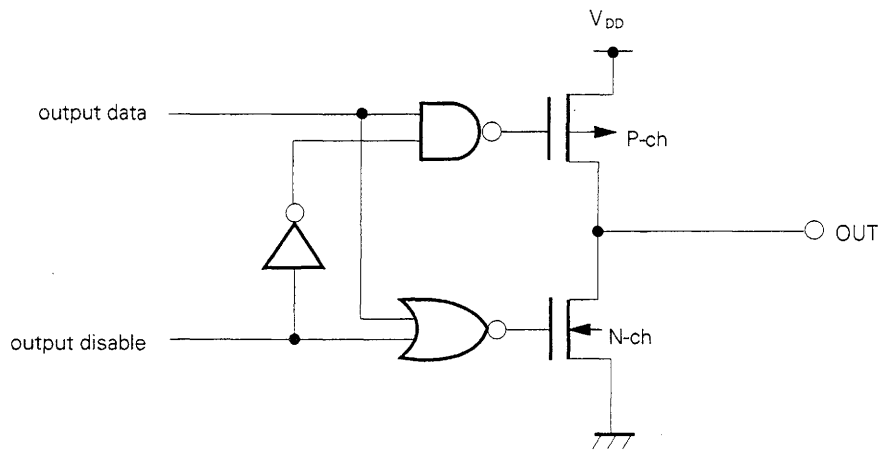
(1) Type 1



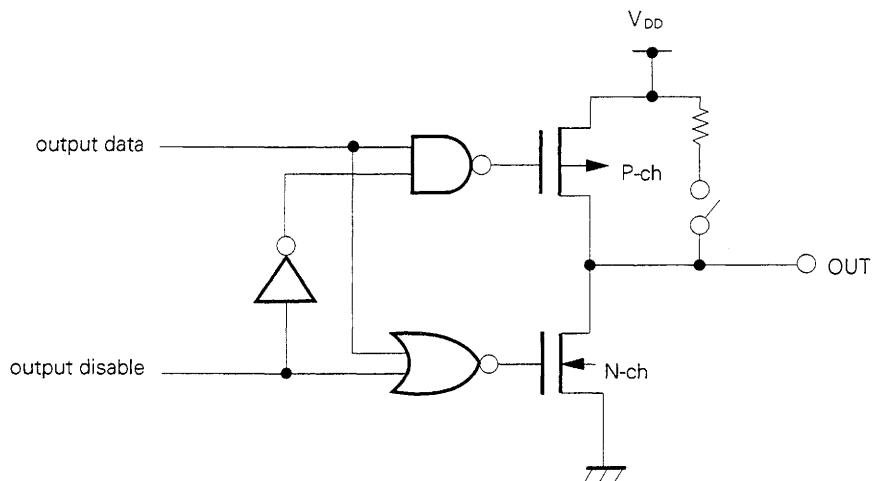
(2) Type 2



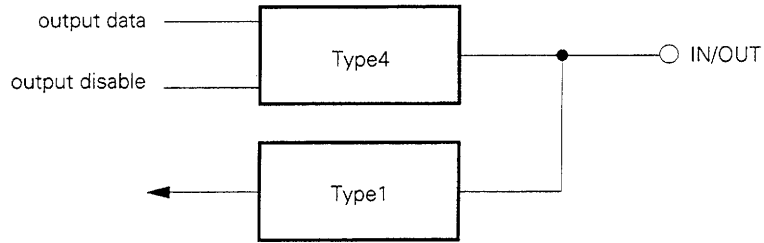
(3) Type 4



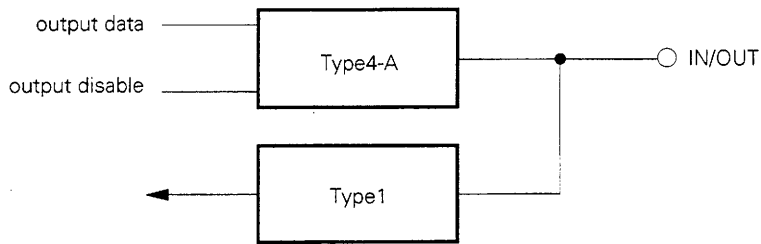
(4) Type 4-A



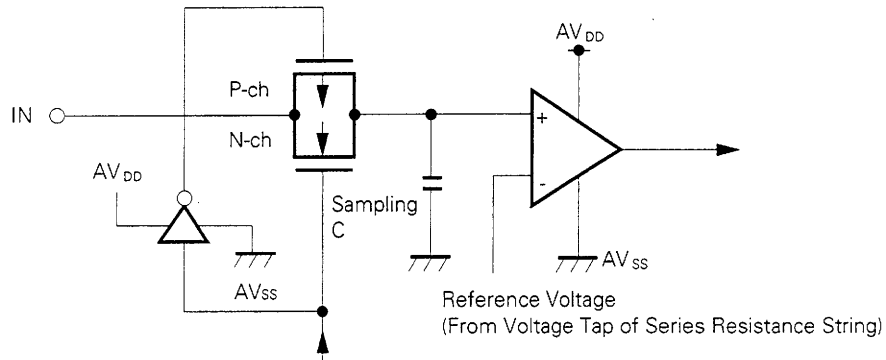
(5) Type 5



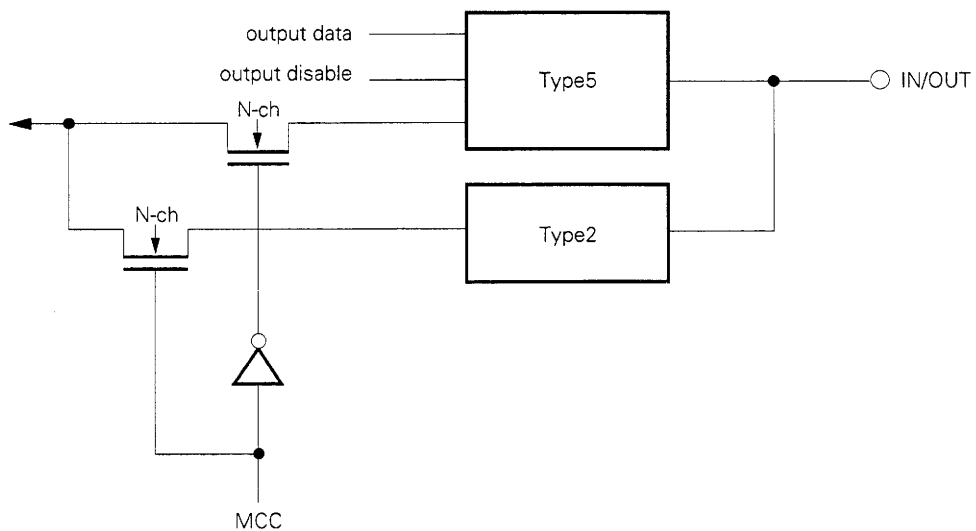
(6) Type 5-A



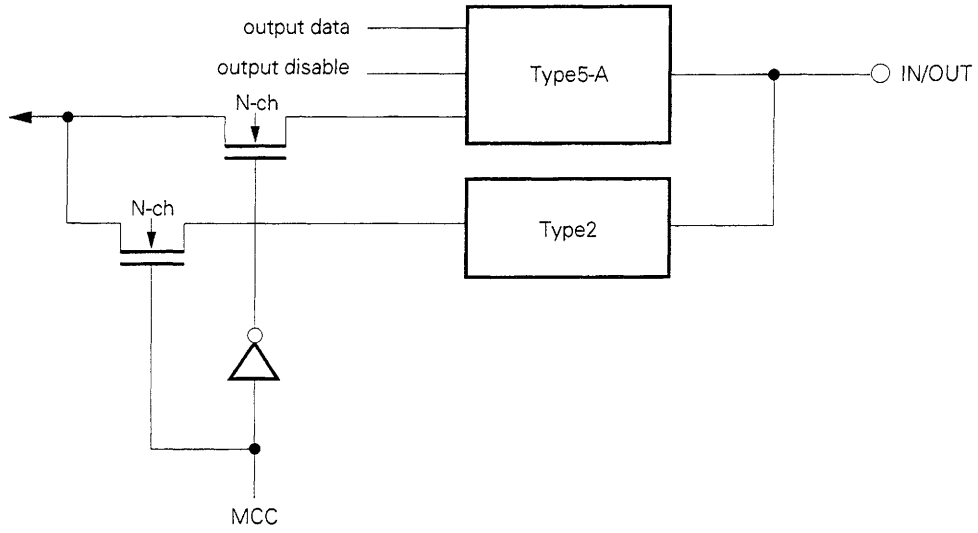
(7) Type 7



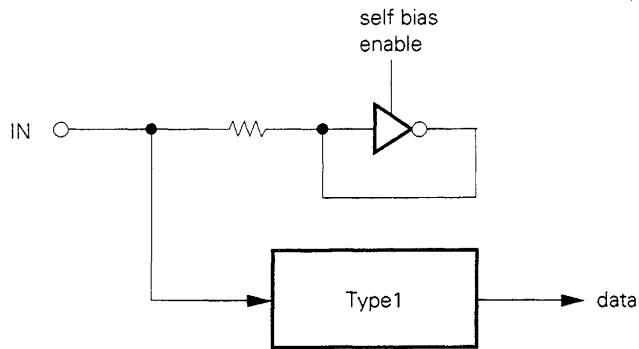
(8) Type 8



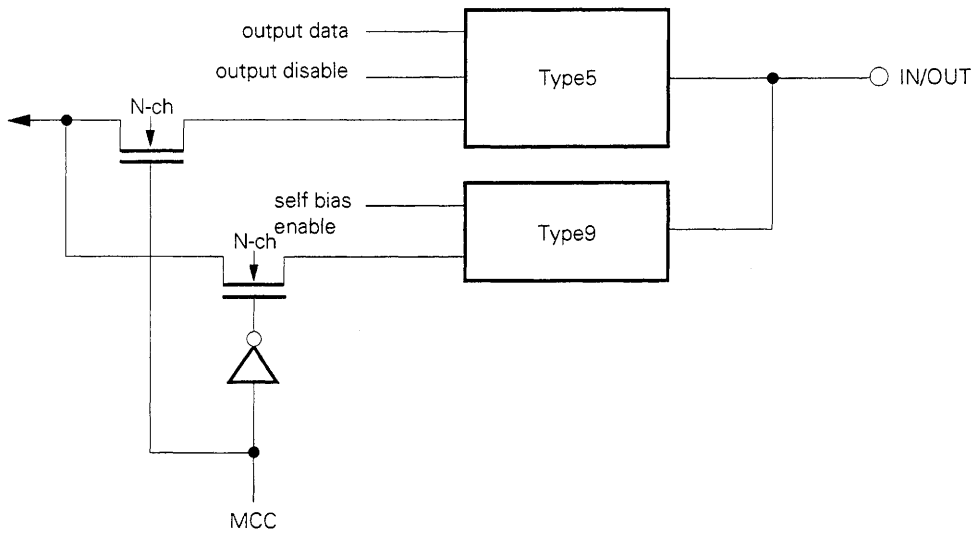
(9) Type 8-A



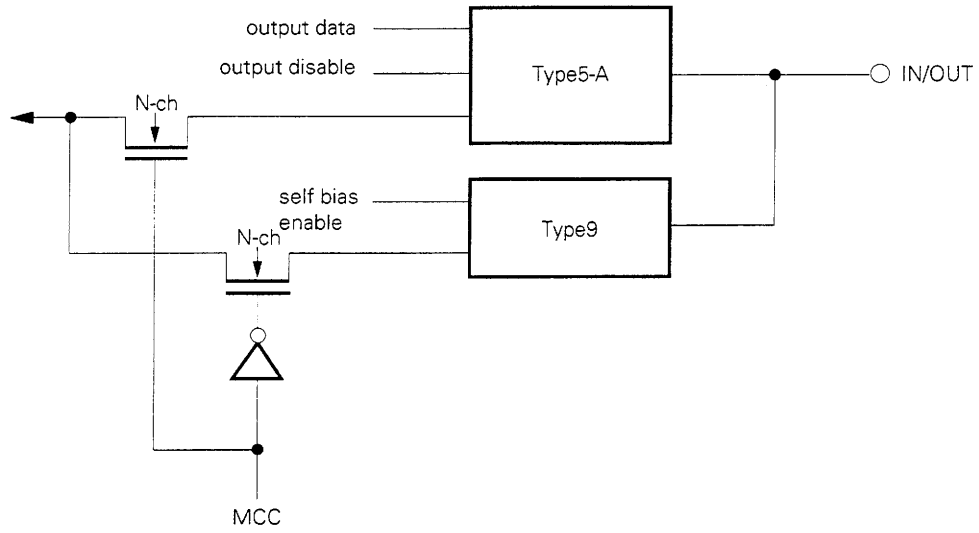
(10) Type 9



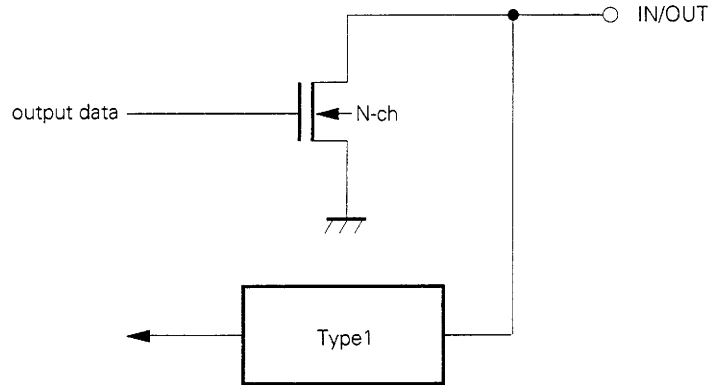
(11) Type 10



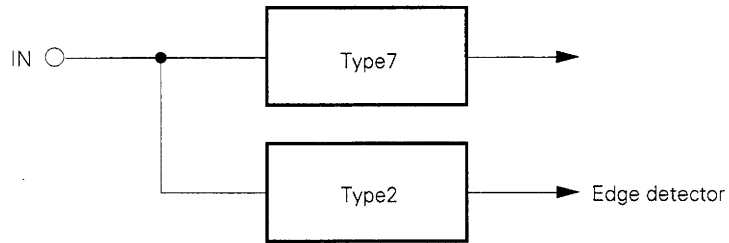
(12) Type 10-A



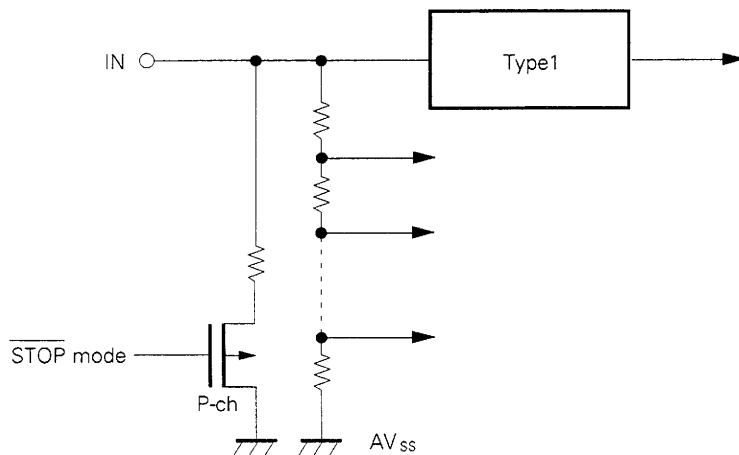
(13) Type 11



(14) Type 12



(15) Type 13



1.3 PIN MASK OPTIONS

μPD78C14A has the following mask options, which can be selected bit-wise according to the application.

Pin Name	Mask Options
PA7-0	① Pull-up resistor incorporated ② Pull-up resistor not incorporated
PB7-0	
PC7-0	

Caution Zero-cross function can not be operated normally if pull-up resistor is incorporated in PC3.

1.4 RECOMMENDED CONNECTION OF UNUSED PINS

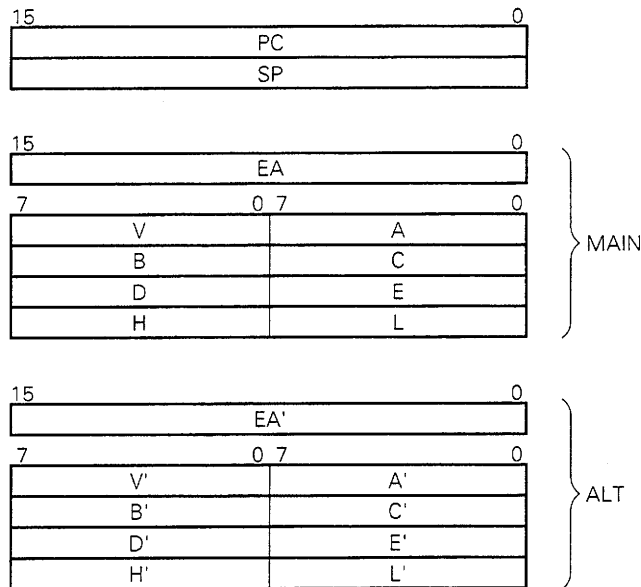
Pin	Recommended Connection
PA7-0 PB7-0 PC7-0 PD7-0 PF7-0	Connect to V _{SS} or V _{DD} via resistor
\overline{RD} \overline{WR} ALE	Leave open
STOP	Connect to V _{DD}
INT1, NMI	Connect to V _{SS} or V _{DD}
AV _{DD}	Connect to V _{DD}
AV _{AREF} AV _{SS}	Connect to V _{SS}
AN7-0	Connect to AV _{SS} or AV _{DD}

2. INTERNAL BLOCK FUNCTIONS

2.1 REGISTERS

Major registers include the sixteen 8-bit registers and four 16-bit registers shown in Figure 2-1.

Figure 2-1. Register Configuration



(a) General-purpose registers (B, C, D, E, H, L)

Two sets (MAIN: B, C, D, E, H, L; ALT: B', C', D', E', H', and L') of general-purpose registers are available. Besides use as auxiliary accumulator registers, they are also used as data pointer register pairs (BC, DE, HL; B'C', D'E', and H'L'). In particular, the four registers DE, D'E', HL, and H'L', are used as base registers. If the two sets are used, when an interrupt occurs, the contents of the registers can be saved in the other registers while performing interrupt services, without saving them in memory. It is also possible to use the other register set as extended data pointer registers. There are one-step auto-increment/decrement and two-step auto-increment addressing modes for the DE, HL, D'E', and H'L' register pairs, thus making it possible to shorten the processing time. BC, DE, and HL can be exchanged with the ALT register simultaneously by the EXX instruction. Also, the EXH instruction can be used to exchange only the HL register with the ALT register.

(b) Working register vector register (V)

When setting a working area in the memory space, the upper 8 bits of the memory address are selected by this V-register, and the lower 8 bits are addressed by the immediate data of the instruction. Accordingly, the memory area specified by the V-register can be used as the working registers configured with 256W x 8.

Because only a 1-byte address field is needed to specify the working register as above, it can be used as the work area for software flags, parameters, and counters, etc., thus economizing on space. The accumulator and the ALT register can be exchanged by using the EXA instruction.

(c) Accumulator (A)

As the accumulator is based on the architecture of the μPD78C14A accumulator system, 8-bit data processing such as 8-bit arithmetic and logical operations are performed centered on the accumulator. The EXA instruction can be used to exchange the vector register (V) with the ALT register as a pair.

(d) Extended accumulator (EA)

16-bit data processing such as 16-bit arithmetic and logical operations are performed centered on EA. The EXA instruction can be used to replace this with the ALT register's EA'.

(e) Program counter (PC)

The program counter refers to the 16-bit register that holds the address of the program step to be executed next. Normally, the counter is automatically incremented in accordance with the number of bytes of the instruction fetched; however, when an instruction involving a branch is executed, immediate data or register contents are loaded into this register. The register is cleared to 0000H by inputting $\overline{\text{RESET}}$.

(f) Stack pointer (SP)

The stack pointer (SP) refers to the 16-bit register that holds the start address of the memory stack area (LIFO format).

The contents of SP are decremented when the call instruction or the PUSH instruction is executed or an interruption has occurred; and incremented when the return instruction or POP instruction is executed.

2.2 ARITHMETIC AND LOGICAL UNIT (ALU): 16 BITS

The ALU performs data processing such as 8-bit arithmetic/logical operations, shifting, and rotation; 16-bit arithmetic/logical operations and shifting; 8-bit multiplications; and division of 16 bits by 8 bits.

2.3 PROGRAM STATUS WORD (PSW)

The program status word (PSW) consists of six flags which are set/reset depending on the results of the instruction execution. Among them, three flags (Z, HC, CY) can be tested by instructions. The contents of the PSW are automatically saved on the stack when an interruption (external, internal or SOFTI instruction) occurs and restored by the RETI instruction. All the bits are reset (to 0) by the $\overline{\text{RESET}}$ input.

Figure 2-2. PSW Configuration

7	6	5	4	3	2	1	0
0	Z	SK	HC	L1	L0	0	CY

(a) Z (Zero)

This is set (to 1) when the operation result is zero. Otherwise, it is reset (to 0).

(b) SK (Skip)

This is set (to 1) when the skip condition is true. Otherwise, it is reset (to 0).

(c) HC (Half Carry)

This is set (to 1) when a carry has been generated from bit-3 as the result of an 8-bit operation, or if a borrow has been generated into bit-3. Otherwise, it is reset (to 0).

(d) L1

This is set (to 1) when vertical accumulation of the MVI A,byte instruction is performed. Otherwise, it is reset (to 0).

(e) L0

This is set (to 1) when vertical accumulation of the MVI L, byte; LXI H, word instructions has been performed. Otherwise, it is reset (to 0).

(f) **CY (Carry)**

This is set (to 1) when a carry has been made from bit-7 or bit-15 as a result of an operation or when a borrow has been made into bit-7 or bit-15. Otherwise, it is reset (to 0).

Execution of 35 types of ALU instructions, rotation instructions, and carry operations affects various flags as shown in Table 2-1.

Table 2-1. Flag Operations

Operation					D6	D5	D4	D3	D2	D0	
reg. memory		immediate			skip	Z	SK	HC	L1	L0	CY
ADD	ADDW	ADDX	ADI								
ADC	ADCW	ADCX	ACI								
SUB	SUBW	SUBX	SUI								
SBB	SBBW	SBBX	SBI								
DADD					↑	0	↑	0	0	↑	
DADC											
DSUB											
DSBB											
EADD											
ESUB											
ANA	ANAW	ANAX	ANI	ANIW							
ORA	ORAW	ORAX	ORI	ORIW							
XRA	XRAW	XRAX	XRI		↑	0	●	0	0	●	
DAN											
DOR											
DXR											
ADDNC	ADDNCW	ADDNCX	ADINC								
SUBNB	SUBNBW	SUBNBX	SUINB	GTIW							
GTA	GTAW	GTAX	GTI	LTIW							
LTA	LTAW	LTAX	LTI		↑	↑	↑	0	0	↑	
DADDNC											
DSUBNB											
DGT											
DLT											
ONA	ONAW	ONAX	ONI	ONIW							
OFFA	OFFAW	OFFAX	OFFI	OFFIW							
DON					↑	↑	●	0	0	●	
DOFF											
NEA	NEAW	NEAX	NEI	NEIW							
EQA	EQAW	EQAX	EQI	EQIW							
DNE					↑	↑	↑	0	0	↑	
DEQ											
INR	INRW				↑	↑	↑	0	0	●	
DCR	DCRW										
DAA					↑	0	↑	0	0	↑	
RLR RLL SLR SLL					●	0	●	0	0	↑	
DRLR DRLL DSLR DSLL					●	↑	●	0	0	↑	
SLRC SLLC					●	0	●	0	0	1	
STC					●	0	●	0	0	1	
CLC					●	0	●	0	0	0	
			MVI A, byte		●	0	●	1	0	●	
			MVI L, byte		●	0	●	0	1	●	
			LXI H, word		●	0	●	0	1	●	
			BIT								
			SK								
			SKN		●	↑	●	0	0	●	
			SKIT								
			SKNIT								
			RETS		●	1	●	0	0	●	
					●	0	●	0	0	●	
All other instructions					●	0	●	0	0	●	

↑.....Affected (set or reset)
 1.....Set
 0.....Reset
 ●.....Unaffected

2.4 MEMORY

The μPD78C14A can address memory of up to 64 Kbytes. The memory map is shown in Figure 2-3. The external memory and internal RAM areas can be used freely as program memory and data memory. The access timing of internal memory and that of external memory are the same, thus enabling high-speed processing.

(a) Interrupt start address

All the interrupt start addresses are fixed as follows:

NMI	0004H
INTT0/INTT1	0008H
INT1/INT2	0010H
INTE0/INTE1	0018H
INTEIN/INTAD	0020H
INTSR/INTST	0028H
SOFTI	0060H

(b) Call address table

The call address of the 1-byte call instruction (CALT) can be stored in the 64-byte area (32-call address portion) at addresses 0080H-00BFH.

(c) Memory specific region

Addresses 0000H-00BFH are assigned as the reset start address, the interrupt start address, and the call table. Addresses 0800H-0FFFH can be addressed directly by the 2-byte call instruction (CALF).

The mask-programmable ROM is integrated in addresses 0000H-3FFFH.

(d) Built-in data memory area

The 256-byte RAM is incorporated in addresses FF00H-FFFFH. In the standby operation, the contents of the 256-byte portion of the built-in data memory area RAM are retained.

(e) External memory area

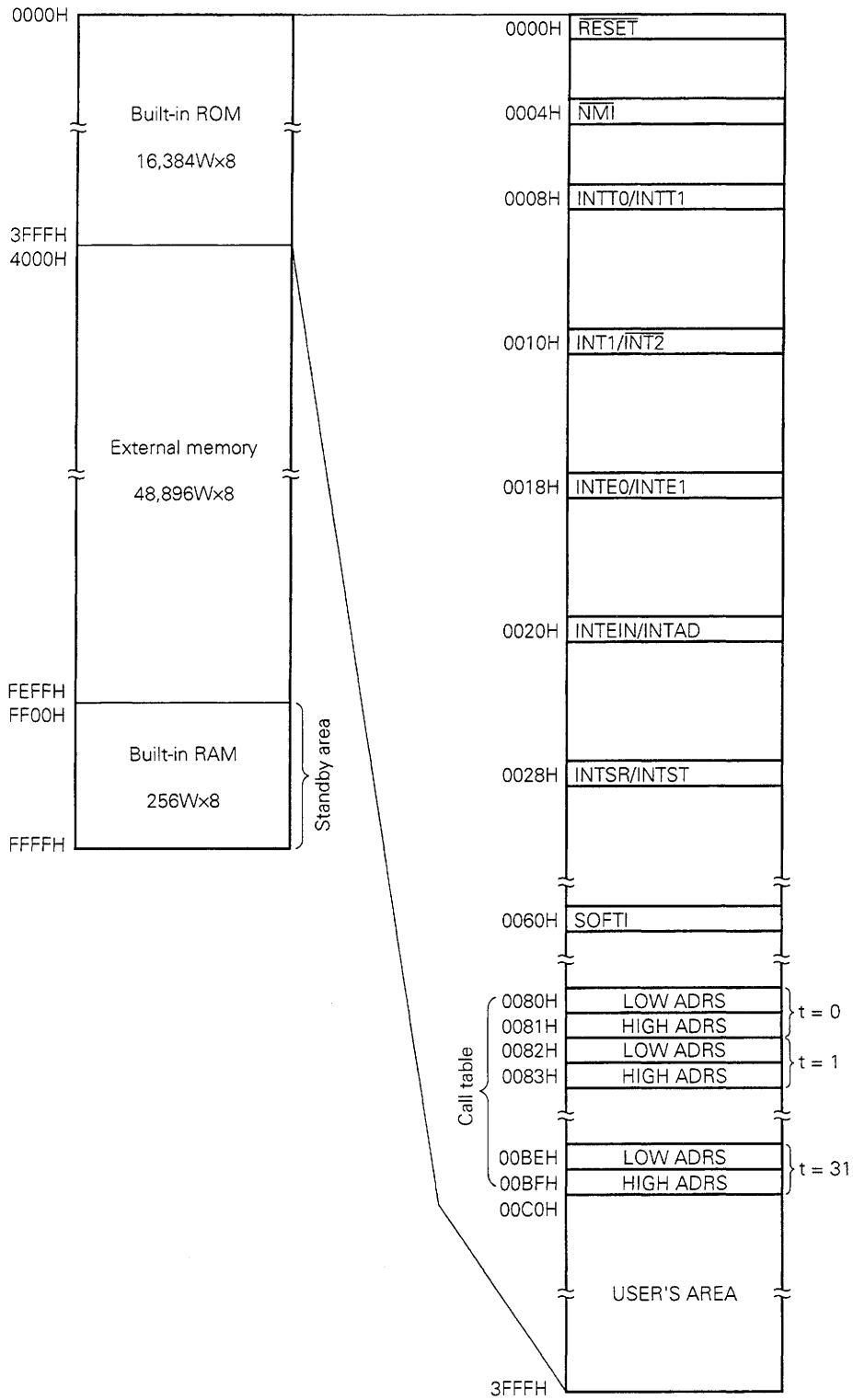
By setting the MEMORY MAPPING register, the 48-Kbyte area (addresses 4000H-FEFFFH) can be used to expand the external memory in steps.

The external memory is accessed with PD7-0 (multiplexed address/data bus), PF7-0 (address bus), and the \overline{RD} , \overline{WR} , and ALE signals. In the external memory, both programs and data can be stored.

(f) Working register area

It is possible to place the working registers (256 bytes long) at an arbitrary location in memory (specified by the V-register), thus enabling working register addressing.

Figure 2-3. Memory Map



2.5 PORT FUNCTIONS

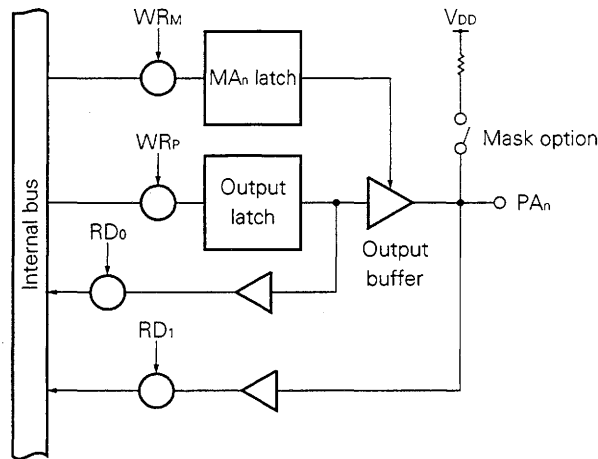
(1) PA7-0 (Port A)

PA7-0 (Port A) is an 8-bit input/output port equipped with input/output buffer and output latch functions. Port A can be set either as an input or output port in bit units by the MODE A register. Port A can also use the mask option to specify the pull-up resistor connection in bit units.

When set as an input port or on reset, Port A is placed in the following status:

- High impedance: without pull-up resistor
- High level : with pull-up resistor

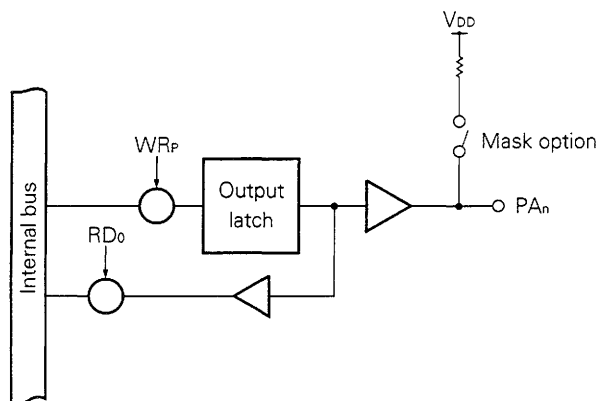
Figure 2-4. Port A



(a) When specified as an output port ($MA_n = 0$)

The output latch is made valid thus making it possible to use the transfer instruction to exchange data between the output latch and the accumulator. The bits of the output latch can be set or reset directly by arithmetic and logical operation instructions without going through the accumulator. Once data is written in the output latch, it is retained there until an instruction manipulating Port A is executed next or a system reset takes effect.

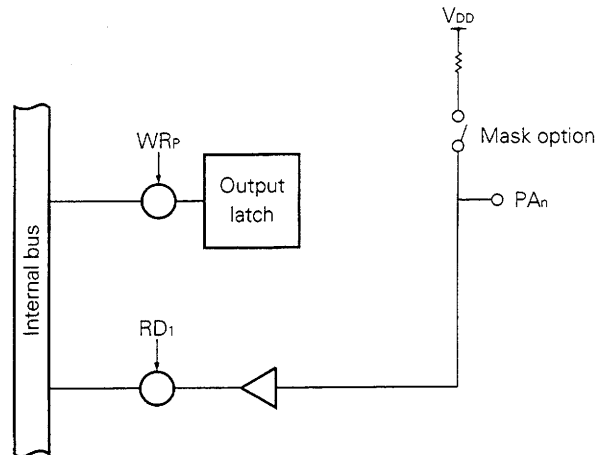
Figure 2-5. Port A Specified as Output Port



(b) When specified as an input port ($MA_n = 1$)

The contents of the PA run can be loaded into the accumulator by the transfer instruction. The contents of the PA lines can also be tested in bit units directly by arithmetic and logical operation instructions without going through the accumulator.

Figure 2-6. Port A Specified as an Input Port



Actual execution of instructions is performed in 8-bit units. If the instruction (MOV A, PA) for Port A is executed, the contents of the input lines of the port specified for input or the contents of the output latch of the port specified for output are loaded in the accumulator. When the instruction (MOV PA, A, etc.) for Port A is executed, writing is performed for the output latches of both the input-specified port and the output-specified port. However, because the contents of the output latch specified as an input port cannot be loaded from the accumulator and the output buffer is off, data cannot be output to the external pins (working as input pins).

•MODE A register (MA)

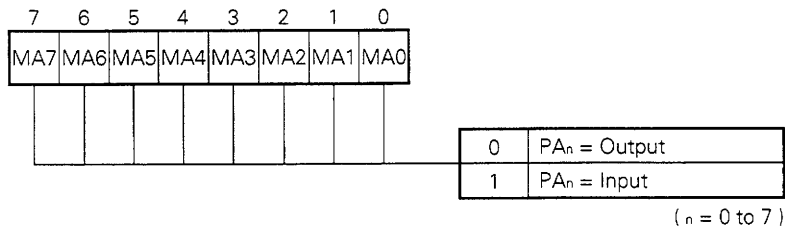
Refers to the 8-bit register which specifies the input/output status of Port A.

Specifying the input/output status of Port A is performed in bit units. The specification is input if the corresponding bit in the MODE A register is set (to 1); and output if reset (to 0).

At the time of $\overline{\text{RESET}}$ input or hardware STOP mode, all the bits are set so that Port A becomes an input port with the following status:

- High impedance : without pull-up resistor
- High level : with pull-up resistor

Figure 2-7. MODE A Register's Format



(2) PB7-0 (Port B)

In the same manner as Port A, Port B is an 8-bit input/output port equipped with input/output buffer and output latch functions. It can be set as either an input or output port in bit units by the MODE B register. It is also possible to specify the pull-up resistor connection in bit units by means of the mask option.

Port B is placed in the following status when set as an input port or on reset:

- High impedance: without pull-up resistor
- High level : with pull-up resistor

Port operation is the same as that of Port A. Thus, the contents of the port can be bit-set/reset and bit-tested directly by arithmetic and logical operation instructions without going through the accumulator. Also, data transfer is possible between the port and the accumulator.

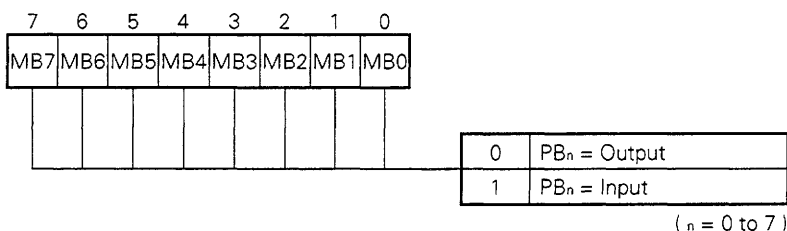
•MODE B register (MB)

In the same manner as the MODE A register of Port A, the MODE B register is an 8-bit register which specifies Port B's input/output in bit units.

At the time of RESET input or hardware STOP mode, all the bits are set (to 1) with Port B becoming an input port with the following status:

- High impedance: without pull-up resistor
- High level : with pull-up resistor

Figure 2-8. Format of MODE B Register



(3) PC7-0 (Port C)

Port C is an 8-bit special input/output port. In the same manner as Port A, it functions as a general-purpose input/output port capable of being set to input/output in bit units. Its bits are also defined as various control signals. Switching between these two functions is performed in bit units by specification of the MODE C register and the MODE CONTROL C register as shown below:

Table 2-2. PC7-0 Operation

	MCC _n = 1	MCC _n = 0	
	MC _n = X	MC _n = 0	MC _n = 1
PC0	TxD output	Output	Input
PC1	RxD input	Output	Input
PC2	SCK input/output	Output	Input
PC3	INT2/T1 input	Output	Input
PC4	TO output	Output	Input
PC5	CI input	Output	Input
PC6	CO0 output	Output	Input
PC7	CO1 output	Output	Input

(n = 0 to 7)

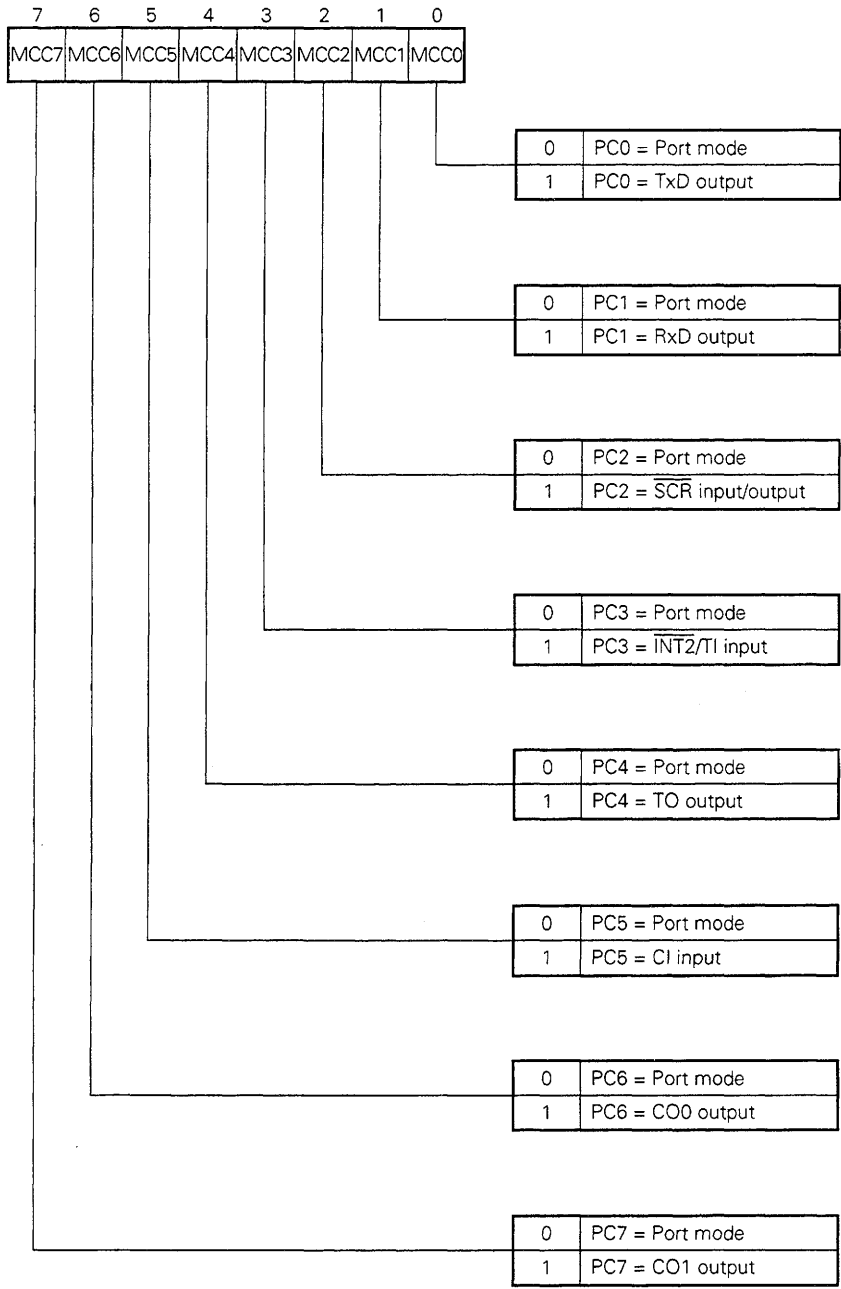
Specification of pull-up resistor connections can be performed in bit units by the mask option. When set as general-purpose input/output ports, the contents of the ports can be bit-set/reset and bit-tested by arithmetic and logical operation instructions, in the same manner as Port A. Also, data transfer is possible between the port and the accumulator.

• **MODE CONTROL C register (MCC)**

Refers to an 8-bit register which specifies Port C's port/control signal input/output mode in bit units. PC7-0 is placed in control signal input/output mode if MODE CONTROL C register's corresponding bit is set (to 1); and in port mode if reset (to 0).

At the time of $\overline{\text{RESET}}$ input or hardware STOP mode, all the bits of the MODE CONTROL C register are reset (to 0), thus being placed in port mode.

Figure 2-9. Format of MODE CONTROL C Register



• **MODE C register (MC)**

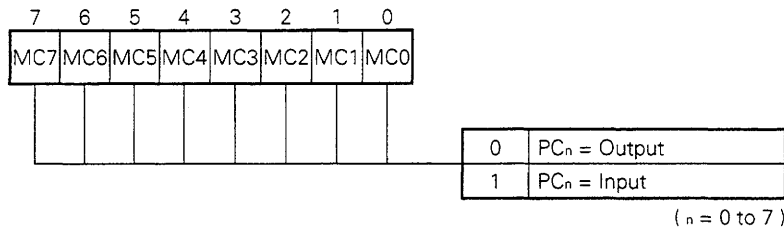
The MODE C register is an 8-bit register specifying the input/output status of Port C in bit units, in the same manner as the MODE A register of Port A.

The bits of the MODE C register corresponding to those bits placed in control mode by the MODE CONTROL C register are ignored.

At the time of RESET input or hardware STOP mode, all the bits of the MODE C register are set (to 1); and, as all the bits of the MODE CONTROL C register are reset (to 0), Port C becomes an input port and is placed in the following status:

- High impedance: without pull-up resistor
- High level : with pull-up resistor

Figure 2-10. MODE C Register's Format



(4) PD7-0 (Port D)

Refers to the 8-bit general-purpose input/output port that is also used as a multiplexed address/data bus. In addition to its ability to be specified for input/output as a general-purpose input/output port as a byte (8-bit) unit, this port functions as the multiplexed address/data bus when connecting an external expanded memory. The switchover between these states is performed by the MEMORY MAPPING register.

Operation by this port when set as a general-purpose input/output port is the same as Port A except for the fact that, with Port D, the input/output specification is performed in a whole byte unit. Thus, the contents of the port can be bit-set/reset and bit-tested directly by the arithmetic and logical operation instructions without going through the accumulator. Data transfer between port and the accumulator is also possible.

(5) PF7-0 (PortF)

Refers to an 8-bit general-purpose input/output port which serves as an address bus as well.

In addition to its ability to specify input/output in bit units as a general-purpose input/output port, Port F outputs address signals in accordance with the size of the external expansion memory when accessing an external expansion memory larger than 256 bytes.

This switchover is performed by the MEMORY MAPPING register and the MODE F register.

PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0	External Memory
Port	Port	Port	Port	Port	Port	Port	Port	Up to 256 bytes
Port	Port	Port	Port	AB11	AB10	AB9	AB8	Up to 4 Kbytes
Port	Port	AB13	AB12	AB11	AB10	AB9	AB8	Up to 16 Kbytes
AB15	AB14	AB13	AB12	AB11	AB10	AB9	AB8	Up to 48 Kbytes

Operation by this port when set as a general-purpose input/output port is the same as that of Port A. Thus, the contents of the port can be bit-set/reset and bit-tested directly by the arithmetic and logical operation instructions without going through the accumulator.

Data transfer between the port and the accumulator is also possible.

• **MEMORY MAPPING register (MM)**

Refers to the 4-bit register that controls whether to enable specification of PD7-0 and PF7-0 port/extend mode and access to the built-in RAM.

The MEMORY MAPPING register's bits 0, 1, and 2 (MM0, MM1, MM2) control specification of PD7-0's port/extend mode and input/output as well as PF7-0's address line.

When the MEMORY MAPPING register's MM1 and 2 bits are 0, both PD7-0 and PF7-0 become general-purpose input/output ports. MM0 specifies input/output of PD7-0, while the MODE F register specifies input/output of PF7-0.

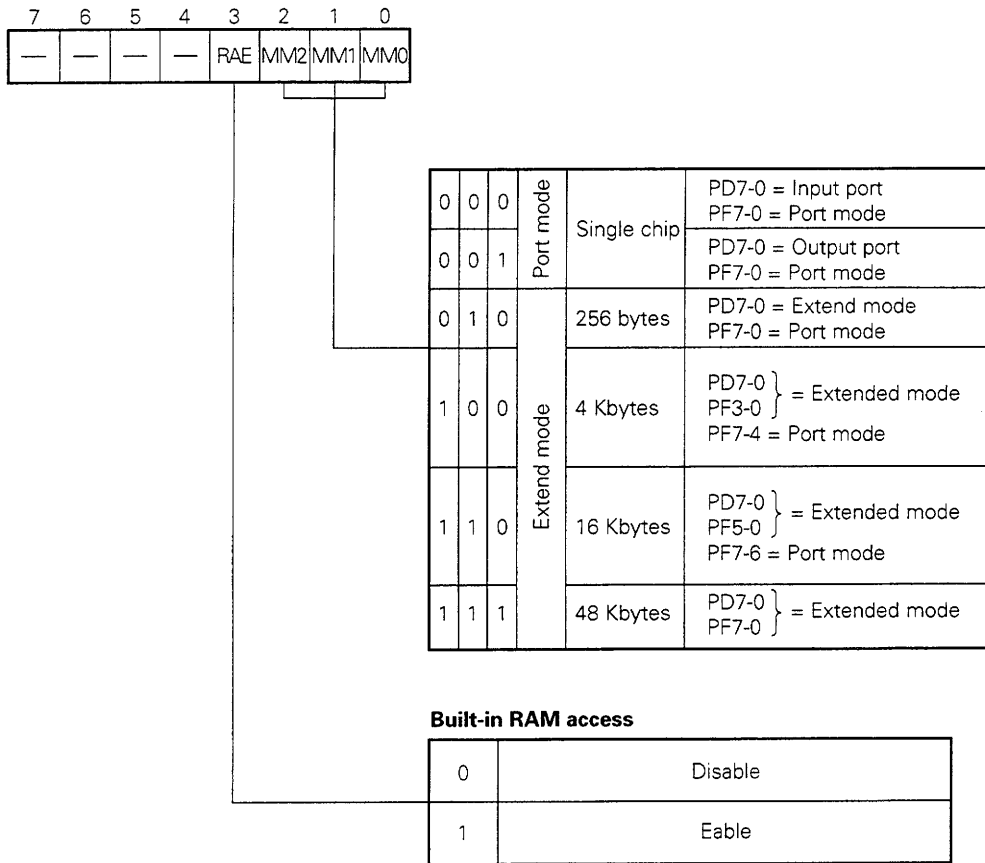
Four types of external expanded memory, viz., 256 bytes, 4 Kbytes, 16 Kbytes and 48 Kbytes, as shown in Figure 2-11, can be selected. The ports of PF7-0 unused as address lines can be used as general-purpose input/output ports.

The MEMORY MAPPING register's bit-3 (RAE) controls whether to enable access to the built-in RAM. When the built-in RAM is not used for external expansion but the memory externally connected uses this space, the RAE bit is set to 0 to disable access to the built-in RAM.

At the time of $\overline{\text{RESET}}$ input or hardware STOP mode, the MEMORY MAPPING register's MM0, 1, and 2 bits are reset (to 0) and PD7-0 becomes an input port (output high impedance).

The RAE bit retains its contents even when the $\overline{\text{RESET}}$ signal is input during normal operation. However, the RAE bit is undefined at the time of power-ON resetting; therefore, it is necessary to use an instruction to initialize it.

Figure 2-11. Format of MEMORY MAPPING Register

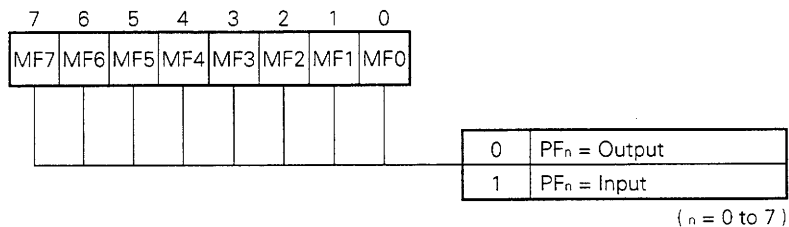


•MODE F register (MF)

In the same manner as Port A's MODE A register, the MODE F register specifies Port F's input/output status. However, the bits of the MODE F register corresponding to bits in Port F specified as address lines by the MEMORY MAPPING register are set to output mode.

At the time of $\overline{\text{RESET}}$ input or hardware STOP mode, all the bits of the MODE F register are set (to 1) and Port F becomes an input port (output high impedance).

Figure 2-12. Format of MODE F Register



2.6 TIMERS

These refer to the two 8-bit interval timers (TIMER0, TIMER1), which are programmable independently of each other. They can also be used as a 16-bit interval timer with a cascaded connection. Furthermore, they can count TI inputs.

As shown in Figure 2-13, each of TIMER0 and TIMER1 consists of an 8-bit TIMER REG (TM0, 1), an 8-bit COMPARATOR, an 8-bit UPCOUNTER, and TIMER F/F. Input selection, timer operation, and TO output are controlled by the timer mode register (TMM).

TIMER0's inputs include the internal clocks ϕ_{12} (1 μ s: 12 MHz operation) and ϕ_{384} (32 μ s: 12 MHz operation) as well as the TI input. TIMER1's inputs include not only ϕ_{12} , ϕ_{384} and the TI input, as with TIMER0, but also the coincidence signal of TIMER0 as well.

As both TIMER0 and TIMER1 perform the same operation, the operation of TIMER0 is described here.

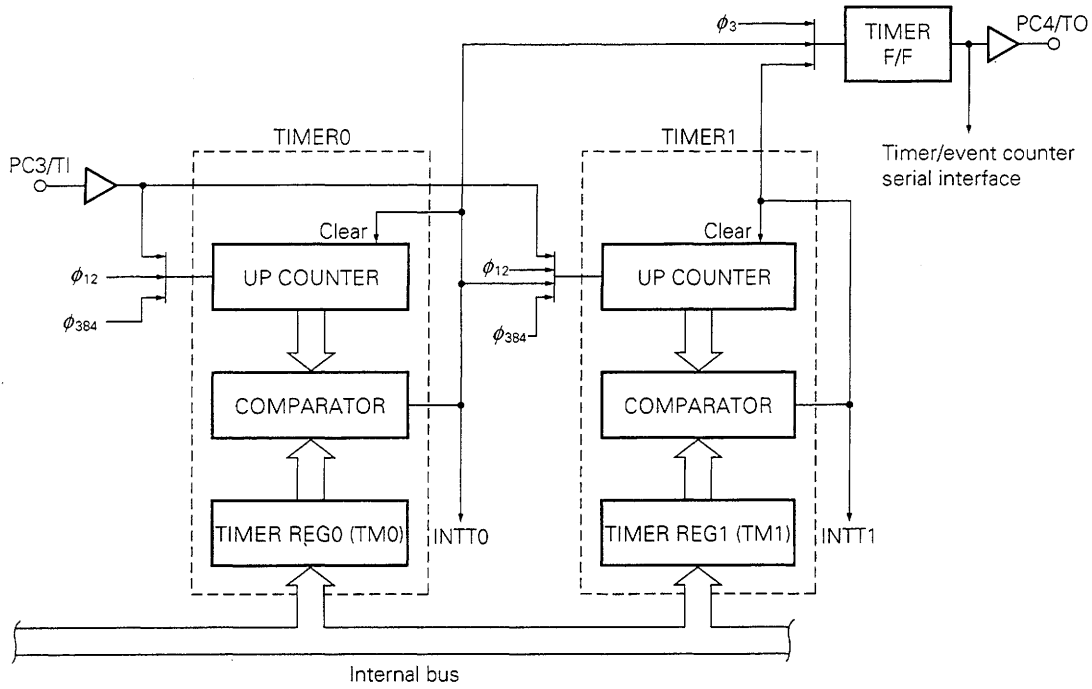
First of all, start TIMER0 by setting a count value in TIMER REG0, then setting the data for TIMER0 input, and starting TIMER0 (timer mode register's bit-4 = 0). The UPCOUNTER counts up each input; the COMPARATOR always compares the counted-up contents of the UPCOUNTER with those of TIMER REG0 and issues a coincidence signal (internal interrupt: INTT0) if they agree with each other. If agreement occurs, the contents of the UPCOUNTER are cleared to start the counting up again starting from 00H. Accordingly, TIMER0 operates as interval timer generating interrupts repeatedly using the count value that has been set in TIMER REG0 as an interval.

By setting (to 1) the interrupt mask register (MKL)'s bit-1 (MKT0), the internal interrupt (INTT0) is disabled.

Since the TO output is equipped with TIMER F/F which reverses for each coincidence signal of the timer's comparators or for each internal clock of ϕ_3 (when operating at 250 ns: 12 MHz), it can produce a square wave whose half-period is the count time or ϕ_3 . This output can be used as the reference time for the timer/event counter, by setting the timer/event counter mode register (ETMM).

By setting the serial mode register (SMH), it can also be used as the serial clock ($\overline{\text{SCK}}$) of the serial interface.

Figure 2-13. Timer's Block Diagram



- Remarks**
1. $\phi_3 = f_{xx} \times \frac{1}{3}$
 2. $\phi_{12} = f_{xx} \times \frac{1}{12}$ provided that f_{xx} = oscillation frequency (MHz)
 3. $\phi_{384} = f_{xx} \times \frac{1}{384}$

(1) Timer mode register (TMM)

Refers to the 8-bit register which controls operation of TIMER0, TIMER1, and TIMER F/F (see Figure 2-14).

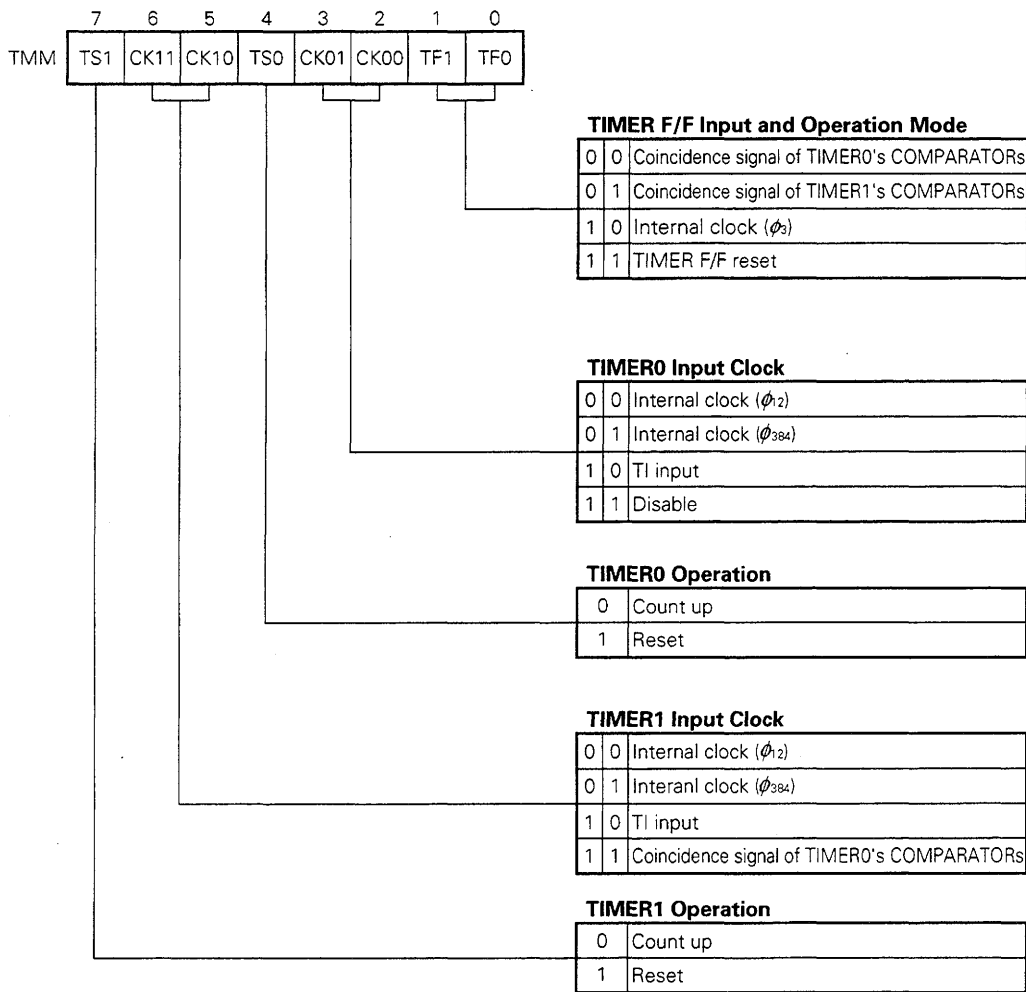
Bits-0 and 1 (TF0 and 1) of the timer mode register control the operation mode of TIMER F/F; bits-2 and 3 (CK00 and CK01) control the input clock of TIMER0; and bit-4 (TS0) controls the operation of TIMER0. Bits-5 and 6 (CK10 and CK11) control the input clock of TIMER1; and bit-7 (TS1) controls the operation of TIMER1.

Bits TS0 and 1 clear the respective UPCOUNTER to 00H at "1" and stop the count-up. When changed from "1" to "0", UPCOUNTER starts the count-up from 00H.

The internal clock ϕ_3 results from dividing the oscillation frequency by 3; the internal clock ϕ_{12} results from dividing it by 12; and the internal clock ϕ_{384} results from dividing it by 384.

As a result of the RESET input, the timer mode register is set to FFH; the UPCOUNTERs of both TIMER0 and TIMER1 are cleared and placed in the halt state; and the TIMER F/F is reset.

Figure 2-14. Format of Timer Mode Register (TMM)



2.7 TIMER/EVENT COUNTER

The μ PD78C14A is equipped with 16-bit multi-function timer/event counter which performs the following operations.

- Interval timer
- External event counter
- Frequency measurement
- Pulse width measurement
- Programmable rectangular wave output
- One-pulse output

The timer/event counter consists of the 16-bit TIMER/EVENT UPCOUNTER (ECNT), the TIMER/EVENT COUNTER CAPTURE register (ECPT), the COMPARATOR, the TIMER/EVENT COUNTER REG0 and 1 (ETM0, ETM1) and various control circuits for input/output, interrupts, and clearing.

The ECNT is a 16-bit upcounter that counts input pulses, which are then cleared by the clear control circuit.

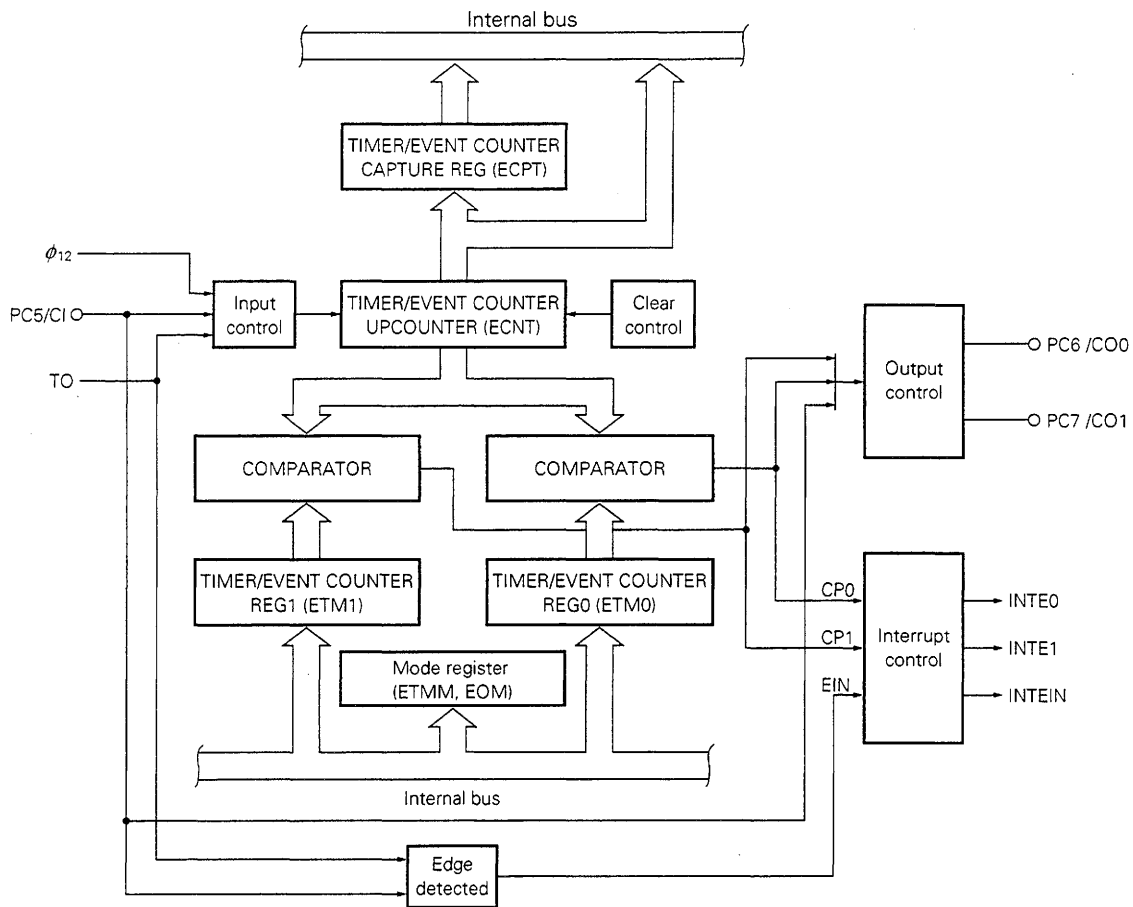
The ECPT register is a 16-bit buffer register which retains the ECNT's contents. The timing of the ECPT register's latching of the ECNT's contents is performed at the falling edge of the CI input when the input to the ECNT is the internal clock; and at the falling edge of the TO output when the input to the ECNT is the CI input.

The ETM0 and ETM1 registers are the two 16-bit registers that set the number of counts. The 16-bit data transfer instruction is used to exchange data with the extended accumulator.

The COMPARATOR compares the contents of ECNT and those of the ETM0 or ETM1 register and generates a coincidence signal for those detected to be coinciding.

The interrupt control circuit, which controls the interrupts from the timer/event counter, generates the following three types of interrupt causes: coincidence signal (INTE0) between ECNT and ETM0 registers; coincidence signal (INTE1) between ECNT and ETM1 registers; and the falling edge (INTEIN) of the CI input or timer output (TO).

Figure 2-15. Block Diagram of Timer/Event Counter



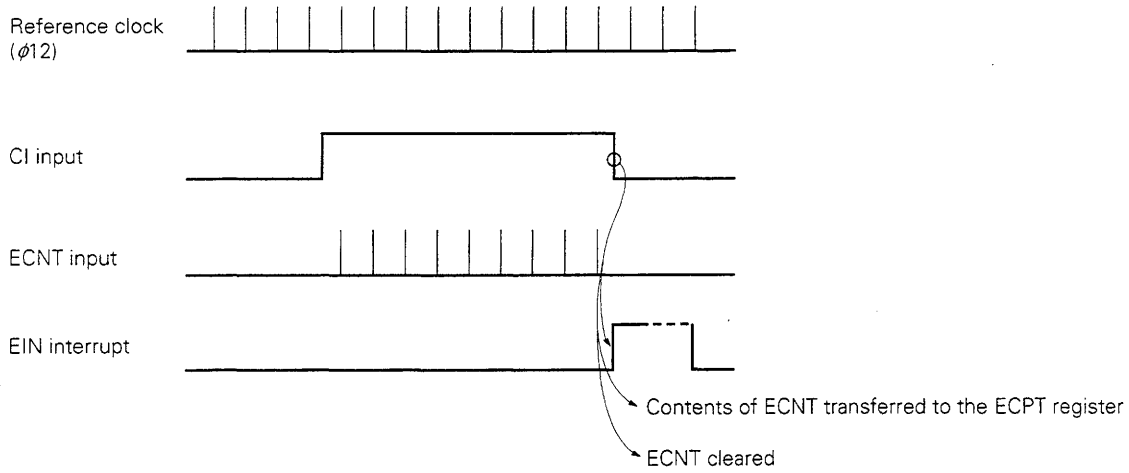
Remark $\phi_{12} = f_{xx} \times \frac{1}{12}$, provided that f_{xx} = oscillation frequency (MHz)

Measurement of pulse width is used as an example below to describe the operation of the timer/event counter.

This operation is performed to measure the width of the high level of an external pulse that is input into CI by setting the timer/event counter mode register (ETMM) to 09H.

ECNT continues counting the internal clocks (ϕ_{12}) while CI maintains the high level. If the external pulse input to CI decays, the contents of ECNT are transferred to the ECPT register, ECNT is cleared, and the internal interrupt (INTEIN) is generated (see Figure 2-16). Therefore, the pulse width can be measured using the contents of the ECPT register and the cycle of internal clocks.

Figure 2-16. Pulse Width Measurement



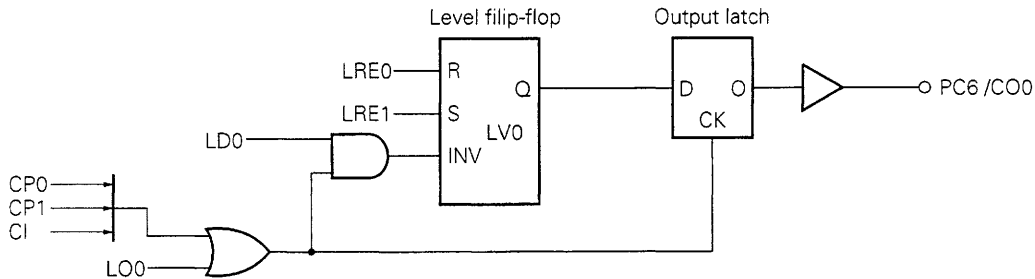
The μPD78C14A is equipped with an output control circuit which outputs pulses with variable pulse width and cycle, interlocked with the timer/event counter.

The outputs of the output control circuit include the CO0 output and the CO1 output. As they are based on the same configuration, the CO0 output is described here. Figure 2-17 shows CO0's configuration. The CO0 output is of the master-slave type, in which the first level F/F (LV0) retains the level that is output next, and the output latch of the next level outputs the level of LV0 externally.

LV0 can be set/reset by setting the timer/event counter output mode register (EOM). Furthermore, LV0 is equipped with a level inversion pin (INV), and its level can be inverted with the output timing based on setting the timer/event counter mode register.

The timing for the output latch to output LV0's level is that created by setting the timer/event counter mode register.

Figure 2-17. Output Control Circuit



The operation of outputting a square wave to the CO0 pin is described below.

First of all, clear ECNT. Then, set the count value (ETM0 < ETM1) in the ETM0 and ETM1 registers. Place in the timer/event counter output mode register the data for enabling the specification of LV0's initial status and its level inversion.

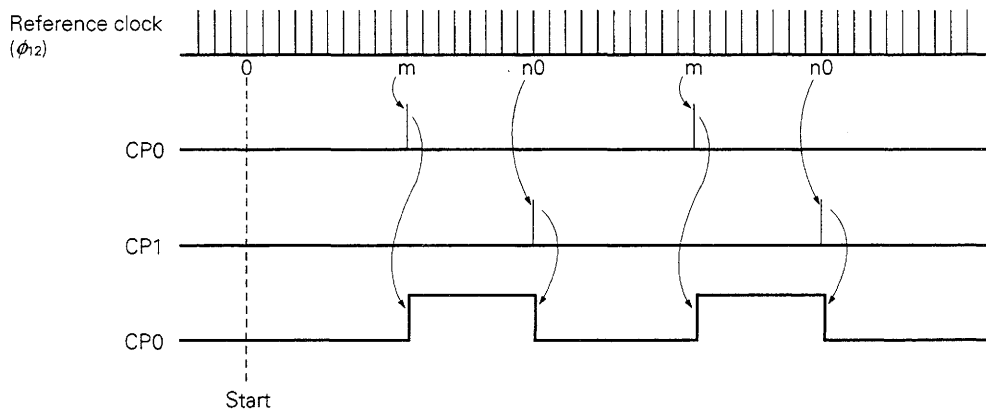
The timer/event counter operation is started by placing in the timer/event counter mode register data such that the input to ECNT is the internal clock ϕ_{12} (when operating at $1 \mu\text{s}$: 12 MHz), the clear mode of ECNT is the coincidence signal of the ECNT and ETM1 registers, and the output timing to the CO0 pin is the coincidence signal of the ECNT and ETM0 registers or the coincidence signal of the ECNT and ETM1 register.

ECNT is counted up for each internal clock of ϕ_{12} . The COMPARATOR compares the counted-up ECNT with the ETM0 and ETM1 registers and, upon detecting a coincidence, issues a coincidence signal (CP0, CP1). Each coincidence signal is used to output LV0's level to the CO0 pin and invert LV0.

The coincidence signal (CP1) of the ECNT and ETM1 registers is used to clear ECNT, which then starts again to perform the count-up from 0000H. This sequence is repeated again and again (see Figure 2-18).

Accordingly, it is possible to output programmable square waves whose pulse widths are the count values of the ETM0 and ETM1 registers.

Figure 2-18. Square Wave Output



Remark ETM0 register = m
 ETM1 register = n (m < n (m & n refer to count values))

(1) Timer/event counter mode register (ETMM)

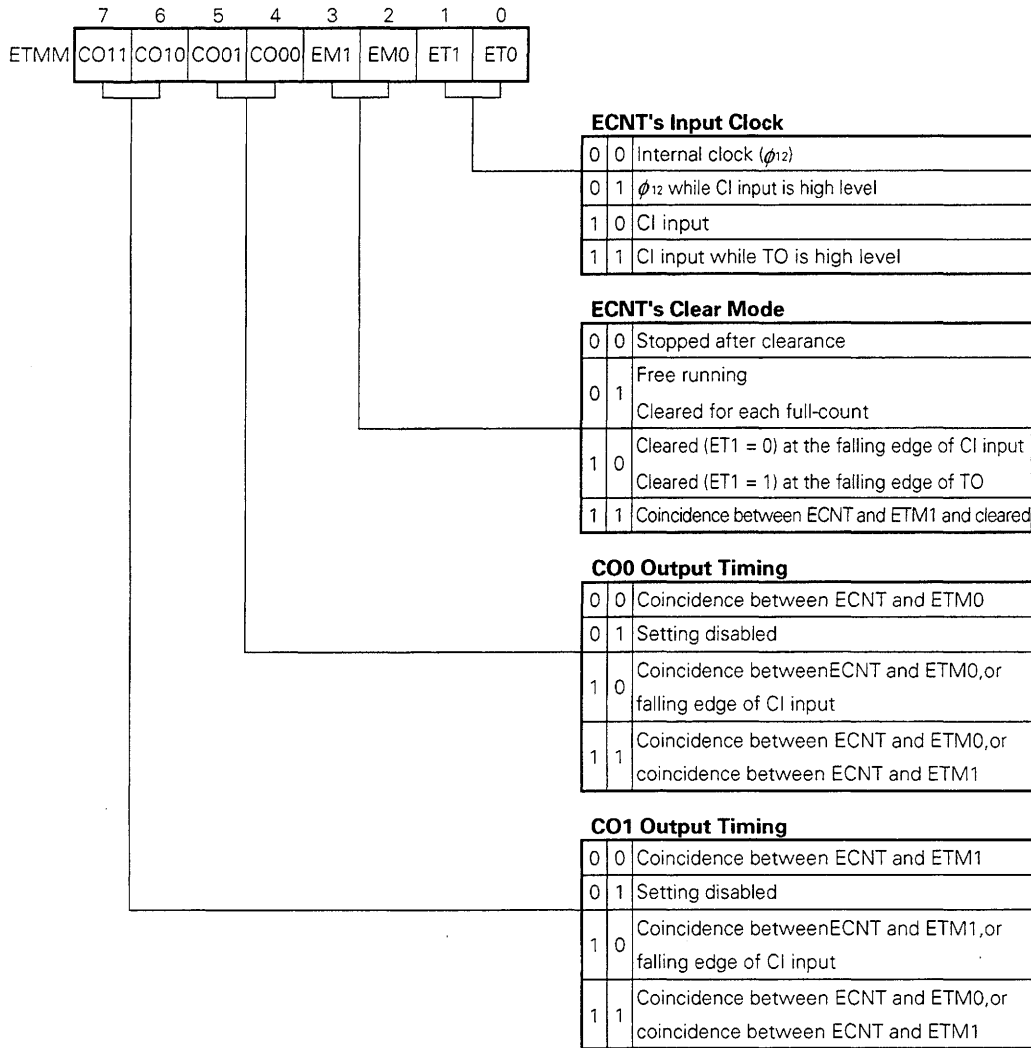
Refers to the 8-bit register that controls the timer/event counter (see Figure 2-19).

Bits-0 and 1 (ET0 and 1) of the timer/event counter mode register control the input clock of the TIMER/EVENT COUNTER UPCOUNTER (ECNT). Bits-2 and 3 (EM0, 1) control ECNT's clear mode. Bits-4 and 5 (CO00, CO01) control the timing of outputting the contents of the output latch to the COUNTER OUTPUT0 (CO0). Bits-6 and 7 (CO10, CO11) control the CO1's output timing.

The internal clock (ϕ_{12}) is based on the oscillation frequency divided by 12.

At the time of RESET input or hardware STOP mode, the timer/event counter mode register is reset to 00H.

Figure 2-19. Format of Timer/Event Counter Mode Register



(2) Timer/event counter output mode register (EOM)

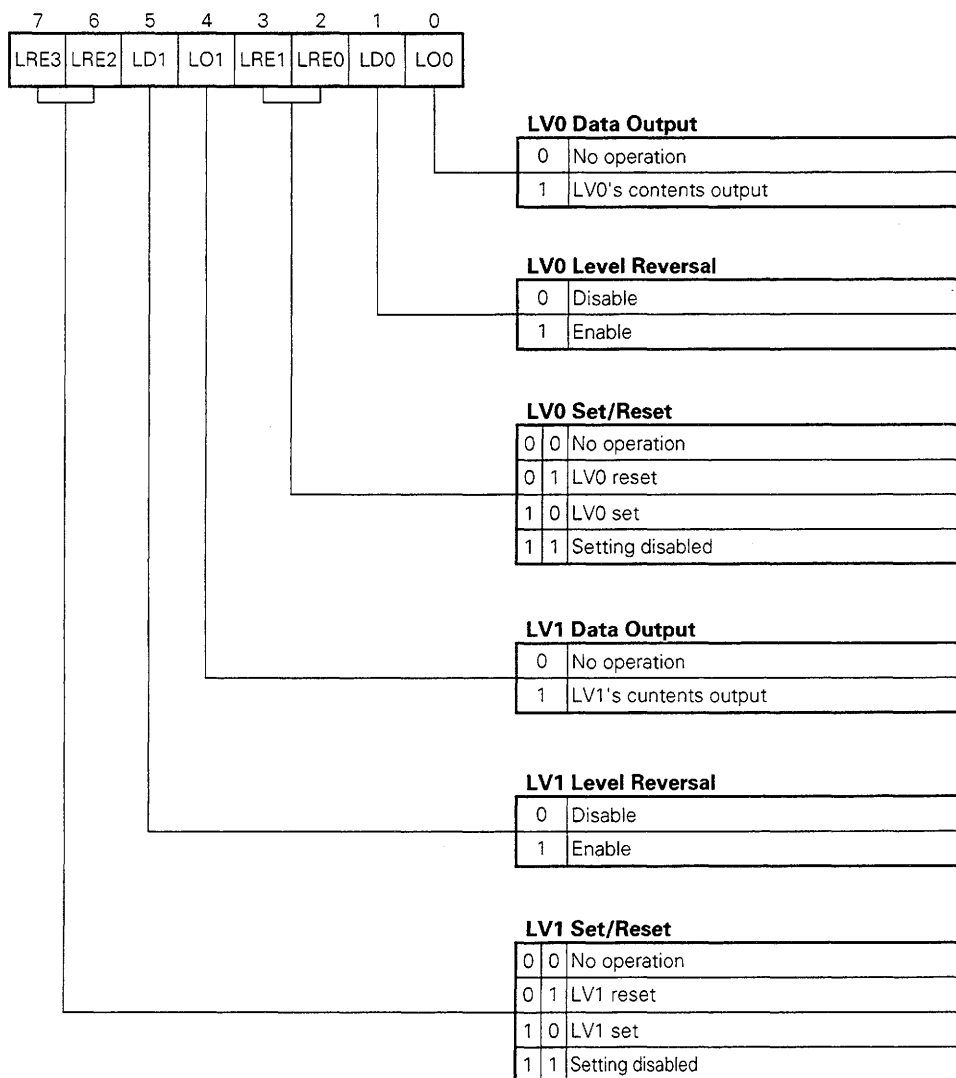
Refers to the 8-bit register that controls the operation modes of the timer/event counter's CO0 and CO1 (counter outputs 0 and 1).

Bits-0 and 4 (LO0 and 1) of the timer/event counter output mode register control whether to output the LV0 and 1 levels to the CO0 and 1 pins or not. Bits-1 and 5 (LD0 and 1) control whether to invert the LV0 and 1 levels or not in accordance with the timing specified by the timer/event counter mode register. Bits-2, 3, 6, and 7 (LRE0, 1, 2, and 3) control the set/reset of LV0 and 1.

The various bits of LO0, LO1, LRE0, LRE1, LRE2, and LRE3 are automatically reset (to 0) after terminating the respective operations.

At the time of RESET input or hardware STOP mode, the timer/event counter output mode register is reset to 00H.

Figure 2-20. Format of Timer/Event Counter Output Mode Register (EOM)



2.8 SERIAL INTERFACE

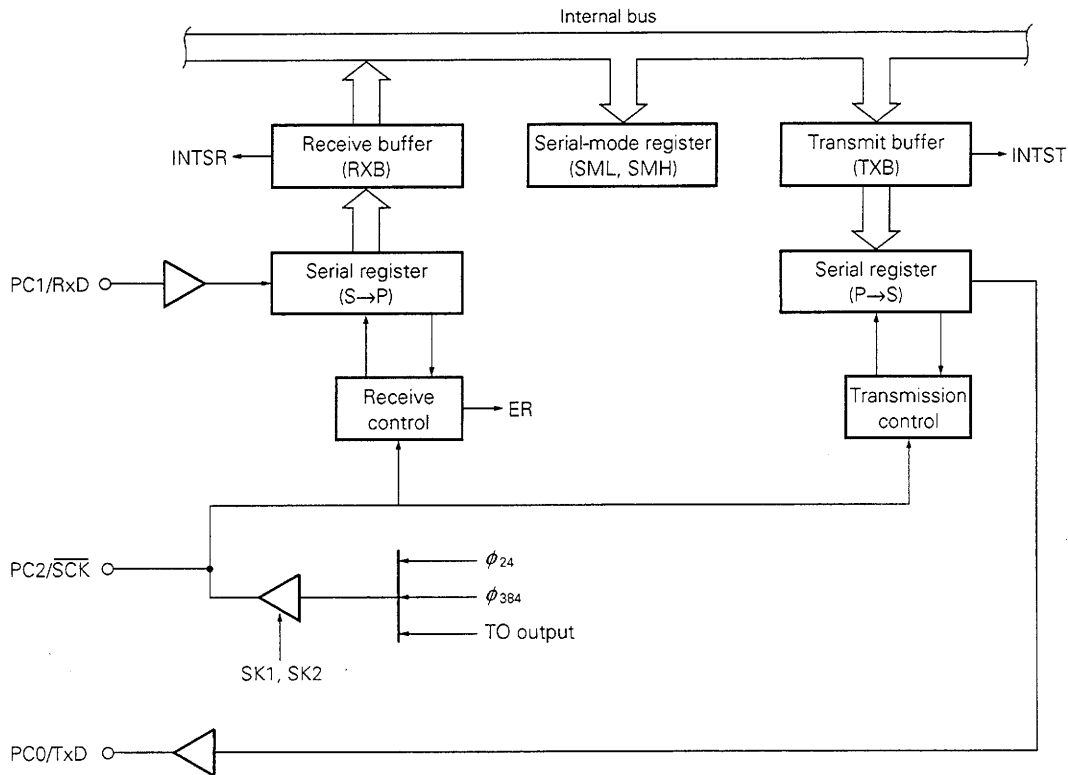
The μPD78C14A is equipped with a serial interface that has three types of operation modes, viz., asynchronous mode, in which data bit synchronization and character synchronization are made by the start bit in the start-stop-bit-based send/receive system; the synchronous mode, in which data transfer is made synchronously with the serial clock; and I/O interface mode, in which data transfer is made synchronously with the serial clock, which is controlled in the same manner as the serial data transfer system for μPD7801 and μPD78C06A.

The serial interface part consists of three pins, viz., serial data input (RxD), serial data output (TxD), and serial clock input/output (\overline{SCK}); two 8-bit serial registers for the transfer controller for sending and receiving; and an 8-bit transmit buffer and receive buffer (see Figure 2-21).

Equipped with the respective serial registers and buffers for both the transmission and the reception of data, the μPD78C14A is capable of performing transmission and reception independently of each other (full-duplex, double-buffer system transmitter receiver).

However, as the serial clock (\overline{SCK}) is shared for both transmitting and receiving, the machine becomes half-duplex in synchronous mode or in I/O interface mode.

Figure 2-21. Serial Interface Block Diagram



Remark $\phi_{24} = f_{xx} \times \frac{1}{24}$

$\phi_{384} = f_{xx} \times \frac{1}{384}$, provided that f_{xx} = oscillation frequency (MHz)

(1) Asynchronous mode

In the case of asynchronous mode, specification of the clock rate, the character length, the number of stop bits, the parity enable, and the odd/even parity can be controlled by the serial-mode register (SML).

The send operation is enabled by setting (to 1) bit-2 (TxE) of the serial mode register (SMH).

Data is written by the MOV TXB, A instruction into the send buffer and, if the preceding data transfer is finished, the contents of the transmit buffer are automatically transferred to the serial register. The data that has been transferred to the serial register is automatically completed by including the start bit (1 bit), the parity bit (odd, even, or no parity), and the stop bit(s) (1 or 2 bits) and then sent out the TxD pin starting from the least significant bit (LSB).

If the transmit buffer becomes empty, an internal interrupt (INTST) occurs.

The send data is sent from the TxD pin at the SCK's falling edge at one of the serial clock ($\overline{\text{SCK}}$)'s $\times 1$, $\times \frac{1}{16}$, and $\times \frac{1}{64}$ clock rates.

The maximum transmission data speeds at 12 MHz vary as follows based on the $\overline{\text{SCK}}$ and clock rates.

Clock Rate \ $\overline{\text{SCK}}$	Internal Clock		External Clock	
	$\overline{\text{SCK}}$	Data rate	$\overline{\text{SCK}}$	Data rate
$\times 1$	500 kHz	500 kbps	660 kHz	660 kbps
$\times 16$	2 MHz	125 kbps	2 MHz	125 kbps
$\times 64$		31.25 kbps		31.25 kbps

The TxD pin is placed in the mark state (1) when TxE is 0 or when the serial register does not wait for the data transmitted. The internal interrupt (INTST) is disabled by setting bit-2 (MKST) of the interrupt mask register (MKH).

Figure 2-22. Asynchronous Data Format

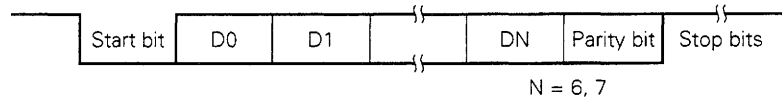
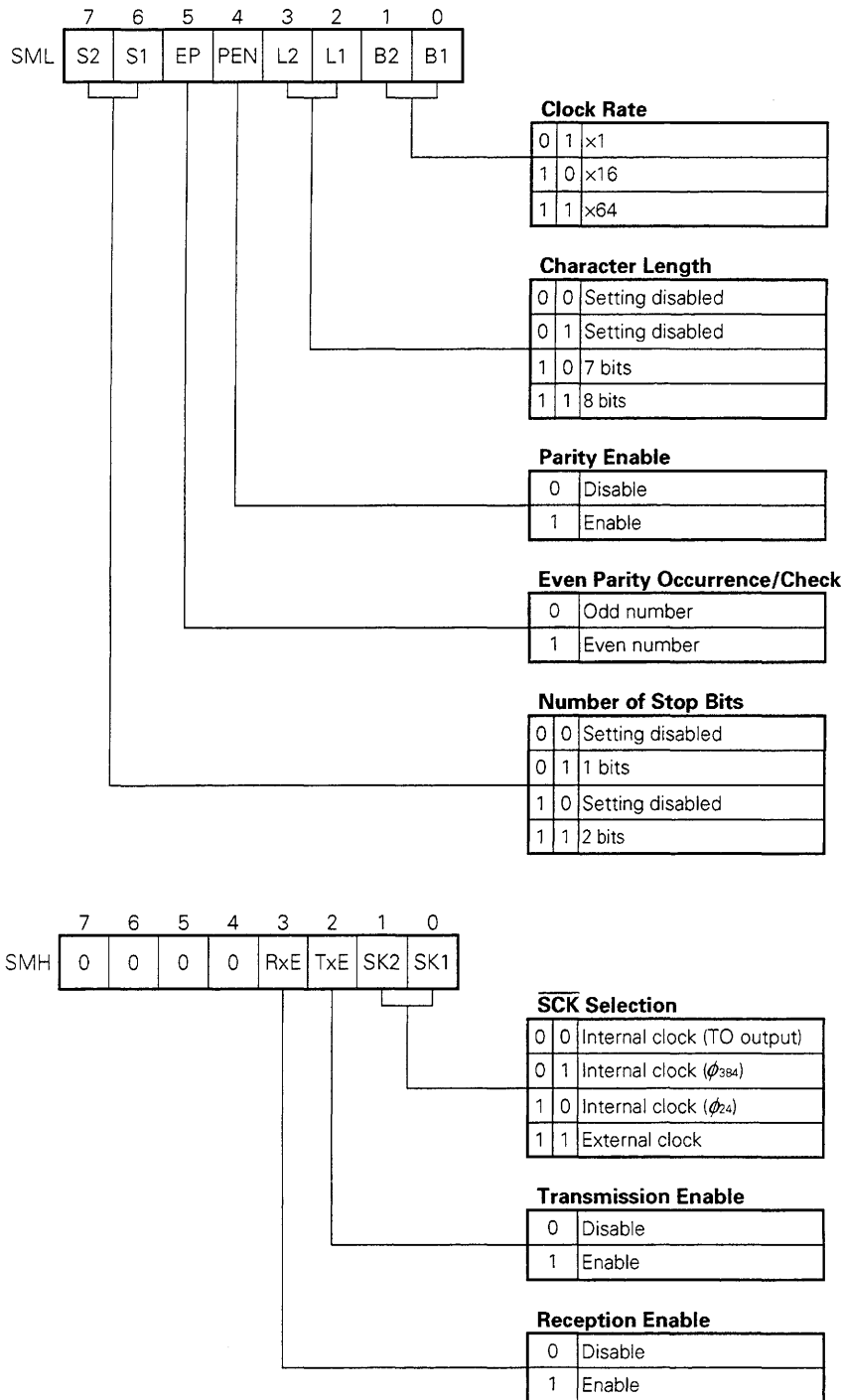


Figure 2-23. Serial Mode Register Format in Asynchronous Mode



The receive operation is enabled by setting bit-3 (RxE) (1) of the serial mode register (SMH).

The start bit detects the low level of the RxD input, and, furthermore, it confirms the low level by detecting it in 1/2-bit time. Then, reception is performed by sampling at the middle of the character bit, parity bit and stop bit(s) that follow. When the data has entered the serial register from RxD, the data is transferred to the receive buffer. When the receive buffer becomes full, an internal interrupt (INTSR) occurs.

The internal interrupt is disabled by setting (to 1) bit-1 (MKSR) of the interrupt mask register (MKH).

On receipt, the odd/even parities are checked (PEN bit = 1) and the error flag is set (to 1) if there is a discrepancy (parity error), if the stop bit is low (framing error), or if the following data is transferred to the receive buffer when the receive buffer is full (overrun error).

However, as an error interrupt mechanism is not available, testing is performed by the program by means of skip instructions (SKIT, SKNIT).

The serial clock (\overline{SCK}) that can be selected by the serial mode register (SMH) is either the external clock or the internal clock.

An internal clock can be selected from among three types, i.e., ϕ_{24} , ϕ_{384} , and the TO output to be output to the outside. It is also possible to input the serial clock from the outside.

The data rate can be freely changed by the program by using an internal clock (TO output) as the \overline{SCK} .

The maximum data rate on receipt varies as follows depending on the \overline{SCK} and the clock rate when operated at 12 MHz.

Clock Rate \ \overline{SCK}	Internal Clock		External Clock	
	\overline{SCK}	Data rate	\overline{SCK}	Data rate
x 1 Note 2	500 kHz	500 kbps	660 kHz 1 MHz	660 kbps 1 Mbps Note 1
x 16	2 MHz	125 kbps	2 MHz	125 kbps
x 64		31.25 kbps		31.25 kbps

- Notes**
1. When receiving data at a transfer rate of 660 kbps to 1 Mbps, two stop bits are required.
 2. If the clock rate is x1, it is necessary for RxD and \overline{SCK} to be synchronized.

For example, in the case of sending/receiving data at a transfer rate of 110 to 9600 bps, the timer's count value (C) is as follows when an internal clock (ϕ_{12}) is used as the timer's input clock.

Data Rate (bps) \ Oscillation Frequency (MHz)	7.3728		11.0592				
	N	16	64	16	64		
9600	C =	2	-	C =	3	-	
4800		4	C =	1	6	-	
2400		8		2	12	C =	3
1200		16		4	24		6
600		32		8	48		12
300		64		16	96		24
150		128		32	192		48
110		175		44	262		65

(2) Synchronous mode

In synchronous mode, data is transferred with the character length fixed to 8 bits and without parity bits. Therefore, please set the serial mode register (SML) to 0 CH (see Figure 2-24).

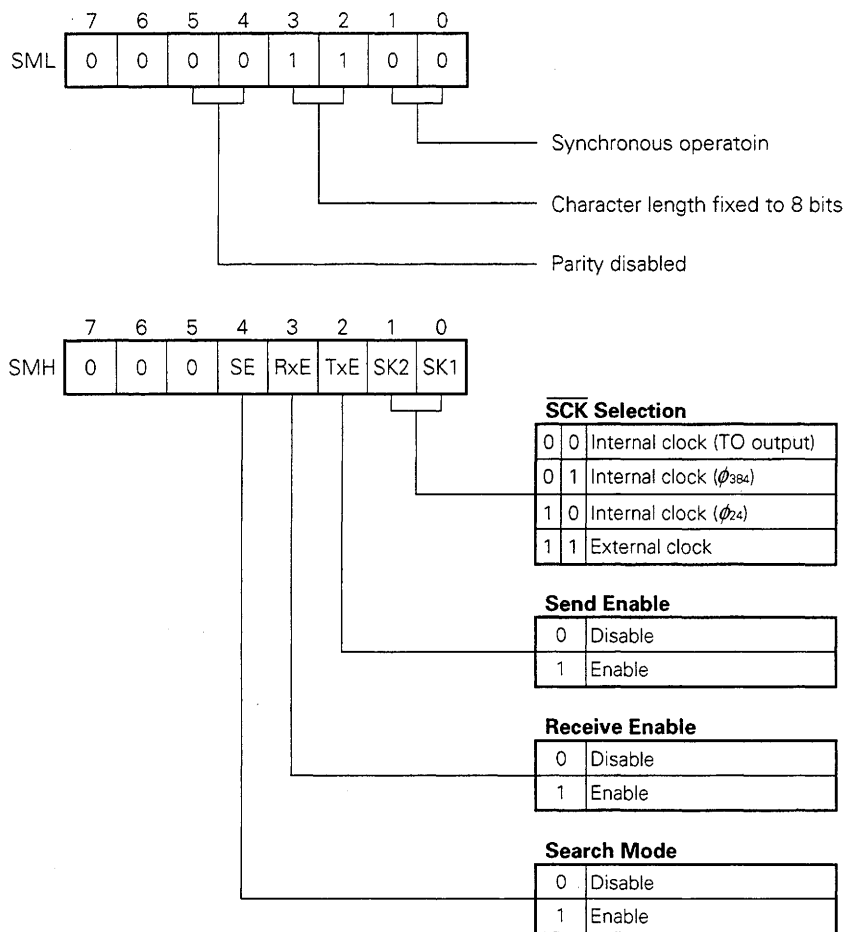
The send operation is enabled by setting (to 1) bit-3 (TxE) of the serial mode register (SMH).

Data is written by the MOV TXB, A instruction into the transmit buffer and, when the preceding data transfer is finished, the contents of the transmit buffer are automatically transferred to the serial register and converted to serial data to be automatically transmitted from the LSB to TxD synchronously with the \overline{SCK} 's falling edge. The serial data is sent at the same rate as the \overline{SCK} . The data rate at the time of transmission is up to 500 kbps when an internal clock is used as the SCK; and up to 1 Mbps when using an external clock as the \overline{SCK} (when operated at 12 MHz).

Data is transferred from the transmit buffer to the serial register; when the transmit buffer is emptied, an internal interrupt (INTST) is generated.

The TxD pin is placed in the mark state (1) when the TxE is 0 or when the serial register does not contain data to send.

Figure 2-24. Serial Mode Register Format in Synchronous Mode



In synchronous mode, there are two types of receive operations available which are controlled by the SE bit of the serial mode register (SMH).

The mode is placed in search mode by setting (to 1) the SE bit. In this mode, each time 1 bit is received from the RxD pin, not only are the contents of the serial register transferred to the receive buffer but an internal interrupt (INTSR) occurs as well. Since the μ PD78C14 is not equipped with a synchronous character detection circuit, it is necessary to detect synchronous characters via software.

If synchronous characters are detected and the reception has been synchronized, the SE bit is reset (to 0). By resetting the SE bit, the mode is switched to character receive mode, thus allowing the contents of the serial register to be transferred to the receive buffer; an internal interrupt (INTSR) occurs each time 8 bits of data are received.

By setting (to 1) the MKSR bit of the interrupt mask register (MKH), the internal interrupt (INTSR) is disabled.

In synchronous mode, data is output from TxD at $\overline{\text{SCK}}$ 'S falling edge; and data is input from RxD at the $\overline{\text{SCK}}$'s rising edge.

Either the internal clock or the external clock can be selected as $\overline{\text{SCK}}$ by setting the serial mode register (SMH).

The data rate on receipt is up to 500 kbps when an internal clock is used as the $\overline{\text{SCK}}$; and up to 660 kbps when an external clock is used (when operated at 12 MHz).

(3) I/O interface mode

This mode, which is the same as μ COM-87's serial interface, is convenient when externally extending I/O devices or when connecting I/O controllers (such as A/D converters and liquid crystal controllers).

In the case of I/O interface mode, data transfer is performed from the most significant bit (MSB) with the character length fixed to 8 bits and without parity bits. Accordingly, please set the serial mode register (SML) to 0 CH, and the serial mode register (SMH)'s bit-5 (IOE) to "1".

This mode is synchronized by the controlled $\overline{\text{SCK}}$ (eight serial clocks). Please set the $\overline{\text{SCK}}$ to High except during data transfer.

The send operation is enabled by setting (to 1) bit-2 (TxE) of the serial mode register (SMH).

If data is written into the transmit buffer by the MOV TXB, A instruction, the data is automatically transferred to the serial register and sent from TxD at the falling edge of the controlled $\overline{\text{SCK}}$. When the transmit buffer is emptied, an internal interrupt (INTST) occurs.

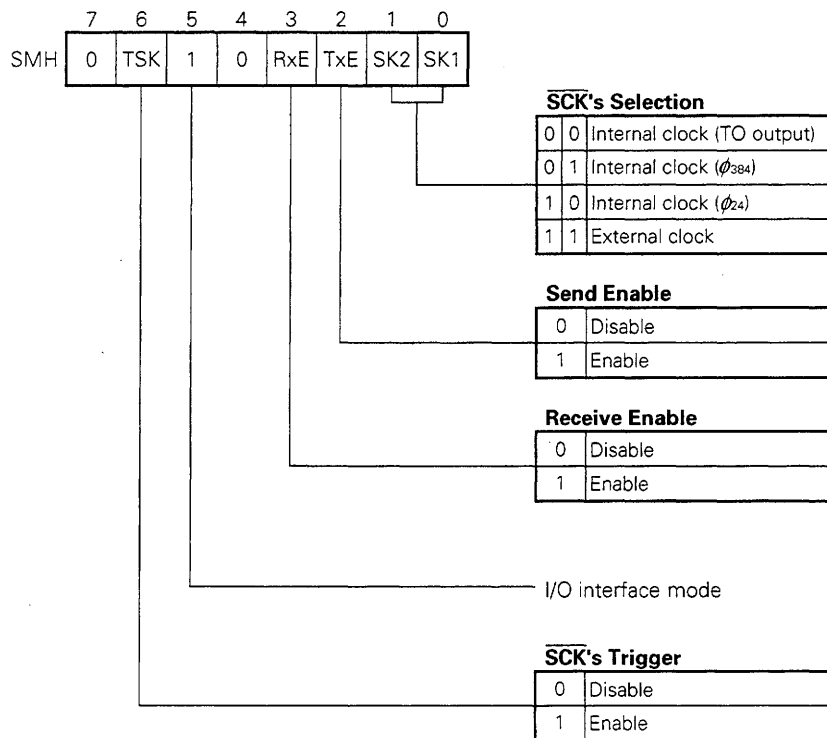
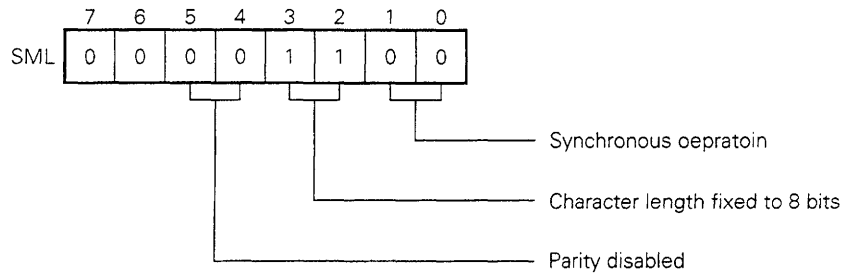
The data rate for transmission is up to 500 kbps when an internal clock is used as the $\overline{\text{SCK}}$; and up to 1 Mbps when an external clock is used (when operated at 12 MHz).

The receive operation is enabled by setting (to 1) bit-3 (RxE) of the serial mode register (SMH), and the received data enters the serial register at the rising edge of the controlled $\overline{\text{SCK}}$. When the serial register receives 8 bits of data, the data is transferred from the serial register to the receive buffer and an internal interrupt (INTSR) occurs.

The internal clock and the external clock are available as $\overline{\text{SCK}}$ s and can be selected by the serial mode register (SMH).

The data rate on receipt is up to 500 kbps when an internal clock is used as the $\overline{\text{SCK}}$; and up to 660 kbps when an external clock is used (when operated at 12 MHz). The high level width of the 8th $\overline{\text{SCK}}$ is required to be at least 6 states.

Figure 2-25. Real Mode Register Format in I/O Interface Mode



(4) Serial mode registers (SML, SMH)

These are the two 8-bit registers that control operation of the serial interface (see Figures 2-26 and 2-27).

Bits-0 and -1 (B1, B2) of the serial mode low register (SML) control switching between asynchronous mode and synchronous mode as well as the clock rate in asynchronous mode. Bits-2 and -3 (L1, L2) control the character length; bit-4 (PEN) controls parity enable; bit-5 (EP) controls odd/even parities, and bits-6 and -7 (S1, S2) control the number of stop bits.

At the time of $\overline{\text{RESET}}$ input or hardware STOP mode, the serial mode low register (SML) is set to 48H.

Bits-0 and -1 (SK1, SK2) of the serial mode high register (SMH) control whether to use an internal clock as the serial clock (SCK) or to use an external clock. Bit-2 (TxE) controls the send operation; bit-3 (RxE) controls the receive operation; and bit-4 (SE) controls whether to use search mode when in synchronous mode. Bit-5 (IOE) controls whether to use synchronous mode or I/O interface mode; bit-6 (TSK) uses an internal clock in I/O interface mode to start the serial clock when receiving data. The TSK bit is automatically reset (to 0) after starting the serial clock.

When the serial clock ($\overline{\text{SCK}}$) has been specified as an internal clock, the $\overline{\text{SCK}}$ value is determined based on the following formula:

If the internal clock is ϕ_{24} : $\overline{\text{SCK}} = \frac{f_{xx}}{24}$

If the internal clock is ϕ_{384} : $\overline{\text{SCK}} = \frac{f_{xx}}{384}$

If the internal clock is the TO output:

$\overline{\text{SCK}} = \frac{f_{xx}}{24 \times C}$ when the timer's input clock is ϕ_{12} ;

$\overline{\text{SCK}} = \frac{f_{xx}}{768 \times C}$ when the timer's input clock is ϕ_{384} ;

$\overline{\text{SCK}} = \frac{f_{xx}}{6}$ when the TIMER FF input is ϕ_3 .

In the formula above, f_{xx} refers to the oscillation frequency; $\overline{\text{SCK}}$ refers to the serial clock; and the C refers to the timer's count value. In the case that the internal clock is "TO output" and the TIMER F/F input is ϕ_3 , they can be used only when the clock rate is 16 or 64 in asynchronous mode.

At the time of $\overline{\text{RESET}}$ input or hardware STOP mode, the serial mode high register (SMH) is reset to 00H.

Figure 2-26. Serial Mode Low Register (SML)'s Format

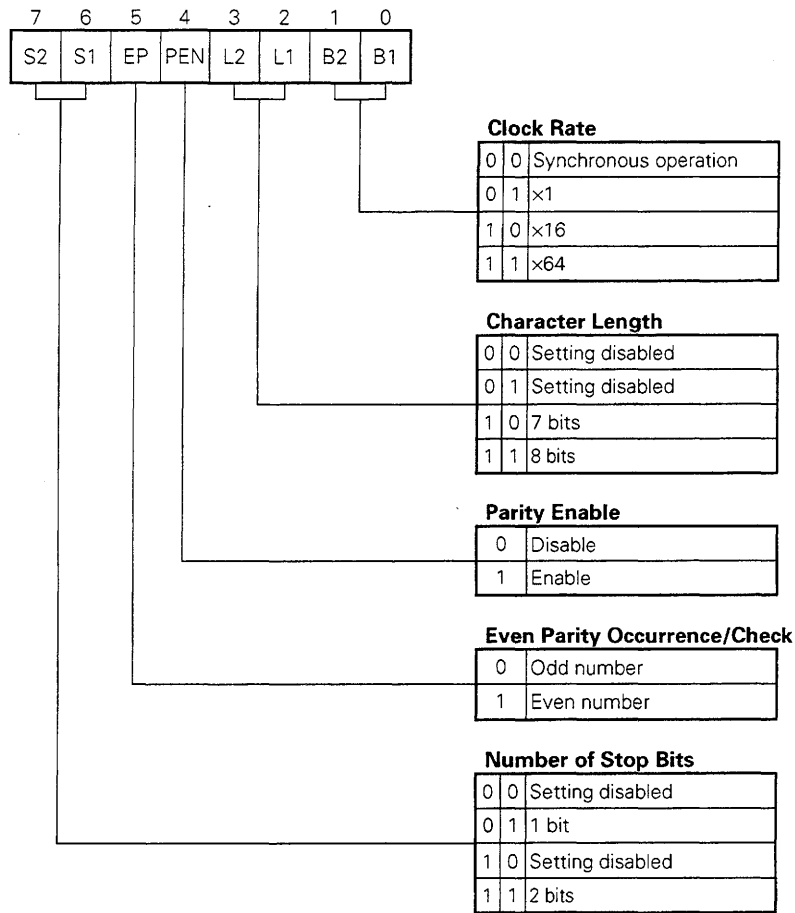
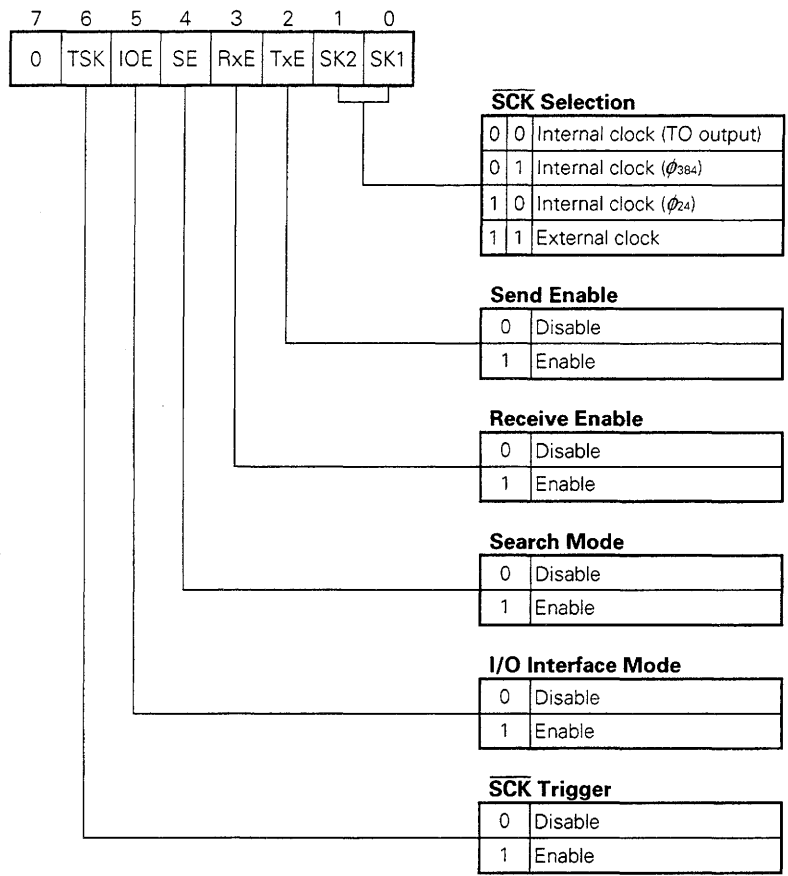


Figure 2-27. Format of the Serial Mode High Register (SMH)



2.9 ANALOG/DIGITAL CONVERTER

The μ PD78C14A includes an 8-bit high-speed high-resolution analog/digital (A/D) converter which has eight multiplex analog inputs (AN7-0) and is equipped with four Conversion Result registers (CR0-CR3) for retaining conversion results. The A/D converter uses the method of successive approximation.

A/D converter operation can be in scan mode or select mode, selected via software. In select mode, the conversion values from analog input are stored in CR0 to CR3 in that order. In scan mode, the analog input conversion values from AN0-AN3 or AN4-AN7 are stored in CR0 to CR3 in that order. Switching between these modes is done by specifying the A/D channel mode register.

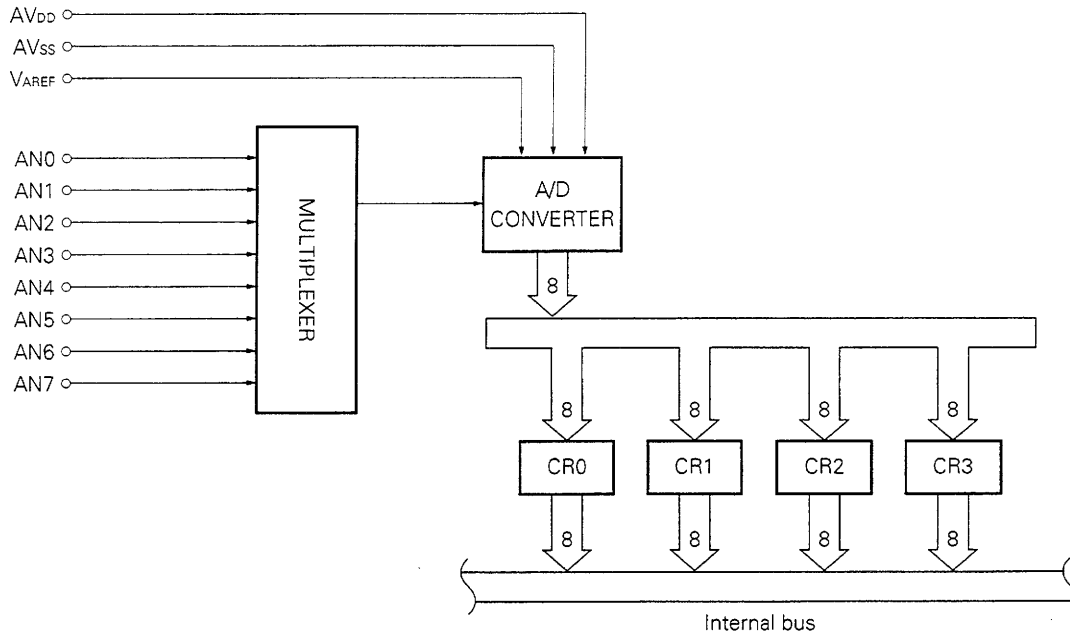
In the case of select mode, the A/D channel mode register is used to select a single analog input to start the A/D conversion. Conversion values are stored in CR0 to CR3 in that order; when conversion values fill the four CR registers, an internal interrupt (INTAD) occurs. The A/D converter continues the A/D conversion and stores the conversion values starting from CR0.

In the case of scan mode, the A/D channel mode register can be used to select the analog inputs (ANI2 = 0) AN0 to AN3, or analog inputs (ANI2 = 1) AN4 to AN7.

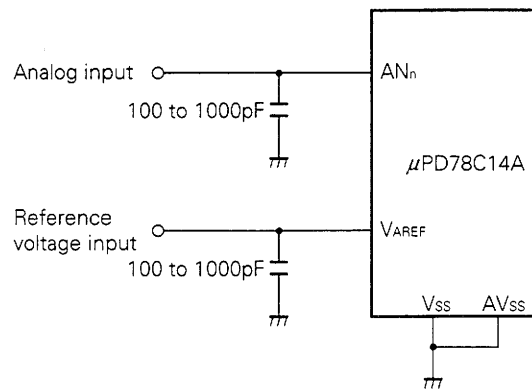
If the A/D channel mode register's bit-3 (ANI2) is set to 0, the analog inputs are selected in the order AN0, AN1, AN2, AN3, and then AN0; and the A/D conversion value of each input is stored in the order CR0, CR1, CR2, CR3, and then CR0. If the ANI2 of the A/D channel mode register is set to 1, the analog inputs are selected in the order AN4, AN5, AN6, AN7, and then AN4; and, the A/D conversion value of each input is stored in the order CR0, CR1, CR2, CR3, and then CR0. In scan mode, as in select mode, if the conversion values fill the four CR registers, an internal interrupt (INTAD) occurs.

In scan mode as well, the above operations are repeated cyclically until the A/D channel mode register is changed. The internal interrupt (INTAD) is disabled by setting (to 1) the interrupt mask register (MKH)'s bit 0 (MKAD).

Figure 2-28. A/D Converter Block Diagram



Caution To avoid malfunctions caused by noise, be sure to connect capacitors to the analog input pin and the reference voltage pin.



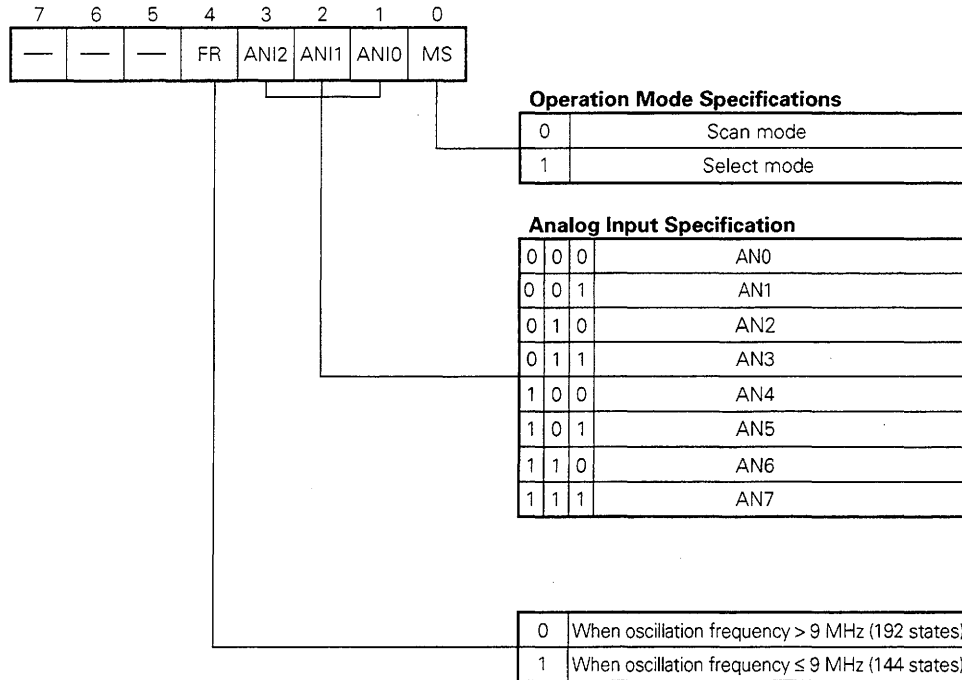
(1) A/D channel mode register (ANM)

Refers to the 8-bit register that controls operation of the A/D converter. The A/D channel mode register's bit-0 (MS) controls the operation mode. Bits-1, -2, and -3 (ANI0, 1, 2) control the input for A/D conversion; and bit-4 (FR) controls A/D operation relating to change in the oscillation frequency.

The A/D channel mode register can be written to specify the operation mode and also read. Therefore, it can tell which analog input the data belongs to when an A/D interrupt occurs.

At the time of RESET input or hardware STOP mode, the A/D channel mode register becomes 00H.

Figure 2-29. Format of the A/D Channel Mode Register



(2) Controlling operation of the A/D converter

The A/D converter can be stopped from doing conversion operations by controlling the V_{AREF} input voltage. If a voltage exceeding V_{IH1} is input to the V_{AREF} pin, the A/D converter starts conversion operations, and the conversion results are assured when V_{AREF} = 3.4 V to AV_{DD}. If the input voltage of the V_{AREF} pin is lowered below V_{IL1} during conversion operations, the conversion operations by the A/D converter are stopped. The contents of CR0 to CR3 at this time are undefined.

When the V_{AREF} input voltage is varied for the purpose of controlling the stoppage of the A/D converter, the A/D channel mode register (ANM) is not affected. Therefore, if the V_{AREF} input voltage is raised above 3.4 volts to return to the operating state that was in effect before stopping, the A/D converter resumes operation, storing the first conversion value into CR0 in the mode that was in effect immediately before entering the stopped state.

When the V_{AREF} input voltage level is changed, the edge detection function of the AN4-AN7 inputs is not affected.

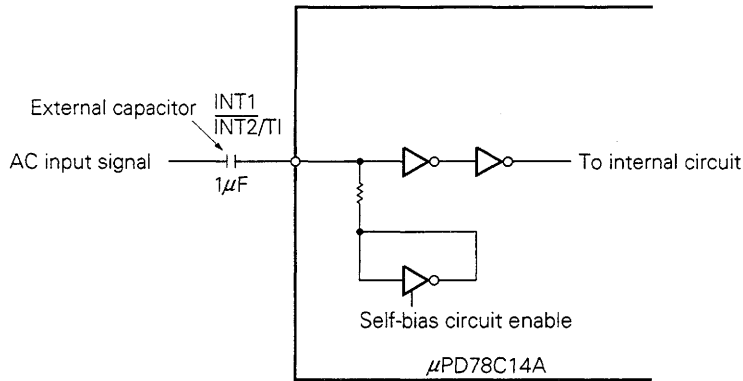
Caution The inputs AN0-AN7 when V_{AREF} is set to the low level must be set in the range of AV_{SS} to AV_{DD}.

2.10 ZERO-CROSS DETECTION CIRCUIT

The INT1 pin and the $\overline{\text{INT2/TI}}$ (combined with PC3) pin can be made to do zero-cross detection by setting the zero-cross mode register.

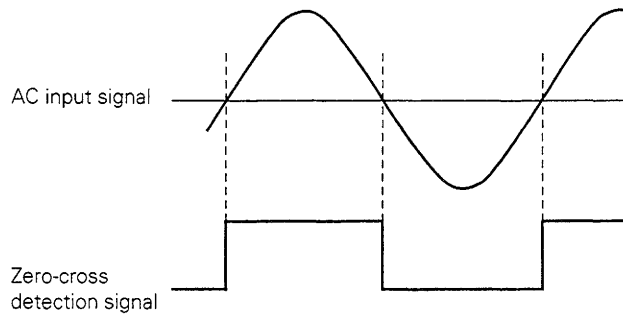
The zero-cross detection circuit consists of self-bias-based high-gain amplifiers. By biasing its input to the switching point, a digital displacement is generated in response to a slight amount of displacement in the input.

Figure 2-30. Zero-Cross Detection Circuit



The zero-cross detection circuit detects a movement from negative to the positive or vice versa of an AC signal that is input through an external capacitor, and generates a digital pulse that changes from 0 to 1 or vice versa at each crossing point.

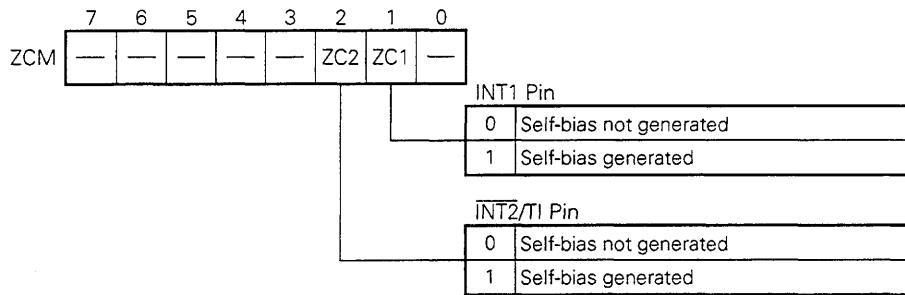
Figure 2-31. Zero-Cross Detection Signal



The digital pulse which is generated from the zero-cross detection circuit of the INT1 pin is sent to the interrupt control circuit to set the INTF1 interrupt request flag at the negative-to-positive zero-cross point (rising edge) of the AC signal and start interrupt operation if the INT1 interrupt is enabled. The digital pulse generated at the zero-cross detection circuit of the $\overline{\text{INT2/TI}}$ pin can also be sent to the interrupt control circuit to start the interrupt operation in the same manner as the INT1 pin at the positive-to-negative zero-cross point (falling edge) of the AC signal. Furthermore, the digital pulses generated by the zero-cross detection circuit of the $\overline{\text{INT2/TI}}$ pin can be used even as the input clock of the timer.

The format of the zero-cross mode register (ZCM) which controls the self-bias for the zero-cross detection of the INT1 pin and the $\overline{\text{INT2/TI}}$ pin is shown in Figure 2-32.

Figure 2-32. Format of the Zero-Cross Mode Register



When the ZC1 and ZC2 bits of the zero-cross mode register are set to 0, each pin responds as a normal digital input without generating the self-bias for detecting the zero-cross of each pin.

When the ZC1 or ZC2 bit is set to 1, the self-bias is generated thus making it possible to detect the zero-cross of the AC input signal by connecting a capacitor to the pin. A pin whose ZC1 or ZC2 bit is set to 1 can be driven without going through an external capacitor, in which case, the pin responds as a digital input. However, in this case, an input load current is required, thus making it necessary to consider the output driver of the external circuit. Therefore, when using a pin simply as an interrupt input, a timer input, or a port input/output, without performing zero-cross detection, please set the ZC1 or ZC2 bit of the zero-cross mode register to 0.

Both the ZC1 and ZC2 bits are set to 1 by the $\overline{\text{RESET}}$ input, thus generating the self-bias.

The zero-cross detection function of the $\overline{\text{INT2/TI}}$ (combined with PC3) pin can be used only when specified in the control mode in the MODE CONTROL C register (MCC). In port mode, the zero-cross detection function does not operate.

Caution In the zero-cross detection circuit, which differs from other CMOS circuits in terms of its operation point, the supply current constantly flows. This is the same in standby modes (HALT, software/hardware STOP mode) as well. Accordingly, a slightly larger amount of current flows when the zero-cross detection circuit is put into operation (self-bias generated; ZCX = 1) than when not, and its effect is large in software/hardware STOP mode.

3. INTERRUPT FUNCTION

Interrupts include three types of external interrupt requests and eight types of internal interrupt requests. These 11 types of interrupt requests are classified into six groups, with six priority levels and six interrupt addresses.

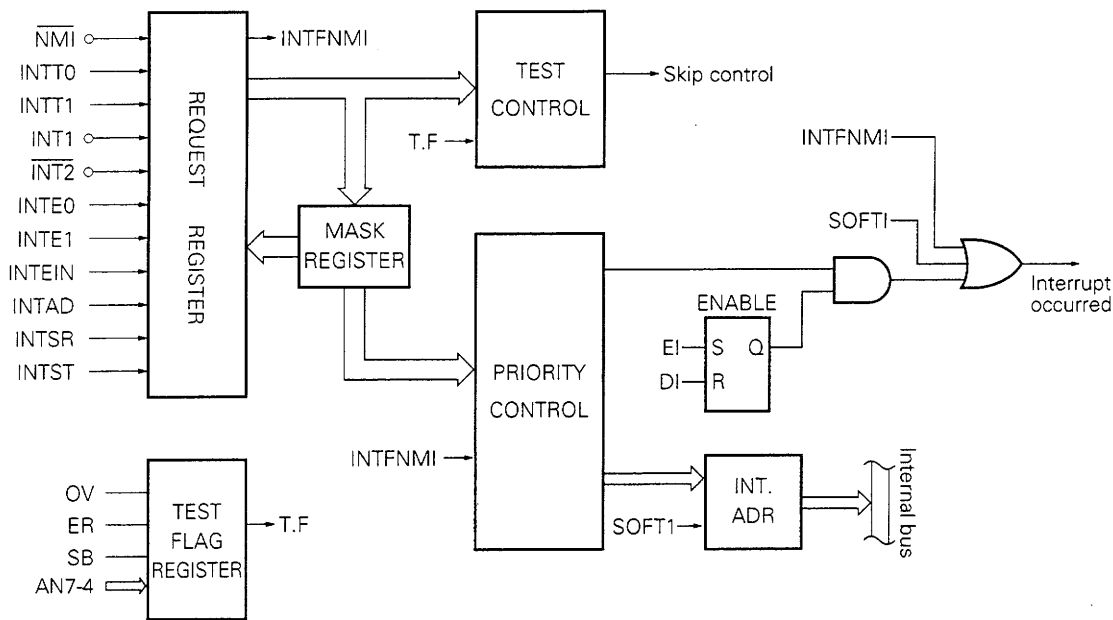
The priority level and interrupt address of each interrupt source are shown below.

Priority Level	Interrupt Address	Interrupt Request	Internal/ External
1	4	$\overline{\text{NMI}}$ (Falling edge)	External
2	8	INTT0 (Coincidence signal from TIMER0)	Internal
		INTT0 (Coincidence signal from TIMER1)	
3	16	INT1 (Rising edge)	External
		$\overline{\text{INT2}}$ (Falling edge)	
4	24	INTE0 (Coincidence signal from timer/event counter)	Internal
		INTE1 (Coincidence signal from timer/event counter)	
5	32	INTEIN (CI and TO's falling edge)	Internal
		INTAD (A/D converter interrupt)	
6	40	INTSR (Serial receiving end interrupt)	Internal
		INTST (Serial sending end interrupt)	

3.1 INTERRUPT CONTROL CIRCUIT

The interrupt control circuit consists of the REQUEST REGISTER, MASK REGISTER, PRIORITY CONTROL, TEST CONTROL, INTERRUPT ENABLE F/F, and TEST FLAG REGISTER (see Figure 3-1).

Figure 3-1. Interrupt Control Circuit Block Diagram



(a) REQUEST REGISTER

Consists of the interrupt request flags that are set by various interrupt requests. Each flag is capable of accepting an interrupt request, or is reset by execution of a skip instruction (SKIT, SKNIT). All the flags are reset by inputting the $\overline{\text{RESET}}$.

There are 11 types of interrupt request flags as follows:

- **INTFNMI**
This flag is set (to 1) by the falling edge input to the $\overline{\text{NMI}}$ pin. It differs from other interrupt request flags and cannot be tested by the skip instruction.
- **INTFT0**
This flag is set (to 1) by the coincidence signal from TIMER0's COMPARATOR.
- **INTFT1**
This flag is set (to 1) by the coincidence signal from TIMER1's COMPARATOR.
- **INTF1**
This flag is set by a rising edge input to the INT1 pin.
- **INTF2**
This flag is set by a falling edge input to the $\overline{\text{INT2}}$ pin.
- **INTFE0**
This flag is set (to 1) by the coincidence signal generated when the contents of the timer/event counter's ECNT and ETM0 registers match.
- **INTFE1**
This flag is set (to 1) by the coincidence signal generated when the contents of the timer/event counter's ECNT and ETM1 registers match.
- **INTFEIN**
This flag is set by the falling edge of the timer/event counter's CI input or TO (timer output).
- **INTFAD**
This flag is set by the conversion values of the A/D converter having been transferred to the four registers, CR0 to CR3.
- **INTFSR**
This flag is set (to 1) if the receive buffer of the serial interface is full.

- INTFST

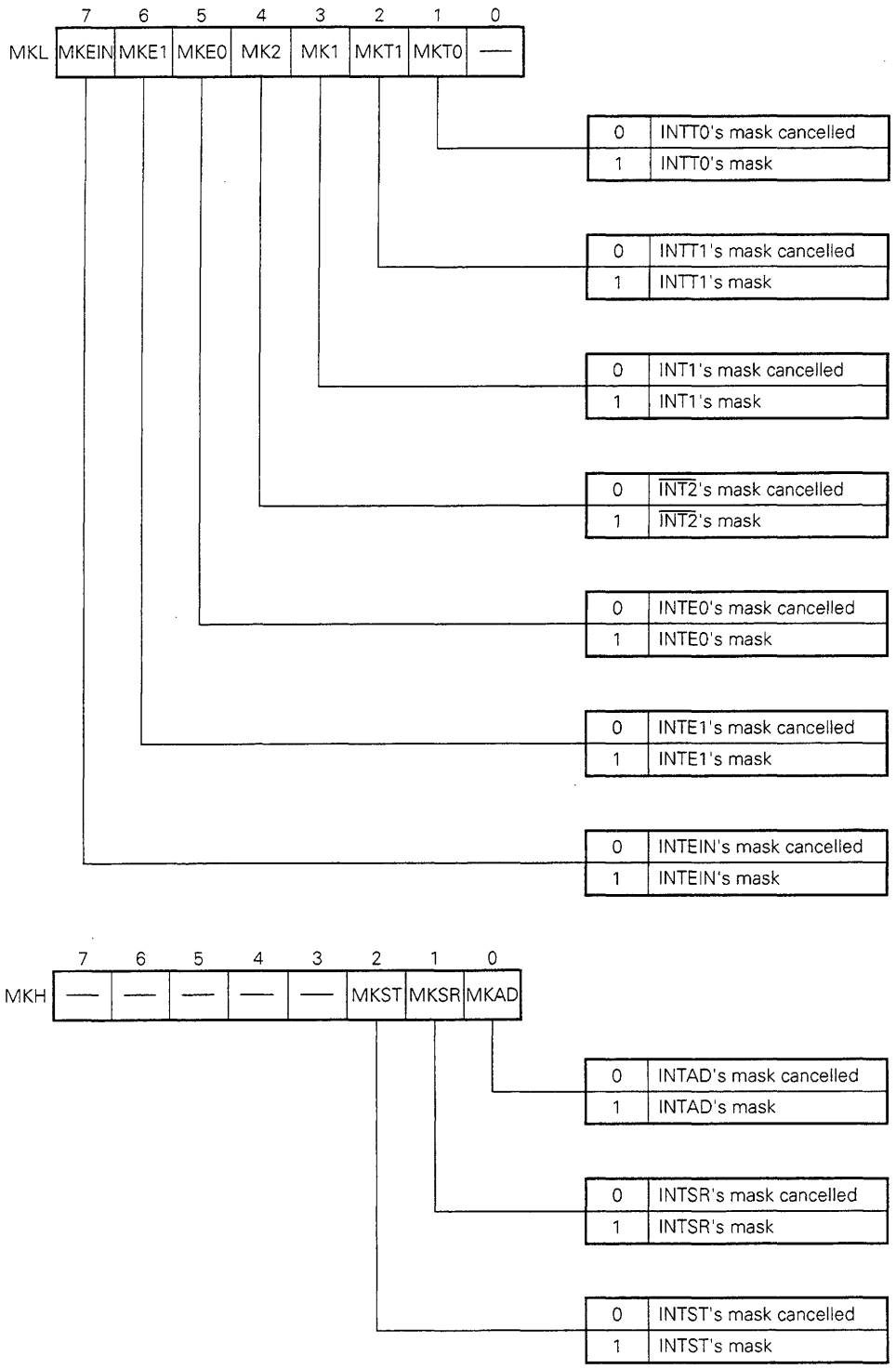
This flag is set (to 1) if the send buffer of the serial interface is emptied.

(b) **MASK REGISTER**

This register has 10 bits corresponding to the interrupt requests other than the non-maskable interrupt ($\overline{\text{NMI}}$), and is set (to 1) or reset (to 0) in bit units by instructions. Each interrupt request is masked when the corresponding bit of the mask register is 1; and enabled when it is 0.

All the bits of the mask register are set by the $\overline{\text{RESET}}$ input, thus masking all interrupt requests other than non-maskable interrupts. In addition, all the bits of the mask register are set in hardware STOP mode.

Figure 3-2. Mask Register (MKL, MKH) Format



(c) PRIORITY CONTROL circuit

This circuit controls the six priority levels described previously. If multiple interrupt request flags are set simultaneously, the one with the highest priority level is accepted.

(d) TEST CONTROL circuit

This circuit operates when executing skip instructions (SKIT, SKNIT) to test an interrupt request flag for any interrupt source (except INTFNMI), to test the $\overline{\text{NMI}}$ pin state, or to test the test flag.

(e) INTERRUPT ENABLE F/F (IE F/F)

This is a flip-flop that is set by the EI instruction and reset by the DI instruction. It is reset once any interrupt is accepted. It is reset also by the $\overline{\text{RESET}}$ input. Setting this flip-flop results in interrupt enable; resetting it results in interrupt disable.

(f) TEST FLAG REGISTER

Consisting of seven types of test flags which do not generate interrupt requests, this register can be tested and reset by skip instructions (SKIT, SKNIT).

•OV

Set (to 1) by overflow of the timer/event counter's ECNT.

•ER

Set (to 1) by a parity error, framing error, or overrun error in the serial interface.

•SB

Set (to 1) when the V_{DD} pin is raised from a level lower than a specified level to a level higher than a specified level.

•AN7-AN4

Set (to 1) by a falling edge to AN7 to AN4 pins.

3.2 OPERATION OF NON-MASKABLE INTERRUPTS

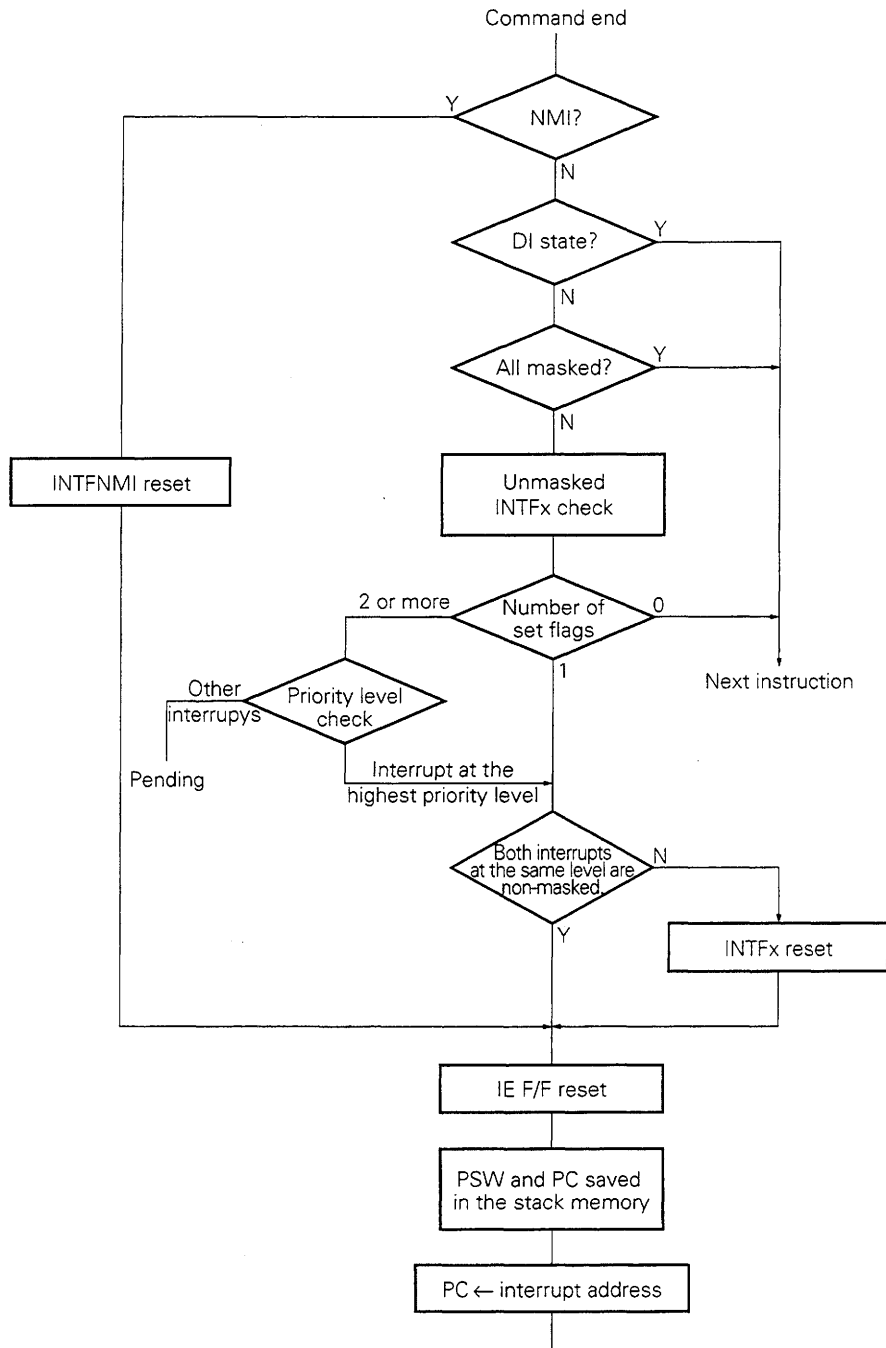
A non-maskable interrupt is accepted in accordance with the following processing sequence regardless of the EI/DI state, if the interrupt request flag (INTFNMI) is set by falling edge input to the $\overline{\text{NMI}}$ pin (see Figure 3-3).

- (i) Whether the INTFNMI is set or not is checked at the end of each instruction and, if set, the non-maskable interrupt is accepted and INTFNMI is reset.
- (ii) If the non-maskable interrupt is accepted, the IE F/F is reset thus disabling (DI status) all the interrupts other than the non-maskable interrupt and the SOFTI instruction.
- (iii) PSW, PC's high byte, and PC's low byte are saved in that order in the stack memory.
- (iv) Jump to the interrupt address (0004H).

The above interrupt operations are automatically done in 16 states.

The interrupt request flag (INTFNMI) cannot be tested by the skip instruction; however, skip instructions (SKIT NMI, SKNIT NMI) can be used to test the state of the $\overline{\text{NMI}}$ pin. Accordingly, by testing the state of the $\overline{\text{NMI}}$ pin several times with the skip instruction in the non-maskable interrupt service routine, it is possible to remove noise with a comparatively long period or periodic noise. The state of the $\overline{\text{NMI}}$ pin does not change even when tested by the skip instruction.

Figure 3-3. Interrupt Operation Procedure



3.3 MASKABLE INTERRUPT OPERATION

Interrupt requests other than the non-maskable interrupt and those caused by the SOFTI instruction are maskable interrupts can be enabled/disabled (IE F/F's set/reset) by the EI/DI instructions. They are also maskable interrupts which can be masked individually by the mask register.

If an external interrupt that is a maskable interrupt is accepted as being a normal interrupt signal because of an active level input for more than a specified period of time, the interrupt request flag is set. For internal interrupts, if an interrupt request occurs, an interrupt request flag is set immediately. Once an interrupt request flag is set, both external and internal interrupts are processed according to the following procedure (see Figure 3-3).

- (i) If in the EI state (IE F/F = 1), whether the interrupt request flag is set or not, the flag is checked at the end of each instruction, and, if set, the system is placed in the interrupt request cycle. Checking of interrupt requests that are masked by the mask register is not performed.
- (ii) If multiple interrupt request flags are set simultaneously, their priority levels are checked and the one with the highest priority level is accepted with the others held pending.
- (iii) When an interrupt request has been accepted, its interrupt request flag is automatically reset. When two interrupt requests at the same priority level are unmasked by the mask register, the interrupt request flags are not reset. This is so that the software can distinguish between the two interrupts later.
- (iv) If an interrupt request is accepted, the IE F/F is reset, thus disabling (DI status) all interrupts other than the non-maskable interrupt and the SOFTI instruction.
- (v) The PSW, PC's high byte, and PC's low byte are saved in that order in the stack memory.
- (vi) Jump to the interrupt address.

The above interrupt operation is automatically made in 16 states.

An interrupt request held pending is accepted when placed in the interrupt enable state by execution of the EI instruction, if there is no other interrupt request at a higher priority level.

Maskable interrupts have two types of interrupt requests at the same priority level and at the same interrupt address. The mask register can be set so that (1) both masks are cancelled; (2) only one mask is canceled; or (3) both masks are selected.

(1) When both masks are cancelled:

Both of the relevant bits of the mask registers corresponding to the two types of interrupt requests are set to 0. In this case, the logical sum of the two types of interrupt request flags becomes the interrupt request.

When an interrupt request is accepted by setting one or both of the interrupt request flags at the same priority level and there is a jump to the interrupt address in accordance with interrupt operation, the interrupt request flag is not reset. Accordingly, the cause of the interrupt can be determined, and the interrupt request flag is reset by executing a skip instructions that tests the interrupt request flag at the beginning of the interrupt service routine.

(2) When one mask is cancelled:

Of the two types of interrupt requests at the same priority level, the relevant bit of the mask register corresponding to the interrupt request whose mask should be cancelled is set to 0 and the other bit is set to 1. In this case, the interrupt request is generated by setting the interrupt request flag that is not masked; if the interrupt is accepted in accordance with interrupt operation, the interrupt request flag is reset automatically.

When the masked interrupt request flag has been set, this interrupt request is held pending. When the mask of the pending interrupt request is cancelled, the request is accepted in the interrupt enable state unless there are interrupts at higher priority levels.

(3) When both are masked:

The relevant bits of the mask register corresponding to the two types of interrupt requests are both set to 1. In this case, even when an interrupt request flag is set, it is not accepted but held pending. When the mask of the interrupt request that is pending is cancelled, the request is accepted in the interrupt enable state unless there are interrupts at higher priority levels.

3.4 INTERRUPT OPERATION BY SOFTI INSTRUCTION

If the SOFTI instruction is executed, it jumps to the interrupt address (0060H) unconditionally. The SOFTI instruction interrupt is not affected by the IF F/F. And, even if this instruction is executed, the IE F/F is not affected.

An interrupt by the SOFTI instruction is processed according to the following procedure.

- (i) The PSW, PC's high byte, and PC's low byte are saved in that order in the stack memory.
- (ii) Jump to the interrupt address (0060H).

Caution Even when the skip condition is generated by the instruction (various instructions for arithmetic operation, logical operation, increment/decrement, shift, skip, and RETS) immediately preceding the SOFTI instruction, the SOFTI instruction is executed without being skipped. By execution of the SOFTI instruction, the PSW's SK flag is saved in the stack area while remaining set (to 1). Accordingly, on return from the SOFTI processing routine, the PSW's SK flag remains set and the instruction following the SOFTI instruction is skipped.

Please pay attention to the fact that the μ PD78C14A's SOFTI instruction differs from μ COM-87 in that the address to be saved on the stack address in the case of SOFTI is the start address of the next instruction.

4. STANDBY FUNCTION

The μPD78C14A makes available three types of standby modes (HALT mode, software STOP mode, hardware STOP mode) to save power consumption while programs are waiting.

4.1 HALT MODE

If the HLT instruction is executed, the system is placed in HALT mode unless an interrupt request flag for an interrupt whose mask has been cancelled is set. In HALT mode, the CPU clock is halted together with program execution; however, the contents of all the registers and built-in RAM are retained. Operation by the timer, the timer/event counter, the serial interface, the A/D converter, and the interrupt control circuit, etc., are still possible even during HALT mode.

The states of the μPD78C14A's output pins during HALT mode are listed in Table 4-1 below.

Table 4-1. Output Pin State

Output Pin	Single Chip	External Extension
PA7-0	Data retained	Data retained
PB7-0	Data retained	Data retained
PC7-0	Data retained	Data retained
PD7-0	Data retained	High impedance
PF7-0	Data retained	Next address retained ^{Note 1} ; data retained ^{Note 2}
WR, RD	High level	High level
ALE	High level	High level

- Notes 1. Address output pin
- 2. Port data output pin

Caution Due to the fact that interrupt request flags are used for cancellation of HALT mode, the system cannot be placed in HALT mode even when the HALT instruction is executed if the situation is such that even a single interrupt request flag for an unmasked interrupt has been set. Accordingly, before attempting to set HALT mode in a location where an interrupt request flag may have been set (or a pending interrupt may be present), please process the pending interrupt, reset the interrupt request flag by executing the skip instruction, or mask all interrupts other than the interrupt used for cancelling the HALT mode.

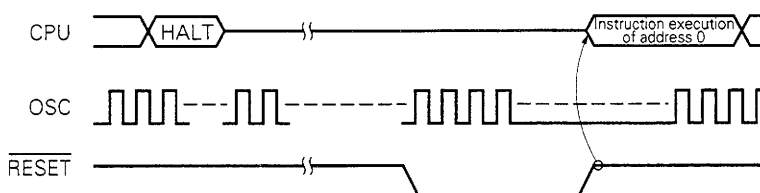
4.2 CANCELLATION OF HALT MODE

(1) Cancellation by the RESET signal

If the RESET signal goes from High to Low during HALT mode, HALT mode is cancelled, thus placing the system in the reset state. If the RESET signal returns to High, the CPU starts execution of the program from address 0.

The contents of RAM are retained even when the RESET signal is input. However, the contents of the registers become undefined.

Figure 4-1. Cancellation Timing of HALT Mode (RESET Signal Input)



(2) Cancellation by the interrupt request flag

HALT mode is cancelled if, during HALT mode, one or more interrupt request flags is set by occurrence of the non-maskable interrupt ($\overline{\text{NMI}}$) or the ten types of maskable interrupts (INTT0, INTT1, INT1, INT2, INTE0, INTE1, INTEIN, INTAD, INTST, and INTSR) whose masks are cancelled.

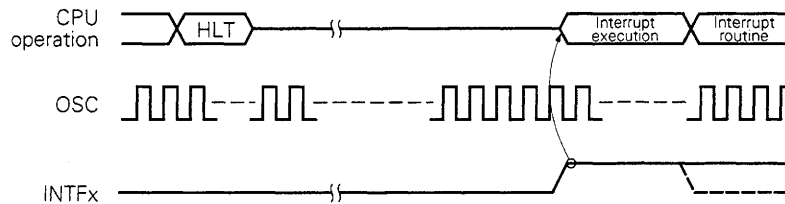
When HALT mode has been cancelled by the non-maskable interrupt, the system jumps to the interrupt address (0004H) regardless of the interrupt enable/disable (EI/DI) state and without executing the instruction following the HLT instruction.

When the HALT mode has been cancelled with a maskable interrupt, operation after the cancellation varies depending on the EI/DI states.

(i) EI state

The system jumps to the relevant interrupt address without executing the instruction following the HLT instruction.

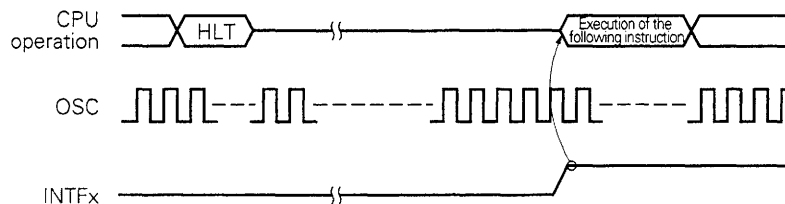
Figure 4-2. Cancellation Timing of HALT Mode (when in the EI state)



(ii) DI state

Execution is restarted from the instruction following the HLT instruction (without jumping to the interrupt address). The interrupt request flag used for the cancellation remains set. Please reset it with the skip instruction if necessary.

Figure 4-3. Cancellation Timing of HALT Mode (when in the DI state)



4.3 SOFTWARE STOP MODE

If the STOP instruction is executed, the system is placed in the software STOP mode unless the interrupt request flag of an unmasked external interrupt is set. In software STOP mode, all the clocks are stopped. If the system is placed in software STOP mode, execution of the program is stopped and the contents of the registers and built-in RAM are retained (timer's UPCOUNTER cleared to 00H). Otherwise, except only for the fact that the $\overline{\text{NMI}}$ and $\overline{\text{RESET}}$ signals that are used for cancelling software STOP mode are valid, all the other functions are stopped.

The state of the μPD78C14A's output pins during software STOP mode is as shown in Table 4-1, in the same manner as HALT mode.

- Cautions**
1. To prevent malfunctions caused by internal interrupts that may occur during the oscillation operation stabilization time when cancelling the software STOP mode, please be sure to mask the internal interrupts before executing the STOP instruction.
 2. When software STOP mode has been cancelled by setting the interrupt request flag of the non-maskable interrupt, TIMER1's coincidence signal is used as the signal for starting CPU operation to provide the stabilization time for oscillation operation. Therefore, it is required to set a count value in TIMER REG, which sets the stabilization time for oscillation operation, before executing the STOP instruction; and set the timer mode register to the timer operation state.

4.4 CANCELLATION OF SOFTWARE STOP MODE

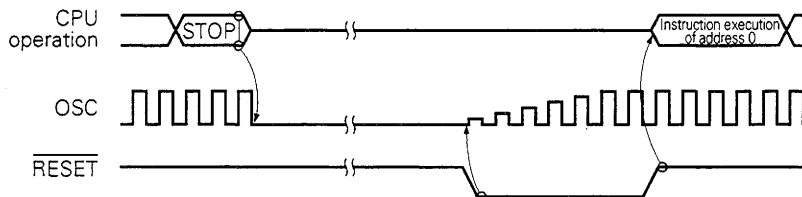
(1) Cancellation by the $\overline{\text{RESET}}$ signal

If the $\overline{\text{RESET}}$ signal goes from High to Low during software STOP mode, software STOP mode is cancelled, resetting the system and starting clock oscillation. If the $\overline{\text{RESET}}$ signal goes High after stabilizing oscillation operation, the CPU starts to execute the program from address 0.

If the $\overline{\text{RESET}}$ signal goes from High to Low, clock oscillation is started; however, it takes some time until oscillation operation is stabilized. Therefore, it is necessary for the low level width of the $\overline{\text{RESET}}$ signal to be greater than the stabilization time for oscillation operation.

The contents of RAM are retained even by the $\overline{\text{RESET}}$ signal input; however, in this case, the contents of the registers become undefined.

Figure 4-4. Cancellation Timing of Software STOP Mode ($\overline{\text{RESET}}$ Signal Input)



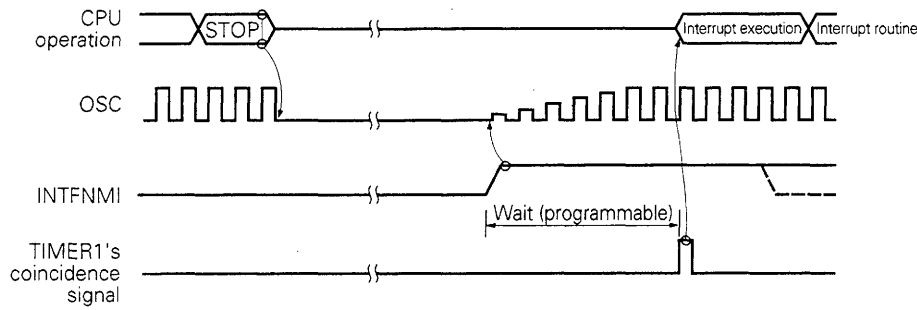
When software STOP mode has been cancelled by the $\overline{\text{RESET}}$ signal, program execution is started from address 0 in the same manner as in normal power-ON resetting. Therefore, to make a distinction between them, the SB (standby) flag can be used. The SB flag is set (to 1) when the V_{DD} pin is increased from a low level below a specified level to a high level above a specified level; and reset by execution of the skip instruction. Therefore, as a result of testing the SB flag with the skip instruction in the program executed after inputting the $\overline{\text{RESET}}$ signal, the program is started by powering-ON if the SB flag is set; and, if the SB flag is reset, the program is started by cancelling the software STOP mode.

(2) Cancellation by the interrupt request flag

If the non-maskable interrupt request flag is set during software STOP mode, software STOP mode is cancelled and at the same time clock oscillation is started. If clock oscillation is started, the timer's UPCOUNTER starts counting up from 00H in accordance with the setting made before executing the STOP instruction. The CPU starts operation by means of the coincidence signal (wait time to allow for stabilization of oscillation operation) from TIMER1's UPCOUNTER. In this case, the UPCOUNTER's coincidence signal does not set the interrupt request flag; and the timer's timer mode register after the coincidence occurrence is set to FFH, thus stopping operation of the timer.

After the stabilization time for oscillation operation has elapsed, the system jumps to the interrupt address (0004H) regardless of the interrupt enable/disable (EI/DI) states and without executing the instruction following the STOP instruction.

Figure 4-5. Cancellation Timing of Software STOP Mode



4.5 HARDWARE STOP MODE

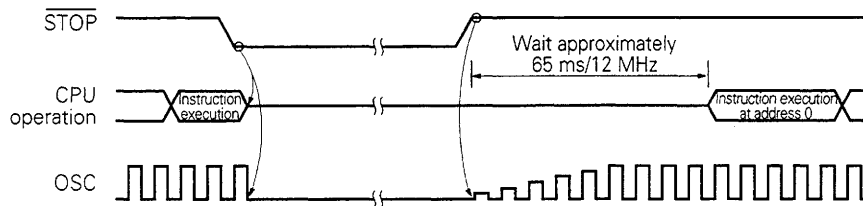
If the $\overline{\text{STOP}}$ signal goes from High to Low, the system is placed in hardware STOP mode. In hardware STOP mode, all the clocks are stopped. In hardware STOP mode, program execution is stopped and the contents of the RAM are retained. Otherwise, except for the fact that only the $\overline{\text{STOP}}$ signal that is used for cancelling the hardware STOP mode is valid, all other functions are stopped, placing the system in the reset state.

All the μPD78C14A's output pins during hardware STOP mode go to high impedance.

4.6 CANCELLATION OF HARDWARE STOP MODE

If the $\overline{\text{STOP}}$ signal goes from Low to High during hardware STOP mode, hardware STOP mode is cancelled and at the same time clock oscillation is started. After this, if the wait time (about 65 ms / 12 MHz) for stabilization of oscillation operation has elapsed, the CPU starts execution of the program from address 0 (see Figure 4-6).

Figure 4-6. Cancellation of Hardware STOP Mode



Hardware STOP mode is not cancelled even when the $\overline{\text{RESET}}$ signal goes from High to Low. If the $\overline{\text{STOP}}$ signal goes from Low to High when the $\overline{\text{RESET}}$ signal is Low, hardware STOP mode is cancelled, thus starting clock oscillation. After this, if the $\overline{\text{RESET}}$ signal is returned from Low to High, the CPU starts program execution from address 0 without taking time for stabilization of oscillation operation (see Figure 4-7). Even when the $\overline{\text{RESET}}$ signal goes from High to Low immediately after hardware STOP mode has been cancelled (the $\overline{\text{STOP}}$ signal going from Low to High), program execution continues just as when the $\overline{\text{RESET}}$ signal returns from Low to High (see Figure 4-8).

Accordingly, be sure to take the stabilization time for oscillation operation into consideration when changing the $\overline{\text{RESET}}$ signal to High.

The contents of RAM are retained even when the $\overline{\text{RESET}}$ signal is input. However, the contents of the registers become undefined.

Figure 4-7. Cancellation Timing of Hardware STOP Mode

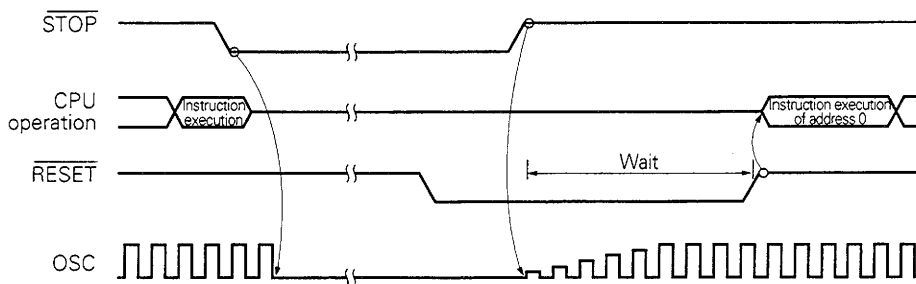
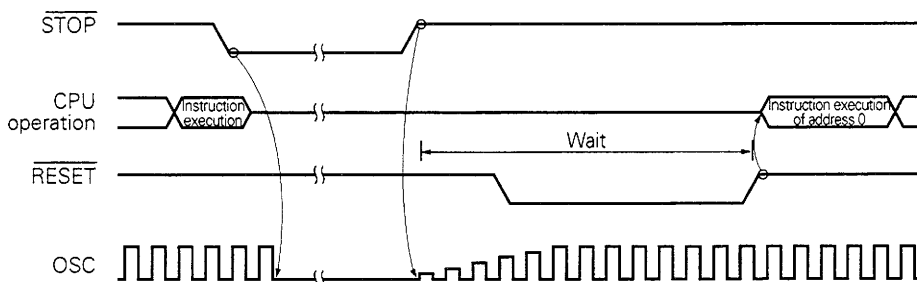


Figure 4-8. Cancellation Timing of Hardware STOP Mode



Cancelling hardware STOP mode is the same as cancelling software STOP mode with the $\overline{\text{RESET}}$ signal in that whether the start is by powering-ON or by cancellation of the hardware STOP mode can be known by testing the SB flag with the skip instruction.

4.7 LOW-SUPPLY-VOLTAGE DATA RETENTION MODE

By simply lowering the V_{DD} supply voltage to 2.5 V after setting the system to software/hardware STOP mode, the system can be set to low-supply-voltage data retention mode; and this software/hardware STOP mode enables the contents of the RAM to be retained with even less power consumption.

Caution Software/hardware STOP mode cannot be cancelled while the system is in low-supply-voltage data retention mode. Be sure to raise V_{DD} to the normal operating voltage.

5. RESET OPERATIONS

When $\overline{\text{RESET}}$ Input becomes low, the system reset is activated to create the following status.

- INTERRUPT ENABLE F/F is reset and interrupt is disabled.
- All the interrupt mask registers are set (1) and interrupt is masked.
- An interrupt request flag is reset (0) and hold interrupt is eliminated.
- Each bit of PSW is reset (0).
- 0000H is loaded into the program counter (PC).
- The MODE A, MODE B, MODE C, and MODE F registers are set to FFH and the bits (MM0, 1, and 2) of the MODE CONTROL C and MEMORY MAPPING registers are respectively reset (0), then all the ports (A, B, C, D, and F) become input port (output high-impedance).
- All the test flags but SB flag are reset (0).
- A timer mode register is set to FFH, and TIMER F/F is reset.
- The mode register (ETMM, EOM) of a timer/event counter is reset (0).
- The serial mode high register (SMH) of serial interface is reset (0), while the serial mode low register (SML) is set to 48H.
- The A/D channel mode register of the A/D converter is reset (0).
- $\overline{\text{WR}}$, $\overline{\text{RD}}$, ALE signals become high-impedance.
- The ZC1, ZC2 bits of the zero-cross mode register (ZCM) are set (1).
- Data memory and the following register contents are undefined.
- The internal timing generator is initialized.

Stack pointer (SP)
 Expansion accumulator (EA, EA'), accumulator (A, A')
 General register (B, C, D, E, H, L, B', C', D', E', H', L')
 Output latch of each port
 TIMER REG0, 1 (TM0, TM1)
 TIMER/EVENT COUNTER REG0, 1 (ETM0, ETM1)
 RAE bit of MEMORY MAPPING register
 SB flag of test flag

When $\overline{\text{RESET}}$ input becomes high, the reset status is released. Then, execution of the program is started from 0000H. The contents of various kinds of registers must be initialized or re-initialized in the program, if necessary.

6. INSTRUCTION SET

6.1 IDENTIFIER/DESCRIPTION OF OPERAND

Identifier	Description
r r1 r2	V, A, B, C, D, E, H, L EAH, EAL, B, C, D, E, H, L A, B, C
sr sr1 sr2 sr3 sr4	PA, PB, PC, PD, PF, MKH, MKL, ANM, SMH, SML, EOM, ETMM, TMM, MM, MCC, MA, MB, MC, MF, TXB, TM0, TM1, ZCM PA, PB, PC, PD, PF, MKH, MKL, ANM, SMH, EOM, TMM, RXB, CR0, CR1, CR2, CR3 PA, PB, PC, PD, PF, MKH, MKL, ANM, SMH, EOM, TMM ETM0, ETM1 ECNT, ECPT
rp rp1 rp2 rp3	SP, B, D, H V, B, D, H, EA SP, B, D, H, EA B, D, H
rpa rpa1 rpa2 rpa3	B, D, H, D+, H+, D-, H- B, D, H B, D, H, D+, H+, D-, H-, D+byte, H+A, H+B, H+EA, H+byte D, H, D++, H++, D+byte, H+A, H+B, H+EA, H+byte
wa	8 bit immediate data
word byte bit	16 bit immediate data 8 bit immediate data 3 bit immediate data
f	CY, HC, Z
irf	NMI ^{Note} , FT0, FT1, F1, F2, FE0, FE1, FEIN, FAD, FSR, FST, ER, OV, AN4, AN5, AN6, AN7, SB

Note NMI can also be described as FNMI.

Remarks

1. sr to sr4 (special register)

PA : PORT A	ETMM : TIMER/EVENT
PB : PORT B	COUNTER MODE
PC : PORT C	EOM : TIMER/EVENT
PD : PORT D	COUNTER OUTPUT
PF : PORT F	MODE
MA : MODE A	ANM : A/D CHANNEL MODE
MB : MODE B	CR0 : A/D CONVERSION
MC : MODE C	to RESULT 0 to 3
MCC : MODE CONTROL C	CR3
MF : MODE F	TXB : Tx BUFFER
MM : MEMORY MAPPING	RXB : Rx BUFFER
TM0 : TIMER REG0	SMH : SERIAL MODE High
TM1 : TIMER REG1	SML : SERIAL MODE Low
TMM : TIMER MODE	MKH : MASK High
ETM0 : TIMER/EVENT	MKL : MASK Low
COUNTER REG0	ZCM : ZERO CROSS MODE
ETM1 : TIMER/EVENT	
COUNTER REG1	
ECNT : TIMER/EVENT	
COUNTER UPCOUNTER	
ECPT : TIMER/EVENT	
COUNTER CAPTURE	

2. rp to rp3 (register pair)

SP : STACK POINTER
B : BC
D : DE
H : HL
V : VA
EA : EXTENDED ACCUMULATOR

3. rpa to rpa3 (rp addressing)

B : (BC)
D : (DE)
H : (HL)
D+ : (DE)+
H+ : (HL)+
D- : (DE)-
H- : (HL)-
D++ : (DE)++
H++ : (HL)++
D + byte : (DE + byte)
H + A : (HL + A)
H + B : (HL + B)
H + EA : (HL + EA)
H + byte : (HL + byte)

4. f (flag)

CY : CARRY
HC : HALF CARRY
Z : ZERO

5. irf (interrupt flag)

NMI : NMI INPUT
FT0 : INTFT0
FT1 : INTFT1
F1 : INTF1
F2 : INTF2
FE0 : INTFE0
FE1 : INTFE1
FEIN : INTFEIN
FAD : INTFAD
FSR : INTFSR
FST : INTFST
ER : ERROR
OV : OVERFLOW
AN4 : ANALOG INPUT 4 to 7
to
AN7
SB : STANDBY

6.2 SYMBOL DESCRIPTION OF OPERATION CODE

R ₂	R ₁	R ₀	reg
0	0	0	V
0	0	1	A
0	1	0	B
0	1	1	C
1	0	0	D
1	0	1	E
1	1	0	H
1	1	1	L

T ₂	T ₁	T ₀	reg
0	0	0	EAH
0	0	1	EAL
0	1	0	B
0	1	1	C
1	0	0	D
1	0	1	E
1	1	0	H
1	1	1	L

A ₃	A ₂	A ₁	A ₀	addressing
0	0	0	0	(BC)
0	0	0	1	(DE)
0	0	1	0	(HL)
0	0	1	1	(DE)+
0	1	0	0	(HL)+
0	1	0	1	(DE)-
0	1	1	0	(HL)-
0	1	1	1	(DE + byte)
1	0	1	1	(HL + A)
1	1	0	0	(HL + B)
1	1	0	1	(HL + EA)
1	1	1	0	(HL + byte)
1	1	1	1	(HL + byte)

S ₅	S ₄	S ₃	S ₂	S ₁	S ₀	Special-reg
0	0	0	0	0	0	PA
0	0	0	0	0	1	PB
0	0	0	0	1	0	PC
0	0	0	0	1	1	PD
0	0	0	1	0	1	PF
0	0	0	1	1	0	MKH
0	0	0	1	1	1	MKL
0	0	1	0	0	0	ANM
0	0	1	0	0	1	SMH
0	0	1	0	1	0	SML
0	0	1	0	1	1	EOM
0	0	1	1	0	0	ETMM
0	0	1	1	0	1	TMM
0	1	0	0	0	0	MM
0	1	0	0	0	1	MCC
0	1	0	0	1	0	MA
0	1	0	0	1	1	MB
0	1	0	1	0	0	MC
0	1	0	1	1	1	MF
0	1	1	0	0	0	TXB
0	1	1	0	0	1	RXB
0	1	1	0	1	0	TM0
0	1	1	0	1	1	TM1
1	0	0	0	0	0	CR0
1	0	0	0	0	1	CR1
1	0	0	0	1	0	CR2
1	0	0	0	1	1	CR3
1	0	1	0	0	0	ZCM

C ₃	C ₂	C ₁	C ₀	addressing
0	0	1	0	(DE)
0	0	1	1	(HL)
0	1	0	0	(DE)++
0	1	0	1	(HL)++
1	0	1	1	(DE + byte)
1	1	0	0	(HL + A)
1	1	0	1	(HL + B)
1	1	1	0	(HL + EA)
1	1	1	1	(HL + byte)

I ₄	I ₃	I ₂	I ₁	I ₀	INTF
0	0	0	0	0	NMI
0	0	0	0	1	FT0
0	0	0	1	0	FT1
0	0	0	1	1	F1
0	0	1	0	0	F2
0	0	1	0	1	FE0
0	0	1	1	0	FE1
0	0	1	1	1	FEIN
0	1	0	0	0	FAD
0	1	0	0	1	FSR
0	1	0	1	0	FST
0	1	0	1	1	ER
0	1	1	0	0	OV
1	0	0	0	0	AN4
1	0	0	0	1	AN5
1	0	0	1	0	AN6
1	0	0	1	1	AN7
1	0	1	0	0	SB

U ₀	special-reg
0	ETM0
1	ETM1

V ₀	special-reg
0	ECNT
1	ECPT

P ₂	P ₁	P ₀	reg-pair
0	0	0	SP
0	0	1	BC
0	1	0	DE
0	1	1	HL
1	0	0	EA

Q ₂	Q ₁	Q ₀	reg-pair
0	0	0	VA
0	0	1	BC
0	1	0	DE
0	1	1	HL
1	0	0	EA

F ₂	F ₁	F ₀	flag
0	0	0	—
0	1	0	CY
0	1	1	HC
1	0	0	Z

6.3 INSTRUCTION EXECUTION TIME

1 state shown here is composed of 3 clock cycles. When a clock cycle of 12 MHz is used, the execution time should be 250 ns ($= 3 \times 1/12 \mu$ s). In this case, the 4-state instruction which is the minimum execution time should be execution time of 1 μ s.

Note 1	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition	
			B1	B2	B3	B4				
8-bit data transfer instructions	MOV	r1, A	0 0 0 1 1 T ₂ T ₁ T ₀				4	r1 ← A		
		A, r1	0 0 0 0 1 T ₂ T ₁ T ₀				4	A ← r1		
		* sr, A	0 1 0 0 1 1 0 1	1 1 S ₅ S ₄ S ₃ S ₂ S ₁ S ₀			10	sr ← A		
		* A, sr1	0 1 0 0 1 1 0 0	1 1 S ₅ S ₄ S ₃ S ₂ S ₁ S ₀			10	A ← sr1		
		r, word	0 1 1 1 0 0 0 0	0 1 1 0 1 R ₂ R ₁ R ₀	Low Adrs	High Adrs	17	r ← (word)		
		word, r	0 1 1 1 0 0 0 0	0 1 1 1 1 R ₂ R ₁ R ₀	Low Adrs	High Adrs	17	(word) ← r		
	MVI	* r, byte	0 1 1 0 1 R ₂ R ₁ R ₀	← Data →				7	r ← byte	
		sr2, byte	0 1 1 0 0 1 0 0	S ₃ 0 0 0 0 S ₂ S ₁ S ₀	Data			14	sr2 ← byte	
	MVIW	* wa, byte	0 1 1 1 0 0 0 1	← Offset →		Data		13	(V. wa) ← byte	
	MVIX	* rpa1, byte	0 1 0 0 1 0 A ₁ A ₀	← Data →				10	(rpa1) ← byte	
	STAW	* wa	0 1 1 0 0 0 1 1	← Offset →				10	(V. wa) ← A	
	LDAW	* wa	0 0 0 0 0 0 0 1	← Offset →				10	A ← (V. wa)	
	STAX	* rpa2	A ₃ 0 1 1 1 A ₂ A ₁ A ₀	Data*1				7/13*3	(rpa2) ← A	
	LDAX	* rpa2	A ₃ 0 1 0 1 A ₂ A ₁ A ₀	Data*1				7/13*3	A ← (rpa2)	
	EXX		0 0 0 1 0 0 0 1					4	{ B ↔ B', C ↔ C', D ↔ D' E ↔ E', H ↔ H', L ↔ L' }	
	EXA		0 0 0 1 0 0 0 0					4	V, A ↔ V', A', EA ↔ EA'	
EXH		0 1 0 1 0 0 0 0					4	H, L ↔ H', L'		
BLOCK		0 0 1 1 0 0 0 1					13 (C + 1)	(DE) ⁺ ← (HL) ⁺ ; C ← C - 1 End if borrow		
Note 2	DMOV	rp3, EA	1 0 1 1 0 1 P ₁ P ₀				4	rp3 _L ← EAL, rp3 _H ← EAH		
		EA, rp3	1 0 1 0 0 1 P ₁ P ₀				4	EAL ← rp3 _L , EAH ← rp3 _H		

- Notes**
1. Instruction Group
 2. 16-bit data transfer instructions

Phase-out/Discontinued

Note 1	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition
			B1	B2	B3	B4			
16-bit data transfer instructions	DMOV	sr3, EA	0 1 0 0 1 0 0 0	1 1 0 1 0 0 1 U ₀			14	sr3 ← EA	
		EA, sr4		1 1 0 0 0 0 0 V ₀			14	EA ← sr4	
	SBCD	word	0 1 1 1 0 0 0 0	0 0 0 1 1 1 1 0	Low Adrs	High Adrs	20	(word) ← C, (word + 1) ← B	
	SDED	word		0 0 1 0 1 1 1 0			20	(word) ← E, (word + 1) ← D	
	SHLD	word		0 0 1 1 1 1 1 0			20	(word) ← L, (word + 1) ← H	
	SSPD	word		0 0 0 0 1 1 1 0			20	(word) ← SP _L , (word + 1) ← SP _H	
	STEAX	rpa3	0 1 0 0 1 0 0 0	1 0 0 1 C ₃ C ₂ C ₁ C ₀	Data*2		14/20 ^{*3}	(rpa3) ← EAL, (rpa3 + 1) ← EAH	
	LBCD	word	0 1 1 1 0 0 0 0	0 0 0 1 1 1 1 1	Low Adrs	High Adrs	20	C ← (word), B ← (word + 1)	
	LDED	word		0 0 1 0 1 1 1 1			20	E ← (word), D ← (word + 1)	
	LHLD	word		0 0 1 1 1 1 1 1			20	L ← (word), H ← (word + 1)	
	LSPD	word		0 0 0 0 1 1 1 1			20	SP _L ← (word), SP _H ← (word + 1)	
	LDEAX	rpa3	0 1 0 0 1 0 0 0	1 0 0 0 C ₃ C ₂ C ₁ C ₀	Data*2		14/20 ^{*3}	EAL ← (rpa3), EAH ← (rpa3 + 1)	
	PUSH	rp1	1 0 1 1 0 Q ₂ Q ₁ Q ₀				13	(SP - 1) ← rp1 _H , (SP - 2) ← rp1 _L SP ← SP - 2	
	POP	rp1	1 0 1 0 0 Q ₂ Q ₁ Q ₀				10	rp1 _L ← (SP), rp1 _H ← (SP + 1) SP ← SP + 2	
	LXI *	rp2, word	0 P ₂ P ₁ P ₀ 0 1 0 0	← Low Byte →	High Byte		10	rp2 ← word	
TABLE		0 1 0 0 1 0 0 0	1 0 1 0 1 0 0 0			17	C ← (PC + 3 + A) B ← (PC + 3 + A + 1)		
Note 2	ADD	A, r	0 1 1 0 0 0 0 0	1 1 0 0 0 R ₂ R ₁ R ₀			8	A ← A + r	
		r, A		0 1 0 0			8	r ← r + A	
	ADC	A, r		1 1 0 1			8	A ← A + r + CY	
		r, A		0 1 0 1			8	r ← r + A + CY	

- Notes**
1. Instruction Group
 2. 8-bit operation instructions (register)

Phase-out/Discontinued

Note	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition
			B1	B2	B3	B4			
8-bit operation instructions (register)	ADDNC	A, r	0 1 1 0 0 0 0 0	1 0 1 0 0 R ₂ R ₁ R ₀			8	$A \leftarrow A + r$	No Carry
		r, A		0 0 1 0			8	$r \leftarrow r + A$	No Carry
	SUB	A, r		1 1 1 0			8	$A \leftarrow A - r$	
		r, A		0 1 1 0			8	$r \leftarrow r - A$	
	SBB	A, r		1 1 1 1			8	$A \leftarrow A - r - CY$	
		r, A		0 1 1 1			8	$r \leftarrow r - A - CY$	
	SUBNB	A, r		1 0 1 1			8	$A \leftarrow A - r$	No Borrow
		r, A		0 0 1 1			8	$r \leftarrow r - A$	No Borrow
	ANA	A, r		1 0 0 0 1 R ₂ R ₁ R ₀			8	$A \leftarrow A \wedge r$	
		r, A		0 0 0 0			8	$r \leftarrow r \wedge A$	
	ORA	A, r		1 0 0 1			8	$A \leftarrow A \vee r$	
		r, A		0 0 0 1			8	$r \leftarrow r \vee A$	
	XRA	A, r		1 0 0 1 0 R ₂ R ₁ R ₀			8	$A \leftarrow A \vee r$	
		r, A		0 0 0 1			8	$r \leftarrow r \vee A$	
	GTA	A, r		1 0 1 0 1 R ₂ R ₁ R ₀			8	$A - r - 1$	No Borrow
		r, A		0 0 1 0			8	$r - A - 1$	No Borrow
LTA	A, r		1 0 1 1			8	$A - r$	Borrow	
	r, A		0 0 1 1			8	$r - A$	Borrow	
NEA	A, r		1 1 1 0			8	$A - r$	No Zero	
	r, A		0 1 1 0			8	$r - A$	No Zero	

Note Instruction Group

Phase-out/Discontinued

Note	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition
			B1	B2	B3	B4			
8-bit operation instructions (register)	EQA	A, r	0 1 1 0 0 0 0 0	1 1 1 1 1 R ₂ R ₁ R ₀			8	A - r	Zero
		r, A		0 1 1 1			8	r - A	Zero
	ONA	A, r		1 1 0 0			8	A ^ r	No Zero
	OFFA	A, r		1 1 0 1			8	A ^ r	Zero
8-bit operation instructions (memory)	ADDX	rpa	0 1 1 1 0 0 0 0	1 1 0 0 0 A ₂ A ₁ A ₀			11	A ← A + (rpa)	
	ADCX	rpa		1 1 0 1			11	A ← A + (rpa) + CY	
	ADDNCX	rpa		1 0 1 0			11	A ← A + (rpa)	No Carry
	SUBX	rpa		1 1 1 0			11	A ← A - (rpa)	
	SBBX	rpa		1 1 1 1			11	A ← A - (rpa) - CY	
	SUBNBX	rpa		1 0 1 1			11	A ← A - (rpa)	No Borrow
	ANAX	rpa		1 0 0 0 1 A ₂ A ₁ A ₀			11	A ← A ^ (rpa)	
	ORAX	rpa		1 0 0 1			11	A ← A v (rpa)	
	XRAX	rpa		1 0 0 1 0 A ₂ A ₁ A ₀			11	A ← A v (rpa)	
	GTAX	rpa		1 0 1 0 1 A ₂ A ₁ A ₀			11	A - (rpa) - 1	No Borrow
	LTAX	rpa		1 0 1 1			11	A - (rpa)	Borrow
	NEAX	rpa		1 1 1 0			11	A - (rpa)	No Zero
	EQAX	rpa		1 1 1 1			11	A - (rpa)	Zero
	ONAX	rpa		1 1 0 0			11	A ^ (rpa)	No Zero
OFFAX	rpa		1 1 0 1			11	A ^ (rpa)	Zero	

Note Instruction Group

Phase-out/Discontinued

Note	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition	
			B1	B2	B3	B4				
Immediate data operation instructions	ADI	*	A, byte	0 1 0 0 0 1 1 0	← Data →			7	$A \leftarrow A + \text{byte}$	
		r, byte	0 1 1 1 0 1 0 0	0 1 0 0 0 R ₂ R ₁ R ₀	Data		11	$r \leftarrow r + \text{byte}$		
		sr2, byte	0 1 1 0	S ₃ 1 0 0 0 S ₂ S ₁ S ₀			20	$sr2 \leftarrow sr2 + \text{byte}$		
	ACI	*	A, byte	0 1 0 1 0 1 1 0	← Data →			7	$A \leftarrow A + \text{byte} + CY$	
		r, byte	0 1 1 1 0 1 0 0	0 1 0 1 0 R ₂ R ₁ R ₀	Data		11	$r \leftarrow r + \text{byte} + CY$		
		sr2, byte	0 1 1 0	S ₃ 1 0 1 0 S ₂ S ₁ S ₀			20	$sr2 \leftarrow sr2 + \text{byte} + CY$		
	ADINC	*	A, byte	0 0 1 0 0 1 1 0	← Data →			7	$A \leftarrow A + \text{byte}$	No Carry
		r, byte	0 1 1 1 0 1 0 0	0 0 1 0 0 R ₂ R ₁ R ₀	Data		11	$r \leftarrow r + \text{byte}$	No Carry	
		sr2, byte	0 1 1 0	S ₃ 0 1 0 0 S ₂ S ₁ S ₀			20	$sr2 \leftarrow sr2 + \text{byte}$	No Carry	
	SUI	*	A, byte	0 1 1 0 0 1 1 0	← Data →			7	$A \leftarrow A - \text{byte}$	
		r, byte	0 1 1 1 0 1 0 0	0 1 1 0 0 R ₂ R ₁ R ₀	Data		11	$r \leftarrow r - \text{byte}$		
		sr2, byte	0 1 1 0	S ₃ 1 1 0 0 S ₂ S ₁ S ₀			20	$sr2 \leftarrow sr2 - \text{byte}$		
	SBI	*	A, byte	0 1 1 1 0 1 1 0	← Data →			7	$A \leftarrow A - \text{byte} - CY$	
		r, byte	0 1 1 1 0 1 0 0	0 1 1 1 0 R ₂ R ₁ R ₀	Data		11	$r \leftarrow r - \text{byte} - CY$		
		sr2, byte	0 1 1 0	S ₃ 1 1 1 0 S ₂ S ₁ S ₀			20	$sr2 \leftarrow sr2 - \text{byte} - CY$		
	SUI NB	*	A, byte	0 0 1 1 0 1 1 0	← Data →			7	$A \leftarrow A - \text{byte}$	No Borrow
		r, byte	0 1 1 1 0 1 0 0	0 0 1 1 0 R ₂ R ₁ R ₀	Data		11	$r \leftarrow r - \text{byte}$	No Borrow	
		sr2, byte	0 1 1 0	S ₃ 0 1 1 0 S ₂ S ₁ S ₀			20	$sr2 \leftarrow sr2 - \text{byte}$	No Borrow	
ANI	*	A, byte	0 0 0 0 0 1 1 1	← Data →			7	$A \leftarrow A \wedge \text{byte}$		
	r, byte	0 1 1 1 0 1 0 0	0 0 0 0 1 R ₂ R ₁ R ₀	Data		11	$r \leftarrow r \wedge \text{byte}$			

Note Instruction Group

Phase-out/Discontinued

Note	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition
			B1	B2	B3	B4			
Immediate data operation instructions	ANI	sr2, byte	0 1 1 0 0 1 0 0	S ₃ 0 0 0 1 S ₂ S ₁ S ₀	Data		20	sr2 ← sr2 ∧ byte	
	*	A, byte	0 0 0 1 0 1 1 1	← Data →			7	A ← A ∨ byte	
	ORI	r, byte	0 1 1 1 0 1 0 0	0 0 0 1 1 R ₂ R ₁ R ₀	Data		11	r ← r ∨ byte	
		sr2, byte	0 1 1 0	S ₃ 0 0 1 1 S ₂ S ₁ S ₀			20	sr2 ← sr2 ∨ byte	
	*	A, byte	0 0 0 1 0 1 1 0	← Data →			7	A ← A ∨ byte	
	XRI	r, byte	0 1 1 1 0 1 0 0	0 0 0 1 0 R ₂ R ₁ R ₀	Data		11	r ← r ∨ byte	
		sr2, byte	0 1 1 0	S ₃ 0 0 1 0 S ₂ S ₁ S ₀			20	sr2 ← sr2 ∨ byte	
		*	A, byte	0 0 1 0 0 1 1 1	← Data →			7	A ← A ∨ byte
	GTI	r, byte	0 1 1 1 0 1 0 0	0 0 1 0 1 R ₂ R ₁ R ₀	Data		11	r - byte - 1	No Borrow
		sr2, byte	0 1 1 0	S ₃ 0 1 0 1 S ₂ S ₁ S ₀			14	sr2 - byte - 1	No Borrow
		*	A, byte	0 0 1 1 0 1 1 1	← Data →			7	A - byte
	LTI	r, byte	0 1 1 1 0 1 0 0	0 0 1 1 1 R ₂ R ₁ R ₀	Data		11	r - byte	Borrow
		sr2, byte	0 1 1 0	S ₃ 0 1 1 1 S ₂ S ₁ S ₀			14	sr2 - byte	Borrow
		*	A, byte	0 1 1 0 0 1 1 1	← Data →			7	A - byte
	NEI	r, byte	0 1 1 1 0 1 0 0	0 1 1 0 1 R ₂ R ₁ R ₀	Data		11	r - byte	No Zero
		sr2, byte	0 1 1 0	S ₃ 1 1 0 1 S ₂ S ₁ S ₀			14	sr2 - byte	No Zero
		*	A, byte	0 1 1 1 0 1 1 1	← Data →			7	A - byte
	EQI	r, byte	0 1 1 1 0 1 0 0	0 1 1 1 1 R ₂ R ₁ R ₀	Data		11	r - byte	Zero
sr2, byte		0 1 1 0	S ₃ 1 1 1 1 S ₂ S ₁ S ₀			14	sr2 - byte	Zero	

Note Instruction Group

Phase-out/Discontinued

Note	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition	
			B1	B2	B3	B4				
Immediate data operation instructions	* ONI	A, byte	0 1 0 0 0 1 1 1	← Data →				7	$A \wedge \text{byte}$	No Zero
		r, byte	0 1 1 1 0 1 0 0	0 1 0 0 1 R ₂ R ₁ R ₀	Data			11	$r \wedge \text{byte}$	No Zero
		sr2, byte	0 1 1 0	S ₃ 1 0 0 1 S ₂ S ₁ S ₀				14	$sr2 \wedge \text{byte}$	No Zero
	* OFFI	A, byte	0 1 0 1 0 1 1 1	← Data →				7	$A \wedge \text{byte}$	Zero
		r, byte	0 1 1 1 0 1 0 0	0 1 0 1 1 R ₂ R ₁ R ₀	Data			11	$r \wedge \text{byte}$	Zero
		sr2, byte	0 1 1 0	S ₃ 1 0 1 1 S ₂ S ₁ S ₀				14	$sr2 \wedge \text{byte}$	Zero
Working register operation instructions	ADDW	wa	0 1 1 1 0 1 0 0	1 1 0 0 0 0 0 0	offset			14	$A \leftarrow A + (V. wa)$	
	ADCW	wa		1 1 0 1				14	$A \leftarrow A + (V. wa) + CY$	
	ADDNCW	wa		1 0 1 0				14	$A \leftarrow A + (V. wa)$	No Carry
	SUBW	wa		1 1 1 0				14	$A \leftarrow A - (V. wa)$	
	SBBW	wa		1 1 1 1				14	$A \leftarrow A - (V. wa) - CY$	
	SUBNBW	wa		1 0 1 1				14	$A \leftarrow A - (V. wa)$	No Borrow
	ANAW	wa		1 0 0 0 1 0 0 0				14	$A \leftarrow A \wedge (V. wa)$	
	ORAW	wa		1 0 0 1				14	$A \leftarrow A \vee (V. wa)$	
	XRAW	wa		1 0 0 1 0 0 0 0				14	$A \leftarrow A \nabla (V. wa)$	
	GTAW	wa		1 0 1 0 1 0 0 0				14	$A - (V. wa) - 1$	No Borrow
	LTAW	wa		1 0 1 1				14	$A - (V. wa)$	Borrow
	NEAW	wa		1 1 1 0				14	$A - (V. wa)$	No Zero
	EQAW	wa		1 1 1 1				14	$A - (V. wa)$	Zero
ONAW	wa		1 1 0 0				14	$A \wedge (V. wa)$	No Zero	

Note Instruction Group

Phase-out/Discontinued

Note	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition		
			B1	B2	B3	B4					
Working register operation instructions	OFFAW	wa	0 1 1 1 0 1 0 0	1 1 0 1 1 0 0 0	Offset		14	$A \wedge (V. wa)$	Zero		
	ANIW *	wa, byte	0 0 0 0 0 1 0 1	← Offset →		Data		19	$(V. wa) \leftarrow (V. wa) \wedge \text{byte}$		
	ORIW *	wa, byte	0 0 0 1					19	$(V. wa) \leftarrow (V. wa) \vee \text{byte}$		
	GTIW *	wa, byte	0 0 1 0					13	$(V. wa) - \text{byte} - 1$	No Borrow	
	LTIW *	wa, byte	0 0 1 1					13	$(V. wa) - \text{byte}$	Borrow	
	NEIW *	wa, byte	0 1 1 0					13	$(V. wa) - \text{byte}$	No Zero	
	EQIW *	wa, byte	0 1 1 1					13	$(V. wa) - \text{byte}$	Zero	
	ONIW *	wa, byte	0 1 0 0					13	$(V. wa) \wedge \text{byte}$	No Zero	
	OFFIW	wa, byte	0 1 0 1					13	$(V. wa) \wedge \text{byte}$	Zero	
16-bit operation instructions	EADD	EA, r2	0 1 1 1 0 0 0 0	0 1 0 0 0 0	$R_1 R_0$			11	$EA \leftarrow EA + r2$		
	DADD	EA, rp3		0 1 0 0	1 1 0 0 0 1	$P_1 P_0$			11	$EA \leftarrow EA + rp3$	
	DADC	EA, rp3			1 1 0 1				11	$EA \leftarrow EA + rp3 + CY$	
	DADDNC	EA, rp3			1 0 1 0				11	$EA \leftarrow EA + rp3$	No Carry
	ESUB	EA, r2		0 0 0 0	0 1 1 0 0 0	$R_1 R_0$			11	$EA \leftarrow EA - r2$	
	DSUB	EA, rp3		0 1 0 0	1 1 1 0 0 1	$P_1 P_0$			11	$EA \leftarrow EA - rp3$	
	DSBB	EA, rp3			1 1 1 1				11	$EA \leftarrow EA - rp3 - CY$	
	DSUBNB	EA, rp3			1 0 1 1				11	$EA \leftarrow EA - rp3$	No Borrow
	DAN	EA, rp3			1 0 0 0 1 1	$P_1 P_0$			11	$EA \leftarrow EA \wedge rp3$	
	DOR	EA, rp3			1 0 0 1				11	$EA \leftarrow EA \vee rp3$	
	DXR	EA, rp3			1 0 0 1 0 1	$P_1 P_0$			11	$EA \leftarrow EA \nabla rp3$	

Note Instruction Group

Phase-out/Discontinued

Note 1	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition
			B1	B2	B3	B4			
16-bit operation instructions	DGT	EA, rp3	0 1 1 1 0 1 0 0	1 0 1 0 1 1 P ₁ P ₀			11	EA ← rp3 - 1	No Borrow
	DLT	EA, rp3		1 0 1 1			11	EA ← rp3	Borrow
	DNE	EA, rp3		1 1 1 0			11	EA ← rp3	No Zero
	DEQ	EA, rp3		1 1 1 1			11	EA ← rp3	Zero
	DON	EA, rp3		1 1 0 0			11	EA ∧ rp3	No Zero
	DOFF	EA, rp3		1 1 0 1			11	EA ∧ rp3	Zero
Note 2	MUL	r2	0 1 0 0 1 0 0 0	0 0 1 0 1 1 R ₁ R ₀			32	EA ← A × r2	
	DIV	r2		0 0 1 1			59	EA ← EA + r2, r2 ← Remainder	
Increment/decrement instructions	INR	r2	0 1 0 0 0 0 R ₁ R ₀				4	r2 ← r2 + 1	Carry
	INRW *	wa	0 0 1 0 0 0 0 0	← Offset →			16	(V. wa) ← (V. wa) + 1	Carry
	INX	rp	0 0 P ₁ P ₀ 0 0 1 0				7	rp ← rp + 1	
		EA	1 0 1 0 1 0 0 0				7	EA ← EA + 1	
	DCR	r2	0 1 0 1 0 0 R ₁ R ₀				4	r2 ← r2 - 1	Borrow
	DCRW *	wa	0 0 1 1 0 0 0 0	← Offset →			16	(V. wa) ← (V. wa) - 1	Borrow
DCX	rp	0 0 P ₁ P ₀ 0 0 1 1				7	rp ← rp - 1		
	EA	1 0 1 0 1 0 0 1				7	EA ← EA - 1		
Note 3	DAA		0 1 1 0 0 0 0 1				4	Decimal Adjust Accumulator	
	STC		0 1 0 0 1 0 0 0	0 0 1 0 1 0 1 1			8	CY ← 1	
	CLC			0 0 1 0 1 0 1 0			8	CY ← 0	
	NEGA			0 0 1 1 1 0 1 0			8	A ← \bar{A} + 1	

- Notes**
1. Instruction Group
 2. Multiplication/division instructions
 3. Other operation instructions

Phase-out/Discontinued

Note	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition
			B1	B2	B3	B4			
Rotation/shift instructions	RLD		0 1 0 0 1 0 0 0	0 0 1 1 1 0 0 0			17	Rotate Left Digit	
	RRD			1 0 0 1			17	Rotate Right Digit	
	RLL	r2		0 1 R ₁ R ₀			8	$r_{2m+1} \leftarrow r_{2m}, r_{20} \leftarrow CY, CY \leftarrow r_{27}$	
	RLR	r2		0 0 R ₁ R ₀			8	$r_{2m-1} \leftarrow r_{2m}, r_{27} \leftarrow CY, CY \leftarrow r_{20}$	
	SLL	r2		0 0 1 0 0 1 R ₁ R ₀			8	$r_{2m+1} \leftarrow r_{2m}, r_{20} \leftarrow 0, CY \leftarrow r_{27}$	
	SLR	r2		0 0 R ₁ R ₀			8	$r_{2m-1} \leftarrow r_{2m}, r_{27} \leftarrow 0, CY \leftarrow r_{20}$	
	SLLC	r2		0 0 0 0 0 1 R ₁ R ₀			8	$r_{2m+1} \leftarrow r_{2m}, r_{20} \leftarrow 0, CY \leftarrow r_{27}$	Carry
	SLRC	r2		0 0 R ₁ R ₀			8	$r_{2m-1} \leftarrow r_{2m}, r_{27} \leftarrow 0, CY \leftarrow r_{20}$	Carry
	DRLL	EA		1 0 1 1 0 1 0 0			8	$EA_{n+1} \leftarrow EA_n, EA_0 \leftarrow CY, CY \leftarrow EA_{15}$	
	DRLR	EA		0 0 0 0			8	$EA_{n-1} \leftarrow EA_n, EA_{15} \leftarrow CY, CY \leftarrow EA_0$	
	DSLL	EA		1 0 1 0 0 1 0 0			8	$EA_{n+1} \leftarrow EA_n, EA_0 \leftarrow 0, CY \leftarrow EA_{15}$	
	DSLRL	EA		0 0 0 0			8	$EA_{n-1} \leftarrow EA_n, EA_{15} \leftarrow 0, CY \leftarrow EA_0$	
Jump instructions	JMP *	word	0 1 0 1 0 1 0 0	← Low Adrs →	High Adrs		10	PC ← word	
	JB		0 0 1 0 0 0 0 1				4	$PC_H \leftarrow B, PC_L \leftarrow C$	
	JR	word	1 1 ← jdisp 1 →				10	PC ← PC + 1 + jdisp 1	
	JRE *	word	0 1 0 0 1 1 1 ← jdisp →				10	PC ← PC + 2 + jdisp	
	JEA		0 1 0 0 1 0 0 0	0 0 1 0 1 0 0 0			8	PC ← EA	
Call Instructions	CALL *	word	0 1 0 0 0 0 0 0	← Low Adrs →	High Adrs		16	$(SP - 1) \leftarrow (PC + 3)_H, (SP - 2) \leftarrow (PC + 3)_L$ PC ← word, SP ← SP - 2	
	CALB		0 1 0 0 1 0 0 0	0 0 1 0 1 0 0 1			17	$(SP - 1) \leftarrow (PC + 2)_H, (SP - 2) \leftarrow (PC + 2)_L$ $PC_H \leftarrow B, PC_L \leftarrow C, SP \leftarrow SP - 2$	
	CALF *	word	0 1 1 1 1 ← fa →				13	$(SP - 1) \leftarrow (PC + 2)_H, (SP - 2) \leftarrow (PC + 2)_L$ $PC_{15-11} \leftarrow 00001, PC_{10-0} \leftarrow fa, SP \leftarrow SP - 2$	

Note Instruction Group

Note 1	Mnemonic	Operand	Operation Code				State	Operation	Skip Condition
			B1	B2	B3	B4			
Note 2	CALT	word	1 0 0 ← ta →				16	$(SP - 1) \leftarrow (PC + 1)_H, (SP - 2) \leftarrow (PC + 1)_L$ $PC_L \leftarrow (128 + 2ta), PC_H \leftarrow (129 + 2ta), SP \leftarrow SP - 2$	
	SOFTI		0 1 1 1 0 0 1 0				16	$(SP - 1) \leftarrow PSW, (SP - 2) \leftarrow (PC + 1)_H, (SP - 3) \leftarrow (PC + 1)_L, PC \leftarrow 0060H, SP \leftarrow SP - 3$	
Return instructions	RET		1 0 1 1 1 0 0 0				10	$PC_L \leftarrow (SP), PC_H \leftarrow (SP + 1)$ $SP \leftarrow SP + 2$	
	RETS		↓ 1 0 0 1				10	$PC_L \leftarrow (SP), PC_H \leftarrow (SP + 1), SP \leftarrow SP + 2$ $PC \leftarrow PC + n$	Unconditional skip
	RETI		0 1 1 0 0 0 1 0				13	$PC_L \leftarrow (SP), PC_H \leftarrow (SP + 1)$ $PSW \leftarrow (SP + 2), SP \leftarrow SP + 3$	
	BIT *	bit, wa	0 1 0 1 1 B ₂ B ₁ B ₀ ← Offset →				10	Skip if (V. wa) bit = 1	(V. wa)bit = 1
Skip instructions	SK	f	0 1 0 0 1 0 0 0	0 0 0 0 1 F ₂ F ₁ F ₀			8	Skip if f = 1	f = 1
	SKN	f	↓	0 0 0 1 ↓			8	Skip if f = 0	f = 0
	SKIT	irf	↓	0 1 0 I ₄ I ₃ I ₂ I ₁ I ₀			8	Skip if irf = 1, then reset irf	irf = 1
	SKNIT	irf	↓	0 1 1 I ₄ I ₃ I ₂ I ₁ I ₀			8	Skip if irf = 0 Reset irf, if irf = 1	irf = 0
	CPU control instructions	NOP		0 0 0 0 0 0 0 0				4	No Operation
	EI		1 0 1 0 1 0 1 0				4	Enable Interrupt	
	DI		1 0 1 1 1 0 1 0				4	Disable Interrupt	
	HLT		0 1 0 0 1 0 0 0	0 0 1 1 1 0 1 1			12	Set Halt Mode	
	STOP		0 1 0 0 1 0 0 0	1 0 1 1 1 0 1 1			12	Set Stop Mode	

- * 1. Data is B2 if rpa2 = D + byte, H + byte.
 2. Data is B3 if rpa3 = D + byte, H + byte.
 3. In the State item, a figure is in the right side of slash if rpa2 and rpa3 are D + byte, H + A, H + B, H + EA, H + byte.

Remark The idle state when each instruction is skipped is different from the execution state as shown below.

1-byte instruction	: 4 states	3-byte instruction (with *)	: 10 states
2-byte instruction (with *)	: 7 states	3-byte instruction	: 11 states
2-byte instruction	: 8 states	4-byte instruction	: 14 states

- Notes** 1. Instruction Group
 2. Call instructions

7. LIST OF MODE REGISTERS

Name of Mode Registers		Read/Write	Function
MA	MODE A register	W	Specifies bit-wise the input/output of the port A.
MB	MODE B register	W	Specifies bit-wise the input/output of the port B.
MCC	MODE CONTROL C register	W	Specifies bit-wise the port/control mode of the port C.
MC	MODE C register	W	Specifies bit-wise the input/output of the port C which is in port mode.
MM	MEMORY MAPPING register	W	Specifies the port/extension mode of port D and port F.
MF	MODE F register	W	Specifies bit-wise the input/output of the port F which is in port mode.
TMM	Timer mode register	R/W	Specifies operating mode of timer.
ETMM	Timer/event counter mode register	W	Specifies the operating mode of timer/event counter.
EOM	Timer/event counter output mode register	R/W	Control the output level of CO0 and CO1.
SML	Serial mode register	W	Specifies the operating mode of serial interface.
SMH		R/W	
MKL	Interrupt mask register	R/W	Specifies the enable/disable of the interrupt request.
MKH			
ANM	A/D channel mode register	R/W	Specifies the operating mode of A/D converter.
ZCM	Zero-cross mode register	W	Specifies the operation of zero-cross detector circuit.

8. ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS (T_A = 25 °C)

Parameter	Symbol	Test Conditions	Rating	Unit
Power supply voltage	V _{DD}		-0.5 to +7.0	V
	AV _{DD}		AV _{SS} to V _{DD} +0.5	V
	AV _{SS}		-0.5 to +0.5	V
Input voltage	V _I		-0.5 to V _{DD} +0.5	V
Output voltage	V _O		-0.5 to V _{DD} +0.5	V
Output current low	I _{OL}	All output pins	4.0	mA
		Total of all output pins	100	mA
Output current high	I _{OH}	All output pins	-2.0	mA
		Total of all output pins	-50	mA
A/D converter reference input voltage	V _{AREF}		-0.5 to AV _{DD} +0.3	V
Operating ambient temperature	T _A		-40 to +85	°C
Storage temperature	T _{stg}		-65 to +150	°C

Caution Even if one of the parameters exceeds its absolute maximum rating even momentarily, the quality of the product may be degraded. The absolute maximum rating therefore specifies the upper or lower limit of the value at which the product can be used without physical damages. Be sure not to exceed or fall below this value when using the product.

OSCILLATOR CHARACTERISTICS ($T_A = -40$ to $+85$ °C, $V_{DD} = AV_{DD} = +5.0$ V ± 10 %, $V_{SS} = AV_{SS} = 0$ V, $V_{DD} - 0.8$ V $\leq AV_{DD} \leq V_{DD}$, 3.4 V $\leq V_{AREF} \leq AV_{DD}$)

Resonator	Recommended Circuit	Parameter	Test Conditions	MIN.	MAX.	Unit
Ceramic ^{Note1} or crystal resonator ^{Note2}		Oscillator frequency (f_{xx})	A/D converter not used	4	15	MHz
			A/D converter used	5.8	15	MHz
External clock		X1 input frequency (f_x)	A/D converter not used	4	15	MHz
			A/D converter used	5.8	15	MHz
		X1 rise time, fall time (t_r, t_f)		0	20	ns
		X1 input high, low level width (t_{0H}, t_{0L})		20	250	ns

- Cautions**
1. Place oscillator circuit as close as possible to X1, X2 pins.
 2. Ensure that no other signal lines pass through the shadow area.

Note 1. The ceramic oscillators and external capacitance given in the following table are recommended.

	Manufacturer	Product Name	Recommended Constants	
			C1[pF]	C2[pF]
15 MHz	Murata Mfg. Co., Ltd	CSA15.0MX3	22	22
		CSA12.0MT	30	30
		CST12.0MT	On-chip	On-chip
		CSA10.0MT	30	30
		CST10.0MT	On-chip	On-chip
		CSA6.00MG	30	30
		CST6.00MG	On-chip	On-chip
	TDK Corp.	FCR12.0MC	On-chip	On-chip
12 MHz	Murata Mfg. Co., Ltd	CSA12.0MT18	30	30
		CST12.0MT18	On-chip	On-chip

Note 2. When a crystal oscillator is used, the following external capacitance is recommended.
 $C1 = C2 = 10$ pF

CAPACITANCE (T_A = 25 °C, V_{DD} = V_{SS} = 0 V)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Input capacitance	C _i	f _c = 1 MHz Unmeasured pins returned to 0 V			10	pF
Output capacitance	C _o				20	pF
Input-output capacitance	C _{io}				20	pF

DC CHARACTERISTICS (T_A = -40 to +85 °C, V_{DD} = AV_{DD} = +5.0 V ±10 %, V_{SS} = AV_{SS} = 0 V)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Input voltage low	V _{IL1}	All except $\overline{\text{RESET}}$, $\overline{\text{STOP}}$, $\overline{\text{NMI}}$, $\overline{\text{SCK}}$, INT1, TI, AN4 to AN7	0		0.8	V
	V _{IL2}	$\overline{\text{RESET}}$, $\overline{\text{STOP}}$, $\overline{\text{NMI}}$, $\overline{\text{SCK}}$, INT1, TI, AN4 to AN7	0		0.2 V _{DD}	V
Input voltage high	V _{1IH}	All except $\overline{\text{RESET}}$, $\overline{\text{STOP}}$, $\overline{\text{NMI}}$, $\overline{\text{SCK}}$, INT1, TI, AN4 to AN7, X1, X2	2.2		V _{DD}	V
	V _{IH2}	$\overline{\text{RESET}}$, $\overline{\text{STOP}}$, $\overline{\text{NMI}}$, $\overline{\text{SCK}}$, INT1, TI, AN4 to AN7, X1, X2	0.8 V _{DD}		V _{DD}	V
Output voltage low	V _{OL}	I _{OL} = 2.0 mA			0.45	V
Output voltage high	V _{OH}	I _{OH} = -1.0 mA	V _{DD} -1.0			V
		I _{OH} = -100 μA	V _{DD} -0.5			V
Input current	I _I	INT1 ^{Note 1} , TI(PC3) ^{Note 2} ; 0 V ≤ V _I ≤ V _{DD}			±200	μA
Input leakage current	I _{LI}	All except INT1, TI (PC3), 0 V ≤ V _I ≤ V _{DD}			±10	μA
Output leakage current	I _{LO}	0 V ≤ V _O ≤ V _{DD}			±10	μA
AV _{DD} power supply current	A _{IDD1}	Operating mode f _{XX} = 15 MHz		0.5	1.3	mA
	A _{IDD2}	STOP mode		10	20	μA
V _{DD} power supply current ^{Note 4}	I _{DD1}	Operating mode f _{XX} = 15 MHz		16	30	mA
	I _{DD2}	HALT mode f _{XX} = 15 MHz		8	15	mA
Data retention voltage	V _{DDDR}	Hardware/software STOP mode	2.5			V
Data retention current ^{Note 4}	I _{DDDR}	Hardware/software ^{Note 3} V _{DDDR} = 2.5 V		1	15	μA
		STOP mode V _{DDDR} = 5 V ±10%		10	50	μA
Pull-up resistor	R _L	Ports A, B and C 3.5 V ≤ V _{DD} ≤ 5.5 V, V _I = 0 V	22	50	150	kΩ

- Notes 1.** If self-bias should be generated by ZCM register.
- 2.** If the control mode is set by MCC register, and self-bias should be generated by ZCM register.
- 3.** If self-bias is not generated.
- 4.** Does not include the current flowing through the internal pull-up resistor.

AC CHARACTERISTICS ($T_A = -40$ to $+85$ °C, $V_{DD} = AV_{DD} = +5.0$ V ± 10 %, $V_{SS} = AV_{SS} = 0$ V)
Read/write Operation:

Parameter	Symbol	Test Conditions	MIN.	MAX.	Unit
X1 input cycle time	t _{CYC}		66	250	ns
Address setup time (to ALE ↓)	t _{AL}	f _{XX} = 15 MHz, C _L = 100 pF	30		ns
Address hold time (from ALE ↓)	t _{LA}		35		ns
\overline{RD} ↓ delay time from address	t _{AR}		100		ns
Address float time from \overline{RD} ↓	t _{AFR}	C _L = 100 pF		20	ns
Data input time from address	t _{AD}	f _{XX} = 15 MHz, C _L = 100 pF		250	ns
Data input time from ALE ↓	t _{LDR}			135	ns
Data input time from \overline{RD} ↓	t _{RD}			120	ns
\overline{RD} ↓ delay time from ALE ↓	t _{LR}		15		ns
Data hold time (from \overline{RD} ↑)	t _{RDH}	C _L = 100 pF	0		ns
ALE ↑ delay time from \overline{RD} ↑	t _{RL}	f _{XX} = 15 MHz, C _L = 100 pF	80		ns
\overline{RD} low level width	t _{RR}	In Data Read f _{XX} = 15 MHz, C _L = 100 pF	215		ns
		In OP Code Fetch f _{XX} = 15 MHz, C _L = 100 pF	415		ns
ALE high level width	t _{LL}	f _{XX} = 15 MHz, C _L = 100 pF	90		ns
$\overline{M1}$ setup time (to ALE ↓)	t _{ML}	f _{XX} = 15 MHz	30		ns
$\overline{M1}$ hold time (from ALE ↓)	t _{LM}		35		ns
$\overline{IO/M}$ setup time (to ALE ↓)	t _{IL}		30		ns
$\overline{IO/M}$ hold time (from ALE ↓)	t _{LI}		35		ns
\overline{WR} ↓ delay time from address	t _{AW}	f _{XX} = 15 MHz, C _L = 100 pF	100		ns
Data output time from ALE ↓	t _{LDW}			180	ns
Data output time from \overline{WR} ↓	t _{WD}	C _L = 100 pF		100	ns
\overline{WR} ↓ delay time from ALE ↓	t _{LW}	f _{XX} = 15 MHz, C _L = 100 pF	15		ns
Data setup time (to \overline{WR} ↑)	t _{DW}		165		ns
Data hold time (from \overline{WR} ↑)	t _{WDH}		60		ns
ALE ↑ delay time from \overline{WR} ↑	t _{WL}		80		ns
\overline{WR} low level width	t _{WW}		215		ns

Serial Operation :

Parameter	Symbol	Test Conditions	MIN.	MAX.	Unit
$\overline{\text{SCK}}$ cycle time	t_{CYK}	$\overline{\text{SCK}}$ input	Note 1	800	ns
			Note 2	400	ns
		$\overline{\text{SCK}}$ output		1.6	μs
$\overline{\text{SCK}}$ low level width	t_{KKL}	$\overline{\text{SCK}}$ input	Note 1	335	ns
			Note 2	160	ns
		$\overline{\text{SCK}}$ output		700	ns
$\overline{\text{SCK}}$ high level width	t_{KKH}	$\overline{\text{SCK}}$ input	Note 1	335	ns
			Note 2	160	ns
		$\overline{\text{SCK}}$ output		700	ns
RxD setup time (to $\overline{\text{SCK}} \uparrow$)	t_{RXK}	Note 1	80	ns	
RxD hold time (from $\overline{\text{SCK}} \uparrow$)	t_{KRX}	Note 1	80	ns	
TxD delay time from $\overline{\text{SCK}} \downarrow$	t_{KTX}	Note 1		210	ns

- Notes 1.** If clock rate is × 1 in asynchronous mode, synchronous mode, or I/O interface mode.
2. If clock rate is × 16 or × 64 in asynchronous mode.

Remark The numeric values in the table are those when $f_{\text{XX}} = 15 \text{ MHz}$, $C_{\text{L}} = 100 \text{ pF}$.

Zero-Cross Characteristics :

Parameter	Symbol	Test Conditions	MIN.	MAX.	Unit
Zero-cross detection input	V_{ZX}	AC combination 60 Hz sine wave	1	1.8	$V_{\text{AC P-P}}$
Zero-cross accuracy	A_{ZX}			±135	mV
Zero-cross detection input frequency	f_{ZX}		0.05	1	kHz

Other Operation :

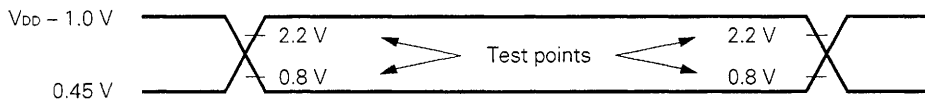
Parameter	Symbol	Test Conditions	MIN.	MAX.	Unit
TI high, low level width	$t_{\text{TIH}}, t_{\text{TIL}}$		6		tcyc
CI high, low level width	$t_{\text{CI1H}}, t_{\text{CI1L}}$	• Event count mode • Frequency test mode	6		tcyc
	$t_{\text{CI2H}}, t_{\text{CI2L}}$	• Pulse width test mode • ECNT latch, clear input • INTEIN set input	48		tcyc
$\overline{\text{NMI}}$ high, low level width	$t_{\text{NIH}}, t_{\text{NIL}}$		10		μs
INT1 high, low level width	$t_{\text{I1H}}, t_{\text{I1L}}$		36		tcyc
$\overline{\text{INT2}}$ high, low level width	$t_{\text{I2H}}, t_{\text{I2L}}$		36		tcyc
$\overline{\text{RESET}}$ high, low level width	$t_{\text{RSH}}, t_{\text{RSL}}$		10		μs

A/D CONVERTER CHARACTERISTICS ($T_A = -40$ to $+85$ °C, $V_{DD} = +5.0$ V ± 10 %, $V_{SS} = AV_{SS} = 0$ V, $V_{DD} - 0.5$ V $\leq AV_{DD} \leq V_{DD}$, 3.4 V $\leq V_{AREF} \leq AV_{DD}$)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Resolution			8			Bits
Absolute accuracy ^{Note}		3.4 V $\leq V_{AREF} \leq AV_{DD}$, 66 ns $\leq t_{CYC} \leq 170$ ns			$\pm 0.8\%$	FSR
		4.0 V $\leq V_{AREF} \leq AV_{DD}$, 66 ns $\leq t_{CYC} \leq 170$ ns			$\pm 0.6\%$	FSR
		$T_A = -10$ to $+70$ °C, 4.0 V $\leq V_{AREF} \leq AV_{DD}$, 66 ns $\leq t_{CYC} \leq 170$ ns			$\pm 0.4\%$	FSR
Conversion time	t_{CONV}	66 ns $\leq t_{CYC} \leq 110$ ns	576			tcyc
		110 ns $\leq t_{CYC} \leq 170$ ns	432			tcyc
Sampling time	t_{SAMP}	66 ns $\leq t_{CYC} \leq 110$ ns	96			tcyc
		110 ns $\leq t_{CYC} \leq 170$ ns	72			tcyc
Analog input voltage	V_{IAN}		0		V_{AREF}	V
Analog input impedance	R_{AN}			50		MΩ
Reference voltage	V_{AREF}		3.4		AV_{DD}	V
V_{AREF} current	I_{AREF1}	Operating mode		1.5	3.0	mA
	I_{AREF2}	STOP mode		0.7	1.5	mA
AV_{DD} power supply current	AI_{DD1}	Operating mode $f_{XX} = 15$ MHz		0.3	1.3	mA
	AI_{DD2}	STOP mode		10	20	μA

Note Quantization error ($\pm 1/2$ LSB) is not included.

AC Timing Test Point



tcyc-Dependent AC Characteristics Expression

Parameter	Expression	MIN./MAX.	Unit
tAL	2T - 100	MIN.	ns
tLA	T - 30	MIN.	ns
tAR	3T - 100	MIN.	ns
tAD	7T - 220	MAX.	ns
tLDR	5T - 200	MAX.	ns
tRD	4T - 150	MAX.	ns
tLR	T - 50	MIN.	ns
trl	2T - 50	MIN.	ns
tRR	4T - 50 (In data read)	MIN.	ns
	7T - 50 (In OP code fetch)		
tLL	2T - 40	MIN.	ns
tML	2T - 100	MIN.	ns
tLM	T - 30	MIN.	ns
tIL	2T - 100	MIN.	ns
tLI	T - 30	MIN.	ns
tAW	3T - 100	MIN.	ns
tLDW	T + 110	MAX.	ns
tLW	T - 50	MIN.	ns
tDW	4T - 100	MIN.	ns
tWDH	2T - 70	MIN.	ns
tWL	2T - 50	MIN.	ns
tWW	4T - 50	MIN.	ns
tcyk	6T ($\overline{\text{SCK}}$ input) ^{Note 1} /12T ($\overline{\text{SCK}}$ input) ^{Note 2}	MIN.	ns
	24T ($\overline{\text{SCK}}$ output)		
tKKL	2.5T + 5 ($\overline{\text{SCK}}$ input) ^{Note 1} /5T + 5 ($\overline{\text{SCK}}$ input) ^{Note 2}	MIN.	ns
	12T - 100 ($\overline{\text{SCK}}$ output)		
tKKh	5T + 5 ($\overline{\text{SCK}}$ input) ^{Note 1} /5T + 5 ($\overline{\text{SCK}}$ input) ^{Note 2}	MIN.	ns
	12T - 100 ($\overline{\text{SCK}}$ output)		

Notes 1. If clock rate is 16 × 64, in asynchronous mode.

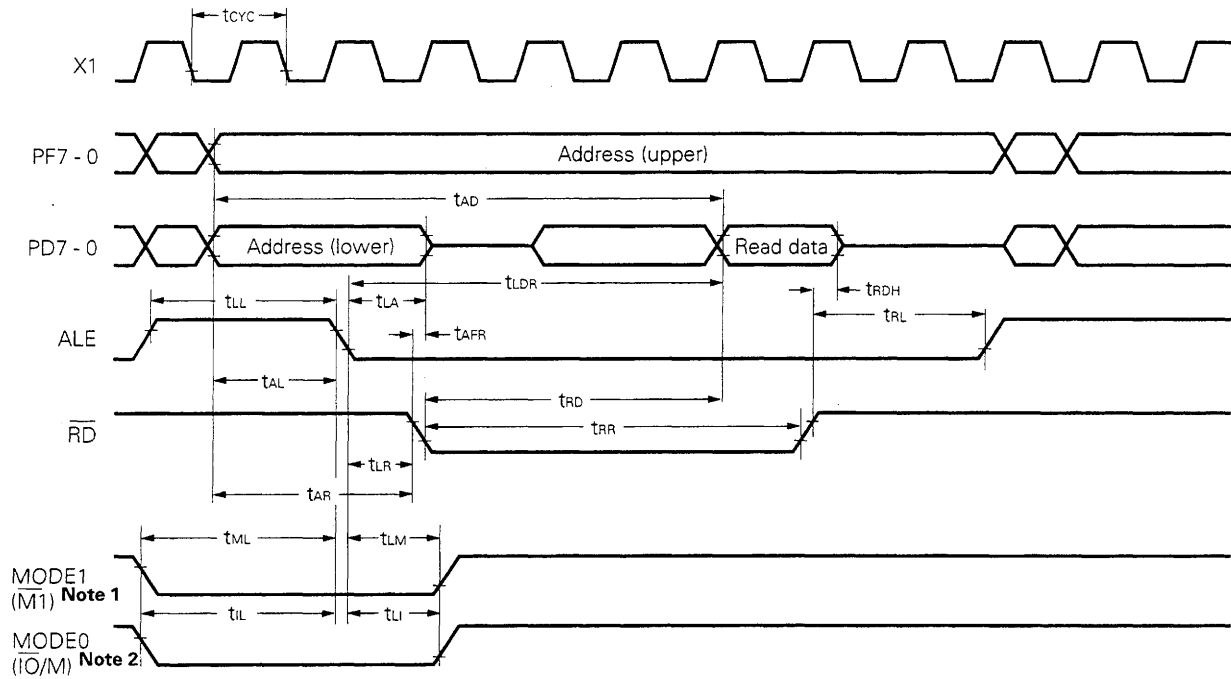
2. If clock rate is ×1, in asynchronous mode, synchronous mode, or I/O interface mode.

Remarks 1. T = tcyc = 1/fxx

2. Other items which are not listed in this table are not dependent on oscillator frequency (fxx).

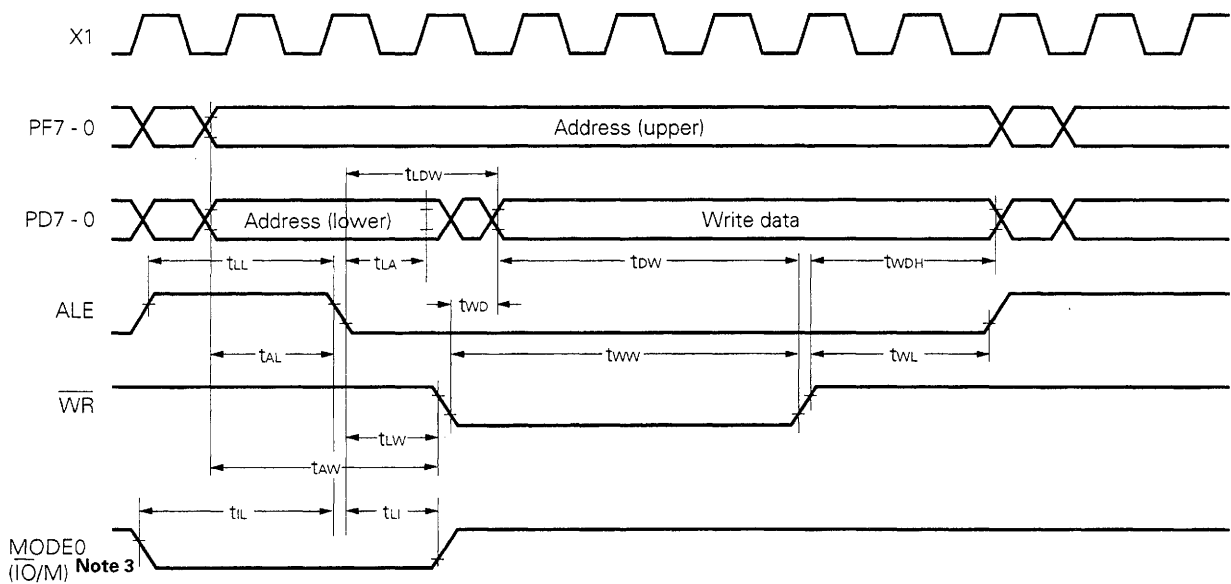
Timing Waveform

Read operation



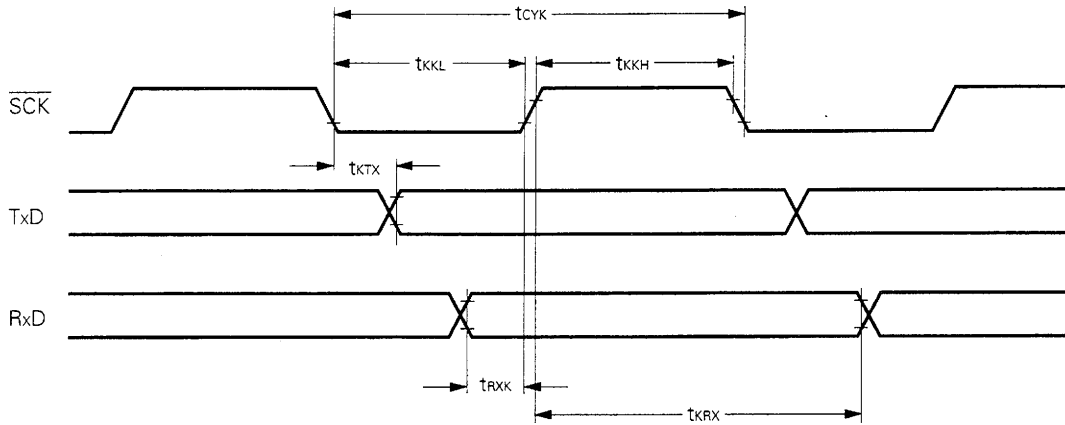
- Note 1.** When MODE1 pin is pulled up, $\overline{M1}$ signal is output to MODE1 pin in the 1st OP code fetch cycle.
- Note 2.** When MODE0 pin is pulled up, $\overline{IO/M}$ signal is output to MODE0 pin in sr to sr2 register read cycle.

Write operation

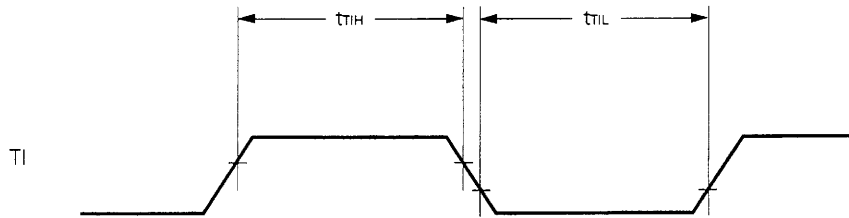


- Note 3.** When MODE0 pin is pulled up, $\overline{IO/M}$ signal is output to MODE0 pin in sr to sr2 register write cycle.

Serial Operation

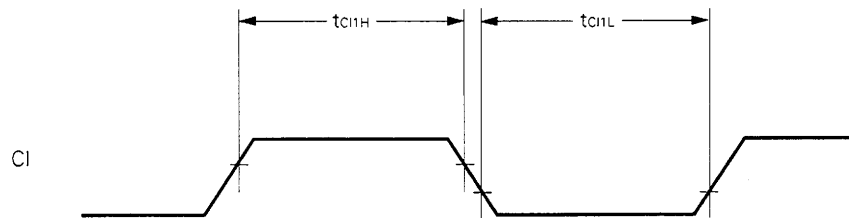


Timer Input Timing

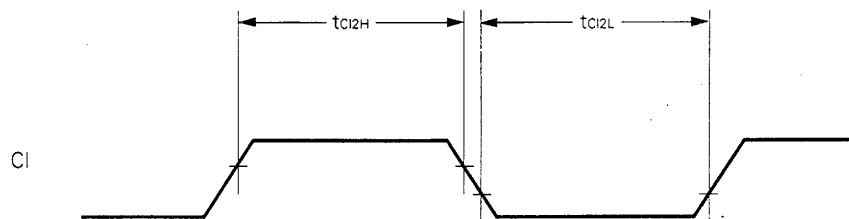


Timer/Event Counter Input Timing

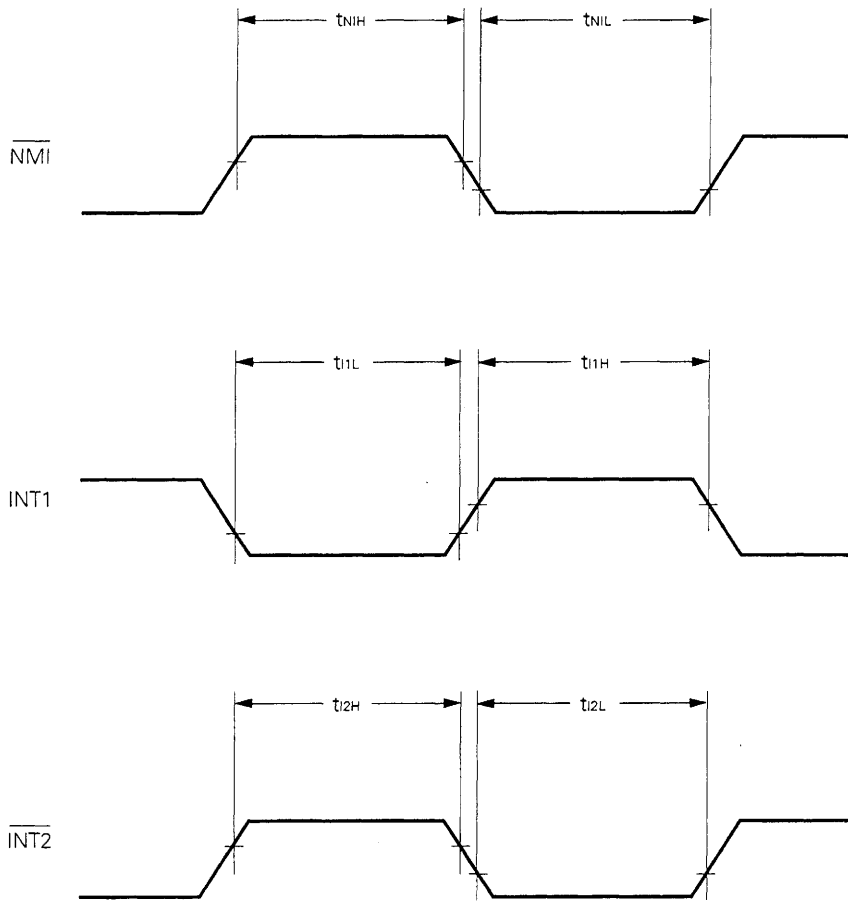
Event counter mode



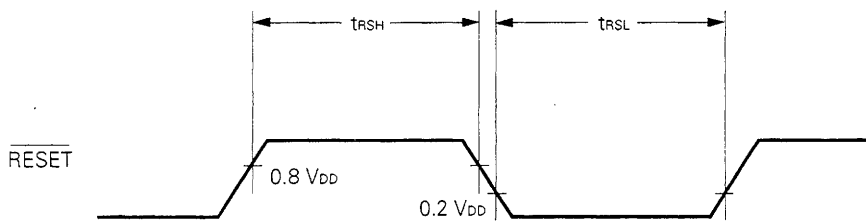
Pulse width test mode



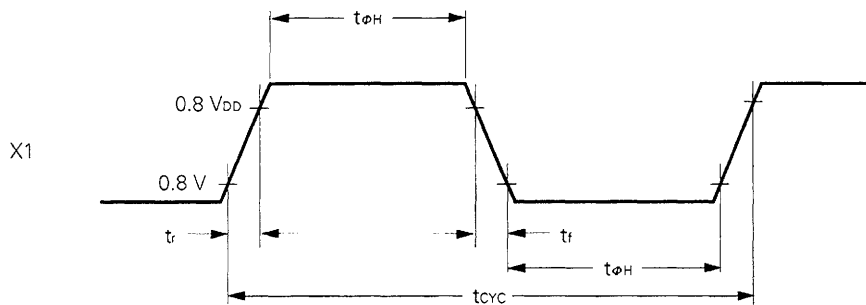
Interrupt Input Timing



Reset Input Timing



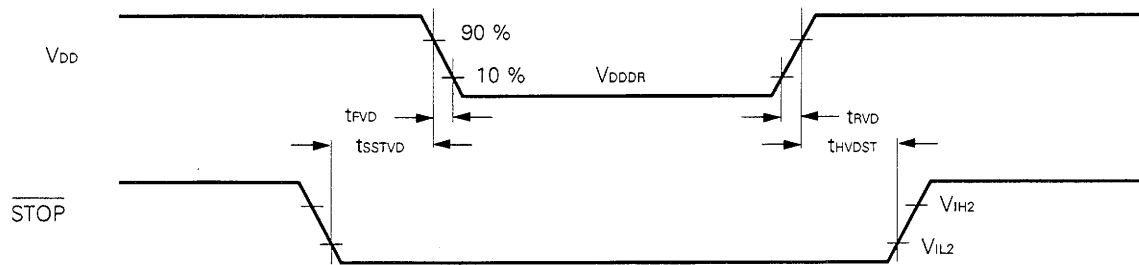
External Clock Timing



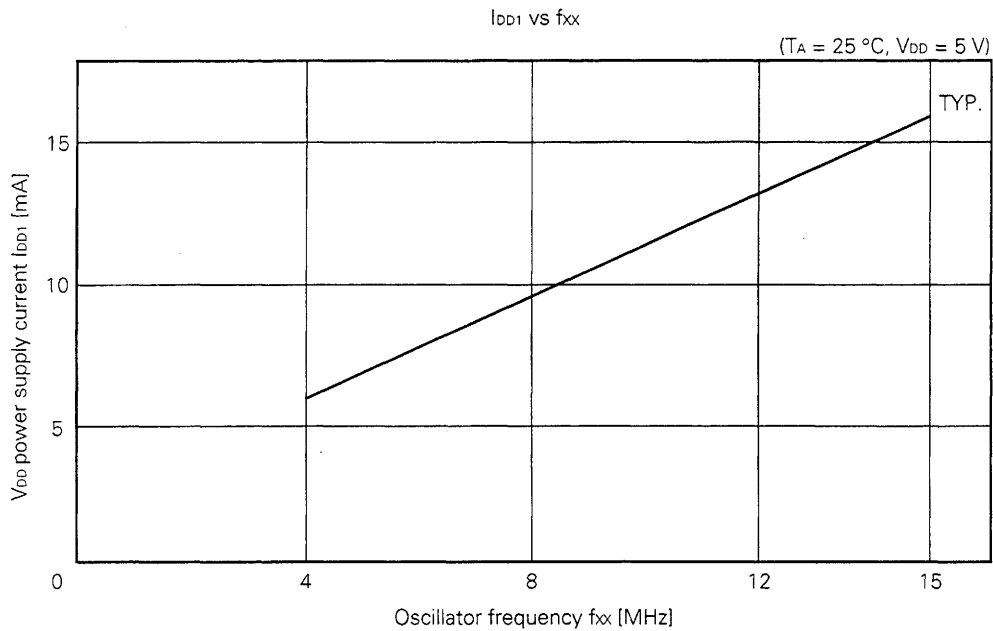
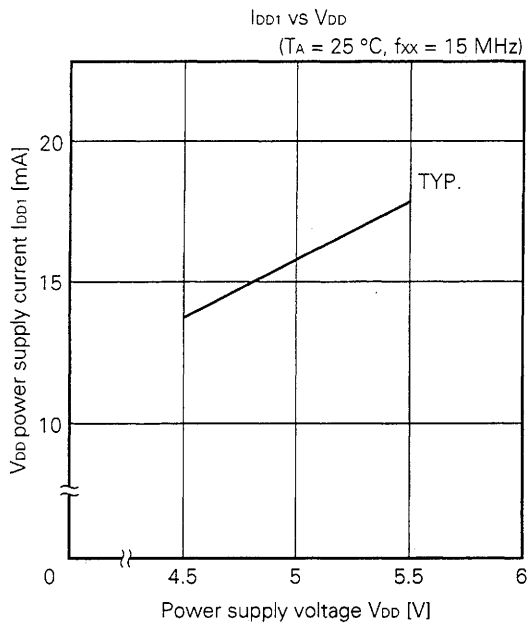
DATA MEMORY STOP MODE LOW POWER SUPPLY VOLTAGE DATA RETENTION CHARACTERISTICS
 (T_A = -40 to +85 °C)

Parameter	Symbol	Test Conditions	MIN.	TYP.	MAX.	Unit
Data retention power supply voltage	V _{DDDR}		2.5		5.5	V
Data retention power supply current	I _{DDDR}	V _{DDDR} = 2.5 V		1	15	μ A
		V _{DDDR} = 5 V \pm 10%		10	50	μ A
V _{DD} rise/fall time	t _{rVD} , t _{fVD}		200			μ s
$\overline{\text{STOP}}$ setup time (to V _{DD})	t _{sSTVD}		12T + 0.5			μ s
$\overline{\text{STOP}}$ hold time (from V _{DD})	t _{hVDST}		12T + 0.5			μ s

Data Retention Timing

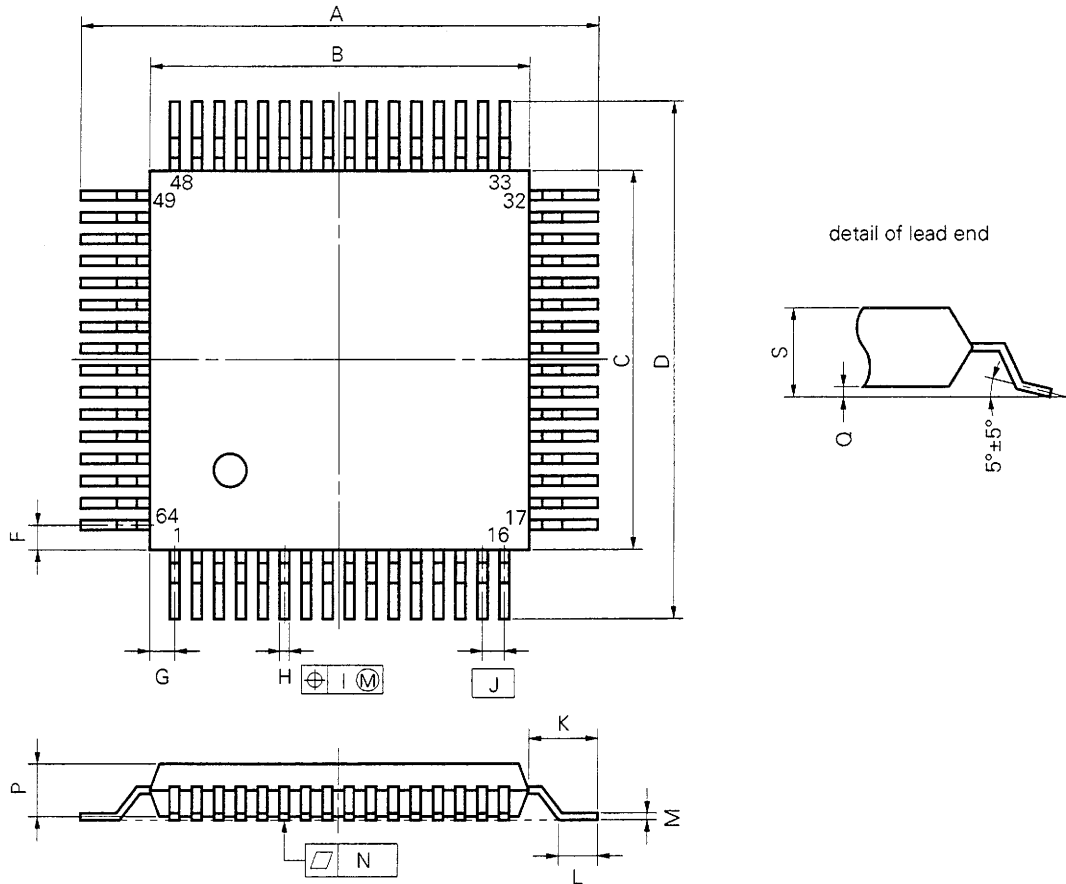


9. CHARACTERISTIC CURVES (REFERENCE VALUES)



10. PACKAGE DRAWINGS

64 PIN PLASTIC QFP (□14)



P64GC-80-AB8-3

NOTE

Each lead centerline is located within 0.15 mm (0.006 inch) of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS	INCHES
A	17.6±0.4	0.693±0.016
B	14.0±0.2	0.551 ^{+0.009} _{-0.008}
C	14.0±0.2	0.551 ^{+0.009} _{-0.008}
D	17.6±0.4	0.693±0.016
F	1.0	0.039
G	1.0	0.039
H	0.35±0.10	0.014 ^{+0.004} _{-0.005}
I	0.15	0.006
J	0.8 (T.P.)	0.031 (T.P.)
K	1.8±0.2	0.071±0.008
L	0.8±0.2	0.031 ^{+0.009} _{-0.008}
M	0.15 ^{+0.10} _{-0.05}	0.006 ^{+0.004} _{-0.003}
N	0.10	0.004
P	2.55	0.100
Q	0.1±0.1	0.004±0.004
S	2.85 MAX.	0.112 MAX.

11. RECOMMENDED SOLDERING CONDITIONS

The μPD78C14A should be soldered and mounted under the conditions recommended in the table below.

For detail of recommended soldering conditions, refer to the information document "**Semiconductor Device Mounting Technology Manual**" (IEI-1207).

For soldering methods and conditions other than those recommended below, contact our sales personnel.

Table 11-1. Surface Mounting Type Soldering Conditions

μPD78C14AG-xxx-AB8 : 64-pin plastic QFP (14 × 14mm)

(1) K, E, P type products

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature : 235 °C, Duration : 30 sec. max. (210 °C min.), Number of times : 2 max. <Points to note> (1) Start the second reflow after the device temperature by the first reflow returns to normal. (2) Flux washing by the water after the first reflow should be avoided.	IR35-00-2
VPS	Package peak temperature : 215 °C, Duration : 40 sec. max. (200 °C min.), Number of times : 2 max. <Points to note> (1) Start the second reflow after the device temperature by the first reflow returns to normal. (2) Flux washing by the water after the first reflow should be avoided.	VP15-00-2
Wave soldering	Solder bath temperature : 260 °C max., Duration : 10 sec. max., Number of times : 1 Pre-heating temperature : 120 °C max. (package surface temperature)	WS60-00-1
Pin part heating	Pin temperature : 300 °C max., Duration: 3 sec. max. (per device side)	—

Caution Do not use two or more soldering methods in combination (except the pin part heating method).

(2) Other products

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature : 230 °C, Duration : 30 sec. max. (210 °C min.), Number of times : 1, Days : 7 days ^{Note} (after that, prebaking is necessary for 10 hours at 125 °C)	IR30-107-1
VPS	Package peak temperature : 215 °C, Duration : 40 sec. max. (200 °C min.), Number of times : 1, Days : 7 days ^{Note} (after that, prebaking is necessary for 10 hours at 125 °C)	VP15-107-1
Pin part heating	Pin temperature : 300 °C max., Duration : 3 sec. max. (per device side)	—

Note The number of days the device can be stored after the dry pack was opened, under the storage conditions of 25°C and 65% RH max.

Caution Do not use two or more soldering methods in combination (except the pin part heating method).

12. DIFFERENCES BETWEEN μPD78C14A AND 78C14

Item	Product Name μPD78C14A	μPD78C14
I/O port PA, PB, PC	A pull-up resistor can be built in by the mask option in bit units.	A pull-up resistor is not built in.
Package	64-pin plastic QFP (14 × 14mm) : Pitch between pins, 0.8mm	64-pin plastic shrink DIP 64-pin plastic QUIP (Straight) 64-pin plastic QUIP 64-pin plastic QFP (14 × 20mm, 2.05mm thick) 64-pin plastic QFP (14 × 20mm, 2.70mm thick) 64-pin plastic QFJ

APPENDIX DEVELOPMENT TOOLS

The following development tools are available to develop a system which uses μPD78C14A series products.

Language Processor

87AD series relocatable assembler (RA87)	This is a program which converts a program written in mnemonic to an object code that micro-computer execution is possible. Besides, it contains a function to automatically create a symbol/table, and optimize a branch instruction.			
	Host Machine	OS	Supply Medium	Ordering Code (Product Name)
	PC-9800 series	MS-DOS™ Ver. 2.11 to Ver. 5.00A ^{Note}	3.5-inch 2HD	μS5A13RA87
			5-inch 2HD	μS5A10RA87
	IBM PC/AT™	PC DOS™ (Ver. 3.1)	3.5-inch 2HC	μS7B13RA87
5-inch 2HC			μS7B10RA87	

PROM Write Tools

Hardware	PG-1500	With an provided board and an optional programmer adapter connected, this PROM programmer can manipulate from a stand-alone or host machine to perform programming on single-chip microcomputer which incorporates PROM. It is also capable of programming a typical PROM ranging from 256K to 4M bits.			
	PA-78CP14CW/ GF/GQ/KB/L	PROM programmer adapter for μPD78CP14. Used by connecting to PG-1500.			
	PA-78CP14CW	For μPD78CP14CW, 78CP14DW			
	PA-78CP14GF	For μPD78CP14GF-3BE			
	PA-78CP14GQ	For μPD78CP14G-36, 78CP14R			
	PA-78CP14KB	For μPD78CP14KB			
	PA-78CP14L	For μPD78CP14L			
Software	PG-1500 controller	Connected PG-1500 to a host machine by using serial and parallel interface, to control the PG-1500 on a host machine.			
		Host Machine	OS	Supply Medium	Ordering Code (Product Name)
		PC-9800 series	MS-DOS Ver. 2.11 to Ver. 5.00A ^{Note}	3.5-inch 2HD	μS5A13PG1500
				5-inch 2HD	μS5A10PG1500
		IBM PC/AT	PC DOS (Ver. 3.1)	3.5-inch 2HD	μS7B13PG1500
5-inch 2HC	μS7B10PG1500				

Note Ver. 5.00/5.00A has a task swap function, but this function cannot be used with this software.

Remark Operation of assemblers and the PG-1500 controller are guaranteed only on the host machines and operating systems quoted above.

Debugging tools

An in-circuit emulator (IE-78C11-M) is available as a program debugging tool for μPD78C14A series. The following table shows its system configuration.

Hardware	IE-78C11-M	The IE-78C11-M is an in-circuit emulator which works with 87AD series. Only the IE-78C11-M should be used for a plastic QUIP package, while it should be used with a conversion socket for a plastic shrink DIP package. It can be connected to a host machine to perform efficient debugging.			
	EV-9001-64	Conversion sockets for plastic shrink DIP. Used in combination with the IE-78C11-M.			
	EV-9200G-64	64-pin WQFN socket. Can be used as a substitute for 64-pin plastic QFP products with window in combination with the μPD78CP14KB.			
Software	IE-78C11-M control program	Connects the IE-78C11-M to host machine by using the RS-232-C, then controls the IE-78C11-M on host machine.			
		Host Machine	OS	Supply Medium	Ordering Code (Product Name)
		PC-9800 series	MS-DOS Ver. 2.11 to Ver. 3.30D	3.5-inch 2HD	μS5A13IE78C11
				5-inch 2HD	μS5A10IE78C11
IBM PC/AT	PC DOS (Ver. 3.1)	5-inch 2HC	μS7B10IE78C11		

Remark Operation of the IE controller is guaranteed only on the host machine and operating systems quoted above.

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 device 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 this product from Japan is regulated by the Japanese government. To export this product may be prohibited without governmental license, the need for which must be judged by the customer. The export or re-export of this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.

NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.

While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customer must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.

NEC devices are classified into the following three quality grades:

"Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated "quality assurance program" for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device before using it in a particular application.

Standard: Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots

Special: Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support)

Specific: Aircrafts, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.

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

Anti-radioactive design is not implemented in this product.

M4 94.11

MS-DOS is a trademark of Microsoft Corporation.
PC/AT and PC DOS are trademarks of IBM Corporation.