Application Note

78K/0 Series

8-bit Single-chip Microcontroller

Basic (I)

μPD78002 subseries	**μPD78002Y subseries**
μPD78014 subseries	**μPD78014Y subseries**
μPD78018F subseries	**μPD78018FY subseries**
μPD780024 subseries	**μPD780024Y subseries**
μPD780034 subseries	**μPD780034Y subseries**
μPD78014H subseries	
μPD780924 subseries	**μPD780964 subseries**
μPD780001	

Document No. U12704EJ7V1AN00 (7th edition)
Date Published July 1998 N CP(K)

[MEMO]

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

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

The application circuits and their parameters are for reference only and are not intended for use in actual design-ins.

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

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, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.

NEC devices are classified into the following three quality grades:

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

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

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

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

Anti-radioactive design is not implemented in this product.

Regional Information

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

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

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

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

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

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

NEC Electronics Italiana s.r.1.
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

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

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

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 01-504-2787
Fax: 01-504-2860

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

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

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

NEC Electronics Singapore Pte. Ltd.
United Square, Singapore 1130
Tel: 65-253-8311
Fax: 65-250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.
Cumbica-Guarulhos-SP, Brasil
Tel: 011-6465-6810
Fax: 011-6465-6829

J98. 2

Major Revisions in This Edition

Page	Description
Throughout	Addition of following products as applicable products: μPD780024, 780024Y, 780034, 780034Y, 78014H, 780924, 780964 subseries, μPD78018F, 78018FY, 780001, 78011F(A), 78012F(A), 78013F(A), 78014F(A), 78015F(A), 78016F(A), 78018F(A), 78P018F(A)
 p.57, 58 p.59 to p.62 p.70, p71 p73 to p.75 p.81, 82 p.83, 84 p.145, 146 p.264 to p267 p.268, 269	Following register formats and tables are shown for each model. Tables 3-1 and 3-2 Maximum Time Required to Switch CPU Clock Figures 3-1 through 3-4 Format of Processor Clock Control Register Figures 4-1 and 4-2 Format of Timer clock Select Register 2 Figures 4-4 through 4-6 Format of Watchdog Timer Mode Register Figures 5-2 and 5-3 Format of 16-Bit Timer Mode Control Register Figures 5-4 and 5-5 Format of 16-Bit Timer Output Control Register Figures 7-2 and 7-3 Format of Watch Timer Mode Control Register Figures 9-1 through 9-4 Format of A/D Converter Mode Register Figures 9-5 and 9-6 Format of A/D Converter Input Select Register
 p.72 p.86 p.87 p.87, 133 p.125 p.126 p.127 p.128 p.129 p.130 p.131 p.269	Addition of following register formats: Figure 4-3 Format of Watchdog Timer Clock Select Register Figure 5-9 Format of Capture/Compare Control Register 0 Figure 5-10 Format of Prescaler Mode Register 0 Figures 5-11 and 5-14 Format of Port Mode Register 7 Figures 6-2 and 6-3 Format of Timer Clock Select Register 50 Figures 6-4 and 6-5 Format of Timer Clock Select Register 51 Figures 6-6 Format of Timer Clock Select Register 52 Figures 6-8 Format of 8-Bit Timer Mode Control Register 5n Figures 6-9 Format of 8-Bit Timer Mode Control Register 50 Figures 6-10 Format of 8-Bit Timer Mode Control Register 51 Figures 6-11 Format of 8-Bit Timer Mode Control Register 52 Figure 9-7 Format of Analog Input Channel Specification Register
p.73	Addition of Note 2 and Caution 2 to Figure 4-4 Format of Watchdog Timer Mode Register
p.85	Addition of Caution to Figure 5-7 Format of External Interrupt Mode Register
p.155	Addition of Table 8-2 Registers of Serial Interface
p.161 to p.166	Addition of Caution to Figures 8-6 through 8-8 Format of Serial Operating Mode Register 0, and Note to Control of Wake-up Function
p.182	Addition of Caution to Figure 8-19 Format of Automatic Transmission/Reception Interval Specification Register
p.185	Change of μPD6252 as maintenance part in 8.1 Interface with EEPROM^{TM} (μPD6252)
p.203	Addition of (5) and (6) Limits when I^{2}C bus mode is used to 8.1.2 Communication in I^{2}C bus mode
p.265	Addition of HSC bit to Figure 9-2 Format of A/D Converter Mode Register

The mark ★ shows major revised points.

INTRODUCTION

Readers

This Application Note is intended for use by engineers who understand the functions of the 78K/0 series and wish to design application programs with the following subseries products:

- **Subseries**

μPD78002 subseries	:	μPD78001B, 78002B, 78001B(A), 78002B(A)
μPD78002Y subseries	:	μPD78001BY, 78002BY
μPD78014 subseries	:	μPD78011B, 78012B, 78013, 78014, 78P014, 78011B(A), 78012B(A), 78013(A), 78014(A)
μPD78014Y subseries	:	μPD78011BY, 78012BY, 78013Y, 78014Y, 78P014Y
μPD78018F subseries	:	μPD78011F, 78012F, 78013F, 78014F, 78015F, 78016F, 78018F, 78P018F, 78011F(A), 78012F(A), 78013F(A), 78014F(A), 78015F(A), 78016F(A), 78018F(A), 78P018F(A), 78012F(A2)
μPD78018Y subseries	:	μPD78011FY, 78012FY, 78013FY, 78014FY, 78015FY, 78016FY, 78018FY, 78P018FY
μPD780001		
μPD78014H subseries	:	μPD78011H, 78012H, 78013H, 78014H, 78011H(A), 78012H(A), 78013H(A), 78014H(A)
μPD780024 subseries	:	μPD780021[Note], 780022[Note], 780023[Note], 780024[Note]
μPD780024Y subseries	:	μPD780021Y[Note], 780022Y[Note], 780023Y[Note], 780024Y[Note]
μPD780034 subseries	:	μPD780031[Note], 780032[Note], 780033[Note], 780034[Note], 78F0034[Note]
μPD780034Y subseries	:	μPD780031Y[Note], 780032Y[Note], 780033Y[Note], 780034Y[Note], 78F0034Y[Note]
μPD780924 subseries	:	μPD780921[Note], 780922[Note], 780923[Note], 780924[Note], 78F0924[Note]
μPD780964 subseries	:	μPD780961[Note], 780962[Note], 780963[Note], 78F0964[Note]

Note Under development

Remarks

1. The μPD78001B(A), and 78002B(A) have higher reliability than the μPD78001B and 78002B.
2. The μPD78011B(A), 78012B(A), 78013(A) and 78014(A) have higher reliability than the μPD78011B, 78012B, 78013 and 78014.
3. The μPD78011F(A), 78012F(A), 78013F(A), 78014F(A), 78015F(A), 78016F(A), 78018F(A), and 78P018F(A) have higher reliability than the μPD78011F, 78012F, 78013F, 78014F, 78015F, 78016F, 78018F, and 78P018F.
4. The μPD78012F(A2) has higher reliability than the μPD78012F.
5. The μPD78011H(A), 78012H(A), 78013H(A), and 78014H(A) have higher reliability than the μPD78011H, 78012H, 78013H, and 78014H.

Purpose

This Application Note is to deepen your understanding of the basic functions of the 78K/0 series by using program examples.

Note that the programs and hardware configuration shown in this document are only examples and not subject to mass production.

Organization This Application Note consists of the following contents:

- General
- Software
- Hardware

★ In addition to this Application Note, the following Application Notes are also available:

Document Name	Document Number		Targeted Subseries	Contents
	Japanese	English		
78K/0 Series Application Note Basic (I)	U12704J	This document	μPD78002, 78002Y μPD78014, 78014Y μPD78018F, 78018FY μPD780001 μPD780024, 780024Y μPD780034, 780034Y μPD78014H μPD780924 μPD780964	Explains basic functions of products in 78K/0 series by using program examples
78K/0 Series Application Note Basic (II)	U10121J	U10121E	μPD78044 μPD78044H μPD780208 μPD780228	
78K/0 Series Application Note Basic (III)	U10182J	U10182E	μPD78054, 78054Y μPD78064, 78064Y μPD78078, 78078Y μPD78083 μPD78098 μPD780018AY μPD780058, 780058Y μPD780308, 780308Y μPD78058F, 78058FY μPD78064B μPD78070A, 78070AY μPD78075B, 78075BY μPD78098B	
78K/0 Series Application Note Floating-Point Operation Program	IEA-718	IEA-1289	All subseries in 78K/0 series (except μPD78002 and 78002Y subseries)	Explains floating-point operation programs of products in 78K/0 series
μPD78014 Series Application Note Electronic Pocketbook	IEA-744	IEA-1301	μPD78014 (only μPD78014 and 78P014)	Explains how to organize electronic pocketbook by using μPD78014 subseries

Caution The application examples and program lists shown in this Application Note assume that the main system clock operates at 8.38 MHz, not at 10.0 MHz.

How to Read This Manual Although this Application Note explains the functions of the 78K/0 series products, the functions of some products in each subseries differ from those of the others.

Subseries Chapter	μPD78002 μPD78002Y	μPD78014 μPD78014Y	μPD78018F μPD78018FY	μPD780001	μPD780024 μPD780024Y	μPD780034 μPD780034Y	μPD78014H	μPD780924 μPD780964
CHAPTER 1 GENERAL	○	○	○	○	○	○	○	○
CHAPTER 2 BASICS OF SOFTWARE	○	○	○	○	○	○	○	○
CHAPTER 3 APPLICATIONS OF SYSTEM CLOCK SELECTION	○	○	○	○	○	○	○	○
CHAPTER 4 APPLICATIONS OF WATCHDOG TIMER	○	○	○	○	○	○	○	○
CHAPTER 5 APPLICATIONS OF 16-BIT TIMER/EVENT COUNTER	–	○	○	–	○	○	○	–
CHAPTER 6 APPLICATIONS OF 8-BIT TIMER/EVENT COUNTER	○	○	○	○	○	○	○	○
CHAPTER 7 APPLICATIONS OF WATCH TIMER	○	○	○	–	○	○	○	–
CHAPTER 8 APPLICATIONS OF SERIAL INTERFACE	○	○	○	○	–	–	○	–
CHAPTER 9 APPLICATIONS OF A/D CONVERTER	–	○	○	○	○	–	○	○
CHAPTER 10 APPLICATIONS OF KEY INPUT	○	○	○	○	–	–	○	–

The (A)-model and standard models differ only in quality grade.

The μPD78012F(A2) differs from standard models and (A)-models in terms of operating temperature range, DC characteristics, and AC characteristics. For details, refer to the individual Data Sheet.

In this document, read (A)-models and (A2)-model as follows:

μPD78001B → μPD78001B(A)	μPD78002B → μPD78002B(A)
μPD78011B → μPD78011B(A)	μPD78012B → μPD78012B(A)
μPD78013 → μPD78013(A)	μPD78014 → μPD78014(A)
μPD78011F → μPD78011F(A)	μPD78012F → μPD78012F(A)
μPD78013F → μPD78013F(A)	μPD78014F → μPD78014F(A)
μPD78015F → μPD78015F(A)	μPD78016F → μPD78016F(A)
μPD78018F → μPD78018F(A)	μPD78P018F → μPD78P018F(A)
μPD78011H → μPD78011H(A)	μPD78012H → μPD78012H(A)
μPD78013H → μPD78013H(A)	μPD78014H → μPD78014H(A)
μPD78012F → μPD78012F(A2)	

Legend

Data significance	:	Left: higher digit, right: lower digit
Low active	:	$\overline{\times\times\times}$ (top bar over pin or signal name)
Note	:	Description of **Note** in the text
Caution	:	Important information
Remark	:	Supplement
Numeric representation	:	Binary ... ×××× or ××××B
		Decimal ... ××××
		Hexadecimal ... ××××H

Quality Grade

- **Standard**

  μPD78001B, 78002B
  μPD78001BY, 78002BY
  μPD78011B, 78012B, 78013, 78014, 78P014
  μPD78011BY, 78012BY, 78013Y, 78014Y, 78P014Y
  μPD78011F, 78012F, 78013F, 78014F, 78015F, 78016F, 78P018F
  μPD78011FY, 78012FY, 78013FY, 78014FY, 78015FY, 78016FY, 78P018FY
  μPD78001
  μPD780021, 780022, 780023, 780024
  μPD780021Y, 780022Y, 780023Y, 780024Y
  μPD780031, 780032, 780033, 780034, 78F0034
  μPD780031Y, 780032Y, 780033Y, 780034Y, 78F0034Y
  μPD78011H, 78012H, 78013H, 78014H
  μPD780921, 780922, 780923, 780924, 78F0924
  μPD780961, 780962, 780963, 780964, 78F0964

- **Special**

  μPD78001B(A), 78002B(A)
  μPD78011B(A), 78012B(A), 78013(A), 78014(A)
  μPD78011F(A), 78012F(A), 78013F(A), 78014F(A), 78015F(A), 78016F(A), 78018F(A), 78P018F(A), 78012F(A2)
  μPD78011H(A), 78012H(A), 78013H(A), 78014H(A)

Please refer to "Quality Grades on NEC Semiconductor Devices" (Document No. C11531E) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.

Application Field

- Consumer appliances

Related documents

Some of the related documents listed below are preliminary versions but not so specified here.

- **Common related documents**

Document Name	Document Number	
	Japanese	English
78K/0 Series Application Note - Basic (I)	U12704J	This document
78K/0 Series User's Manual - Instruction	U12326J	U12326E
78K/0 Series Instruction Set	U10904J	–
78K/0 Series Instruction Table	U10903J	–

- **Documents dedicated to product**

- **μPD78002, 78002Y subseries**

Document Name	Document Number	
	Japanese	English
μPD78002, 78002Y Series User's Manual	U10039J	U10039E
μPD78001B, 78002B Data Sheet	U10674J	U10674E
μPD78001B(A), 78002B(A) Data Sheet	IC-9078	IC-3599
μPD78001BY, 78002BY Data Sheet	IC-8571	IC-3173
μPD78002, 78002Y Series Special Function Register Table	IEM-5547	–

- **μPD78014, 78014Y subseries**

Document Name	Document Number	
	Japanese	English
μPD78014, 78014Y Series User's Manual	U10085J	U10085E
μPD78011B, 78012B, 78013, 78014 Data Sheet	IC-8201	IC-3179
μPD78011B(A), 78012B(A), 78013(A), 78014(A) Data Sheet	IC-8874	IC-3411
μPD78011BY, 78012BY, 78013Y, 78014Y Data Sheet	IC-8573	IC-3405
μPD78P014 Data Sheet	IC-8111	IC-3098
μPD78P014Y Data Sheet	IC-8572	IC-3180
μPD78014, 78014Y Series Special Function Register Table	IEM-5527	–

- **μPD78018F, 78018FY subseries**

Document Name	Document Number	
	Japanese	English
μPD78018F, 78018FY Subseries User's Manual	U10659J	U10659E
μPD78011F, 78012F, 78013F, 78014F, 78015F, 78016F Data Sheet	U10280J	U10280E
μPD78011F(A), 78012F(A), 78013F(A), 78014F(A), 78015F(A), 78016F(A) 78018F(A) Data Sheet	U11921J	U11921E
μPD78011FY, 78012FY, 78013FY, 78014FY, 78015FY, 78016FY Data Sheet	U10281J	U10281E
μPD78P018F Data Sheet	U10955J	U10955E
μPD78P018F(A) Data Sheet	U12132J	U12132E
μPD78P018FY Data Sheet	U10989J	U10989E
μPD78018F Subseries Special Function Register Table	IEM-5594	–
μPD78018FY Subseries Special Function Register Table	U10287J	–

★ - **μPD780001 subseries**

Document Name	Document Number	
	Japanese	English
μPD780001 User's Manual	U10885J	U10885E
μPD780001 Data Sheet	U10324J	U10324E

★ - **μPD780024, 780024Y, 780034, 780034Y subseries**

Document Name	Document Number	
	Japanese	English
μPD780021, 780022, 780023, 780024 Preliminary Product Information	U12299J	U12299E
μPD780031, 780032, 780033, 780034 Preliminary Product Information	U12300J	U12300E
μPD78F0034 Preliminary Product Information	U11993J	U11993E
μPD780021Y, 780022Y, 780023Y, 780024Y Preliminary Product Information	U12165J	U12165E
μPD780031Y, 780032Y, 780033Y, 780034Y Preliminary Product Information	U12166J	U12166E
μPD78F0034Y Preliminary Product Information	U11994J	U11994E
μPD780024, 780034, 780024Y, 780034Y Subseries User's Manual	U12022J	U12022E

★ - **μPD78014H subseries**

Document Name	Document Number	
	Japanese	English
μPD78014H Subseries User's Manual	U12220J	U12220E
μPD78011H, 78012H, 78013H, 78014H Data Sheet	U11898J	U11898E
μPD78011H(A), 78012H(A), 78013H(A), 78014H(A) Data Sheet	U12174J	U12174E

★ • **μPD780924, 780964 subseries**

Document Name	Document Number	
	Japanese	English
μPD780921, 780922, 780923, 780924 Preliminary Product Information	U11804J	U11804E
μPD78F0924 Preliminary Product Information	U11930J	U11930E
μPD780961, 780962, 780963, 780964 Preliminary Product Information	U11879J	U11879E
μPD78F0964 Preliminary Product Information	U11956J	U11956E
μPD780924, 780964 Subseries User's Manual	U12071J	U12071E
μPD780924, 780964 Subseries Special Funciton Register Table	U12230J	–

The contents of the above related documents are subject to change without notice. Be sure to use the latest edition when you design your system.

[MEMO]

CONTENTS

LIST OF FIGURES (1/4)

Fig. No. Title Page

LIST OF FIGURES (2/4)

LIST OF FIGURES (3/4)

LIST OF FIGURES (4/4)

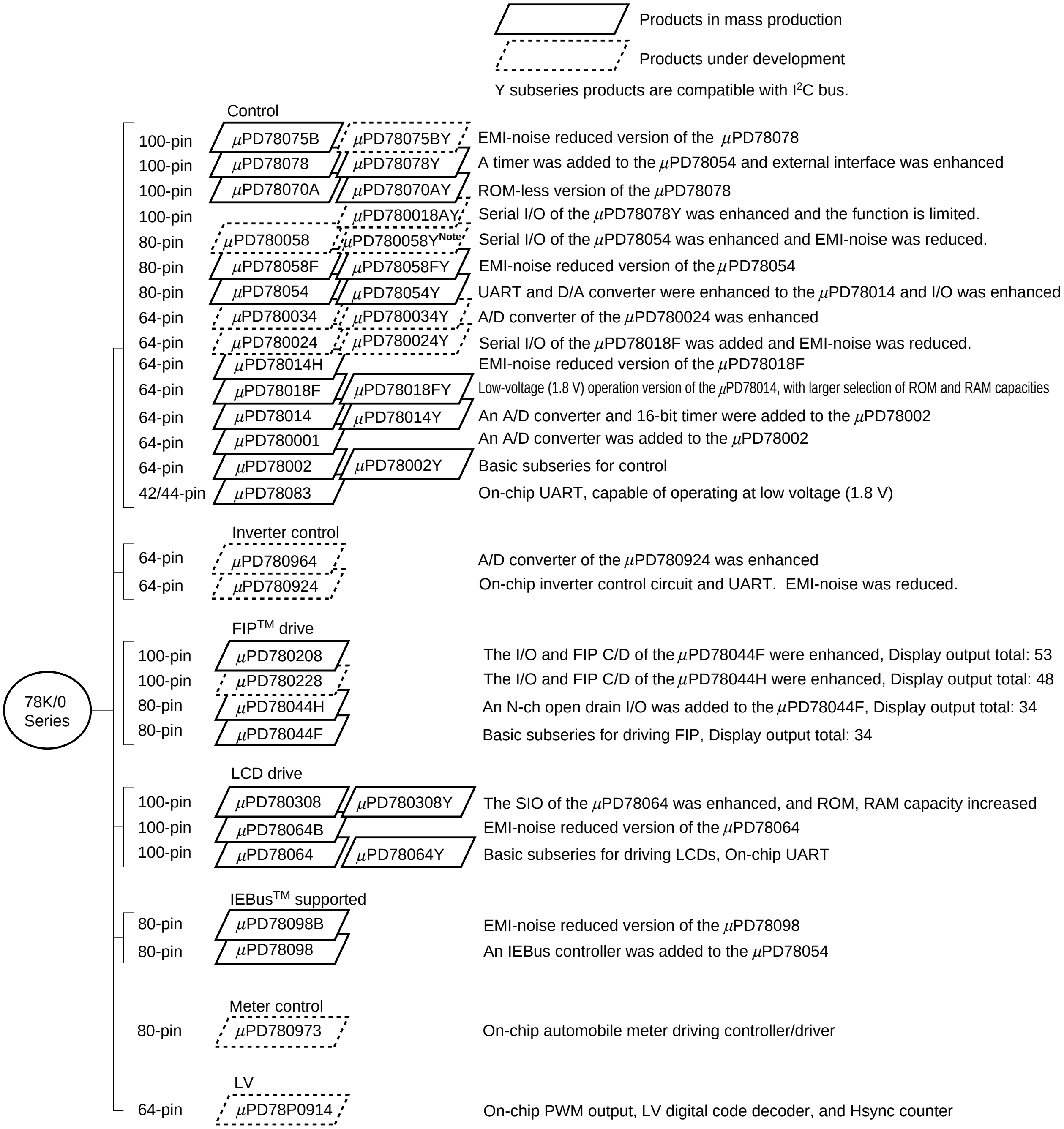
LIST OF TABLES

[MEMO]

CHAPTER 1 GENERAL

★ 1.1 Product Development of 78K/0 Series

The following shows the products organized according to usage. The names in the parallelograms are subseries names.



Y subseries products are compatible with I^{2}C bus.

Control			
100-pin	μPD78075B	μPD78075BY	EMI-noise reduced version of the μPD78078
100-pin	μPD78078	μPD78078Y	A timer was added to the μPD78054 and external interface was enhanced
100-pin	μPD78070A	μPD78070AY	ROM-less version of the μPD78078
100-pin		μPD780018AY	Serial I/O of the μPD78078Y was enhanced and the function is limited.
80-pin	μPD780058	μPD780058Y^Note	Serial I/O of the μPD78054 was enhanced and EMI-noise was reduced.
80-pin	μPD78058F	μPD78058FY	EMI-noise reduced version of the μPD78054
80-pin	μPD78054	μPD78054Y	UART and D/A converter were enhanced to the μPD78014 and I/O was enhanced
64-pin	μPD780034	μPD780034Y	A/D converter of the μPD780024 was enhanced
64-pin	μPD780024	μPD780024Y	Serial I/O of the μPD78018F was added and EMI-noise was reduced.
64-pin	μPD78014H		EMI-noise reduced version of the μPD78018F
64-pin	μPD78018F	μPD78018FY	Low-voltage (1.8 V) operation version of the μPD78014, with larger selection of ROM and RAM capacities
64-pin	μPD78014	μPD78014Y	An A/D converter and 16-bit timer were added to the μPD78002
64-pin	μPD780001		An A/D converter was added to the μPD78002
64-pin	μPD78002	μPD78002Y	Basic subseries for control
42/44-pin	μPD78083		On-chip UART, capable of operating at low voltage (1.8 V)
Inverter control			
64-pin	μPD780964		A/D converter of the μPD780924 was enhanced
64-pin	μPD780924		On-chip inverter control circuit and UART. EMI-noise was reduced.
FIP™ drive			
100-pin	μPD780208		The I/O and FIP C/D of the μPD78044F were enhanced, Display output total: 53
100-pin	μPD780228		The I/O and FIP C/D of the μPD78044H were enhanced, Display output total: 48
80-pin	μPD78044H		An N-ch open drain I/O was added to the μPD78044F, Display output total: 34
80-pin	μPD78044F		Basic subseries for driving FIP, Display output total: 34
LCD drive			
100-pin	μPD780308	μPD780308Y	The SIO of the μPD78064 was enhanced, and ROM, RAM capacity increased
100-pin	μPD78064B		EMI-noise reduced version of the μPD78064
100-pin	μPD78064	μPD78064Y	Basic subseries for driving LCDs, On-chip UART
IEBus™ supported			
80-pin	μPD78098B		EMI-noise reduced version of the μPD78098
80-pin	μPD78098		An IEBus controller was added to the μPD78054
Meter control			
80-pin	μPD780973		On-chip automobile meter driving controller/driver
LV			
64-pin	μPD78P0914		On-chip PWM output, LV digital code decoder, and Hsync counter

Note Under planning

The following lists the main functional differences between subseries products.

Subseries Name	Function	ROM Capacity	Timer				8-bit A/D	10-bit A/D	8-bit D/A	Serial Interface	I/O	V_{DD} MIN. Value	External Expansion
			8-bit	16-bit	Watch	WDT							
Control	μPD78075B	32K-40K	4ch	1ch	1ch	1ch	8ch	–	2ch	3ch (UART: 1ch)	88	1.8 V	○
	μPD78078	48K-60K											
	μPD78070A	–									61	2.7 V	
	μPD780058	24K-60K	2ch						2ch	3ch (time division UART: 1ch)	68	1.8 V	
	μPD78058F	48K-60K								3ch (UART: 1ch)	69	2.7 V	
	μPD78054	16K-60K										2.0 V	
	μPD780034	8K-32K					–	8ch	–	3ch (UART: 1ch, time division 3-wire: 1ch)	51	1.8 V	
	μPD780024						8ch	–					
	μPD78014H									2ch	53	1.8 V	
	μPD78018F	8K-60K											
	μPD78014	8K-32K										2.7 V	
	μPD780001	8K		–	–					1ch	39		–
	μPD78002	8K-16K			1ch		–				53		○
	μPD78083				–		8ch			1ch (UART: 1ch)	33	1.8 V	–
Inverter control	μPD780964	8K-32K	3ch	**Note**	–	1ch	–	8ch	–	2ch (UART: 2ch)	47	2.7 V	○
	μPD780924						8ch	–					
FIP drive	μPD780208	32K-60K	2ch	1ch	1ch	1ch	8ch	–	–	2ch	74	2.7 V	–
	μPD780228	48K-60K	3ch	–	–					1ch	72	4.5 V	
	μPD78044H	32K-48K	2ch	1ch	1ch						68	2.7 V	
	μPD78044F	16K-40K								2ch			
LCD drive	μPD780308	48K-60K	2ch	1ch	1ch	1ch	8ch	–	–	3ch (time division UART: 1ch)	57	2.0 V	–
	μPD78064B	32K								2ch (UART: 1ch)			
	μPD78064	16K-32K											
IEBus supported	μPD78098	40K-60K	2ch	1ch	1ch	1ch	8ch	–	2ch	3ch (UART: 1ch)	69	2.7 V	○
	μPD78098B	32K-60K											
Meter control	μPD780973	24K-32K	3ch	1ch	1ch	1ch	5ch	–	–	2ch (UART: 1ch)	56	4.5 V	–
LV	μPD78P0914	32K	6ch	–	–	1ch	8ch	–	–	2ch	54	4.5 V	○

Note 10-bit timer: 1 channel

1.2 Features of 78K/0 Series

The 78K/0 series is a collection of 8-bit single-chip microcontrollers ideal for consumer applications.

The μPD78002 and 78002Y subseries are microcontrollers with many hardware peripherals such as ROM, RAM, I/O Ports, timers, serial interface, and interrupt control functions, as well as a high-speed, high-performance CPU.

The μPD78014 and 78014Y subseries are models with an A/D converter and a reinforced timer and serial interface in addition to the above hardware peripherals.

The μPD78018F and 78018FY subseries are models that can operate at a voltage lower than the μPD78014 and 78014Y subseries.

★ The μPD780001 is based on the μPD78002 subseries model and is provided with an A/D converter.

★ The μPD780024 and 780024Y subseries are low-EMI noise versions of the μPD78018F and 78018FY subseries with a reinforced serial I/O interface.

★ The μPD780034 and 780034Y subseries are low-EMI noise versions of the μPD780024 and 780024Y subseries with a reinforced A/D converter.

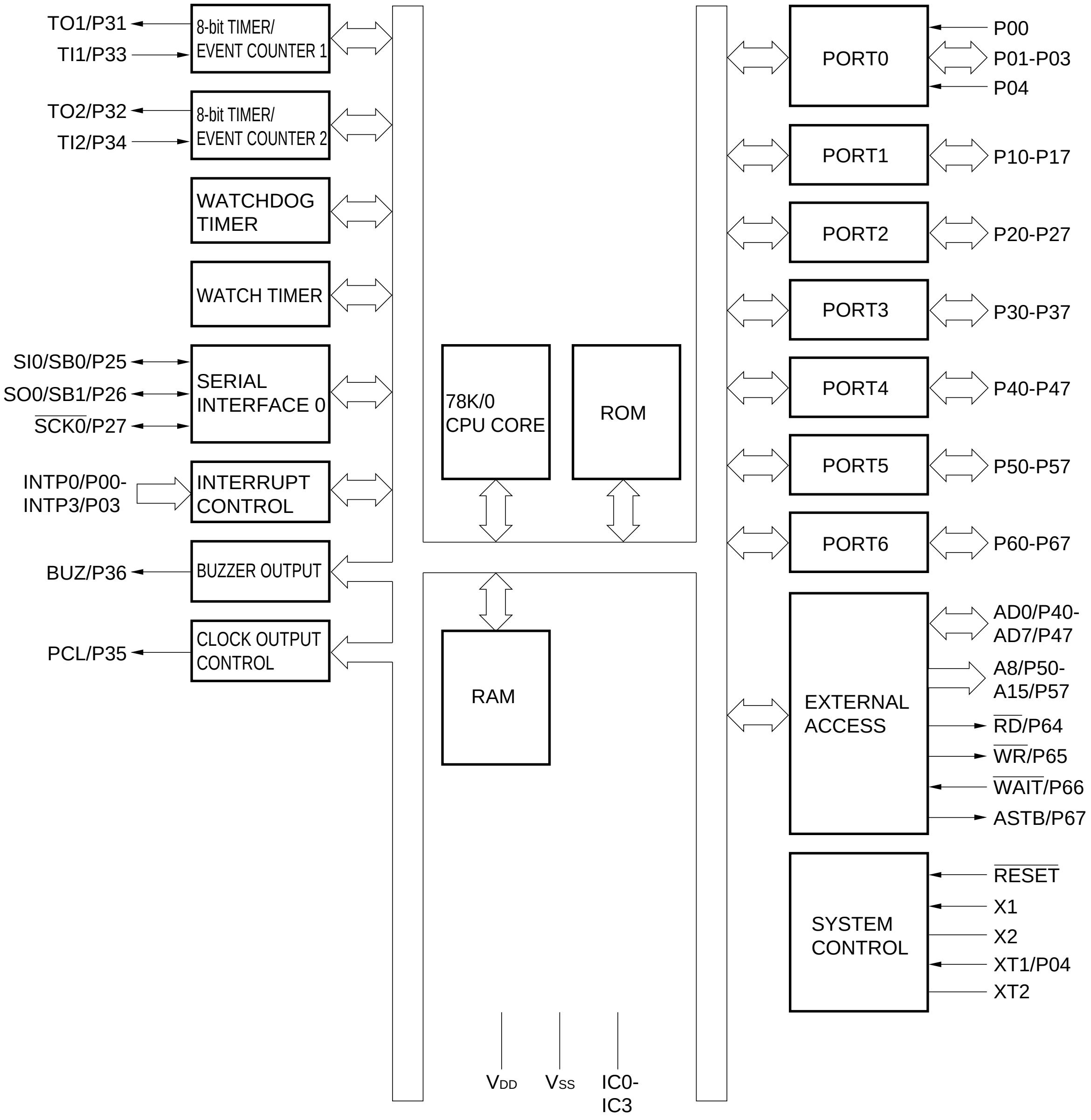
★ The μPD78014H subseries is a low-EMI noise version of the μPD78018F subseries.

★ The μPD780924 and 780964 subseries are provided with an inverter control circuit. The μPD780924 subseries is a low-EMI noise model. The μPD780964 subseries is a version of the μPD780924 subseries with a reinforced A/D converter. The μPD78002Y, 78014Y, 780024Y, and 780034Y subseries add an I^{2}C bus control function to the μPD78002, 78014, 780024, and 780034 subseries. The μPD78018FY subseries is a version of the μPD78018F subseries with an I^{2}C bus control function in the place of the SBI function.

In addition, one-time PROM, EPROM, or flash-memory models μPD78P014, 78P014Y (V_{DD} = 2.7 to 6.0 V), μPD78P018F, 78P018FY (V_{DD} = 1.8 to 5.5 V), μPD78F0034, 78F0034Y (V_{DD} = 1.8 to 5.5 V), μPD78F0924, 78F0964 (V_{DD} = 2.7 to 5.5 V) that can operate at the same operating voltage as the mask ROM models and that are ideal for early and small-scale production of the application system are also available.

The block diagram and function outline of each series is shown on the following pages.

Figure 1-1. Block Diagram of μPD78002 Subseries

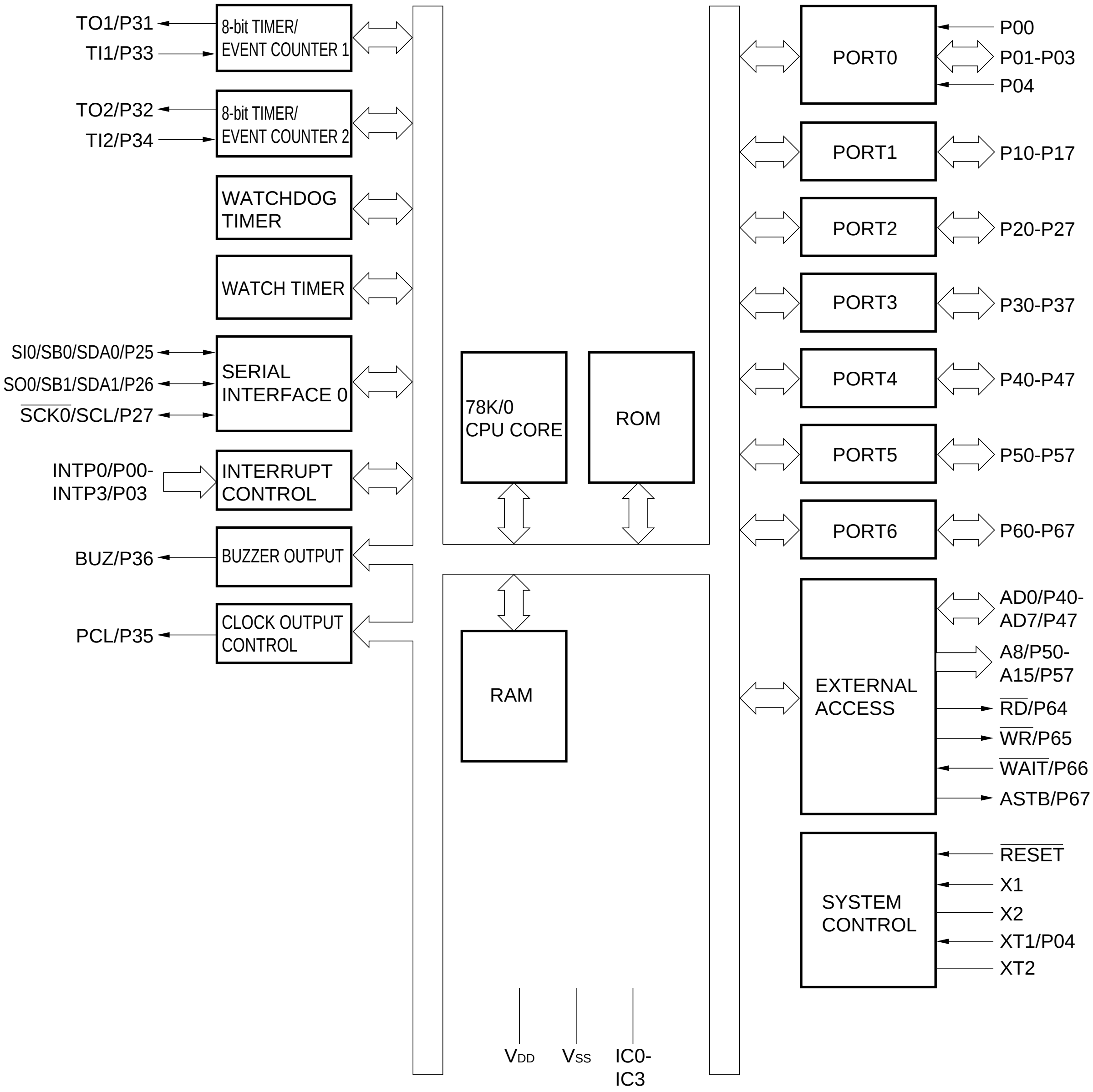


Remark The internal ROM and RAM capacities differ depending on the model.

Table 1-1. Functional Outline of µPD78002 Subseries

Part Number Item		µPD78001B	µPD78002B
Internal memory	ROM	Mask ROM	
		8K bytes	16K bytes
	High-speed RAM	256 bytes	384 bytes
Memory space		64K bytes	
General-purpose register		8 bits × 8 × 4 banks	
Minimum instruction execution time	With main system clock	0.4 µs/0.8 µs/1.6 µs/3.2 µs/6.4 µs (at 10.0 MHz)	
	With subsystem clock	122 µs (at 32.768 kHz)	
Instruction set		• 16-bit operation • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.	
I/O port		• Total : 53 • CMOS input : 2 • CMOS I/O : 47 (On-chip pull-up resistor ON/OFF selected by software : 47) • N-ch open drain I/O : 4 • (15-V withstand, pull-up resistor connected by mask option : 4)	
Serial interface		• 3-wire serial I/O/SBI/2-wire serial I/O mode selectable : 1 channel	
Timer		• 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel	
Timer output		3 (14-bit PWM output: 1)	
Clock output		39.1 kHz, 78.1 kHz, 156 kHz, 313 kHz, 625 kHz, 1.25 MHz, (with main system clock of 10.0 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)	
Buzzer output		2.4 kHz, 4.9 kHz, 9.8 kHz (with main system clock of 10.0 MHz)	
Vectored interrupt source	Maskable	Internal: 5, external: 4	
	Non-maskable	Internal: 1	
	Software	1	
Test input		Internal: 1, external: 1	
Supply voltage		V_{DD} = 2.7 to 6.0 V	
Operating temperature		T_A = –40 to +85 °C	
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm)	

Figure 1-2. Block Diagram of μPD78002Y Subseries

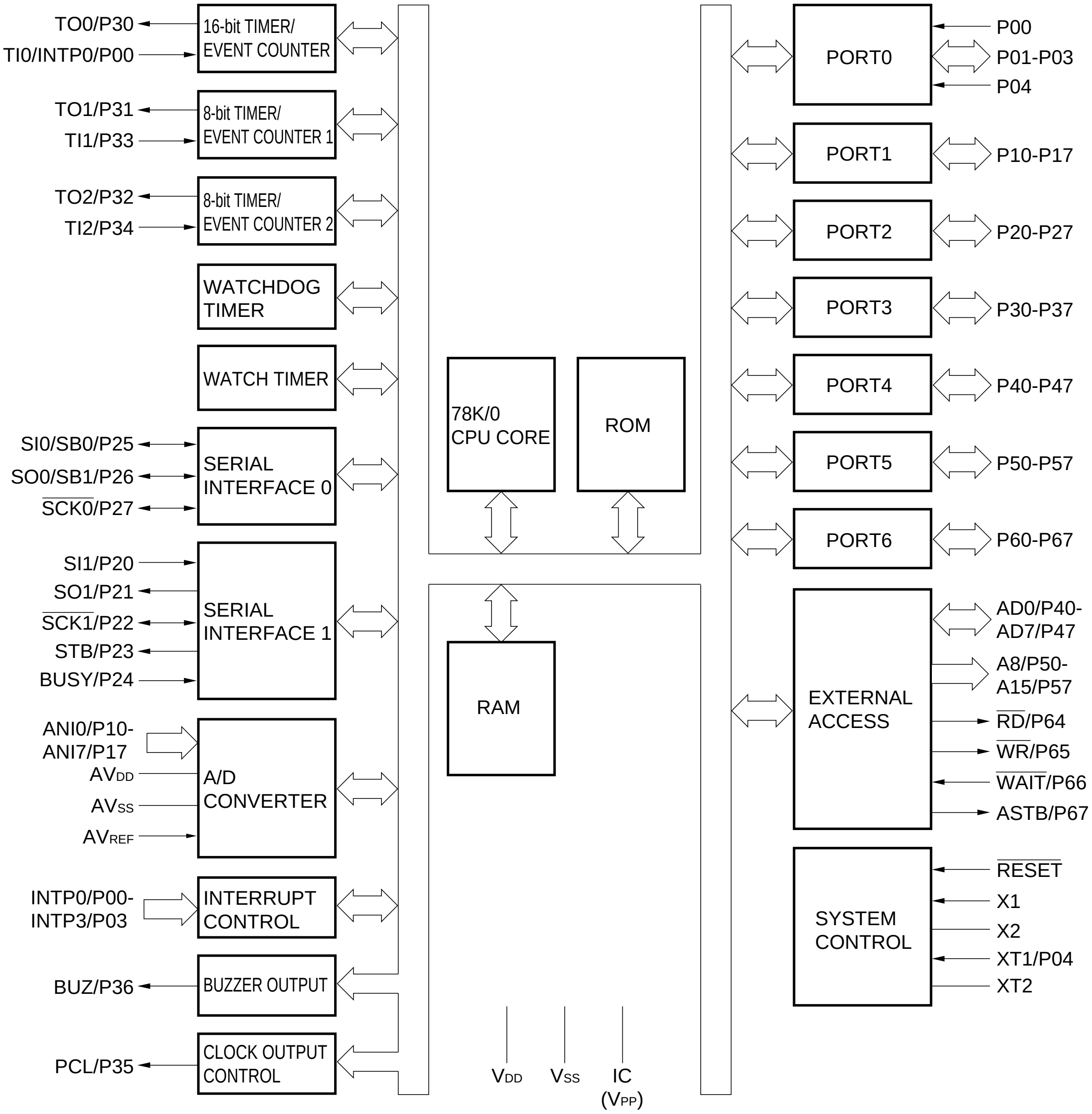


Remark The internal ROM and RAM capacities differ depending on the model.

Table 1-2. Functional Outline of µPD78002Y Subseries

Item \ Part Number		µPD78001BY	µPD78002BY
Internal memory	ROM	Mask ROM	
		8K bytes	16K bytes
	High-speed RAM	256 bytes	384 bytes
Memory space		64K bytes	
General-purpose register		8 bits × 8 × 4 banks	
Minimum instruction execution time	With main system clock	0.4 µs/0.8 µs/1.6 µs/3.2 µs/6.4 µs (at 10.0 MHz)	
	With subsystem clock	122 µs (at 32.768 kHz)	
Instruction set		• 16-bit operation • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.	
I/O port		• Total : 53 • CMOS input : 2 • CMOS I/O : 47 (On-chip pull-up resistor ON/OFF selected by software : 47) • N-ch open drain I/O : 4 • (15-V withstand, pull-up resistor connected by mask option : 4)	
Serial interface		• 3-wire serial I/O/SBI/2-wire serial I/O/I^{2}C bus mode selectable : 1 channel	
Timer		• 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel	
Timer output		3 (14-bit PWM output: 1)	
Clock output		39.1 kHz, 78.1 kHz, 156 kHz, 313 kHz, 625 kHz, 1.25 MHz (with main system clock of 10.0 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)	
Buzzer output		2.4 kHz, 4.9 kHz, 9.8 kHz (with main system clock of 10.0 MHz)	
Vectored interrupt source	Maskable	Internal: 5, external: 4	
	Non-maskable	Internal: 1	
	Software	1	
Test input		Internal: 1, external: 1	
Supply voltage		V_{DD} = 2.7 to 6.0 V	
Operating temperature		T_A = –40 to +85 °C	
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm)	

Figure 1-3. Block Diagram of μPD78014 Subseries



Remarks 1. The internal ROM and RAM capacities differ depending on the model.

2. ( ): μPD78P014

Table 1-3. Functional Outline of µPD78014 Subseries (1/2)

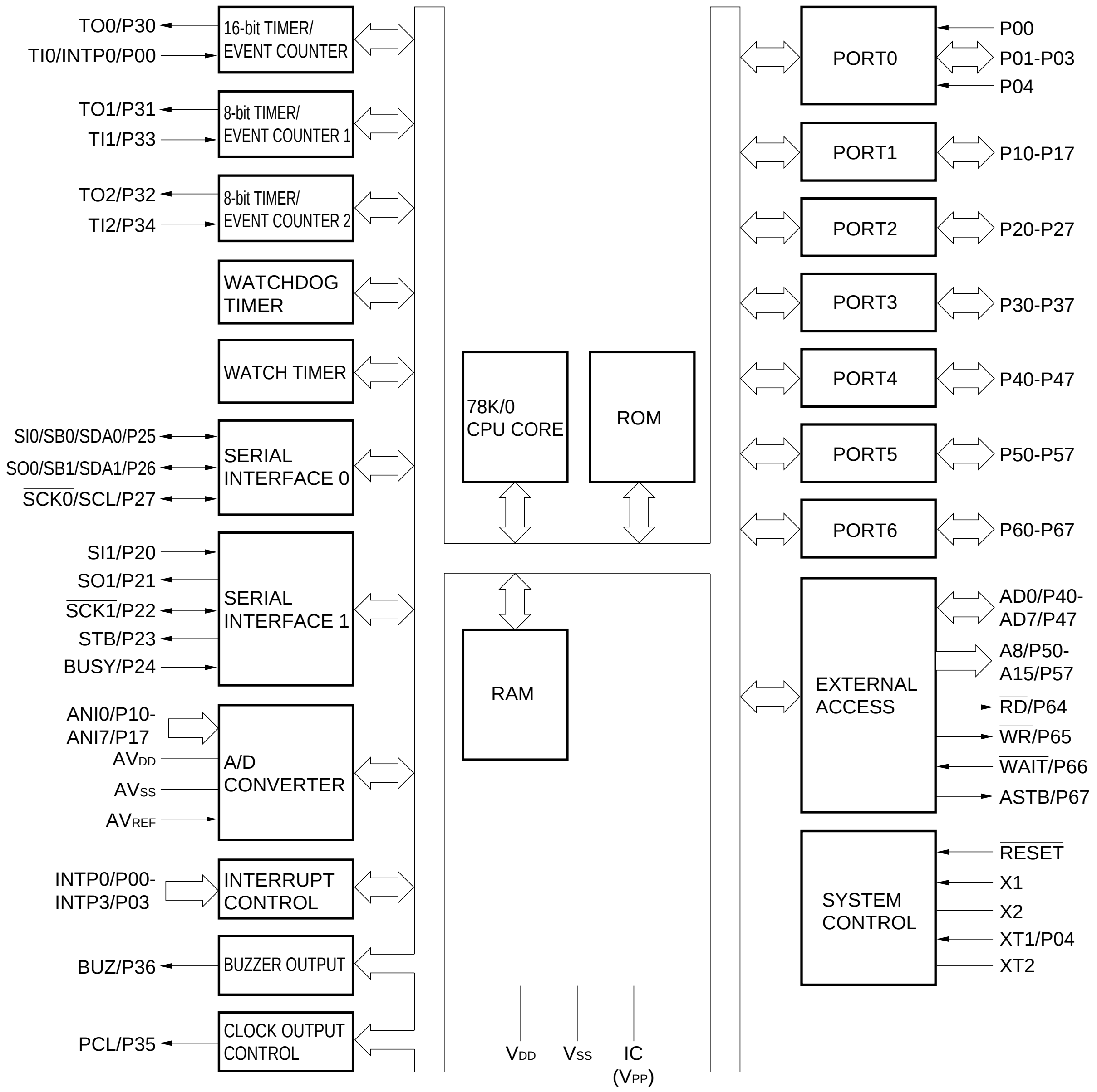
Item \ Part Number		µPD78011B	µPD78012B	µPD78013	µPD78014	µPD78P014
Internal memory	ROM	Mask ROM				PROM
		8K bytes	16K bytes	24K bytes	32K bytes	32K bytes$^{\text{Note}}$
	High-speed RAM	512 bytes		1024 bytes		1024 bytes$^{\text{Note}}$
	Buffer RAM	32 bytes				
Memory space		64K bytes				
General-purpose register		8 bits × 8 × 4 banks				
Minimum instruction execution time	With main system clock	0.4 µs/0.8 µs/1.6 µs/3.2 µs/6.4 µs (at 10.0 MHz)				
	With subsystem clock	122 µs (at 32.768 kHz)				
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.				
I/O port		• Total : 53 • CMOS input : 2 • CMOS I/O : 47 (On-chip pull-up resistor ON/OFF selected by software : 47) • N-ch open drain I/O : 4 (15-V withstand, pull-up resistor connected by mask option : 4)				
A/D converter		• 8-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 2.7 to 6.0 V				
Serial interface		• 3-wire serial I/O/SBI/2-wire serial I/O mode selectable : 1 channel • 3-wire serial I/O mode (with function to automatically transfer/receive up to 32 bytes) : 1 channel				
Timer		• 16-bit timer/event counter : 1 channel • 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel				
Timer output		3 (14-bit PWM output: 1)				
Clock output		39.1 kHz, 78.1 kHz, 156 kHz, 313 kHz, 625 kHz, 1.25 MHz (with main system clock of 10.0 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)				
Buzzer output		2.4 kHz, 4.9 kHz, 9.8 kHz (with main system clock of 10.0 MHz)				

Note The internal PROM and internal high-speed RAM capacities can be changed by using a memory size select register (IMS).

Table 1-3. Functional Outline of μPD78014 Subseries (2/2)

Part Number Item		μPD78011B	μPD78012B	μPD78013	μPD78014	μPD78P014
Vectored interrupt source	Maskable	Internal: 8, external: 4				
	Non-maskable	Internal: 1				
	Software	1				
Test input		Internal: 1, external: 1				
Supply voltage		V_{DD} = 2.7 to 6.0 V				
Operating temperature		T_A = –40 to +85 °C				
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm) • 64-pin ceramic shrink DIP (750 mil) (μPD78P014 only)				

Figure 1-4. Block Diagram of μPD78014Y Subseries



Remarks 1. The internal ROM and RAM capacities differ depending on the model.

2. ( ): μPD78P014Y

Table 1-4. Functional Outline of μPD78014Y Subseries (1/2)

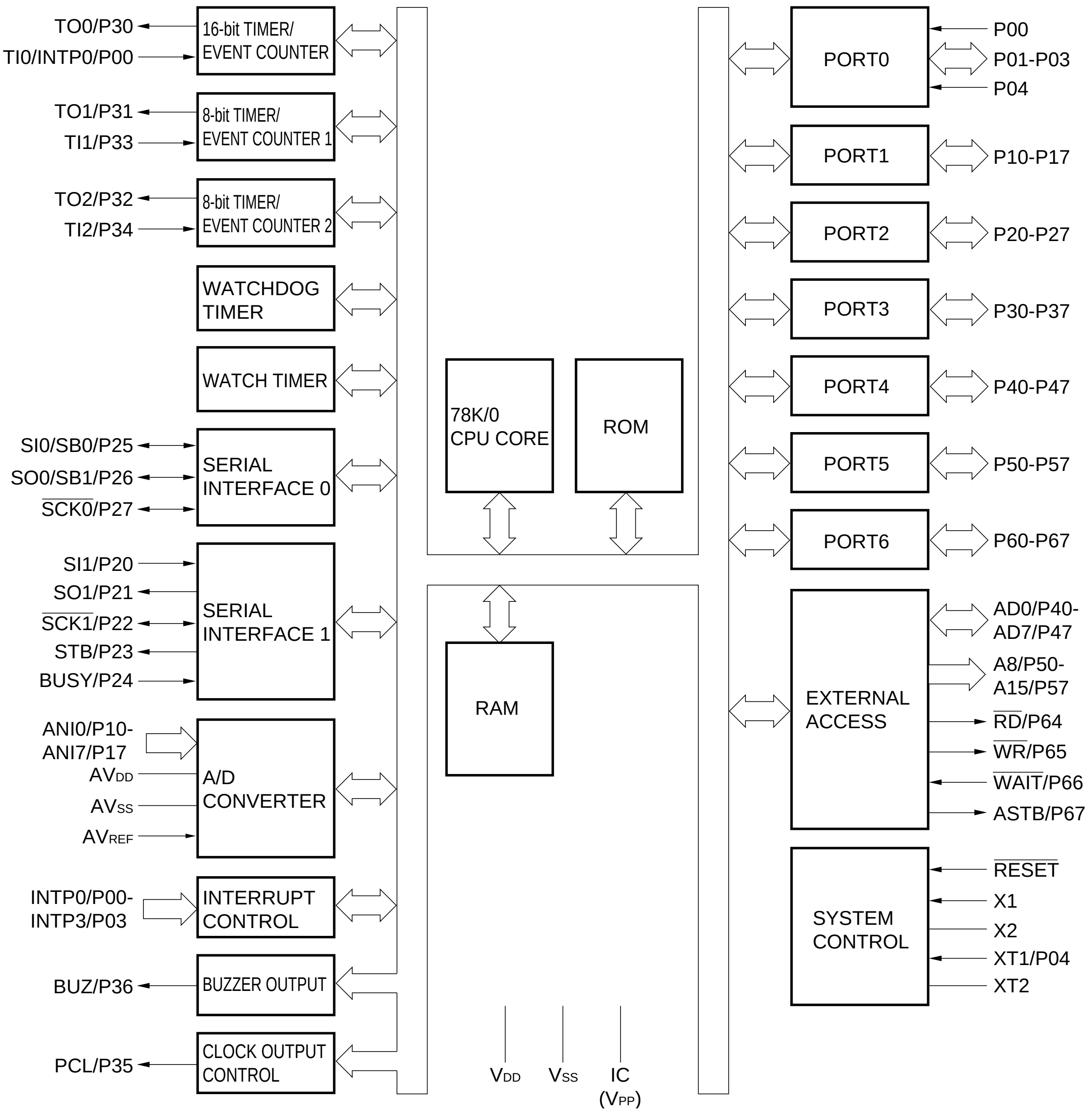
Item	Part Number	μPD78011BY	μPD78012BY	μPD78013Y	μPD78014Y	μPD78P014Y
Internal memory	ROM	Mask ROM				PROM
		8K bytes	16K bytes	24K bytes	32K bytes	32K bytes[Note]
	High-speed RAM	512 bytes		1024 bytes		1024 bytes[Note]
	Buffer RAM	32 bytes				
Memory space		64K bytes				
General-purpose register		8 bits × 8 × 4 banks				
Minimum instruction execution time	With main system clock	0.4 μs/0.8 μs/1.6 μs/3.2 μs/6.4 μs (at 10.0 MHz)				
	With subsystem clock	122 μs (at 32.768 kHz)				
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.				
I/O port		• Total : 53 • CMOS input : 2 • CMOS I/O : 47 (On-chip pull-up resistor ON/OFF selected by software : 47) • N-ch open drain I/O : 4 (15-V withstand, pull-up resistor connected by mask option : 4)				
A/D converter		• 8-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 2.7 to 6.0 V				
Serial interface		• 3-wire serial I/O/SBI/2-wire serial I/O/I^2C bus mode selectable : 1 channel • 3-wire serial I/O mode (with function to automatically transfer/receive up to 32 bytes) : 1 channel				
Timer		• 16-bit timer/event counter : 1 channel • 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel				
Timer output		3 (14-bit PWM output: 1)				
Clock output		39.1 kHz, 78.1 kHz, 156 kHz, 313 kHz, 625 kHz, 1.25 MHz (with main system clock of 10.0 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)				
Buzzer output		2.4 kHz, 4.9 kHz, 9.8 kHz (with main system clock of 10.0 MHz)				

Note The internal PROM and internal high-speed RAM capacities can be changed by using a memory size select register (IMS).

Table 1-4. Functional Outline of μPD78014 Subseries (2/2)

Item \ Part Number		μPD78011BY	μPD78012BY	μPD78013Y	μPD78014Y	μPD78P014Y
Vectored interrupt source	Maskable	Internal: 8, external: 4				
	Non-maskable	Internal: 1				
	Software	1				
Test input		Internal: 1, external: 1				
Supply voltage		V_{DD} = 2.7 to 6.0 V				
Operating temperature		T_A = –40 to +85 °C				
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm) • 64-pin ceramic shrink DIP (750 mil) (μPD78P014Y only)				

Figure 1-5. Block Diagram of μPD78018F Subseries



Remarks 1. The internal ROM and RAM capacities differ depending on the model.

2. ( ): μPD78P018F

Table 1-5. Functional Outline of μPD78018F Subseries (1/2)

★

Part Number Item		μPD78011F	μPD78012F	μPD78013F	μPD78014F	μPD78015F	μPD78016F	μPD78018F	μPD78P018F
Internal memory	ROM	Mask ROM							PROM
		8K bytes	16K bytes	24K bytes	32K bytes	40K bytes	48K bytes	60K bytes	60K bytes[Note 1]
	High-speed RAM	512 bytes		1024 bytes					1024 bytes[Note 1]
	Expansion RAM	None				512 bytes		1024 bytes	1024 bytes[Note 2]
	Buffer RAM	32 bytes							
Memory space		64K bytes							
General-purpose register		8 bits × 8 × 4 banks							
Minimum instruction execution time	With main system clock	0.4 μs/0.8 μs/1.6 μs/3.2 μs/6.4 μs (at 10.0 MHz)							
	With subsystem clock	122 μs (at 32.768 kHz)							
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.							
I/O port		• Total : 53 • CMOS input : 2 • CMOS I/O : 47 (On-chip pull-up resistor ON/OFF selected by software : 47) • N-ch open drain I/O : 4 (15-V withstand, pull-up resistor connected by mask option : 4)							
A/D converter		• 8-bit resolution × 8 channels • Low-voltage operation : AV$_{DD}$ = 1.8 to 5.5 V							
Serial interface		• 3-wire serial I/O/SBI/2-wire serial I/O mode selectable : 1 channel • 3-wire serial I/O mode (with function to automatically transfer/receive up to 32 bytes) : 1 channel							
Timer		• 16-bit timer/event counter : 1 channel • 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel							
Timer output		3 (14-bit PWM output: 1)							
Clock output		39.1 kHz, 78.1 kHz, 156 kHz, 313 kHz, 625 kHz, 1.25 MHz (with main system clock of 10.0 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)							

Notes 1. The capacities of the internal PROM and internal high-speed RAM can be changed by using a memory size select register (IMS).

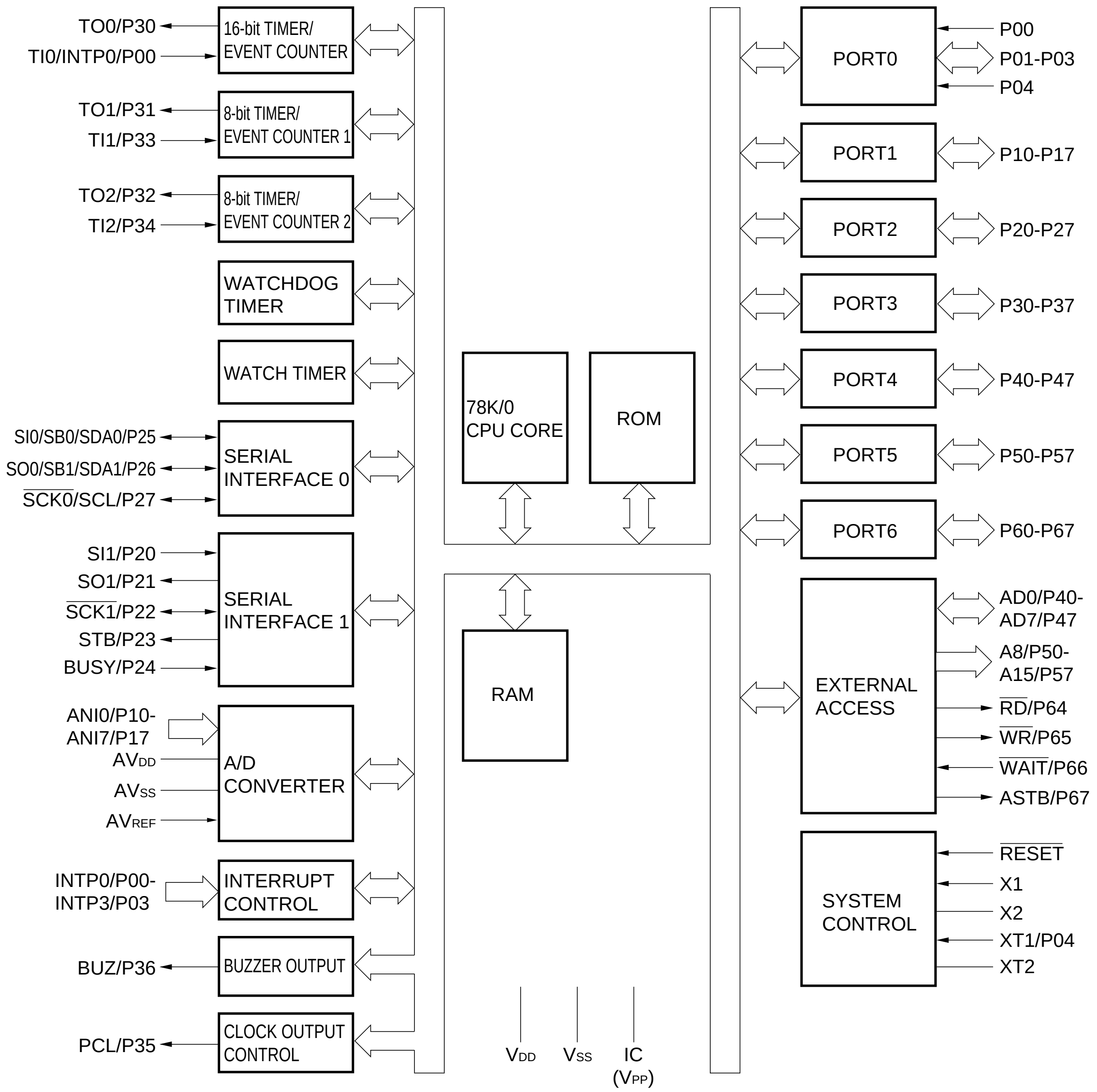
2. The internal expansion RAM capacity can be changed by using an internal expansion RAM size select register (IXS).

Table 1-5. Functional Outline of μPD78018F Subseries (2/2)

Part Number Item		μPD78011F	μPD78012F	μPD78013F	μPD78014F	μPD78015F	μPD78016F	μPD78018F	μPD78P018F
Buzzer output		2.4 kHz, 4.9 kHz, 9.8 kHz (with main system clock of 10.0 MHz)							
Vectored interrupt source	Maskable	Internal: 8, external: 4							
	Non-maskable	Internal: 1							
	Software	1							
Test input		Internal: 1, external: 1							
Supply voltage		V_{DD} = 1.8 to 5.5 V							
Operating Temperature		T_A = –40 to +85 °C							
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm) • 64-pin plastic QFP (12 × 12 mm) • 64-pin ceramic shrink DIP (w/window) (750 mil) : μPD78P018F only • 64-pin ceramic WQFN (14 × 14 mm)[Note] : μPD78P018F only							

Note Under planning

Figure 1-6. Block Diagram of μPD78018FY Subseries



Remarks 1. The internal ROM and RAM capacities differ depending on the model.

2. ( ): μPD78P018FY

Table 1-6. Functional Outline of µPD78018FY Subseries (1/2)

★

Item	Part Number	µPD78011FY	µPD78012FY	µPD78013FY	µPD78014FY	µPD78015FY	µPD78016FY	µPD78018FY	µPD78P018FY
Internal memory	ROM	Mask ROM							PROM
		8K bytes	16K bytes	24K bytes	32K bytes	40K bytes	48K bytes	60K bytes	60K bytes[Note 1]
	High-speed RAM	512 bytes		1024 bytes					1024 bytes[Note 1]
	Expansion RAM	None				512 bytes		1024 bytes	1024 bytes[Note 2]
	Buffer RAM	32 bytes							
Memory space		64K bytes							
General-purpose register		8 bits × 8 × 4 banks							
Minimum instruction execution time	With main system clock	0.4 µs/0.8 µs/1.6 µs/3.2 µs/6.4 µs (at 10.0 MHz)							
	With subsystem clock	122 µs (at 32.768 kHz)							
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.							
I/O port		• Total : 53 • CMOS input : 2 • CMOS I/O : 47 (On-chip pull-up resistor ON/OFF selected by software : 47) • N-ch open drain I/O : 4 (15-V withstand, pull-up resistor connected by mask option : 4)							
A/D converter		• 8-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 1.8 to 5.5 V							
Serial interface		• 3-wire serial I/O/2-wire serial I/O/I^{2}C bus mode selectable : 1 channel • 3-wire serial I/O mode (with function to automatically transfer/receive up to 32 bytes) : 1 channel							
Timer		• 16-bit timer/event counter : 1 channel • 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel							
Timer output		3 (14-bit PWM output: 1)							
Clock output		39.1 kHz, 78.1 kHz, 156 kHz, 313 kHz, 625 kHz, 1.25 MHz (with main system clock of 10.0 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)							

Notes 1. The capacities of the internal PROM and internal high-speed RAM can be changed by using a memory size select register (IMS).

2. The internal expansion RAM capacity can be changed by using an internal expansion RAM size select register (IXS).

Table 1-6. Functional Outline of μPD78018FY Subseries (2/2)

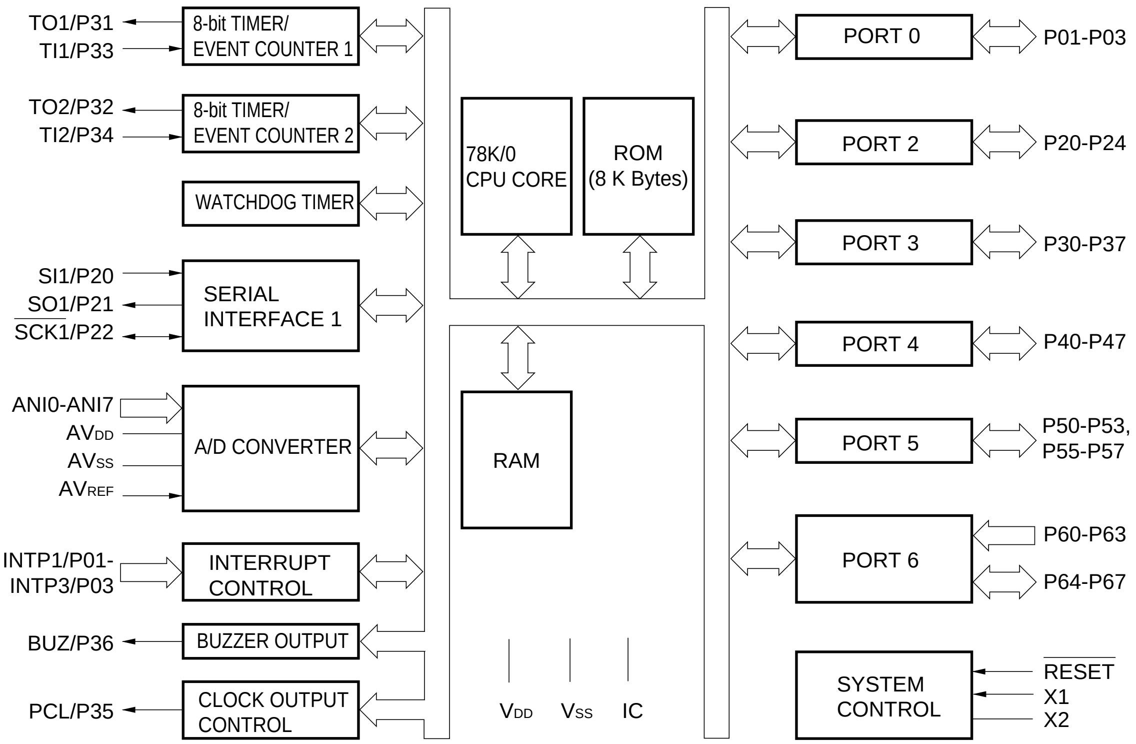
★

Part Number Item		μPD78011FY	μPD78012FY	μPD78013FY	μPD78014FY	μPD78015FY	μPD78016FY	μPD78018FY	μPD78P018FY
Buzzer output		2.4 kHz, 4.9 kHz, 9.8 kHz (with main system clock of 10.0 MHz)							
Vectored interrupt source	Maskable	Internal: 8, external: 4							
	Non-maskable	Internal: 1							
	Software	1							
Test input		Internal: 1, external: 1							
Supply voltage		V_{DD} = 1.8 to 5.5 V							
Operating Temperature		T_A = –40 to +85 °C							
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm) • 64-pin ceramic shrink DIP (w/window) (750 mil)Note : μPD78P018FY only • 64-pin ceramic WQFN (14 × 14 mm)Note : μPD78P018FY only							

Note Under planning

★

Figure 1-7. Block Diagram of μPD780001



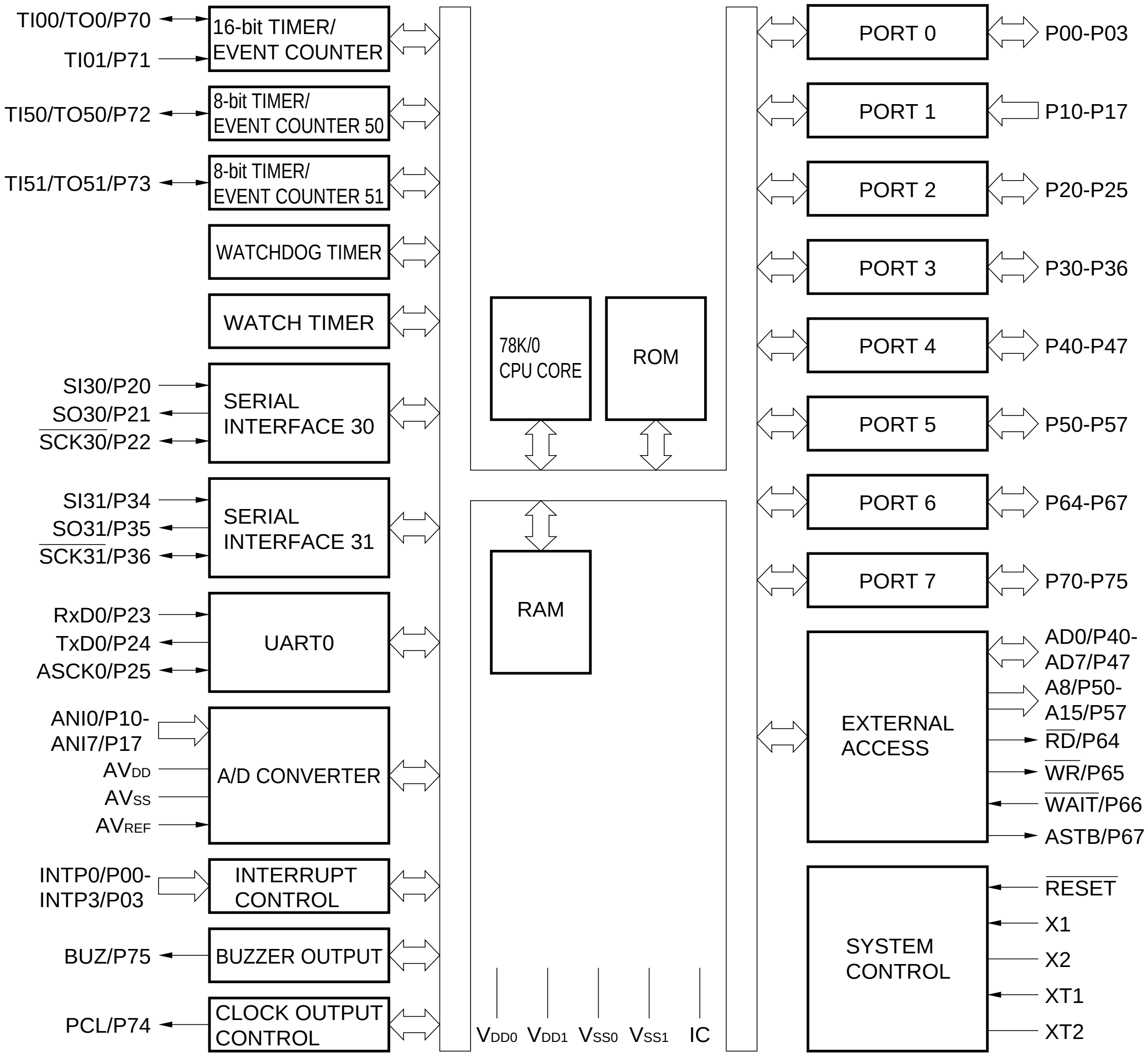
★

Table 1-7. Functional Outline of μPD780001

Item		Function
Internal memory	ROM	Mask ROM
		8K bytes
	High-speed RAM	192 bytes
Memory space		64K bytes
General-purpose register		8 bits × 8 × 4 banks
Minimum instruction time		0.4 μs/0.8 μs/1.6 μs/3.2 μs/6.4 μs (at 10.0 MHz)
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.
I/O port		• Total : 39 • CMOS input : 4 • CMOS I/O : 35 (Pull-up resistor ON/OFF selectable by software : 35)
A/D converter		• 8-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 2.7 to 5.5 V
Serial interface		• 3-wire serial I/O mode : 1 channel
Timer		• 8-bit timer/event counter : 2 channels • Watchdog timer : 1 channel
Timer output		2
Clock output		39.1 kHz, 78.1 kHz, 156 kHz, 313 kHz, 625 kHz, 1.25 MHz (with main system clock of 10.0 MHz)
Buzzer output		2.4 kHz, 4.9 kHz, 9.8 kHz (with main system clock of 10.0 MHz)
Vectored interrupt source	Maskable	Internal: 5, external: 3
	Non-maskable	Internal: 1
	Software	1
Test input		external: 1
Supply voltage		V_{DD} = 2.7 to 5.5 V
Operating temperature		T_A = –40 to +85 °C
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm)

★

Figure 1-8. Block Diagram of μPD780024 Subseries



Remark The internal ROM and RAM capacities differ depending on the model.

★

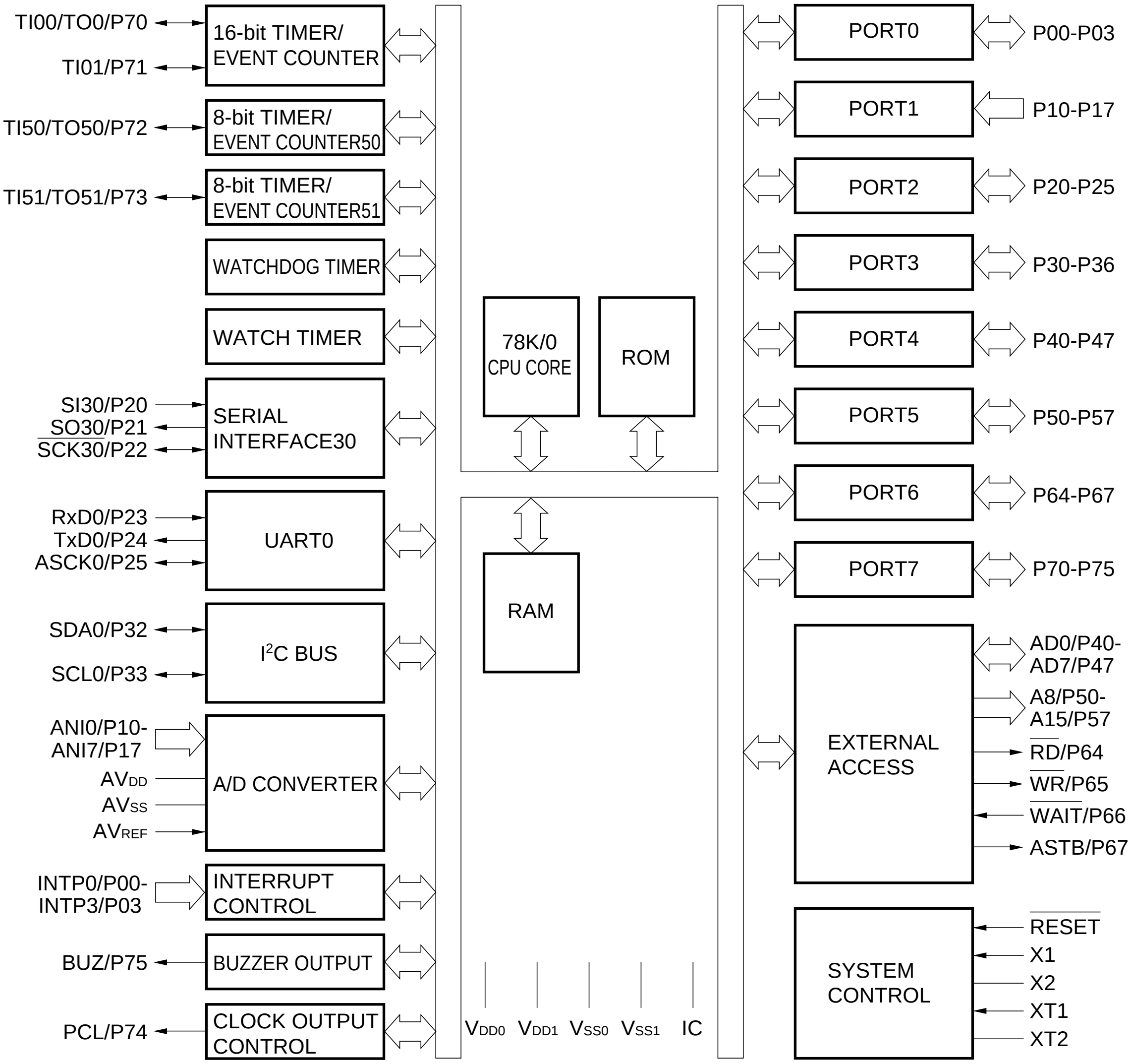
Table 1-8. Functional Outline of µPD780024 Subseries

Item \ Part Number		µPD780021	µPD780022	µPD780023	µPD780024
Internal memory	ROM	Mask ROM			
		8K bytes	16K bytes	24K bytes	32K bytes
	High-speed RAM	512 bytes		1024 bytes	
Memory space		64K bytes			
General-purpose register		8 bits × 8 × 4 banks			
Minimum instruction execution time	With main system clock	0.24 µs/0.48 µs/0.95 µs/1.91 µs/3.81 µs (at 8.38 MHz)			
	With subsystem clock	122 µs (at 32.768 kHz)			
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.			
I/O port		• Total : 51 • CMOS input : 8 • CMOS I/O : 39 (On-chip pull-up resistor ON/OFF selected by software : 39) • N-ch open drain I/O : 4 (5-V withstand, pull-up resistor connected by mask option : 4)			
A/D converter		• 8-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 1.8 to 5.5 V			
Serial interface		• 3-wire serial I/O mode : 2 channels • UART mode : 1 channel			
Timer		• 16-bit timer/event counter : 1 channel • 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel			
Timer output		3 (8-bit PWM output: 2)			
Clock output		131 kHz, 262 kHz, 524 kHz, 1.05 MHz, 2.10 MHz, 4.19 MHz, 8.38 MHz (with main system clock of 8.38 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)			
Buzzer output		65.5 kHz, 1.02 kHz, 2.05 kHz, 4.10 kHz, 8.19 kHz (with main system clock of 8.38 MHz)			
Vectored interrupt source	Maskable	Internal: 13, external: 5			
	Non-maskable	Internal: 1			
	Software	1			
Supply voltage		V_{DD} = 1.8 to 5.5 V			
Operating temperature		T_A = –40 to +85 °C			
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm) • 64-pin plastic LQFP (12 × 12 mm)			

Caution The µPD780024 subseries is under development.

★

Figure 1-9. Block Diagram of μPD780024Y Subseries



Remark The internal ROM and RAM capacities differ depending on the model.

★

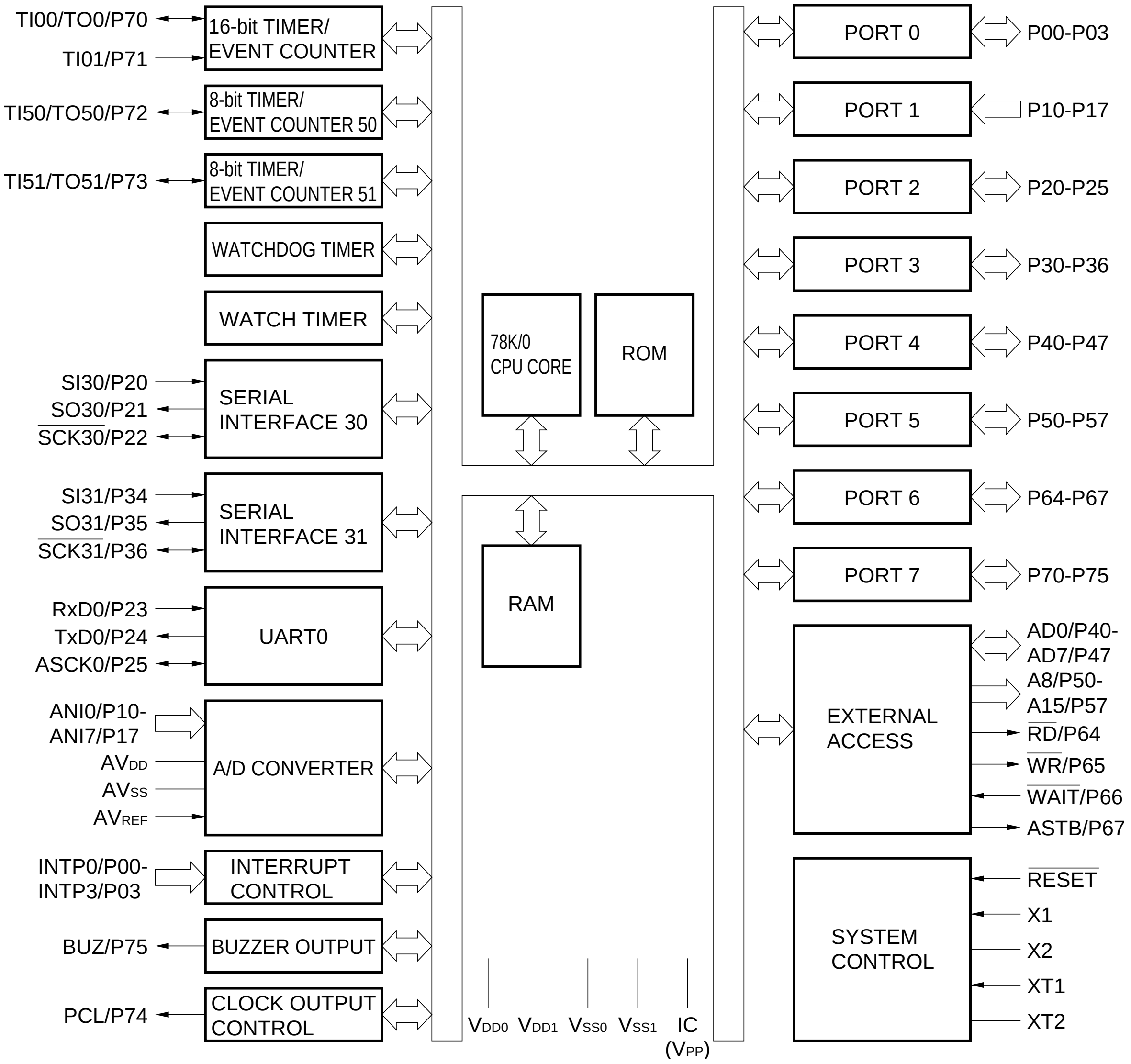
Table 1-9. Functional Outline of μPD780024Y Subseries

Part Number Item		μPD780021Y	μPD780022Y	μPD780023Y	μPD780024Y
Internal memory	ROM	Mask ROM			
		8K bytes	16K bytes	24K bytes	32K bytes
	High-speed RAM	512 bytes		1024 bytes	
Memory space		64K bytes			
General-purpose register		8 bits × 8 × 4 banks			
Minimum instruction execution time	With main system clock	0.24 μs/0.48 μs/0.95 μs/1.91 μs/3.81 μs (at 8.38 MHz)			
	With subsystem clock	122 μs (at 32.768 kHz)			
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.			
I/O port		• Total : 51 • CMOS input : 8 • CMOS I/O : 39 (On-chip pull-up resistor ON/OFF selected by software : 39) • N-ch open drain I/O : 4 (5-V withstand, pull-up resistor connected by mask option : 4)			
A/D converter		• 8-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 1.8 to 5.5 V			
Serial interface		• 3-wire serial I/O mode : 1 channel • UART mode : 1 channel • I^{2}C bus mode : 1 channel			
Timer		• 16-bit timer/event counter : 1 channel • 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel			
Timer output		3 (8-bit PWM output: 2)			
Clock output		131 kHz, 262 kHz, 524 kHz, 1.05 MHz, 2.10 MHz, 4.19 MHz, 8.38 MHz (with main system clock of 8.38 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)			
Buzzer output		65.5 kHz, 1.02 kHz, 2.05 kHz, 4.10 kHz, 8.19 kHz (with main system clock of 8.38 MHz)			
Vectored interrupt source	Maskable	Internal: 13, external: 5			
	Non-maskable	Internal: 1			
	Software	1			
Supply voltage		V_{DD} = 1.8 to 5.5 V			
Operating temperature		T_A = –40 to +85 °C			
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm) • 64-pin plastic LQFP (12 × 12 mm)			

Caution The μPD780024Y subseries is under development.

★

Figure 1-10. Block Diagram of μPD780034 Subseries



Remarks 1. The internal ROM and RAM capacities differ depending on the model.

2. ( ): μPD78F0034

★ **Table 1-10. Functional Outline of μPD780034 Subseries**

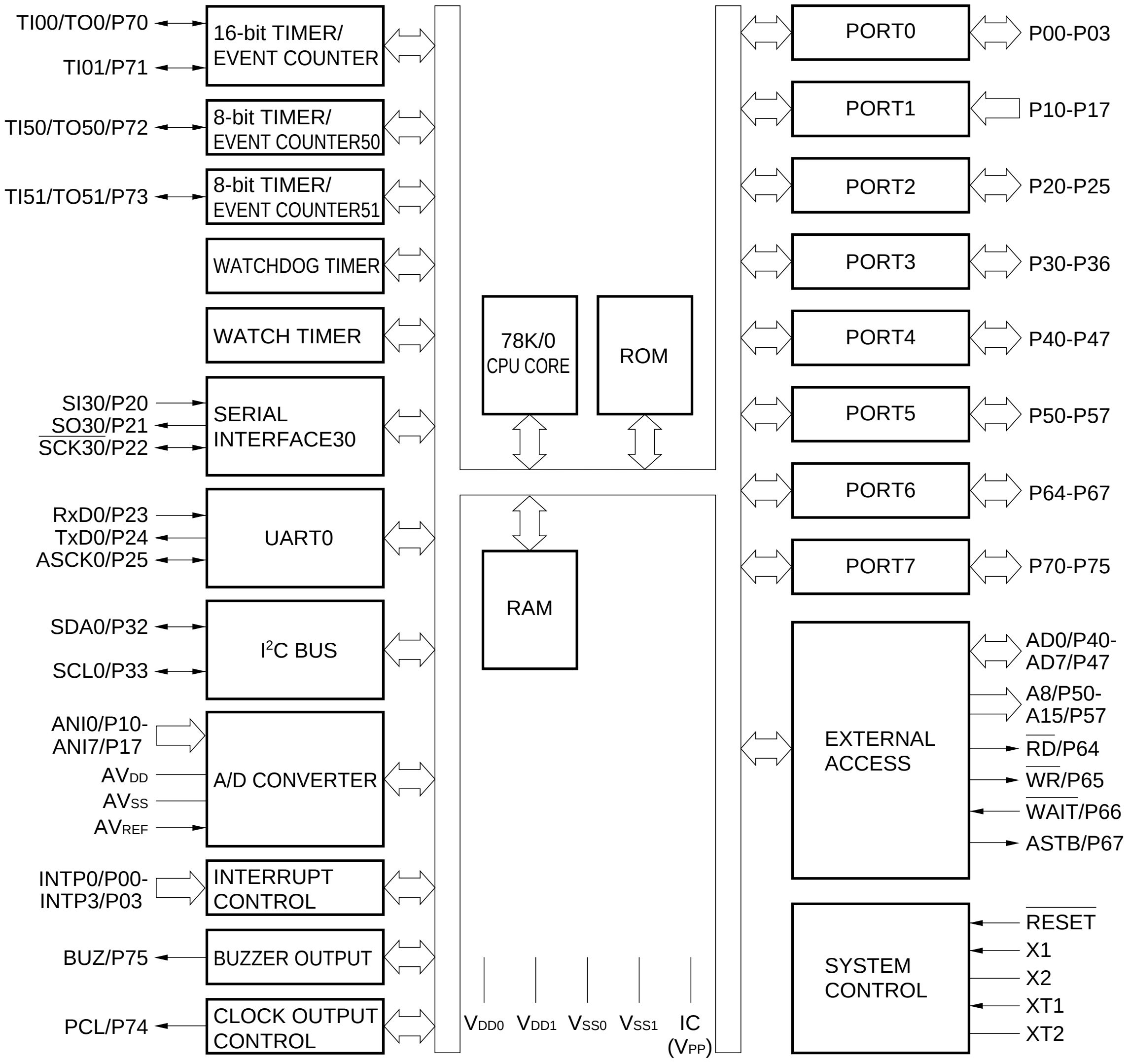
Part Number Item		μPD780031	μPD780032	μPD780033	μPD780034	μPD78F0034
Internal memory	ROM	Mask ROM				Flash memory
		8K bytes	16K bytes	24K bytes	32K bytes	32K bytes[Note]
	High-speed RAM	512 bytes		1024 bytes		1024 bytes[Note]
Memory space		64K bytes				
General-purpose register		8 bits × 8 × 4 banks				
Minimum instruction execution time	With main system clock	0.24 μs/0.48 μs/0.95 μs/1.91 μs/3.81 μs (at 8.38 MHz)				
	With subsystem clock	122 μs (at 32.768 kHz)				
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.				
I/O port		• Total : 51 • CMOS input : 8 • CMOS I/O : 39 (On-chip pull-up resistor ON/OFF selected by software : 39) • N-ch open drain I/O : 4 (5-V withstand, pull-up resistor connected by mask option : 4)				
A/D converter		• 10-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 1.8 to 5.5 V				
Serial interface		• 3-wire serial I/O mode : 2 channels • UART mode : 1 channel				
Timer		• 16-bit timer/event counter : 1 channel • 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel				
Timer output		3 (8-bit PWM output: 2)				
Clock output		131 kHz, 262 kHz, 524 kHz, 1.05 MHz, 2.10 MHz, 4.19 MHz, 8.38 MHz (with main system clock of 8.38 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)				
Buzzer output		65.5 kHz, 1.02 kHz, 2.05 kHz, 4.10 kHz, 8.19 kHz (with main system clock of 8.38 MHz)				
Vectored interrupt source	Maskable	Internal: 13, external: 5				
	Non-maskable	Internal: 1				
	Software	1				
Supply voltage		V_{DD} = 1.8 to 5.5 V				
Operating temperature		T_A = –40 to +85 °C				
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm) • 64-pin plastic LQFP (12 × 12 mm)				

Note The capacities of the flash memory and internal high-speed RAM can be changed by using a memory size select register (IMS).

Caution The μPD780034 subseries is under development.

★

Figure 1-11. Block Diagram of μPD780034Y Subseries



Remarks 1. The internal ROM and RAM capacities differ depending on the model.

2. ( ): μPD78F0034Y

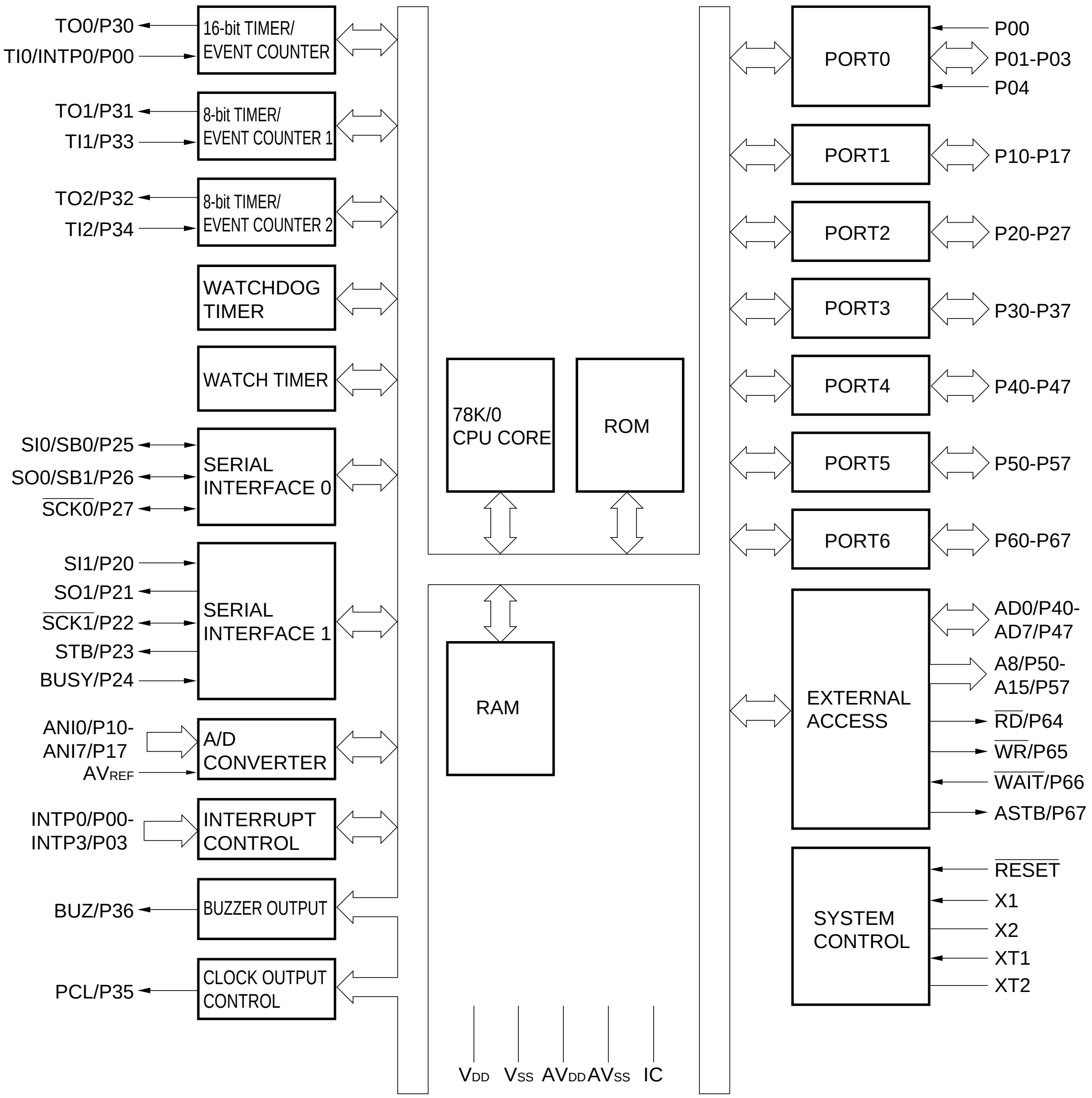
★ **Table 1-11. Functional Outline of μPD780034Y Subseries**

Part Number Item		μPD780031Y	μPD780032Y	μPD780033Y	μPD780034Y	μPD78F0034Y
Internal memory	ROM	Mask ROM				Flash memory
		8K bytes	16K bytes	24K bytes	32K bytes	32K bytes[Note]
	High-speed RAM	512 bytes		1024 bytes		1024 bytes[Note]
Memory space		64K bytes				
General-purpose register		8 bits × 8 × 4 banks				
Minimum instruction execution time	With main system clock	0.24 μs/0.48 μs/0.95 μs/1.91 μs/3.81 μs (at 8.38 MHz)				
	With subsystem clock	122 μs (at 32.768 kHz)				
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.				
I/O port		• Total : 51 • CMOS input : 8 • CMOS I/O : 39 (On-chip pull-up resistor ON/OFF selected by software : 39) • N-ch open drain I/O : 4 (5-V withstand, pull-up resistor connected by mask option : 4)				
A/D converter		• 10-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 1.8 to 5.5 V				
Serial interface		• 3-wire serial I/O mode : 1 channel • UART mode : 1 channel • I^{2}C bus mode : 1 channel				
Timer		• 16-bit timer/event counter : 1 channel • 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel				
Timer output		3 (8-bit PWM output: 2)				
Clock output		131 kHz, 262 kHz, 524 kHz, 1.05 MHz, 2.10 MHz, 4.19 MHz, 8.38 MHz (with main system clock of 8.38 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)				
Buzzer output		65.5 kHz, 1.02 kHz, 2.05 kHz, 4.10 kHz, 8.19 kHz (with main system clock of 8.38 MHz)				
Vectored interrupt source	Maskable	Internal: 13, external: 5				
	Non-maskable	Internal: 1				
	Software	1				
Supply voltage		V_{DD} = 1.8 to 5.5 V				
Operating temperature		T_A = –40 to +85 °C				
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm) • 64-pin plastic LQFP (12 × 12 mm)				

Note The capacities of the flash memory and internal high-speed RAM can be changed by using a memory size select register (IMS).

Caution The μPD780034Y subseries is under development.

★ Figure 1-12. Block Diagram of μPD78014H Subseries



Remark The internal ROM and RAM capacities differ depending on the model.

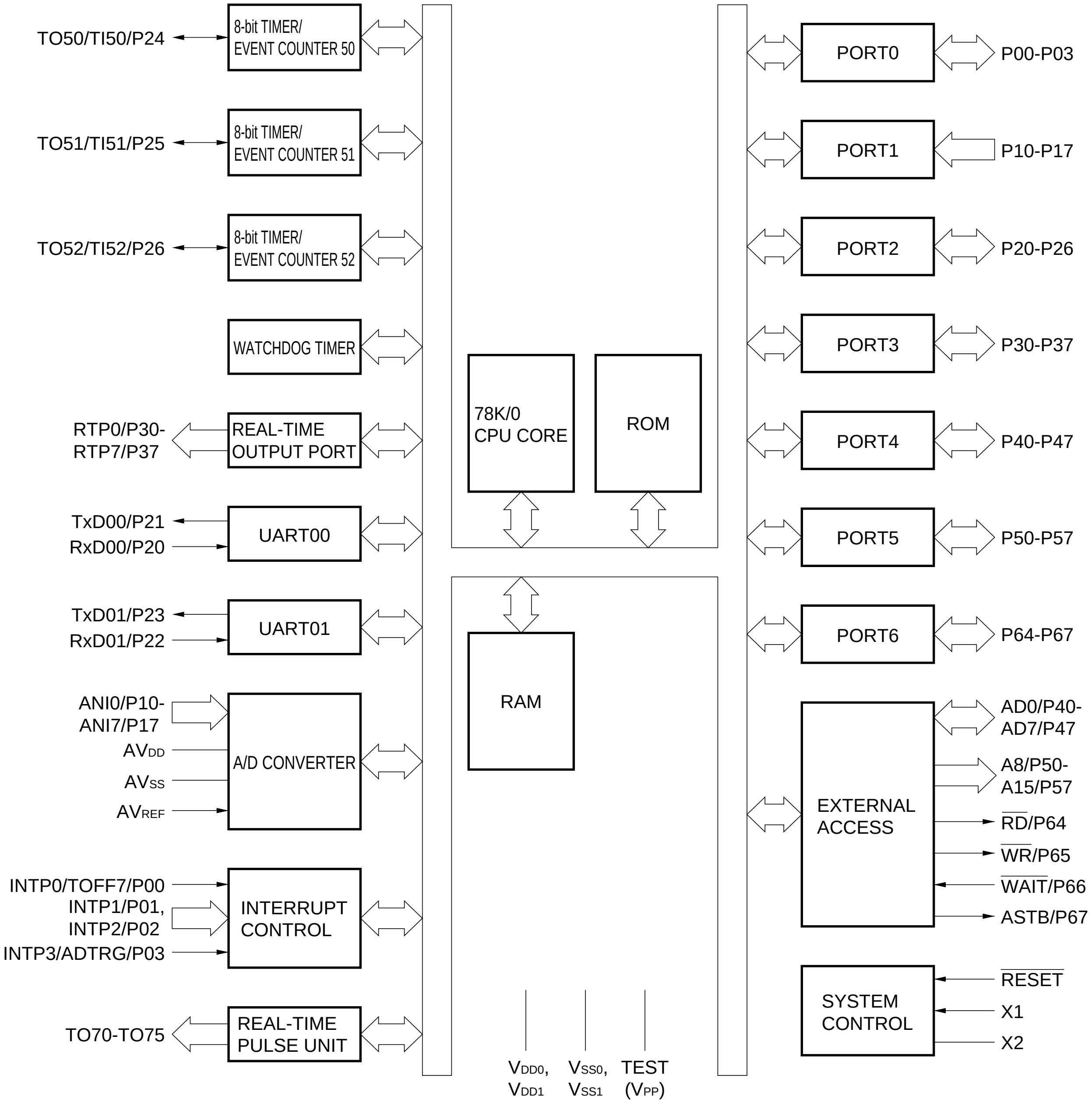
★

Table 1-12. Functional Outline of μPD78014H Subseries

Part Number Item		μPD78011H	μPD78012H	μPD78013H	μPD78014H
Internal memory	ROM	Mask ROM			
		8K bytes	16K bytes	24K bytes	32K bytes
	High-speed RAM	512 bytes		1024 bytes	
	Buffer RAM	32 bytes			
Memory space		64K bytes			
General-purpose register		8 bits × 8 × 4 banks			
Minimum instruction execution time	With main system clock	0.4 μs/0.8 μs/1.6 μs/3.2 μs/6.4 μs (at 10.0 MHz)			
	With subsystem clock	122 μs (at 32.768 kHz)			
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.			
I/O port		• Total : 53 • CMOS input : 2 • CMOS I/O : 47 (On-chip pull-up resistor ON/OFF selected by software : 47) • N-ch open drain I/O : 4 (15-V withstand, pull-up resistor connected by mask option : 4)			
A/D converter		• 8-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 1.8 to 5.5 V			
Serial interface		• 3-wire serial I/O/SBI/2-wire serial I/O mode selectable : 1 channel • 3-wire serial I/O mode (with function to automatically transfer/receive up to 32 bytes) : 1 channel			
Timer		• 16-bit timer/event counter : 1 channel • 8-bit timer/event counter : 2 channels • Watch timer : 1 channel • Watchdog timer : 1 channel			
Timer output		3 (14-bit PWM output: 1)			
Clock output		39.1 kHz, 78.1 kHz, 156 kHz, 313 kHz, 625 kHz, 1.25 MHz (with main system clock of 10.0 MHz), 32.768 kHz (with subsystem clock of 32.768 kHz)			
Buzzer output		2.4 kHz, 4.9 kHz, 9.8 kHz (with main system clock of 10.0 MHz)			
Vectored interrupt source	Maskable	Internal: 8, external: 4			
	Non-maskable	Internal: 1			
	Software	1			
Test input		Internal: 1, external: 1			
Supply voltage		V_{DD} = 1.8 to 5.5 V			
Operating temperature		T_A = –40 to +85 °C			
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm) • 64-pin plastic LQFP (12 × 12 mm)			

★

Figure 1-13. Block Diagram of μPD780924 Subseries



Remarks 1. The internal ROM and RAM capacities differ depending on the model.

2. ( ): μPD78F0924

★

Table 1-13. Functional Outline of μPD780924 Subseries

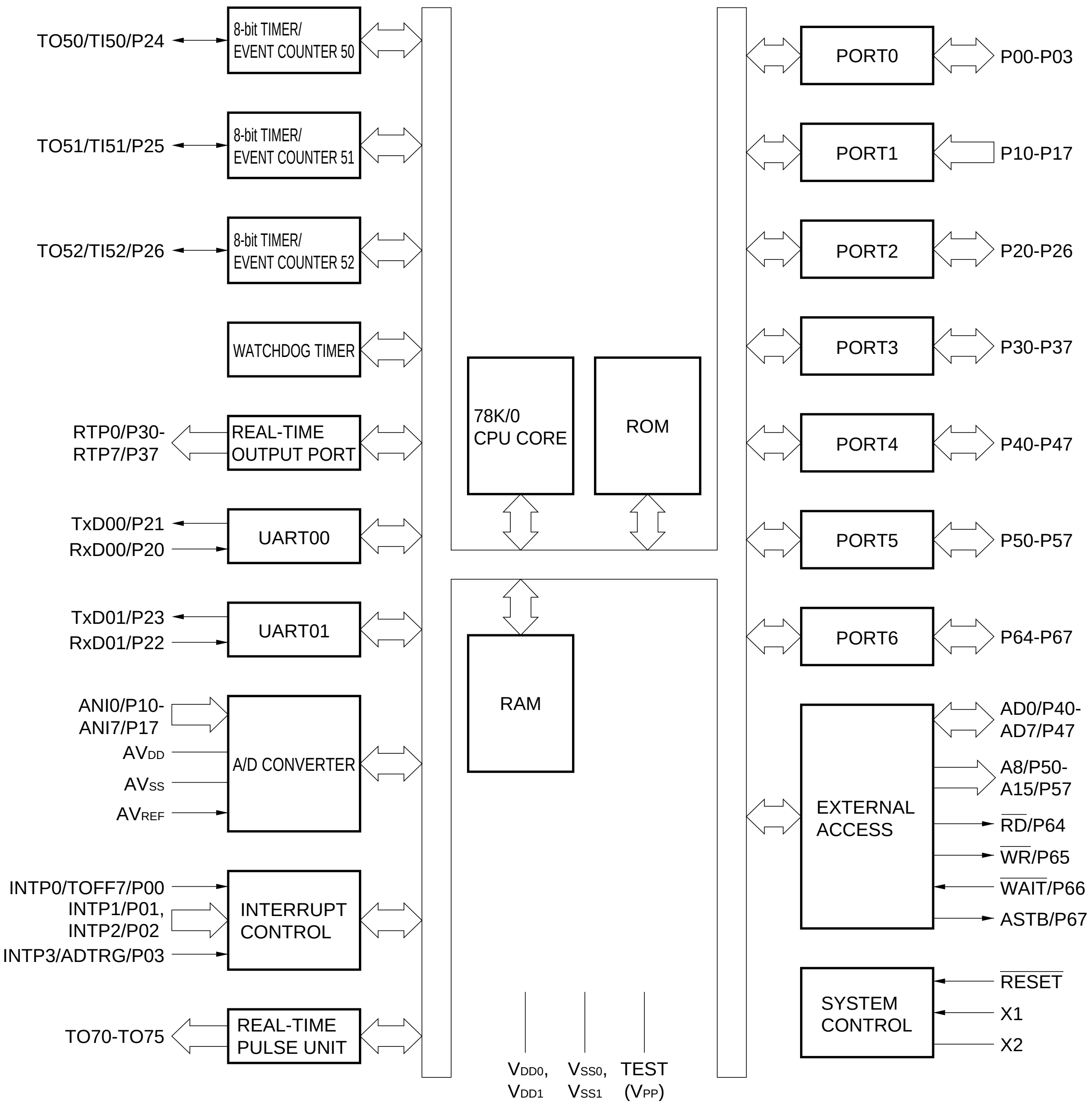
Item \ Part Number		μPD780921	μPD780922	μPD780923	μPD780924	μPD78F0924
Internal memory	ROM	Mask ROM				Flash memory
		8K bytes	16K bytes	24K bytes	32K bytes	32K bytes[Note]
	High-speed RAM	512 bytes		1024 bytes		1024 bytes[Note]
Memory space		64K bytes				
General-purpose register		8 bits × 8 × 4 banks				
Minimum instruction execution time		0.24 μs/0.48 μs/0.96 μs/1.9 μs/3.8 μs (with system clock of 8.38 MHz)				
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.				
I/O port		• Total : 47 • CMOS input : 8 • CMOS I/O : 39 (On-chip pull-up resistor ON/OFF selected by software : 39)				
Real-time output port		• 8 bits × 1 or 4 bits × 2				
A/D converter		• 8-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 2.7 to 5.5 V				
Serial interface		• UART mode : 2 channels				
Timer		• 8-bit timer/event counter : 3 channels • 10-bit inverter control timer : 1 channel • Watchdog timer : 1 channel				
Timer output		9 (8-bit PWM output: 3, inverter control output: 6)				
Vectored interrupt source	Maskable	Internal: 12, external: 4				
	Non-maskable	Internal: 1				
	Software	1				
Supply voltage		V_{DD} = 2.7 to 5.5 V				
Operating temperature		T_A = –40 to +85 °C				
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm)				

Note The capacities of the flash memory and internal high-speed RAM can be changed by using a memory size select register (IMS).

Caution The μPD780924 subseries is under development.

★

Figure 1-14. Block Diagram of μPD780964 Subseries



Remarks 1. The internal ROM and RAM capacities differ depending on the model.

2. ( ): μPD78F0964

★ **Table 1-14. Functional Outline of μPD780964 Subseries**

Part Number Item		μPD780961	μPD780962	μPD780963	μPD780964	μPD78F0964
Internal memory	ROM	Mask ROM				Flash memory
		8K bytes	16K bytes	24K bytes	32K bytes	32K bytes[Note]
	High-speed RAM	512 bytes		1024 bytes		1024 bytes[Note]
Memory space		64K bytes				
General-purpose register		8 bits × 8 × 4 banks				
Minimum instruction execution time		0.24 μs/0.48 μs/0.96 μs/1.9 μs/3.8 μs (with system clock of 8.38 MHz)				
Instruction set		• 16-bit operation • Multiplication/division (8 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (set, reset, test, Boolean operation) • BCD adjustment, etc.				
I/O port		• Total : 47 • CMOS input : 8 • CMOS I/O : 39 (On-chip pull-up resistor ON/OFF selected by software : 39)				
Real-time output port		• 8 bits × 1 or 4 bits × 2				
A/D converter		• 10-bit resolution × 8 channels • Low-voltage operation : AV_{DD} = 2.7 to 5.5 V				
Serial interface		• UART mode : 2 channels				
Timer		• 8-bit timer/event counter : 3 channels • 10-bit inverter control timer : 1 channel • Watchdog timer : 1 channel				
Timer output		9 (8-bit PWM output: 3, inverter control output: 6)				
Vectored interrupt source	Maskable	Internal: 12, external: 4				
	Non-maskable	Internal: 1				
	Software	1				
Supply voltage		V_{DD} = 2.7 to 5.5 V				
Operating temperature		T_A = –40 to +85 °C				
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 × 14 mm)				

Note The capacities of the flash memory and internal high-speed RAM can be changed by using a memory size select register (IMS).

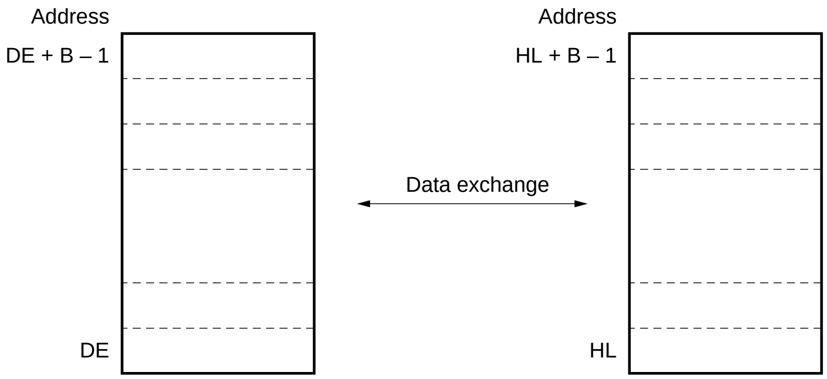
Caution The μPD780964 subseries is under development.

[MEMO]

2.1 Data Transfer

Data is exchanged by using an address specified by the DE and HL registers as the first address. The number of bytes of the data to be exchanged is specified by the B register.

Figure 2-1. Data Exchange



(1) Registers used

A, B, DE, HL

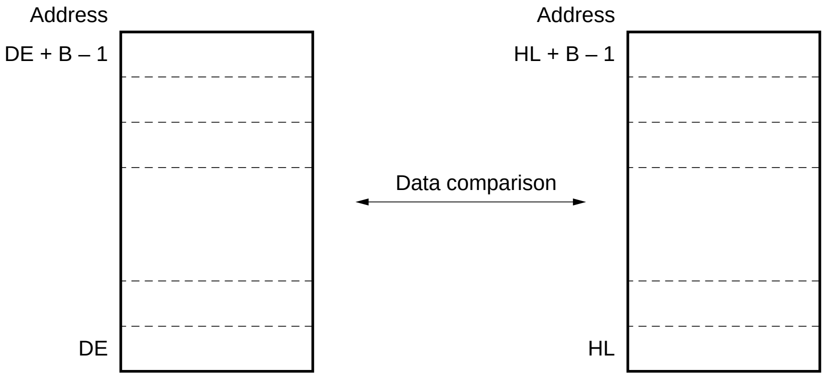
(2) Program list

```
EXCH:
        MOV     A,[DE]
        XCH     A,[HL]
        XCH     A,[DE]
        INCW    DE
        INCW    HL
        DBNZ    B,$EXCH
        RET
```

2.2 Data Comparison

Data is compared by using an address specified by the DE and HL registers as the first address. The number of bytes of the data to be compared is specified by the B register. If the result of comparison is equal, CY is cleared to 0; if not, CY is set to 1.

Figure 2-2. Data Comparison



(1) Registers used

A, B, DE, HL

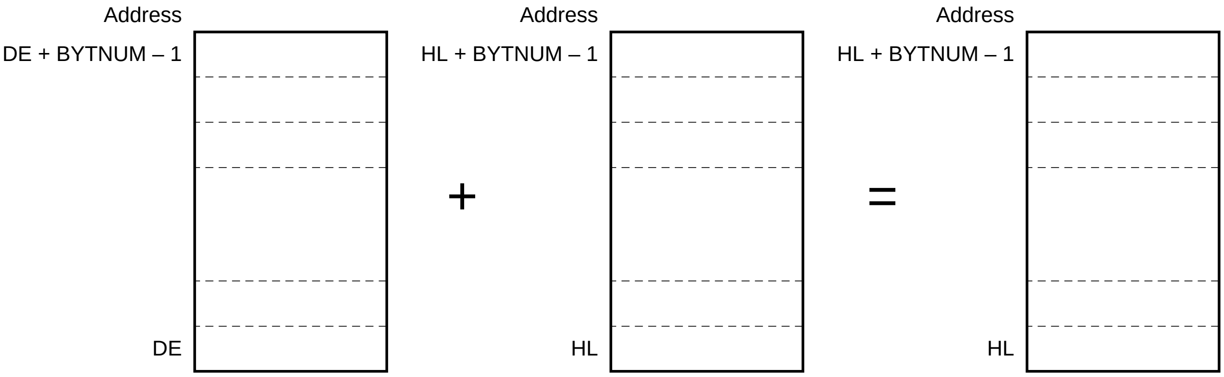
(2) Program list

```
COMP:
        MOV     A,[DE]
        CMP     A,[HL]
        BNZ     $ERROR
        INCW    DE
        INCW    HL
        DBNZ    B,$COMP
        CLR1    CY
        BR      RTN
ERROR:
        SET1    CY
RTN:
        RET
```

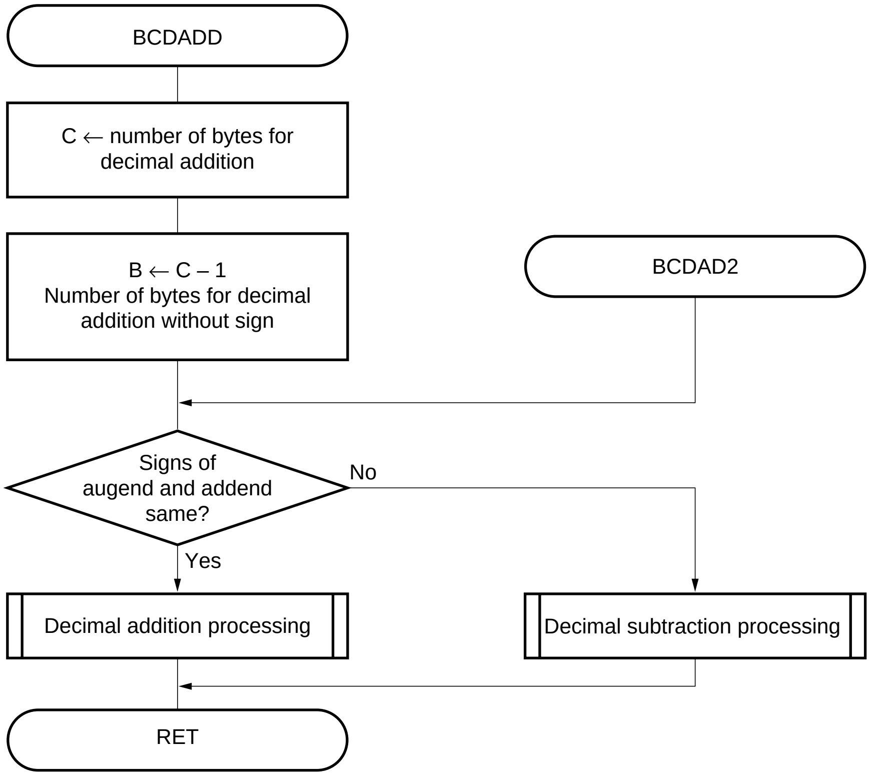
2.3 Decimal Addition

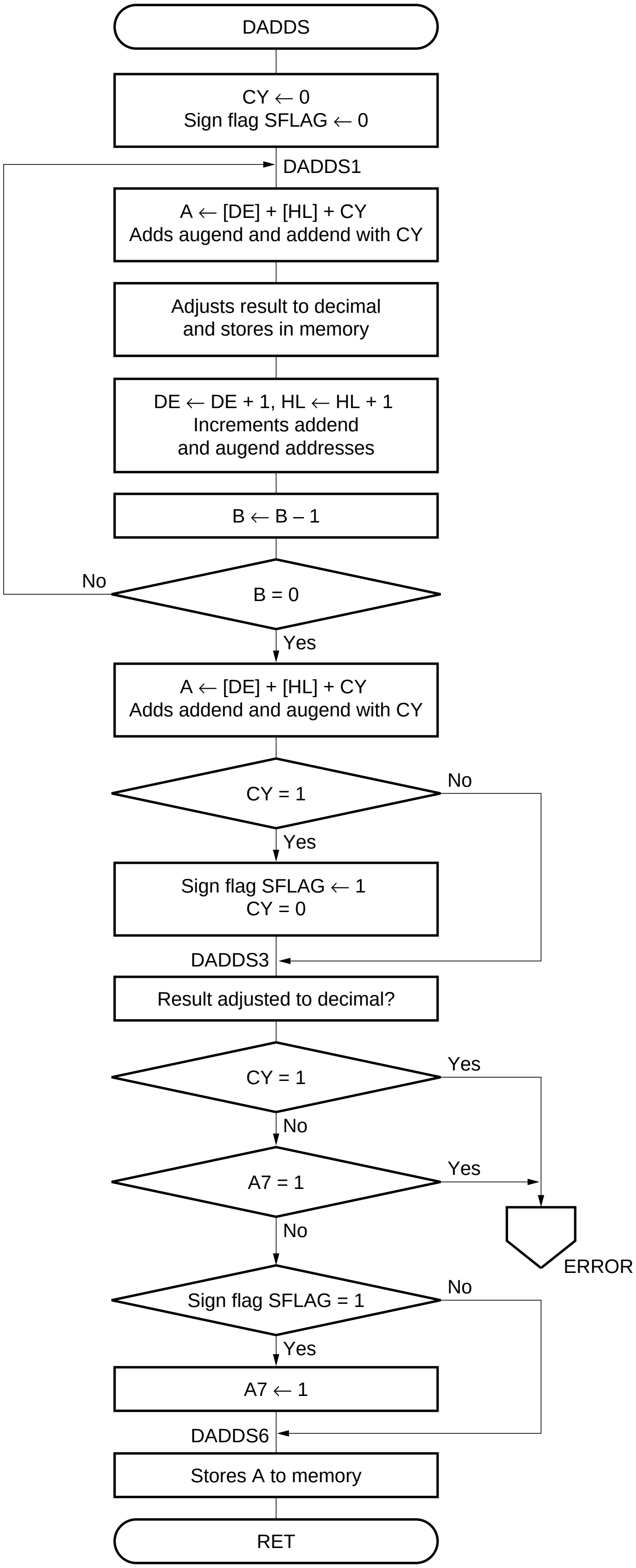
The lowest address for decimal addition is specified by the DE and HL registers, and the number of digits specified by BYTNUM is added. The result of the addition is stored to an area specified by the HL register. If an overflow or underflow occurs as a result of the addition, execution branches to error processing. Define the branch address as 'ERROR' in the main routine. Also declare it as PUBLIC.

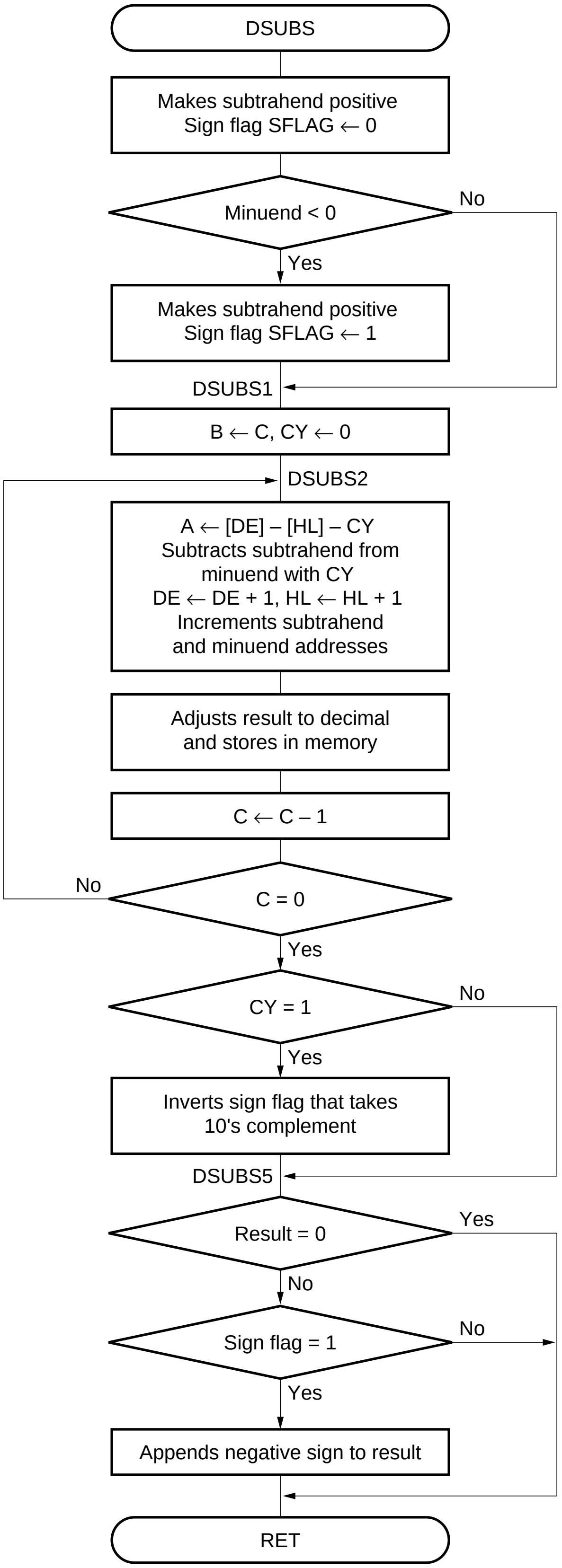
Figure 2-3. Decimal Addition



(1) Flowchart


BCDADD
C ← number of bytes for decimal addition
B ← C – 1
Number of bytes for decimal addition without sign
BCDAD2
Signs of augend and addend same?
No
Yes
Decimal addition processing
Decimal subtraction processing
RET


DADDS
CY ← 0
Sign flag SFLAG ← 0
DADDS1
A ← [DE] + [HL] + CY
Adds augend and addend with CY
Adjusts result to decimal
and stores in memory
DE ← DE + 1, HL ← HL + 1
Increments addend
and augend addresses
B ← B – 1
No
B = 0
Yes
A ← [DE] + [HL] + CY
Adds addend and augend with CY
CY = 1
No
Yes
Sign flag SFLAG ← 1
CY = 0
DADDS3
Result adjusted to decimal?
CY = 1
Yes
No
A7 = 1
Yes
No
ERROR
Sign flag SFLAG = 1
No
Yes
A7 ← 1
DADDS6
Stores A to memory
RET


DSUBS
Makes subtrahend positive
Sign flag SFLAG ← 0
Minuend < 0
No
Yes
Makes subtrahend positive
Sign flag SFLAG ← 1
DSUBS1
B ← C, CY ← 0
DSUBS2
A ← [DE] – [HL] – CY
Subtracts subtrahend from
minuend with CY
DE ← DE + 1, HL ← HL + 1
Increments subtrahend
and minuend addresses
Adjusts result to decimal
and stores in memory
C ← C – 1
No
C = 0
Yes
CY = 1
No
Yes
Inverts sign flag that takes
10's complement
DSUBS5
Result = 0
Yes
No
Sign flag = 1
No
Yes
Appends negative sign to result
RET

(2) Registers used

AX, BC, DE, HL

(3) Program list

```
;*******************************************************************
;                                                                  *
;       Input parameter                                            *
;              HL register: addend first address                   *
;              DE register: augend first address                   *
;       Output parameter                                           *
;              HL register: Operation result first address         *
;                                                                  *
;*******************************************************************

        PUBLIC  BCDADD,BCDAD1,BCDAD2
        PUBLIC  DADDS
        PUBLIC  DSUBS
        EXTRN   ERROR                        ; Error processing branch address
        EXTBIT  SFLAG                        ; Sign flag
;
BYTNUM  EQU     4                            ; Sets number of digits for operation
;
        CSEG
BCDADD:
        MOV     C,#BYTNUM                    ; Sets number of digits for operation to C register
BCDAD1:
        MOV     A,C
        MOV     B,A
        DEC     B
BCDAD2:
        MOV     A,[HL+BYTNUM–1]              ; Loads MSB (sign data) of augend
        XCHW    AX,DE
        XCHW    AX,HL
        XCHW    AX,DE
        XOR     A,[HL+BYTNUM–1]              ; Loads MSB (sign data) of augend
        XCHW    AX,HL
        XCHW    AX,DE
        XCHW    AX,HL

        BT      A.7,$BCDAD3                  ; Signs coincide? ELSE subtraction processing
        CALL    !DADDS                       ; THEN addition processing
        RET
BCDAD3:
        CALL    !DSUBS
        RET
```

```
;==============================================================
;              ***** 10 Decimal addition *****
;==============================================================

DADDS:
        CLR1    CY
        CLR1    SFLAG
DADDS1:
        MOV     A,[DE]                  ; Starts addition from lowest digit
        ADDC    A,[HL]
        ADJBA
        MOV     [HL],A
        INCW    HL
        INCW    DE
        DBNZ    B,$DADDS1               ; End of addition (number of digits for operation – 1)

        MOV     A,[DE]
        ADDC    A,[HL]
DADDS2:
        BNC     $DADDS3                 ; Negative addition
        SET1    SFLAG                   ; THEN sets negative status
        CLR1    CY
DADDS3:
        ADJBA
        BNC     $DADDS4
        BR      ERROR
DADDS4:
        BF      A.7,$DADDS5
        BR      ERROR
DADDS5:
        BF      SFLAG,$DADDS6           ; Sets sign
        SET1    A.7
DADDS6:
        MOV     [HL],A
        RET
```

```
;================================================================
;              ***** 10 Decimal subtraction *****
;================================================================

DSUBS:
       PUSH   HL
       CLR1   SFLAG
       MOV    A,[HL+BYTNUM–1]          ; Sets subtrahend as positive value
       CLR1   A.7
       MOV    [HL+BYTNUM–1],A
       XCHW   AX,DE
       XCHW   AX,HL
       XCHW   AX,DE
       MOV    A,[HL+BYTNUM–1]
       BF     A.7,$DSUBS1              ; Minuend is negative
       CLR1   A.7                      ; THEN sets minuend as positive value
       MOV    [HL+BYTNUM–1],A
       SET1   SFLAG                    ; Sets sign as negative
DSUBS1:
       XCHW   AX,HL
       XCHW   AX,DE
       XCHW   AX,HL
       MOV    A,C
       MOV    B,A
       CLR1   CY
DSUBS2:
       MOV    A,[DE]
       SUBC   A,[HL]
       ADJBS
       MOV    [HL],A
       INCW   HL
       INCW   DE
       DBNZ   C,$DSUBS2                ; End of subtraction of number of digits for operation

       BNC    $DSUBS5                  ; THEN subtrahend > minuend
       POP    HL
       PUSH   HL
       MOV    A,B
       MOV    C,A
DSUBS3:
       MOV    A,#99H                   ; Complement operation of result of subtraction
       SUB    A,[HL]                   ;                    (result of subtraction – 99H)
       ADJBS
       MOV    [HL],A
       INCW   HL
       DBNZ   C,$DSUBS3


       POP    HL
       PUSH   HL
       SET1   CY

       MOV    A,B
       MOV    C,A
DSUBS4:
       MOV    A,#0                     ; Adds 1 to result of complement operation
       ADDC   A,[HL]
       ADJBA
```

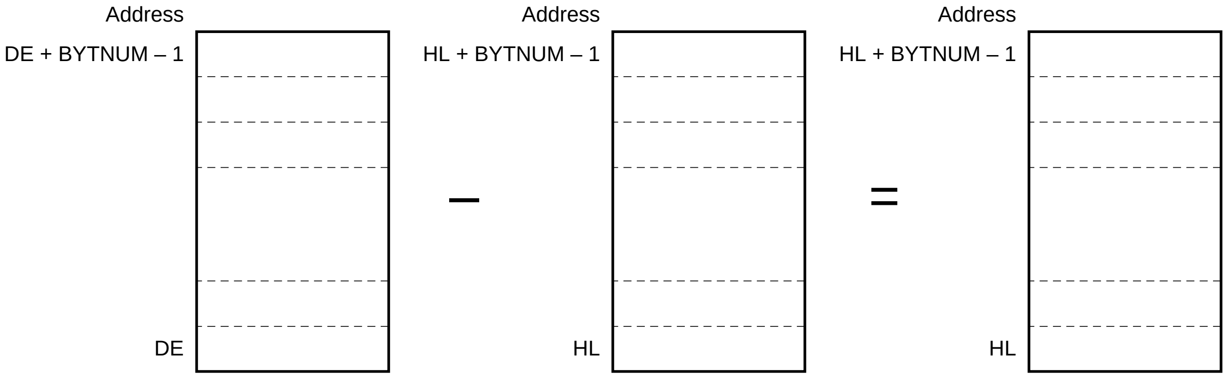
```
        MOV     [HL],A
        INCW    HL
        DBNZ    C,$DSUBS4
        MOV1    CY,SFLAG
        NOT1    CY
        MOV1    SFLAG,CY
;======================================================
;       ***** 0 check of operation result *****
;======================================================

DSUBS5:
        MOV     A,B
        MOV     C,A
        POP     HL
        PUSH    HL
        MOV     A,#0
DSUBS6:
        CMP     A,[HL]                  ; 0 check from lowest digit
        INCW    HL
        BNZ     $DSUBS7
        DBNZ    C,$DSUBS6               ; 0 check of all digits completed
        POP     HL                      ; THEN result of subtraction = 0
        RET
DSUBS7:
        BF      SFLAG,$DSUBS8           ; Result of subtraction is negative
        POP     HL                      ; THEN sets sign
        PUSH    HL
        MOV     A,[HL+BYTNUM-1]
        SET1    A.7
        MOV     [HL+BYTNUM-1],A
DSUBS8:
        POP     HL
        RET
```

2.4 Decimal Subtraction

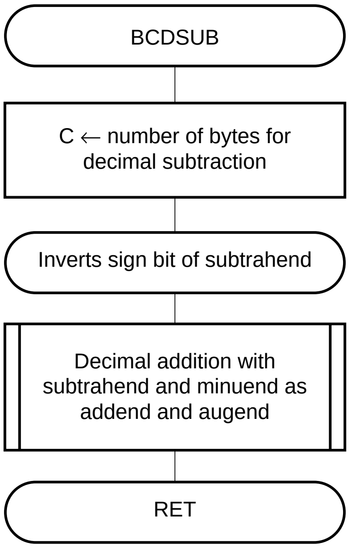
The lowest address for decimal subtraction is specified by the DE and HL registers, and the number of digits specified by BYTNUM is subtracted. The result of the subtraction is stored to an area specified by the HL register. If an overflow or underflow occurs as a result of the subtraction, execution branches to error processing. Define the branch address as ‘ERROR’ in the main routine. Also declare it as PUBLIC.

This program replaces minuend and subtrahend with augend and addend, and calls a program of decimal addition.

Figure 2-4. Decimal Subtraction



(1) Flowchart


BCDSUB
C ← number of bytes for decimal subtraction
Inverts sign bit of subtrahend
Decimal addition with subtrahend and minuend as addend and augend
RET

(2) Registers used

AX, BC, DE, HL

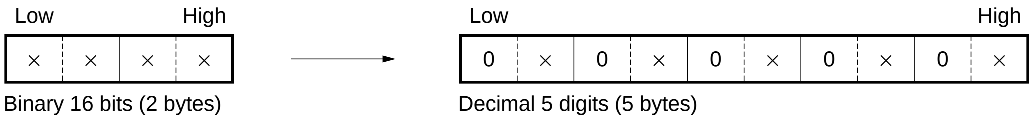
(3) Program list

```
;******************************************************************
;                                                                *
;       Input parameter                                          *
;               HL register: subtrahend first address            *
;               DE register: minuend first address               *
;       Output parameter                                         *
;               HL register: Operation result first address      *
;                                                                *
;******************************************************************

        PUBLIC  BYTNUM
        PUBLIC  BCDSUB
        EXTRN   BCDADD,BCDAD2
;
BYTNUM  EQU     4                       ; Sets number of digits for operation
;
        CSEG
BCDSUB:
        MOV     C,#BYTNUM               ; Sets number of digits for operation to C register
BCDSU1:
        MOV     A,C
        MOV     B,A
        DEC     B

        MOV     A,[HL+BYTNUM-1]         ; Sets MSB (sign data) of subtrahend for addition
        MOV1    CY,A.7                  ; Inverts sign data
        NOT1    CY
        MOV1    A.7,CY
        MOV     [HL+BYTNUM-1],A
        CALL    !BCDAD2                 ; Calls decimal addition processing
        RET
```

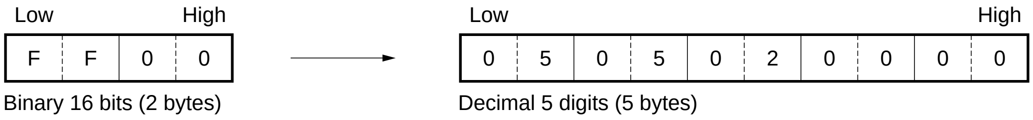
2.5 Binary-to-Decimal Conversion

Binary data of 16 bits in data memory is converted into 5-digit decimal data and stored in data memory. Binary data of 16 bits is divided by decimal 10 by the number of times equal to the number of digits (4 times), and conversion is carried out with the result of the operation and the value of the remainder at that time.

Figure 2-5. Binary-to-Decimal Conversion



Example To convert FFH into decimal number



(1) Registers used

AX, BC, HL

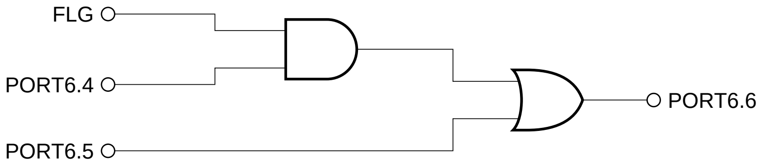
(2) Program list

```
        PUBLIC  B_DCONV
        DATDEC  EQU    10

        DSEG    SADDRP
REGA:   DS      2                       ;  Stores binary 16-bit data
REGB:   DS      5                       ;  Stores decimal 5-digit data

        COLNUM  EQU    4

B_DCONV:
        MOVW    AX,REGA
        MOV     B,#COLNUM
        MOVW    HL,#REGB
B_D1:
        MOV     C,#DATDEC
        DIVUW   C
        XCH     A,C
        MOV     [HL],A
        INCW    HL
        XCH     A,C
        DBNZ    B,$B_D1
        MOV     A,X
        MOV     [HL],A
        RET
```

2.6 Bit Manipulation Instruction

A 1 bit of a flag in the data memory is ANDed with the bit 4 of port 6, and the result is ANDed with the bit 5 of port 6 and is output to the bit 6 of port 6.

Figure 2-6. Bit Operation



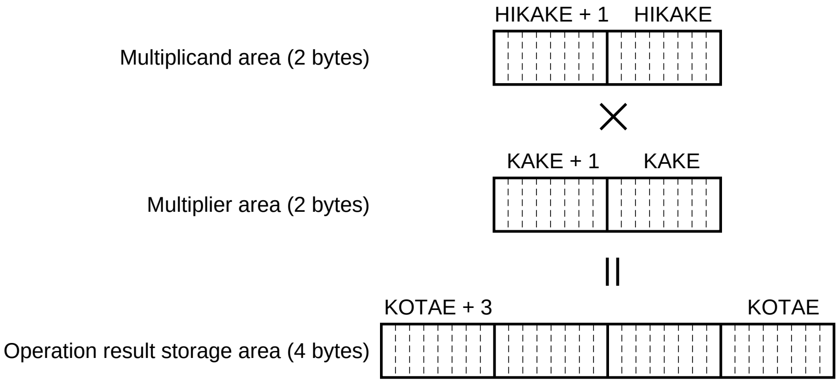
(1) Program list

```
        PUBLIC  BIT_OP,FLG

        BSEG
FLG     DBIT

BIT_OP:
        MOV1    CY,FLG
        AND1    CY,P6.4
        OR1     CY,P6.5
        MOV1    P6.6,CY
        RET
```

2.7 Binary Multiplication (16 bits × 16 bits)

Data in a multiplicand area (HIKAKE; 16 bits) and multiplier area (KAKE; 16 bits) are multiplied, and the result is stored in an operation result storage area (KOTAE).

Figure 2-7. Binary Multiplication


HIKAKE + 1
HIKAKE
Multiplicand area (2 bytes)
×
KAKE + 1
KAKE
Multiplier area (2 bytes)
||
KOTAE + 3
KOTAE
Operation result storage area (4 bytes)

<Processing contents>

Multiplication is performed by adding the multiplicand by the number of bits of the multiplier that are "1".

<Contents used>

Set the data in the multiplicand (HIKAKE) and multiplier (KAKE) areas, and call subroutine S_KAKERU.

```
        EXTRN   S_KAKERU
        EXTRN   HIKAKE,KAKE,KOTAE
MAIN:                           ; Multiplier
                .
                .
        HIKAKE=WORKA  (A)       ; Stores multiplicand data to multiplicand area
        HIKAKE+1=WORKA+1 (A)    ;
        KAKE=WORKB (A)          ; Stores multiplier data to multiplier area
        KAKE+1=WORKB+1 (A)      ;
        CALL    !S_KAKERU       ; Calls multiplication routine
        HL=#KOTAE               ; HL ← RAM address of operation result storage area
                .               ; Stores result by indirect address transfer
                .
                .
```

Caution Manipulate the data memory in 8-bit units.

(1) Input/output condition

- Input parameter

  HIKAKE : Store the multiplicand data in this area.

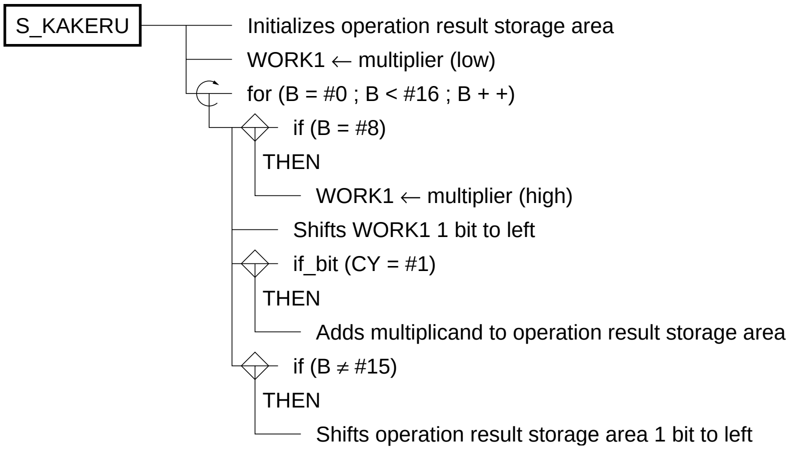
  KAKE : Store the multiplier data in this area.

- Output parameter

  KOTAE : Store the result of the operation in this area.

(2) SPD chart

[Multiplication subroutine]



```
S_KAKERU
    Initializes operation result storage area
    WORK1 ← multiplier (low)
    for (B = #0 ; B < #16 ; B + +)
        if (B = #8)
        THEN
            WORK1 ← multiplier (high)
        Shifts WORK1 1 bit to left
        if_bit (CY = #1)
        THEN
            Adds multiplicand to operation result storage area
        if (B ≠ #15)
        THEN
            Shifts operation result storage area 1 bit to left
```

(3) Registers used

A, B

(4) Program list

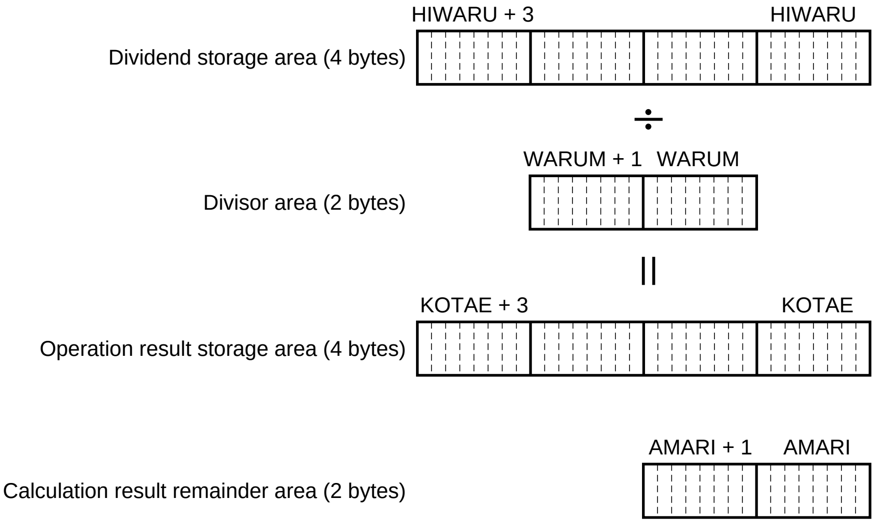
```
$PC(014)
;
PUBLIC HIKAKE,S_KAKERU,KAKE,KOTAE
;
;************************************************
;            RAM definition
;************************************************
           DSEG   SADDR
HIKAKE:    DS     2                              ; Multiplicand area
KAKE:      DS     2                              ; Multiplier area
WORK1:     DS     1                              ; Work area
KOTAE:     DS     4                              ; Operation result storage area
;
;************************************************
;            Multiplication
;************************************************
       CSEG                                      ;
S_KAKERU:                                        ;
       WORK1=KAKE+1 (A)                          ; Stores multiplier (low) in work area
       KOTAE=#0                                  ; Initializes operation result storage area
       KOTAE+1=#0                                ;
       KOTAE+2=#0                                ;
       KOTAE+3=#0                                ;
       for(B=#0;B<#16;B++)(A)                    ; Stores higher multiplier in work area
           if(B == #8)(A)                        ; if low multiplication is completed
               WORK1=KAKE (A)                    ;
               endif                             ;
               A=WORK1                           ; Shifts multiplier 1 bit to left
               CLR1   CY                         ;
               ROLC   A,1                        ;
               WORK1=A                           ;
               if_bit(CY)                        ; Adds multiplicand to operation
                  KOTAE+=HIKAKE (A)              ;     result storage area if carry occurs
                  (KOTAE+1)+=HIKAKE+1,CY (A)     ;
                  (KOTAE+2)+=#0,CY (A)           ;
                  (KOTAE+3)+=#0,CY (A)           ;
               endif                             ;
               if(B != #15) (A)                  ;
                   KOTAE+=KOTAE (A)              ; Shifts operation result storage area 1 bit to left
                   KOTAE+1+=KOTAE+1,CY (A)       ;
                   KOTAE+2+=KOTAE+2,CY (A)       ;
                   KOTAE+3+=KOTAE+3,CY (A)       ;
               endif                             ;
           next                                  ;
           RET                                   ;
           END
```

2.8 Binary Division (32 bits ÷ 16 bits)

Data in a dividend area (HIWARU; 32 bits) is divided by data in a divisor area (WARUM; 16 bits), and the result is stored in an operation result storage area (KOTAE). If a remainder is generated, it is stored in a calculation result reminder area (AMARI).

If division is executed with the divisor being 0, an error occurs.

Figure 2-8. Binary Division


HIWARU + 3
HIWARU
Dividend storage area (4 bytes)
÷
WARUM + 1
WARUM
Divisor area (2 bytes)
||
KOTAE + 3
KOTAE
Operation result storage area (4 bytes)
AMARI + 1
AMARI
Calculation result remainder area (2 bytes)

<Processing contents>

The dividend is shifted to the left to the work area starting from the highest digit. If the contents of the work area are greater than the divisor, the divisor is subtracted from the work area, and the least significant bit of the dividend is set to 1. In this way, division is carried out by executing the program by the number of bits of the dividend.

If the divisor is 0, an error flag (F_ERR) is set.

<Usage>

Set data in the dividend area (HIWARU) and divisor area (WARUM), and call subroutine S_WARU.

```
          EXTRN   S_WARU
          EXTRN   HIWARU,WARUM,KOTAE
          EXBIT   F_ERR

  MAIN:
                  .                         ;
                  .                         ;
          HIWARU=WORKA  (A)                 ;  Stores dividend data to dividend area
          HIWARU+1=WORKA+1 (A)              ;
          WARUM=WORKB (A)                   ;  Stores divisor data to divisor area
          WARUM+1=WORKB+1 (A)               ;
          CALL    !S_WARU                   ;  Calls division calculation routine
          HL=#KOTAE                         ;  HL ← stores RAM address of operation result storage area
                  .                         ;
                  .                         ;
          if_bit(F_ERR)                     ;
             Calculation error processing   ;
          endif                             ;
                  .
                  .
                  .
```

Caution Manipulate the data memory in 8-bit units.

(1) Input/output conditions

- Input parameter

  HIWARU: Store the dividend data in this area.

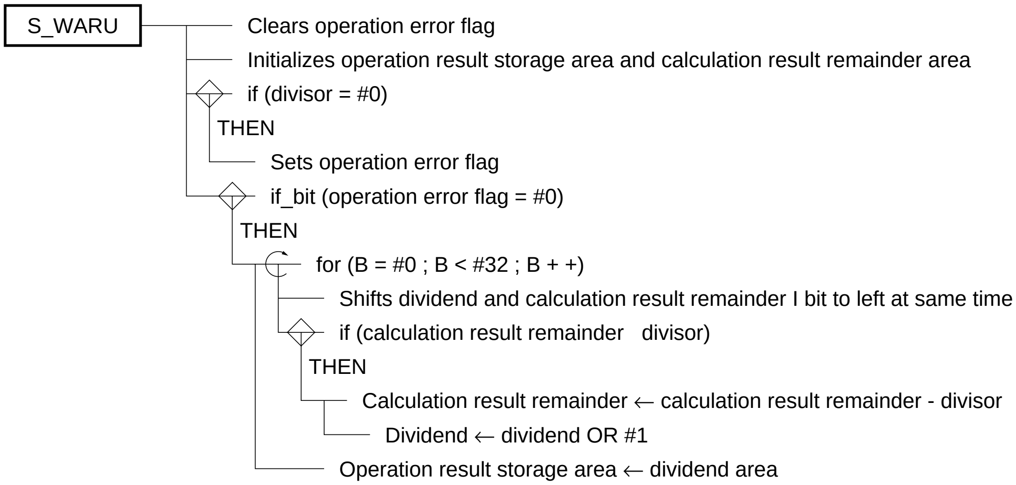
  WARUM : Store the divisor data in this area.

- Output parameter

  KOTAE : Store the result of the calculation in this area.

(2) SPD chart

[Division subroutine]



```
S_WARU
  Clears operation error flag
  Initializes operation result storage area and calculation result remainder area
  if (divisor = #0)
    THEN
      Sets operation error flag
  if_bit (operation error flag = #0)
    THEN
      for (B = #0 ; B < #32 ; B + +)
        Shifts dividend and calculation result remainder I bit to left at same time
        if (calculation result remainder   divisor)
          THEN
            Calculation result remainder ← calculation result remainder - divisor
            Dividend ← dividend OR #1
      Operation result storage area ← dividend area
```

(3) Registers used

A, B

(4) Program list

```
$PC(014)
;
PUBLIC S_WARU,HIWARU,WARUM,F_ERR
EXTRN  KOTAE
;
;************************************************
;            RAM definition
;************************************************
           DSEG    SADDR
HIWARU:    DS      4                             ; Dividend area
WARUM:     DS      2                             ; Divisor area
AMARI:     DS      2                             ; Calculation result remainder storage area
           BSEG
F_ERR      DBIT                                  ; Operation error flag
;************************************************
;                 Division
;************************************************
        CSEG                                     ;
S_WARU:                                          ;
        CLR1       F_ERR                         ; Clears operation error flag
        AMARI=#0                                 ; Clears calculation result storage area to 0
        AMARI+1=#0                               ;
        KOTAE=#0                                 ; Clears operation result storage area to 0
        KOTAE+1=#0                               ;
        KOTAE+2=#0                               ;
        KOTAE+3=#0                               ;
        if(WARUM == #0)                          ; Divisor = 0?
            if(WARUM+1 == #0)                    ;
                SET1    F_ERR                    ; Sets operation error flag if divisor is 0
            endif                                ;
        endif                                    ;
        if_bit(!F_ERR)                           ; Operation error?
            for(B=#0;B < #32;B++) (A)            ; Starts 32-bit division
                HIWARU+=HIWARU (A)               ; Shifts dividend and remainder 1 bit to left
                HIWARU+1+=HIWARU+1,CY  (A)       ;
                HIWARU+2+=HIWARU+2,CY  (A)       ;
                HIWARU+3+=HIWARU+3,CY  (A)       ;
                AMARI+=AMARI,CY (A)              ;
                AMARI+1+=AMARI+1,CY (A)          ;
                                                 ;
                if(AMARI+1 > WARUM+1) (A)        ; Remainder ≥ divisor?
                    AMARI–=WARUM (A)             ;    Remainder = remainder – divisor
                    AMARI+1–=WARUM+1,CY (A)      ;
                    HIWARU |= #1                 ; Stores 1 to first bit of dividend area
                elseif_bit(Z)                    ;
                    if(AMARI >= WARUM) (A)       ;
                        AMARI–=WARUM(A)          ;
                        AMARI+1–=WARUM+1,CY (A)  ;
                        HIWARU |= #1             ;
                    endif                        ;
                endif                            ;
            next                                 ;
            KOTAE=HIWARU (A)                     ; Stores operation result
            KOTAE+1=HIWARU+1 (A)                 ;
            KOTAE+2=HIWARU+2 (A)                 ;
            KOTAE+3=HIWARU+3 (A)                 ;
        endif                                    ;
        RET                                      ;
        END
```

CHAPTER 3 APPLICATION OF SYSTEM CLOCK SELECTION

The 78K/0 series allows you to select a CPU clock and controls the operation of the oscillator by rewriting the contents of the processor clock control register (PCC).

When the CPU clock is changed, the time shown in Table 3-1 and 3-2 is required since the contents of the PCC have been rewritten until the CPU clock is actually changed. It is therefore not apparent for a while after the contents of the PCC have been rewritten, whether the processor operates on the new or old clock. To stop the main system clock or execute the STOP instruction, therefore, the wait time shown in Table 3-1 and 3-2 is necessary.

★ **Table 3-1. Maximum Time Required for Changing CPU Clock**
(μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 780024, 780024Y, 780034, 780034Y, 78014H subseries)

Set Value before Change				Set Value after Change					
CSS	PCC2	PCC1	PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0	CSS PCC2 PCC1 PCC0
				0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	1 × × ×
0	0	0	0		16 instructions	16 instructions	16 instructions	16 instructions	$f_X/4f_{XT}$ instructions (77 instructions)
	0	0	1	8 instructions		8 instructions	8 instructions	8 instructions	$f_X/8f_{XT}$ instructions (39 instructions)
	0	1	0	4 instructions	4 instructions		4 instructions	4 instructions	$f_X/16f_{XT}$ instructions (20 instructions)
	0	1	1	2 instructions	2 instructions	2 instructions		2 instructions	$f_X/32f_{XT}$ instructions (10 instructions)
	1	0	0	1 instruction	1 instruction	1 instruction	1 instruction		$f_X/64f_{XT}$ instructions (5 instructions)
1	×	×	×	1 instruction	1 instruction	1 instruction	1 instruction	1 instruction	

Caution Do not select dividing the CPU clock (PCC0-PCC2) and changing from the main system clock to subsystem clock (by setting CSS to 0 → 1) at the same time.
However, dividing the CPU clock (PCC0-PCC2) can be selected at the same time as changing from the subsystem clock to the main system clock.

Remarks 1. One instruction is the minimum instruction execution time of the CPU clock before change.
2. ( ): f_X = 10.0 MHz, f_{XT} = 32.768 kHz

★

Table 3-2. Maximum Time Required for Changing CPU Clock (μPD780924, 780964 subseries, μPD780001)

Set Value before Change			Set Value after Change														
PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0
			0	0	0	0	0	1	0	1	0	0	1	1	1	0	0
0	0	0				16 instructions			16 instructions			16 instructions			16 instructions		
0	0	1	8 instructions						8 instructions			8 instructions			8 instructions		
0	1	0	4 instructions			4 instructions						4 instructions			4 instructions		
0	1	1	2 instructions			2 instructions			2 instructions						2 instructions		
1	0	0	1 instruction			1 instruction			1 instruction			1 instruction					

Remark One instruction is the minimum instruction execution time of the CPU clock before change.

★ **Figure 3-1. Format of Processor Clock Control Register (μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
PCC	MCC	FRC	CLS	CSS	0	PCC2	PCC1	PCC0	FFFBH	04H	R/W[Note1]

R/W	CSS	PCC2	PCC1	PCC0	Selects CPU clock (f_{CPU})
	0	0	0	0	f_X
		0	0	1	$f_X/2$
		0	1	0	$f_X/2^2$
		0	1	1	$f_X/2^3$
		1	0	0	$f_X/2^4$
	1	0	0	0	f_{XT}
		0	0	1	
		0	1	0	
		0	1	1	
		1	0	0	
	Others				Setting prohibited

R	CLS	Status of CPU clock
	0	Main system clock
	1	Subsystem clock

R/W	FRC	Selects feedback resistor of subsystem clock
	0	Uses internal feedback resistor
	1	Does not use internal feedback resistor

R/W	MCC	Controls oscillation of main system clock[Note 2]
	0	Enables oscillation
	1	Stops oscillation

Notes 1. Bit 5 is a read-only bit.

2. Use MCC to stop the oscillation of the main system clock when the CPU operates on the subsystem clock. Do not use the STOP instruction.

Caution Be sure to clear bit 3 to 0.

Remarks 1. f_X : main system clock oscillation frequency

2. f_{XT} : subsystem clock oscillation frequency

★ Figure 3-2. Format of Processor Clock Control Register (μPD780001)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
PCC	0	0	0	0	0	PCC2	PCC1	PCC0	FFFBH	04H	R/W

PCC2	PCC1	PCC0	Selects CPU clock (f_{CPU})
0	0	0	f_X
0	0	1	$f_X/2$
0	1	0	$f_X/2^2$
0	1	1	$f_X/2^3$
1	0	0	$f_X/2^4$
Others			Setting prohibited

Caution Be sure to clear bit 3 to 0.

Remark f_X : main system clock oscillation frequency

The fastest instruction of the μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries, and μPD780001 is executed in two CPU clocks. Therefore, the relation between the CPU clock (f_{CPU}) and minimum instruction execution time is as shown in Table 3-3.

★ Table 3-3. Relation between CPU Clock and Minimum Instruction Execution Time (μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries, μPD780001)

CPU Clock (f_{CPU})	Minimum Instruction Execution Time: $4/f_{CPU}$
f_X	0.4 μs
$f_X/2$	0.8 μs
$f_X/2^2$	1.6 μs
$f_X/2^3$	3.2 μs
$f_X/2^4$	6.4 μs
f_{XT}**Note**	122 μs

Note Except μPD780001

Remark f_X = 10.0 MHz, f_{XT} = 32.768 kHz
f_X : Main system clock oscillation frequency
f_{XT}: Subsystem clock oscillation frequency

★

Figure 3-3. Format of Processor Clock Control Register (μPD780024, 780024Y, 780034, 780034Y subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
PCC	MCC	FRC	CLS	CSS	0	PCC2	PCC1	PCC0	FFFBH	04H	R/W^{Note 1}

CSS	PCC2	PCC1	PCC0	Selects CPU clock (f_{CPU})
0	0	0	0	f_X
	0	0	1	$f_X/2$
	0	1	0	$f_X/2^2$
	0	1	1	$f_X/2^3$
	1	0	0	$f_X/2^4$
1	0	0	0	f_{XT}
	0	0	1	
	0	1	0	
	0	1	1	
	1	0	0	
Others				Setting prohibited

CLS	Status of CPU clock
0	Main system clock
1	Subsystem clock

FRC	Selects feedback resistor of subsystem clock
0	Uses internal feedback resistor
1	Does not use internal feedback resistor

MCC	Controls oscillation of main system clock^{Note 2}
0	Enables oscillation
1	Stops oscillation

Notes 1. Bit 5 is a read-only bit.

2. Use MCC to stop the oscillation of the main system clock when the CPU operates on the subsystem clock. Do not use the STOP instruction.

Caution Be sure to clear bit 3 to 0.

Remarks 1. f_X : main system clock oscillation frequency

2. f_{XT}: subsystem clock oscillation frequency

★ **Figure 3-4. Format of Processor Clock Control Register (μPD780924, 780964 subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
PCC	0	0	0	0	0	PCC2	PCC1	PCC0	FFFBH	04H	R/W

PCC2	PCC1	PCC0	Selects CPU clock (f_{CPU})
0	0	0	f_X
0	0	1	$f_X/2$
0	1	0	$f_X/2^2$
0	1	1	$f_X/2^3$
1	0	0	$f_X/2^4$
Others			Setting prohibited

Caution Be sure to clear bits 3 through 7 to 0.

Remark f_X : main system clock oscillation frequency

The fastest instruction of the μPD780024, 780024Y, 780034, 780034Y, 780924, 780964 subseries is executed in two CPU clocks. Therefore, the relation between the CPU clock (f_{CPU}) and minimum instruction execution time is as shown in Table 3-4.

Table 3-4. Relation between CPU Clock and Minimum Instruction Execution Time (μPD780024, 780024Y, 780034, 780034Y, 780924, 780964 subseries)

CPU Clock (f_{CPU})	Minimum Instruction Execution Time: $2/f_{CPU}$
f_X	0.24 μs
$f_X/2$	0.48 μs
$f_X/2^2$	0.96 μs
$f_X/2^3$	1.9 μs
$f_X/2^4$	3.8 μs
$f_{XT}/2$[Note]	122 μs

Note Except μPD780924 and 780964 subseries

Remark f_X = 8.38 MHz, f_{XT} = 32.768 kHz
f_X : Main system clock oscillation frequency
f_{XT}: Subsystem clock oscillation frequency

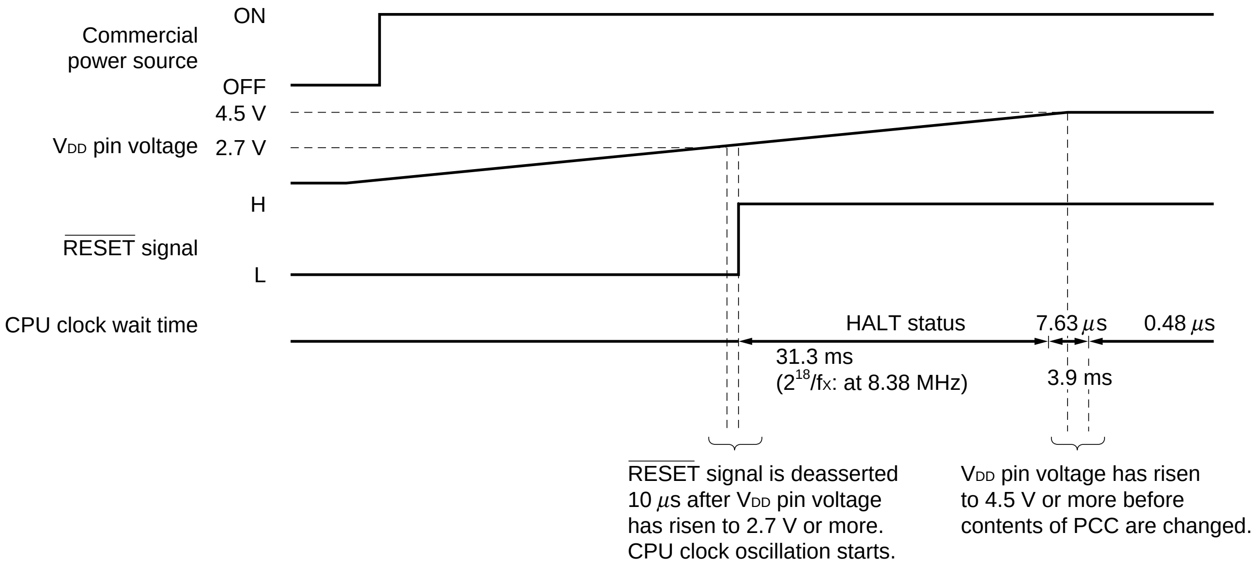
3.1 Changing PCC Immediately after $\overline{\text{RESET}}$

When the $\overline{\text{RESET}}$ signal is asserted, the slowest mode of the main system clock is selected for the CPU clock. To set the highest speed of the CPU clock, therefore, the contents of the processor clock control register (PCC) must be rewritten. To use the fasted mode, however, the voltage on the V_{DD} pin has to have risen to a sufficient level and be stable.

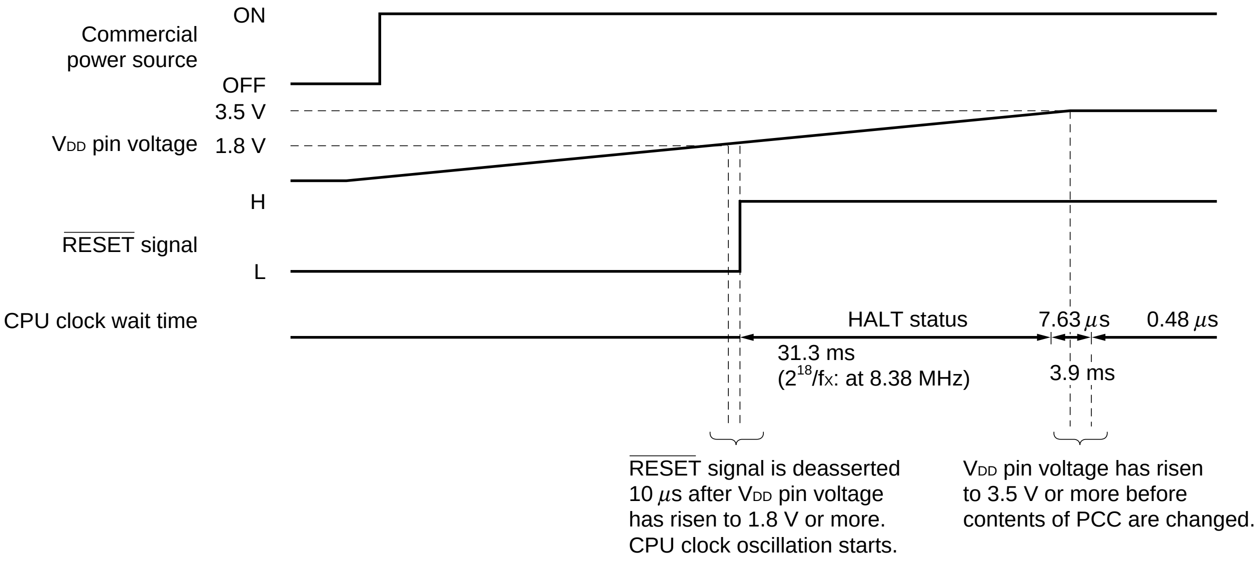
In the following example, the CPU waits until the V_{DD} pin voltage has risen to the sufficient level by using the watch timer (the interval time is set to 3.91 ms). After that, the CPU operates on the fastest clock.

Figure 3-5. Example of Selecting CPU Clock after $\overline{\text{RESET}}$

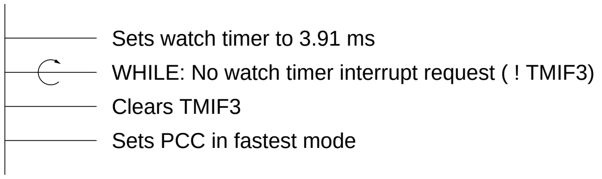
(1) μPD78014 subseries


Commercial power source
ON
OFF
4.5 V
V_{DD} pin voltage
2.7 V
H
$\overline{\text{RESET}}$ signal
L
CPU clock wait time
HALT status
7.63 μs
0.48 μs
31.3 ms
(2^{18}/fx: at 8.38 MHz)
3.9 ms
$\overline{\text{RESET}}$ signal is deasserted 10 μs after V_{DD} pin voltage has risen to 2.7 V or more. CPU clock oscillation starts.
V_{DD} pin voltage has risen to 4.5 V or more before contents of PCC are changed.

(2) μPD78018F subseries


Commercial power source
ON
OFF
3.5 V
V_{DD} pin voltage
1.8 V
H
$\overline{\text{RESET}}$ signal
L
CPU clock wait time
HALT status
7.63 μs
0.48 μs
31.3 ms
(2^{18}/fx: at 8.38 MHz)
3.9 ms
$\overline{\text{RESET}}$ signal is deasserted 10 μs after V_{DD} pin voltage has risen to 1.8 V or more. CPU clock oscillation starts.
V_{DD} pin voltage has risen to 3.5 V or more before contents of PCC are changed.

(1) SPD chart



Sets watch timer to 3.91 ms
WHILE: No watch timer interrupt request (! TMIF3)
Clears TMIF3
Sets PCC in fastest mode

(2) Program list

```
;**************************************
;*       Sets wait time
;**************************************
       TMC2=#00110110B            ;  Sets watch timer to 3.91 ms
       while_bit(!TMIF3)          ;  3.91 ms?
       endw
       CLR1   WTIF
       PCC=#00000000B             ;  Sets CPU clock in fastest mode
```

3.2 Selecting Power ON/OFF

The 78K/0 series can operate in an ultra low current consumption mode by using the processor clock control register (PCC) and selecting the subsystem clock. By providing a backup power supply such as a Ni-Cd battery or super capacitor to the system, therefore, the system can continue operating even if a power failure occurs.

In this example, a power failure is detected by using INTP1 (both the rising and falling edges are selected as the edge to be detected), and the contents of the PCC are changed depending on the port level at that time. Figure 3-6 shows a circuit example, and Figure 3-7 shows the system clock changing timing.

Figure 3-6. Example of System Clock Changing Circuit

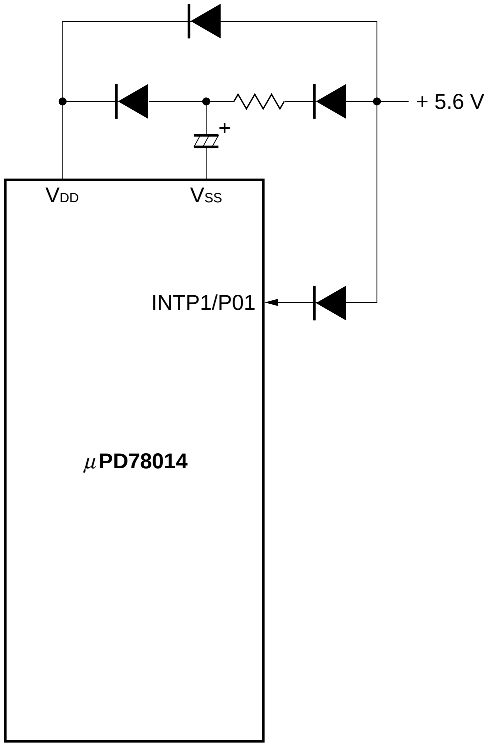
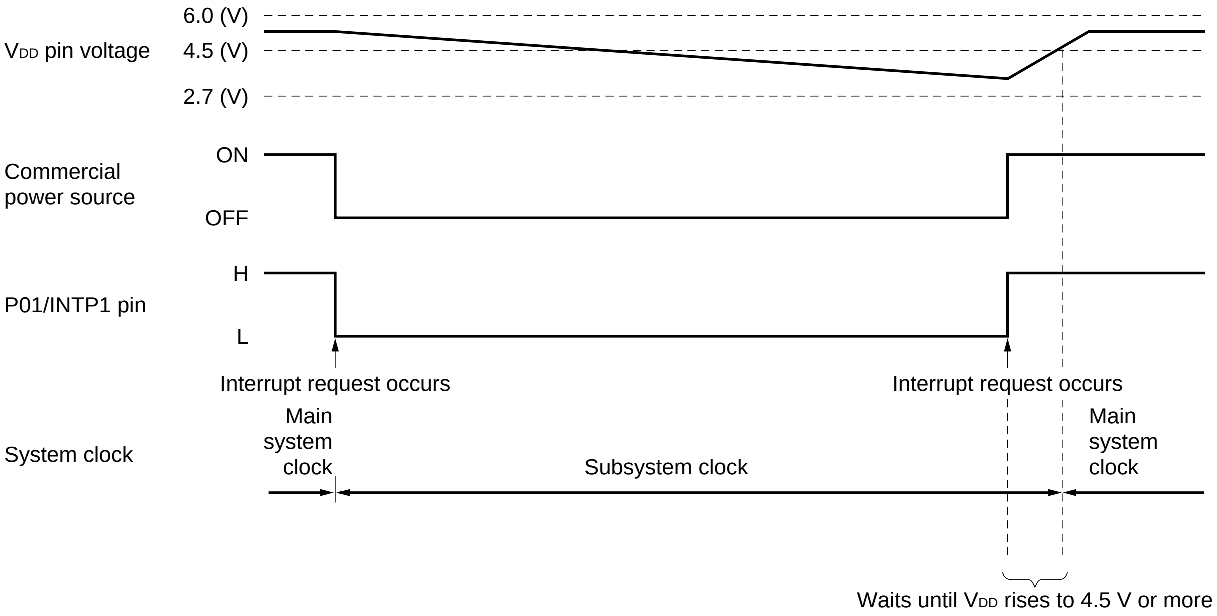
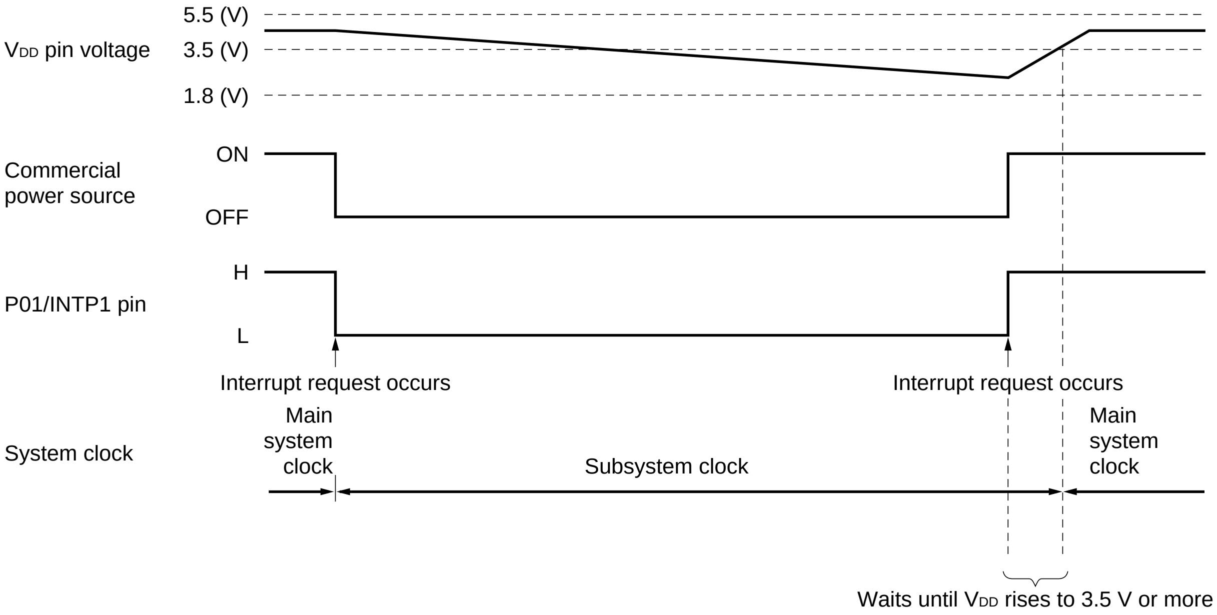


Figure 3-7. Example of Changing System Clock on Power ON/OFF

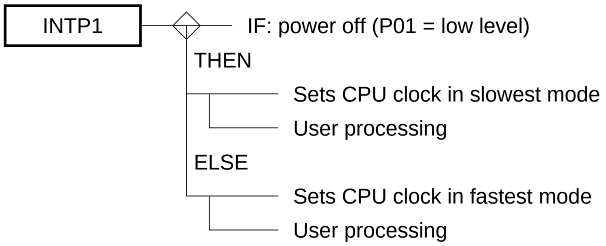
(1) μPD78014 subseries



(2) μPD78018F subseries



(1) SPD chart


INTP1
IF: power off (P01 = low level)
THEN
Sets CPU clock in slowest mode
User processing
ELSE
Sets CPU clock in fastest mode
User processing

(2) Program list

```
VEP0    CSEG    AT 08H
        DW      INTP1                   ; Sets vector address of INTP1

        MOV     INTM0,#00110000B        ; Both edge detection mode
        CLR1    PMK1
        EI

;******************************************
;*      Sets low-/high-speed mode
;******************************************
INTP1:
        if_bit(!P0.1)
;           Setting of internal hardware (low speed)
;           User processing

            PCC=#10010000B              ; Sets low-speed mode

        else
;           Sets internal hardware (high speed)
;           User processing

            PCC=#00000000B              ; Sets high-speed mode

    endif
    RETI
```

[MEMO]

CHAPTER 4 APPLICATIONS OF WATCHDOG TIMER

The watchdog timer of the 78K/0 series has two modes: watchdog timer mode in which a hang-up of the microcontroller is detected, and interval timer mode.

The watchdog timer is set by the following registers:

- Timer clock select register 2 (TCL2) : μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries, μPD780001

- Watchdog timer clock select regisrer (WDCS) : μPD780024, 780024Y, 780034, 780034Y, 780924, 780964 subseries

- Watchdog timer mode register (WDTM)

★ **Caution The format of the registers provided on the μPD780024, 780024Y, 780034, 780034Y, 780924, and 780964 subseries differs from the format of the registers used in the program examples in this chapter. When using a program example in this chapter with any of the μPD780024, 780024Y, 780034, 780034Y, 780924, and 780964 subseries, change the setting of the registers according to the registers of the microcontroller used.**

Figure 4-1. Format of Timer Clock Select Register 2 (μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL2	TCL27	TCL26	TCL25	TCL24	0	TCL22	TCL21	TCL20	FF42H	00H	R/W

TCL22	TCL21	TCL20	Selects count clock of watchdog timer
0	0	0	$f_X/2^4$ (625 kHz)
0	0	1	$f_X/2^5$ (313 kHz)
0	1	0	$f_X/2^6$ (156 kHz)
0	1	1	$f_X/2^7$ (78.1 kHz)
1	0	0	$f_X/2^8$ (39.1 kHz)
1	0	1	$f_X/2^9$ (19.5 kHz)
1	1	0	$f_X/2^{10}$ (9.8 kHz)
1	1	1	$f_X/2^{12}$ (2.4 kHz)

TCL24	Selects count clock of watch timer
0	$f_X/2^8$ (39.1 kHz)
1	f_{XT} (32.768 kHz)

TCL27	TCL26	TCL25	Selects frequency of buzzer output
0	×	×	Disables buzzer output
1	0	0	$f_X/2^{10}$ (9.8 kHz)
1	0	1	$f_X/2^{11}$ (4.9 kHz)
1	1	0	$f_X/2^{12}$ (2.4 kHz)
1	1	1	Setting prohibited

Caution To change the data of TCL2 except when writing the same data, once stop the timer operation.

Remarks 1. f_X : main system clock oscillation frequency

2. f_{XT} : subsystem clock oscillation frequency

3. × : don't care

4. ( ) : f_X = 10.0 MHz or f_{XT} = 32.768 kHz

★

Figure 4-2. Format of Timer Clock Select Register 2 (μPD780001)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL2	TCL27	TCL26	TCL25	0	0	TCL22	TCL21	TCL20	FF42H	00H	R/W

TCL22	TCL21	TCL20	Selects count clock of watchdog timer
0	0	0	$f_X/2^4$ (625 kHz)
0	0	1	$f_X/2^5$ (313 kHz)
0	1	0	$f_X/2^6$ (156 kHz)
0	1	1	$f_X/2^7$ (78.1 kHz)
1	0	0	$f_X/2^8$ (39.1 kHz)
1	0	1	$f_X/2^9$ (19.5 kHz)
1	1	0	$f_X/2^{10}$ (9.8 kHz)
1	1	1	$f_X/2^{12}$ (2.4 kHz)

TCL27	TCL26	TCL25	Selects frequency of buzzer output
0	×	×	Disables buzzer output
1	0	0	$f_X/2^{10}$ (9.8 kHz)
1	0	1	$f_X/2^{11}$ (4.9 kHz)
1	1	0	$f_X/2^{12}$ (2.4 kHz)
1	1	1	Setting prohibited

Caution To change the data of TCL2 except when writing the same data, once stop the timer operation.

Remarks 1. f_X : main system clock oscillation frequency

2. × : don’t care

3. ( ) : f_X = 10.0 MHz

★

Figure 4-3. Format of Watchdog Timer Clock Select Register (μPD780024, 780024Y, 780034, 780034Y, 780924, 780964 subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
WDCS	0	0	0	0	0	WDCS2	WDCS1	WDCS0	FF42H	00H	R/W

WDCS2	WDCS1	WDCS0	Overflow time of watchdog timer/interval timer
0	0	0	$2^{12}/f_X$ (489 μs)
0	0	1	$2^{13}/f_X$ (978 μs)
0	1	0	$2^{14}/f_X$ (1.96 ms)
0	1	1	$2^{14}/f_X$ (3.91 ms)
1	0	0	$2^{16}/f_X$ (7.82 ms)
1	0	1	$2^{17}/f_X$ (15.6 ms)
1	1	0	$2^{18}/f_X$ (31.3 ms)
1	1	1	$2^{20}/f_X$ (125 ms)

Remarks 1. f_{XX} : main system clock oscillation frequency

2. ( ) : f_X = 8.38 MHz

Figure 4-4. Format of Watchdog Timer Mode Register (μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries, μPD780001)

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

WDTM4	WDTM3	Selects operation mode of watchdog timer[Note 1]
0	×	Interval timer mode (maskable interrupt[Note 2] request occurs when overflow occurs)
1	0	Watchdog timer mode 1 (non-maskable interrupt request occurs when overflow occurs)
1	1	Watchdog timer mode 2 (reset operation starts when overflow occurs)

RUN	Selects watchdog timer operation[Note 3]
0	Stops counting
1	Clears counter and starts counting

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

2. When RUN is set to 1, the WDTM starts interval timer operation.

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

Cautions 1. When RUN is set to 1 and the watchdog timer is cleared, the actual overflow time is up to 0.5% shorter than the time set by the timer clock select register 2 (TCL2).

2. When using the watchdog timer modes 1 and 2, confirm that the interrupt request flag (TMIF4) is 0 and then set WDTM4 to 1.

If WDTM4 is set to 1 while TMIF4 is 1, the non-maskable interrupt occurs regardless of the contents of WDTM3.

Remark × : don't care

★ **Figure 4-5. Format of Watchdog Timer Mode Register**
(μPD780024, 780024Y, 780034, 780034Y subseries)

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

WDTM4	WDTM3	Selects operation mode of watchdog timer[Note 1]
0	×	Interval timer mode (maskable interrupt request occurs when overflow occurs)
1	0	Watchdog timer mode 1 (non-maskable interrupt request occurs when overflow occurs)
1	1	Watchdog timer mode 2 (reset operation starts when overflow occurs)

RUN	Selects watchdog timer operation[Note 2]
0	Stops counting
1	Clears counter and starts counting

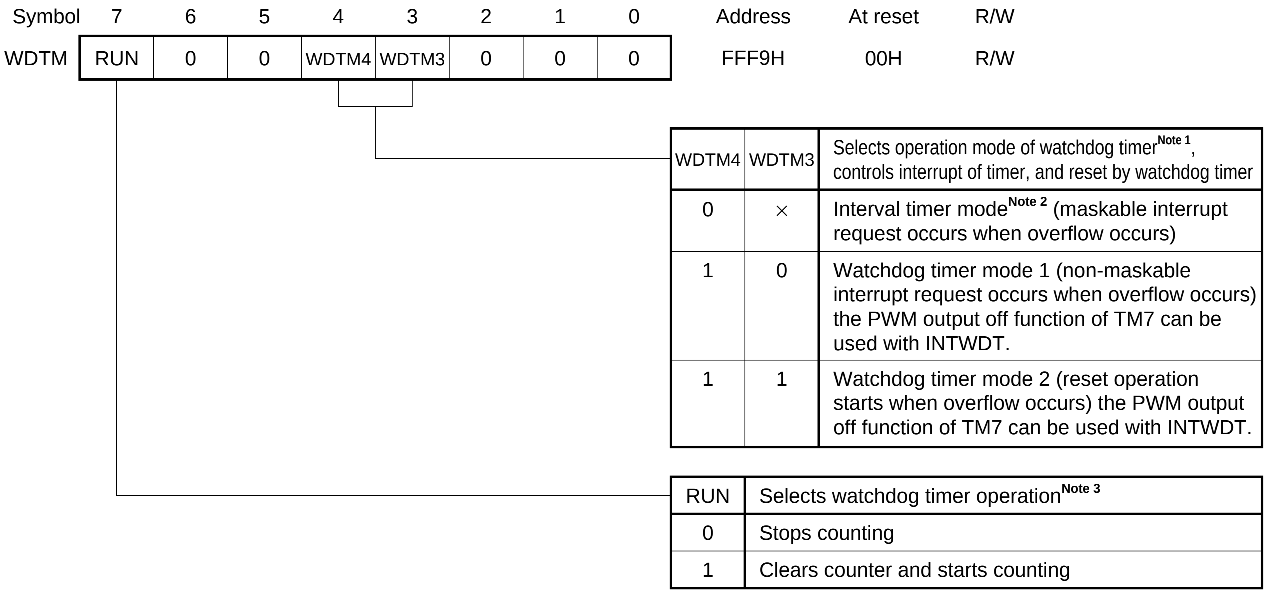
Notes 1. Once WDTM3 and WDTM4 have been set to 1, they cannot be cleared to 0 by software.

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

Caution When RUN is set to 1 and the watchdog timer is cleared, the actual overflow time is up to 0.5% shorter than the time set by the watchdog timer clock select register (WDCS).

Remark × : don't care

★ **Figure 4-6. Format of Watchdog Timer Mode Register (μPD780924, 780964 subseries)**



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

WDTM4	WDTM3	Selects operation mode of watchdog timer[Note 1], controls interrupt of timer, and reset by watchdog timer
0	×	Interval timer mode[Note 2] (maskable interrupt request occurs when overflow occurs)
1	0	Watchdog timer mode 1 (non-maskable interrupt request occurs when overflow occurs) the PWM output off function of TM7 can be used with INTWDT.
1	1	Watchdog timer mode 2 (reset operation starts when overflow occurs) the PWM output off function of TM7 can be used with INTWDT.

RUN	Selects watchdog timer operation[Note 3]
0	Stops counting
1	Clears counter and starts counting

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

2. When RUN is set to 1, the WDTM starts interval timer operation.

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

Caution When RUN is set to 1 and the watchdog timer is cleared, the actual overflow time is up to 0.5% shorter than the time set by the watchdog timer clock select register (WDCS).

Remark × : don’t care

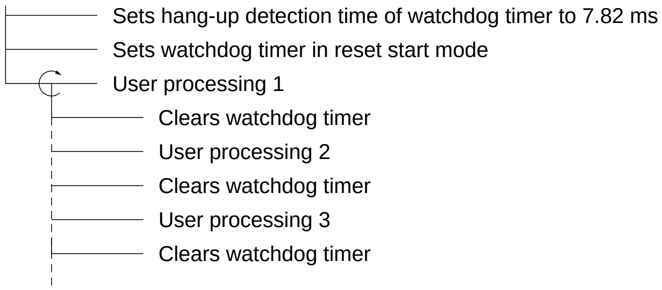
4.1 Setting Watchdog Timer Mode

Reset processing or non-maskable interrupt processing is performed after the watchdog timer has detected a hang-up. You can select which processing is to be performed by the watchdog timer mode register (WDTM). When the watchdog timer mode is used, the timer must be cleared at intervals shorter than the set hang-up detection time. If the timer is not cleared, an overflow occurs, and reset or interrupt processing is executed.

The hang-up detection time of the watchdog timer is set by the timer clock select register 2 (TCL2).

In the following example, the hang-up detection time is set to 7.82 ms and the reset processing is performed when an overflow occurs.

(1) SPD chart



(2) Program list

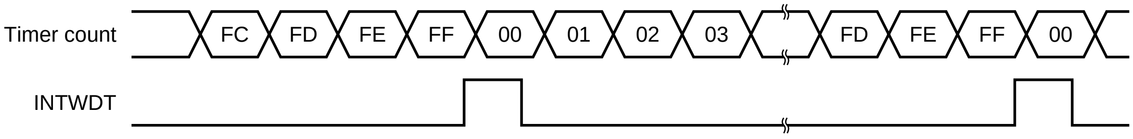
```
;************************************
;*   Sets watchdog timer
;************************************

       TCL2=#00000100B          ; Sets watchdog timer to 7.82 ms
       WDTM=#10011000B          ; Sets reset start mode
;          :
       User processing 1
           :
       SET1   RUN               ; Clears timer
;          :
       User processing 2
           :
       SET1   RUN               ; Clears timer
;          :
       User processing 3
           :
       SET1   RUN               ; Clears timer
           :
```

4.2 Setting Interval Timer Mode

When the interval timer mode is used, the interval time is set by the timer clock select register 2 (TCL2) (interval time = 0.488 ms to 125 ms). In this mode, an interrupt request flag (TMIF4) is set when an overflow occurs in the timer.

In the following example, three types of times, 0.977 ms, 7.82 ms, and 125 ms, are set.

Figure 4-7. Count Timing of Watchdog Timer



(1) Program list

```
<1>  To set 0.977 ms
     TCL2=#00000001B       ;  Sets 0.977 ms
     WDTM=#10001000B       ;  Selects interval timer mode

<2>  To set 7.82 ms
     TCL2=#00000100B       ;  Sets 7.82 ms
     WDTM=#10001000B       ;  Selects interval timer mode

<3>  To set 125 ms
     TCL2=#00000111B       ;  Sets 125 ms
     WDTM=#10001000B       ;  Selects interval timer mode
```

[MEMO]

CHAPTER 5 APPLICATIONS OF 16-BIT TIMER/EVENT COUNTER

The 16-bit timer/event counter of the 78K/0 series has the following six functions:

- Interval timer
- PWM output (μPD78014, 78014Y, 78018F, 78018FY, and 78014H subseries only)
- Pulse width measurement
- External event counter
- Square wave output
- One-shot pulse output (μPD780024, 780024Y, 780034, and 780034Y subseries only)

The 16-bit timer/event counter is set by the following registers:

<μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries>

- Timer clock select register 0 (TCL0)
- 16-bit timer mode control register (TMC0)
- Capture/compare control register 0 (CRC0)
- 16-bit timer output control register (TOC0)
- Port mode register 3 (PM3)
- External interrupt mode register (INTM0)
- Sampling clock select register (SCS)

<μPD780024, 780024Y, 780034, 780034Y subseries>

- 16-bit timer mode control register (TMC0)
- Capture/compare control register 0 (CRC0)
- 16-bit timer output control register (TOC0)
- Prescaler mode register (PRM0)
- Port mode register 7 (PM7)

★ **Caution The format of the registers provided on the μPD780024, 780024Y, 780034, and 780034Y subseries differs from the format of the registers used in the program examples in this chapter. When using a program example in this chapter with any of the μPD780024, 780024Y, 780034, and 780034Y subseries, change the setting of the registers according to the registers of the microcontroller used.**

Figure 5-1. Format of Timer Clock Select Register 0 (μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL0	CLOE	TCL06	TCL05	TCL04	TCL03	TCL02	TCL01	TCL00	FF40H	00H	R/W

TCL03	TCL02	TCL01	TCL00	Selects clock of PCL output
0	0	0	0	f_{XT} (32.768 kHz)
0	1	1	1	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^7$ (78.1 kHz)
1	1	0	0	$f_X/2^8$ (39.1 kHz)
Others				Setting prohibited

TCL06	TCL05	TCL04	Selects count clock of 16-bit timer register
0	0	0	TI0 (valid edge can be specified)
0	1	0	$f_X/2$ (5.0 MHz)
0	1	1	$f_X/2^2$ (2.5 MHz)
1	0	0	$f_X/2^3$ (1.25 MHz)
Others			Setting prohibited

CLOE	Controls PCL output
0	Disables output
1	Enables output

Cautions 1. Set the valid edge of the TI0/INTP0 pin by the external interrupt mode register (INTM0). The frequency of the sampling clock is selected by the sampling clock select register (SCS).

2. To enable PCL output, set TCL00 through TCL03, and then set CLOE to 1 by using a 1-bit memory manipulation instruction.

3. Read the count value from TM0, not from the capture/compare register 01(CR01), when TI0 is specified as the count clock of TM0.

4. Before writing new data to TCL0, stop the timer operation once.

Remarks 1. f_X : main system clock oscillation frequency

2. f_{XT} : subsystem clock oscillation frequency

3. TI0 : input pin of 16-bit timer/event counter

4. TM0 : 16-bit timer register

5. ( ) : at f_X = 10.0 MHz or f_{XT} = 32.768 kHz

Figure 5-2. Format of 16-Bit Timer Mode Control Register (μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TMC0	0	0	0	0	TMC03	TMC02	TMC01	OVF0	FF48H	00H	R/W

OVF0	Detects overflow of 16-bit timer register
0	Overflow does not occur
1	Overflow occurs

TMC03	TMC02	TMC01	Selects operation mode and clear mode	Selects output timing of TO0	Occurrence of interrupt request
0	0	0	Stops operation (clears TM0 to 0)	Not affected	Does not occur
0	0	1	PWM mode (free running)	PWM pulse output	Occurs if TM0 and CR00 coincide
0	1	0	Free running mode	Coincidence between TM0 and CR00	
0	1	1		Coincidence between TM0 and CR00, or valid edge of TI0	
1	0	0	Clears and starts at valid edge of TI0	Coincidence between TM0 and CR00	
1	0	1		Coincidence between TM0 and CR00 or valid edge of TI0	
1	1	0	Clears and start at coincidence between TM0 and CR00	Coincidence between TM0 and CR00	
1	1	1		Coincidence between TM0 and CR00 or valid edge of TI0	

Cautions 1. Before setting the clear mode or changing the output timing of TO0, stop the timer operation (by clearing TMC01 through TMC03 to 0, 0, 0).

2. Set the valid edge of the TI0/INTP0 pin by the external interrupt mode register (INTM0). The frequency of the sampling clock is selected by the sampling clock select register (SCS).

3. When using the PWM mode, set data to CR00 after setting the PWM mode.

4. When a mode in which the timer is cleared and started on coincidence between TM0 and CR00, the OVF0 flag is set to 1 when the set value of CR00 is FFFFH and the value of TM0 changes from FFFFH to 0000H.

Remarks 1. TO0 : output pin of 16-bit timer/event counter
2. TI0 : input pin of 16-bit timer/event counter
3. TM0 : 16-bit timer register
4. CR00 : compare register 00

★ **Figure 5-3. Format of 16-Bit Timer Mode Control Register**
(μPD780024, 780024Y, 780034, 780034Y subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TMC0	0	0	0	0	TMC03	TMC02	TMC01	OVF0	FF60H	00H	R/W

OVF0	Detects overflow of 16-bit timer register
0	Overflow does not occur
1	Overflow occurs

TMC03	TMC02	TMC01	Selects operation mode and clear mode	Selects output timing of TO0	Occurrence of interrupt request
0	0	0	Stops operation (clears TM0 to 0)	Not affected	Does not occur
0	0	1			
0	1	0	Free running mode	Coincidence between TM0 and CR00 or between TM0 and CR01	Occurs if TM0 and CR00 coincide and if TM0 and CR01 coincide
0	1	1		Coincidence between TM0 and CR00, or between TM0 and CR01, or valid edge of TI00	
1	0	0	Clears and starts at valid edge of TI00	Coincidence between TM0 and CR00 or between TM0 and CR01	
1	0	1		Coincidence between TM0 and CR00, or between TM0 and CR01, or valid edge of TI00	
1	1	0	Clears and start at coincidence between TM0 and CR00	Coincidence between TM0 and CR00 or between TM0 and CR01	
1	1	1		Coincidence between TM0 and CR00, or between TM0 and CR01, or valid edge of TI00	

Cautions 1. Before setting the clear mode or changing the output timing of TO0, stop the timer operation (by clearing TMC02 and TMC03 to 0, 0).

2. Set the valid edge of the TI00/TO0/P70 pin by the prescaler mode register 0 (PRM0). The frequency of the sampling clock is selected by the sampling clock select register (SCS).

3. When a mode in which the timer is cleared and started on coincidence between TM0 and CR00, the OVF0 flag is set to 1 when the set value of CR00 is FFFFH and the value of TM0 changes from FFFFH to 0000H.

Remarks 1. TO0 : output pin of 16-bit timer/event counter

2. TI00 : input pin of 16-bit timer/event counter

3. TM0 : 16-bit timer register

4. CR00 : compare register 00

5. CR01 : compare regisrer 01

Figure 5-4. Format of 16-Bit Timer Output Control Register (μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TOC0	0	0	0	0	LVS0	LVR0	TOC01	TOE0	FF4EH	00H	R/W

TOE0	Controls output of 16-bit timer/event counter
0	Disables output (port mode)
1	Enables output

TOC01	PWM mode	Other than PWM mode
	Selects active level	Controls timer output F/F
0	Active high	Disables reverse operation
1	Active low	Enables reverse operation

LVS0	LVR0	Sets status of timer output F/F of 16-bit timer/ event counter
0	0	Not affected
0	1	Resets timer output F/F (to 0)
1	0	Sets timer output F/F (to 1)
1	1	Setting prohibited

Cautions 1. Be sure to stop the timer operation before setting TOC0.

2. LVS0 and LVR0 are always 0 when they are read immediately after data has been set.

★ **Figure 5-5. Format of 16-Bit Timer Output Control Register (μPD780024, 780024Y, 780034, 780034Y subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TOC0	0	OSPT	OSPE	TOC04	LVS0	LVR0	TOC01	TOE0	FF63H	00H	R/W

TOE0	Controls output of timer 0
0	Disables output (port mode)
1	Enables output

TOC01	Controls timer output F/F on coincidence between CR00 and TM0
0	Disables reverse operation
1	Enables reverse operation

LVS0	LVR0	Sets status of timer output F/F of timer 0
0	0	Not affected
0	1	Resets timer output F/F (to 0)
1	0	Sets timer output F/F (to 1)
1	1	Setting prohibited

TOC04	Controls timer output F/F on coincidence between CR01 and TM0
0	Disables reverse operation
1	Enables reverse operation

OSPE	Controls one-shot pulse output operation
0	Successive pulse output
1	One-shot pulse output

OSPT	Controls output trigger of one-shot pulse by software
0	No one-shot pulse trigger
1	One-shot pulse trigger

Cautions 1. Be sure to stop the timer operation before setting TOC0 (except OSPT).

2. LVS0 and LVR0 are always 0 when they are read immediately after data has been set.

3. OSPT is automatically cleared after data has been set. It is therefore always 0 when read.

Figure 5-6. Format of Port Mode Register 3 (μPD78014, 78014Y, 78018, 78018FY, 78014H subseries)

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

PM3n	Selects input/output mode of P3n pin (n = 0-7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

Caution When using the P30/TO0 pin as a timer output, set the output latch of PM30 and P30 to 0.

Figure 5-7. Format of External Interrupt Mode Register (μPD78014, 78014Y, 78018, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
INTM0	ES31	ES30	ES21	ES20	ES11	ES10	0	0	FFECH	00H	R/W

ES11	ES10	Selects valid edge of INTP0
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES21	ES20	Selects valid edge of INTP1
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES31	ES30	Selects valid edge of INTP2
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

★ **Caution Before setting the valid edge of the INTP0 pin, clear bits 1 through 3 (TMC01 through TMC03) of the 16-bit timer mode control register (TMC0) to 0, 0, 0, and stop the timer.**

Figure 5-8. Format of Sampling Clock Select Register
(μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
SCS	0	0	0	0	0	0	SCS1	SCS0	FF47H	00H	R/W

SCS1	SCS0	Selects sampling clock of INTP0
0	0	$f_X/2^{N+1}$
0	1	Setting prohibited
1	0	$f_X/2^6$ (156 kHz)
1	1	$f_X/2^7$ (78.1 kHz)

Caution **$f_X/2^{N+1}$ is the clock supplied to the CPU, and $f_X/2^6$ and $f_X/2^7$ are the clocks supplied to the peripheral hardware. $f_X/2^{N+1}$ is stopped in the HALT mode.**

Remarks 1. N : Value (N = 0 to 4) set to the bits 0 through 2 (PCC0 through PCC2) of the processor clock control register (PCC)

2. f_X : main system clock oscillation frequency

3. ( ) : at f_X = 10.0 MHz

★ **Figure 5-9. Format of Capture/Compare Control Register 0**
(μPD780024, 780024Y, 780034, 780034Y subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
CRC0	0	0	0	0	0	CRC02	CRC01	CRC00	FF62H	04H	R/W

CRC00	Selects operation mode of CR00
0	Operates as compare register
1	Operates as capture register

CRC01	Selects capture trigger of CR00
0	Captures at valid edge of TI01
1	Captures at valid edge of TI00

CRC02	Selects operation mode of CR01
0	Operates as compare register
1	Operates as capture register

Cautions 1. Be sure to stop the timer operation before setting CRC0.

2. When a mode in which the timer is cleared and started on coincidence between TM0 and CR00 is selected by the 16-bit timer mode control register (TMC0), do not specify CR00 as the capture register.

★

**Figure 5-10. Format of Prescaler Mode Register 0
(μPD780024, 780024Y, 780034, 780034Y subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
PRM0	ES11	ES10	ES01	ES10	0	0	PRM01	PRM00	FF61H	00H	R/W

PRM01	PRM00	Selects count clock
0	0	f_x (8.38 MHz)
0	1	$f_x/2^2$ (2.09 MHz)
1	0	$f_x/2^6$ (131 kHz)
1	1	TI00 valid edge

ES01	ES00	Selects valid edge of TI00
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES11	ES10	Selects valid edge of TI01
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

Caution When setting the valid edge of TI00 for the count clock, do not set clear & start mode by the valid edge of TI00, and do not set TI00 for the capture trigger.

Remarks 1. f_x : main system clock oscillation frequency

2. TI00, TI01 : input pin of 16-bit timer/event counter

3. ( ) : at f_x = 8.38 MHz

★

**Figure 5-11. Format of Port Mode Register 7
(μPD780024, 780024Y, 780034, 780034Y subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
PM7	1	1	PM75	PM74	PM73	PM72	PM71	PM70	FF27H	FFH	R/W

PM7n	Selects input/output mode of P7n pin (n = 0-5)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

5.1 Setting of Interval Timer

To set the 16-bit timer/event counter as an interval timer, first set the timer clock select register 0 (TCL0) and the 16-bit timer mode control register (TMC0). The clear mode of the 16-bit timer is set by TMC0 and the interval time is set by TCL0.

After that, set the value of the compare register (CR00) from the setup time and count clock. Determine the setup time by using the following expression:

Setup time = (Compare register value + 1) × Count clock cycle

This section shows two examples of setup times of the interval timer: 10 ms and 50 ms.

(a) Interval of 10 ms

<1> Setting of TMC0

Selects a mode in which the timer is cleared and started on coincidence between TM0 and CR00.

<2> Setting of TCL0

Select the $f_X/2$ mode in which an interval time of 10 ms or more can be set and the resolution is the highest.

<3> Setting of CR00

$$10\text{ ms} = (N + 1) \times \frac{1}{8.38\text{ MHz}/2}$$

$$N = 10\text{ ms} \times 8.38\text{ MHz}/2 - 1 \fallingdotseq 4.1899$$

(1) Program list

```
CR00 = #41899
TCL0 = #00100000B  ;  Selects count clock fX/2
TMC0 = #00001100B ;  Clears and starts 16-bit timer/event counter when TM0 and CR00 coincide
```

(b) Interval of 50 ms

<1> Setting of TMC0

Selects a mode in which the timer is cleared and started on coincidence between TM0 and CR00.

<2> Setting of TCL0

Select the $f_X/2^3$ mode in which an interval time of 50 ms or more can be set and the resolution is the highest.

<3> Setting of CR00

$$50\text{ ms} = (N + 1) \times \frac{1}{8.38\text{ MHz}/2^3}$$

$$N = 50\text{ ms} \times 8.38\text{ MHz}/2^3 - 1 \fallingdotseq 52374$$

(1) Program list

```
CR00 = #52374
TCL0 = #01000000B  ;  Selects count clock fX/2^3
TMC0 = #00001100B ;  Clears and starts 16-bit timer/event counter when TM0 and CR00 coincide
```

5.2 PWM Output

When using the 16-bit timer/event counter in the PWM output mode, set the PWM mode by the 16-bit timer mode control register (TMC0) and enables the output of the 16-bit timer/event counter by the 16-bit timer output control register (TOC0).

The pulse width (active level) of PWM is determined by the value set to the compare register 00 (CR00). Because the PWM of the 78K/0 series has a resolution of 14 bits, however, bits 2 through 15 of CR00 are valid (clear bits 0 and 1 of CR00 to ‘0, 0’).

In the example below, the basic cycle of the PWM mode is set to 61.0 μs ($2/f_X \times 2^8$) and the low level is selected as the active level. The high-order 4 bits of the pulse width are rewritten depending on the value of the parameter (00H to FFH). Therefore, in the following application example, PWM output can be performed in 16 steps (CR00 = 0FFCH to FFFCH).

(1) Description of package

<Public declaration symbol>

PWM	:	PWM output subroutine name
PWMOUT	:	input parameter of PWM active level

<Registers used>

AX

<RAM used>

Name	Usage	Attribute	Bytes
PWMOUT	Sets PWM active level	SADDR	1

<Nesting>

1 level 2 bytes

<Hardware units used>

- 16-bit timer/event counter
- P30/TO0

<Initial setting>

- Setting of 16-bit timer/event counter

PWM output mode	TMC0 = #00000010B
PWM basic cycle: 61.0 μs	TCL0 = #00100000B
Low-active output	TOC0 = #00000011B
• P30 output mode	PM30 = 0
• P30 output latch	P30 = 0

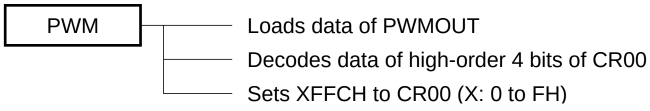
<Starting>

After setting data to PWMOUT in RAM, call subroutine PWM.

(2) Example of use

```
EXTRN     PWM, PWMOUT
   ⋮
TOC0 = #00000011B       ;  Sets low-active PWM output
TCL0 = #00100000B       ;  Selects count clock fx/2
TMC0 = #00000010B       ;  Sets PWM mode
   ⋮
PWMOUT = A              ;  Sets input parameter of active level
CALL      !PWM
```

(3) SPD chart


PWM
Loads data of PWMOUT
Decodes data of high-order 4 bits of CR00
Sets XFFCH to CR00 (X: 0 to FH)

(4) Program list

```
        PUBLIC  PWM,PWMOUT
PWM_DAT DSEG    SADDR
PWMOUT: DS      1                          ;  PWM output data area (0-15)
;************************************
;*      PWM output (16 steps)
;************************************
P0_SEG  CSEG
PWM:
        A=PWMOUT                           ;  Loads high-order data of PWMOUT
        A<<=1
        A<<=1
        A<<=1
        A<<=1
        A|=#0FH                            ;  Sets low-order 12 bits to 0FFCH
        X=#0FCH
        CR00=AX
        RET
```

5.3 Remote Controller Signal Reception

This section introduces two examples of programs of the μPD78014 subseries that receives signals from a remote controller by using the 16-bit timer/event counter.

- The counter is cleared each time the valid edge of the remote controller signal has been detected, and measures a pulse width from the timer count value (capture register CR01) when the next valid edge has been detected.
- The timer operates in the free running mode to measure a pulse width from the difference of the counter between valid edges. PWM output is also performed at the same time.

The remote controller signal is received by a PIN receiver diode and is input to the P00/INTP0 pin via receive amplifier μPC1490. Figure 5-12 shows an example of a remote controller signal receiver circuit, and Figure 5-13 shows the format of the remote controller signal.

Figure 5-12. Example of Remote Controller Signal Receiver Circuit

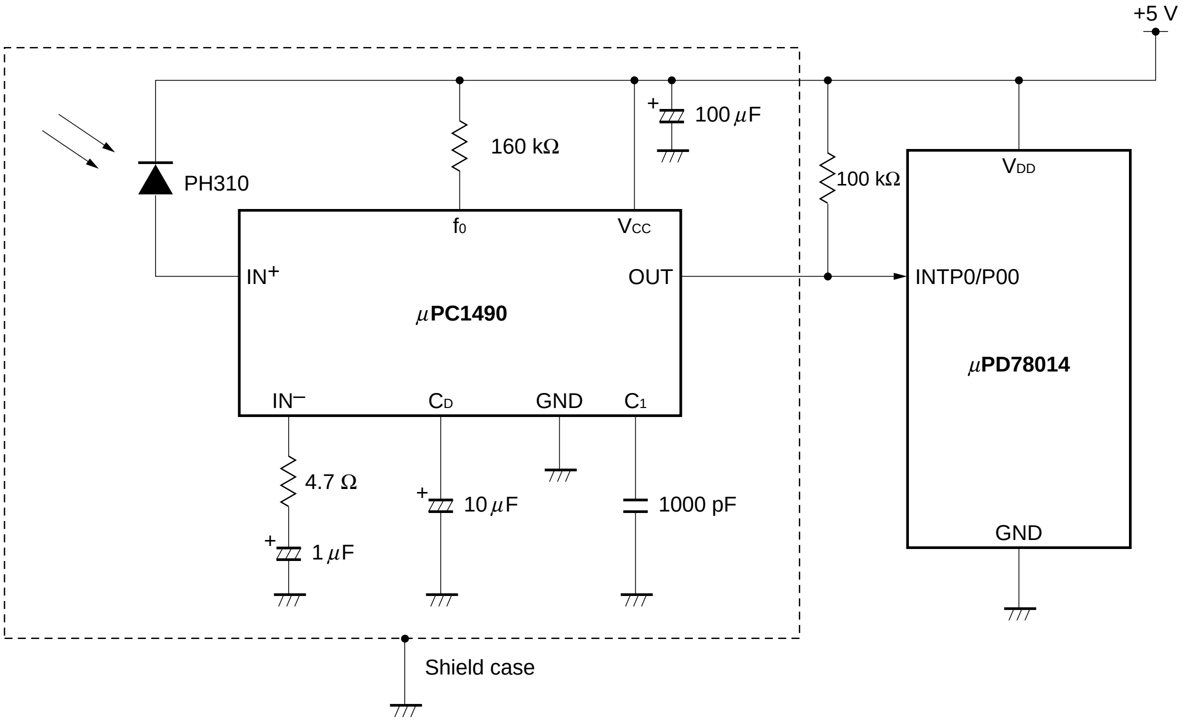
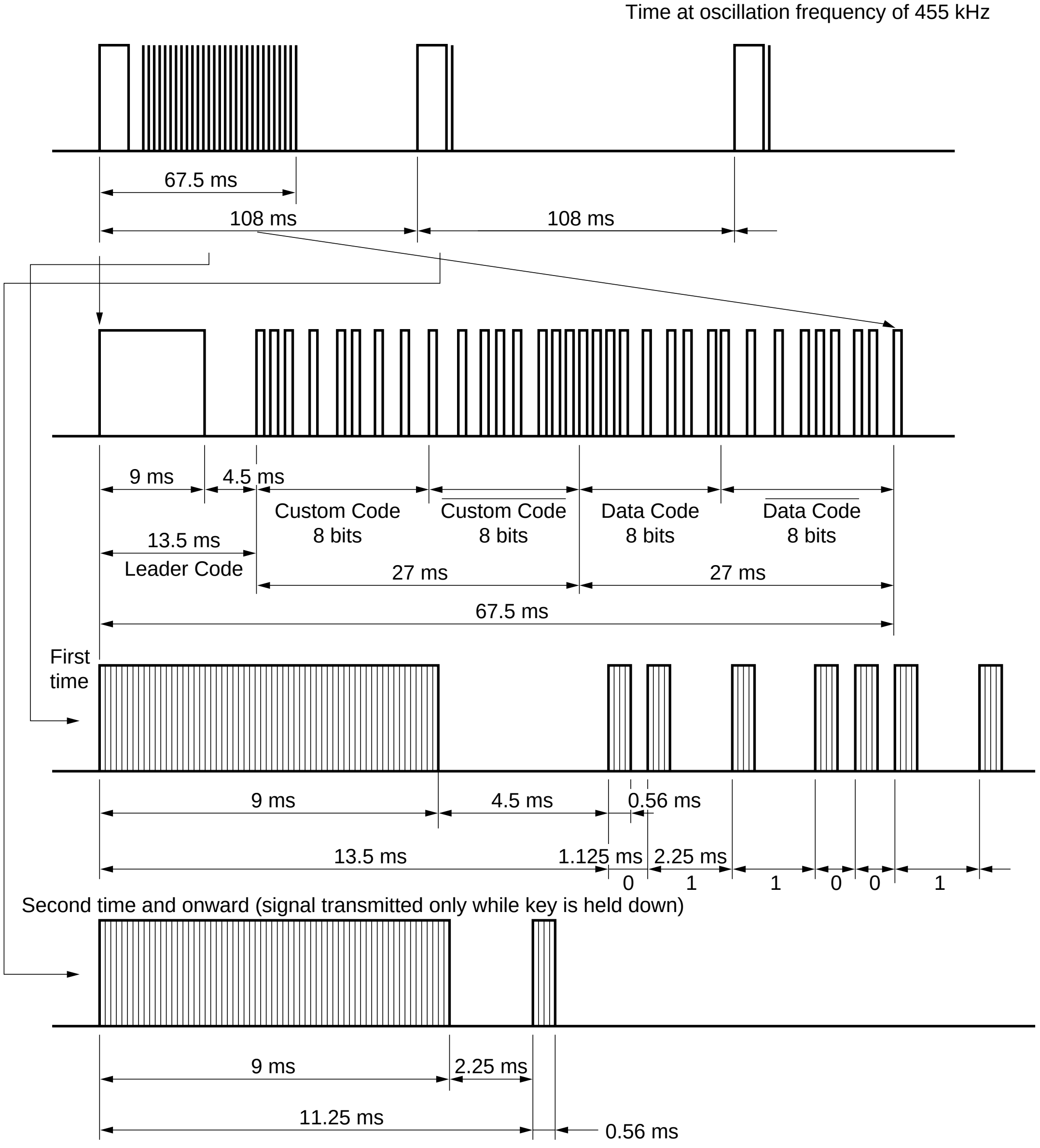
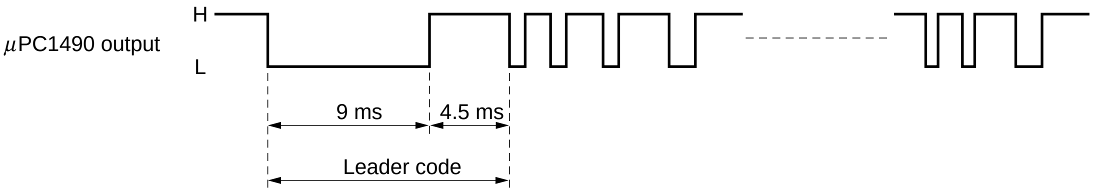


Figure 5-13. Remote Controller Signal Transmitter IC Output Signal



Because the receiver preamplifier μPC1490 used in the circuit example on the previous page is low-active, the level input to the μPD78014 subseries is the inverted data of the remote controller transmit data.

Figure 5-14. Output Signal of Receiver Preamplifier



5.3.1 Remote controller signal reception by counter clearing

Table 5-1 shows the valid pulse width for receiving a remote controller signal in the program example shown in this section, and <1> through <6> describes how to process each signal. The repeat signal of the remote controller signal is valid only within 250 ms after a valid signal has been input. If a signal input within 3 ms after the normal data has been loaded, the data is invalid.

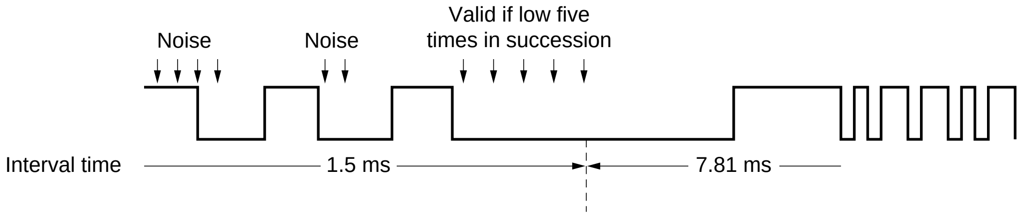
Table 5-1. Valid Time of Input Signal

Signal Name		Output Time	Valid Time
Leader code (low)		9 ms	6.8 ms-11.8 ms
Leader code (high)	Normal	4.5 ms	3 ms-5 ms
	Repeat	2.25 ms	1.8 ms-3 ms
Custom/data code	0	1.125 ms	0.5 ms-1.8 ms
	1	2.25 ms	1.8 ms-2.5 ms

<1> Leader code (low)

The interval time of the 16-bit timer/event counter is set to 1.5 ms, and the port level is sampled by means of interrupt processing. When five low levels have been detected in succession, these low levels are identified as a leader code, and the interval time is changed to 7.81 ms. After that, the pulse width of the low level of the leader code is measured by using rising-edge interrupt request INTP0.

Figure 5-15. Sampling of Remote Controller Signal



<2> Leader code (high)

The pulse width while the leader code is high is measured by using the falling-edge interrupt request INTP0 and the count value of the timer.

<3> Custom/data code

The pulse width of each 1 bit (1 cycle) is measured by using the falling-edge interrupt request INTP0. After the data of the 32nd bit has been loaded, the system tests if the inverted data and custom code coincide. It also checks that there is no data in the 33rd bit.

<4> Repeat code detection

When the high level of the leader code is less than 3 ms, the pulse width from output of the leader code to the rising edge of the INTP0 is measured.

<5> Valid period of repeat code

After the valid data has been input, sampling is performed by the interrupt processing (1.5 ms interval) of the 16-bit timer/event counter to measure the valid time of the repeat code of 250 ms.

<6> Time out during pulse width measurement

If the interrupt request of the 16-bit timer/event counter (7.81 ms) occurs during pulse width measurement, it is judged to be time out, and the data is invalid.

(1) Description of package

<Public declaration symbol>

RMDATA	:	Stores remote controller receive data
RPT	:	Repeat valid period identification flag
IPDTFG	:	Valid data identification flag
RMDTOK	:	Input signal validity identification flag
RMDTSET	:	Input signal identification flag

<Registers used>

Bank 0: AX, BC, HL

<RAM used>

Name	Usage	Attribute	Bytes
RPTCT	Repeat code valid time counter	SADDR	1
RMENDCT	No-input time counter after data input		
SELMOD	Mode selection		
LD_CT	Leader signal detection counter		
RMDATA	Valid data storage area		
WORKP	Input signal storage area	SADDRP	4

<Flags used>

Name	Usage
IPDTFG	Presence/absence of valid data
RMDTOK	Validity of input signal
RMDTSET	Presence/absence of input signal
RPT	Judgment whether repeat valid period elapsed

<Nesting>

5 levels 12 bytes

<Hardware used>

- 16-bit timer/event counter
- P00/INTP0

<Initial setting>

- Setting of 16-bit timer/event counter

Clears timer on coincidence between TM0 and CR00	TMC0 = #00001100B
Count clock fx/2	TCL0 = #00100000B
Compare register 00	CR00 = #6290

- INTP0 sampling clock fx/2^7 SCS = #00000011B
- INTP0 high-priority interrupt PPR0 = 0
- Enables 16-bit timer/event counter interrupt TMMK0 = 0
- Defines custom code to be CSTM and declares PUBLIC
- RAM clear

<Starting>

Started by INTP0 and INTTM0 interrupt requests

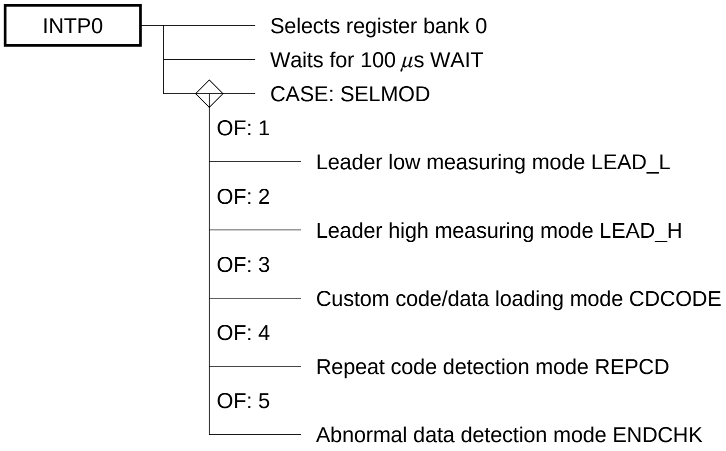
(2) Example of use

```
        PUBLIC  CSTM
        EXTRN   RMDATA,RPTCT
        EXTBIT  RPT,RMDTSET,IPDTFG

CSTM    EQU     9DH                     ; Remote controller custom code

        CR00=#6290
        TCL0=#00100000B                 ; Sets 1.5 ms
        TMC0=#00001100B
        SCS=#00000011B                  ; fX/128 as INTP0 sampling clock

        CLR1    PPR0                    ; INTP0 with high priority
        CLR1    RPT                     ; Clears flag
        CLR1    IPDTFG
        CLR1    RMDTSET

        CLR1    TMMK0                   ; Enables timer interrupt
        EI

DT_TEST:
        if_bit(RMDTSET)
            CLR1    RMDTSET
            if_bit(RPT)
;
;               Repeat processing
;
            else
;
;               Processing when there is input
;
            endif
        else
            if_bit(!RPT)
;
;               Processing when there is no input
;
            endif
        endif
```

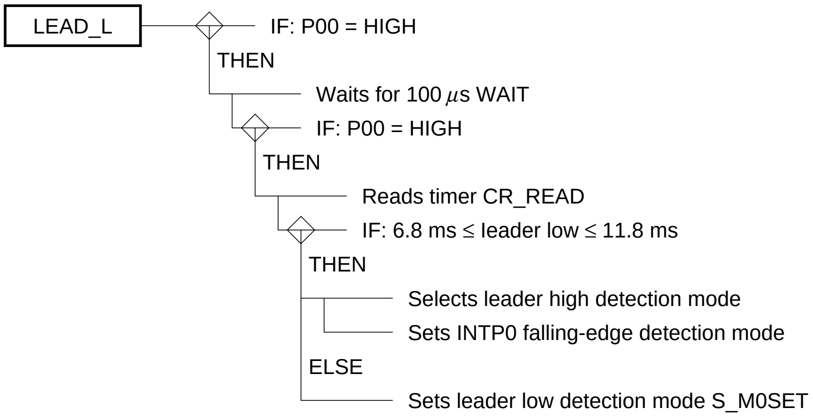
(3) SPD chart

```
INTTM00
    Selects register bank 1
    Enables master interrupt
    IF: input signal exists (IPDTFG)
    THEN
        IF: valid data exists (RMDTOK)
        THEN
            IF: no input within repeat valid time (250 ms)
            THEN
                Invalidates repeat code
            ELSE    Clears RPT, IPDTFG, and RMDTOK
                Counts repeat valid time
            Counts leader low time S_LOWCT
        ELSE
            IF: No input after end of data input (within 4.5 ms)
            THEN
                Valid data exists
                    Sets RMDTOK and RMDTSET
                Sets leader low detection mode S_M0SET
            Initializes leader low detection counter
    ELSE
        Counts leader low time S_LOWCT
```

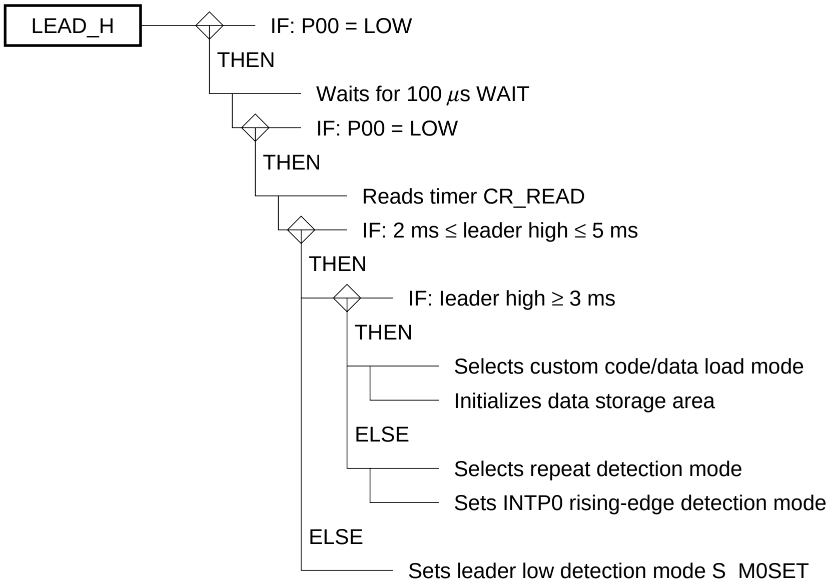
```
S_LOWCT
    IF: Leader low detection mode
    THEN
        IF: P00 = LOW
        THEN
            IF: P00 = LOW five times in succession
            THEN
                Selects leader low measuring mode
                Sets 16-bit timer to 7.81 ms
                Sets INTP0 rising-edge detection mode
                Enables INTP0
                Initializes leader low detection counter
        ELSE
            Initializes leader detection counter
    ELSE
        Sets leader low detection mode S_M0SET
        Initializes leader detection counter
```



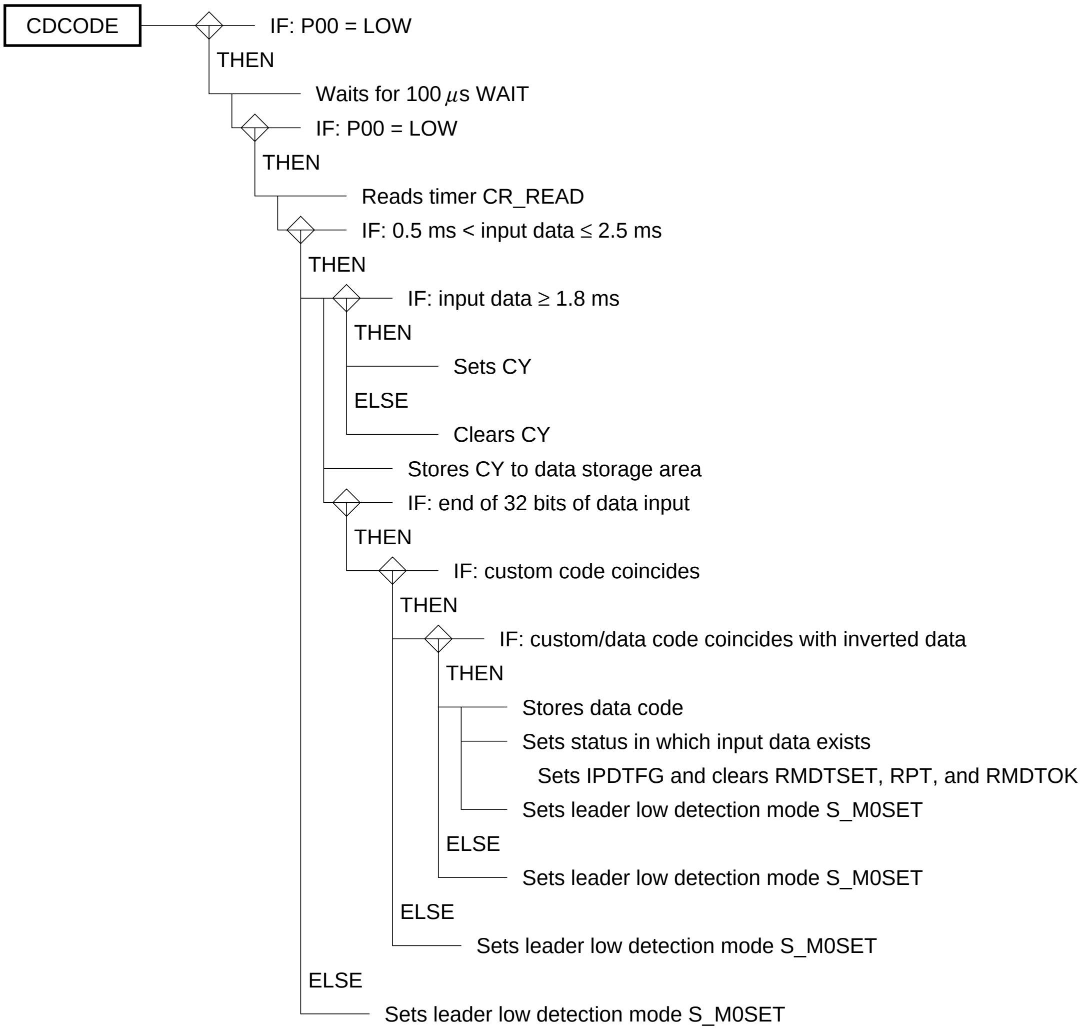
```
INTP0
    Selects register bank 0
    Waits for 100 µs WAIT
    CASE: SELMOD
    OF: 1
        Leader low measuring mode LEAD_L
    OF: 2
        Leader high measuring mode LEAD_H
    OF: 3
        Custom code/data loading mode CDCODE
    OF: 4
        Repeat code detection mode REPCD
    OF: 5
        Abnormal data detection mode ENDCHK
```



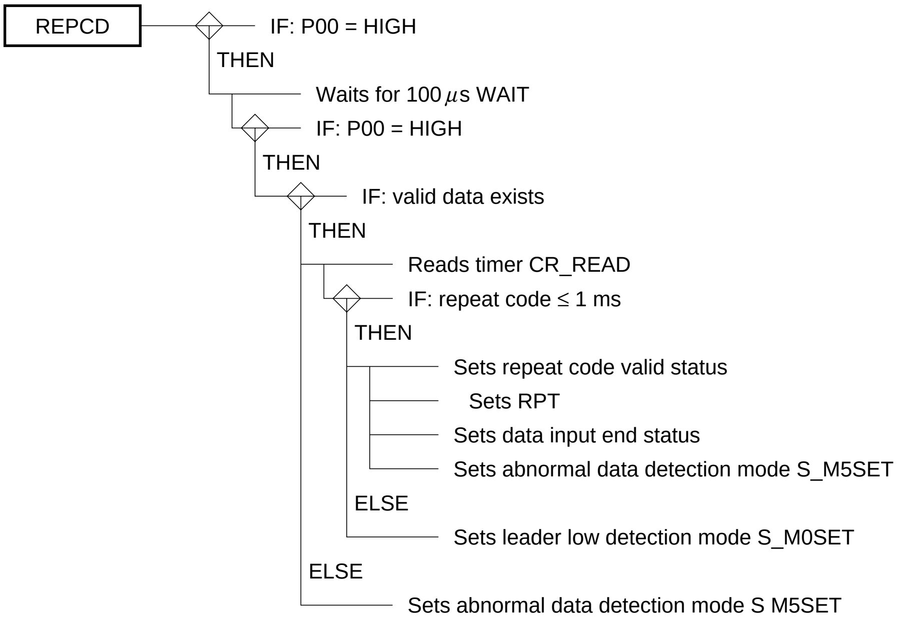
```
LEAD_L
    IF: P00 = HIGH
    THEN
        Waits for 100 µs WAIT
        IF: P00 = HIGH
        THEN
            Reads timer CR_READ
            IF: 6.8 ms ≤ leader low ≤ 11.8 ms
            THEN
                Selects leader high detection mode
                Sets INTP0 falling-edge detection mode
            ELSE
                Sets leader low detection mode S_M0SET
```



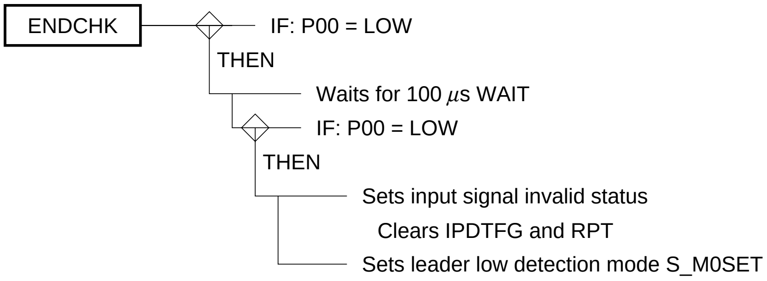
```
LEAD_H
    IF: P00 = LOW
    THEN
        Waits for 100 µs WAIT
        IF: P00 = LOW
        THEN
            Reads timer CR_READ
            IF: 2 ms ≤ leader high ≤ 5 ms
            THEN
                IF: leader high ≥ 3 ms
                THEN
                    Selects custom code/data load mode
                    Initializes data storage area
                ELSE
                    Selects repeat detection mode
                    Sets INTP0 rising-edge detection mode
            ELSE
                Sets leader low detection mode S  M0SET
```

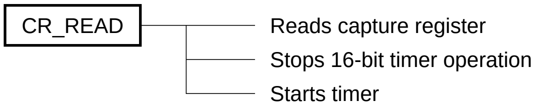


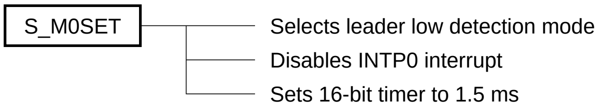
```
CDCODE
  IF: P00 = LOW
  THEN
    Waits for 100 μs WAIT
    IF: P00 = LOW
    THEN
      Reads timer CR_READ
      IF: 0.5 ms < input data ≤ 2.5 ms
      THEN
        IF: input data ≥ 1.8 ms
        THEN
          Sets CY
        ELSE
          Clears CY
        Stores CY to data storage area
        IF: end of 32 bits of data input
        THEN
          IF: custom code coincides
          THEN
            IF: custom/data code coincides with inverted data
            THEN
              Stores data code
              Sets status in which input data exists
                Sets IPDTFG and clears RMDTSET, RPT, and RMDTOK
              Sets leader low detection mode S_M0SET
            ELSE
              Sets leader low detection mode S_M0SET
          ELSE
            Sets leader low detection mode S_M0SET
      ELSE
        Sets leader low detection mode S_M0SET
```

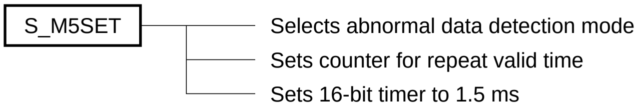


```
REPCD
  IF: P00 = HIGH
  THEN
    Waits for 100 μs WAIT
    IF: P00 = HIGH
    THEN
      IF: valid data exists
      THEN
        Reads timer CR_READ
        IF: repeat code ≤ 1 ms
        THEN
          Sets repeat code valid status
            Sets RPT
          Sets data input end status
          Sets abnormal data detection mode S_M5SET
        ELSE
          Sets leader low detection mode S_M0SET
      ELSE
        Sets abnormal data detection mode S M5SET
```


ENDCHK
IF: P00 = LOW
THEN
Waits for 100 μs WAIT
IF: P00 = LOW
THEN
Sets input signal invalid status
Clears IPDTFG and RPT
Sets leader low detection mode S_M0SET


CR_READ
Reads capture register
Stops 16-bit timer operation
Starts timer


S_M0SET
Selects leader low detection mode
Disables INTP0 interrupt
Sets 16-bit timer to 1.5 ms


S_M5SET
Selects abnormal data detection mode
Sets counter for repeat valid time
Sets 16-bit timer to 1.5 ms

(4) Program list

```
        PUBLIC  RPT,IPDTFG,RMDTOK,RMDTSET
        PUBLIC  RMENDCT,RPTCT,SELMOD,LD_CT,RMDATA
        EXTRN   CSTM
RM_DAT  DSEG    SADDR
RPTCT:  DS      1                                   ; Repeat code valid time counter
RMENDCT:        DS                                  ; No-inpu1t time counter after data input
SELMOD: DS      1                                   ; Selects mode
LD_CT:  DS      1                                   ; Leader signal detection counter
RMDATA: DS      1                                   ; Valid data storage area

RM_DATP DSEG    SADDRP
WORKP:  DS      4                                   ; Input signal storage area

        BSEG
IPDTFG  DBIT                                        ; Valid data exists
RMDTOK  DBIT                                        ; Input signal is valid
RMDTSET DBIT                                        ; Input signal exists
RPT     DBIT                                        ; Repeat code valid period

VEP0    CSEG    AT 06H
        DW      INTP0                               ; Sets vector address of INTP0

VETM0   CSEG    AT 14H
        DW      INTTM0                              ; Sets vector address of 16-bit timer

;*******************************************************
;      Remote controller signal timer processing
;*******************************************************
TM0_SEG     CSEG
INTTM0:

        SEL RB1
        EI                                          ; Enables interrupt (INTP0)
        if_bit(IPDTFG)                              ; Input signal exists?
            if_bit(RMDTOK)                          ; Valid data exists?
            RPTCT––
                if(RPTCT==#0)                       ; Repeat invalid time
                    CLR1    RPT                     ; Repeat code invalid status
                    CLR1    IPDTFG
                    CLR1    RMDTOK
                endif
                CALL        !S_LOWCT
            else
                RMENDCT––
                if(RMENDCT==#0)
                    SET1    RMDTOK                  ; Sets that valid data exists
                    SET1    RMDTSET
                    CALL    !S_M0SET                ; Sets leader (low) detection mode
                endif
                LD_CT=#5
            endif
        else
            CALL            !s_LOWCT
        endif
        RETI
```

```
S_LOWCT:
        if(SELMOD==#0)                                  ; Leader (low) detection mode?
            if_bit(!P0.0)
                LD_CT--
                if(LD_CT==#0)
                    SELMOD=#1                           ; Leader (low) measuring mode
                    TMC0=#00000000B
                    CR00=#32767                         ; Timer:  7.81 ms
                    TMC0=#00001100B
                    INTM0=#00000100B
                    CLR1    PIF0
                    CLR1    PMK0                        ; Enables INTP0 interrupt
                    LD_CT=#5
                endif
            else
                LD_CT=#5
            endif
        else
            CALL        !S_MOSET                        ; Sets leader (low) detection mode
            LD_CT=#5
        endif
        RET
$EJECT
;************************************************************
;*    Remote controller signal edge detection processing
;************************************************************
P0_SEG  CSEG
INTP0:

        SEL     RB0
        CALL    !WAIT                                   ; Waits for 100 μs
        switch(SELMOD)
        case 1:
            CALL        !LEAD_L                         ; Leader low detection processing
            break
        case 2:
            CALL        !LEAD_H                         ; Leader high detection processing
            break
        case 3:
            CALL        !CDCODE                         ; Custom/data code loading processing
            break
        case 4:
            CALL        !REPCD                          ; Repeat code detection processing
            break
        case 5:
            CALL        !ENDCHK                         ; Abnormal data detection processing
        ends
        RET1
```

```
;****************************************
;*        Leader low detection
;****************************************
LEAD_L:

        if_bit(P0.0)                          ; Level check P0.0 = 0:  noise
            CALL       !WAIT                  ; Waits for 100 µs
            if_bit(P0.0)
                CALL       !CR_READ           ; Reads timer value
                if(AX>=#3354)                 ; 6.8 ms – (1.5 ms * 4)
                    if(AX<#18035)             ; 11.8 ms – (1.5 ms * 5)
                        SELMOD=#2             ; Leader high detection mode
                        INTM0=#00000000B      ; INTP0 falling edge
                    else
                        CALL    !S_MOSET      ; Sets leader (low) detection mode
                    endif
                else
                    CALL    !S_MOSET          ; Sets leader (low) detection mode
                endif
            endif
        endif
        RET
$EJECT
;****************************************
;*      Leader high detection
;****************************************
LEAD_H:
        if_bit(!P0.0)                         ; Level check P0.0 = 1:  noise
            CALL       !WAIT                  ; Waits for 100 µs
            if_bit(!P0.0)
                CALL       !CR_READ           ; Reads timer value
                if(AX>=#6710–160/2)           ; 1.8 ms – 100 µs * 2 – 160 clocks (edge detection → timer starts)
                    if(AX<#20132–160/2)       ; 5 ms – 100 µs * 2 – 160 clocks (edge detection → timer starts)
                        if(AX>#11743–160/2)   ; Custom/data code (3 ms – 100 µs * 2)?
                            SELMOD=#3         ; Data loading mode
                            WORKP=#0000H      ; Initializes work area
                            (WORKP)+2=#8000H  ; Sets most significant bit to 1 (to check end of data)
                        else
                            SELMOD=#4         ; Repeat detection mode
                            INTM0=#00000100B  ; INTP0 rises
                        endif
                    else
                        CALL    !S_M0SET      ; Sets leader (low) detection mode
                    endif
                else
                    CALL    !S_M0SET          ; Sets leader (low) detection mode
                endif
            endif
        endif
        RET
$EJECT
```

```
;********************************
;*  Custom/data code loading
;********************************
CDCODE:
        if_bit(!P0.0)                           ; Level check P0.0 = 1:  noise
            CALL        !WAIT                   ; Waits for 100 μs
            if_bit(!P0.0)
                CALL        !CR_READ            ; Reads timer value
                if(AX>=#1257–190/2)             ; 0.5 ms – 100 μs * 2 – 190 clocks (edge detection → timer starts)
                    if(AX<#9646–190/2)          ; 2.5 ms – 100 μs * 2 –190 clocks (edge detection → timer starts)
                        if(AX>=#6710–190/2)     ; 1.8 ms – 100 μs * 2 – 190 clocks (edge detection → timer starts)
                            SET1    CY
                        else
                            CLR1    CY
                        endif
                        HL=#WORKP+3             ; Sets work area address
                        C=#4                    ; Sets number of digits of work area
                    WKSHFT:
                        A=[HL]                  ; Stores 1-bit data
                        RORC        A,1         ; Shifts 1 bit
                        [HL]=A
                        HL– –
                        DBNZ        C,$WKSHFT   ; End of shifting all bits
                        if_bit(CY)              ; End of 32-bit input?
                            if(WORKP+0==#CSTM) (A)
                                                ; Custom code check
                                A^WORKP+1
                                if(A==#0FFH)    ; Custom code inverted data check
                                    A=WORKP+2
                                    A^=WORKP+3  ; Data code inverted data check
                                    if(A==#0FFH)
                                                ; Stores input data
                                        RMDATA=WORKP+2 (A)
                                                ; Sets status in which input data exists
                                        SET1    IPDTFG
                                        CLR1    RMDTSET
                                        CLR1    RPT
                                        CLR1    RMDTOK
                                        CALL    !S_M5SET
                                    else
                                                ; Sets leader (low) detection mode
                                        CALL    !S_M0SET
                                    endif
                                else
                                                ; Sets leader (low) detection mode
                                    CALL    !S_M0SET
                                endif
                            else
                                CALL        !S_M0SET
                            endif
                        endif
                    else
                        CALL    !S_M0SET      ; Sets leader (low) detection mode
                    endif
                else
                    CALL    !S_M0SET          ; Sets leader (low) detection mode
                endif
            endif
        endif
        RET
$EJECT
;************************************
;*      Repeat code detection
;************************************
REPCD:
        if_bit(P0.0)                          ; Level check P0.0 = 0:  noise
            CALL        !WAIT                 ; Waits for 100 μs
            if_bit(P0.0)
                if_bit(RMDTOK)                ; Valid data exists?
                    CALL    !CR_READ          ; Reads timer value
                    if(AX<=#3354-190/2)       ; 1 ms – 100 μs * 2 – 190 clocks (edge detection → timer starts)
                        SET1    RPT
                        CLR1    RMDTOK        ; Input signal check after end of data
                        CLR1    RMDTSET
                        CALL    !S_M5SET
                    else
                        CALL    !S_M0SET      ; Sets leader (low) detection mode
                    endif
                else
                    CALL    !S_M0SET          ; Sets leader (low) detection mode
                endif
            endif
        endif
        RET
$EJECT
```

```
;****************************************
;*      Abnormal data detection
;****************************************
ENDCHK:
        if_bit(!P0.0)                         ; Level check P0.0 = 1:  noise
            CALL       !WAIT                  ; Waits for 100 μs
            if_bit(!P0.0)
                CLR1       IPDTFG             ; Abnormal data input
                CLR1       RPT                ; Input signal invalid
                CALL       !S_M0SET           ; Sets leader (low) detection mode
            endif
        endif
        RET

;****************************************
;*         Waits for 100 μs
;****************************************
WAIT:
        B=#(838–14–12–8)/12                   ; CALL(14), RET(12), MOV(8)
WAITCT:                                       ; Sets 100 μs
        DBNZ    B,$WAITCT                     ; 1 instruction 12 clocks
        RET

;*****************************************
;*   Sets leader (low) detection mode
;*****************************************
S_M0SET:
        TMC0=#00000000B
        CR00=#6290
        TCL0=#00100000B                       ; Sets timer to 1.5 ms
        TMC0=#00001100B
        SELMOD=#0                             ; Leader (low) detection mode
        SET1    PMK0
        RET

;*****************************************
;*   Sets abnormal data detection mode
;*****************************************
S_M5SET:
        RPTCT=#173                            ; 250 ms measuring counter
        SELMOD=#5                             ; Data input end mode
        RMENDCT=#3                            ; No-input checking counter
        TMC0=#00000000B                       ; Stops operation
        CR00=#6290                            ; Sets 1.5 ms
        TMC0=#00001100B
        RET
;****************************************
;*      Reads timer count value
;****************************************
CR_READ:
        AX=CR01
        TMC0=#00000000B                       ; Stops operation
        TMC0=#00001100B                       ; Starts timer
        RET
```

5.3.2 Remote controller signal reception by PWM output and free running mode (μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries only)

Table 5-2 shows the valid pulse width when a remote controller signal is received by this program. <1> through <6> below describes how each signal is processed.

Table 5-2. Valid Time of Input Signal

Signal Name		Output Time	Valid Time
Leader code (low)		9 ms	3 ms-10 ms
Leader code (high)	Normal	4.5 ms	3 ms-5 ms
	Repeat	2.25 ms	1.8 ms-3 ms
Custom/data code	0	1.125 ms	0.5 ms-1.8 ms
	1	2.25 ms	1.8 ms-2.5 ms

<1> Leader code (low)

The value of the capture register (CR01) is stored to memory by an interrupt request that occurs when the falling edge of INTP0 is detected.

The pulse width is measured from the difference between the values of CR01 and the compare register (CR00) when the rising edge is generated.

<2> Leader code (high)

The pulse width between the high levels of the leader code is measured by the falling-edge interrupt request INTP0 and the count value of the timer.

<3> Custom/data code

The pulse width of each 1 bit (1 cycle) is measured by the falling-edge interrupt request INTP0. After the data of the 32nd bit has been loaded, the system tests if the inverted data and custom code coincide. It also checks that there is no data of the 33rd bit.

<4> Repeat code detection

When the high level of the leader code is less than 3 ms, the pulse width from output of the leader code to the rising edge of the INTP0 is measured.

<5> Valid period of repeat code

After the valid data has been input, the overflow flag (OVF0) of the 16-bit timer/event counter is tested by the main program, and the repeat code valid time of 250 ms is measured.

<6> Time out during pulse width measurement

The OVF0 of the 16-bit timer/event counter is tested during pulse width measurement. If it is detected two times, time out is assumed and the data is assumed to be invalid.

Because the 16-bit timer/event counter operates in the PWM mode in this example, the remote controller signal is received and, at the same time, PWM output can be performed by linking the program of **5.2 PWM Output.**

(1) Description of package

<Public declaration symbol>

TIM_PRO	: name of subroutine processing timer overflow
RMDATA	: stores remote controller receive data
RPT	: repeat valid period identification flag
IPDTFG	: valid data identification flag
RMDTOK	: valid input signal identification flag
RMDTSET	: input signal identification flag
OVSENS	: INTP0 processing timer overflow detection flag

<Registers used>

Bank 0: AX, BC, HL

<RAM used>

Name	Usage	Attribute	Bytes
RPTCT	Repeat code invalid time counter	SADDR	1
RMENDCT	No-input time counter after data input		
SELMOD	Mode selection		
LD_CT	Leader signal detection counter		
RMDATA	Valid data storage area		
TO_CNT	Timer overflow detection counter		
CR01_NP	Newest timer count value storage area	SADDRP	2
CR01_OP	Previous timer count value storage area		
WORKP	Input signal storage area		4

<Flag used>

Name	Usage
IPDTFG	Presence/absence of valid data
RMDTOK	Presence/absence of valid input signal
RMDTSET	Presence/absence of input signal
RPT	Judgment whether repeat valid period elapsed
TO_FLG	Occurrence of timer overflow
OVSENS	Detection of timer overflow by INTP0 processing

<Nesting>

5 levels 11 bytes

<Hardware used>

- 16-bit timer/event counter
- P00/INTP0
- P30/TO0

<Initial setting>

- Setting of 16-bit timer/event counter

PWM output mode	TMC0 = #00000010B
★ PWM basic cycle: 61.0 μs	TCL0 = #00100000B
Low-active output	TOC0 = #00000011B
• P30 output mode	PM30 = 0
• INTP0 sampling clock $f_X/2^7$	SCS = #00000011B
• INTP0 high-priority interrupt	PPR0 = 0
• Enables INTP0 interrupt	PMK0 = 0

- Defines custom code to CSTM and declares PUBLIC
- RAM clear

<Starting>

- Test the OVF0 of the 16-bit timer/event counter. When OVF0 is set, call subroutine TIM_PRO.
- Start by an interrupt request when the valid edge of the remote controller signal is detected.

(2) Example of use

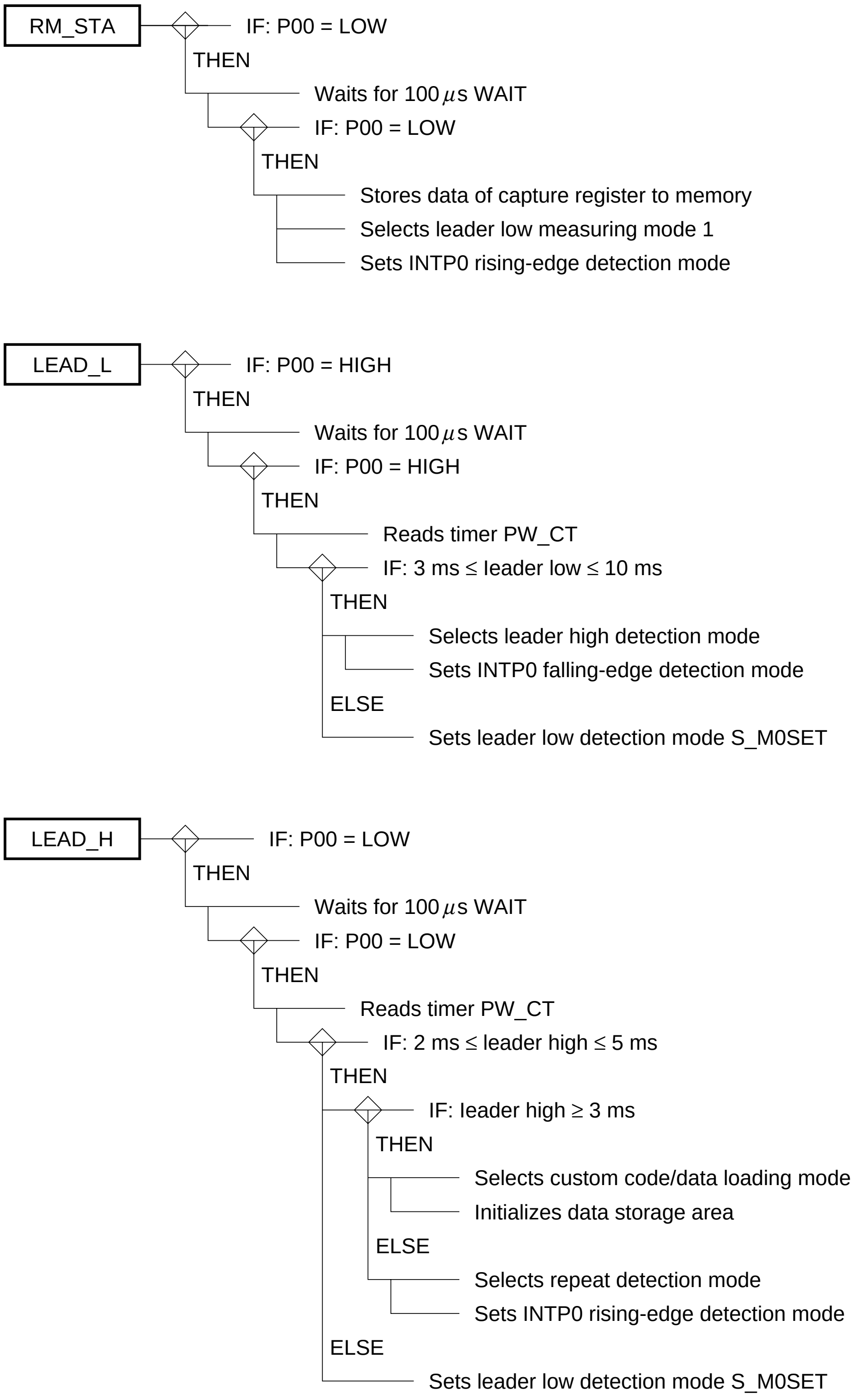
```
        PUBLIC CSTM
        EXTRN  RMDATA,RPTCT,PWM,PWMOUT,TIM_PRO
        EXTBIT RPT,RMDTSET,IPDTFG,TO_FLG,OVSENS
CSTM    EQU    9DH                        ; Custom code

        TOC0=#00000011B                   ; PWM output, low active setting
        TCL0=#00100000B                   ; Selects count clock fX/2
        TMC0=#00000010B                   ; PWM mode, overflow occurs
        INTM0=#00000000B                  ; INTP0 falling edge
        SCS=#00000011B                    ; INTP0 sampling clock fX/128

        CLR1   PPR0                       ; INTP0 with high priority
        CLR1   RPT                        ; Clears flag
        CLR1   IPDTFG
        CLR1   RMDTSET

        CLR1   PMK0                       ; Enables INTP0 interrupt
        EI
DT_TEST:
        if_bit(OVSENS)                    ; Detects timer overflow by INTP0 processing
           CLR1   OVSENS
           CALL   !TIM_PRO
        elseif_bit(OVF0)                  ; Timer overflow occurs
           CLR1   OVF0
           SET1   TO_FLG
           CALL   !TIM_PRO
        endif
        if_bit(RMDTSET)
           CLR1   RMDTSET
           if_bit(RPT)
;
;              Repeat processing
;
           else
;
;              Processing when input exists
;
           endif
        else
           if_bit(!RPT)
;
;              Processing when input does not exist
;
           endif
        endif
        MOV    PWMOUT,A
        CALL   !PWM
```

(3) SPD chart

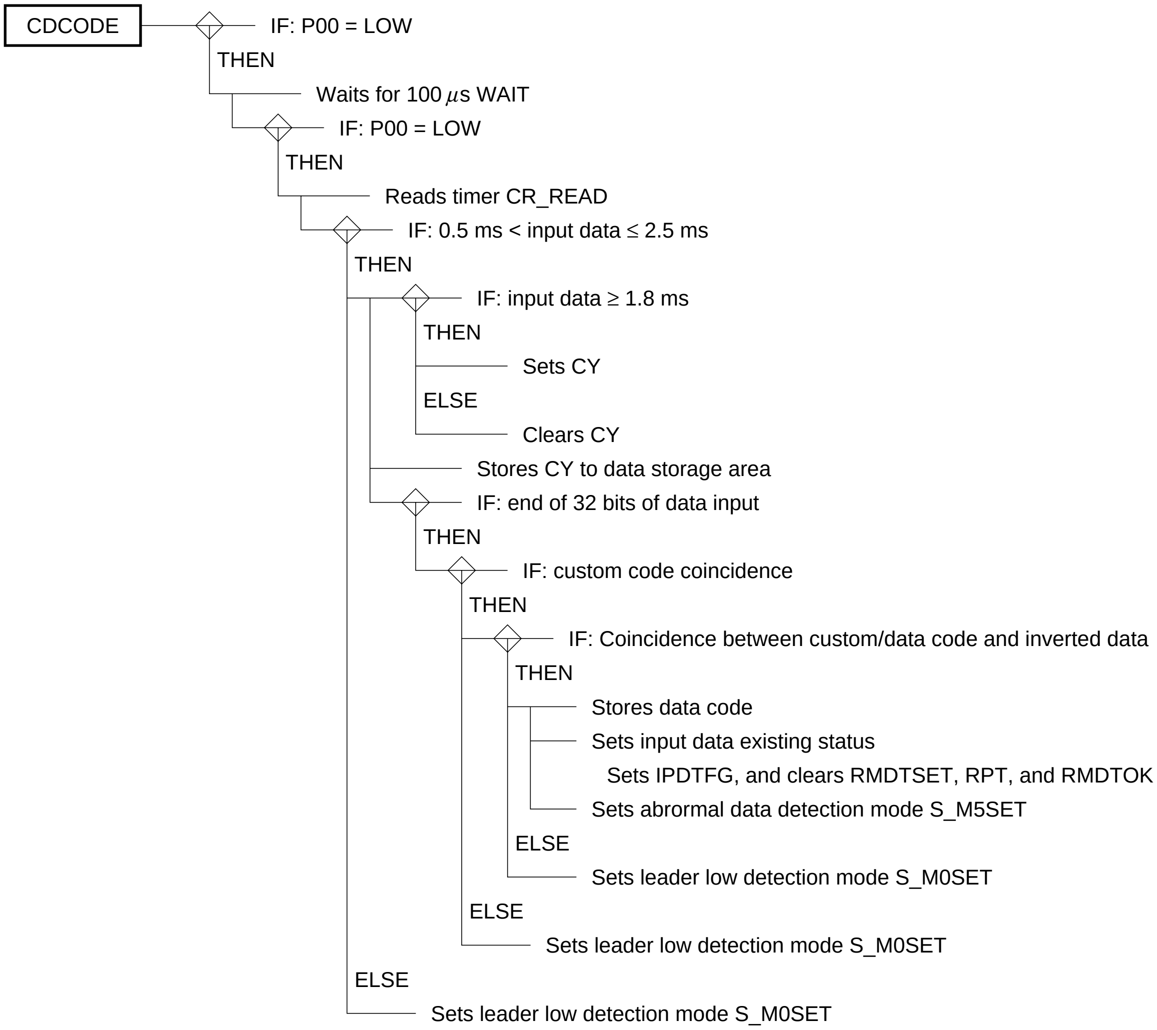
```
TIM_PRO
  IF: input signal exists
  THEN
    IF: valid data exists
    THEN
      IF: repeat code invalid time
      THEN
        Sets repeat code invalid status
          Clears RPT, IPDTFG, RMDTOK
      Checks timer overflow TO_CHK
    ELSE
★     IF: No input exists after input of data (within 61.0 µs × 2)
      THEN
        Sets that valid data exists
          Sets RMDTOK, RMDTSET
        Sets leader low detection mode S_M0SET
  ELSE
    Checks timer overflow TO_CHK
```

```
TO_CHK
  IF: leader low detection mode
  THEN
    Sets that timer overflow does not occur
  ELSE
    Timer overflow count
      IF: Timer overflow occurs 2 times
      THEN
        Sets leader low detection mode S_M0SET
```

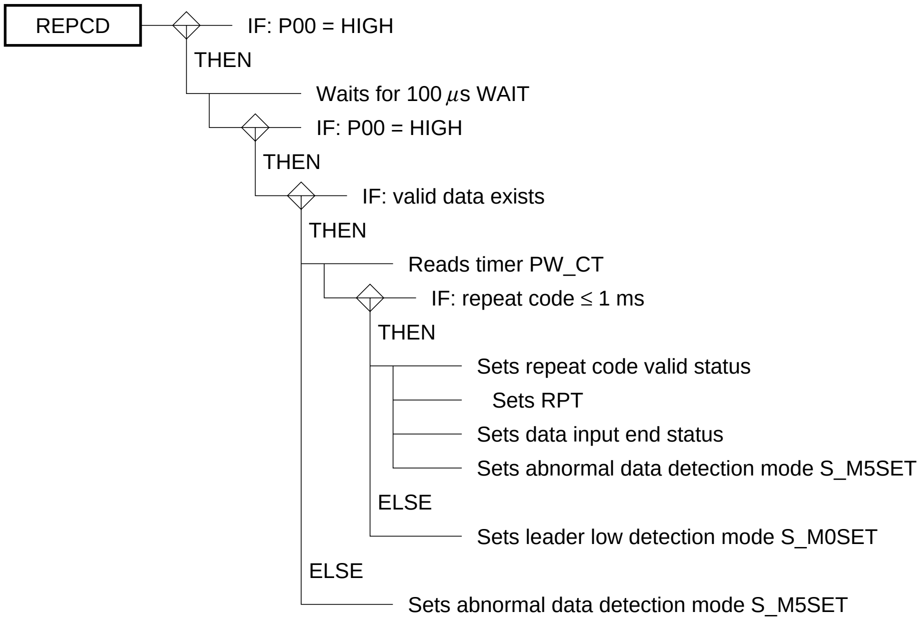
```
INTP0
  Selects register bank 0
  Waits for 100 µs WAIT
  CASE: SELMOD
  OF: 0
    Leader low detection mode RM_STA
  OF: 1
    Leader low measuring mode LEAD_L
  OF: 2
    Leader high measuring mode LEAD_H
  OF: 3
    Custom code/data loading mode CDCODE
  OF: 4
    Repeat code detection mode REPCD
  OF: 5
    Abnormal data detection mode ENDCHK
```



```
RM_STA
    IF: P00 = LOW
    THEN
        Waits for 100 µs WAIT
        IF: P00 = LOW
        THEN
            Stores data of capture register to memory
            Selects leader low measuring mode 1
            Sets INTP0 rising-edge detection mode

LEAD_L
    IF: P00 = HIGH
    THEN
        Waits for 100 µs WAIT
        IF: P00 = HIGH
        THEN
            Reads timer PW_CT
            IF: 3 ms ≤ leader low ≤ 10 ms
            THEN
                Selects leader high detection mode
                Sets INTP0 falling-edge detection mode
            ELSE
                Sets leader low detection mode S_M0SET

LEAD_H
    IF: P00 = LOW
    THEN
        Waits for 100 µs WAIT
        IF: P00 = LOW
        THEN
            Reads timer PW_CT
            IF: 2 ms ≤ leader high ≤ 5 ms
            THEN
                IF: leader high ≥ 3 ms
                THEN
                    Selects custom code/data loading mode
                    Initializes data storage area
                ELSE
                    Selects repeat detection mode
                    Sets INTP0 rising-edge detection mode
            ELSE
                Sets leader low detection mode S_M0SET
```



```
CDCODE
  IF: P00 = LOW
  THEN
    Waits for 100 µs WAIT
    IF: P00 = LOW
    THEN
      Reads timer CR_READ
      IF: 0.5 ms < input data ≤ 2.5 ms
      THEN
        IF: input data ≥ 1.8 ms
        THEN
          Sets CY
        ELSE
          Clears CY
        Stores CY to data storage area
        IF: end of 32 bits of data input
        THEN
          IF: custom code coincidence
          THEN
            IF: Coincidence between custom/data code and inverted data
            THEN
              Stores data code
              Sets input data existing status
                Sets IPDTFG, and clears RMDTSET, RPT, and RMDTOK
              Sets abrormal data detection mode S_M5SET
            ELSE
              Sets leader low detection mode S_M0SET
          ELSE
            Sets leader low detection mode S_M0SET
      ELSE
        Sets leader low detection mode S_M0SET
```



```
REPCD
  IF: P00 = HIGH
  THEN
    Waits for 100 µs WAIT
    IF: P00 = HIGH
    THEN
      IF: valid data exists
      THEN
        Reads timer PW_CT
        IF: repeat code ≤ 1 ms
        THEN
          Sets repeat code valid status
            Sets RPT
          Sets data input end status
          Sets abnormal data detection mode S_M5SET
        ELSE
          Sets leader low detection mode S_M0SET
      ELSE
        Sets abnormal data detection mode S_M5SET
```

```
ENDCHK
    IF: P00 = LOW
    THEN
        Waits for 100 μs WAIT
        IF: P00 = LOW
        THEN
            Sets input signal invalid status
              Clears IPDTFG, RPT
            Sets leader low detection mode S_M0SET

PW_CT
    IF: OVF occurs after edge detection processing
    THEN
        IF: OVF occurs < interrupt acknowledgment processing time (65 clocks)
        THEN
            Sets that timer overflow occurs
    Loads capture register value
    Subtracts capture register value from previous value
    IF: borrow occurs as result of subtraction (CY = 1)
    THEN
        IF: timer overflow occurs (TO_FLG = 1)
        THEN
            Clears CY flag
    ELSE
        IF: timer overflow occurs (TO_FLG = 1)
        THEN
            Sets CY flag
    Stores capture register value to memory

S_M0SET
    Selects leader low detection mode
      Clears TO_FLG
    Sets INTP0 falling-edge detection mode

S_M5SET
    Selects abnormal data detection mode
    Sets repeat valid time counter
```

(4) Program list

```
        PUBLIC  TIM_PRO,RPT,IPDTFG,RMDTOK,RMDTSET
        PUBLIC  RMENDCT,RPTCT,SELMOD,LD_CT,RMDATA
        PUBLIC  TO_FLG,OVSENS
        EXTRN   CSTM

RM_DAT  DSEG    SADDR
RPTCT:  DS      1                                  ; Repeat code valid time counter
RMENDCT:DS      1                                  ; No-input time counter after data input
SELMOD: DS      1                                  ; Mode selection
LD_CT:  DS      1                                  ; Leader signal detection counter
RMDATA: DS      1                                  ; Valid data storage area
TO_CNT: DS      1                                  ; Timer overflow counter

RM_DATP DSEG    SADDRP
CR01_NP:DS      2                                  ; Newest timer counter value storage area
CR01_OP:DS      2                                  ; Previous timer counter value storage area
WORKP:  DS      4                                  ; Input signal storage area

        BSEG
IPDTFG  DBIT                                       ; Valid data exists
RMDTOK  DBIT                                       ; Input signal valid
RMDTSET DBIT                                       ; Input signal exists
RPT     DBIT                                       ; Repeat code valid period
TO_FLG  DBIT                                       ; Timer overflow occurs
OVSENS  DBIT                                       ; Detects timer overflow by INTP0 processing

VEP0    CSEG    AT 06H
        DW      INTP0                              ; Sets vector address of INTP0

$EJECT
;*******************************************************
;      Remote controller signal timer processing
;*******************************************************
TM0_SEG CSEG
TIM_PRO:
        if_bit(IPDTFG)                             ; Input signal exists?
            if_bit(RMDTOK)                         ; Valid data exists?
            RPTCT– –
                if(RPTCT==#0)                      ; Repeat invalid time
                    CLR1    RPT                    ; Repeat code valid status
                    CLR1    IPDTFG
                    CLR1    RMDTOK
                endif
            else
                RMENDCT– –
                if(RMENDCT==#0)
                    SET1    RMDTOK                 ; Valid data exists
                    SET1    RMDTSET
                    CALL    !S_M0SET               ; Sets leader (low) detection mode
                endif
            endif
        else
            CALL    !TO_CHK                        ; Checks timer overflow
        endif
        RET
```

```
 TO_CHK:
       if(SELMOD==#0)
           CLR1       TO_FLG
       else
           TO_CNT++
           if(TO_CNT==#2)
               CALL       !S_M0SET              ; Sets start edge detection mode
           endif
       endif
       RET
    $EJECT
;***********************************************************
;*    Remote controller signal edge detection processing
;***********************************************************
P0_SEG  CSEG
INTP0:
       SEL RB0

       CALL    !WAIT                             ; Waits for 100 µs

       switch(SELMOD)

       case 0:
           CALL    !RM_STA                       ; Start edge detection processing
           break
       case 1:
           CALL    !LEAD_L                       ; Leader low detection processing
           break
       case 2:
           CALL    !LEAD_H                       ; Leader high detection processing
           break
       case 3:
           CALL    !CDCODE                       ; Custom/data code loading processing
           break
       case 4:
           CALL    !REPCD                        ; Repeat code detection processing
           break
       case 5:
           CALL    !ENDCHK                       ; Abnormal data detection processing
       ends
       RET1

;***********************************************************
;*                Start edge detection
;***********************************************************
RM_STA:
       CLR1    TO_FLG                            ; Starts timer count
       if_bit(!P0.0)                             ; Level check P0.0 = 1:  noise
           CALL    !WAIT                         ; Waits for 100 µs
           if_bit(!P0.0)
               CR01_OP=CR01 (AX)                 ; Stores capture register
               SELMOD=#1                         ; Leader low detection mode
               INTM0=#00000100B                  ; INTP0 rising edge
               TO_CNT=#0
           endif
       endif
       RET
```

```
;****************************************
;*         Leader low detection
;****************************************
LEAD_L:
        if_bit(P0.0)                               ;  Level check P0.0 = 1:  noise
            CALL        !WAIT                      ;  Waits for 100 µs
            if_bit(P0.0)
                CALL        !PW_CT                 ;  Reads timer value
                if_bit(!CY)
                    TO_CNT=#0
                    if(AX>=#12582)                 ;  3 ms
                        if(AX<#41942)              ;  10 ms
                            SELMOD=#2              ;  Leader high detection mode
                            INTM0=#00000000B       ;  INTP0 falling edge
                        else
                            CALL    !S_M0SET       ;  Sets start edge detection mode
                        endif
                    else
                        CALL    !S_M0SET           ;  Sets start edge detection mode
                    endif
                else
                    CALL    !S_M0SET               ;  Sets start edge detection mode
                endif
            endif
        endif
        RET
    $EJECT
;****************************************
;*      Leader high detection
;****************************************
LEAD_H:
        if_bit(!P0.0)                              ;  Level check P0.0 = 0:  noise
            CALL        !WAIT                      ;  Waits for 100 µs
            if_bit(!P0.0)
                CALL        !PW_CT                 ;  Reads timer value
                if_bit(!CY)
                    TO_CNT=#0
                    if(AX>=#7549)                  ;  1.8 ms
                        if(AX<#20971)              ;  5 ms
                            if(AX>#12582)          ;  Custom/data code (3 ms)?
                                SELMOD=#3          ;  Data loading mode
                                WORKP=#0000H       ;  Initializes work area
                                (WORKP)+2=#8000H   ;  Sets most significant bit to 1 (to confirm end of data)
                            else
                                SELMOD=#4          ;  Repeat detection mode
                                INTM0=#00000100B   ;  INTP0 rises
                            endif
                        else
                            CALL    !S_M0SET       ;  Sets start edge detection mode
                        endif
                    else
                        CALL    !S_M0SET           ;  Sets start edge detection mode
                    endif
                else
                    CALL    !S_M0SET               ;  Sets start edge detection mode
                endif
            endif
        endif
        RET
    $EJECT
```

```
;******************************
;*  Custom/data code loading
;******************************
CDCODE:
        if_bit(!P0.0)                               ; Level check P0.0 = 1:  noise
            CALL    !WAIT                           ; Waits for 100 μs
            if_bit(!P0.0)
                CALL    !PW_CT                      ; Reads timer value
                if_bit(!CY)
                    TO_CNT=#0
                    if(AX>=#2096)                   ; 0.5 ms
                        if(AX<#10485)               ; 2.5 ms
                            if(AX>=#7549)           ; 1.8 ms
                                SET1    CY
                            else
                                CLR1    CY
                            endif

                            HL=#WORKP+3             ; Sets work area address
                            C=#4                    ; Sets number of work area digits
                        WKSHFT:
                            A=[HL]                  ; Stores 1-bit data
                            RORC    A,1             ; Shifts 1 bit
                            [HL]=A
                            HL--
                            DBNZ    C,$WKSHFT       ; End of shifting all digits

                            if_bit(CY)              ; End of input of 32 bits?
                                                    ; Checks custom code
                                if(WORKP+0==#CSTM) (A)
                                    A^=WORKP+1
                                    if(A==#0FFH)    ; Checks custom code inverted data
                                        A=WORKP+2
                                                    ; Checks data code inverted data
                                        A^=WORKP+3
                                        if(A==#0FFH)
                                                    ; Stores input data
                                            RMDATA=WORKP+2 (A)
                                                    ; Sets input data existing status
                                        SET1    IPDTFG
                                        CLR1    RMDTSET
                                        CLR1    RPT
                                        CLR1    RMDTOK
                                        CALL    !S_M5SET
                                    else
                                                    ; Sets start edge detection mode
                                        CALL    !S_M0SET
                                    endif
                                else
                                                    ; Sets start edge detection mode
                                    CALL    !S_M0SET
                                endif
                            else
                                CALL    !S_M0SET
                            endif
                        endif
                    else
                        CALL    !S_M0SET            ; Sets start edge detection mode
                    endif
                else
```

```
                        CALL        !S_M0SET    ; Sets start edge detection mode
                    endif
                else
                    CALL    !S_M0SET            ; Sets start edge detection mode
                endif
            endif
        endif
        RET
$EJECT

;************************************
;*      Repeat code detection
;************************************
REPCD:
        if_bit(P0.0)                            ; Level check P0.0 = 1:  noise
            CALL        !WAIT                   ; Waits for 100 µs
            if_bit(P0.0)
                if_bit(RMDTOK)                  ; Valid data?
                    CALL    !PW_CT              ; Reads timer value
                    if_bit(!CY)
                        TO_CNT=#0
                        if(AX<=#4193)           ; 1 ms
                            SET1    RPT
                            CLR1    RMDTOK      ; Checks input signal after end of data
                            CLR1    RMDTSET
                            CALL    !S_M5SET
                        else
                            CALL    !S_M0SET    ; Sets start edge detection mode
                        endif
                    else
                        CALL    !S_M0SET        ; Sets start edge detection mode
                    endif
                else
                    CALL    !S_M0SET            ; Sets start edge detection mode
                endif
            endif
        endif
        RET
    $EJECT
```

```
;*********************************************
;*          Abnormal data detection
;*********************************************
ENDCHK:
        if_bit(!P0.0)                   ; Level check P0.0 = 1:  noise
           CALL    !WAIT                ; Waits for 100 µs
           if_bit(!P0.0)
               CLR1   IPDTFG            ; Abnormal data input
               CLR1   RPT               ; Input signal invalid
               CALL   !S_M0SET          ; Sets start edge detection mode
           endif
        endif
        RET

;*********************************************
;*   Calculation of capture register value
;*********************************************
PW_CT:
        if_bit(OVF0)                    ; OVF0 after edge detection?
           if(CR01<#10000–33) (AX)      ; Interrupt acknowledgment processing time = 65 clocks (MAX)
               CLR1   OVF0
               SET1   OVSENS
               SET1   TO_FLG
           endif
        endif

        CR01_NP=CR01 (AX)               ; Loads capture register value

        A=CR01_NP+0                     ; AX = CR01_NP – CR01_OP
        A–=CR01_OP
        X=A
        A=CR01_NP+1
        SUBC   A,CR01_OP+1

        BC=AX                           ; Saves operation result
        if_bit(CY)                      ; CR01_NP > CR01_OP
           if_bit(TO_FLG)               ; Timer overflow occurs (flag test)
               CLR1   CY                ; Normal data
           endif
        else
           if_bit(TO_FLG)               ; Timer overflow
               SET1   CY                ; Error occurs
           endif
        endif

        CR01_OP=CR01_NP (AX)
        AX=BC                           ; Restores operation result
        CLR1   TO_FLG
        RET
```

```
;**********************************************
;*             Waits for 100 µs
;**********************************************
WAIT:
        B=#(838–14–12–8)/12                      ; CALL (14), RET (12), MOV (8)
WAITCT:                                          ; Sets 100 µs
        DBNZ    B,$WAITCT                        ; 1 instruction 12 clocks
        RET

;**********************************************
;*      Sets start edge detection mode
;**********************************************
S_M0SET:
        TO_CNT=#0
        SELMOD=#0                                ; Start edge detection mode
        INTM0=#00000000B                         ; INTP0 falling edge
        RET

;**********************************************
;*   Setting of abnormal data detection mode
;**********************************************
S_M5SET:
        RPTCT=#16                                ; 250 ms measuring counter
        SELMOD=#5                                ; Data input end mode
        RMENDCT=#2                               ; No-input checking counter
        RET
```

[MEMO]

CHAPTER 6 APPLICATIONS OF 8-BIT TIMER/EVENT COUNTER

The 8-bit timer/event counter of the 78K/0 series has the following functions:

- Interval timer
- External event counter
- Square wave output
- PWM output (μPD780024, 780024Y, 780034, 780034Y, 780924, and 780964 subseries only)

Two channels or three channels of 8-bit timers/event counters are provided and these timers/event counters can be used as a 16-bit timer/event counter when connected in cascade.

The 8-bit timers/event counters are set by the following registers:

<μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries, μPD780001>

- Timer clock select register 1 (TCL1)
- 8-bit timer mode control register (TMC1)
- 8-bit timer output control register (TOC1)
- Port mode register 3 (PM3)
- Port 3 (P3)

<μPD780024, 780024Y, 780034, 780034Y subseries>

- Timer clock select register 50, 51 (TCL50, TCL51)
- 8-bit timer mode control register 50, 51 (TMC50, TMC51)
- Port mode register (PM7)

<μPD780924, 780964 subseries>

- Timer clock select register 50, 51, 52 (TCL50, TCL51, TCL52)
- 8-bit timer mode control register 50, 51, 52 (TMC50, TMC51, TMC52)

★ **Caution The format of the registers provided on the μPD780024, 780024Y, 780034, 780034Y, 780924, and 780964 subseries differs from the format of the registers used in the program examples in this chapter. When using a program example in this chapter with any of the μPD780024, 780024Y, 780034, 780034Y, 780924, and 780964 subseries, change the setting of the registers according to the registers of the microcontroller used.**

Figure 6-1. Format of Timer Clock Select Register 1
(μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries, μPD780001)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL1	TCL17	TCL16	TCL15	TCL14	TCL13	TCL12	TCL11	TCL10	FF41H	00H	R/W

TCL13	TCL12	TCL11	TCL10	Selects count clock of 8-bit timer register 1
0	0	0	0	Falling edge of TI1
0	0	0	1	Rising edge of TI1
0	1	1	0	$f_X/2^2$ (2.5 MHz)
0	1	1	1	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^7$ (78.1 kHz))
1	1	0	0	$f_X/2^8$ (39.1 kHz)
1	1	0	1	$f_X/2^9$ (19.6 kHz)
1	1	1	0	$f_X/2^{10}$ (9.8 kHz)
1	1	1	1	$f_X/2^{12}$ (2.4 kHz)
Others				Setting prohibited

TCL17	TCL16	TCL15	TCL14	Selects count clock of 8-bit timer register 2
0	0	0	0	Falling edge of TI2
0	0	0	1	Rising edge of TI2
0	1	1	0	$f_X/2^2$ (2.5 MHz)
0	1	1	1	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^7$ (78.1 kHz)
1	1	0	0	$f_X/2^8$ (39.1 kHz)
1	1	0	1	$f_X/2^9$ (19.6 kHz)
1	1	1	0	$f_X/2^{10}$ (9.8 kHz)
1	1	1	1	$f_X/2^{12}$ (2.4 kHz)
Others				Setting prohibited

Caution Before writing new data to TCL1, stop the timer operation once.

Remarks 1. f_X : main system clock oscillation frequency

2. TI1 : input pin of 8-bit timer register 1

3. TI2 : input pin of 8-bit timer register 2

4. ( ) : at f_X = 10.0 MHz

★ **Figure 6-2. Format of Timer Cock Select Register 50 (μPD780024, 780024Y, 780034, 780034Y subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL50	0	0	0	0	0	TCL502	TCL501	TCL500	FF71H	00H	R/W

TCL502	TCL501	TCL500	Selects count clock
0	0	0	Falling edge of TI50
0	0	1	Rising edge of TI50
0	1	0	f_X (8.38 MHz)
0	1	1	$f_X/2^2$ (2.09 MHz)
1	0	0	$f_X/2^4$ (523 kHz)
1	0	1	$f_X/2^6$ (131 kHz)
1	1	0	$f_X/2^8$ (32.7 kHz)
1	1	1	$f_X/2^{10}$ (8.18 kHz)

Cautions 1. Before writing new data to TCL50, stop the timer operation.

2. Be sure to clear bits 3 through 7 to 0.

Remarks 1. f_X: main system clock oscillation frequency

2. ( ): f_X = 8.38 MHz

★ **Figure 6-3. Format of Timer Cock Select Register 50 (μPD780924, 780964 subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL50	0	0	0	0	0	TCL502	TCL501	TCL500	FF69H	00H	R/W

TCL502	TCL501	TCL500	Selects count clock
0	0	0	Falling edge of TI50
0	0	1	Rising edge of TI50
0	1	0	$f_X/2$ (4.19 MHz)
0	1	1	$f_X/2^3$ (1.05 MHz)
1	0	0	$f_X/2^5$ (262 kHz)
1	0	1	$f_X/2^7$ (65.5 kHz)
1	1	0	$f_X/2^9$ (16.4 kHz)
1	1	1	$f_X/2^{11}$ (4.09 kHz)

Cautions 1. Before writing new data to TCL50, stop the timer operation.

2. Be sure to clear bits 3 through 7 to 0.

Remarks 1. f_X: main system clock oscillation frequency

2. ( ): f_X = 8.38 MHz

★ **Figure 6-4. Format of Timer Cock Select Register 51 (μPD780024, 780024Y, 780034, 780034Y subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL51	0	0	0	0	0	TCL512	TCL511	TCL510	FF79H	00H	R/W

TCL512	TCL511	TCL510	Selects count clock
0	0	0	Falling edge of TI51
0	0	1	Rising edge of TI51
0	1	0	$f_X/2$ (4.19 MHz)
0	1	1	$f_X/2^3$ (1.04 MHz)
1	0	0	$f_X/2^5$ (261 kHz)
1	0	1	$f_X/2^7$ (65.4 kHz)
1	1	0	$f_X/2^9$ (16.3 kHz)
1	1	1	$f_X/2^{11}$ (4.09 kHz)

Cautions 1. Before writing new data to TCL51, stop the timer operation.

2. Be sure to clear bits 3 through 7 to 0.

Remarks 1. f_X: main system clock oscillation frequency

2. ( ): f_X = 8.38 MHz

★ **Figure 6-5. Format of Timer Cock Select Register 51 (μPD780924, 780964 subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL51	0	0	0	0	0	TCL512	TCL511	TCL510	FF71H	00H	R/W

TCL512	TCL511	TCL510	Selects count clock
0	0	0	Falling edge of TI51
0	0	1	Rising edge of T5I1
0	1	0	f_X (8.38 MHz)
0	1	1	$f_X/2$ (4.19 MHz)
1	0	0	$f_X/2^2$ (2.1 MHz)
1	0	1	$f_X/2^3$ (1.05 MHz)
1	1	0	$f_X/2^4$ (524 kHz)
1	1	1	$f_X/2^5$ (262 kHz)

Cautions 1. Before writing new data to TCL51, stop the timer operation.

2. Be sure to clear bits 3 through 7 to 0.

Remarks 1. f_X: main system clock oscillation frequency

2. ( ): f_X = 8.38 MHz

★

Figure 6-6. Format of Timer Cock Select Register 52 (μPD780924, 780964 subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL52	0	0	0	0	0	TCL522	TCL521	TCL520	FF79H	00H	R/W

TCL522	TCL521	TCL520	Selects count clock
0	0	0	Falling edge of TI52
0	0	1	Rising edge of TI52
0	1	0	$f_X/2^4$ (524 kHz)
0	1	1	$f_X/2^5$ (262 kHz)
1	0	0	$f_X/2^6$ (131 kHz)
1	0	1	$f_X/2^7$ (65.5 kHz)
1	1	0	$f_X/2^8$ (32.7 kHz)
1	1	1	$f_X/2^9$ (16.4 kHz)

Cautions 1. Before writing new data to TCL52, stop the timer operation.

2. Be sure to clear bits 3 through 7 to 0.

Remarks 1. f_X: main system clock oscillation frequency

2. ( ): f_X = 8.38 MHz

Figure 6-7. Format of 8-Bit Timer Mode Control Register (μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries, μPD780001)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TMC1	0	0	0	0	0	TMC12	TCE2	TCE1	FF49H	00H	R/W

TCE1	Controls operation of 8-bit timer register 1
0	Stops operation (clears TM1 to 0)
1	Enables operation

TCE2	Controls operation of 8-bit timer register 2
0	Stops operation (clears TM2 to 0)
1	Enables operation

TMC12	Selects operation mode
0	8-bit timer register × 2 channel mode (TM1, TM2)
1	16-bit timer register × 1 channel mode (TMS)

Cautions 1. Before changing the operation mode, stop the timer operation.

2. When using the 8-bit timer register as a 16-bit timer register, enable or stop the operation by using TCE1.

★

Figure 6-8. Format of 8-Bit Timer Mode Control Register 5n (μPD780024, 780024Y, 780034, 780034Y subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TMC5n	TCE5n	TMC5n6	0	TMC5n4	LVS5n	LVR5n	TMC5n1	TOE5n	FF70H[Note1] FF78H[Note2]	00H	R/W

TOE5n	Controls timer output
0	Disables output (port mode)
1	Enables output

TMC5n1	Other than PWM mode (TMC5n6 = 0)	PWM mode (TMC5n6 = 1)
	Controls timer F/F	Selects active level
0	Disables reverse operation	Active high
1	Enables reverse operation	Active low

LVS5n	LVR5n	Sets status of timer output F/F
0	0	Not affected
0	1	Resets timer output F/F (0)
1	0	Sets timer output F/F (1)
1	1	Setting prohibited

TMC5n4	Selects single mode/cascade connection mode
0	Single mode (used with the lowest timer)
1	Cascade connection mode (connected to the lower timer)

TMC5n6	Selects operating mode of TM5n
0	Clear & start mode on coincidence of TM50 and CR50
1	PWM (free running) mode

TCE5n	Controls count operation of TM5n
0	Clears count to 0 and disables counting (prescaler disabled)
1	Starts count operation

Notes 1. Address of TMC50

2. Address of TMC51

Remarks 1. PWM output becomes inactive level because TCE5n = 0 in PWM mode.

2. LVS5n and LVR5n are always 0 when they are read after data has been set.

3. n = 0, 1

★

Figure 6-9. Format of 8-Bit Timer Mode Control Register 50 (μPD780924, 780964 subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TMC50	TCE50	TMC506	0	0	LVS50	LVR50	TMC501	TOE50	FF68H	04H	R/W

TOE50	Controls timer output
0	Disables output (port mode)
1	Enables output

TMC501	Other than PWM mode (TMC506 = 0)	PWM mode (TMC506 = 1)
	Controls timer F/F	Selects active level
0	Disables reverse operation	Active high
1	Enables reverse operation	Active low

LVS50	LVR50	Sets status of timer output F/F
0	0	Not affected
0	1	Resets timer output F/F (0)
1	0	Sets timer output F/F (1)
1	1	Setting prohibited

TMC506	Selects operating mode of TM50
0	Clear & start mode on coincidence of TM50 and CR50
1	PWM (free running) mode

TCE50	Controls count operation of TM50
0	Clears count to 0 and disables counting (prescaler disabled)
1	Starts count operation

Remarks 1. PWM output becomes inactive level because TCE50 = 0 in PWM mode.

2. LVS50 and LVR50 are always 0 when they are read after data has been set.

★ **Figure 6-10. Format of 8-Bit Timer Mode Control Register 51 (μPD780924, 780964 subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TMC51	TCE51	TMC516	0	TMC514	LVS51	LVR51	TMC511	TOE51	FF70H	04H	R/W

TOE51	Controls timer output
0	Disables output (port mode)
1	Enables output

TMC511	Other than PWM mode (TMC516 = 0)	PWM mode (TMC516 = 1)
	Controls timer F/F	Selects active level
0	Disables reverse operation	Active high
1	Enables reverse operation	Active low

LVS51	LVR51	Sets status of timer output F/F
0	0	Not affected
0	1	Resets timer output F/F (0)
1	0	Sets timer output F/F (1)
1	1	Setting prohibited

TMC514	Selects single mode/cascade connection mode
0	Single mode
1	Cascade connection mode (connected to TM50)

TMC516	Selects operating mode of TM51
0	Clear & start mode on coincidence of TM51 and CR51
1	PWM (free running) mode

TCE51	Controls count operation of TM51
0	Clears count to 0 and disables counting (prescaler disabled)
1	Starts count operation

Remarks 1. PWM output becomes inactive level because TCE51 = 0 in PWM mode.

2. LVS51 and LVR51 are always 0 when they are read after data has been set.

★

Figure 6-11. Format of 8-Bit Timer Mode Control Register 52 (μPD780924, 780964 subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TMC52	TCE52	TMC526	0	TMC524	LVS52	LVR52	TMC521	TOE52	FF78H	04H	R/W

TOE52	Controls timer output
0	Disables output (port mode)
1	Enables output

TMC521	Other than PWM mode (TMC526 = 0)	PWM mode (TMC526 = 1)
	Controls timer F/F	Selects active level
0	Disables reverse operation	Active high
1	Enables reverse operation	Active low

LVS52	LVR52	Sets status of timer output F/F
0	0	Not affected
0	1	Resets timer output F/F (0)
1	0	Sets timer output F/F (1)
1	1	Setting prohibited

TMC524	Selects single mode/cascade connection mode
0	Single mode
1	Cascade connection mode (connected to TM51)

TMC526	Selects operating mode of TM52
0	Clear & start mode on coincidence of TM52 and CR52
1	PWM (free running) mode

TCE52	Controls count operation of TM52
0	Clears count to 0 and disables counting (prescaler disabled)
1	Starts count operation

Remarks 1. PWM output becomes inactive level because TCE52 = 0 in PWM mode.

2. LVS52 and LVR52 are always 0 when they are read after data has been set.

Figure 6-12. Format of 8-Bit Timer Output Control Register
(μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries, μPD780001)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TOC1	LVS2	LVR2	TOC15	TOE2	LVS1	LVR1	TOC11	TOE1	FF4FH	00H	R/W

TOE1	Controls output of 8-bit timer/event counter 1
0	Disables output (port mode)
1	Enables output

TOC11	Controls timer output F/F of 8-bit timer/event counter 1
0	Disables reverse operation
1	Enables reverse operation

LVS1	LVR1	Sets status of timer output F/F of 8-bit timer/ event counter 1
0	0	Not affected
0	1	Resets timer output F/F (to 0)
1	0	Sets timer output F/F (to 1)
1	1	Setting prohibited

TOE2	Controls output of 8-bit timer/event counter 2
0	Disables output (port mode)
1	Enables output

TOC15	Controls timer output F/F of 8-bit timer/event counter 2
0	Disables reverse operation
1	Enables reverse operation

LVS2	LVR2	Sets status of timer output F/F of 8-bit timer/ event counter 2
0	0	Not affected
0	1	Resets timer output F/F (to 0)
1	0	Sets timer output F/F (to 1)
1	1	Setting prohibited

Cautions 1. Before setting TOC1, be sure to stop the timer operation.

2. LVS1, LVS2, LVR1, and LVR2 are always 0 when they are read.

**Figure 6-13. Format of Port Mode Register 3
(μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries, μPD780001)**

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

PM3n	Selects input/output mode of P3n pin (n = 0-7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

Caution When using P31/TO1, P32/TO2 pins as timer outputs, set both PM31, PM32 and P31, P32 output latches to 0.

★ **Figure 6-14. Format of Port Mode Register 7 (μPD780024, 780024Y, 780034, 780034Y subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
PM7	1	1	PM75	PM74	PM73	PM72	PM71	PM70	FF27H	FFH	R/W

PM7n	Selects input/output mode of P7n pin (n = 0-5)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

6.1 Setting of Interval Timer

When using an 8-bit timer/event counter as an interval timer, set an operation mode by the 8-bit timer mode control register (TMC1) and interval time by the timer clock select register 1 (TCL1).

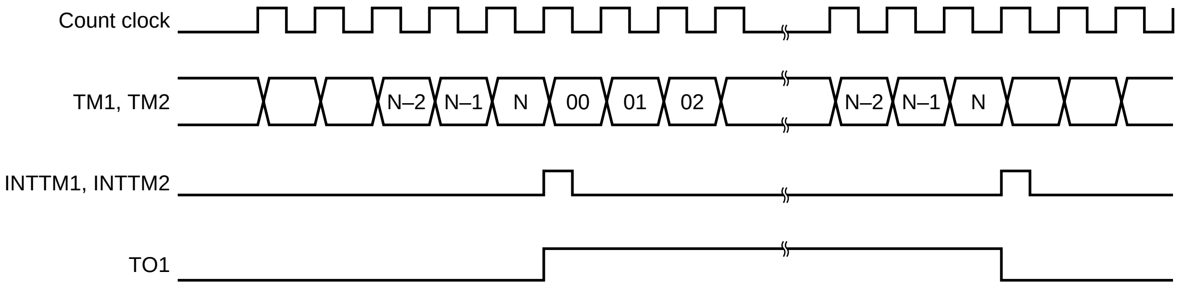
After that, set the values of the compare registers (CR10 and CR20) from the setup time and count clock. The setup time is determined by using the following expression:

Setup time = (Compare register value + 1) × Count clock cycle

The setup time can be calculated in the same manner regardless of whether each 8-bit timer/event counter is used or two 8-bit timers/event counters are used as a 16-bit timer/event counter. The count clock when two 8-bit timers/event counters are used as a 16-bit timer/event counter, however, is selected by the bits 0 through 3 (TCL10 through TCL13) of TCL1.

Examples of the modes of the 8-bit timers and 16-bit timer are described next.

Figure 6-15. Count timing of 8-Bit Timers



6.1.1 Setting of 8-bit timers

In this example, 8-bit timer 2 of the μPD78014 subseries is used to set two types of interval times: 500 μs and 100 ms.

(a) To set interval of 500 μs

<1> Setting of TMC1
Select the 8-bit timer register × 2 channel mode and enables the operation of the 8-bit timer 2.

<2> Setting of TCL1
Select $f_X/2^5$ that allows setting of 500 μs or more and has the highest resolution.

<3> Setting of CR20

$$500\ \mu s = (N + 1) \times \frac{1}{8.38\ \text{MHz}/2^5}$$

$$N = 500\ \mu s \times 8.38\ \text{MHz}/2^5 - 1 \fallingdotseq 130$$

(1) Program list

```
TCL1 = #10011001B    ;  Selects fx/2^5 as count clock
CR20 = #130
TMC1 = #00000010B
```

(b) To set interval of 100 ms

<1> Setting of TMC1
Select the 8-bit timer register × 2 channel mode and enables the operation of the 8-bit timer 2.

<2> Setting of TCL1
Select $f_X/2^{12}$ that allows setting of 100 ms or more and has the highest resolution.

<3> Setting of CR20

$$100\ \text{ms} = (N + 1) \times \frac{1}{8.38\ \text{MHz}/2^{12}}$$

$$N = 100\ \text{ms} \times 8.38\ \text{MHz}/2^{12} - 1 \fallingdotseq 204$$

(1) Program list

```
TCL1 = #11111111B    ;  Selects fx/2^12 as count clock
CR20 = #204
TMC1 = #00000010B
```

6.1.2 Setting of 16-bit timer

In this example, 8-bit timers 1 and 2 of the μPD78014 subseries are connected in cascade as a 16-bit timer to set two types of interval times: 500 ms and 10 s.

(a) To set interval of 500 ms

<1> Setting of TMC1
Select the 16-bit timer register × 1 channel mode and enables the operation of the 8-bit timers 1 and 2.

<2> Setting of TCL1
Select $f_X/2^6$ that allows setting of 500 ms or more and has the highest resolution.

<3> Setting of CR10 and CR20

$$500 \text{ ms} = \frac{N + 1}{8.38 \text{ MHz}/2^6}$$

$N = 500 \text{ ms} \times 8.38 \text{ MHz}/2^6 - 1 \doteqdot 65468 = \text{FF6CH}$

CR10 = 6CH, CR20 = FFH

(1) Program list

```
TCL1 = #00001010B
CR10 = #06CH          ;  Sets 65468 to CR10 and CR20
CR20 = #0FFH          ;  CR10 = 6CH, CR20 = FFH
TMC1 = #00000111B
```

(b) To set interval of 10 s

<1> Setting of TMC1
Select the 16-bit timer register × 1 channel mode and enable the operation of the 8-bit timers 1 and 2.

<2> Setting of TCL1
Select $f_X/2^{12}$ that allows setting of 10 s or more and has the highest resolution.

<3> Setting of CR10 and CR20

$$10 \text{ s} = \frac{N + 1}{8.38 \text{ MHz}/2^{12}}$$

$N = 10 \text{ s} \times 8.38 \text{ MHz}/2^{12} - 1 \fallingdotseq 20458 = \text{4FEAH}$

CR10 = EAH, CR20 = 4FH

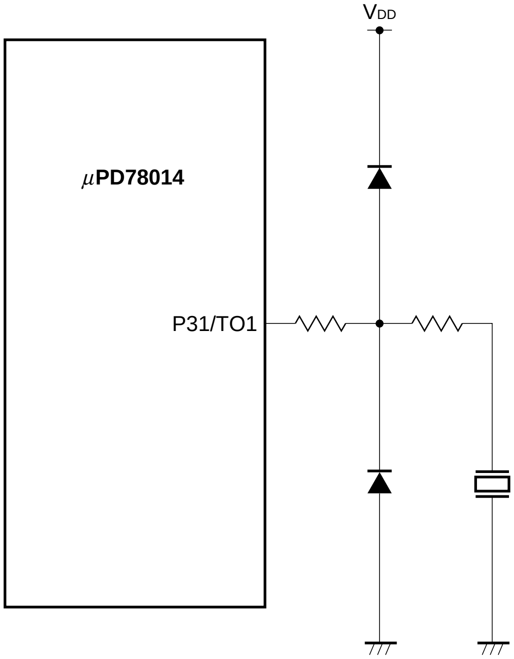
(1) Program list

```
TCL1 = #00001111B
CR10 = #0EAH          ;  Sets 20458 to CR10 and CR20
CR20 = #4FH           ;  CR10 = EAH, CR20 = 4FH
TMC1 = #00000111B
```

6.2 Musical Scale Generation

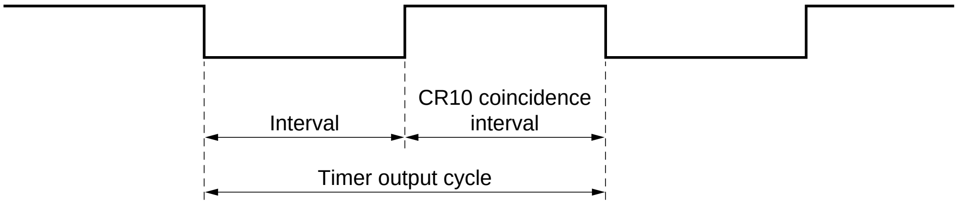
This section shows an example of a program that uses the square wave output (P31/TO1) of an 8-bit timer/event counter of the μPD78014 subseries and generates a musical scale by supplying pulses to an external buzzer.

Figure 6-16. Musical Scale Generation Circuit



The output frequency of the P31/TO1 pin is set by the count clock and a compare register. In this example, the central frequency of the musical scale is set to a range of 523 to 1046 Hz. Therefore, $f_X/2^6$ is selected as the count clock. Table 6-1 shows the musical scale, the set value of the compare register, and frequency of the output pulse. Because one cycle of the timer output is created when the value of the timer coincides with the value of the compare register two times, the interval time is set as half a cycle time.

Figure 6-17. Timer Output and Interval



As for the time length of a sound, the output time is determined by setting an interval time with 8-bit timer/event counter 2 and by counting the number of times the interrupt generated by the timer/event counter. In this example, 8-bit timer/event counter 2 is set to 20 ms.

Table 6-1. Musical Scale and Frequency

Musical Scale	Musical Scale Frequency Hz	Compare Register Value	Output Frequency Hz
Do	523.25	124	524.3
Re	587.33	111	585.1
Mi	659.25	98	662.0
Fa	698.46	93	697.2
So	783.98	83	780.2
La	880.00	73	885.6
Tee	987.77	65	993.0
Do	1046.5	62	1040

The format of the data table for this program is shown below.

```
TABLE:
        DB  musical scale data 1, sound length data 1
        DB  musical scale data 2, sound length data 2
                    ⋮                   ⋮
        DB  musical scale data n, sound length data n
        DB  0,                    0
```

The musical scale data is set to 0 for rest, and the sound length data is set to 0 for the end of data.

Example Number of counts of 8-bit timer/event counter to output sound for 1 second
Number of counts = 1 s/20 ms = 50 (50 is set as number of counts)

This program sequentially outputs do, re, mi, and so on, for 1 second each.

(1) Description of package

<Public declaration symbol>

MLDY: Subroutine name of musical scale generation program

<Registers used>

Bank 0: A, B, HL

<RAM used>

Name	Usage	Attribute	Bytes
POINT	Stores pointer value of table data	SADDR	1
LNG	Counts sound length data		

<Nesting>

1 level 3 bytes

<Hardware used>

- 8-bit timer/event counters 1 and 2
- P31/TO1

<Initial setting>

- Sets by subroutine MLDY
- Enables interrupt

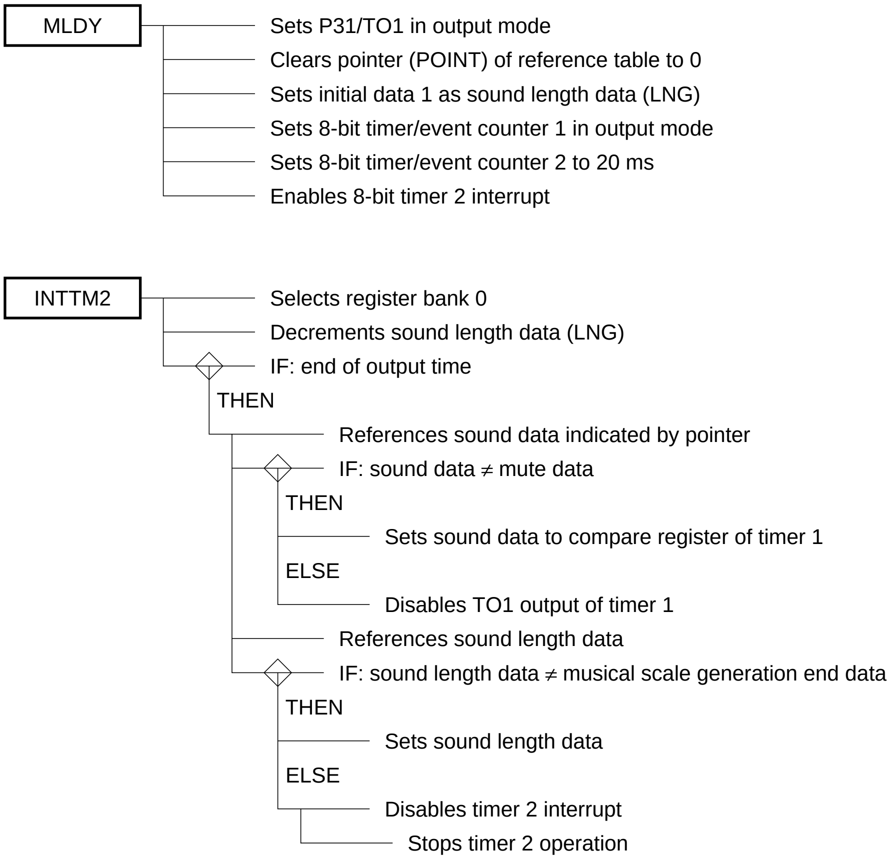
<Starting>

- Call subroutine MLDY

(2) Example of use

```
EXTRN  MLDY
   ⋮
CALL   !MLDY
EI
```

(3) SPD chart



```
MLDY
    Sets P31/TO1 in output mode
    Clears pointer (POINT) of reference table to 0
    Sets initial data 1 as sound length data (LNG)
    Sets 8-bit timer/event counter 1 in output mode
    Sets 8-bit timer/event counter 2 to 20 ms
    Enables 8-bit timer 2 interrupt

INTTM2
    Selects register bank 0
    Decrements sound length data (LNG)
    IF: end of output time
    THEN
        References sound data indicated by pointer
        IF: sound data ≠ mute data
        THEN
            Sets sound data to compare register of timer 1
        ELSE
            Disables TO1 output of timer 1
        References sound length data
        IF: sound length data ≠ musical scale generation end data
        THEN
            Sets sound length data
        ELSE
            Disables timer 2 interrupt
            Stops timer 2 operation
```

(4) Program list

```
       PUBLIC MLDY

VETM2  CSEG   AT 18H
       DW     INTTM2                         ; Sets vector address of 8-bit timer/event counter

ML_DAT DSEG   SADDR
POINT: DS     1                              ; Pointer for table data
LNG:   DS     1                              ; Sound length data

;**********************************************
;*     Musical scale generation initialize
;**********************************************
ML_SEG CSEG
MLDY:
       CLR    PM3.1                          ; Sets P3.1 in output mode
       POINT=#0                              ; Initial setting of pointer
       LGN=#1
       OSMS=#00000001B                       ; Does not use divider circuit
       TOC1=#00000011B                       ; Sets TO1 output mode
       TCL1=#11101010B
       CR20=#163                             ; Sets timer 2 to 20 ms
       TMC1=#00000010B                       ; Enables timer 2 operation
       CLR1   TMMK2                          ; Enables timer 2 interrupt
       RET
$EJECT
```

```
;**********************************************
;      Sets musical scale generation data
;**********************************************
TM2_SEG CSEG
INTTM2:
       SEL RB0
       LNG––
       if(LNG==#0)
          B=POINT (A)
          HL=#TABLE                               ; Sets table first address
          A=[HL+B]
          if(A!=#0)
             CLR1   TCE1                          ; Sets sound data
             CR10=A
             SET1   TOE1
             SET1   TCE1
          else
             CLR1   TOE1
          endif

          B++                                     ; Increments pointer
          A=[HL+B]                                ; Loads sound length data
          if(A!=#0)                               ; Sound output in progress?
             LNG=A                                ; Sets sound length data
             B++
             POINT=B (A)
          else
             SET1   TMMK2                         ; Disables timer 2 interrupt
             CLR1   TCE2                          ; Stops timer 2 operation
          endif
       endif
       RETI
;**********************************************
;          Musical scale data table
;**********************************************
TABLE:
       DB  124,50                                 ; Do
       DB  111,50                                 ; Re
       DB  98,50                                  ; Mi
       DB  93,50                                  ; Fa
       DB  83,50                                  ; So
       DB  73,50                                  ; La
       DB  65,50                                  ; Tee
       DB  62,50                                  ; Do
       DB  00,00                                  ; End
```

[MEMO]

CHAPTER 7 APPLICATIONS OF WATCH TIMER

The watch timer of the 78K/0 series has a watch timer function that causes the timer to overflow every 0.5 second by using the main system clock or subsystem clock as the clock source, and an interval timer function that allows you to set six types of reference times. These two functions can be simultaneously used.

The watch timer is set by using the following registers.

- Timer clock select register 2 (TCL2) : μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries
- Watch timer mode control register (TMC2) : μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries
- Watch timer mode control register(WTM) : μPD780024, 780024Y, 780034, 780034Y subseries

★ **Caution The format of the registers provided on the μPD780024, 780024Y, 780034, and 780034Y subseries differs from the format of the registers used in the program examples in this chapter. When using a program example in this chapter with any of the μPD780024, 780024Y, 780034, and 780034Ysubseries, change the setting of the registers according to the registers of the microcontroller used.**

Figure 7-1. Format of Timer Clock Select Register 2
(µPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL2	TCL27	TCL26	TCL25	TCL24	0	TCL22	TCL21	TCL20	FF42H	00H	R/W

TCL22	TCL21	TCL20	Selects count clock of watchdog timer
0	0	0	$f_X/2^4$ (625 kHz)
0	0	1	$f_X/2^5$ (313 kHz)
0	1	0	$f_X/2^6$ (156 kHz)
0	1	1	$f_X/2^7$ (78.1 kHz)
1	0	0	$f_X/2^8$ (39.1 kHz)
1	0	1	$f_X/2^9$ (19.5 kHz)
1	1	0	$f_X/2^{10}$ (9.8 kHz)
1	1	1	$f_X/2^{12}$ (2.4 kHz)

TCL24	Selects count clock of watch timer
0	$f_X/2^8$ (39.1 kHz)
1	f_{XT} (32.768 kHz)

TCL27	TCL26	TCL25	Selects frequency of buzzer output
0	×	×	Disables buzzer output
1	0	0	$f_X/2^{10}$ (9.8 kHz)
1	0	1	$f_X/2^{11}$ (4.9 kHz)
1	1	0	$f_X/2^{12}$ (2.4 kHz)
1	1	1	Setting prohibited

Caution Before writing new data to TCL2, stop the timer operation once.

Remarks 1. f_X : main system clock oscillation frequency

2. f_{XT} : subsystem clock oscillation frequency

3. × : don't care

4. ( ) : at f_X = 10.0 MHz or f_{XT} = 32.768 kHz

Figure 7-2. Format of Watch Timer Mode Control Register (μPD78002, 78002Y, 78014, 78014Y, 78018F, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TMC2	0	TMC26	TMC25	TMC24	TMC23	TMC22	TMC21	TMC20	FF4AH	00H	R/W

TMC23	TMC20	Selects set time of watch flag
0	0	$2^{14}/f_W$ (0.5s)
1		$2^{13}/f_W$ (0.25s)
0	1	$2^{5}/f_W$ (977 μs)
1		$2^{4}/f_W$ (488 μs)

TMC21	Controls operation of prescaler
0	Clears after operation stopped
1	Enables operation

TMC22	Controls operation of 5-bit counter
0	Clears after operation stopped
1	Enables operation

TMC26	TMC25	TMC24	Selects interval time of prescaler
0	0	0	$2^{4}/f_W$ (488 μs)
0	0	1	$2^{5}/f_W$ (977 μs)
0	1	0	$2^{6}/f_W$ (1.95 ms)
0	1	1	$2^{7}/f_W$ (3.91 ms)
1	0	0	$2^{8}/f_W$ (7.81 ms)
1	0	1	$2^{9}/f_W$ (15.6 ms)
Others			Setting prohibited

Caution Do not often clear the prescaler when the watch timer is used.

Remarks 1. f_W : watch timer clock frequency ($f_X/2^8$ or f_{XT})

2. ( ): at f_W = 32.768 kHz

★ **Figure 7-3. Format of Watch Timer Mode Control Register (μPD780024, 780024Y, 780034, 780034Y subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
WTM	WTM7	WTM6	WTM5	WTM4	0	0	WTM1	WTM0	FF41H	00H	R/W

WTM0	Enables watch timer operation
0	Stops operation (clears both prescaler and timer)
1	Enables operation

WTM1	Controls 5-bit counter operation
0	stops and clears
1	Starts

WTM6	WTM5	WTM4	Selects interval timer of prescaler
0	0	0	$2^4/f_W$ (488 μs)
0	0	1	$2^5/f_W$ (977 μs)
0	1	0	$2^6/f_W$ (1.95 ms)
0	1	1	$2^7/f_W$ (3.91 ms)
1	0	0	$2^8/f_W$ (7.81 ms)
1	0	1	$2^9/f_W$ (15.6 ms)
Others			Setting prohibited

WTM7	Selects count clock of watch timer
0	$f_X/2^7$ (65.4 kHz)
1	f_{XT} (32.768 kHz)

Remarks 1. f_W : Watch timer clock frequency ($f_X/2^7$ or f_{XT})

2. f_X : main system clock oscillation frequency

3. f_{XT}: subsystem clock oscillation frequency

4. ( ): f_X = 8.38 MHz or f_W = 32.768 kHz

7.1 Watch and LED Display Program

As an example of using the watch timer of the μPD78014 subseries, this section introduces a program that counts time by using an 0.5 second overflow and dynamically displays LED at intervals of 1.95 ms.

To count time, an overflow flag is tested each time a subroutine is called. When the flag is set, time is counted up in seconds. Because an overflow occurs every 0.5 second, it takes 1 minute to count 120 times. The overflow flag is tested at intervals of 1.95 ms so that the flag is tested without fail. The watch of this program is 24-hour watch. The high-order and low-order digits of minute and hour data are stored in separate areas of memory.

Figure 7-4. Concept of Watch Data

Second data	Minute data		Hour data	
0-120	Low-order digit 0-9	High-order digit 0-5	Low-order digit 0-9	High-order digit 0-2

As LED dynamic display, four digits are displayed with the display digit changed at intervals of 1.95 ms. In this example, the high-order 4 bits of P3 are used as a digit signal, and P5 that can directly drive an LED is selected as a segment signal.

The digit of an LED specified by a display digit area (DIGCT) in an LED display area is displayed. To change the digit signal, the segment signal is turned off so that the adjacent digits are not displayed.

Figure 7-5. LED Display Timing

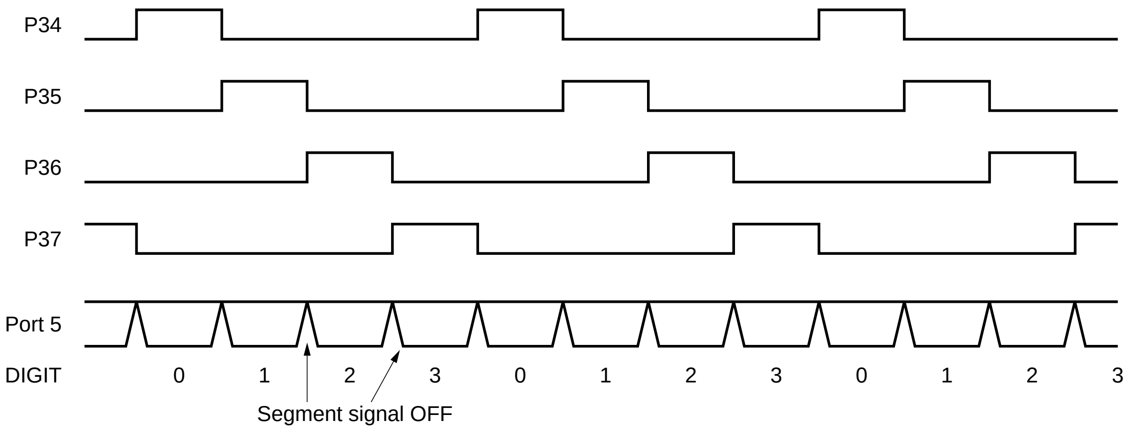
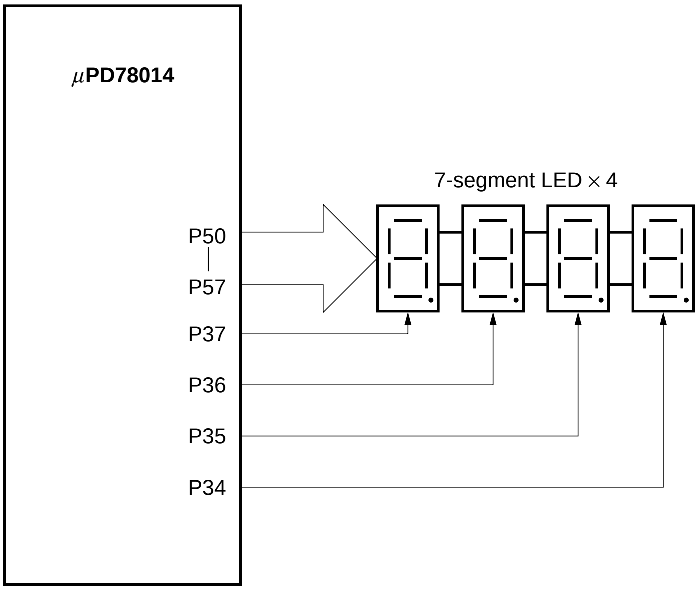


Figure 7-6. Circuit Example of Watch Timer



(1) Description of package

<Public declaration symbol>

SECD : second data storage area
MINDP : minute data storage area
HOURDP : hour data storage area
LEDDP : LED display area

<Register used>

Bank 0: AX, B, HL

<RAM used>

Name	Usage	Attribute	Bytes
MINDP	Stores minute data	SADDRP	2
HOURDP	Stores hour data		
SECD	Stores second data		1
DIGCT	Stores LED display digit data		
LEDDP	LED display data		4

<Hardware used>

- Watch timer
- P34-37
- P5

<Initial setting>

- 0.5-second watch operation at 1.95 ms interval TMC2 = #00100110B
- Enables watch timer interrupt TMMK3 = 0

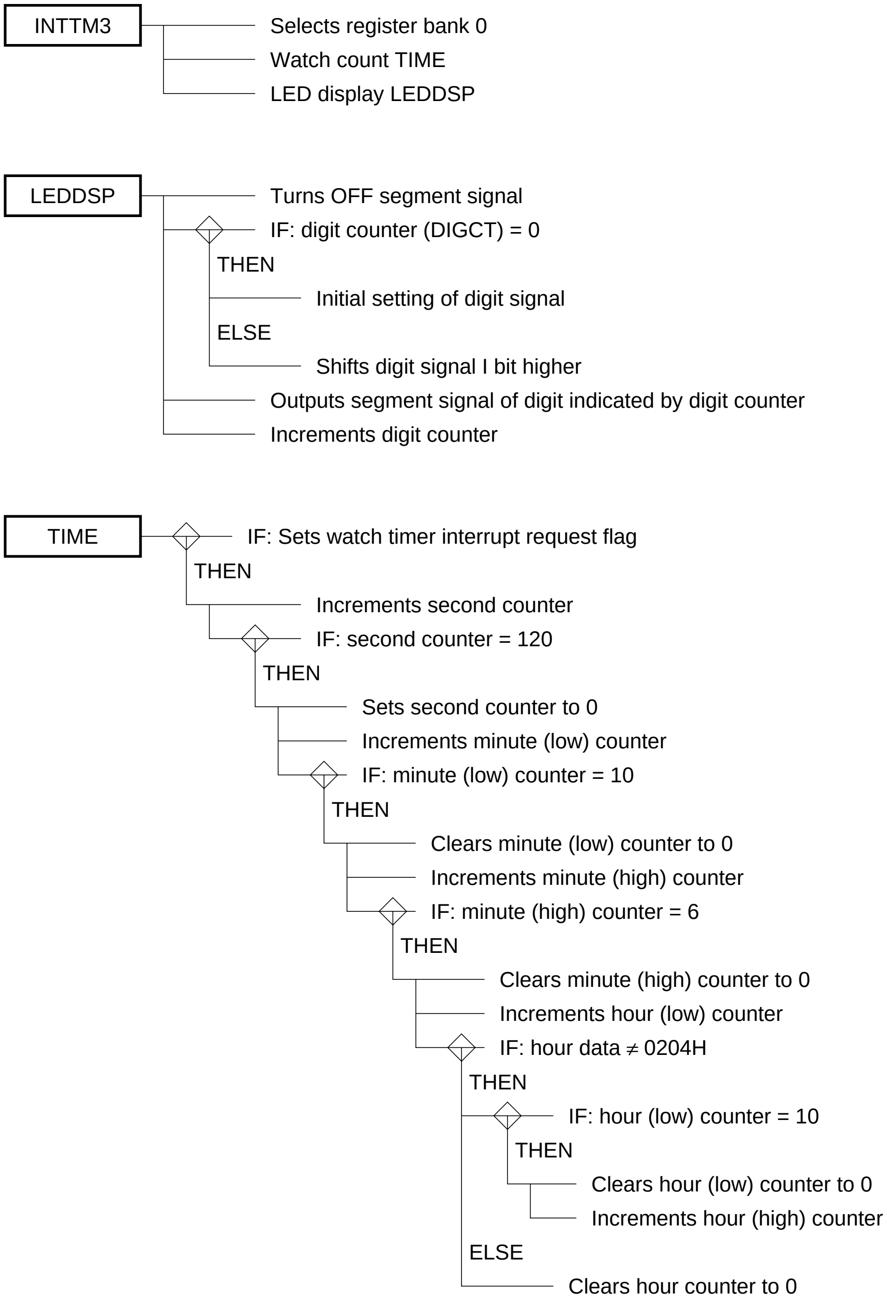
<Starting>

Started by the interval timer interrupt request of the watch timer.

(2) Example of use

```
EXTRN  MINDP, HOURDP, SECD, LEDDP

TMC2 = #00100110B   ;  0.5-second watch operation at 1.95 ms interval
CLR1    TMMK3       ;  Enables watch timer interrupt
EI
```

(3) SPD chart



```
INTTM3
    Selects register bank 0
    Watch count TIME
    LED display LEDDSP

LEDDSP
    Turns OFF segment signal
    IF: digit counter (DIGCT) = 0
    THEN
        Initial setting of digit signal
    ELSE
        Shifts digit signal I bit higher
    Outputs segment signal of digit indicated by digit counter
    Increments digit counter

TIME
    IF: Sets watch timer interrupt request flag
    THEN
        Increments second counter
        IF: second counter = 120
        THEN
            Sets second counter to 0
            Increments minute (low) counter
            IF: minute (low) counter = 10
            THEN
                Clears minute (low) counter to 0
                Increments minute (high) counter
                IF: minute (high) counter = 6
                THEN
                    Clears minute (high) counter to 0
                    Increments hour (low) counter
                    IF: hour data ≠ 0204H
                    THEN
                        IF: hour (low) counter = 10
                        THEN
                            Clears hour (low) counter to 0
                            Increments hour (high) counter
                    ELSE
                        Clears hour counter to 0
```

(4) Program list

```
        PUBLIC HOURDP,MINDP,SECD,LEDDP

WT_DATP DSEG    SADDRP
MINDP:  DS      2                         ; Minute data storage area
HOURDP: DS      2                         ; Hour data storage area
SECD:   DS      1                         ; Second data storage area
DIGCT:  DS      1                         ; LED display digit area
LEDDP:  DS      4                         ; LED display area

VETM3   CSEG    AT 1EH
        DW      INTTM3                    ; Sets vector address of watch timer

;***************************************
;*    Interval interrupt processing
;***************************************
TM3_SEG CSEG
INTTM3:
        SEL RB0
        CALL    !TIME
        CALL    !LEDDPSP
        RETI
```

```
;************************************
;           LED display
;************************************
LEDDPSP:
        P5=#0FFH                      ; Turns OFF segment output
        DIGCT&=#00000011B             ; Adjusts digit counter (0-3)
        if(DIGCT==#0)
            A=P3
            A&=#00001111B             ; Initial setting of digit signal (high-order 4 bits)
            A|=#00010000B
            P3=A
        else
            A=P3
            A&=#11110000B             ; Shifts high-order 4 bits
            X=A
            A=P3
            A+=X
            P3=A
        endif

        B=DIGCT (A)                   ; Sets address of display data
        HL=#LEDDP                     ; Display area first address
        B=[HL+B] (A)                  ; Sets display data
        HL=#SEGDT                     ; Conversion to segment data
        P5=[HL+B] (A)                 ; Outputs segment signal

        DIGCT++
        RET
SEGDT:
        DB  11000000B                 ; 0
        DB  11111001B                 ; 1
        DB  10100100B                 ; 2
        DB  10110000B                 ; 3
        DB  10011001B                 ; 4
        DB  10010010B                 ; 5
        DB  10000010B                 ; 6
        DB  11111000B                 ; 7
        DB  10000000B                 ; 8
        DB  10010000B                 ; 9
        DB  10001000B                 ; A
        DB  10000011B                 ; B
        DB  11000110B                 ; C
        DB  10100001B                 ; D
        DB  10000110B                 ; E
        DB  10001110B                 ; F
$EJECT
```

```
;********************************
;*       Watch count up
;********************************
TIME:                                           ; 0.5 second test
        if_bit(WTIF)
            CLR1       WTIF                     ; 120 = 60 seconds/0.5
            SECD++
            if(SECD==#120)
                SECD=#0                         ; Increments minute (low)
                (MINDP+0)++                     ; Carry occurs
                if((MINDP+0)==#10)
                    (MINDP+0)=#0                ; Increments minute (high)
                    (MINDP+1)++                 ; Carry occurs
                    if(MINDP+1==#6)
                        (MINDP+1)=#0
                        (HOURDP+0)++            ; Hour data 24?
                        if(HOURDP!=#0204H) (AX) ; Carry occurs
                            if((HOURDP+0)==#10)
                                (HOURDP+0)=#0
                                (HOURDP+1)++
                            endif
                        else
                            HOURDP=#0000H
                        endif
                    endif
                endif
            endif
        endif
        RET
```

[MEMO]

CHAPTER 8 APPLICATIONS OF SERIAL INTERFACE

The 78K/0 series is provided with the serial interface shown in Table 8-1.

★ **Table 8-1. Serial Interface Channel of Each Subseries**

Configuration of Serial Interface Subseries	Channel 0				Channel 1		SIO3	UART0
	3-wire	2-wire	SBI	I^2C bus	3-wire	3-wire with automatic transmit/ receive function	3-wire	async. serial interface
μPD78002	○	○	○	×	×	×	×	×
μPD78002Y	○	○	○	○	×	×	×	×
μPD78014	○	○	○	×	○	○	×	×
μPD78014Y	○	○	○	○	○	○	×	×
μPD78018F	○	○	○	×	○	○	×	×
μPD78018FY	○	○	×	○	○	○	×	×
μPD780001	×	×	×	×	○	×	×	×
μPD780024	×	×	×	×	×	×	○	○
μPD780024Y	×	×	×	×	×	×	○	○
μPD780034	×	×	×	×	×	×	○	○
μPD780034Y	×	×	×	×	×	×	○	○
μPD78014H	○	○	○	×	○	○	×	×
μPD780924	×	×	×	×	×	×	×	○
μPD780964	×	×	×	×	×	×	×	○

Remark ○ : Function provided, ×: Function not provided

The functions and operations of the serial interface are specified by using the following registers:

★ **Table 8-2. Registers of Serial Interface**

Serial Interface	Register Used
Channel 0	• Timer clock select register (TCL3) • Serial operating mode register 0 (CSIM0) • Serial bus interface control register (SBIC) • Interrupt timing specification register (SINT)
Channel 1	• Timer clock select register (TCL3) • Serial operating mode register 1 (CSIM1) • Automatic data transmit/receive control register (ADTC) • Automatic data transmit/receive interval specification register (ADTI)

Remark This chapter describes the register formats and application examples of serial interface channels 0, 1, and 2. For details of the register formats of serial interface SIO3 and UART0, refer to the User's Manual of each subseries.

Figure 8-1. Format of Timer Clock Select Register 3 (μPD78002 subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL3	1	0	0	0	TCL33	TCL32	TCL31	TCL30	FF43H	88H	R/W

TCL33	TCL32	TCL31	TCL30	Selects serial clock of serial interface channel 0
0	1	1	0	$f_X/2^2$ Note
0	1	1	1	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^7$ (78.1 kHz)
1	1	0	0	$f_X/2^8$ (39.1 kHz)
1	1	0	1	$f_X/2^9$ (19.5 kHz)
Others				Setting prohibited

Note Can be set only when the main system clock frequency is 4.19 MHz or less.

Cautions 1. Be sure to set bit 7 to 1, and bits 6 through 4 to 0.

2. Before writing new data to TCL3, stop serial transfer once.

Remarks 1. f_X : main system clock oscillation frequency

2. ( ) : at f_X = 10.0 MHz

Figure 8-2. Format of Timer Clock Select Register 3 (μPD78002Y subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL3	1	0	0	0	TCL33	TCL32	TCL31	TCL30	FF43H	88H	R/W

TCL33	TCL32	TCL31	TCL30	Selects serial clock of serial interface channel 0	
				Serial clock in I^{2}C bus mode	Serial clock in 3-wire serial I/O/SBI/2-wire serial I/O mode
0	1	1	0	$f_X/2^6$ (156 kHz)	$f_X/2^2$ **Note**
0	1	1	1	$f_X/2^7$ (78.1 kHz)	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^8$ (39.1 kHz)	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^9$ (19.5 kHz)	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^{10}$ (9.8 kHz)	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^{11}$ (4.9 kHz)	$f_X/2^7$ (78.1 kHz)
1	1	0	0	$f_X/2^{12}$ (2.4 kHz)	$f_X/2^8$ (39.1 kHz)
1	1	0	1	$f_X/2^{13}$ (1.2 kHz)	$f_X/2^9$ (19.5 kHz)
Others				Setting prohibited	

Note Can be set only when the main system clock frequency is 4.19 MHz or less.

Cautions 1. Be sure to set bit 7 to 1, and bits 6 through 4 to 0.

2. Before writing new data to TCL3, stop serial transfer once.

Remarks 1. f_X : main system clock oscillation frequency

2. ( ) : at f_X = 10.0 MHz

Figure 8-3. Format of Timer Clock Select Register 3 (μPD78014, 78018F, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL3	TCL37	TCL36	TCL35	TCL34	TCL33	TCL32	TCL31	TCL30	FF43H	88H	R/W

TCL33	TCL32	TCL31	TCL30	Selects serial clock of serial interface channel 0
0	1	1	0	$f_X/2^2$ Note
0	1	1	1	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^7$ (78.1 kHz)
1	1	0	0	$f_X/2^8$ (39.1 kHz)
1	1	0	1	$f_X/2^9$ (19.5 kHz)
Others				Setting prohibited

TCL37	TCL36	TCL35	TCL34	Selects serial clock of serial interface channel 1
0	1	1	0	$f_X/2^2$ Note
0	1	1	1	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^7$ (78.1 kHz)
1	1	0	0	$f_X/2^8$ (39.1 kHz)
1	1	0	1	$f_X/2^9$ (19.5 kHz)
Others				Setting prohibited

Note Can be set only when the main system clock frequency is 4.19 MHz or less.

Caution Before writing new data to TCL3, stop serial transfer once.

Remarks 1. f_X : main system clock oscillation frequency

2. ( ) : at f_X = 10.0 MHz

Figure 8-4. Format of Timer Clock Select Register 3 (μPD78014Y, 78018FY subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL3	TCL37	TCL36	TCL35	TCL34	TCL33	TCL32	TCL31	TCL30	FF43H	88H	R/W

TCL33	TCL32	TCL31	TCL30	Selects serial clock of serial interface channel 0	
				Serial clock in I^2C bus mode	Serial clock in 3-wire serial I/O/SBI/ 2-wire serial I/O mode Note 1
0	1	1	0	$f_X/2^6$ (156 kHz)	$f_X/2^2$ Note 2
0	1	1	1	$f_X/2^7$ (78.1 kHz)	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^8$ (39.1 kHz)	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^9$ (19.5 kHz)	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^{10}$ (9.8 kHz)	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^{11}$ (4.9 kHz)	$f_X/2^7$ (78.1 kHz)
1	1	0	0	$f_X/2^{12}$ (2.4 kHz)	$f_X/2^8$ (39.1 kHz)
1	1	0	1	$f_X/2^{13}$ (1.2 kHz)	$f_X/2^9$ (19.5 kHz)
Others				Setting prohibited	

TCL37	TCL36	TCL35	TCL34	Selects serial clock of serial interface channel 1
0	1	1	0	$f_X/2^2$ Note 2
0	1	1	1	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^7$ (78.1 kHz)
1	1	0	0	$f_X/2^8$ (39.1 kHz)
1	1	0	1	$f_X/2^9$ (19.5 kHz)
Others				Setting prohibited

Notes 1. SBI mode is not provided for the μPD78018FY subseries.

2. Can be set only when the main system clock frequency is 4.19 MHz or less.

Caution Before writing new data to TCL3, stop serial transfer once.

Remarks 1. f_X : main system clock oscillation frequency

2. ( ) : at f_X = 10.0 MHz

★ **Figure 8-5. Format of Timer Clock Select Register 3 (μPD780001)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
TCL3	TCL37	TCL36	TCL35	TCL34	1	0	0	0	FF43H	88H	R/W

TCL37	TCL36	TCL35	TCL34	Selects serial clock of serial interface channel 1
0	1	1	0	$f_X/2^2$ Note
0	1	1	1	$f_X/2^3$ (1.25 MHz)
1	0	0	0	$f_X/2^4$ (625 kHz)
1	0	0	1	$f_X/2^5$ (313 kHz)
1	0	1	0	$f_X/2^6$ (156 kHz)
1	0	1	1	$f_X/2^7$ (78.1 kHz)
1	1	0	0	$f_X/2^8$ (39.1 kHz)
1	1	0	1	$f_X/2^9$ (19.5 kHz)
Others				Setting prohibited

Note Can be set only when the main system clock frequency is 4.19 MHz or less.

Caution Before writing new data to TCL3, stop serial transfer once.

Remarks 1. f_X : main system clock oscillation frequency

2. ( ) : at f_X = 10.0 MHz

Figure 8-6. Format of Serial Operating Mode Register 0 (μPD78002, 78014, 78018F, 78014H subseries) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
CSIM0	CSIE0	COI	WUP	CSIM04	CSIM03	CSIM02	CSIM01	CSIM00	FF60H	00H	R/W$^{\text{Note 1}}$

R/W

CSIM01	CSIM00	Selects clock of serial interface channel 0
0	×	Clock externally input to $\overline{\text{SCK0}}$ pin
1	0	Output of 8-bit timer register 2 (TM2)
1	1	Clock specified by bits 0 through 3 of timer clock select register 3 (TCL3)

R/W

CSIM04	CSIM03	CSIM02	PM25	P25	PM26	P26	PM27	P27	Operating mode	First bit	Function of SI0/SB0/P25 pin	Function of SO0/SB1/P26 pin	Function of $\overline{\text{SCK0}}$/P27 pin
0	×	0	1	×	0	0	0	1	3-wire serial I/O mode	MSB	SI0$^{\text{Note 2}}$ (input)	SO0 (CMOS output)	$\overline{\text{SCK0}}$ (CMOS I/O)
		1								LSB			
1	0	0	Note 3 ×	Note 3 ×	0	0	0	1	SBI mode	MSB	P25 (CMOS I/O)	SB1 (N-ch open drain I/O)	$\overline{\text{SCK0}}$ (CMOS I/O)
		1	0	0	Note 3 ×	Note 3 ×	0	1			SB0 (N-ch open drain I/O)	P26 (CMOS I/O)	
1	1	0	Note 3 ×	Note 3 ×	0	0	0	1	2-wire serial I/O mode	MSB	P25 (CMOS I/O)	SB1 (N-ch open drain I/O)	$\overline{\text{SCK0}}$ (N-ch open drain I/O)
		1	0	0	Note 3 ×	Note 3 ×	0	1			SB0 (N-ch open drain I/O)	P26 (CMOS I/O)	

R/W

WUP	Controls wake-up function$^{\text{Note 4}}$
0	Generates interrupt request signal in all modes each time serial transfer is executed
1	Generates interrupt request signal when address received after bus has been released (when CMDD = RELD = 1) coincides with data of slave address register in SBI mode

Notes 1. Bit 6 (COI) is a read-only bit.

2. When only the transmission function is used, this pin can be used as P25 (CMOS I/O).

3. These pins can be used as port pins.

★ **4.** When using the wake-up function (WUP = 1), clear bit 5 (SIC) of the interrupt timing specification register (SINT) to 0.

★ **Caution Do not change the operating mode (3-wire serial I/O/2-wire serial I/O/SBI) while the operation of the serial interface channel 0 is enabled. To change the operating mode, stop the serial operation.**

Remark × : don't care

PM××: Port mode register

P×× : Output latch of port

Figure 8-6. Format of Serial Operating Mode Register 0 (μPD78002, 78014, 78018F, 78014H subseries) (2/2)

R	COI	Slave address comparison result flag[Note]
	0	Data of slave address register does not coincide with data of serial I/O shift register
	1	Data of slave address register coincides with data of serial I/O shift register

R/W	CSIE0	Controls operation of serial interface channel 0
	0	Stops operation
	1	Enables operation

Note COI is 0 when CSIE0 = 0.

★ **Caution Do not change the operating mode (3-wire serial I/O/2-wire serial I/O/SBI) while the operation of the serial interface channel 0 is enabled. To change the operating mode, stop the serial operation.**

Figure 8-7. Format of Serial Operating Mode Register 0 (μPD78002Y, 78014Y subseries) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
CSIM0	CSIE0	COI	WUP	CSIM04	CSIM03	CSIM02	CSIM01	CSIM00	FF60H	00H	R/W[Note 1]

R/W

CSIM01	CSIM00	Selects clock of serial interface channel 0
0	×	Clock externally input to $\overline{\text{SCK0}}$/SCL pin
1	0	Output of 8-bit timer register 2 (TM2)[Note 2]
1	1	Clock specified by bits 0 through 3 of timer clock select register 3 (TCL3)

R/W

CSIM04	CSIM03	CSIM02	PM25	P25	PM26	P26	PM27	P27	Operating mode	First bit	Function of SI0/SB0/SDA0/P25 pin	Function of SO0/SB1/SDA1/P26 pin	Function of $\overline{\text{SCK0}}$/SCL/P27 pin
0	×	0 1	1	×	0	0	0	1	3-wire serial I/O mode	MSB LSB	SI0[Note 3] (input)	SO0 (CMOS output)	$\overline{\text{SCK0}}$ (CMOS I/O)
1	0	0	Note 4 ×	Note 4 ×	0	0	0	1	SBI mode	MSB	P25 (CMOS I/O)	SB1 (N-ch open drain I/O)	$\overline{\text{SCK0}}$ (CMOS I/O)
		1	0	0	Note 4 ×	Note 4 ×	0	1			SB0 (N-ch open drain I/O)	P26 (CMOS I/O)	
1	1	0	Note 4 ×	Note 4 ×	0	0	0	1	2-wire serial I/O mode or I2C bus mode	MSB	P25 (CMOS I/O)	SB1 (N-ch open drain I/O)	$\overline{\text{SCK0}}$/SCL (N-ch open drain I/O)
		1	0	0	Note 4 ×	Note 4 ×	0	1			SB0/SDA0 (N-ch open drain I/O)	P26 (CMOS I/O)	

R/W

WUP	Controls wake-up function[Note 5]
0	Generates interrupt request signal in all modes each time serial transfer is executed
1	Generates interrupt request signal when address received after bus release (when CMDD = RELD = 1 in SBI mode, CMDD = 1 in I^2C bus mode) coincides with data of slave address register in SBI or I^2C mode

Notes 1. Bit 6 (COI) is a read-only bit.

2. In the I^2C bus mode, the clock frequency is 1/16 of the clock frequency output by TO2

3. When only the transmission function is used, this pin can be used as P25 (CMOS I/O).

4. These pins can be used as port pins.

★ **5.** When using the wake-up function (WUP = 1), clear bit 5 (SIC) of the interrupt timing specification register (SINT) to 0. While WUP = 1, do not execute an instruction that writes data to the I/O shift register 0 (SIO0).

★ **Caution Do not change the operating mode (3-wire serial I/O/SBI/2-wire serial I/O/I^2C bus) while the operation of the serial interface channel 0 is enabled. To change the operating mode, stop the serial operation.**

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

Figure 8-7. Format of Serial Operating Mode Register 0 (μPD78002Y, 780014Y subseries) (2/2)

R	COI	Slave address comparison result flag[Note]
	0	Data of slave address register does not coincide with data of serial I/O shift register
	1	Data of slave address register coincides with data of serial I/O shift register

R/W	CSIE0	Controls operation of serial interface channel 0
	0	Stops operation
	1	Enables operation

Note COI is 0 when CSIE0 = 0.

★ **Caution Do not change the operating mode (3-wire serial I/O/SBI/2-wire serial I/O/I^{2}C bus) while the operation of the serial interface channel 0 is enabled. To change the operating mode, stop the serial operation.**

Figure 8-8. Format of Serial Operating Mode Register 0 (μPD78018FY subseries) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
CSIM0	CSIE0	COI	WUP	CSIM04	CSIM03	CSIM02	CSIM01	CSIM00	FF60H	00H	R/W[Note 1]

R/W

CSIM01	CSIM00	Selects clock of serial interface channel 0
0	×	Clock externally input to $\overline{\text{SCK0}}$/SCL pin
1	0	Output of 8-bit timer register 2 (TM2)[Note 2]
1	1	Clock specified by bits 0 through 3 of timer clock select register 3 (TCL3)

R/W

CSIM04	CSIM03	CSIM02	PM25	P25	PM26	P26	PM27	P27	Operating mode	First bit	Function of SI0/SB0/SDA0/P25 pin	Function of SO0/SB1/SDA1/P26 pin	Function of $\overline{\text{SCK0}}$/SCL/P27 pin
0	×	0	1	×	0	0	0	1	3-wire serial I/O mode	MSB	SI0[Note 3] (input)	SO0 (CMOS output)	$\overline{\text{SCK0}}$ (CMOS I/O)
		1								LSB			
1	1	0	Note 4 ×	Note 4 ×	0	0	0	1	2-wire serial I/O mode or I^{2}C bus mode	MSB	P25 (CMOS I/O)	SB1 (N-ch open drain I/O)	$\overline{\text{SCK0}}$/SCL (N-ch open drain I/O)
		1	0	0	Note 4 ×	Note 4 ×	0	1			SB0/SDA0 (N-ch open drain I/O)	P26 (CMOS I/O)	

R/W

WUP	Controls wake-up function[Note 5]
0	Generates interrupt request signal in all modes each time serial transfer is executed
1	Generates interrupt request signal when address received after start condition has been detected (when CMDD = 1) coincides with data of slave address register in I^{2}C mode

Notes 1. Bit 6 (COI) is a read-only bit.

2. In the I^{2}C bus mode, the clock frequency is 1/16 of the clock frequency output by TO2

3. When only the transmission function is used, this pin can be used as P25 (CMOS I/O).

4. These pins can be used as port pins.

★ **5.** When using the wake-up function (WUP = 1), clear bit 5 (SIC) of the interrupt timing specification register (SINT) to 0. While WUP = 1, do not execute an instruction that writes data to the I/O shift register 0 (SIO0).

★ **Caution** **Do not change the operating mode (3-wire serial I/O/2-wire serial I/O/I^{2}C bus) while the operation of the serial interface channel 0 is enabled. To change the operating mode, stop the serial operation.**

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

Figure 8-8. Format of Serial Operating Mode Register 0 (μPD78018FY subseries) (2/2)

R	COI	Slave address comparison result flag[Note]
	0	Data of slave address register does not coincide with data of serial I/O shift register
	1	Data of slave address register coincides with data of serial I/O shift register

R/W	CSIE0	Controls operation of serial interface channel 0
	0	Stops operation
	1	Enables operation

Note COI is 0 when CSIE0 = 0.

★ **Caution Do not change the operating mode (3-wire serial I/O/2-wire serial I/O/I^{2}C bus) while the operation of the serial interface channel 0 is enabled. To change the operating mode, stop the serial operation.**

Figure 8-9. Format of Serial Operating Mode Register 1 (μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries only)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
CSIM1	CSIE1	DIR	ATE	0	0	0	CSIM11	CSIM10	FF68H	00H	R/W

CSIM11	CSIM10	Selects clock of serial interface channel 1
0	×	Clock externally input to $\overline{\text{SCK1}}$ pin[Note 1]
1	0	Output of 8-bit timer register 2 (TM2)
1	1	Clock specified by bits 4 through 7 of timer clock select register 3 (TCL3)

ATE	Selects operation mode of serial interface channel 1
0	3-wire serial I/O mode
1	3-wire serial I/O mode with automatic transmit/receive function

DIR	First bit	Function of SI1 pin	Function of SO1 pin
0	MSB	SI1/P20 (input)	SO1 (CMOS output)
1	LSB		

CSIE1	CSIM11	PM20	P20	PM21	P21	PM22	P22	Operation of shift register 1	Controls operation of counter of serial clock	Function of SI1/P20 pin	Function of SO1/P21 pin	Function of $\overline{\text{SCK1}}$/P22
0	×	Note 2 ×	Note 2 ×	Note 2 ×	Note 2 ×	Note 2 ×	Note 2 ×	Stops operation	Clear	P20 (CMOS I/O)	P21 (CMOS I/O)	P22 (CMOS I/O)
1	0	Note 3 1	Note 3 ×	0	0	1	×	Enables operation	Count operation	SI1[Note 3] (input)	SO1 (CMOS output)	$\overline{\text{SCK1}}$ (input)
	1					0	1					$\overline{\text{SCK1}}$ (CMOS output)

Notes 1. Clear bit 2 (STRB) and bit 1 (BUSY1) of the automatic data transmit/receive control register (ADTC) to 0, 0 when the external clock input is selected by clearing CSIM11 to 0.

2. These pins can be used as port pins.

3. When only transmission is executed, this pin can be used as P20 (CMOS I/O). (Set bit 7 (RE) of the automatic data transmit/receive control register (ADTC) to 0.)

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

★

Figure 8-10. Format of Serial Operating Mode Register 1 (μPD780001)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
CSIM1	CSIE1	DIR	0[Note 1]	0	0	0	CSIM11	CSIM10	FF68H	00H	R/W

CSIM11	CSIM10	Selects clock of serial interface channel 1
0	×	Clock externally input to $\overline{\text{SCK1}}$ pin[Note 1]
1	0	Output of 8-bit timer register 2 (TM2)
1	1	Clock specified by bits 4 through 7 of timer clock select register 3 (TCL3)

DIR	First bit	Function of SI1 pin	Function of SO1 pin
0	MSB	SI1/P20 (input)	SO1 (CMOS output)
1	LSB		

CSIE1	CSIM11	PM20	P20	PM21	P21	PM22	P22	Operation of shift register 1	Controls operation of counter of serial clock	Function of SI1/P20 pin	Function of SO1/P21 pin	Function of $\overline{\text{SCK1}}$/P22
0	×	Note 2 ×	Note 2 ×	Note 2 ×	Note 2 ×	Note 2 ×	Note 2 ×	Stops operation	Clear	P20 (CMOS I/O)	P21 (CMOS I/O)	P22 (CMOS I/O)
1	0	Note 3 1	Note 3 ×	0	0	1	×	Enables operation	Count operation	SI1[Note 3] (input)	SO1 (CMOS output)	$\overline{\text{SCK1}}$ (input)
	1					0	1					$\overline{\text{SCK1}}$ (CMOS output)

Notes 1. Be sure to clear bit 5 to 0.

2. These pins can be used as port pins.

3. When only transmission is executed, this pin can be used as P20.

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

Figure 8-11. Format of Interrupt Timing Specification Register (μPD78002, 78014 subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
SINT	0	CLD	SIC	SVAM	0	0	0	0	FF63H	00H	R/W[Note 1]

R/W

SVAM	Bits of SVA used as slave address
0	Bits 0 through 7
1	Bits 1 through 7

R/W

SIC	Selects INTCSI0 interrupt source[Note 2]
0	Sets CSIIF0 at end of transfer of serial interface channel 0
1	Sets CSIIF0 on detection of bus release

R

CLD	Level of $\overline{\text{SCK0}}$ pin[Note 3]
0	Low level
1	High level

Notes 1. Bit 6 (CLD) is a read-only bit.

2. Clear SIC to 0 when using the wake-up function.

3. CLD is 0 when CSIE0 = 0.

Caution Be sure to clear bits 0 through 3 to 0.

Remark SVA : slave address register

CSIIF0 : interrupt request flag corresponding to INTCSI0

CSIE0 : bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-12. Format of Interrupt Timing Specification Register (μPD78002Y, 78014Y subseries) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
SINT	0	CLD	SIC	SVAM	CLC	WREL	WAT1	WAT0	FF63H	00H	R/W[Note 1]

R/W	WAT1	WAT0	Controls wait and interrupt processing request
	0	0	Generates interrupt request at rising edge of 8th clock of $\overline{SCK0}$ (clock output goes into high-impedance state)
	0	1	Setting prohibited
	1	0	Used in I^{2}C bus mode (8-clock wait). Generates interrupt processing request at rising edge of 8th clock of SCL (master makes SCL output low and waits after outputting 8 clocks. Slave makes SCL pin low and requests for wait after inputting 8 clocks).
	1	1	Used in I^{2}C bus mode (9-clock wait). Generates interrupt processing request at rising edge of 9th clock of SCL (master makes SCL output low and waits after outputting 9 clocks. Slave makes SCL pin low and requests for wait after inputting 9 clocks).

R/W	WREL	Controls wait release
	0	Wait release status
	1	Releases wait status. After wait status has been released, this bit is automatically cleared to 0 (used to release wait status set by WAT1 and WAT0)

R/W	CLC	Controls clock level[Note 2]
	0	Used in I^{2}C bus mode. Makes output level of SCL pin low when serial transfer is not executed
	1	Used in I^{2}C bus mode. Makes output level of SCL pin high impedance when serial transfer is not executed (clock line goes high). Used by master to generate start/stop condition.

Notes 1. Bit 6 (CLD) is a read-only bit.

2. Clear CLC to 0 when the I^{2}C bus mode is not used.

Figure 8-12. Format of Interrupt Timing Specification Register (μPD78002Y, 78014Y subseries) (2/2)

R/W

SVAM	Bits of SVA used as slave address
0	Bits 0 through 7
1	Bits 1 through 7

R/W

SIC	Selects INTCSI0 interrupt source[Note 1]
0	Sets CSIIF0 to 1 at end of transfer of serial interface channel 0
1	Sets CSIIF0 to 1 on detection of bus release in the SBI mode, or on detection of stop condition in the I^{2}C bus mode

R/W

CLD	Level of $\overline{\text{SCK0}}$/SCL/P27 pin[Note 2]
0	Low level
1	High level

Notes 1. Clear SIC to 0 when using the wake-up function.

2. CLD is 0 when CSIE0 = 0.

Remark SVA : slave address register
CSIIF0 : interrupt request flag corresponding to INTCSI0
CSIE0 : bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-13. Format of Interrupt Timing Specification Register (μPD78018F, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
SINT	0	CLD	SIC	SVAM	0	0	0	0	FF63H	00H	R/W[Note 1]

R/W

SVAM	Bits of SVA used as slave address
0	Bits 0 through 7
1	Bits 1 through 7

R/W

SIC	Selects INTCSI0 interrupt source[Note 2]
0	Sets CSIIF0 at end of transfer of serial interface channel 0
1	Sets CSIIF0 at end of transfer of serial interface channel 0 or on detection of bus release

R

CLD	Level of $\overline{\text{SCK0}}$ pin[Note 3]
0	Low level
1	High level

Notes 1. Bit 6 (CLD) is a read-only bit.

2. Clear SIC to 0 when using the wake-up function in the SBI mode.

3. CLD is 0 when CSIE0 = 0.

Caution Be sure to clear bits 0 through 3 to 0.

Remark SVA : slave address register
CSIIF0 : interrupt request flag corresponding to INTCSI0
CSIE0 : bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-14. Format of Interrupt Timing Specification Register (μPD78018FY subseries) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
SINT	0	CLD	SIC	SVAM	CLC	WREL	WAT1	WAT0	FF63H	00H	R/W[Note 1]

R/W

WAT1	WAT0	Controls wait and interrupt processing request
0	0	Generates interrupt request at rising edge of 8th clock of $\overline{\text{SCK0}}$ (clock output goes into high-impedance state)
0	1	Setting prohibited
1	0	Used in I^{2}C bus mode (8-clock wait). Generates interrupt processing request at rising edge of 8th clock of SCL (master makes SCL output low and waits after outputting 8 clocks. Slave makes SCL pin low and requests for wait after inputting 8 clocks).
1	1	Used in I^{2}C bus mode (9-clock wait). Generates interrupt processing request at rising edge of 9th clock of SCL (master makes SCL output low and waits after outputting 9 clocks. Slave makes SCL pin low and requests for wait after inputting 9 clocks).

R/W

WREL	Controls wait release
0	Wait release status
1	Releases wait status. After wait status has been released, this bit is automatically cleared to 0 (used to release wait status set by WAT1 and WAT0)

R/W

CLC	Controls clock level[Note 2]
0	Used in I^{2}C bus mode. Makes output level of SCL pin low when serial transfer is not executed
1	Used in I^{2}C bus mode. Makes output level of SCL pin high impedance when serial transfer is not executed (clock line goes high). Used by master to generate start/stop condition.

Notes 1. Bit 6 (CLD) is a read-only bit.

2. Clear CLC to 0 when the I^{2}C bus mode is not used.

Figure 8-14. Format of Interrupt Timing Specification Register (μPD78018FY subseries) (2/2)

R/W

SVAM	Bits of SVA used as slave address
0	Bits 0 through 7
1	Bits 1 through 7

R/W

SIC	Selects INTCSI0 interrupt source[Note 1]
0	Sets CSIIF0 to 1 at end of transfer of serial interface channel 0
1	Sets CSIIF0 to 1 at end of transfer of serial interface channel 0 or on detection of stop condition in the I^2C bus mode

R/W

CLD	Level of $\overline{\text{SCK0}}$/SCL/P27 pin[Note 2]
0	Low level
1	High level

Notes 1. Clear SIC to 1 when using the wake-up function in the I^2C mode.

2. CLD is 0 when CSIE0 = 0.

Remark SVA : slave address register
CSIIF0 : interrupt request flag corresponding to INTCSI0
CSIE0 : bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-15. Format of Serial Bus Interface Control Register (μPD78002, 78014, 78018F, 78014H subseries) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
SBIC	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	FF61H	00H	R/W[Note]

R/W	RELT	Used to output bus release signal. When RELT = 1, SO latch is set to 1. After SO latch has been set, this bit is automatically cleared to 0. It is also cleared to 0 when CSIE = 0.

R/W	CMDT	Used to output command signal. When CMDT = 1, SO latch is cleared to 0. After SO latch has been cleared, this bit is automatically cleared to 0. It is also cleared to 0 when CSIE0 = 0.

R	RELD	Bus release detection
	Clear condition (RELD = 0)	Set condition (RELD = 1)
	• On execution of transfer start instruction • If values of SIO0 and SVA do not coincide when address is received • When CSIE0 = 0 • At $\overline{\text{RESET}}$ input	• When bus release signal (REL) is detected

R	CMDD	Command detection
	Clear condition (CMDD = 0)	Set condition (CMDD = 1)
	• On execution of transfer start instruction • When bus release signal (REL) is detected • When CSIE0 = 0 • At $\overline{\text{RESET}}$ input	• When command signal (CMD) is detected

R/W	ACKT	Outputs acknowledge signal in synchronization with falling edge of $\overline{\text{SCK0}}$ clock immediately after instruction that sets this bit to 1 has been executed. After acknowledge signal has been output, this bit is automatically cleared to 0. ACKE is cleared to 0. This bit is also cleared to 0 when transfer of serial interface is started and when CSIE0 = 0.

Note Bits 2, 3, and 6 (RELD, CMDD, and ACKD) are read-only bits.

Remarks 1. Bits 0, 1, and 4 (RELD, CMDT, and ACKT) are cleared to 0 when they are read after data has been set.

2. CSIE0: Bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-15. Format of Serial Bus Interface Control Register (μPD78002, 78014, 78018F, 78014H subseries) (2/2)

R/W	ACKE	Controls acknowledge signal output	
	0	Disables automatic output of acknowledge signal (output by ACKT is enabled)	
	1	Before completion of transfer	Acknowledge signal is output in synchronization with falling edge of 9th clock of $\overline{\text{SCK0}}$ (automatically output when ACKE = 1)
		After completion of transfer	Acknowledge signal is output in synchronization with falling edge of $\overline{\text{SCK0}}$ clock immediately after instruction that sets this bit to 1 has been executed (automatically output when ACKE = 1). However, this bit is not automatically cleared to 0 after acknowledge signal has been output.

R	ACKD	Acknowledge detection
	Clear condition (ACKD = 0)	Set condition (ACKD = 1)
	• Falling edge of $\overline{\text{SCK0}}$ clock immediately after busy mode has been released after execution of transfer start instruction • When CSIE0 = 0 • At $\overline{\text{RESET}}$ input	• When acknowledge signal ($\overline{\text{ACK}}$) is detected at rising edge of $\overline{\text{SCK0}}$ clock after completion of transfer

R/W	BSYE[Note]	Controls output of synchronization busy signal
	0	Disables output of busy signal in synchronization with falling edge of $\overline{\text{SCK0}}$ clock immediately after instruction that clears this bit to 0 has been executed
	1	Outputs busy signal at falling edge of $\overline{\text{SCK0}}$ clock following acknowledge signal

Note The busy mode can be released by starting serial interface transfer and receiving of an address signal. However, the BSYE flag is not cleared to 0.

Remark CSIE0: Bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-16. Format of Serial Bus Interface Control Register (μPD78002Y, 78014Y subseries) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
SBIC	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	FF61H	00H	R/W$^{\text{Note}}$

R/W	RELT	Used to output bus release signal in SBI mode. Used to output stop condition in I^{2}C bus mode. When RELT = 1, SO latch is set to 1. After SO latch has been set, this bit is automatically cleared to 0. It is also cleared to 0 when CSIE0 = 0.

R/W	CMDT	Used to ouptut command signal in SBI mode. Used to output start condition in I^{2}C bus mode. When CMDT = 1, SO latch is cleared to 0. After SO latch has been cleared, this bit is automatically cleared to 0. It is also cleared to 0 when CSIE0 = 0.

R

RELD	Stop condition detection
Clear condition (RELD = 0)	Set condition (RELD = 1)
• On execution of transfer start instruction • If values of SIO0 and SVA do not coincide when address is received • When CSIE0 = 0 • At $\overline{\text{RESET}}$ input	• When a bus release signal (REL) is detected in SBI mode • When stop condition is detected in I^{2}C bus mode

R

CMDD	Start condition detection
Clear condition (CMDD = 0)	Set condition (CMDD = 1)
• On execution of transfer start instruction • When a bus release signal (REL) is detected in SBI mode • When stop condition is detected in I^{2}C bus mode • When CSIE0 = 0 • At $\overline{\text{RESET}}$ input	• When a command signal (CMD) is detected in SBI mode • When start condition is detected in I^{2}C bus mode

R/W	ACKT	In the SBI mode, this bit outputs an acknowledge signal in synchronization with the falling edge of the $\overline{\text{SCK0}}$ clock immediately after the instruction that sets this bit to 1. After output, this bit is automatically cleared to 0. Used as ACKE = 0. Also cleared to 0 when transfer by serial interface is started and CSIE = 0. In the I^{2}C mode, this bit makes SDA0 (SDA1) low immediately after instruction that sets this bit to 1 (ACKT = 1) until next SCL falls. Used to generate $\overline{\text{ACK}}$ signal by software when 8-clock wait is selected. Cleared to 0 when transfer by serial interface is started and CSIE0 = 0.

Note Bits 2, 3, and 6 (RELD, CMDD, and ACKD) are read-only bits.

Remarks 1. Bits 0, 1, and 4 (RELD, CMDT, and ACKT) are cleared to 0 when they are read after data has been set.

2. CSIE0: Bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-16. Format of Serial Bus Interface Control Register (μPD78018FY subseries) (2/2)

R/W	ACKE	Controls acknowledge signal output (In SBI mode)	
	0	Disables automatic output of acknowledge signal (output by ACKT is enabled)	
	1	Before completion of transfer	Acknowledge signal is output in synchronization with falling edge of 9th clock of $\overline{\text{SCK0}}$ (automatically output when ACKE = 1)
		After completion of transfer	Acknowledge signal is output in synchronization with falling edge of $\overline{\text{SCK0}}$ clock immediately after instruction that sets this bit to 1 has been executed (automatically output when ACKE = 1). However, this bit is not automatically cleared to 0 after acknowledge signal has been output.

R/W	ACKE	Controls automatic output of acknowledge signal[Note 1] (In I^{2}C Bus mode)
	0	Disables automatic output of acknowledge signal (output by ACKT is enabled). Used for transmission or reception with 8-clock wait selected[Note 2].
	1	Enables automatic output of acknowledge signal. Acknowledge signal is output in synchronization with falling edge of 9th clock of SCL (automatically output when ACKE = 1). After output, this bit is not automatically cleared to 0. Used for reception when 9-clock wait is selected.

R	ACKD	Acknowledge detection
	Clear condition (ACKD = 0)	Set condition (ACKD = 1)
	• Falling edge of $\overline{\text{SCK0}}$ clock immediately after busy mode has been released after execution of transfer start instruciton in SBI mode • On execution of transfer start instruction in I^{2}C bus mode • When CSIE0 = 0 • At $\overline{\text{RESET}}$ input	• When acknowledge signal is detected at rising edge of $\overline{\text{SCK0}}$/SCL clock after completion of transfer

R/W	BSYE[Note 3]	Controls output of synchronization busy signal
	0	In the SBI mode, this bit disables output of busy signal in synchronization with falling edge of $\overline{\text{SCK0}}$/SCL clock immediately after instruction that clears this bit to 0 has been executed. Be sure to clear BSYE to 0 in the I^{2}C bus mode.
	1	In the SBI mode, this bit outputs busy signal at falling edge of $\overline{\text{SCK0}}$/SCL clock following acknowledge signal.

Notes 1. Set this bit before starting transfer.

2. Output the acknowledge signal on reception by using ACKT when 8-clock wait is selected.

3. The busy status can be released by starting transfer of serial interface or receiving an address signal. However, BSYE is not cleared to 0.

Remark CSIE0: Bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-17. Format of Serial Bus Interface Control Register (μPD78018FY subseries) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
SBIC	BSYE	ACKD	ACKE	ACKT	CMDD	RELD	CMDT	RELT	FF61H	00H	R/W[Note]

R/W	RELT	Used to output stop condition in I^{2}C bus mode. When RELT = 1, SO latch is set to 1. After SO latch has been set, this bit is automatically cleared to 0. It is also cleared to 0 when CSIE0 = 0.

R/W	CMDT	Used to output start condition in I^{2}C bus mode. When CMDT = 1, SO latch is cleared to 0. After SO latch has been cleared, this bit is automatically cleared to 0. It is also cleared to 0 when CSIE0 = 0.

R	RELD	Stop condition detection
	Clear condition (RELD = 0)	Set condition (RELD = 1)
	• On execution of transfer start instruction • If values of SIO0 and SVA do not coincide when address is received • When CSIE0 = 0 • At $\overline{\text{RESET}}$ input	• When stop condition is detected in I^{2}C bus mode

R	CMDD	Start condition detection
	Clear condition (CMDD = 0)	Set condition (CMDD = 1)
	• On execution of transfer start instruction • When stop condition is detected in I^{2}C bus mode • When CSIE0 = 0 • At $\overline{\text{RESET}}$ input	• When start condition is detected in I^{2}C bus mode

R/W	ACKT	Makes SDA0 (SDA1) low immediately after instruction that sets this bit to 1 (ACKT = 1) until next SCL falls. Used to generate $\overline{\text{ACK}}$ signal by software when 8-clock wait is selected. Cleared to 0 when transfer by serial interface is started and CSIE0 = 0

Note Bits 2, 3, and 6 (RELD, CMDD, and ACKD) are read-only bits.

Remarks 1. Bits 0, 1, and 4 (RELD, CMDT, and ACKT) are cleared to 0 when they are read after data has been set.

2. CSIE0: Bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-17. Format of Serial Bus Interface Control Register (μPD78018FY subseries) (2/2)

R/W

ACKE	Controls automatic output of acknowledge signal[Note 1]
0	Disables automatic output of acknowledge signal (output by ACKT is enabled). Used for transmission or reception with 8-clock wait selected[Note 2].
1	Enables automatic output of acknowledge signal. Acknowledge signal is output in synchronization with falling edge of 9th clock of SCL (automatically output when ACKE = 1). After output, this bit is not automatically cleared to 0. Used for reception when 9-clock wait is selected.

R

ACKD	Acknowledge detection
Clear condition (ACKD = 0)	Set condition (ACKD = 1)
• On execution of transfer start instruction • When CSIE0 = 0 • At $\overline{\text{RESET}}$ input	• When acknowledge signal is detected at rising edge of SCL clock after completion of transfer

R/W

BSYE[Note 3]	Controls transmission N-ch open drain output in I^2C bus mode[Note 4]
0	Enables output (transmission)
1	Disables output (reception)

Notes 1. Set this bit before starting transfer.

2. Output the acknowledge signal on reception by using ACKT when 8-clock wait is selected.

3. The wait status can be released by starting transfer of serial interface or receiving an address signal. However, BSYE is not cleared to 0.

4. Be sure to set BSYE to 1 when using the wake-up function.

Remark CSIE0: Bit 7 of the serial operating mode register 0 (CSIM0)

Figure 8-18. Format of Automatic Data Transmit/Receive Control Register (μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries only)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADTC	RE	ARLD	ERCE	ERR	TRF	STRB	BUSY1	BUSY0	FF69H	00H	R/W[Note 1]

R/W

BUSY1	BUSY0	Controls busy input
0	×	Does not use busy input
1	0	Enables busy input (active high)
1	1	Enables busy input (low active)

R/W

STRB	Controls strobe output
0	Disables strobe output
1	Enables strobe output

R

TRF	Status of automatic transmit/receive function[Note 2]
0	Detects end of automatic transmission/reception (0 when automatic tansmission/reception is stopped or when ARLD = 0)
1	Automatic transmission/reception in progress (1 when SIO1 is written)

R

ERR	Detects error of automatic transmit/receive function
0	No error on automatic tansmission/reception (0 when 1 is written to SIO1)
1	Error on automatic tansmission/reception

R/W

ERCE	Controls error check of automatic transmit/receive function
0	Disables error check on automatic transmission/reception
1	Enables error check on automatic transmission/reception (only when BUSY1 = 1)

R/W

ARLD	Selects operation mode of automatic transmit/receive function
0	Single mode
1	Repetitive mode

R/W

RE	Controls reception of automatic transmit/receive function
0	Disables reception
1	Enables reception

Notes 1. Bits 3 and 4 (TRF and ERR) are read-only bits.

2. Identify the end of automatic transmission/reception by using TRF instead of CSIIF1. (interrupt request flag)

Caution When external clock input is selected by clearing bit 1 (CSIM11) of the serial operating mode register 1 (CSIM1) to 0, clear bits 1 and 2 (BUSY1 and STRB) of ADTC to 0, 0.

Remark ×: don’t care

Figure 8-19. Format of Automatic Data Transmit/Receive Interval Specification Register (μPD78018F, 78018FY, 78014H subseries only) (1/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADTI	ADTI7	0	0	ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	FF6BH	00H	R/W

ADTI7	Controls interval time of data transfer
0	Does not control interval time by ADTI[Note 1]
1	Controls interval time by ADTI (ADTI0 through ADTI4)

ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	Specifies interval time of data transfer (f_X = 10.0 MHz)	
					Minimum value[Note 2]	Maximum value[Note 2]
0	0	0	0	0	18.4 μs + 0.5/f_{SCK}	20.0 μs + 1.5/f_{SCK}
0	0	0	0	1	31.2 μs + 0.5/f_{SCK}	32.8 μs + 1.5/f_{SCK}
0	0	0	1	0	44.0 μs + 0.5/f_{SCK}	45.6 μs + 1.5/f_{SCK}
0	0	0	1	1	56.8 μs + 0.5/f_{SCK}	58.4 μs + 1.5/f_{SCK}
0	0	1	0	0	69.6 μs + 0.5/f_{SCK}	71.2 μs + 1.5/f_{SCK}
0	0	1	0	1	82.4 μs + 0.5/f_{SCK}	84.0 μs + 1.5/f_{SCK}
0	0	1	1	0	95.2 μs + 0.5/f_{SCK}	96.8 μs + 1.5/f_{SCK}
0	0	1	1	1	108.0 μs + 0.5/f_{SCK}	109.6 μs + 1.5/f_{SCK}
0	1	0	0	0	120.8 μs + 0.5/f_{SCK}	122.4 μs + 1.5/f_{SCK}
0	1	0	0	1	133.6 μs + 0.5/f_{SCK}	135.2 μs + 1.5/f_{SCK}
0	1	0	1	0	146.4 μs + 0.5/f_{SCK}	148.0 μs + 1.5/f_{SCK}
0	1	0	1	1	159.2 μs + 0.5/f_{SCK}	160.8 μs + 1.5/f_{SCK}
0	1	1	0	0	172.0 μs + 0.5/f_{SCK}	173.6 μs + 1.5/f_{SCK}
0	1	1	0	1	184.8 μs + 0.5/f_{SCK}	186.4 μs + 1.5/f_{SCK}
0	1	1	1	0	197.6 μs + 0.5/f_{SCK}	199.2 μs + 1.5/f_{SCK}
0	1	1	1	1	210.4 μs + 0.5/f_{SCK}	212.0 μs + 1.5/f_{SCK}

Notes 1. The interval time is dependent on only the CPU processing.

2. The interval time of data transfer includes an error. The minimum and maximum values of the interval time for data transfer can be calculated by the following expressions (where n is the value set to ADTI0 through ADTI4). However, if the minimum value calculated by the expression below is less than 2/f_{SCK}, the minimum interval time is 2/f_{SCK}.

$$\text{Minimum value} = (n + 1) \times \frac{2^7}{f_X} + \frac{56}{f_X} + \frac{0.5}{f_{SCK}}$$

$$\text{Maximum value} = (n + 1) \times \frac{2^7}{f_X} + \frac{72}{f_X} + \frac{1.5}{f_{SCK}}$$

Cautions 1. Do not write ADTI during automatic transmission/reception operation.

2. Be sure to clear bits 5 and 6 to 0.

★ **3. When controlling interval time of data transfer by automatic transmission/reception using ADTI, the busy control option is invalid.**

Remarks 1. f_X : main system clock oscillation frequency

2. f_{SCK}: serial clock frequency

Figure 8-19. Format of Automatic Data Transmit/Receive Interval Specification Register (μPD78018F, 78018FY, 78014H subseries only) (2/2)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADTI	ADTI7	0	0	ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	FF6BH	00H	R/W

ADTI4	ADTI3	ADTI2	ADTI1	ADTI0	Specifies interval time of data transfer (f_X = 10.0 MHz)	
					Minimum value[Note]	Maximum value[Note]
1	0	0	0	0	223.2 μs + 0.5/f_{SCK}	224.8 μs + 1.5/f_{SCK}
1	0	0	0	1	236.0 μs + 0.5/f_{SCK}	237.6 μs + 1.5/f_{SCK}
1	0	0	1	0	248.8 μs + 0.5/f_{SCK}	250.4 μs + 1.5/f_{SCK}
1	0	0	1	1	261.6 μs + 0.5/f_{SCK}	263.2 μs + 1.5/f_{SCK}
1	0	1	0	0	274.4 μs + 0.5/f_{SCK}	276.0 μs + 1.5/f_{SCK}
1	0	1	0	1	287.2 μs + 0.5/f_{SCK}	288.8 μs + 1.5/f_{SCK}
1	0	1	1	0	300.0 μs + 0.5/f_{SCK}	301.6 μs + 1.5/f_{SCK}
1	0	1	1	1	312.8 μs + 0.5/f_{SCK}	314.4 μs + 1.5/f_{SCK}
1	1	0	0	0	325.6 μs + 0.5/f_{SCK}	327.2 μs + 1.5/f_{SCK}
1	1	0	0	1	338.4 μs + 0.5/f_{SCK}	340.0 μs + 1.5/f_{SCK}
1	1	0	1	0	351.2 μs + 0.5/f_{SCK}	352.8 μs + 1.5/f_{SCK}
1	1	0	1	1	364.0 μs + 0.5/f_{SCK}	365.6 μs + 1.5/f_{SCK}
1	1	1	0	0	376.8 μs + 0.5/f_{SCK}	378.4 μs + 1.5/f_{SCK}
1	1	1	0	1	389.6 μs + 0.5/f_{SCK}	391.2 μs + 1.5/f_{SCK}
1	1	1	1	0	402.4 μs + 0.5/f_{SCK}	404.0 μs + 1.5/f_{SCK}
1	1	1	1	1	415.2 μs + 0.5/f_{SCK}	416.8 μs + 1.5/f_{SCK}

Note The interval time of data transfer includes an error margin. The minimum and maximum values of the interval time for data transfer can be calculated by the following expressions (where n is the value set to ADTI0 through ADTI4). However, if the minimum value calculated by the expression below is less than 2/f_{SCK}, the minimum interval time is 2/f_{SCK}.

$$\text{Minimum value} = (n + 1) \times \frac{2^7}{f_X} + \frac{56}{f_X} + \frac{0.5}{f_{SCK}}$$

$$\text{Maximum value} = (n + 1) \times \frac{2^7}{f_X} + \frac{72}{f_X} + \frac{1.5}{f_{SCK}}$$

Cautions 1. Do not write ADTI during automatic transfer/reception operation.

2. Be sure to clear bits 5 and 6 to 0.

★ **3. When controlling interval time of data transfer by automatic transmission/reception using ADTI, the busy control option is invalid.**

Remarks 1. f_X : main system clock oscillation frequency

2. f_{SCK} : serial clock frequency

8.1 Interface with EEPROM^{TM} (μPD6252)

The μPD6252[Note] is a 2048-bit EEPROM which can be electrically written or erased. To write or read data to or from the μPD6252, the 3-wire serial interface is used.

★ **Note** μPD6252 is for maintenance use.

Figure 8-20. Pin Configuration of μPD6252

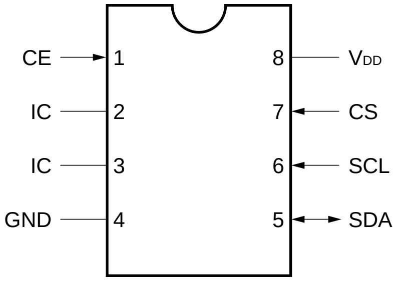
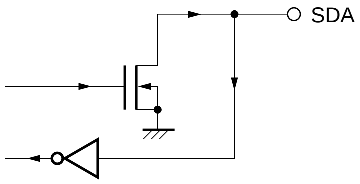


Table 8-3. Pin Function of µPD6252

Pin Number	Pin Name	I/O	Function
1	CE	CMOS input	Keep this pin high during data transfer. **Caution Do not change the level of this pin from high to low during data transfer.** To change the level of this pin from high to low, make sure that the CS pin (pin 7) is low. By making both the CE and CS pins low, you can set the standby status in which the power consumption is reduced.
2 3	IC	–	Fix the IC pins to the high or low level via resistor.
4	GND	–	Ground
5	SDA	CMOS input/ N-ch open-drain output	Data input/output pin. Because this pin is an N-ch open-drain I/O pin, externally pull it up with a resistor. 
6	SCL	CMOS input	Inputs a clock for data transfer.
7	CS	CMOS input	Chip select pin. When this pin is high, the µPD6252 is enabled to operate. When it is low, memory cells cannot be read or written. When the level of this pin is changed from high to low with the SCL pin high, the operation of the serial bus interface is started. To end the operation of the serial bus interface, change the level of this pin from high to low.
8	V_{DD}	–	Positive power: +5 V ±10%

8.1.1 Communication in 2-wire serial I/O mode

The 3-wire mode of the μPD6252[Note] is implemented by serial clock (SCL), data (SDA), and chip select (CS) lines. Excluding the handshaking line, therefore, only two lines, clock and data lines, are necessary for interfacing. To interface the μPD6252 with a 78K/0 series microcontroller, the 2-wire serial I/O mode is used. In the example shown in this section, the μPD78014 subseries is used.

★ **Note** μPD6252 is for maintenance use.

Figure 8-21. Example of Connection of μPD6252

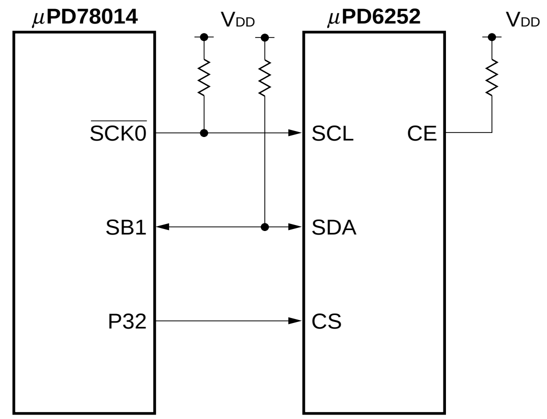
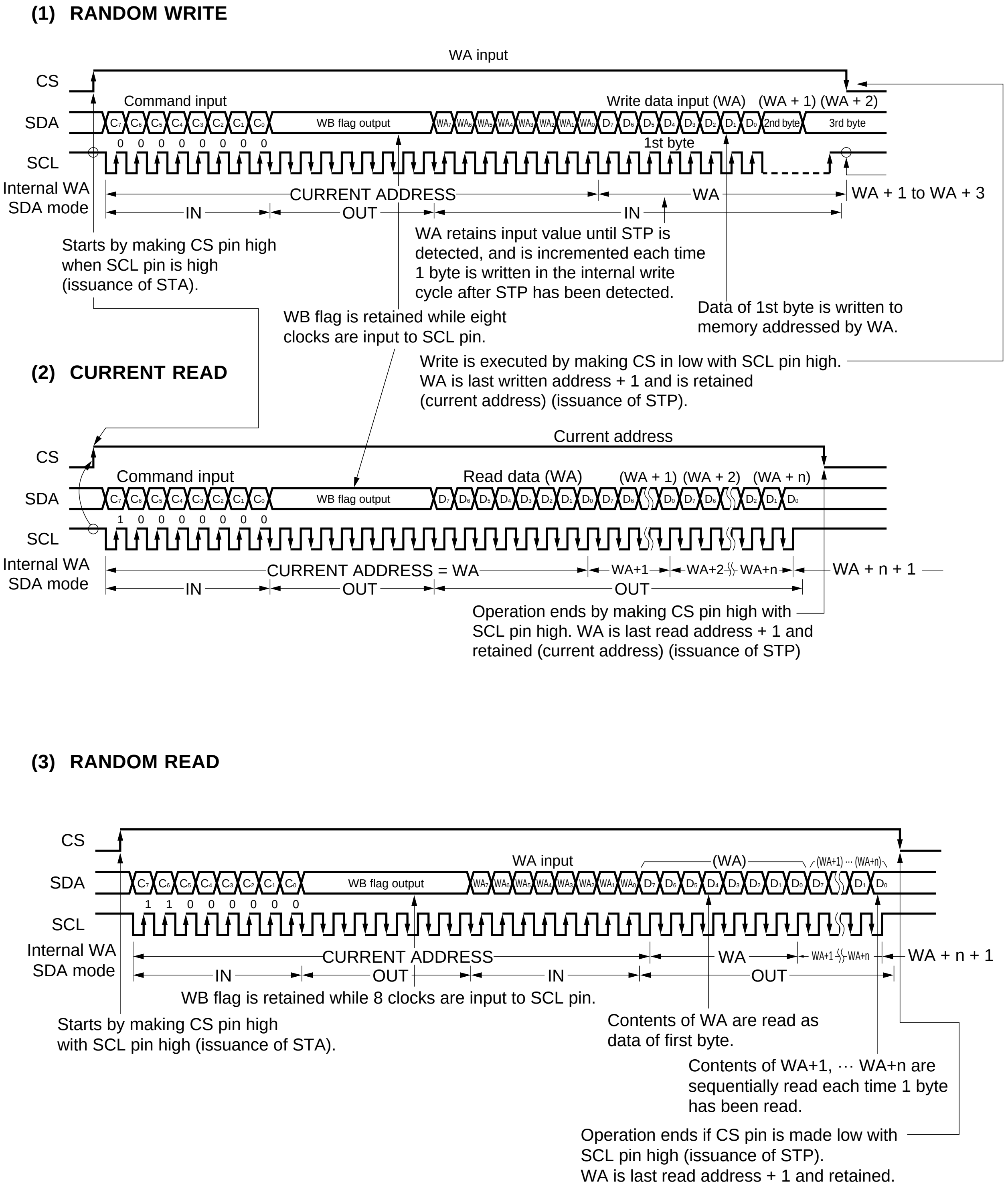


Table 8-4 and Figure 8-22 shows the commands to write and read data to/from the μPD6252 and communication format.

Table 8-4. μPD6252 Commands

Command Name	Command	Operation
RANDOM WRITE	00000000B [00H] MSB C_7-C_0	Transfers write data after setting an 8-bit word address (WA). Up to 3 bytes of write data can be set successively. Correspondence between word address and data WA Data of first byte WA+1 Data of second byte WA+2 Data of third byte The write operation is executed in the internal write cycle after the CS pin has gone low.
CURRENT READ	10000000B [80H] MSB C_7-C_0	Transfers the contents of memory specified by the word address (WA) (current address) specified when the command is set, to the read data buffer. Each time 8 bits of data have been read from the SDA pin, the word address (WA) is incremented, and the corresponding memory contents are transferred to the data buffer.
RANDOM READ	11000000B [C0H] MSB C_7-C_0	Executes data read starting from a set word address (WA) after the word address has been set. The difference from CURRENT READ is that this command sets a word address (WA) after it has been executed. After the word address has been set, this command performs the same operation as CURRENT READ.

Figure 8-22. Communication Format of μPD6252



(1) RANDOM WRITE

Starts by making CS pin high when SCL pin is high (issuance of STA).

WA retains input value until STP is detected, and is incremented each time 1 byte is written in the internal write cycle after STP has been detected.

WB flag is retained while eight clocks are input to SCL pin.

Data of 1st byte is written to memory addressed by WA.

Write is executed by making CS in low with SCL pin high. WA is last written address + 1 and is retained (current address) (issuance of STP).

(2) CURRENT READ

Operation ends by making CS pin high with SCL pin high. WA is last read address + 1 and retained (current address) (issuance of STP)

(3) RANDOM READ

WB flag is retained while 8 clocks are input to SCL pin.

Starts by making CS pin high with SCL pin high (issuance of STA).

Contents of WA are read as data of first byte.

Contents of WA+1, ··· WA+n are sequentially read each time 1 byte has been read.

Operation ends if CS pin is made low with SCL pin high (issuance of STP). WA is last read address + 1 and retained.

Steps <1> through <5> below are the operating procedure of the μPD6252. In this example, the number of data to be written or read per interface operation is fixed to 1 byte. If the μPD6252 is in the write busy (WB) status when interfaced, the busy flag is set.

<1> Make the CS pin (P32) high to start interfacing.
<2> Transmit the write or read command.
<3> Receive the data of WRITE BUSY. If interfacing the μPD6252 is enabled, 00H is received. If a code other than 00H is received, it is judged that the μPD6252 is in the WRITE BUSY status. In this case, communication is stopped.
<4> Transfer the data corresponding to the command.
<5> Make the CS pin (P32) low to end the communication.

(1) Description of package

<Public declaration symbol>

T3_6252	:	μPD6252 transfer subroutine name
RWRITE	:	RANDOM WRITE command value
RREAD	:	RANDOM READ command value
CREAD	:	CURRENT READ command value
WADAT	:	Word address storage area
TRNDAT	:	Transmit data storage area
RCVDAT	:	Receive data storage area
CMDDAT	:	Command data storage area
BUSYFG	:	Busy status test flag
CS6252	:	CS pin (P32) of μPD6252

<Register used>

A

<RAM used>

Name	Usage	Attribute	Bytes
WAADR	Stores word address (before start of transfer)	SADDR	1
TRNDAT	Stores transmit data (before start of transfer)		
RCVDAT	Stores receive data (after end of transfer)		
CMDDAT	Stores command data (before start of transfer)		

<Flag used>

Name	Usage
BUSYFG	Sets WRITE BUSY status

<Nesting>

1 level 3 bytes

<Hardware used>

- Serial interface channel 0
- P32

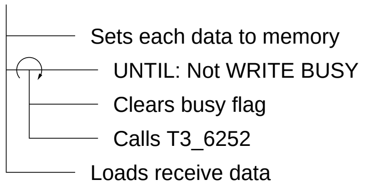
<Initial setting>

• Setting of serial interface channel 0 Selects 2-wire serial I/O mode and SB1 pin	CSIM0 = #10011011B
• Serial clock fx/2^5	TCL3 = #××××1001B
• Makes SB1 latch high	RELT = 1

<Starting>

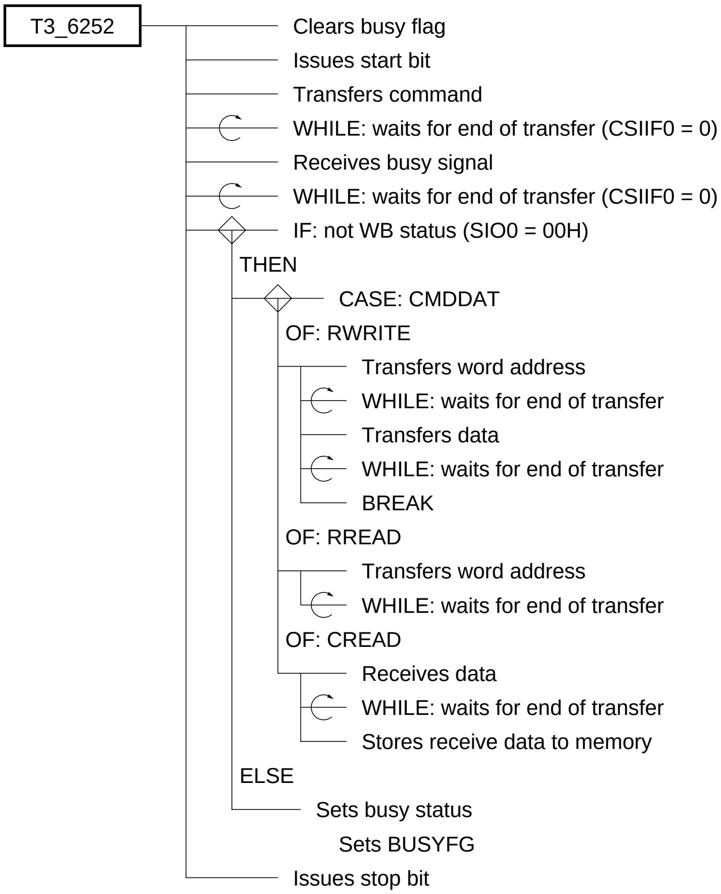
Set the necessary data corresponding to the commands and call T3_6252. After execution returns from the subroutine, the busy flag (BUSYFG) is tested. If the busy flag is set, transfer is not executed. It is therefore necessary to execute transfer again. In the receive mode, the receive data is stored RCVDAT after execution has returned from the subroutine.

(2) Example of use



```
EXTRN   RWRITE,RREAD,CREAD
EXTRN   WADAT,TRNDAT,RCVDAT,CMDDAT,T3_6252
EXTBIT  BUSYFG,CS6252

    CSIM0=#10011011B                      ; Sets 2-wire serial I/O mode and SB1 pin
    TCL3=#10011001B                       ; Sets SCK0 = 262 kHz
    CLR1    SB0
    CLR1    CS6252                        ; Makes CS of μPD6252 low
    CLR1    PM3.2

    CMDDAT=A
      :
      :
    WADAT=A
      :
      :
    TRNDAT=A
      :
      :
    repeat
            CLR1     BUSYFG
            CALL     !T3_6252
    until_bit(!BUSYFG)
      :
      :
    A=RCVDAT
```

(3) SPD chart



```
T3_6252
    Clears busy flag
    Issues start bit
    Transfers command
    WHILE: waits for end of transfer (CSIIF0 = 0)
    Receives busy signal
    WHILE: waits for end of transfer (CSIIF0 = 0)
    IF: not WB status (SIO0 = 00H)
    THEN
        CASE: CMDDAT
        OF: RWRITE
            Transfers word address
            WHILE: waits for end of transfer
            Transfers data
            WHILE: waits for end of transfer
            BREAK
        OF: RREAD
            Transfers word address
            WHILE: waits for end of transfer
        OF: CREAD
            Receives data
            WHILE: waits for end of transfer
            Stores receive data to memory
    ELSE
        Sets busy status
          Sets BUSYFG
    Issues stop bit
```

(4) Program list

```
        PUBLIC  RWRITE,RREAD,CREAD
        PUBLIC  WADAT,TRNDAT,RCVDAT,CMDDAT,T3_6252
        PUBLIC  BUSYFG,CS6252
CSI_DAT DSEG    SADDR
WADAT:  DS      1                       ; Word address storage area
TRNDAT: DS      1                       ; Transmit data storage area
RCVDAT: DS      1                       ; Receive data storage area
CMDDAT: DS      1                       ; Command data storage area

CSI_FLG BSEG
BUSYFG  DBIT                            ; Sets busy status

RWRITE  EQU     00H                     ; RANDOM WRITE mode
RREAD   EQU     0C0H                    ; RANDOM READ mode
CREAD   EQU     080H                    ; CURRENT READ mode
CS6252  EQU     0FF03H.2                ; 0FF03H=PORT3

CSI_SEG CSEG
;*************************************
;*   μPD6252 (3-wire) communication
;*************************************
T3_6252:
        CLR1    BUSYFG
        SET1    CS6252                  ; Issues start bit
        SI00=CMDDAT (A)                 ; Transfers command
        while_bit(!CSIIF0)              ; Waits for end of transfer
        endw
        CLR1    CSIIF0
        SIO0=#0FFH                      ; Starts reception of busy signal
        while_bit(!CSIIF0)              ; Waits for end of transfer
        endw
        CLR1    CSIIF0
        if(SIO0==#00H)                  ; Busy check
            switch (CMDDAT)
                case RWRITE:
                    SIO0=WADAT (A)      ; Transfers word address
                    while_bit(!CSIIF0)  ; Waits for end of transfer
                    endw
                    CLR1    CSIIF0
                    SIO0=TRNDAT (A)     ; Starts data transfer
                    while_bit(!CSIIF0)  ; Waits for end of transfer
                    endw
                    CLR1    CSIIF0
                break

                case RREAD:
                    SIO0=WADAT (A)      ; Transfers word address
                    while_bit(!CSIIF0)  ; Waits for end of transfer
                    endw
                    CLR1    CSIIF0
                case CREAD:
                    SIO0=#0FFH          ; Starts data reception
                    while_bit(!CSIIF0)  ; Waits for end of transfer
                    endw
                    CLR1    CSIIF0
                    RCVDAT=SIO0 (A)     ; Stores receive data
            ends
```

```
else
    SET1    BUSYFG          ; Sets busy status
endif
CLR1    CS6252
RET
```

8.1.2 Communication in I^{2}C bus mode

In the 2-wire mode of the μPD6252[Note], two lines, serial clock (SCL) and data (SDA) lines are used for communication. This mode conforms to the communication format of I^{2}C. Therefore, the I^{2}C mode is selected when communicating with the μPD6252 by using the μPD78002Y, 78014Y, or 78018FY subseries.

In the example shown in this section, the μPD78014Y subseries is used.

★ **Note** μPD6252 is for maintenance use.

Figure 8-23. Example of Connection between μPD6252 and I^{2}C Bus Mode

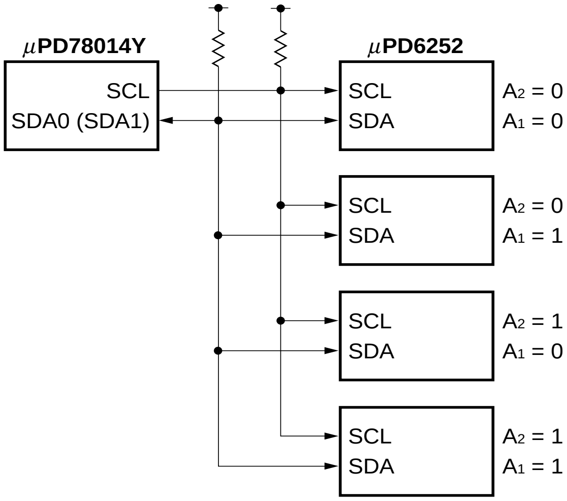
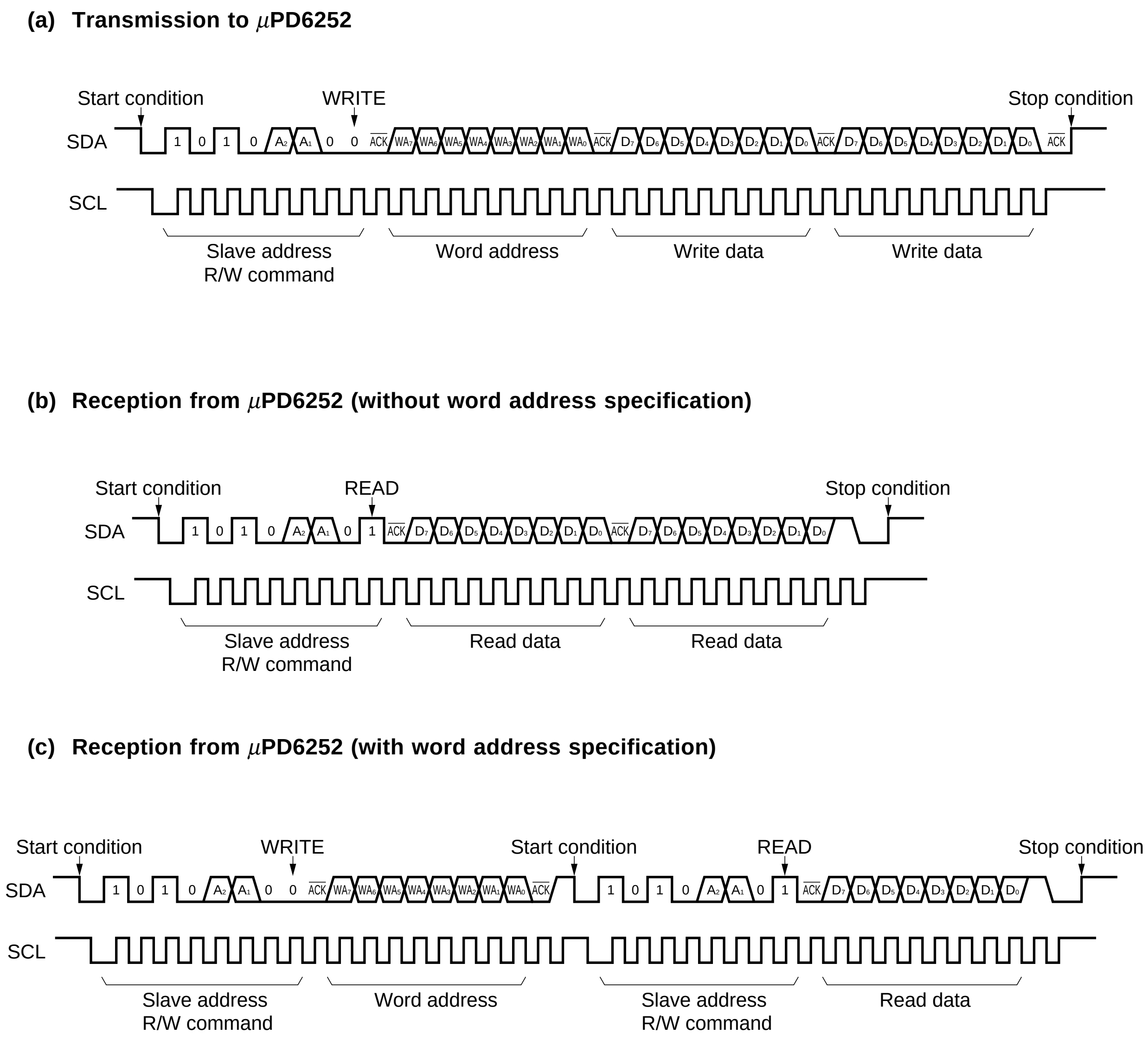


Figure 8-24 shows the communication format in which data is written to or read from the μPD6252.

Figure 8-24. μPD6252 Operation Timing



(a) Transmission to μPD6252

(b) Reception from μPD6252 (without word address specification)

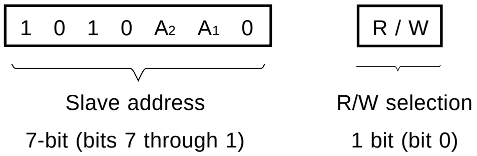
(c) Reception from μPD6252 (with word address specification)

(a) Transmission to μPD6252
Start condition
WRITE
Stop condition
SDA
1 0 1 0 A2 A1 0 0 ACK WA7 WA6 WA5 WA4 WA3 WA2 WA1 WA0 ACK D7 D6 D5 D4 D3 D2 D1 D0 ACK D7 D6 D5 D4 D3 D2 D1 D0 ACK
SCL
Slave address R/W command
Word address
Write data
Write data
(b) Reception from μPD6252 (without word address specification)
Start condition
READ
Stop condition
SDA
1 0 1 0 A2 A1 0 1 ACK D7 D6 D5 D4 D3 D2 D1 D0 ACK D7 D6 D5 D4 D3 D2 D1 D0
SCL
Slave address R/W command
Read data
Read data
(c) Reception from μPD6252 (with word address specification)
Start condition
WRITE
Start condition
READ
Stop condition
SDA
1 0 1 0 A2 A1 0 0 ACK WA7 WA6 WA5 WA4 WA3 WA2 WA1 WA0 ACK 1 0 1 0 A2 A1 0 1 ACK D7 D6 D5 D4 D3 D2 D1 D0
SCL
Slave address R/W command
Word address
Slave address R/W command
Read data

Steps <1> through <5> below are the communication procedure of the μPD6252. In this example, the number of data to be written or read is fixed to 1 byte. If the master receives data in the I^{2}C bus format, and if it has received the last data, the $\overline{\text{ACK}}$ signal is not output. Because the master does not output the $\overline{\text{ACK}}$ signal in this example, bit 5 (ACKE) of serial bus interface control register (SBIC) is always 0.

<1> Set a start condition to start communication.
Fall the data with the serial clock high.

<2> Transmit the slave address value (bits 1 through 7) of the μPD6252 and write (bit 0 = 0)/read (bit 0 = 1) select bit.



Remark A_2 and A_1 are set by external pins.

<3> Transfer the data.

- In transmission mode
  (i) Transmit the word address of the μPD6252.
  (ii) Transmit the write data.

- In reception mode
  Receive the read data.

<4> Set an end condition to end the communication.
Rise the data with the serial clock high.

<5> Because a word address is specified only in the write mode, to read data by specifying an address, the address must be specified by once setting the write mode.

If the μPD6252 does not return the $\overline{\text{ACK}}$ signal during data transfer, communication is stopped.

The start and end conditions are set by bit 3 (CLC) of interrupt timing specification register (SINT) when the serial clock is manipulated, and by RELT and CMDT (bits 0 and 1 of SBIC) when data is manipulated.

(1) Description of package

<Public declaration symbol>

T2_6252	:	μPD6252 transfer subroutine name
WAADR	:	Word address storage area
TRNDAT	:	Transmit data storage area
RCVDAT	:	Receive data storage area
SLVADR	:	Slave address storage area
BUSYFG	:	Busy status test flag
WRCHG	:	Write → read mode change flag
ERRFG	:	Error status test flag

<Register used>

A

<RAM used>

Name	Usage	Attribute	Bytes
WAADR	Stores word address (before start of transfer)	SADDR	1
TRNDAT	Stores transmit data (before start of transfer)		
RCVDAT	Stores receive data (after end of transfer)		
SLVADR	Stores slave address		

<Flag used>

Name	Usage
BUSYFG	Sets WRITE BUSY status
WRCHG	Changes write mode to read mode
ERRFG	Sets error status

<Nesting>

1 level 2 bytes

<Hardware used>

Serial interface 0

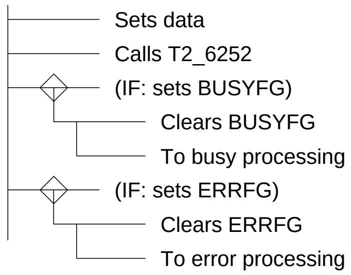
<Initial setting>

- Setting of serial interface channel 0
  Selects 2-wire serial I/O mode and SB0 pin CSIM0 = #10011011B
- Selects serial clock $f_X/2^4$ and 16 TCL3 = #××××1000B
- Generates interrupts at rising edge of 9th serial clock and sets clock line to high level SINT = #00001011B

<Starting>

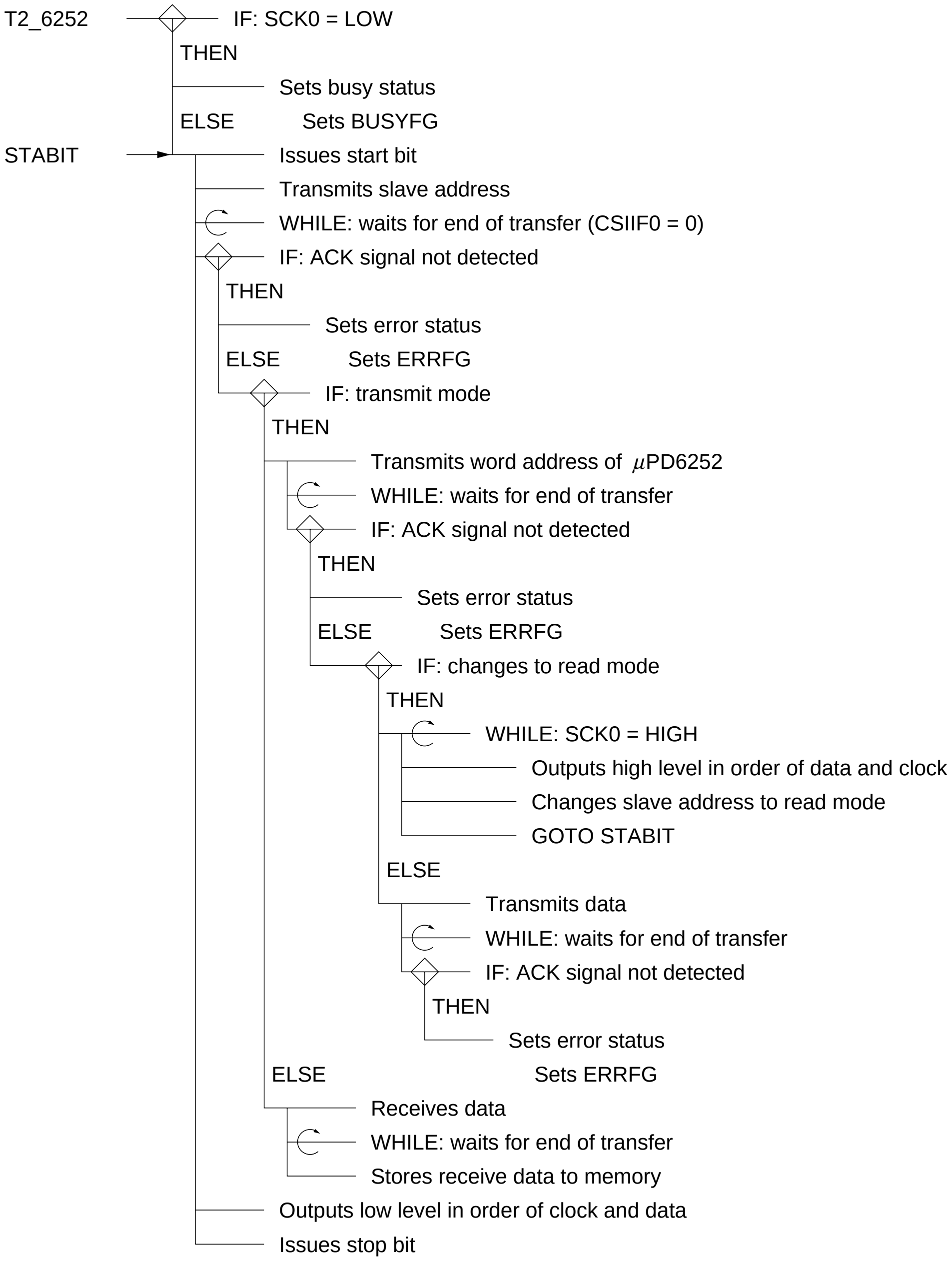
- Set the necessary data corresponding to the commands and call T2_6252. In the reception mode, the receive data is stored to RCVDAT after execution has returned from the subroutine.
- If the serial clock is low (busy status) when communication is started or if ACK cannot be received during data transfer, the BUSYFG and ERRFG are set. Test and clear these flags with the main processing.

(2) Example of use



```
EXTRN   WAADR,TRNDAT,RCVDAT,SLVADR,T2_6252
EXTBIT  BUSYFG,WRCHG,ERRFG

SET1    SB0
CSIM0=#10011011B                            ; Serial interface 2-wire, SB0
SINT=#00001011B                             ; Sets I$^2$C mode
TCL3=#10001000B                             ; SCK = 32.7 kHz
SET1    RELT
SET1    SCK0
CLR1    SB0
  ⋮
WAADR=A
  ⋮
TRNDAT=A

SLVADR=A
CALL    !T2_6252
if_bit(BUSYFG)
        CLR1    BUSYFG
    ⋮
endif
    ⋮
if|bit(ERRFG)
        CLR1    ERRFG
    ⋮
ENDIF
```

(3) SPD chart



```
T2_6252    IF: SCK0 = LOW
             THEN
               Sets busy status
             ELSE     Sets BUSYFG
STABIT       Issues start bit
             Transmits slave address
             WHILE: waits for end of transfer (CSIIF0 = 0)
             IF: ACK signal not detected
               THEN
                 Sets error status
               ELSE     Sets ERRFG
                 IF: transmit mode
                   THEN
                     Transmits word address of μPD6252
                     WHILE: waits for end of transfer
                     IF: ACK signal not detected
                       THEN
                         Sets error status
                       ELSE     Sets ERRFG
                         IF: changes to read mode
                           THEN
                             WHILE: SCK0 = HIGH
                               Outputs high level in order of data and clock
                               Changes slave address to read mode
                               GOTO STABIT
                           ELSE
                             Transmits data
                             WHILE: waits for end of transfer
                             IF: ACK signal not detected
                               THEN
                                 Sets error status
                                 Sets ERRFG
                   ELSE
                     Receives data
                     WHILE: waits for end of transfer
                     Stores receive data to memory
             Outputs low level in order of clock and data
             Issues stop bit
```

(4) Program list

```
        PUBLIC  WAADR,TRNDAT,RCVDAT,SLVADR,T2_6252
        PUBLIC  BUSYFG,WRCHG,ERRFG

CSI_DAT DSEG    SADDR
WAADR:  DS      1                      ; Word address storage area
TRNDAT: DS      1                      ; Transmit data storage area
RCVDAT: DS      1                      ; Receive data storage area
SLVADR: DS      1                      ; Salve address storage area

CSI_FLG BSEG
BUSYFG  DBIT                           ; Sets busy status
WRCHG   DBIT                           ; Changes mode
ERRFG   DBIT                           ; Sets error status

SCK0    EQU     P2.7

CSI_SEG CSEG
;*************************************
;*   μPD6252 (2-wire) communication
;*************************************
T2_6252:
        if_bit(!CLD)
            SET1    BUSYFG             ; Busy status
        else
STABIT:
            SET1    CMDT               ; Issues start bit
            NOP                        ; Waits for start bit valid width
            NOP
            NOP
            NOP
            NOP
            CLR1    CLC                ; Changes clock to low level
            SIO0=SLVADR (A)            ; Starts transmitting slave address
            while_bit(!CSIIF0)         ; Waits for end of transfer
            endw
            CLR1    CSIIF0
            if_bit(!ACKD)              ; ACK signal not detected
                SET1    ERRFG
            elseif_bit(!SLVADR.0)      ; Transmission mode
                SI00=WAADR (A)         ; Starts transmitting word address
                while_bit(!CSIIF0)     ; Waits for end of transfer
                endw
                CLR1    CSIIF0
                if_bit(!ACKD)          ; ACK signal not detected
                    SET1    ERRFG
                elseif_bit(WRCHG)
                    while_bit(CLD)
                    endw
                    SET1    RELT
                    SET1    CLC
                    while_bit(!CLD)    ; Checks high level of clock
                    endw
                    NOP                ; Waits for high level valid width of clock
                    NOP
                    NOP
                    NOP
                    NOP
                    NOP
                    NOP
                    NOP
                    SET1    SLVADR.0   ; Changes to read mode address
                    goto    STABIT

                else
```

```
EXTRN   WAADR,TRNDAT,RCVDAT,SLVADR,T2_6252
EXTBIT  BUSYFG,WRCHG,ERRFG

SET1    SB0
CSIM0=#10011011B
SINT=#00001011B                             ; Serial interface 2-wire, SB0
TCL3=#10001000B                             ; Sets I$^2$C mode
SET1    RELT                                ; SCK = 32.7 kHz
SET1    SCK0
CLR1    SB0

WAADR=A

TRNDAT=A

SLVADR=A
CALL    !T2_6252
if_bit(BUSYFG)
        CLR1    BUSYFG
  ⋮
endif
  ⋮
if|bit(ERRFG)
        CLR1    ERRFG
  ⋮
ENDIF
```

★ **(5) Limitations when I^{2}C bus mode is used (μPD78002Y and 78014Y series)**

The following limits apply to the μPD78002Y and 78014Y subseries.

(a) When using microcontroller as master device in I^{2}C bus mode

Applicable model: μPD78P014Y

Description: If the rise time of SCL is longer than 1/32 of the transfer clock cycle when the master device outputs SCL, the SCL output may be stopped or whisker may occur.

The "rise time" means the time required for the SCL signal line to reach 0.8 V_{DD} or more after the master device has started communication. Therefore, the "rise time" includes the time during which the master device tries to communicate but the slave device makes SCL low because of wait control.

(b) When using microcontroller as slave device in I^{2}C bus mode

Applicable models: μPD78001BY, 78002BY, 78011BY, 78012BY, 78013Y, 78014Y, 78P014Y

Description: When all the following conditions are satisfied, none of the slaves in the system can transmit data.

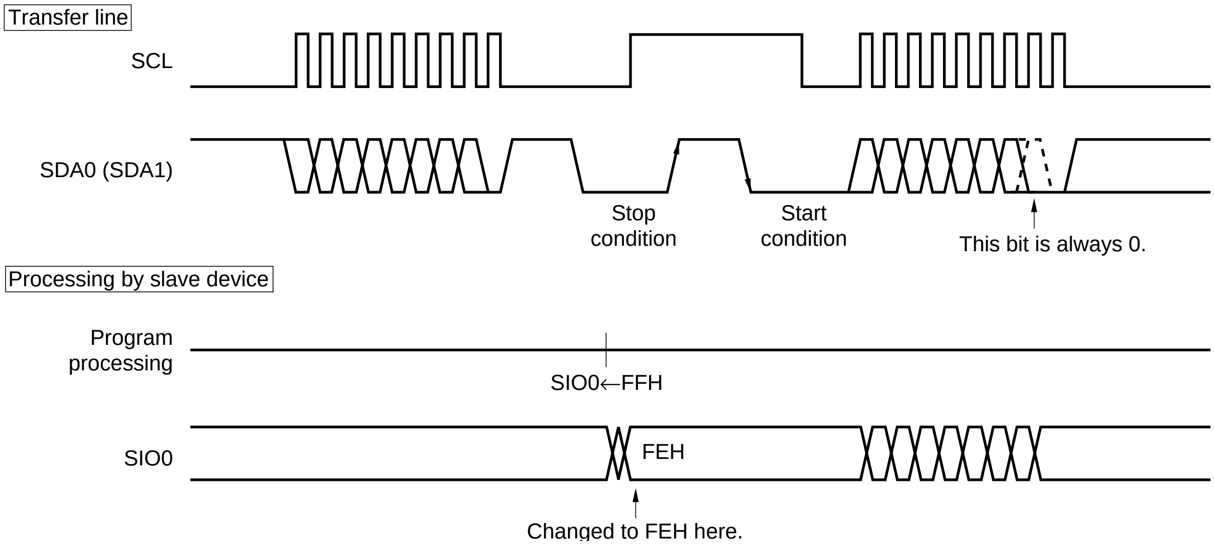
- If the μPD78002Y or 78014Y subseries is used as a slave device in the I^{2}C bus mode
- If the master outputs a stop condition on completion of transmission to the μPD78002Y or 78014Y subseries (= slave reception).
- If communication following the master transmission (= slave reception) to the μPD78002Y or 78014Y subseries is a master reception (= slave transmission) request to any unit.

The μPD78002Y and 78014Y subseries writes a communication start command to the serial I/O shift register 0 (SIO0). When these microcontrollers receive data, they write FFH to SIO0 to turn off the N-ch open-drain output for transmission.

If the master device makes SCL high to output a start condition or stop condition after the μPD78002Y or 78014Y subseries has written FFH to SIO0, the μPD78002Y or 78014Y at the slave side shifts the contents of SIO0. As a result, FFH, which has been written as a start command, is shifted, and the LSB of SIO0 becomes equal to the level of SDA0 (SDA1).

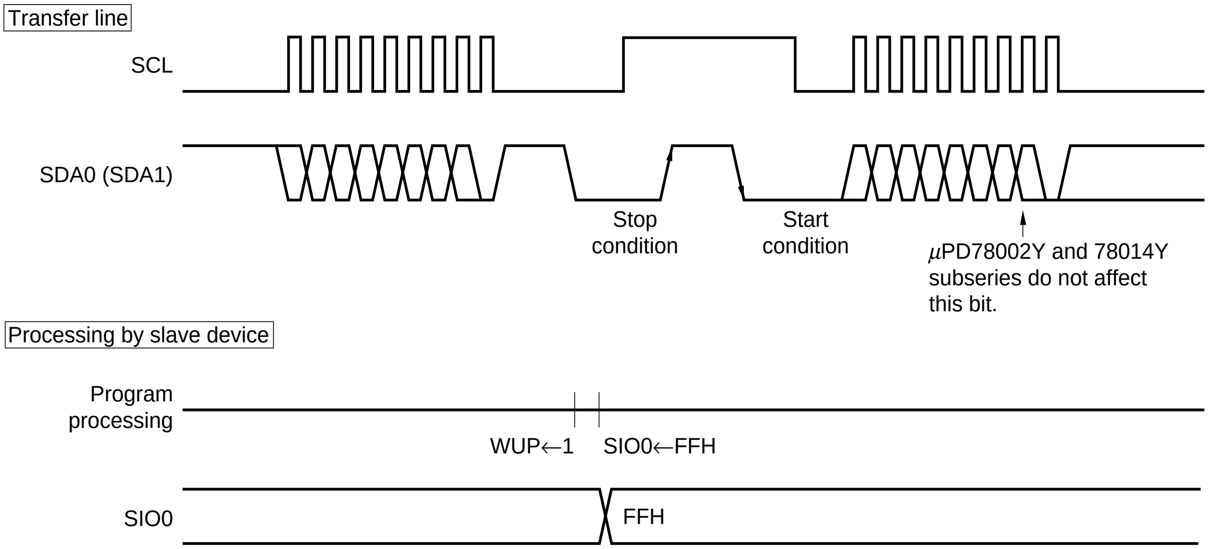
When SDA0 (SDA1) is made low and SCL is made high, as shown in the figure below, so that the master device outputs a stop condition, the value of SIO0 is changed to FEH (LSB is 0). Therefore, the LSB of the value received next is always 0.

Because the value received after the stop condition information is considered as a slave address field, the LSB (transfer direction specification bit) of the slave address field is always 0 (slave reception) regardless of the output data of the master.



Preventive measure : In a system where the timing to output a stop condition to the μPD78002Y or 78014Y subseries (where the number of data communicated between the μPD78002Y or 78014Y subseries and master is determined), the problem discussed in (b) above can be avoided in software.

Set bit 5 (WUP) of the serial operation mode register 0 (CSIM0) of the slave device to 1 and the serial I/O shift register 0 (SIO0) to FFH before the stop condition is output. In this way, the wake-up function is effected on the slave address field output next by the master device, the N-ch open-drain output is automatically turned off, and the receive data of the slave device is not longer affected.



★ **(6) Limitation when using I²C bus mode (μPD78018FY subseries)**

The following limitation applies when the μPD78018FY subseries is used.

- **When the device is used as a slave device in the I²C bus mode**

Applicable models: μPD78011FY, 78012FY, 78013FY, 78014FY, 78015FY, 78016FY, 78018FY, 78P018FY

Description: If the wake-up function is executed (by setting the WUP flag (bit 5 of serial operation mode register 0 (CSIM0) to 1) in the serial transfer status**Note**, the data between other slave device and the master devices is checked as an address. If that data coincides with the slave address of the μPD78018FY subseries, therefore, the μPD78018FY subseries takes part in communication, destroying the communication data.

Note The serial transfer status is the status from when the serial I/O shift register 0 (SIO0) has been written until the interrupt request flag (CSIIF0) is set to 1 by completion of serial transfer.

Preventive measures: The above problem can be prevented by modifying the program.

Before executing the wake-up function, execute the following program that clears the serial transfer status. When executing the wake-up function, do not execute an instruction that writes data to SIO0. Even if such an instruction is executed, data can be received when the wake-up function is executed.

This program is to clear the serial transfer status. To clear the serial transfer status, serial interface channel 0 must be stopped (by clearing the CSIE0 flag (bit 7 of the serial operation mode register (CSIM0) to 0). If the serial interface channel 0 is stopped in the I²C bus mode, however, the SCL pin outputs a high level and the SDA0 (SDA1) pin outputs a low level, affecting communication on the I²C bus. Therefore, this program allows the SCL and SDA0 (SDA1) pin to go into a high-impedance state to prevent the I²C bus from being affected.

Note that, in this example, the serial data input/output pin is SDA0 (/P25). If SDA1 (/P26) is used as the serial data input/output pin, take P2.5 and PM2.5 in the program as P2.6 and PM2.6.

- **Example of program that clears serial transfer status**

```
SET1    P2.5     ; <1>
SET1    PM2.5    ; <2>
SET1    PM2.7    ; <3>
CLR1    CSIE0    ; <4>
SET1    CSIE0    ; <5>
SET1    RELT     ; <6>
CLR1    PM2.7    ; <7>
CLR1    P2.5     ; <8>
CLR1    PM2.5    ; <9>
```

<1> When the I^{2}C bus mode is restored by instruction <5>, the SDA0 pin does not output a low level. The output of the SDA0 pin goes into a high-impedance state.

<2> The P25(/SDA0) pin is set in the input mode to prevent the SDA0 line from being affected when the port mode is set by instruction <4>. The P25 pin is set in the input mode when instruction <2> is executed.

<3> The P27 (/SCL) pin is set in the input mode to prevent the SCL line from being affected when the port mode is set by instruction <4>. The P27 pin is set in the input mode when instruction <3> is executed.

<4> The I^{2}C bus mode is changed to the port mode.

<5> The port mode is changed to the I^{2}C bus mode.

<6> Instruction <8> prevents the SDA0 pin from outputting a low level.

<7> The P27 pin is set in the output mode because it must be in the output mode in the I^{2}C bus mode.

<8> The output latch of the P25 pin is cleared to 0 because it must be cleared to 0 in the I^{2}C bus mode.

<9> The P25 pin is set in the output mode because it must be in the output mode in the I^{2}C bus mode.

Remark RELT: Bit 0 of serial bus interface control register (SBIC)

8.2 Interface with OSD LSI (μPD6451A)

The OSD (On Screen Display) LSI μPD6451A displays the program information of a VCR and TV channels on a display when used in combination with a microcontroller. The μPD6451A is interface with four lines: DATA, CLK, STB, and BUSY. In the example shown in this section, the μPD78014 subseries is used to interface the μPD6451A.

Figure 8-25. Example of Connecting μPD6451A

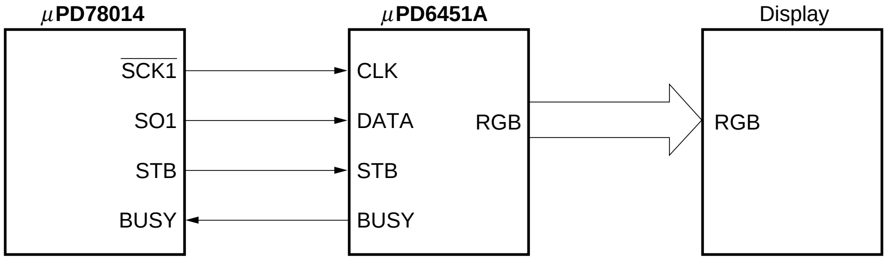
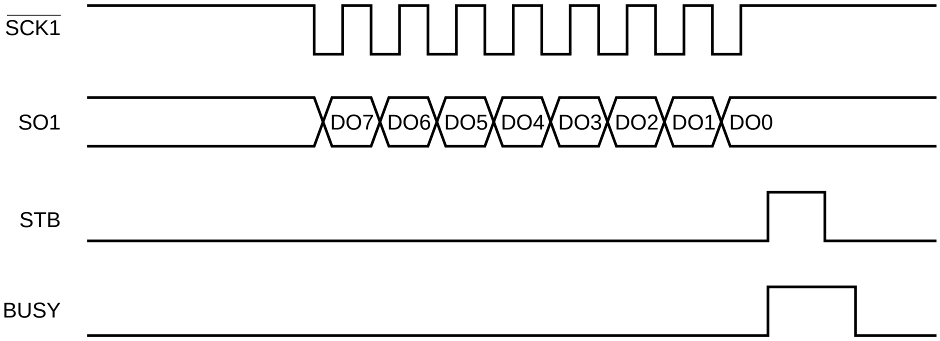


Figure 8-26. Communication Format of μPD6451A



The strobe signal (STB) is output and busy signal (BUSY) is tested automatically by the serial interface channel 1 of the 78K/0 series to establish handshaking with and to interface the μPD6451A. To match the communication format of the μPD6451A, the μPD78014 subseries is set in a mode in which output of the strobe signal and input of the busy signal (high active) are enabled. By setting the transmit data (32 bytes MAX) in a buffer area (FAC0H through FADFH) and the number of transmit data to the automatic data transmit/receive address pointer (ADTP), you can automatically transmit plural data successively.

(1) Description of package

<Public declaration symbol>

TR6451 : μPD6451A transfer subroutine name
DTVAL : Number of transmit data setting area

<Register used>

A

<RAM used>

Name	Usage	Attribute	Bytes
DTVAL	Stores number of transmit data	SADDR	1

<Nesting>

1 level 2 bytes

<Hardware used>

- Serial interface channel 1

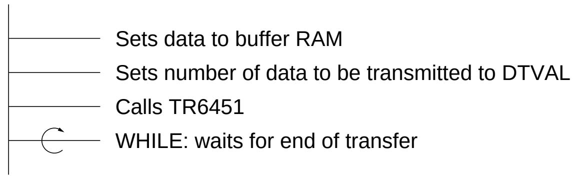
<Initial setting>

- Setting of serial interface channel 1
  Enables automatic transmission/reception with MSB first CSIM1 = #10100011B
  Enables busy input (high active) and strobe output in single mode ADTC = #00000110B
- Interval time of data transfer ADTI = #00000000B
- Serial clock $f_X/2^5$ TCL3 = #1001××××B
- Makes P22 output latch high
- Sets P21, P22, and P23 in output mode and P24 in input mode PM2 = #×××1000×B

<Starting>

Set the data to be transmitted to the buffer RAM (starting from the highest address), and the number of data to be transmitted to DTVAL, and call TR6451. You can check the end of data transfer by testing the bit 3 (TRF) of the automatic data transfer/reception control register (ADTC).

(2) Example of use



Sets data to buffer RAM

Sets number of data to be transmitted to DTVAL

Calls TR6451

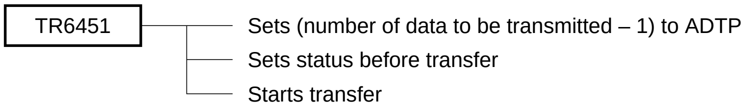
WHILE: waits for end of transfer

```
        EXTRN   TR6451,DTVAL

SCK1    EQU     P2.2
             :
             :
        P2=#00000100B
        PM2=#11110001B
        CSIM1=#10100011B                 ; Sets automatic transfer/reception function
        TCL3=#10011001B                  ; SCK1 = 262 kHz
        ADTC=#00000110B                  ; Enable strobe and busy signals
        ADTI=#00000000B                  ; Interval time of data transfer
             :
             :
        DE=#TABLE1                       ; Sets table reference address of transmit data
        HL=#0FAC0H                       ; Sets first address of buffer RAM
        B=32                             ; Sets number of data to be transmitted

        while(B>#0)                      ; Transfers transmit data to buffer RAM
                B--
                [HL+B]=[DE] (A)
                DE++
        endw

        DATVAL=#32                       ; Sets number of data to be transmitted
        CALL    !TR6451
        while_bit(TRF)                   ; Waits for end of transfer
        endw
```

```
TABLE1:
        DB      11111111B                 ; Power-ON reset, command 1
        DB      01000000B                 ; Vertical address 0
        DB      11000000B                 ; Horizontal address 0
        DB      10000000B                 ; Character size
        DB      11111100B                 ; Command 0
        DB      11101001B                 ; Turns LC transmission ON, blinking OFF, display ON

        DB      10001100B                 ; Turns blinking ON. Character: red

        DB      11011011B                 ; Color specification, background filled in cyan
        DB      10010101B                 ; Number of display lines: 5
        DB      10100000B                 ; Number of display digits: 0


        DB      07H                       ; 7
        DB      08H                       ; 8
        DB      1BH                       ; K
        DB      6DH                       ; /
        DB      00H                       ; 0
        DB      10H
        DB      11H                       ; A
        DB      20H                       ; P
        DB      20H                       ; P
        DB      1CH                       ; L
        DB      19H                       ; I
        DB      13H                       ; C
        DB      11H                       ; A
        DB      24H                       ; T
        DB      19H                       ; I
        DB      00H                       ; O
        DB      1EH                       ; N
        DB      10H
        DB      1EH                       ; N
        DB      00H                       ; O
        DB      24H                       ; T
        DB      15H                       ; E
```

Remark For the command and data of the output table data, refer to *μ***PD6451A Data Sheet (Document No. IC-2337)**.

(3) SPD chart



TR6451

Sets (number of data to be transmitted – 1) to ADTP

Sets status before transfer

Starts transfer

(4) Program list

```
        PUBLIC  TR6451,DTVAL

CSI_DAT DSEG    SADDR
DTVAL:  DS      1                       ; Number of data setting area

CSI_SEG CSEG
;**************************************
;*       μPD6451A communication
;**************************************
TR6451:
        A=DTVAL                         ; Sets number of data
        A--
        ADTP=A
        SIO1=#0FFH                      ; Starts transfer
        RET
```

8.3 Interface in SBI Mode

The 78K/0 series has an SBI mode conforming to NEC serial bus format. In this mode, one master CPU can communicate with two or more slave CPUs by using two lines: clock and data. In the example shown in this section, the μPD78014 subseries is used.

Figure 8-27 shows an example of connection to use the SBI mode, and Figure 8-28 shows the communication format.

Figure 8-27. Example of Connection in SBI Mode

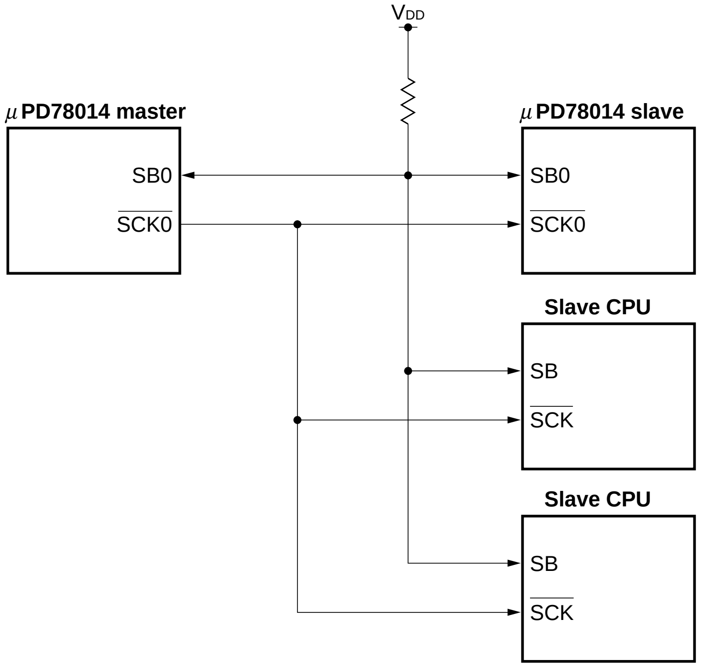
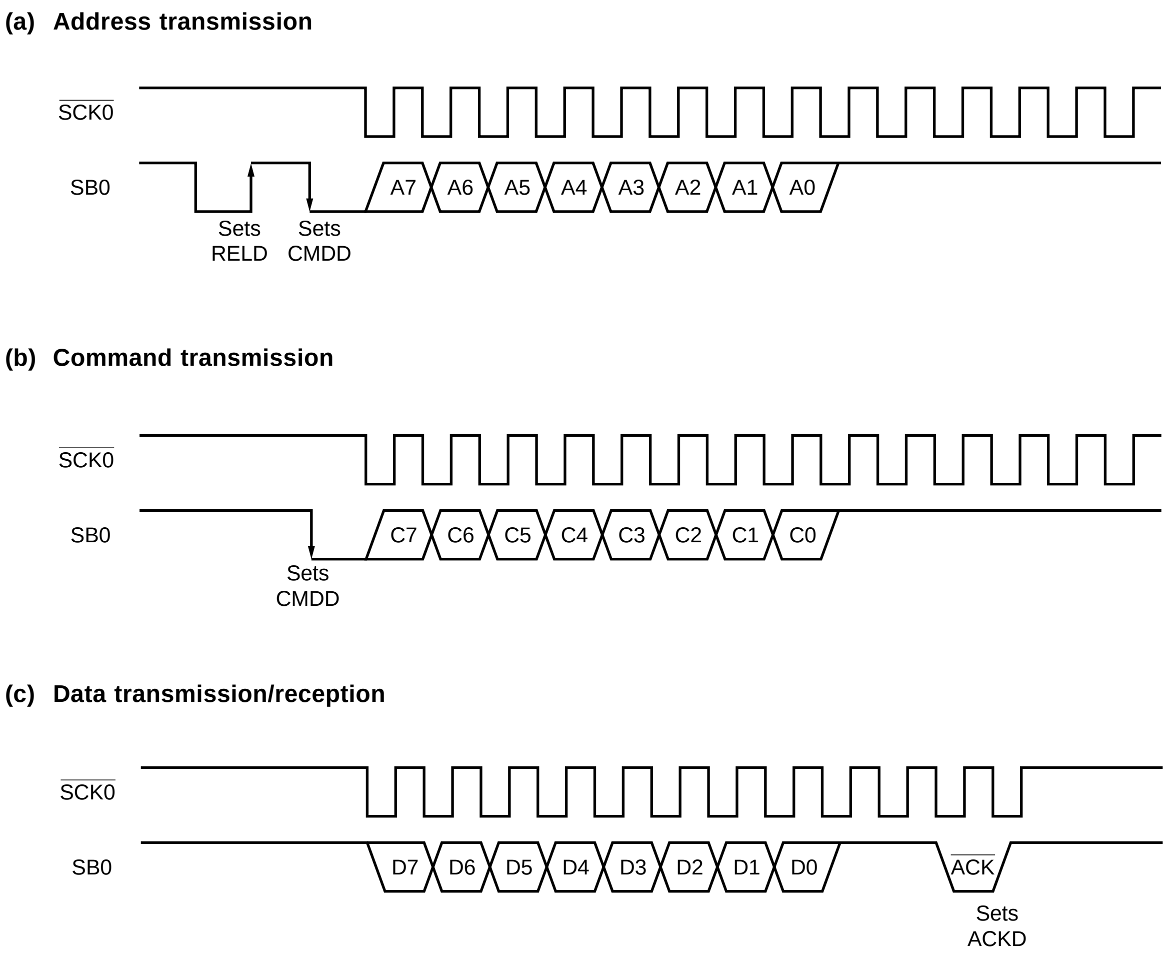


Figure 8-28. Communication Format in SBI Mode



(a) Address transmission

(b) Command transmission

(c) Data transmission/reception

Table 8-5. Signals in SBI Mode

Signal Name	Output by:	Meaning
Address	Master	Selects slave device
Command	Master	Command to slave device
Data	Master/slave	Data to be processed by slave or master
Clock	Master	Serial data transmission/reception synchronization signal
$\overline{\text{ACK}}$	Receiver side[Note]	Reception acknowledge signal
$\overline{\text{BUSY}}$	Slave	Busy status

Note This signal is output by the receiver side during normal operation. However, it is output by the master CPU in case of an error such as time out.

8.3.1 Application as master CPU

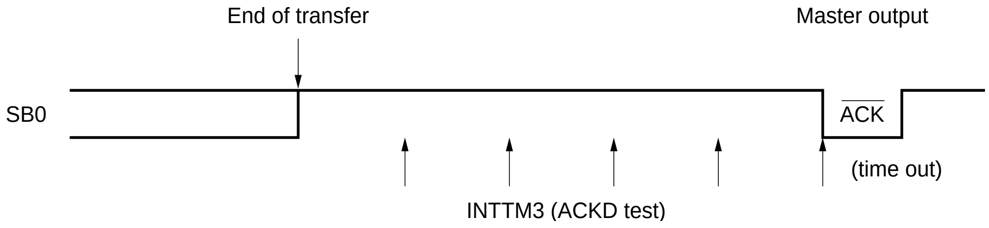
When the μPD78014 subseries is used as a master CPU, it performs processing (a) through (d) below with respect to slave CPUs.

(a) Address transmission
(b) Command transmission
(c) Data transmission
(d) Data reception

While the above processing is performed, errors <1> and <2> below are checked.

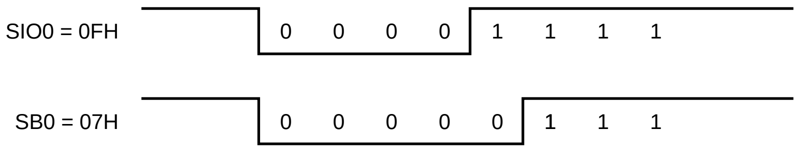
<1> Time out processing
If the master CPU transmits data and a slave does not return the $\overline{\text{ACK}}$ signal within a specific time (in this example, before the watch interrupt request occurs five times), the master judges that an error has occurred. The master CPU then outputs an $\overline{\text{ACK}}$ signal and terminates the processing.

Figure 8-29. $\overline{\text{ACK}}$ Signal in Case of Time out



<2> Testing bus line
The master CPU tests whether data has been correctly output to the bus line by setting the transmit data to the serial I/O shift register 0 (SIO0) and the slave address register (SVA). Because the data on the bus line is received by SIO0, it confirms that the data has been output normally by testing bit 6 (COI) of the serial operating mode register 0 (CSIM0) (that is set when SIO0 coincides with SVA) at the end of transfer.

Figure 8-30. Testing Bus Line



In Figure 8-30, the values of SIO0 and SVA do not coincide (SIO0 = 07H and SVA = 0FH). Consequently, COI = 0, and an error has occurred on the bus line.

(1) Description of package

<Public declaration symbol>

M_TRANS : Master SBI transfer subroutine name
TR_MODE : Storage area of transfer mode select value
TRNDAT : Transmit data storage area
RCVDAT : Receive data storage area
TRADR : Address transmit mode select value
TRCMD : Command transmit mode select value
TRDAT : Data transmit mode select value
RCDAT : Data reception mode select value
ERRORF : Error status test flag

<Register used>

Subroutine A

<RAM used>

Name	Usage	Attribute	Bytes
TR_MODE	Stores transfer mode select value	SADDR	1
ACKCT	Time out counter		
TRNDAT	Stores transmit data		
RCVDAT	Stores receive data		

<Flag used>

Name	Usage
RCVFLG	Sets reception mode
BUSYFG	Sets busy status
ERRORF	Sets error status
ACKWFG	Sets $\overline{\text{ACK}}$ signal wait status

<Nesting>

2 levels 5 bytes

<Hardware used>

- Serial interface channel 0
- Watch timer

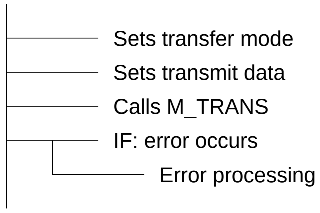
<Initial setting>

- Sets serial interface channel 0
  Selects SBI mode and SB1 pin CSIM0=#10010011B
- Serial clock: $f_{XX}/2^5$ TCL3=#××××1001B
- Makes SO0 latch high RELT=1
- Makes P27 output latch high P27=1
- Watch timer interval: 1.95 ms TMC2=#00100110B
- Enables watch timer interrupt

<Starting>

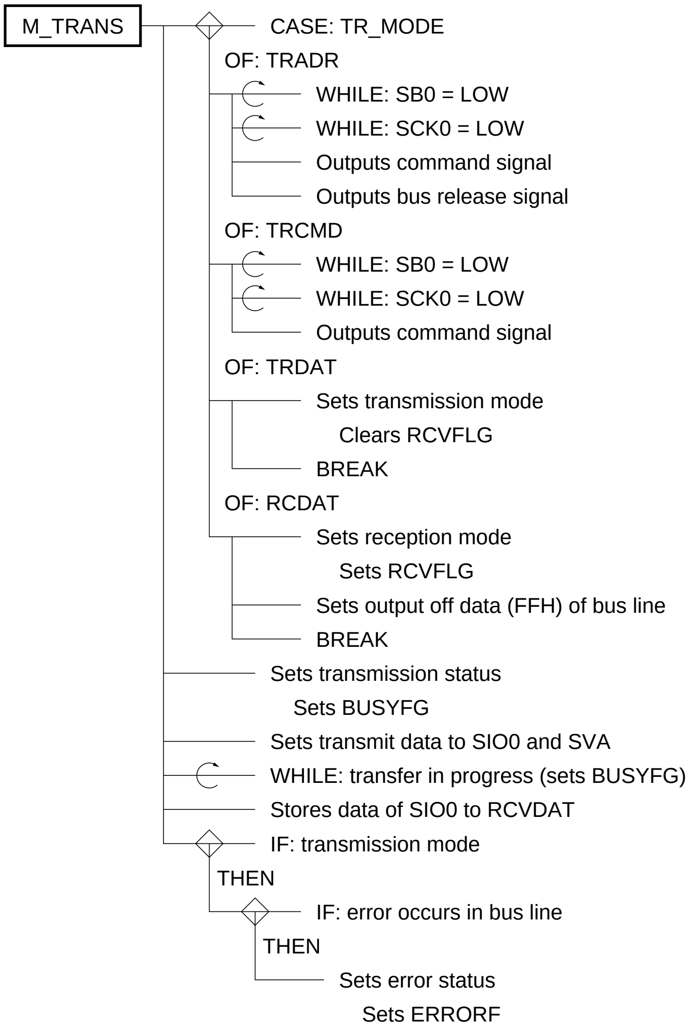
Set the transfer mode and necessary data, and call M_TRANS. When execution has returned from the subroutine, occurrence of a transfer error can be checked by testing the error flag (ERRORF). In the reception mode, the receive data is stored to RCVDAT after execution has returned from the subroutine.

(2) Example of use

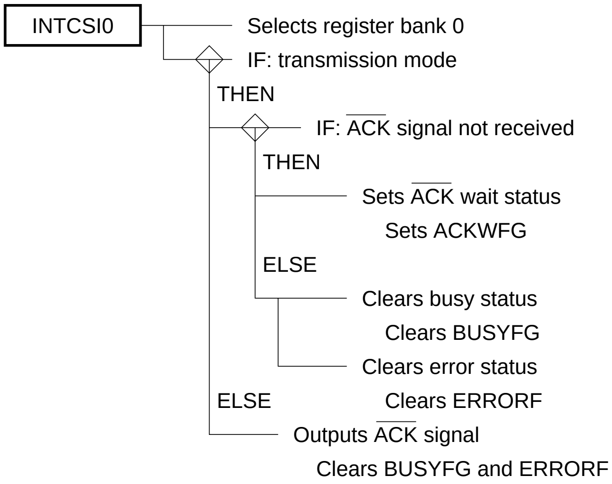


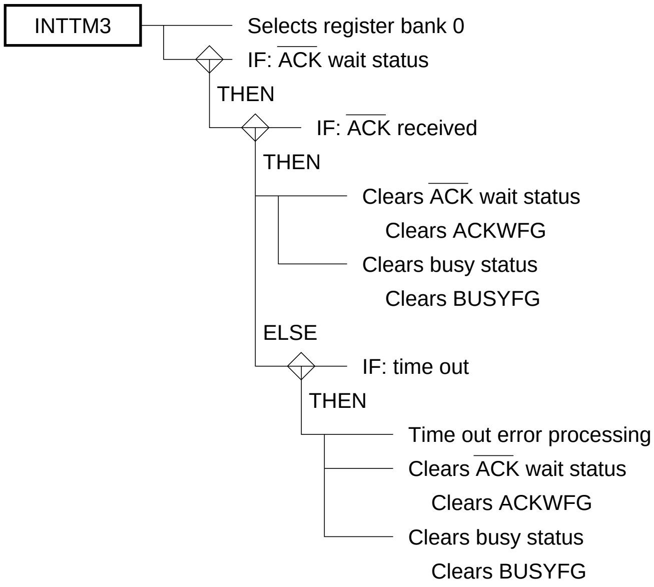
```
EXTRN    M_TRANS,TR_MODE,TRADR,TRCMD,TRDAT,RCDAT
EXTRN    TRNDAT,RCVDAT
EXTBIT   ERRORF

SCK0     EQU     P2.7
SB1      EQU     P2.5
      :
      :
SET1     SB1
CSIM0=#10010111B                ; Operates in SBI mode
TCL3=#10011001B                 ; SCK0 = 262 kHz
TMC2=#00100110B                 ; Sets interval of watch timer to 1.95 ms
CLR1     BSYE                   ; Disables output of busy signal
SET1     RELT                   ; Sets output latch
SET1     SCK0
CLR1     SB1
CLR1     CSIMK0                 ; Enables serial interface channel 0 interrupt
CLR1     TMMK3                  ; Enables watch timer interrupt
EI                              ; Enables master interrupt

      :
      :
TR_MODE=#TRADR
TRNDAT=#5AH
CALL     !M_TRANS
if_bit(ERRORF)
    Error processing
endif
```

(3) SPD chart



```
M_TRANS
    CASE: TR_MODE
    OF: TRADR
        WHILE: SB0 = LOW
        WHILE: SCK0 = LOW
        Outputs command signal
        Outputs bus release signal
    OF: TRCMD
        WHILE: SB0 = LOW
        WHILE: SCK0 = LOW
        Outputs command signal
    OF: TRDAT
        Sets transmission mode
           Clears RCVFLG
        BREAK
    OF: RCDAT
        Sets reception mode
           Sets RCVFLG
        Sets output off data (FFH) of bus line
        BREAK
    Sets transmission status
       Sets BUSYFG
    Sets transmit data to SIO0 and SVA
    WHILE: transfer in progress (sets BUSYFG)
    Stores data of SIO0 to RCVDAT
    IF: transmission mode
    THEN
        IF: error occurs in bus line
        THEN
            Sets error status
               Sets ERRORF
```


INTCSI0
Selects register bank 0
IF: transmission mode
THEN
IF: $\overline{\text{ACK}}$ signal not received
THEN
Sets $\overline{\text{ACK}}$ wait status
Sets ACKWFG
ELSE
Clears busy status
Clears BUSYFG
Clears error status
Clears ERRORF
ELSE
Outputs $\overline{\text{ACK}}$ signal
Clears BUSYFG and ERRORF


INTTM3
Selects register bank 0
IF: $\overline{\text{ACK}}$ wait status
THEN
IF: $\overline{\text{ACK}}$ received
THEN
Clears $\overline{\text{ACK}}$ wait status
Clears ACKWFG
Clears busy status
Clears BUSYFG
ELSE
IF: time out
THEN
Time out error processing
Clears $\overline{\text{ACK}}$ wait status
Clears ACKWFG
Clears busy status
Clears BUSYFG

(4) Program list

```
        PUBLIC  M_TRANS,TR_MODE,TRADR,TRCMD,TRDAT,RCDAT
        PUBLIC  TRNDAT,RCVDAT,ERRORF

VECSI0  CSEG    AT 0EH
        DW      INTCSI0       ; Sets vector address of serial interface channel 0
VETM3   CSEG    AT 12H
        DW      INTTM3        ; Sets vector address of watch timer

SBI_DAT DSEG    SADDR
TRNDAT: DS      1             ; Transmit data
RCVDAT: DS      1             ; Receive data
TR_MODE:DS      1             ; Sets transfer mode
ACKCT:  DS      1             ; ACK time out count

SBI_FLG BSEG
RCVFLG  DBIT                  ; Sets reception mode
BUSYFG  DBIT                  ; Transfer status
ERRORF  DBIT                  ; Error display
ACKWFG  DBIT                  ; ACK wait status

SB0     EQU     P2.5
SCK0    EQU     P2.7

TRADR   EQU     1             ; Selects address transmission mode
TRCMD   EQU     2             ; Selects command transmission mode
TRDAT   EQU     3             ; Selects data transmission mode
RCDAT   EQU     4             ; Selects data reception mode
```

```
;*******************************************
;*       SBI data transfer processing
;*******************************************
SBI_SEG CSEG
M_TRANS:
    switch(TR_MODE)
    case TRADR:
        SET1    PM2.5
        while_bit(!SB0)                      ; SB0 = high?
        CLR1    PM2.5
        endw
        while_bit(!SCK0)                     ; SCK = high?
        endw
        SET1    CMDT                         ; Outputs command signal
        NOP                                  ; Wait
        SET1    RELT                         ; Outputs bus release signal
        A=#TRCMD
    case TRCMD:
        SET1    PM2.5
        while_bit(!SB0)                      ; SB0 = high?
        CLR1    PM2.5
        endw
        while_bit(!SCK0)                     ; SCK = high?
        endw
        SET1    CMDT                         ; Outputs command signal
        A=#TRDAT
    case TRDAT:
        CLR1    RCVFLG                       ; Sets transmission mode
        A=TRNDAT                             ; Sets transmit data
        break
    case RCDAT:
        SET1    RCVFLG                       ; Sets reception mode
        MOV     A,#0FFH                      ; Turns off receive buffer
        break
    ends

        SET1    BUSYFG                       ; Sets transfer status
        SVA=A                                ; Tests bus line
        SIO0=A                               ; Starts transfer

        while_bit(BUSYFG)                    ; Transfer in progress
        endw
        RCVDAT=SIO0 (A)                      ; Stores receive data
        if_bit(!RCVFLG)                      ; Transmission mode
            if_bit(!COI)                     ; Bus line output abnormal
                SET1    ERRORF               ; Sets error status
            endif
        endif
        RET
```

```
;***************************************
;*     INTCSI0 interrupt processing
;***************************************
CSI_SEG CSEG
INTCSI0:
        SEL RB0
        if_bit(!RCVFLG)                       ; Transmission mode
            if_bit(!ACKD)                     ; Acknowledge signal not received
                ACKCT=#5                      ; Sets acknowledge signal wait status
                SET1    ACKWFG
            else
                CLR1    BUSYFG                ; Clears busy status
                CLR1    ERRORF                ; Clears error status
            endif
        else
            SET1    ACKT                      ; Outputs acknowledge signal
            CLR1    BUSYFG                    ; Clears busy status
            CLR1    ERRORF                    ; Clears error status
        endif
        RET

;***************************************
;*         Time out processing
;***************************************
TM3_SEG CSEG
INTTM3:
        SEL RB0
        if_bit(ACKWFG)                        ; Acknowledge signal wait status?
            if_bit(ACKD)                      ; Acknowledge signal received?
                CLR1     ACKWFG               ; Clears acknowledge signal wait status
                CLR1     BUSYFG               ; Clears busy status
            else
                ACKCT--
                if(ACKCT==#0)                 ; Time out?
                    SET1    ACKT              ; Time out error processing
                    SET1    ERRORF
                    CLR1    ACKWFG            ; Clears acknowledge signal wait status
                    CLR1    BUSYFG            ; Clears busy status
                endif
            endif
        endif
```

8.3.2 Application as slave CPU

A slave CPU receives addresses, commands, and data from the master CPU and transmits data to the master CPU.

In the example shown in this section, addresses are received by using the wake-up function. This function is to generate an interrupt only when the address value transmitted by the master to the slave coincides with the value set to the slave address register (SVA) of the slave in the SBI mode. Therefore, only the slave CPU selected by the master CPU generates INTCSI0, and the slave CPUs not selected operates without generating an inadvertent interrupt request.

The slave CPU clears the wake-up function when it has been selected by the master (the interrupt request signal is generated at the end of transmission), and interfaces with the master CPU. Addresses, commands, and data being transmitted are identified by using RELD and CMDD (bits 2, 3 of serial bus interface control register (SBIC)) of the serial bus interface control register (SB IC).

Because the slave CPU is not automatically placed in the unselect status, a program that returns the slave CPU to the unselect status must be prepared by processing commands between the master and CPU.

(1) Description of package

<Public declaration symbol>

RCVDAT: Receive data storage area

<Register used>

Bank 0: A

<RAM used>

Name	Usage	Attribute	Bytes
RCVDAT	Stores receive data	SADDR	1

<Flag used>

Name	Usage
RCVFLG	Sets reception mode

<Nesting>

1 level 3 bytes

<Hardware used>

- Serial interface channel 0

<Initial setting>

• Setting of serial interface channel 0	
Sets SBI mode, SBI pin, and wake-up mode, and inputs serial clock from external source	CSIM0=#10010011B
• Outputs synchronous busy signal	BYSE=1
• Makes SO0 latch high	RELT=1
• Slave address	SVA=#SLVADR
• Enables serial interface channel 0 interrupt	

<Starting>

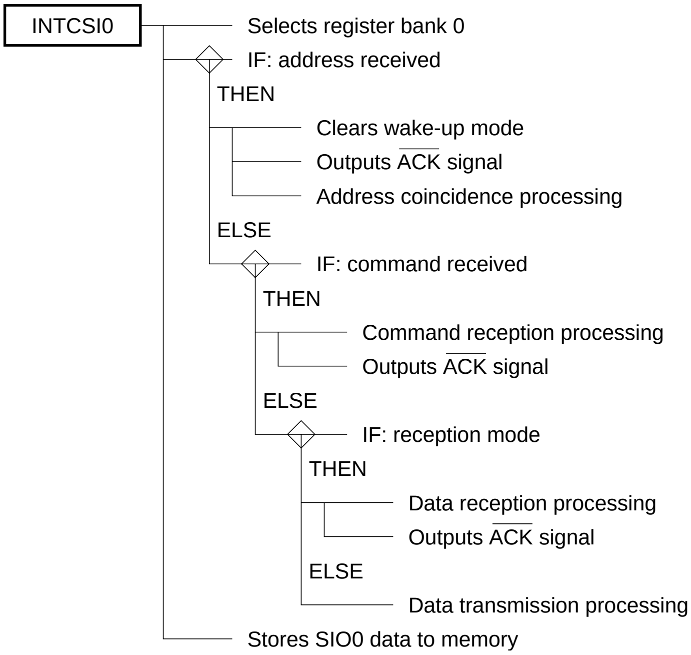
The interrupt processing is started by generation of INTCSI0. The interrupt processing performs the following processing:

- Identifies address/command/data
- Outputs $\overline{\text{ACK}}$ signal
- Stores receive data to RCVDAT

(2) Example of use

```
EXTRN   RCVDAT
EXTBIT  RCVFLG

SLVADR  EQU    5AH
SB1     EQU    P2.5
     :
     :
SET1    SB1
CSIM0=#10110100B      ; Inputs external clock, sets SB1 pin, and selects wake-up mode
SET1    RELT          ; Sets output latch to high level
SET1    BSYE          ; Sets busy automatic output
SVA=#SLVADR           ; Sets slave address
SIO0=#0FFH            ; Serial transfer start command
CLR1    SB1
CLR1    CSIMK0        ; Enables serial interface channel 0 interrupt
EI                    ; Enables master interrupt
```

(3) SPD chart


INTCSI0
Selects register bank 0
IF: address received
THEN
Clears wake-up mode
Outputs $\overline{\text{ACK}}$ signal
Address coincidence processing
ELSE
IF: command received
THEN
Command reception processing
Outputs $\overline{\text{ACK}}$ signal
ELSE
IF: reception mode
THEN
Data reception processing
Outputs $\overline{\text{ACK}}$ signal
ELSE
Data transmission processing
Stores SIO0 data to memory

(4) Program list

```
VECSI0  CSEG   AT 0EH
        DW     INTCSI0                                       ; Sets vector address of serial
                                                               interface channel 0
SCI_DAT DSEG   SADDR
RCVDAT: DS     1                                             ; Receive data storage area

CSI_FLG BSEG
RCVFLG  DBIT                                                 ; Sets reception mode

CSI_SEG CSEG
;***************************************
;*     INTCSI0 interrupt processing
;***************************************
INTCSI0:
        SEL RB0
        if_bit(RELD)                                         ; To address reception
            CLR1       WUP                                   ; Clears wake-up mode
            SET1       ACKT                                  ; Outputs acknowledge signal
;    User processing (address reception)

;***************************************

        elseif_bit(CMDD)                                     ; To command reception
;           User processing (command reception)

            SET1       ACKT                                  ; Outputs acknowledge signal
        else

            if_bit(RCVFLG)
;               User processing (data reception processing)
                SET1    ACKT                                 ; Outputs acknowledge signal
            else
;               User processing (data transmission processing)
            endif
;***************************************
        endif
        RCVDAT=SIO0 (A)

        RETI
```

8.4 Interface in 3-Wire Serial I/O Mode

In this section, examples of communication between the master and a slave by using the 3-wire serial I/O mode (serial clock, data input, data output) of the serial channel 0 of the 78K/0 series are shown. In these examples, one extra busy signal is used as a handshake signal for simultaneous transmission/reception between the master and slave. This busy signal is active-low and is output by the slave. The data is 8 bits long and transmitted with the MSB first. In the examples in this section, the μPD78014 subseries is used.

Figure 8-31. Example of Connection in 3-Wire Serial I/O Mode

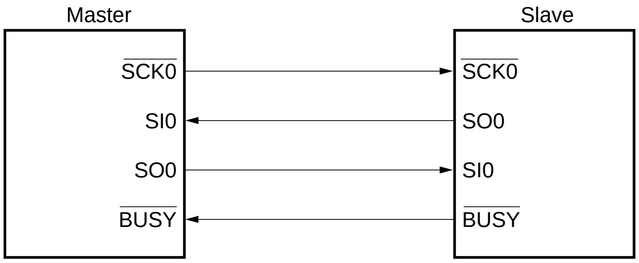
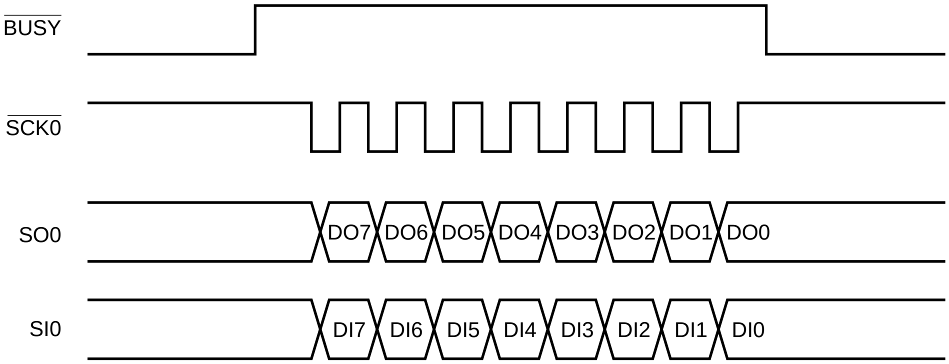


Figure 8-32. Communication Format in 3-Wire Serial I/O Mode



8.4.1 Application as master CPU

The serial clock is set to $f_X/2^5$, and communication is executed in synchronization with this serial clock between the master and slave CPUs.

The master CPU starts transmission after it has set the transmit data. If the slave CPU is busy (when the busy signal is low), however, the master does not transmit data and sets the busy flag (BUSYFG).

(1) Description of package

<Public declaration symbol>

TRANS : Name of 3-wire transfer subroutine of master
TDATA : Transmit data storage area
RDATA : Receive data storage area
BUSY : Busy signal input port
TREND : Transfer end test flag
BUSYFG : Busy status test flag

<Register used>

Interrupt : Bank 0, A
Subroutine : A

<RAM used>

Name	Usage	Attribute	Bytes
TDATA	Stores transmit data	SADDR	1
RDATA	Stores receive data		

<Flag used>

Name	Usage
TREND	Sets transfer end status
BUSYFG	Sets busy status

<Nesting>

2 levels 5 bytes

<Hardware used>

- Serial interface channel 0
- P33

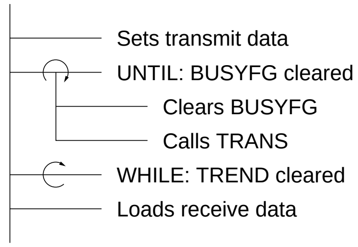
<Initial setting>

- Setting of serial interface channel 0
  3-wire serial I/O mode, MSB first CSIM0=#10000011B
- Serial clock $f_X/2^5$ TCL3=#××××1001B
- Makes P27 output latch high P27=1
- P33 input mode
- Enables serial interface channel 0 interrupt

<Starting>

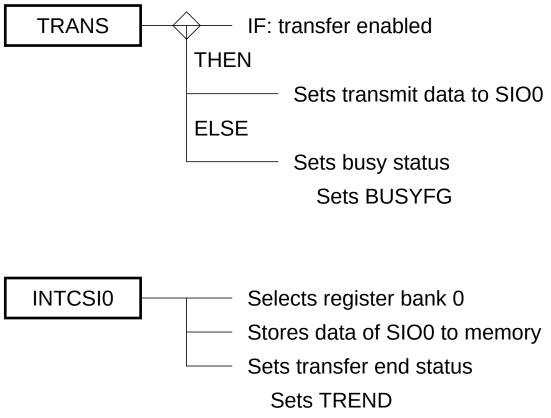
Set the transmit data to TDATA and call TRANS. After execution has returned from the subroutine, test the busy flag (BUSYFG). If the busy flag is set, transfer has not been executed and therefore, you must execute it again. If the busy flag is cleared, transfer has ended and the receive data has been stored to RDATA.

(2) Example of use



```
EXTRN   TDATA,RDATA,TRANS
EXTBIT  TREND,BUSYFG,BUSY

SCK0    EQU     P2.7
         :
         :
    CSIM0=#10000011B                ; Sets 3-wire serial I/O mode with MSB first
    TCL3=#10011001B                 ; Sets SCK0 = 262 kHz
    SET1    SCK0
    SET1    PM3.3                   ; Sets P3.3 input mode
    CLR1    CSIMK0                  ; Enables serial interface channel 0
    EI

         :
         :
    TDATA=A                         ; Sets transmit data
    repeat
            CLR1    BUSYFG          ; Busy test
            CALL    !TRANS
    until_bit(!BUSYFG)
    while_bit(!TREND)               ; Ends transfer
    endw
    A=RDATA                         ; Loads receive data
```

(3) SPD chart



(4) Program list

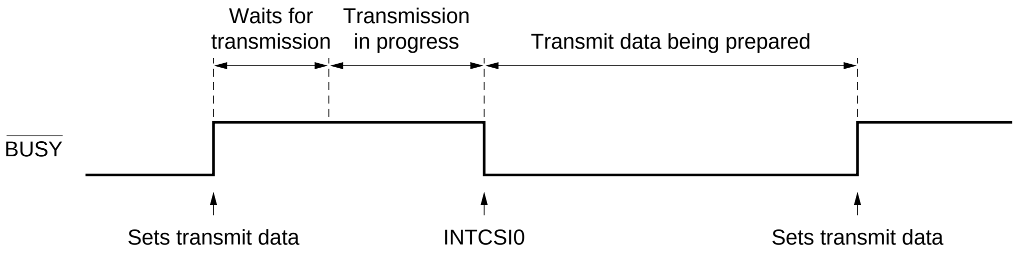
```
        PUBLIC  TRANS,RDATA,TDATA,BUSY,TREND,BUSYFG
VECSI0  CSEG    AT 0EH
        DW      INTCSI0                            ; Sets vector address of serial interface channel 0

BUSY    EQU     0FF03H.3                           ; 0FF03H = PORT3

CSI_DAT DSEG    SADDR
RDATA:  DS      1                                  ; Receive data storage area
TDATA:  DS      1                                  ; Transmit data storage area

CSI_FLG BSEG
TREND   DBIT                                       ; Sets transfer end status
BUSYFG  DBIT                                       ; Sets busy status

CSI_SEG CSEG

;*************************************
;:*    INTCSI0 interrupt processing
;*************************************
INTCSI0:
        SEL RB0
        RDATA=SIO0 (A)                             ; Stores receive data
        SET1    TREND                              ; Sets transfer end status
        RETI

;*************************************
;*        3-wire (master)
;*************************************
TRANS:
        if_bit(BUSY)
            SIO0=TDATA (A)                         ; Enables transfer
        else                                       ; Sets transmit data
            SET1    BUSYFG
        endif                                      ; Sets busy status
        RET
```

8.4.2 Application as slave CPU

In this example, a slave CPU simultaneously transmits and receives 8-bit data in synchronization with the serial clock from the master CPU. The busy signal output by the slave CPU is low (busy status) while the transmit data is prepared. This busy signal is cleared (high level) when the transmit data is set (CALL !TRANS), and is output (low level) when interrupt INTCSI0 occurs at the end of transfer.

Therefore, the busy status remains after the end of transfer until the data is set.

Figure 8-33. Output of Busy Signal



(1) Description of package

<Public declaration symbol>

TRANS	:	Name of 3-wire transfer subroutine of slave
TDATA	:	Transmit data storage area
RDATA	:	Receive data storage area
BUSY	:	Busy signal output port
TREND	:	Transfer end test flag

<Register used>

Interrupt	:	Bank 0, A
Subroutine	:	A

<RAM used>

Name	Usage	Attribute	Bytes
TDATA	Stores transmit data	SADDR	1
RDATA	Stores receive data		

<Flag used>

Name	Usage
TREND	Sets transfer end status

<Nesting>

2 level 5 bytes

<Hardware used>

- Serial interface channel 0
- P33

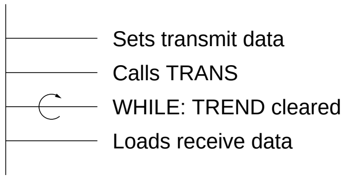
<Initial setting>

- Setting of serial interface channel 0
  Sets 3-wire serial I/O mode with MSB first, and inputs external clock CSIM0=#10000000B
- P33 output mode P33=0
- Setting of busy status
- Enables serial interface channel 0

<Starting>

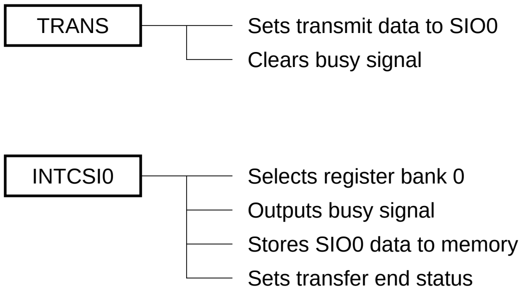
Set the transmit data to TDATA and call TRANS. Because the busy signal is cleared by the processing of TRANS, the slave waits for communication with the master. After the communication has ended, INTCSI0 occurs and interrupt processing is started. You can check the end of transfer by testing TREND. After TREND has been set, the receive data has been stored to RDTA.

(2) Example of use



```
EXTRN    TDATA,RDATA,TRANS
EXTBIT   TREND,BUSY
    :
    :
CSIM0=#10000000B                ; Sets 3-wire I/O mode with MSB first
CLR1     BUSY                   ; Busy status
CLR1     PM3.3                  ; P3.3 output mode
CLR1     CSIMK0                 ; Enables serial interface channel 0
EI
    :
    :
TDATA=A                         ; Sets transmit data
CALL     !TRANS
while_bit(!TREND)               ; Ends transfer
endw
A=RDATA                         ; Loads receive data
```

(3) SPD chart



(4) Program list

```
        PUBLIC  RDATA,TDATA,BUSY,TREND,BUSYFG
        PUBLIC  TRANS
VECSI0  CSEG    AT 0EH
        DW      INTCSI0                         ; Sets vector address of serial interface channel 0

CSI_DAT DSEG    SADDR
RDATA:  DS      1                               ; Stores receive data
TRADA:  DS      1                               ; Stores transmit data

CSI_FLG BSEG
TREND   DBIT                                    ; Sets transfer end status
BUSYFG  DBIT                                    ; Sets busy status

BUSY    EQU     0FF03H.3                        ; 0FF03H = PORT3

CSI_SEG CSE
;**************************************
;*    INTCSI0 interrupt processing
;**************************************
INTCSI0:
        SEL RB0
        CLR1    BUSY                            ; Sets busy status
        RDATA=SI00 (A)                          ; Stores receive data
        SET1    TREND                           ; Sets transfer end status
        RETI

;**************************************
;*        3-wire (slave)
;**************************************
TRANS:
        SIO0=TDATA (A)                          ; Sets transmit data
        SET1    BUSY                            ; Clears busy status
        RET
```

8.5 Half-Duplex Start-Stop Synchronization Communication

Half-duplex start-stop synchronization communication can be executed by using clocked serial interface channel 0. The three-wire serial I/O mode and SBI mode may be used. In this example, the μPD78014 subseries is used. The communication protocol is as follows:

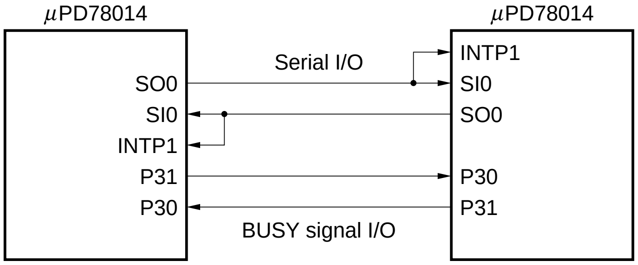
Transfer rate	:	9600 bps
Start bit	:	1 bit
Character length	:	8 bits (LSB first)
Parity bit	:	1 bit (even or odd parity selectable)
Stop bit	:	2 bits

The serial clock is generated by using 8-bit timer/event counter 2 because the transfer rate is set to 9600 bps.

8.5.1 Half-duplex start-stop synchronization communication in 3-wire serial I/O mode

Figure 8-34 shows the system configuration. Serial data are input to the SI0 pin and output from the SO0 pin. Bits 0 and 1 of port 3 are used to input and output the BUSY signal. When the BUSY signal is "L", serial communication is enabled.

Figure 8-34. System Configuration (in 3-wire serial I/O mode)



(1) Transmission in 3-wire serial I/O mode

Data are transmitted as follows:

<1> Start bit → Wait for transmission time by manipulating the output latch of the serial interface and by using 8-bit timer/event counter 2.

Caution To prevent data reception timing from being delayed because of missing of the start bit, increase the priority of INTP1 interrupt request.

<2> Data → Transmitted by serial buffer.

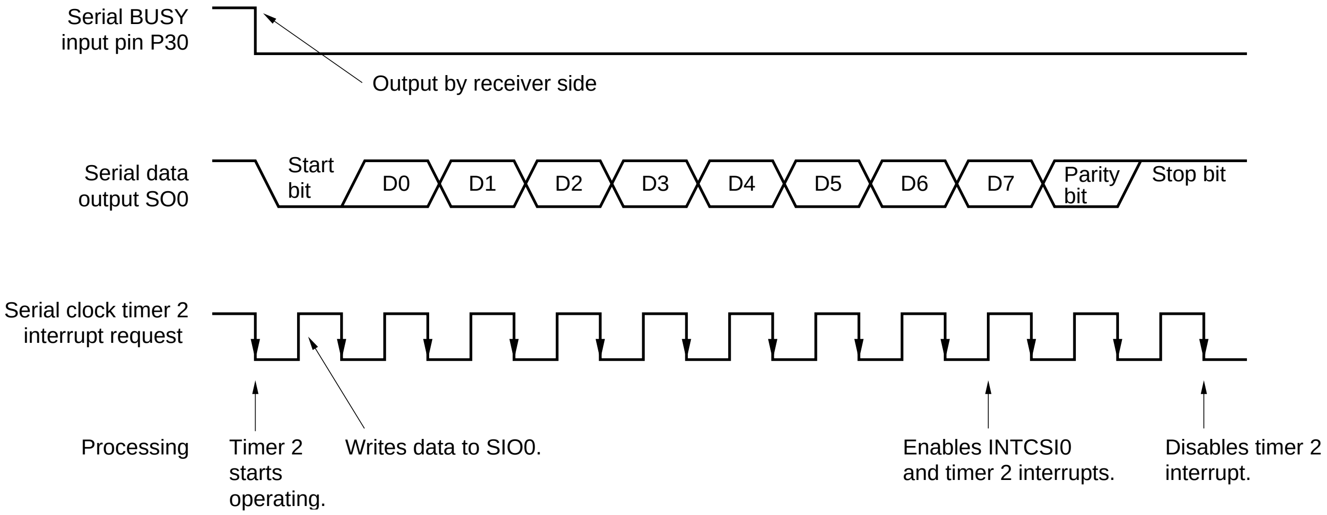
<3> Parity bit → Output the parity bit by manipulating the output latch of the serial interface through interrupt processing of 8-bit timer/even counter 2.

Caution To prevent transmission timing from being delayed, increase the priority of the 8-bit timer/event counter 2 interrupt request.

<4> Stop bit → Set the output latch of the serial interface through interrupt servicing of 8-bit timer/event counter 2 and output the stop bit.

Caution To prevent transmission timing from being delayed, increase the priority of the 8-bit timer/ event counter 2 interrupt request.

Figure 8-35. Transmission Format in 3-Wire Serial I/O Mode


Serial BUSY input pin P30
Output by receiver side
Serial data output SO0
Start bit
D0
D1
D2
D3
D4
D5
D6
D7
Parity bit
Stop bit
Serial clock timer 2 interrupt request
Processing
Timer 2 starts operating.
Writes data to SIO0.
Enables INTCSI0 and timer 2 interrupts.
Disables timer 2 interrupt.

(2) Reception in 3-wire serial I/O mode

Data are received as follows:

<1> Start bit → Reception is started by detecting the falling of the INTP1 pin and through port test.

Caution To prevent data reception timing from being delayed because of missing of the start bit, increase the priority of INTP1 interrupt request.

<2> Data → Received by serial buffer.

<3> Parity bit → Output the parity bit by testing the port with the interrupt processing of 8-bit timer/event counter 2.

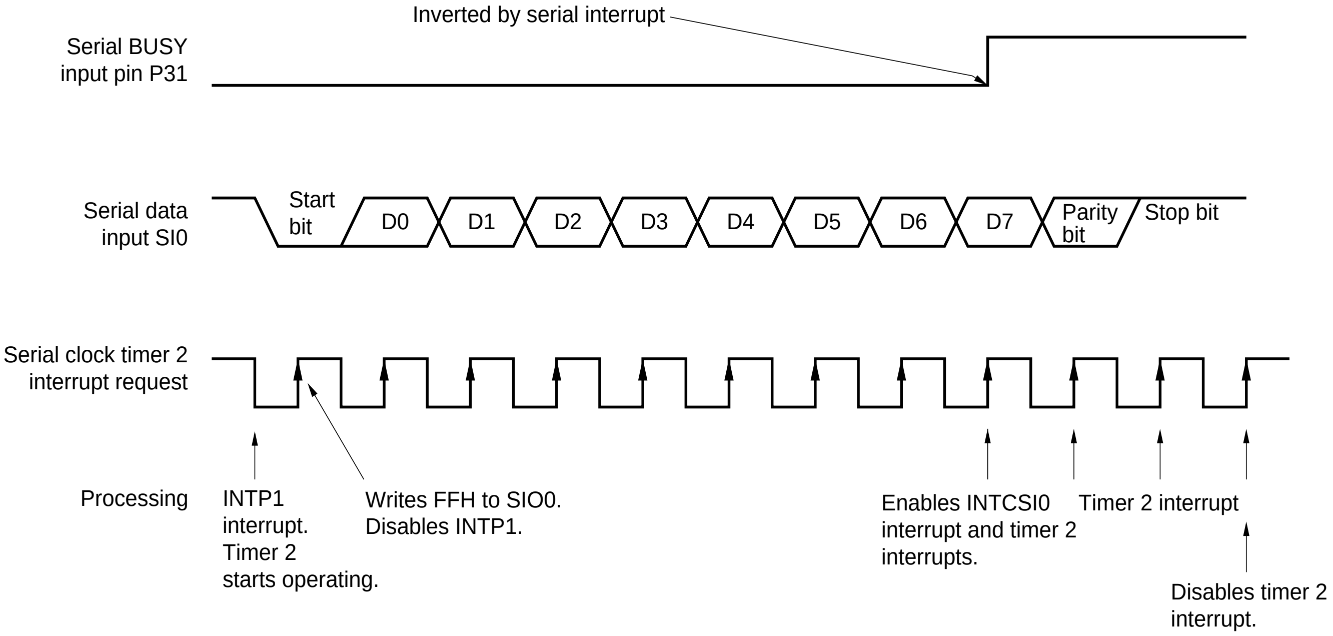
Caution To prevent reception timing from being delayed, increase the priority of the 8-bit timer/ event counter 2 interrupt request.

<4> Stop bit → Output the stop bit by testing the port with the interrupt servicing of 8-bit timer/event counter 2.

Caution To prevent reception timing from being delayed, increase the priority of the 8-bit timer/ event counter 2 interrupt request.

If a parity error or overrun error occurs, the flag is set.

Figure 8-36. Reception Format in 3-Wire Serial I/O Mode


Inverted by serial interrupt
Serial BUSY input pin P31
Serial data input SI0
Start bit
D0
D1
D2
D3
D4
D5
D6
D7
Parity bit
Stop bit
Serial clock timer 2 interrupt request
Processing
INTP1 interrupt. Timer 2 starts operating.
Writes FFH to SIO0. Disables INTP1.
Enables INTCSI0 interrupt and timer 2 interrupts.
Timer 2 interrupt
Disables timer 2 interrupt.

(3) Description of package

<Public declaration symbol>

- Subroutine name

S_SOSHIN	: Subroutine for transmission
S_JUSHIN	: Subroutine for reception

- Input parameters

SODATA	: Stores transmit data.
F_PARITY	: Indicates selected even or odd parity.
F_TUSHIN	: Indicates reception or transmission in progress.

- Output parameters

JUDATA	: Stores receive data.
F_DATA	: Set when reception is completed.
F_ERRP	: Indicates error of parity.
F_ERRE	: Indicates error of end bit

- I/O parameters

F_PADATA	: Stores transmitted/received parity bit.

<Registers used>

Bank 0	A
Bank 1	A
Bank 2	A

<RAM used>

Name	Usage	Attribute	Bytes
SODATA	Transmit data storage area	SADDR	1
JUDATA	Receive data storage area	SADDR	1
C_WORK	Status storage counter	SADDR	1
i	Loop processing work counter	SADDR	1
j	Loop processing work counter	SADDR	1

<Flags used>

Name		Usage
F_PARITY	Parity select flag	Set when odd parity is selected
F_PADATA	Parity bit storage flag	Stores parity.
F_TUSHIN	Communication flag	Set during communication.
F_ERRP	Parity error flag	Set in case of parity error.
F_ERRE	End bit error flag	Set in case of end bit error.
F_DATA	Reception completion flag	Set on completion of reception.
F_WORK	Work flag	For work

<Nesting>

1 level 3 bytes

<Hardware used>

- Serial interface channel 0 (3-wire serial I/O mode)
- 8-bit timer/event counter 2
- External interrupt edge detection (INTP1 pin)

<Initial setting>

• Set by subroutines S_SOSHIN and S_JUSHIN.	
• Port 2: 5-bit input port, 6-bit output port	PM2 = #×01×××××B
• Port 3: 0-bit input port, 1-bit output port	PM3 = #××××××01B
• Setting of serial interface channel 0	
3-wire serial I/O mode, serial clock = selected by 8-bit timer/event counter 2	
	CSIM0 = #10000110B
• Setting of 8-bit timer/event counter 2	
Baud rate: 9600 bps	CR20 = #54
8-bit timer register × 2 channels mode	TCL1 = #01110000B
8-bit timer/event counter 2 disabled	TOC1 = #0000000B
	TMC1 = #0000000B
• Setting of INTP1: Falling edge of INTP1	INTM0 = #0000000B
• 8-bit timer/event counter 2 interrupt priority: high	CLR1 TMPR2
• INTP1 interrupt priority: high	CLR1 PPR1
• Serial interface interrupt enabled	CLR1 CSIMK0

<Starting>

- Start data transmission and reception in the following sequence:

  - Starting data transmission
    <1> Store the transmit data to the SODATA area.
    <2> Set the transmit flag.
    <3> Call subroutine S_SOSHIN.
  - Starting data reception
    <1> Clear the communication flag (F_TUSHIN) to 0.
    <2> Invert the BUSY signal.
    <3> Call subroutine S_JUSHIN.

- When using an interrupt request other than those of the 78K/0 series package, clear the ISP flag to 0 before interrupt servicing to enable the interrupt request in order to enable a high-priority interrupt.

(4) Example

Here is an example of selecting odd or even parity bit, and transmission or reception through key input.

```
EXTRN   SODATA
EXTRN   JUDATA,S_SOSHIN,S_JUSHIN
EXTBIT  F_PARITY,F_DATA,F_PADATA,F_TUSHIN
EXTBIT  F_ERRE,F_ERRP
;
BUSY_0  EQU P3.1
BUSY_1  EQU P3.0
PARIKEY EQU 22                                  ; Decoded value of parity key
JYUSHIN EQU 21                                  ; Decoded value of reception key
TUSHIN  EQU 20                                  ; Decoded value of transmission key
;************************************************
;                   Initialize
;************************************************
VERES   CSEG    AT 00H
    DW  RES_STA
M3          CSEG                                ;
RES_STA:                                        ;
    MOV P2,#0BFH                                ; P2.5 = H, P2.6 = L
    MOV P3,#0FFH                                ;
    MOV PM2,#00100000B                          ; P2.5 = input port, P2.6 = output port
    MOV PM3,#00000001B                          ; P3.0 = input port, P3.1 = output port
;***Setting of 8-bit timer register***
    CR20=#54                                    ;
    TCL1=#01110000B                             ; Count clock: 1.05 MHz
    TOC1=#00000000B                             ;
    TMC1=#00000000B                             ; Selects 8-bit timer register, disables timer 2 operation
;***Setting of serial interface 0***
    CSIM0=#10000110B                            ; Selects 3-wire mode, serial clock, 8-bit timer 2
    SET1    RELT                                ;
;***Setting of INTP1***
    INTM0=#00000000B                            ; INTP1 falling edge
    CLR1    TMPR2                               ; Timer 2 interrupt high priority
    CLR1    PPR1                                ; INTP1 interrupt high priority
    CLR1    PIF1                                ; Clears INTP1 request flag
    CLR1    TMIF2                               ; Clears timer 2 interrupt request flag
    CLR1    CSIIF0                              ; Clears serial interface interrupt flag
    CLR1    CSIMK0                              ; Enables serial interface interrupt
    while(forever)                              ;
                                                ;
```

```
if_bit(F_KEYON)                         ; Key ON flag 1?
    switch(M_KEYON)                     ;
    case PARIKEY:                       ; Parity key is pressed.
        SET1    CY                      ;    Alternately detects odd and even parities.
        CY ^= F_PARITY                  ;
        F_PARITY=CY                     ;
        break                           ;
    case TUSHIN:                        ; Communication key is pressed.
        SET1    F_TUSHIN                ;    Sets communication flag (during transmission).
        CLR1    F_SOEND                 ;
        break                           ;
    case JYUSHIN:                       ; Communication key is pressed.
        CLR1    F_TUSHIN                ; Clears communication flag (during reception).
        CY=BUSY_0                       ;    Outputs inverted BUSY signal data.
        NOT1    CY                      ;
        BUSY_0=CY                       ;
        if_bit(CY)                      ;
            SET1    PMK1                ; Disables INTP1 interrupt.
        else                            ;
            CLR1    F_ERRP              ;
            CLR1    F_ERRE              ;
            CALL    !S_JUSHIN           ;
        endif                           ;
        break                           ;
    ends                                ;
endif                                   ;
      •
      •
      •
if_bit(!F_SOEND)                        ;
    if_bit(F_TUSHIN)                    ; Communication flag set?
        CY=BUSY_I                       ;    BUSY signal non-active?
        if_bit(!CY)                     ;
            SODATA=#0                   ;
            SET1    F_SOEND             ;
            SODATA=WORK                 ; Transmit data storage area ← Transmit data
            CALL    !S_SOSHIN           ;
        endif                           ;
    endif                               ;
endif                                   ;
```

(5) SPD chart

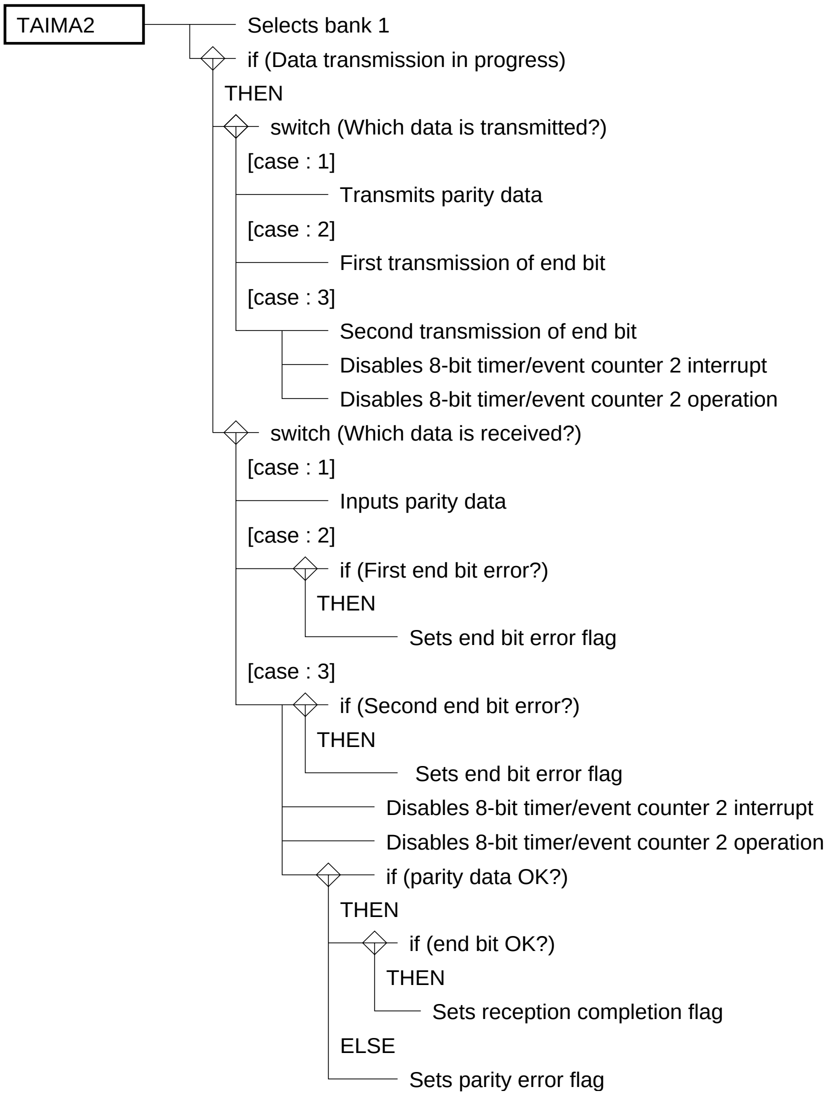
[Reception subroutine]

```
S_JUSHIN
    Clears INTP1 request flag
    Enables INTP1 interrupt
```

[Transmission subroutine]

```
S_SOSHIN
    if (odd parity selected)
      THEN
        Sets parity data flag
    for (i = # 0 ; i < #8 ; i++)
        CY ← Least significant bit of transmit data
        Exclusive-ORs CY and parity data flag
        Transfers result to parity data flag
    Resets timer output F/F of 8-bit timer/event counter 2
    and enables inversion operation
    Clears request flag of 8-bit timer/event counter 2
    Disables interrupt
    Enables 8-bit timer/event counter 2 operation
    Transmits start bit
    Waits for start bit transmission time
    SIO0 ← transmit data
    Enables interrupt
```

[Parity end bit transmission processing (8-bit timer/event counter 2 interrupt)]



```
TAIMA2
    Selects bank 1
    if (Data transmission in progress)
    THEN
        switch (Which data is transmitted?)
            [case : 1]
                Transmits parity data
            [case : 2]
                First transmission of end bit
            [case : 3]
                Second transmission of end bit
                Disables 8-bit timer/event counter 2 interrupt
                Disables 8-bit timer/event counter 2 operation
        switch (Which data is received?)
            [case : 1]
                Inputs parity data
            [case : 2]
                if (First end bit error?)
                THEN
                    Sets end bit error flag
            [case : 3]
                if (Second end bit error?)
                THEN
                    Sets end bit error flag
                Disables 8-bit timer/event counter 2 interrupt
                Disables 8-bit timer/event counter 2 operation
                if (parity data OK?)
                THEN
                    if (end bit OK?)
                    THEN
                        Sets reception completion flag
                ELSE
                    Sets parity error flag
```

[Data transmission/reception completion processing]

```
INTSI0
    Selects bank 2
    Clears 8-bit timer/event counter 2 request flag
    Enables 8-bit timer/event counter 2 interrupt
    Outputs "H" to BUSY0 signal
    if  (reception in progress)
      THEN
        Inputs receive data
```

[Data reception start processing (INTP1 interrupt processing)]

```
INTP1
    Selects bank 1
    Clears 8-bit timer/event counter 2 request flag
    Clears 8-bit timer/event counter
    Enables 8-bit timer/event counter 2 operation
    Waits for start bit input time
    if (INTP1 pin check OK?)
      THEN
        Disables INTP1 interrupt
          Prepares for input of receive data
```

(6) Program list

```
PUBLIC  F_PADATA,F_PARITY
PUBLIC  F_DATA,F_TUSHIN
PUBLIC  JUDATA,SODATA,S_JUSHIN,S_SOSHIN
PUBLIC  F_ERRP,F_ERRE
;
VEINTP1     CSEG    AT 08H
            DW      INTP1                       ; Sets vector address of INTP1
VEINTSI0    CSEG    AT 0EH
            DW      INTSI0                      ; Sets vector address of serial interface channel 0
VETIM2      CSEG    AT 18H
            DW      TAIMA2                      ; Sets vector address of 8-bit timer 2
;
SI0         EQU     P2.5
BUSY_0      EQU     P3.1
BUSY_I      EQU     P3.0
;
MORAM       DSEG    SADDR
SODATA:     DS      1                           ; Transmit data storage area
C_WORK:     DS      1                           ; Work counter
JUDATA:     DS      1                           ; Receive data storage area
i:          DS      1                           ; Work counter
k:          DS      1                           ; Work counter
;
MOFLG       BSEG
F_PARITY    DBIT                                ; Parity select flag
F_ERRP      DBIT                                ; Parity error flag
F_ERRE      DBIT                                ; End bit error flag
F_DATA      DBIT                                ; Reception completion flag
F_PADATA    DBIT                                ; Parity data flag
F_WORK      DBIT                                ; Work flag
F_TUSHIN    DBIT                                ; Communication flag
;************************************************
;               Reception routine
;************************************************
JUSHIN  CSEG                                    ;
S_JUSHIN:                                       ;
    CLR1    PIF1                                ; Clears INTP1 request flag
    CLR1    PMK1                                ; Enables INTP1 interrupt
    RET                                         ;
```

```
;************************************************
;              Transmission routine
;************************************************
SOSHIN  CSEG
S_SOSHIN:                                       ;
    CLR1    F_PADATA                            ; Clears parity data flag
    if_bit(F_PARITY)                            ; Odd parity selected?
        SET1    F_PADATA                        ;    Sets parity data
    endif                                       ;
    A=SODATA                                    ;
    for(i=#0;i<#8;i++)                          ; Determines parity data
        RORC    A,1                             ;
        CY ^= F_PADATA                          ;
        F_PADATA = CY                           ;
    next                                        ;
    TOC1=#01100000B (A)                         ;
    CLR1    TMIF2                               ; Clears timer 2 request flag
    DI                                          ;
    SET1    TCE2                                ; Enables 8-bit timer operation
    SET1    CMDT                                ; Transmits start bit
    while_bit(!TMIF2)                           ; Waits for start bit transmission time
    endw                                        ;
    CLR1    TMIF2                               ;
    SIO0=SODATA (A)                             ; Starts data transmission
    EI                                          ;
    RET                                         ;
;************************************************
;           Timer 2 interrupt processing
;************************************************
TIM2        CSEG                                ;
TAIMA2:                                         ;
    SEL RB1                                     ; Sets bank 1
    if_bit(F_TUSHIN)                            ; Communication flag set?
        if(C_WORK <= #4)                        ;    Contents of work counter
            switch(C_WORK)                      ;    0: Transmits parity data
            case 0:                             ;
                if_bit(F_PADATA)                ;
                    SET1    RELT                ;
                else                            ;
                    SET1    CMDT                ;
                endif                           ;
                break                           ;
            case 2:                             ;    2: Transmits end bit
                SET1    RELT                    ;    Transmits “H”
                break                           ;
            case 4:                             ;    4: Transmits end bit
                SET1    RELT                    ;    Transmits “H”
                SET1    TMMK2                   ;    Disables timer 2 interrupt
                CLR1    TCE2                    ;    Disables 8-bit timer operation
                C_WORK=#0                       ;
                break                           ;
            ends                                ;
            C_WORK++                            ;
        else                                    ;
            C_WORK=#0                           ;
        endif                                   ;
```

```
else                                         ;
    if(C_WORK <= #6)                         ; Reception in progress?
        switch(C_WORK)                       ;    Contents of work counter
        case 1:                              ;    1: Inputs parity data
            CY=SI0                           ;
            F_PADATA=CY                      ;
            break                            ;
        case 3:                              ;    3: Checks end bit
            if_bit(!SI0)                     ;    If error, sets end bit error flag
                SET1    F_ERRE               ;
            endif                            ;
            break                            ;
        case 5:                              ;    5: Checks end bit
            if_bit(!SI0)                     ;    If error, sets end bit error flag
                SET1    F_ERRE               ;
            endif                            ;
            C_WORK=#0                        ;
            SET1    TMMK2                    ;    Disables timer 2 interrupt
            CLR1    TCE2                     ;    Disables 8-bit timer operation
            CLR1    F_WORK                   ;
            if_bit(F_PARITY)                 ;
                SET1    F_WORK               ;
            endif                            ;
            A=JUDATA                         ;
            for(i=#0;i<#8;i++)               ; Adds receive data
                RORC  A,1                    ;
                CY ^= F_WORK                 ;
                F_WORK = CY                  ;
            next                             ;
            CLR1    F_ERRP                   ;
            CLR1    F_DATA                   ;
            F_WORK ^= F_PADATA (CY)          ;
            if_bit(!F_WORK)                  ; Checks parity bit
               if_bit(!F_ERRE)               ; Checks end bit data
                  SET1    F_DATA             ;
               endif                         ;
            else                             ;    Sets F_DATA if parity data is OK
                SET1    F_ERRP               ;    Sets parity error flag if parity data is not OK
            endif                            ;
            break                            ;
        ends                                 ;
        C_WORK++                             ;
    else                                     ;
        C_WORK=#0                            ;
    endif                                    ;
endif                                        ;
RETI                                         ;
```

```
;************************************************
;     INTSI0 interrupt processing (reception)
;************************************************
S_SI0   CSEG                                    ;
INTSI0:                                         ;
    SEL RB2                                     ; Sets bank 2
    CLR1    TMIF2                               ; Clears timer 2 request flag
    CLR1    TMMK2                               ; Enables timer 2 interrupt
    SET1    BUSY_O                              ; Outputs BUSY signal “H”
    if_bit(!F_TUSHIN)                           ;
        JUDATA=SIO0 (A)                         ;
    endif                                       ;
    C_WORK=#0                                   ; Clears work counter to zero
    RETI                                        ;
;************************************************
;     INTP1 interrupt processing (reception)
;************************************************
S_P1    CSEG                                    ;
INTP1:                                          ;
    SEL RB1                                     ;
    CLR1    TMIF2                               ; Clears timer 2 request flag
    CLR1    TCE2                                ; Clears timer 2 counter
    SET1    TCE2                                ; Enables timer operation
    while_bit(!TMIF2)                           ;
    endw                                        ;
    CLR1    TMIF2                               ;
    if_bit(!SI0)                                ; Chattering processing of INTP1
        TOC1=#10100000B                         ;
        SET1    PMK1                            ; Disables INTP1 interrupt
        SI00=#0FFH                              ;
    endif                                       ;
    RETI                                        ;
    END
```

8.5.2 Half-duplex start-stop synchronization communication in SBI mode

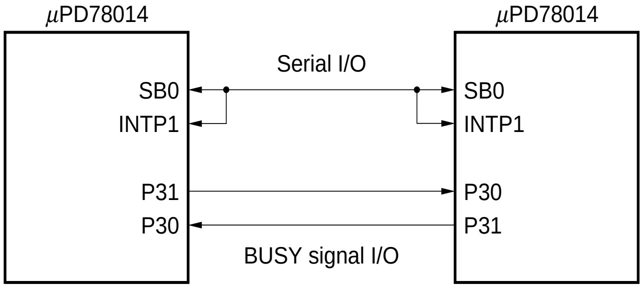
Figure 8-37 shows the system configuration. Serial data is input or output to or from the SB0 pin. Bits 0 and 1 of port 3 are used to input or output the BUSY signal. Serial communication is enabled when the BUSY signal is ‘L’.

Note the following points when using the SBI mode.

<1> Set the fifth bit (SB0) of port 2 in the output mode on resetting and restarting. To test SB0, however, set SB0 in the input mode. After testing, set SB0 in the output mode again.

<2> After transmission and detection of the last stop bit for serial transmission/reception have been completed, disable the serial operation, and then enable it again.
Usually, the end of SBI communication is detected by checking the ready signal after the acknowledge signal has been output. Because the acknowledge signal is used to transmit or receive the parity bit in this example, however, the SBI communication end condition is not satisfied if parity bit “1” is transmitted or received. If the end of serial communication is not detected, the next communication may not be performed correctly.

Figure 8-37. System Configuration (SBI mode)



(1) Transmission in SBI mode

Data are transmitted as follows:

<1> Start bit → Wait for transmission time by manipulating the output latch of the serial interface and by using 8-bit timer/event counter 2.

Caution To prevent data reception timing from being delayed because of missing of the start bit, increase the priority of INTP1 interrupt request.

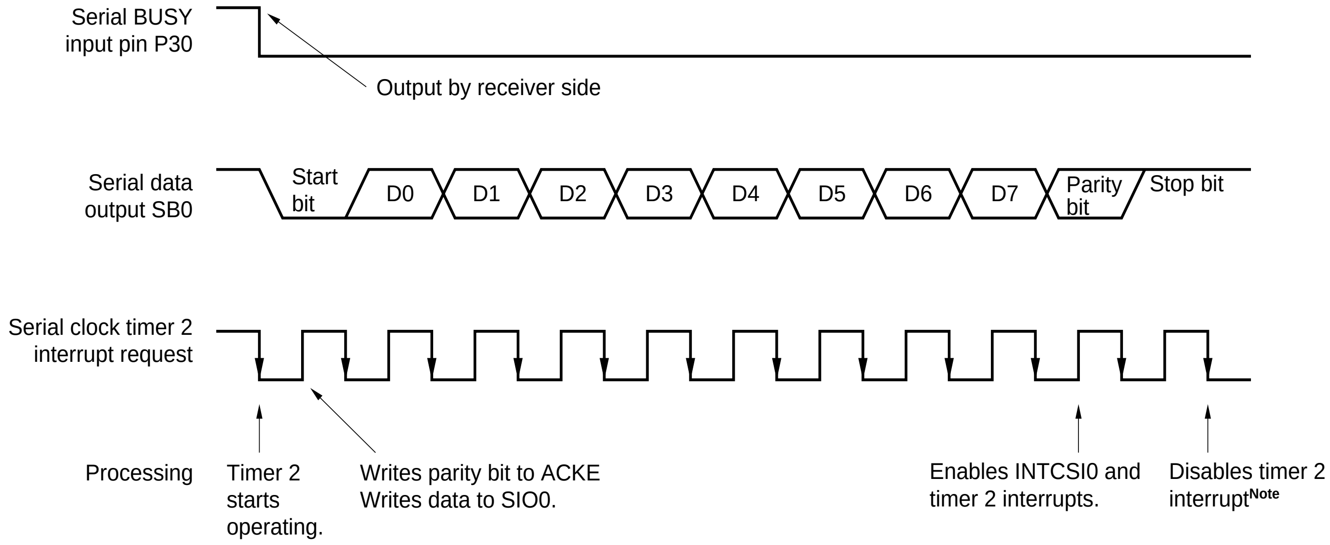
<2> Data and parity bit → Transmit 9 bits by using the serial buffer and acknowledge signal.

<3> Stop bit → Set the output latch of the serial interface through interrupt servicing of 8-bit timer/event counter 2 and output the stop bit.

Cautions 1. To prevent transmission timing from being delayed, increase the priority of the 8-bit timer/event counter 2 interrupt request.

2. After transmission of the second stop bit has been completed, disable the serial operation and then enable it again to check the completion of transmission.

Figure 8-38. Transmission Format in SBI Mode


Serial BUSY input pin P30
Output by receiver side
Serial data output SB0
Start bit
D0
D1
D2
D3
D4
D5
D6
D7
Parity bit
Stop bit
Serial clock timer 2 interrupt request
Processing
Timer 2 starts operating.
Writes parity bit to ACKE
Writes data to SIO0.
Enables INTCSI0 and timer 2 interrupts.
Disables timer 2 interrupt[Note]

Note Enable the interrupt after disabling the serial operation.

(2) Reception in SBI mode

Data are received as follows:

<1> Start bit → Reception is started by detecting the falling of the INTP1 pin and through port test.

Cautions 1. Test the port in the following sequence:

<1> Set bit 5 (SI0) of port 2 in the input mode.

<2> Test the port. Write data to SIO0.

<3> Set bit 5 of port 2 in the output mode again.

2. To prevent data reception timing from being delayed because of missing of the start bit, increase the priority of INTP1 interrupt request.

<2> Data and parity bit → Reception by serial buffer and acknowledge detection

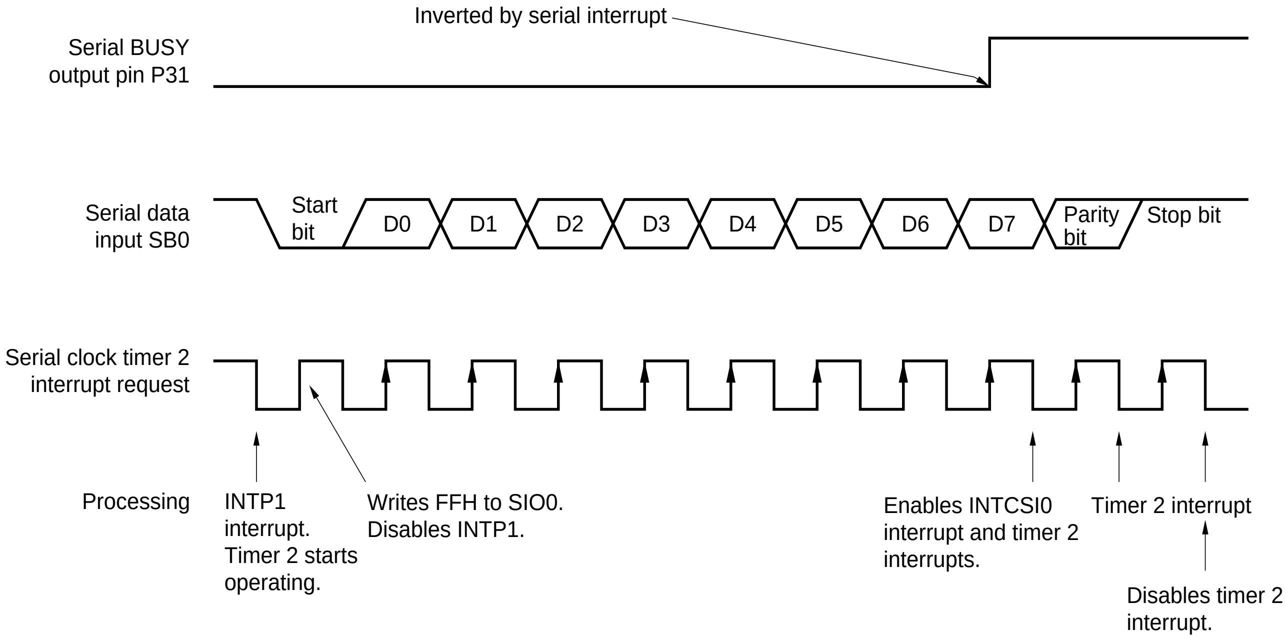
<3> Stop bit → Test the port by 8-bit timer/event counter 2 interrupt servicing and output the parity bit.

Cautions 1. To prevent transmission timing from being delayed, increase the priority of the 8-bit timer/ event counter 2 interrupt request.

2. After transmission of the second stop bit has been completed, disable the serial operation and then enable it again to check the completion of transmission.

If a parity error or overrun error occurs, the flag is set.

Figure 8-39. Reception Format in SBI Mode


Inverted by serial interrupt
Serial BUSY output pin P31
Serial data input SB0
Start bit
D0
D1
D2
D3
D4
D5
D6
D7
Parity bit
Stop bit
Serial clock timer 2 interrupt request
Processing
INTP1 interrupt. Timer 2 starts operating.
Writes FFH to SIO0. Disables INTP1.
Enables INTCSI0 interrupt and timer 2 interrupts.
Timer 2 interrupt
Disables timer 2 interrupt.

(3) Description of package

<Public declaration symbol>

- Subroutine name

S_SOSHIN	: Subroutine for transmission
S_JUSHIN	: Subroutine for reception

- Input parameters

SODATA	: Stores transmit data.
F_PARITY	: Indicates selected even or odd parity.
F_TUSHIN	: Indicates reception or transmission in progress.

- Output parameters

JUDATA	: Stores receive data.
F_DATA	: Set when reception is completed.
F_ERRP	: Indicates error of parity.
F_ERRE	: Indicates error of end bit

- I/O parameters

F_PADATA	: Stores transmitted/received parity bit.

<Registers used>

Bank 0	A
Bank 1	A
Bank 2	A

<RAM used>

Name	Usage	Attribute	Bytes
SODATA	Transmit data storage area	SADDR	1
JUDATA	Receive data storage area	SADDR	1
C_WORK	Status storage counter	SADDR	1
i	Loop processing work counter	SADDR	1
j	Loop processing work counter	SADDR	1

<Flags used>

Name		Usage
F_PARITY	Parity select flag	Set when odd parity is selected
F_PADATA	Parity bit storage flag	Stores parity.
F_TUSHIN	Communication flag	Set during communication.
F_ERRP	Parity error flag	Set in case of parity error.
F_ERRE	End bit error flag	Set in case of end bit error.
F_DATA	Reception completion flag	Set on completion of reception.
F_WORK	Work flag	For work

<Nesting>

1 level 3 bytes

<Hardware used>

- Serial interface channel 0 (SBI mode)
- 8-bit timer/event counter 2
- External interrupt edge detection (INTP1 pin)

<Initial setting>

- Set the pin used to input/output data (P25) as follows after resetting and restarting, and before serial transfer of the first byte:
  <1> Set the output latch of P25 to 1.
  <2> Set bit 0 (RELT) of the serial bus control register (SBIC) to 1.
  <3> Clear the output latch of P25, which has been set to 1, to 0.
- Set by subroutines S_SOSHIN and S_JUSHIN.
- Port 2: 5-bit input port, 6-bit output port   PM2 = #×01×××××B
- Port 3: 0-bit input port, 1-bit output port   PM3 = #××××××01B
- Setting of serial interface channel 0
  SBI mode, serial clock = selected by 8-bit timer/event counter 2   CSIM0 = #10010110B
- Setting of 8-bit timer/event counter 2
  Baud rate: 9600 bps   CR20 = #54
  8-bit timer register × 2 channels mode   TCL1 = #01110000B
  8-bit timer/event counter 2 disabled   TOC1 = #00000000B
  TMC1 = #00000000B
- Setting of INTP1: Falling edge of INTP1   INTM0 = #00000000B
- 8-bit timer/event counter 2 interrupt priority: high   CLR1 TMPR2
- INTP1 interrupt priority: high   CLR1 PPR1
- Serial interface interrupt enabled   CLR1 CSIMK0

<Starting>

- Start data transmission and reception in the following sequence:

  - Starting data transmission
    <1> Store the transmit data to the SODATA area.
    <2> Set the transmit flag.
    <3> Call subroutine S_SOSHIN.
  - Starting data reception
    <1> Clear the communication flag (F_TUSHIN) to 0.
    <2> Invert the BUSY signal.
    <3> Call subroutine S_JUSHIN.

- When using an interrupt request other than those of the 78K/0 series package, clear the ISP flag to 0 before interrupt servicing to enable the interrupt request in order to enable a high-priority interrupt.

(4) Example

Here is an example showing how to select an even or odd parity bit, and transmission or reception through key input.

```
EXTRN   SODATA
EXTRN   JUDATA,S_SOSHIN,S_JUSHIN
EXTBIT  F_PADATA,F_PARITY,F_DATA,F_TUSHIN
EXTBIT  F_ERRP,F_ERRE
;
TUSHIN  EQU 20
JYUSHIN EQU 21
PARIKEY EQU 22
BUSY_O  EQU P3.1
BUSY_I  EQU P3.0
SB0     EQU P2.5
;***********************************************
;                   Initialize
;***********************************************
M3S         CSEG                                ;
RES_STA:                                        ;
    MOV P2,#9FH                                 ; P2.5=H, P2.6=L
    MOV P3,#0FFH                                ;
    MOV PM2,#00000000B                          ; P2.5 = output mode
    MOV PM3,#00000001B                          ; P3.0 = input mode, P3.1 = output mode
;***Setting of 8-bit timer register***
    CR20=#54                                    ;
    TCL1=#01110000B                             ; Count clock: 1.05 MHz
    TOC1=#00000000B                             ;
    TMC1=#00000000B                             ; Selects 8-bit timer register and disables timer 2 operation
;***Setting of serial interface 0***
    SET1    SB0                                 ;
    CSIM0=#10000110B                            ; Selects SBI mode, serial clock, and 8-bit timer 2
    SET1    RELT                                ;
    CLR1    SB0                                 ;
;***Setting of INTP1***
    CLR1    TMPR2                               ; Increases priority of timer 2 interrupt
    CLR1    PPR1                                ; Increases priority of INTP1 interrupt
    INTM0=#00000000B                            ; Falling edge of INTP1
    CLR1    PIF1                                ; Clears INTP1 request flag
    CLR1    TMIF2                               ; Clears timer 2 request flag
    CLR1    CSIIF0                              ; Clears serial interface request flag
    CLR1    KSIF                                ; Clears interrupt request flag
    CLR1    CSIMK0                              ; Enables serial interface interrupt
    CLR1    KSMK                                ; Enables INTKS interrupt
;
```

```
while(forever)                                 ;
      •                                        ;
      •                                        ;
    if_bit(F_KEYON)                            ; Key ON flag 1?
        switch(M_KEYON)                        ;
        case PARIKEY:                          ; Parity key is pressed.
            SET1    CY                         ;    Alternately detects odd and even parities.
            CY ^= F_PARITY                     ;
            F_PARITY=CY                        ;
            break                              ;
        case TUSHIN:                           ; Communication key is pressed.
            SET1    F_TUSHIN                   ;    Clears communication flag (during transmission).
            CLR1    F_SOEND                    ;
            break                              ;
        case JYUSHIN:                          ; Reception key is pressed.
            CLR1    F_TUSHIN                   ;    Clears communication flag (during reception)
            CY=BUSY_O                          ;       Outputs inverted BUSY signal data.
            NOT1    CY                         ;
            BUSY_O=CY                          ;
            if_bit(CY)                         ;
                SET1    PMK1                   ; Disables INTP1 interrupt.
            else                               ;
                CLR1    F_ERRP                 ;
                CLR1    F_ERRE                 ;
                CALL    !S_JUSHIN              ;
            endif                              ;
            break                              ;
        ends                                   ;
    endif
          •
          •
          •
    if_bit(!F_SOEND)                           ; Communication flag set?
        if_bit(F_TUSHIN)                       ;    BUSY signal non-active?
            CY=BUSY_I                          ;
            if_bit(!CY)                        ;
                SET1    F_SOEND                ;
                SODATA=#0                      ;
                SODATA=WORK (A)                ; Transmit data storage area ← Transmit data
                CALL    !S_SOSHIN              ; Calls transmission routine
            endif                              ;
        endif                                  ;
    endif                                      ;
```

(5) SPD chart

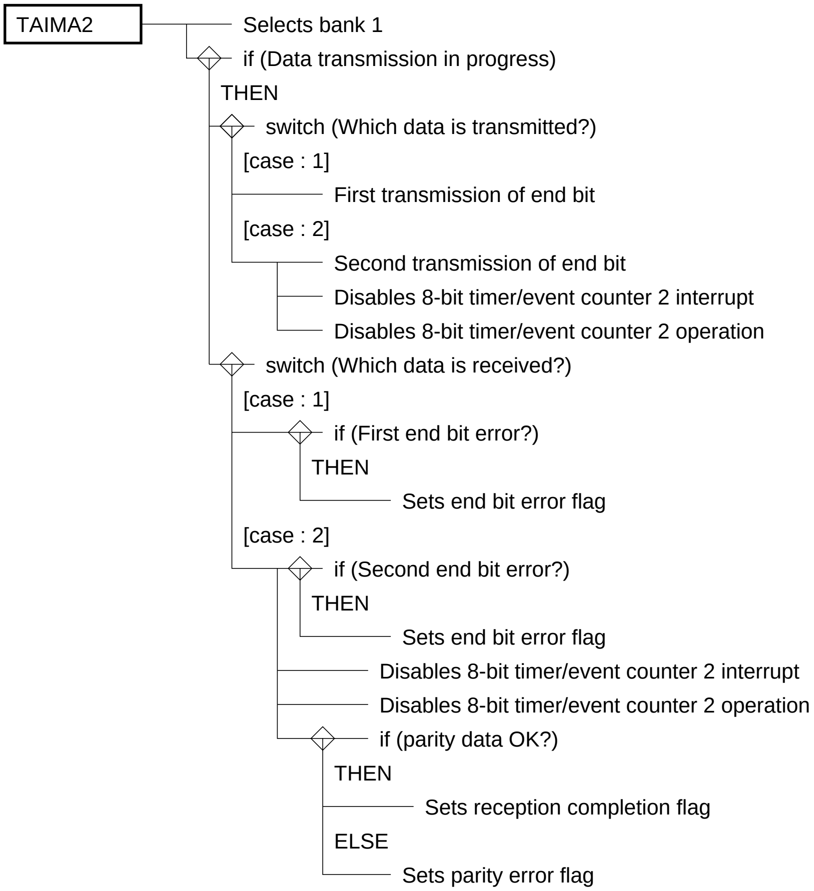
[Reception subroutine]

```
S_JUSHIN
    Clears INTP1 request flag
    Enables INTP1 interrupt
```

[Transmission subroutine]

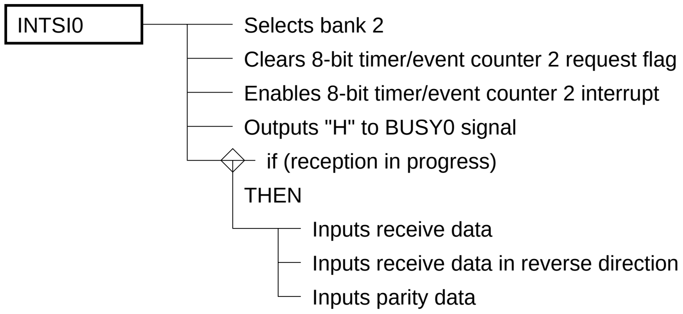
```
S_SOSHIN
    Converts transmit data in reverse direction
    if (odd parity selected)
      THEN
        Sets parity data flag
    for (i = # 0 ; i < # 8 ; i++)
        CY ← Least significant bit of transmit data
        Exclusive-ORs CY and parity data flag
        Transfers result to parity data flag
    Resets timer output F/F of 8-bit timer/event counter 2 and enables inversion operation
    Clears request flag of 8-bit timer/event counter 2
    Disables interrupt
    Enables 8-bit timer/event counter 2 operation
    Transmits start bit
    Waits for start bit transmission time
    ACKE ← $\overline{\text{parity bit data}}$
    SIO0 ← transmit data
    Enables interrupt
```

[Stop bit transmission/reception processing (8-bit timer/event counter 2 interrupt processing)]



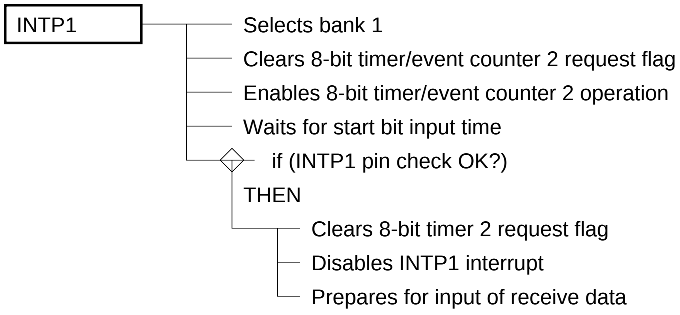
```
TAIMA2
    Selects bank 1
    if (Data transmission in progress)
    THEN
        switch (Which data is transmitted?)
            [case : 1]
                First transmission of end bit
            [case : 2]
                Second transmission of end bit
                Disables 8-bit timer/event counter 2 interrupt
                Disables 8-bit timer/event counter 2 operation
        switch (Which data is received?)
            [case : 1]
                if (First end bit error?)
                THEN
                    Sets end bit error flag
            [case : 2]
                if (Second end bit error?)
                THEN
                    Sets end bit error flag
                Disables 8-bit timer/event counter 2 interrupt
                Disables 8-bit timer/event counter 2 operation
                if (parity data OK?)
                THEN
                    Sets reception completion flag
                ELSE
                    Sets parity error flag
```

[Data transmission/reception completion processing (INTSI0 interrupt processing)]



```
INTSI0
    Selects bank 2
    Clears 8-bit timer/event counter 2 request flag
    Enables 8-bit timer/event counter 2 interrupt
    Outputs "H" to BUSY0 signal
    if (reception in progress)
    THEN
        Inputs receive data
        Inputs receive data in reverse direction
        Inputs parity data
```

[Data reception start processing (INTP1 interrupt processing)]



```
INTP1
    Selects bank 1
    Clears 8-bit timer/event counter 2 request flag
    Enables 8-bit timer/event counter 2 operation
    Waits for start bit input time
    if (INTP1 pin check OK?)
    THEN
        Clears 8-bit timer 2 request flag
        Disables INTP1 interrupt
        Prepares for input of receive data
```

(6) Program list

```
PUBLIC  JUDATA
PUBLIC  SODATA,F_PARITY,S_SOSHIN
PUBLIC  F_DATA,S_JUSHIN,F_PADATA,F_TUSHIN
PUBLIC  F_ERRE,F_ERRP
;
VEINTP1     CSEG    AT 08H
            DW      INTP1
VEINTSI0    CSEG    AT 0EH
            DW      INTSI0
VETIM2      CSEG    AT 18H
            DW      TAIMA2
;
SB0         EQU     P2.5
BUSY_O      EQU     P3.1
BUSY_I      EQU     P3.0
PORT25      EQU     PM2.5
;
MOSRAM      DSEG    SADDR
SODATA:     DS      1                            ; Transmit data storage area
C_WORK:     DS      1                            ; Work area
JUDATA:     DS      1                            ; Receive data storage area
i:          DS      1                            ; Work counter
k:          DS      1                            ; Work counter
;
MOSFLG      BSEG
F_ERRP      DBIT                                 ; Parity error flag
F_ERRE      DBIT                                 ; End bit error flag
F_DATA      DBIT                                 ; Reception completion flag
F_PADATA    DBIT                                 ; Parity data flag
F_PARITY    DBIT                                 ; Parity select flag
F_WORK      DBIT                                 ; Flag work area
F_TUSHIN    DBIT                                 ; Communication flag
;
;************************************************
;               Reception routine
;************************************************
JUSHIN  CSEG                                     ;
S_JUSHIN:                                        ;
    CLR1    PIF1                                 ; Clears request flag
    CLR1    PMK1                                 ; Enables INTP1 interrupt
    RET                                          ;
```

```
;************************************************
;              Transmission routine
;************************************************
SOSHON  CSEG
S_SOSHIN:                                       ;
    A=SODATA                                    ; Reverses direction of transmit data
    SODATA=#0                                   ;
    if_bit(A.7)                                 ;
        SET1    SODATA.0                        ;
    endif                                       ;
    if_bit(A.6)                                 ;
        SET1    SODATA.1                        ;
    endif                                       ;
    if_bit(A.5)                                 ;
        SET1    SODATA.2                        ;
    endif                                       ;
    if_bit(A.4)                                 ;
        SET1    SODATA.3                        ;
    endif                                       ;
    if_bit(A.3)                                 ;
        SET1    SODATA.4                        ;
    endif                                       ;
    if_bit(A.2)                                 ;
        SET1    SODATA.5                        ;
    endif                                       ;
    if_bit(A.1)                                 ;
        SET1    SODATA.6                        ;
    endif                                       ;
    if_bit(A.0)                                 ;
        SET1    SODATA.7                        ;
    endif                                       ;
    CLR1    F_PADATA                            ; Clears parity data flag
    if_bit(F_PARITY)                            ;  Odd parity selected?
        SET1    F_PADATA                        ;     Sets parity data
    endif                                       ;
    A=SODATA                                    ;
    for(k=#0;k<#8;k++)                          ; Determines parity data
        RORC    A.1                             ;
        CY ^= F_PADATA                          ;
        F_PADATA = CY                           ;
    next                                        ;
    TOC1=#01100000B (A)                         ;
    CLR1    TMIF2                               ; Clears timer 2 request flag
    DI                                          ;
    SET1    TCE2                                ; Enables 8-bit timer operation
    SET1    CMDT                                ; Transmits start bit
    while_bit(!TMIF2)                           ; Waits for start bit transmission time
    endw                                        ;
    CLR1    TMIF2                               ;
    SET1    ACKE                                ; Clears acknowledge
    if_bit(F_PADATA)                            ; Clears acknowledge if parity data is 1
        CLR1    ACKE                            ;
    endif                                       ;
    SIO0=SODATA (A)                             ; Starts data transmission
    EI                                          ;
    RET                                         ;
```

```
;************************************************
;          Timer 2 interrupt processing
;************************************************
TIM2    CSEG                                    ;
TAIMA2:                                         ;
    SEL RB1                                     ; Sets bank 1
    if_bit(F_TUSHIN)                            ; Communication in progress?
        if(C_WORK < #3)                         ;    Contents of work mode
            switch(C_WORK)                      ;    0: Transmits end bit
            case 0:                             ;
                SET1   RELT                     ;
                break                           ;
            case 2:                             ;    2: Transmits end bit
                SET1   RELT                     ;      Disables 8-bit timer 2 interrupt
                SET1   TMMK2                    ;
                CLR1   TCE2                     ;      Disables 8-bit timer 2 operation
                SET1   SB0                      ;      Sets port 2.5 in input mode
                CLR1   CSIE0                    ;      Disables serial operation
                SET1   CSIE0                    ;      Enables serial operation
                SET1   RELT                     ;
                CLR1   SB0                      ;      Sets port 2.5 in output mode
                C_WORK=#0                       ;
                break                           ;
            ends                                ;
            C_WORK++                            ;
        else                                    ;
            C_WORK=#0                           ;
        endif                                   ;
```

```
else                                        ;
    if(C_WORK < #4)                         ; Reception in progress?
        SET1    PORT25                      ; Sets port 2.5 in input mode.
        switch(C_WORK)                      ;    Contents of work mode
        case 1:                             ;    1: Sets end bit error flag if end bit is “H”
            if_bit(!SB0)                    ;
                SET1    F_ERRE              ;
            endif                           ;
            break                           ;
        case 3:                             ;    3: Sets end bit error flag if end bit is “H”
            if_bit(!SB0)                    ;
                SET1    F_ERRE              ;
            endif                           ;
            SET1    SB0                     ;    Port 2.5 = H
            CLR1    CSIE0                   ;    Disables serial operation
            SET1    CSIE0                   ;    Enables serial operation
            SET1    RELT                    ;
            CLR1    SB0                     ;    Port 2.5 = L
            C_WORK=#0                       ;
            SET1    TMMK2                   ;    Disables 8-bit timer 2 interrupt
            CLR1    TCE2                    ;    Disables 8-bit timer operation
            CLR1    F_WORK                  ;
            if_bit(F_PARITY)                ;
                SET1    F_WORK              ;
            endif                           ;
            A=JUDATA                        ;
            for(i=#0;i<#8;i++)              ; Adds receive data
                RORC  A,1                   ;
                CY ^= F_WORK                ;
                F_WORK = CY                 ;
            next                            ;
            CLR1    F_ERRP                  ;
            CLR1    F_DATA                  ;
            F_WORK ^= F_PADATA (CY)         ;
            if_bit(!F_WORK)                 ; Checks parity data
                if_bit(!F_ERRE)             ;
                    SET1    F_DATA          ; Sets F_DATA flag if reception is completed normally
                endif                       ;
            else                            ;
                SET1    F_ERRP              ; Sets F_ERRP flag in case of parity error
            endif                           ;
            CLR1    F_WORK                  ;
            break                           ;
        ends                                ;
        CLR1    PORT25                      ; Sets port 2.5 in output mode
        C_WORK++                            ;
    else                                    ;
        C_WORK=#0                           ;
    endif                                   ;
endif                                       ;
RETI                                        ;
```

```
;
;************************************************
;     INTSI0 interrupt processing (reception)
;************************************************
S_SI0       CSEG                                ;
INTSI0:                                         ;
    SEL RB2                                     ;
    CLR1    TMIF2                               ; Clears timer 2 request flag
    CLR1    TMMK2                               ; Enables timer 2 interrupt
    SET1    BUSY_O                              ;
    if_bit(!F_TUSHIN)                           ;
        A=SIO0                                  ;
        JUDATA=#0                               ;
        if_bit(A.7)                             ; Inputs receive data in reverse direction
            SET1    JUDATA.0                    ;
        endif                                   ;
        if_bit(A.6)                             ;
            SET1    JUDATA.1                    ;
        endif                                   ;
        if_bit(A.5)                             ;
            SET1    JUDATA.2                    ;
        endif                                   ;
        if_bit(A.4)                             ;
            SET1    JUDATA.3                    ;
        endif                                   ;
        if_bit(A.3)                             ;
            SET1    JUDATA.4                    ;
        endif                                   ;
        if_bit(A.2)                             ;
            SET1    JUDATA.5                    ;
        endif                                   ;
        if_bit(A.1)                             ;
            SET1    JUDATA.6                    ;
        endif                                   ;
        if_bit(A.0)                             ;
            SET1    JUDATA.7                    ;
        endif                                   ;
        CLR1    F_PADATA                        ; Inputs parity data
        CY=ACKD                                 ;
        NOT1    CY                              ;
        F_PADATA=CY                             ;
    endif                                       ;
    C_WORK=#0                                   ;
    RETI                                        ;
```

```
;************************************************
;     INTP1 interrupt processing (reception)
;************************************************
S_P1        CSEG                                ;
INT1:                                           ;
    SEL RB2                                     ;
    CLR1    TMIF2                               ; Clears timer 2 request flag
    CLR1    TCE2                                ; Clears timer 2 counter
    SET1    TCE2                                ; Enables timer operation
    while_bit(!TMIF2)                           ;
    endw                                        ;
    CLR1    TMIF2                               ;
    SET1    PORT25                              ; Sets port in input mode
    if_bit(!SB0)                                ; Chattering processing of INTP1
        CLR1    ACKE                            ;
        TOC1=#10100000B                         ;
        ST1     PMK1                            ; Disables INTP1 interrupt
        SIO0=#0FFH                              ;
    endif                                       ;
    CLR1    PORT25                              ; Sets port 2.5 in output mode
    RETI                                        ;
    END
```

CHAPTER 9 APPLICATIONS OF A/D CONVERTER

The A/D converter of the 78K/0 series is a successive approximation type with an 8-bit resolution and eight channels. Although only a select mode is supported as the operation mode, conversion can be started by an external trigger. If the external trigger is not used, the analog data of a selected channel is repeatedly converted into a digital signal.

The A/D converter is set by the following registers:

• A/D converter mode register (ADM, ADM0)	
• A/D converter input select register (ADIS)	: μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries, μPD780001
• Analog input channel specification register (ADS0)	: μPD780024, 780024Y, 780924 subseries

Caution The format of the registers provided on the μPD780024, 780024Y, and 780924 subseries differs from the format of the registers used in the program examples in this chapter. When using a program example in this chapter with any of the μPD780024, 780024Y, and 780924 subseries, change the setting of the registers according to the registers of the microcontroller used.

★ **Figure 9-1. Format of A/D Converter Mode Register (μPD78014, 78014Y subseries, μPD780001)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADM	CS	TRG	FR1	FR0	ADM3	ADM2	ADM1	1	FF80H	01H	R/W

ADM3	ADM2	ADM1	Selects analog input channel
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
1	1	0	ANI6
1	1	1	ANI7

FR1	FR0	Selects A/D conversion time[Note 1]			
			At f_X = 10.0 MHz	At f_X = 8.38 MHz	At f_X = 4.19 MHz
0	0	160/f_X	Setting prohibited[Note 2]	19.1 μs	38.1 μs
0	1	80/f_X	Setting prohibited[Note 2]	Setting prohibited[Note 2]	19.1 μs
1	0	200/f_X	20.0 μs	23.9 μs	47.7 μs
1	1	Setting prohibited			

TRG	Selects external trigger
0	No external trigger (software start)
1	Conversion started by external trigger (hardware start)

CS	Controls A/D conversion operation
0	Stops operation
1	Starts operation

Notes 1. Set the A/D conversion time to 19.1 μs or longer.

2. These settings are prohibited because the A/D conversion time is less than 19.1 μs.

Cautions 1. Set bit 0 to 1.

2. To reduce the power consumption of the A/D converter when the standby function is used, stop the A/D conversion operation by clearing bit 7 (CS) to 0, and then execute the HALT or STOP instruction.

Remark f_X : main system clock oscillation frequency

★

Figure 9-2. Format of A/D Converter Mode Register (μPD78018F, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADM	CS	TRG	FR1	FR0	ADM3	ADM2	ADM1	HSC	FF80H	01H	R/W

ADM3	ADM2	ADM1	Selects analog input channel
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
1	1	0	ANI6
1	1	1	ANI7

FR1	FR0	HSC	Selects A/D conversion time[Note 1]				
				At f_X = 10.0 MHz	At fx = 8.38 MHz	At fx = 5.0 MHz	At fx = 4.19 MHz
0	0	1	160/f_X	Setting prohibited[Note 2]	19.1 μs	32.0 μs	38.1 μs
0	1	1	80/f_X	Setting prohibited[Note 2]	Setting prohibited[Note 2]	Setting prohibited[Note 2]	19.1 μs
1	0	0	100/f_X	Setting prohibited[Note 2]	Setting prohibited[Note 2]	20.0 μs	23.9 μs
1	0	1	200/f_X	20.0 μs	23.9 μs	40.0 μs	47.7 μs
Others			Setting prohibited				

TRG	Selects external trigger
0	No external trigger (software start)
1	Conversion started by external trigger (hardware start)

CS	Controls A/D conversion operation
0	Stops operation
1	Starts operation

Notes 1. Set the A/D conversion time to 19.1 μs or longer.

2. These settings are prohibited because the A/D conversion time is less than 19.1 μs.

Cautions 1. To reduce the power consumption of the A/D converter when the standby function is used, stop the A/D conversion operation by clearing bit 7 (CS) to 0, and then execute the HALT or STOP instruction.

2. To resume the A/D conversion operation which has been once stopped, clear the interrupt request flag (ADIF) to 0 and then start the A/D conversion operation.

Remark f_X : main system clock oscillation frequency

★ Figure 9-3. Format of A/D Converter Mode Register (μPD780024, 780024Y subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADM0	ADCS0	TRG0	FR02	FR01	FR00	EGA01	EGA00	0	FF80H	00H	R/W

EGA01	EGA00	Specifies external trigger signal and edge
0	0	No external trigger
0	1	Detects falling edge
1	0	Detects rising edge
1	1	Detects both rising and falling edges

FR02	FR01	FR00	Selects conversion time[Note 1]
0	0	0	144/f_X (17.1 μs)
0	0	1	120/f_X (14.3 μs)
0	1	0	96/f_X (setting prohibited[Note 2])
1	0	0	72/f_X (setting prohibited[Note 2])
1	0	1	60/f_X (setting prohibited[Note 2])
1	1	0	48/f_X (setting prohibited[Note 2])
Others			Setting prohibited

TRG0	Selects software start/hardware start
0	Software start
1	Hardware start

ADCS0	Controls A/D conversion operation
0	Stops operation
1	Enables operation

Notes 1. Set the A/D conversion time to 14 μs or longer.

2. These settings are prohibited because the A/D conversion time is less than 14 μs.

Remarks 1. f_X : main system clock oscillation frequency

2. ( ): f_X = 8.38 MHz

★ **Figure 9-4. Format of A/D Converter Mode Register (μPD780924 subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADM0	ADCS0	TRG0	FR02	FR01	FR00	EGA01	EGA00	0	FF80H	00H	R/W

EGA01	EGA00	Specifies external trigger signal and edge
0	0	No external trigger
0	1	Detects falling edge
1	0	Detects rising edge
1	1	Detects both rising and falling edges

FR02	FR01	FR00	Selects conversion time
0	0	0	144/f_X (17.1 μs)
0	0	1	120/f_X (14.3 μs)
1	0	0	288/f_X (34.4 μs)
1	0	1	240/f_X (28.6 μs)
1	1	0	192/f_X (22.9 μs)
Others			Setting prohibited

TRG0	Selects software start/hardware start
0	Software start
1	Hardware start

ADCS0	Controls A/D conversion operation
0	Stops operation
1	Enables operation

Note Set the A/D conversion time to 14 μs or longer

Remarks 1. f_X : main system clock oscillation frequency

2. ( ): f_X = 8.38 MHz

Figure 9-5. Format of A/D Converter Input Select Register (μPD78014, 78014Y, 78018F, 78018FY, 78014H subseries)

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADIS	0	0	0	0	ADIS3	ADIS2	ADIS1	ADIS0	FF84H	00H	R/W

ADIS3	ADIS2	ADIS1	ADIS0	Selects number of analog input channels
0	0	0	0	No analog input channel (P10-P17)
0	0	0	1	1 channel (ANI0, P11-P17)
0	0	1	0	2 channels (ANI0, ANI1, P12-P17)
0	0	1	1	3 channels (ANI0-ANI2, P13-P17)
0	1	0	0	4 channels (ANI0-ANI3, P14-P17)
0	1	0	1	5 channels (ANI0-ANI4, P15-P17)
0	1	1	0	6 channels (ANI0-ANI5, P16-P17)
0	1	1	1	7 channels (ANI0-ANI6, P17)
1	0	0	0	8 channels (ANI0-ANI7)
Others				Setting prohibited

Cautions 1. Set analog input channels in the following steps:

<1> Set the number of analog input channels by using ADIS.

<2> Select one channel whose data is to be converted, from the channels selected by ADIS, by using the A/D converter mode register (ADM).

2. The internal pull-up resistor is not used to the channel selected by ADIS as an analog input channel, regardless of the value of the bit 1 (PUO1) of the pull-up resistor option register (PUO).

★ **Figure 9-6. Format of A/D Converter Input Select Register (μPD780001)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADIS	0	0	0	0	ADIS3	ADIS2	ADIS1	ADIS0	FF84H	08H	R/W

ADIS3	ADIS2	ADIS1	ADIS0	Selects number of analog input channels
1	0	0	0	8 channels (ANI0-ANI7)
Others				Setting prohibited

Cautions 1. Set analog input channels in the following steps:

<1> Set the number of analog input channels by using ADIS.

<2> Select one channel whose data is to be converted, from the channels selected by ADIS, by using the A/D converter mode register (ADM).

2. The internal pull-up resistor is not used to the channel selected by ADIS as an analog input channel, regardless of the value of the bit 1 (PUO1) of the pull-up resistor option register (PUO).

★ **Figure 9-7. Format of Analog Input Channel Specification Register (μPD780024, 780024Y, 780924 subseries)**

Symbol	7	6	5	4	3	2	1	0	Address	At reset	R/W
ADS0	0	0	0	0	0	ADS02	ADS01	ADS00	FF81H	00H	R/W

ADS02	ADS01	ADS00	Specifies analog input channel
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
1	1	0	ANI6
1	1	1	ANI7

9.1 Level Meter

In this application example, the analog voltage input to the A/D converter is displayed on an LED matrix consisting of 4 × 4, i.e., 16 LEDs. In the example shown in this section, the μPD78014 subseries is used.

Because a level meter has been included in this example, the LED display is given in decibel units. Figure 9-8 shows the circuit of the level meter, and Figure 9-9 shows the relations between the result of the A/D conversion and the number of display digits.

Figure 9-8. Example of Level Meter Circuit

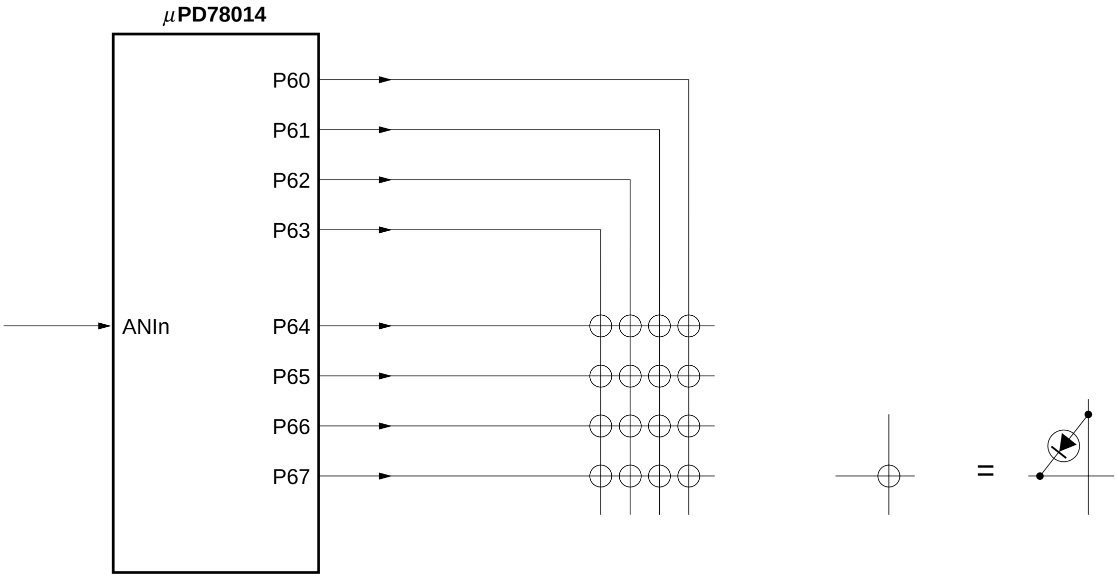
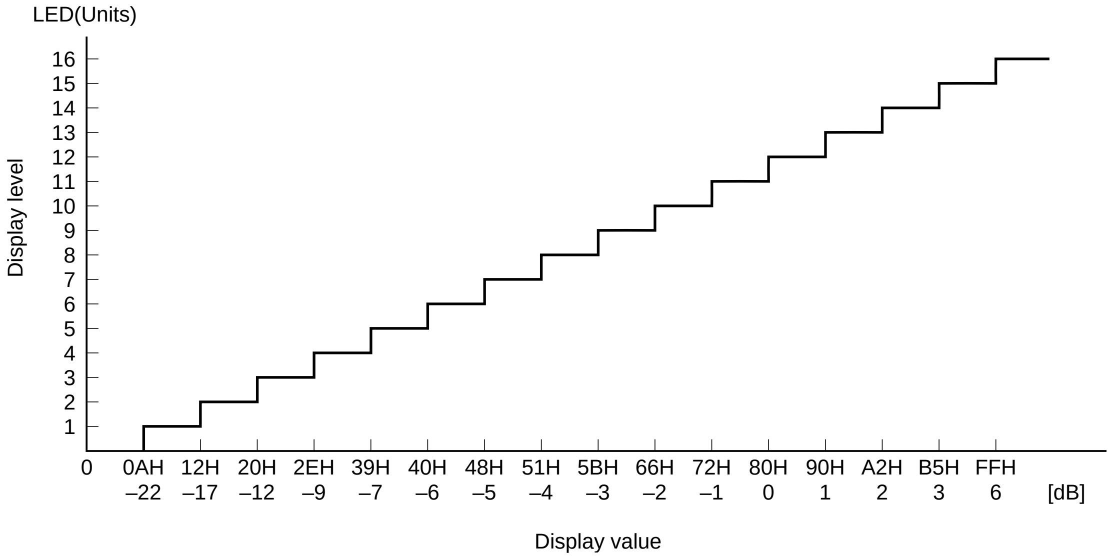


Figure 9-9. A/D Conversion Result and Display



The level meter in this example operates with specifications <1> through <3> below.

<1> Measurement method

A/D conversion is performed every 20 ms, and the average value of four previous data is calculated and displayed on the LEDs.

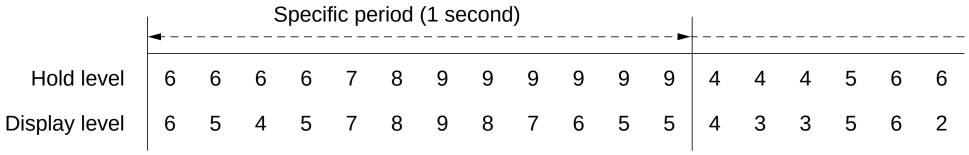
<2> Display method

The LED display is updated every 20 ms. The LED matrix consists of 4 × 4 = 16 LEDs and performs dynamic display. For the dynamic display, 8-bit timer/event counter 1 (interval time: 2 ms) is used.

<3> Peak hold

Holding the maximum display level for a specific period (1 second) is called peak hold. Even if the display level drops during a specific period, only the LED at the maximum display level is held. Therefore, the hold period of the hold level is 20 ms to 1 s.

Figure 9-10. Concept of Peak Hold



Hold level	6	6	6	6	7	8	9	9	9	9	9	9	4	4	4	5	6	6
Display level	6	5	4	5	7	8	9	8	7	6	5	5	4	3	3	5	6	2

(1) Description of package

<Public declaration symbol>

LEVEL	:	Name of LED display subroutine
DSPLEV	:	Display level storage area
HLDLEV	:	Hold level storage area
CT20MS	:	Counter measuring 20 ms
CT1S	:	Counter measuring 1 s

<Register used>

AX, HL, BC (subroutine processing)

Bank 0: A, HL, B (interrupt processing)

<RAM used>

Name	Usage	Attribute	Bytes
ADDAT	Stores A/D conversion value	SADDR	4
DSPLEV	Stores display level		1
HLDLEV	Stores hold level		
CT20MS	Counter measuring 20 ms		
CT1S	Counter measuring 1 s		
DIGCNT	Display digit counter		
DSPDAT	Stores display data		4
WORKCT	Work counter for loop processing		1

<Flag used>

Name	Usage
T20MSF	Set every 20 ms
T1SF	Set every 1 s

<Nesting>

2 levels 5 bytes

<Hardware used>

- A/D converter
- 8-bit timer/event counter 1
- P6

<Initial setting>

• Selects channel of A/D converter and starts operation	ADM = #1000×××1B
• Interval time of 2 ms of 8-bit timer/event counter 1	TCL1 = #10111011B
	TMC1 = #00000001B
	CR10 = 130
• P6 output mode	
• Makes P6 output latch low	
• Enables INTTM1 interrupt	

<Starting>

This program performs two types of processing: A/D conversion (subroutine) and LED display (interrupt).

- A/D conversion processing
  Call LEVEL at least once every 20 ms from the main processing. The LEVEL processing performs A/D conversion processing only when 20 has elapsed.
- LED display
  The 4 × 4 LED matrix performs dynamic display by using the interrupt processing of 8-bit timer/event counter 1 (interval: 2 ms). The interrupt processing of 8-bit timer/event counter 1 sets the T20MSF (loading of A/D conversion value) and T1SF (end of hold period) used for the A/D conversion processing at an interval of 2 ms.

(2) Example of use

```
EXTRN   LEVEL,CT20MS,CT1S

MOV     CT20MS,#10
MOV     CT1S,#50
MOV     TMC2,#00100110B
CLR1    TMMK3

P6=#00H             ; Turns OFF LED display
PM6=#00000000B
ADM=#10000001B      ; ANI0 pin starts operation
TCL1=#10111011B     ; Sets 8-bit timer/event counter 1 to 2 ms

CR10=#130
TMC1=#00000001B
CLR1    TMMK1       ; Enables 8-bit timer/event counter 1 interrupt
EI
```

(3) SPD chart

```
LEVEL
    IF: 20 ms elapses (T20MSF = 1)
    THEN
        Clears T20MSF
        Stores A/D conversion value to memory
        Averages four previous A/D conversion values
        (FOR: WORKCT = #0; WORKCT < #16; WORKCT + +)
            IF: conversion result > display level comparison data
            THEN
                Updates comparison data
            ELSE
                BREAK
        Stores display data to memory
        IF: less then 1 second (T1SF = 0)
        THEN
            IF: hold level < display level
            THEN
                Sets display level to hold level
        ELSE
            Clears T1SF
            Sets display level to hold level
        Converts display level and hold level to segment signal
        Stores digit signal and segment signal to memory in combination

INTTM1
    Selects register bank 0
    Outputs OFF signal to digit and segment
    Outputs memory contents indicated by digit counter
    Increments digit counter
    Decrements 20 ms counter
    IF: 20 ms elapses (CT20MS = 0)
    THEN
        Sets 20 ms counter to 10
        Sets 20 ms elapse status
            Sets T20MSF
        Decrements 1-s counter
        IF: 1 second elapses (CT1S = 0)
        THEN
            Sets 1-s counter to 50
            Sets 1-s elapse status
                Sets T1SF
```

(4) Program list

```
        PUBLIC  LEVEL,HLDLEV,DSPLEV,CT20MS,CT1S

AD_DAT  DSEG    SADDR
ADDAT:  DS      4                              ; A/D conversion result storage area
DSPLEV: DS      1                              ; Display level value
HLDLEV: DS      1                              ; Hold level value
CT20MS: DS      1                              ; 20 ms counter
CT1S:   DS      1                              ; 1 s counter
DIGCNT: DS      1                              ; Display digit counter
DSPDAT: DS      4                              ; Display data
WORKCT: DS      1

AD_FLG  BSEG
T20MSF  DBIT                                   ; Measures 20 ms
T1SF    DBIT                                   ; Measures 1 s

VETM1   CSEG    AT 16H
        DW      INTTM1                         ; Sets vector address of 8-bit timer/event counter 1

AD_SEG  CSEG
;*********************************
*      Sets level meter data
;*********************************
LEVEL:
        IF_BIT(T20MSF)                         ; Checks 20 ms
            CLR1    T20MSF
            A=ADCR                             ; Inputs A/D conversion value
            A<->ADDAT                          ; Stores A/D conversion value
            A<->ADDAT+1
            A<->ADDAT+2
            A<->ADDAT+3
                                               ; Averages four A/D conversion values
            AX=#0H
            HL=#ADDAT                          ; Data storage address
            for(WORKCT=#0;WORKCT<#4;WORKCT++)
                A+=[HL]
                HL++
                if_bit(CY)                     ; Carry
                    X++                        ; Higher digit
                endif
            next

            A<->X
            C=#4                               ; Averages four values
            AX/=C                              ; AX/C = AX (quotient) ... C (remainder)
            if(C>=#2) (A)                      ; Remainder processing (2 or higher is carried)
                X++                            ; Carry processing
            endif

            HL=#LEVTBL
            B=#0                               ; Conversion result storage register
            for(WORKCT=#0;WORKCT<#16;WORKCT++)
                if(X>=[HL+B]) (A)              ; Compares data
                    B++
                else
                    break
                endif
            next
        DSPLEV=B (A)                    ; Determines display data

        if_bit(!T1SF)                   ; 1 s (hold level updated)
            X=HLDLEV (A)                ; Compares hold and display levels
            if(X<DSPLEV) (A)
                HLDLEV=DSPLEV (A)
            endif
        else
            CLR1    T1SF
            HLDLEV=DSPLEV (A)
        endif

        HL=#DSPTBL
        A=DSPLEV                        ; Creates display level
        A+=A
        B=A
        A=HLDLEV
        A+=A
        C=A
        X=[HL+B] (A)
        B++
        A=[HL+B]
        HL=#HLDTBL                      ; Creates hold level
        A<->X
        A|=[HL+C]
        A<->X
        C++
        A|=[HL+C]
        BC=AX

        HL=#DSPDAT                      ; Sets segment signal of first digit
        A=C
        A&=#0FH
        A|=#00010000B                   ; Sets digit signal
        [HL]=A
        HL++
        A=C                             ; Sets segment signal of second digit
        A>>=1
        A>>=1
        A>>=1
        A>>=1
        A&=#0FH
        A|=#00100000B                   ; Sets digit signal
        [HL]=A
        HL++
        A=B                             ; Sets segment signal of third digit
        A&=#0FH
        A|=#01000000B                   ; Sets digit signal
        [HL]=A
        HL++
        A=B                             ; Sets segment signal of fourth digit
        A>>=1
        A>>=1
        A>>=1
        A>>=1
        A&=#0FH                         ; Sets digit signal
        A|=#10000000B
        [HL]=A
    endif
```

```
        RET
LEVTBL:
        DB      0AH
        DB      12H
        DB      20H
        DB      2EH
        DB      39H
        DB      40H
        DB      48H
        DB      51H
        DB      5BH
        DB      66H
        DB      72H
        DB      80H
        DB      90H
        DB      0A2H
        DB      0B5H
        DB      0FFH

DSPTBL:
        DW      0000000000000000B
        DW      0000000000000001B
        DW      0000000000000011B
        DW      0000000000000111B
        DW      0000000000001111B
        DW      0000000000011111B
        DW      0000000000111111B
        DW      0000000001111111B
        DW      0000000011111111B
        DW      0000000111111111B
        DW      0000001111111111B
        DW      0000011111111111B
        DW      0000111111111111B
        DW      0001111111111111B
        DW      0011111111111111B
        DW      0111111111111111B
        DW      1111111111111111B

HLDTBL:
        DW      0000000000000000B
        DW      0000000000000001B
        DW      0000000000000010B
        DW      0000000000000100B
        DW      0000000000001000B
        DW      0000000000010000B
        DW      0000000000100000B
        DW      0000000001000000B
        DW      0000000010000000B
        DW      0000000100000000B
        DW      0000001000000000B
        DW      0000010000000000B
        DW      0000100000000000B
        DW      0001000000000000B
        DW      0010000000000000B
        DW      0100000000000000B
        DW      1000000000000000B
$EJECT
```

```
;********************************
*        Level meter data
;********************************
TM1_SEG CSEG
INTTM1:
        SEL RB0
        P6=#00000000B              ; Turns OFF digit and segment signals
        HL=#DSPDAT
        B=DIGCNT (A)
        P6=[HL+B] (A)
        DIGCNT++
        DIGCNT&=#00000011B
        CT20MS--                   ; 20 ms?
        if(CT20MS==#0)
            CT20MS=#10             ; Sets initial counter value
            SET1    T20MSF
            CT1S--                 ; 1s?
            if(CT1S==#0)
                CT1S=#50           ; Sets initial counter value
                SET1    T1SF
            endif
        endif
        RETI
```

9.2 Thermometer

In this application example, a temperature in a range of –20°C to +50°C is measured by using a thermistor (6 kΩ/ 0°C) as a temperature sensor. Changes in the resistance of the thermistor with respect to temperature are given by the following expression:

$$R = R_0 \exp \{ B (1/T - 1/T_0) \}$$

where,

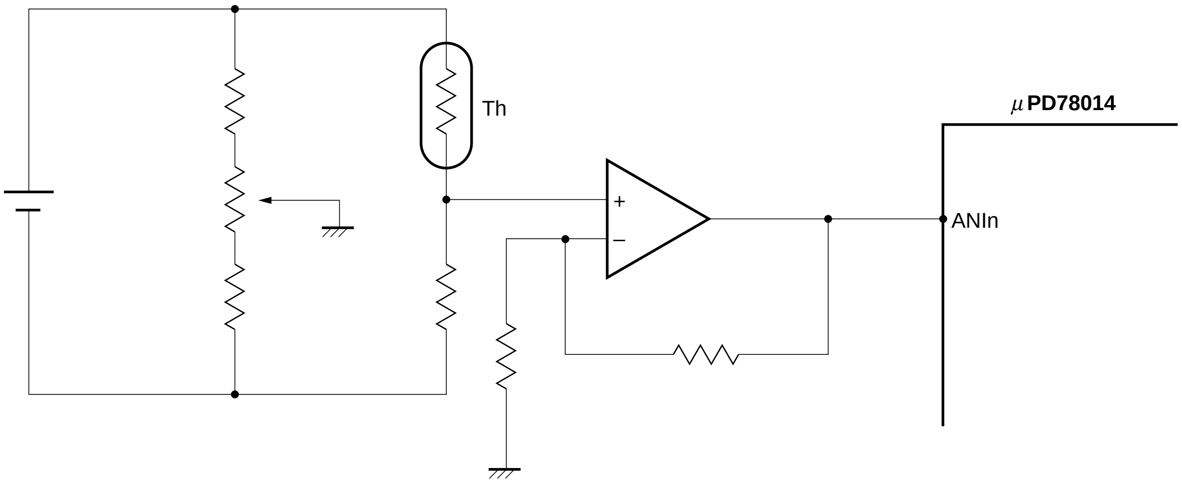
R : resistance at given temperature T [°K]
T : given temperature [°K]
R_0 : resistance at reference temperature T_0 [°K]
T_0 : reference temperature [°K]
B : constant obtained by reference temperature T_0 [°K] and T_0 [°K]

Constant B changes with the temperature. This constant can be calculated by changing the above expression as follows:

$$B = \frac{1}{(1/T - 1/T_0)} \ln \frac{R}{R_0}$$

Figure 9-11 shows a circuit example. This circuit is designed to input 0 V at –20°C, and 5 V at + 50°C.

Figure 9-11. Circuit Example of Thermometer



Because the characteristic of the thermistor is non linear in this example, the input analog voltage is not converted to a temperature in a range of –20 °C to +50 °C through calculation but by comparison with table data. This conversion result is stored to RAM (DSPDAT) as 2-digit BCD. Figure 9-12 shows the characteristics of the thermistor, and Table 9-1 shows the relations between temperature and A/D conversion value.

To measure the temperature, four conversion values are averaged and converted to a temperature. The result of the conversion is stored in a display area. Therefore, the data is updated once every four times. For example, if measurement processing is executed every 250 ms, the display updating cycle is 1 second.

Figure 9-12. Temperature vs. Output Characteristic

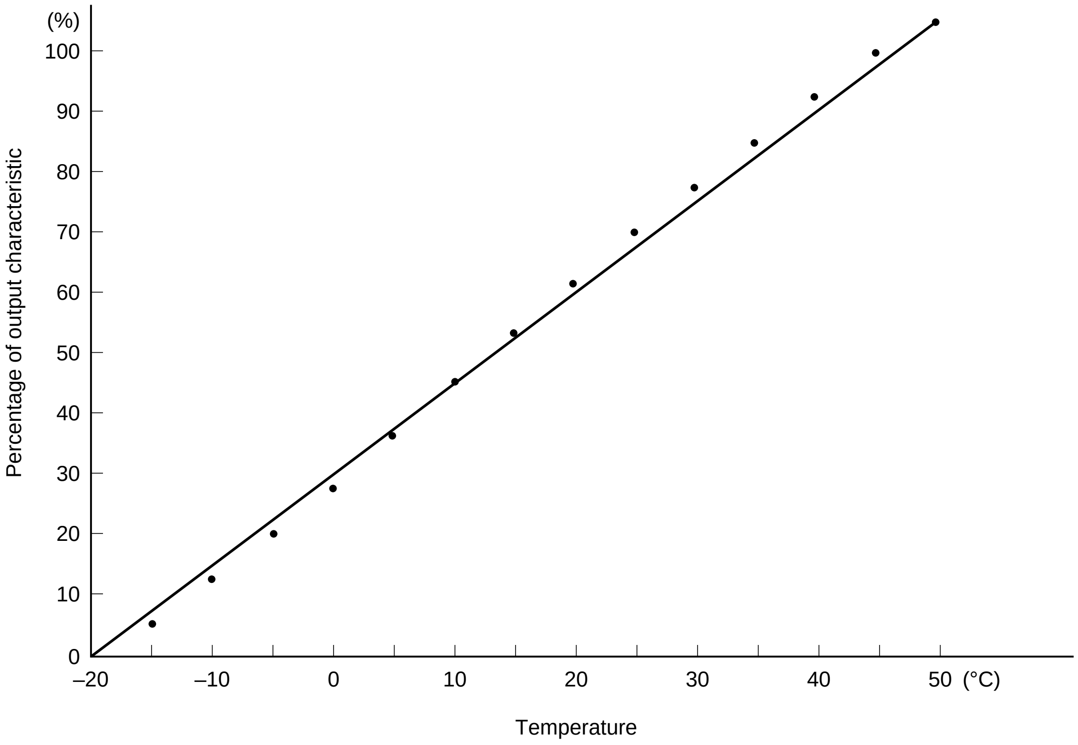


Table 9-1. A/D Conversion Value and Temperature

Conversion Value	Temperature [°C]	Conversion Value	Temperature [°C]	Conversion Value	Temperature [°C]	Conversion Value	Temperature [°C]
00	–20.0	38	–2.5	82	15.5	CB	33.5
01	–19.5	3C	–1.5	86	16.5	CE	34.5
04	–18.5	40	–0.5	8B	17.5	D2	35.5
07	–17.5	44	0.5	8F	18.5	D6	36.5
0A	–16.5	48	1.5	93	19.5	D9	37.5
0C	–15.5	4C	2.5	97	20.5	DC	38.5
0F	–14.5	50	3.5	9B	21.5	E0	39.5
12	–13.5	54	4.5	9F	22.5	E3	40.5
16	–12.5	58	5.5	A3	23.5	E7	41.5
19	–11.5	5C	6.5	A8	24.5	EA	42.5
1C	–10.5	60	7.5	AC	25.5	ED	43.5
1F	–9.5	64	8.5	B0	26.5	F0	44.5
23	–8.5	69	9.5	B4	27.5	F3	45.5
26	–7.5	6D	10.5	B7	28.5	F6	46.5
2A	–6.5	71	11.5	BB	29.5	F9	47.5
2D	–5.5	75	12.5	BF	30.5	FC	48.5
31	–4.5	7A	13.5	C3	31.5	FE	49.5
35	–3.5	7E	14.5	C7	32.5	FF	50.0

(1) Description of package

<Public declaration symbol>

THMETER	:	Thermometer subroutine call name
DSPDAT	:	Display data storage area
CNTPRO	:	Test counter counting number of inputs
MINUSF	:	Minus temperature display flag
T250MSF	:	250-ms setting flag

<Register used>

AX, BC, HL

<RAM used>

Name	Usage	Attribute	Bytes
ADDAT	Stores A/D conversion value	SADDR	4
DSPDAT	Stores display data		2
CNTPRO	Test counter for number of inputs		1
WORKCT	Work counter for loop processing		

<Flag used>

Name	Usage
T250MSF	Executes measurement processing when set
MINUSF	Set when temperature is below zero

<Nesting>

1 level 2 bytes

<Hardware used>

A/D converter

<Initial setting>

Selects A/D converter channel and starts operation ADM = #1000×××1B;

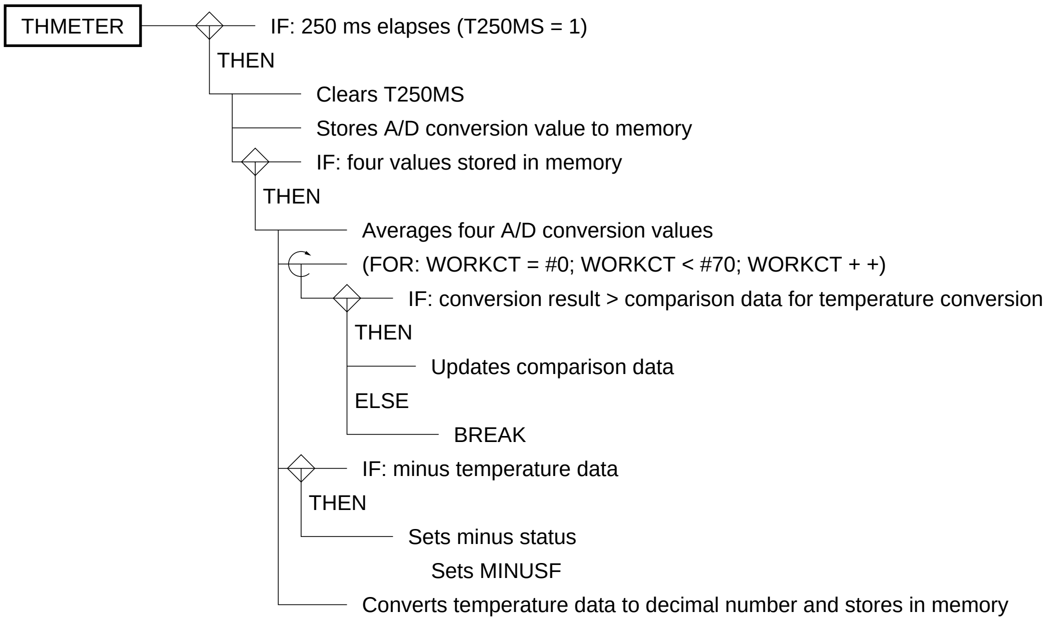
<Starting>

Set the T250MSF flag in each measurement cycle by using timer processing. After that, call THMETER at least once in measurement cycle.

(2) Example of use

```
        EXTRN   THMETER,DSPDAT,CNTPRO
        EXTBIT  MINUSF,T250MSF

AD_DAT  DSEG    SADDR
CT250MS:DS      1                               ; 250 ms counter
LEDD:   DS      4                               ; LED display area
DIGCT:  DS      1                               ; LED display digit counter

VETM3   CSEG    AT 12H
        DW      INTTM3                          ; Sets vector address of watch timer

        MOV     TMC2,#00100110B                 ; Sets watch timer to 1.95 ms
        CLR1    TMMK3
          .
          .
          .
        CT250MS=#128
        CNTPR0=#4
        ADM=#10000011B                          ; Selects ANI1 pin and starts operation
          .
          .
          .

;**********************************************
;       Watch timer interrupt processing
;       Interval time:  1.95 ms
;**********************************************
INTTM3:                                         ; 1.95 ms interrupt processing
          .
          .
          .
        DBNZ    CT250MS,$RTNTM3
        MOV     CT250MS,#128                    ; 250 ms elapses
        SET1    T250MSF
RTNTM3:
          .
          .
          .
        RETI
```

(3) SPD chart



```
THMETER
  IF: 250 ms elapses (T250MS = 1)
  THEN
    Clears T250MS
    Stores A/D conversion value to memory
    IF: four values stored in memory
    THEN
      Averages four A/D conversion values
      (FOR: WORKCT = #0; WORKCT < #70; WORKCT + +)
        IF: conversion result > comparison data for temperature conversion
        THEN
          Updates comparison data
        ELSE
          BREAK
      IF: minus temperature data
      THEN
        Sets minus status
          Sets MINUSF
      Converts temperature data to decimal number and stores in memory
```

(4) Program list

```
        PUBLIC  THMETER,DSPDAT,CNTPR0,T250MSF,MINUSF

AD_DAT  DSEG    SADDR
ADDAT:  DS      4                                ; A/D conversion result storage area
DSPDAT: DS      2                                ; Display data
CNTPR0: DS      1                                ; Tests number of inputs
WORKCT: DS      1

AD_FLG  BSEG
T250MSF DBIT                                     ; Sets 250 ms
MINUSF  DBIT                                     ; Sets minus data

TH_SEG  CSEG
;*********************************
*      Sets temperature data
;*********************************
THMETER:
        if_bit(T250MSF)                          ; 250 ms
            CLR1    T250MSF
            A=ADCR
            A<->ADDAT
            A<->ADDAT+1
            A<->ADDAT+2
            A<->ADDAT+3

            CNTPR0--
            if(CNTPR0==#0)
                CNTPR0=#4
                AX=#0H
                HL=#ADDAT                        ; Data storage address
                for(WORKCT=#0;WORKCT<#4;WORKCT++)
                    A+=[HL]
                    HL++
                    if_bit(CY)                   ; Carry occurs
                        X++                      ; Carry
                    endif
                next

                A<->X
                C=#4
                AX/=C                            ;  AX/C = AX (quotient) ... C (remainder)
                if(C>=#2) (A)                    ; Remainder processing (2 digits or more carried)
                    X++                          ; Carry processing
                endif

                A=X                              ; Converts to temperature data
                B=#0
                HL=#THRTBL
                if(A==#0FFH)
                    B=#70
                else
                    for(WORKCT=#0;WORKCT<#70;WORKCT++)
                        if(X>=[HL+B]) (A)
                            B++
                        else
                            break
                        endif
                    next
        endif

        CLR1    MINUSF
        A=#20                     ; Temperature data 20
        B–=A
        if_bit(CY)                ; To decimal conversion
            SET1    MINUSF
            A=#0
            A–=B                  ; Absolute value of data
            A<->B
        endif
        X=#0                      ; Decimal conversion
        A=B
        A<->X
        C=#10
        AX/=C                     ; Temperature data/10
        DSPDAT=C (A)              ; Updates display data
        (DSPDAT+1)=X (A)
    endif
endif
RET
```

```
THRTBL;
;
        DB      1                       ; –19.5
        DB      4                       ; –18.5
        DB      7                       ; –17.5
        DB      0AH                     ; –16.5
        DB      0CH                     ; –15.5
        DB      0FH                     ; –14.5
        DB      12H                     ; –13.5
        DB      16H                     ; –12.5
        DB      19H                     ; –11.5
        DB      1CH                     ; –10.5
        DB      1FH                     ; –9.5
        DB      23H                     ; –8.5
        DB      26H                     ; –7.5
        DB      2AH                     ; –6.5
        DB      2DH                     ; –5.5
        DB      31H                     ; –4.5
        DB      35H                     ; –3.5
        DB      38H                     ; –2.5
        DB      3CH                     ; –1.5
        DB      40H                     ; –0.5
        DB      44H                     ; +0.5
        DB      48H                     ; 1.5
        DB      4CH                     ; 2.5
        DB      50H                     ; 3.5
        DB      54H                     ; 4.5
        DB      58H                     ; 5.5
        DB      5CH                     ; 6.5
        DB      60H                     ; 7.5
        DB      64H                     ; 8.5
        DB      69H                     ; 9.5
        DB      6DH                     ; 10.5
        DB      71H                     ; 11.5
        DB      75H                     ; 12.5
        DB      7AH                     ; 13.5
        DB      7EH                     ; 14.5
        DB      82H                     ; 15.5
        DB      86H                     ; 16.5
        DB      8BH                     ; 17.5
        DB      8FH                     ; 18.5
        DB      93H                     ; 19.5
        DB      97H                     ; 20.5
        DB      9BH                     ; 21.5
        DB      9FH                     ; 22.5
        DB      0A3H                    ; 23.5
        DB      0A8H                    ; 24.5
        DB      0ACH                    ; 25.5
        DB      0B0H                    ; 26.5
        DB      0B4H                    ; 27.5
        DB      0B7H                    ; 28.5
        DB      0BBH                    ; 29.5
        DB      0BFH                    ; 30.5
        DB      0C3H                    ; 31.5
        DB      0C7H                    ; 32.5
        DB      0CBH                    ; 33.5
        DB      0CEH                    ; 34.5
        DB      0D2H                    ; 35.5
        DB      0D6H                    ; 36.5
        DB      0D9H                    ; 37.5
        DB      0DCH                    ; 38.5
        DB      0E0H                    ; 39.5
        DB      0E3H                    ; 40.5
        DB      0E7H                    ; 41.5
        DB      0EAH                    ; 42.5
        DB      0EDH                    ; 43.5
        DB      0F0H                    ; 44.5
        DB      0F3H                    ; 45.5
        DB      0F6H                    ; 46.5
        DB      0F9H                    ; 47.5
        DB      0FCH                    ; 48.5
        DB      0FEH                    ; 49.5
```

9.3 Analog Key Input

In this example, sixteen keys are input by using the A/D converter. To input keys, a circuit must be designed so that a voltage peculiar to a key is input to the A/D converter when the key is pressed. In the example shown in this section, the μPD78014 subseries is used.

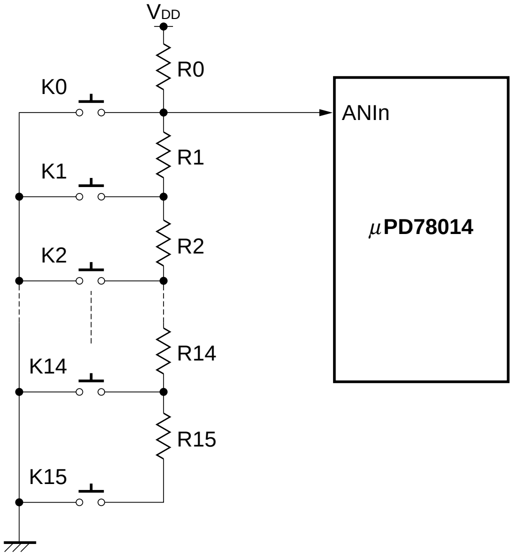
Because sixteen keys are input in this example, V_{DD} is divided by 16 and the voltage of each key is converted into a key code. Table 9-2 shows the relations between the input voltages and key codes (00H through 0FH). When no key input is made, the key code is 10H.

Table 9-2. Input Voltage and Key Code

Input Voltage V	A/D Conversion Value	Key Code
GND	00-07H	00H
$1/16V_{DD}$	08-17H	01H
$2/16V_{DD}$	18-27H	02H
$3/16V_{DD}$	28-37H	03H
$4/16V_{DD}$	38-47H	04H
$5/16V_{DD}$	48-57H	05H
$6/16V_{DD}$	58-67H	06H
$7/16V_{DD}$	68-77H	07H
$8/16V_{DD}$	78-87H	08H
$9/16V_{DD}$	88-97H	09H
$10/16V_{DD}$	98-A7H	0AH
$11/16V_{DD}$	A8-B7H	0BH
$12/16V_{DD}$	B8-C7H	0CH
$13/16V_{DD}$	C8-D7H	0DH
$14/16V_{DD}$	D8-E7H	0EH
$15/16V_{DD}$	E8-F7H	0FH
V_{DD}	F8-FFH	10H

Figure 9-13 shows an example of the circuit that satisfies the above relations between the input voltages and key codes. Note, however, that this circuit gives a priority to the key with the lower number if two or more keys are pressed at the same time.

Figure 9-13. Example of Analog Key Input Circuit



Resistances R0 through R15 used in the circuit in Figure 9-13 can be calculated by the following expression:

$$\sum_{K=1}^{n} R_K = \frac{n \times R0}{16 - n}$$

Table 9-3 shows the resistances of R1 through R15 where R0 is 1 kΩ in the above expression (the calculation result of a resistance may slightly different from the resistance of commercial resistors indicated by a color code).

Table 9-3. Resistances of R1 through R5

Resistor No.	Resistance Value Ω	Resistor No.	Resistance Value Ω	Resistor No.	Resistance Value Ω
R1	68	R6	150	R11	560
R2	75	R7	180	R12	750
R3	82	R8	220	R13	1.3 k
R4	100	R9	270	R14	2.7 k
R5	120	R10	390	R15	8.2 k

This program converts an input analog voltage into the corresponding key code shown in Table 9-2, absorbs chattering, and then stores the input voltage to RAM. To absorb chattering, a key code is assumed to be valid when it coincides with a given value five times in succession. For example, if an analog voltage is sampled every 5 ms, chattering of 20 to 25 ms is absorbed. If a key input is changed, a key change flag (KEYCHG) is set.

(1) Description of package

<Public declaration symbol>

AKEYIN : Analog key input subroutine name
KEYDAT : Key code storage area
PASTDT : Key code storage area for chattering absorption
CHATCT : Chattering absorption counter
KEYCHG : Key change test flag
CHTENDF : Flag to test end of chattering absorption
KEYOFF : Key code when there is no key input

<Register used>

A

<RAM used>

Name	Usage	Attribute	Bytes
PASTDAT	Stores key code for chattering absorption	SADDR	1
KEYDAT	Stores key code		
CHATCNT	Chattering counter		

<Flag used>

Name	Usage
KEYCHG	Set when key is changed
CHTENDF	Sets when chattering absorption ends

<Nesting>

1 level 2 bytes

<Hardware used>

A/D converter

<Initial setting>

Selects A/D converter channel and starts operation ADM = #1000×××1B

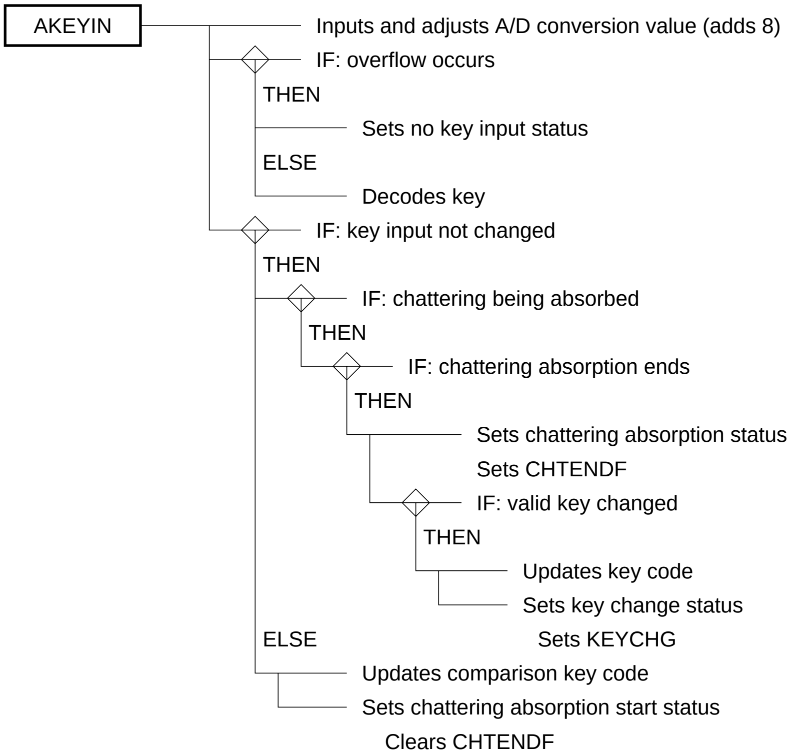
<Starting>

- Call AKEYIN at fixed interval.
- Input a key code after testing the key change flag. Note that this flag is not cleared by the subroutine and must be cleared after the flag has been tested.

(2) Example of use

```
        EXTRN   AKEYIN,KEYDAT,PASTDT,CHATCT
        EXTRN   KEYOFF

        EXTBIT  KEYCHG,CHTENDF

VETM3   CSEG    AT 12H
        DW      INTTM3                    ; Sets vector address of watch timer

MAINDAT DSEG    SADDR
CT5MS:  DS      1

        TMC2=#00100110B
        CLR1    TMMK3
        CT5MS=#3

        KEYDAT=#KEYOFF                    ; Sets OFF data as key data
        PASTDT=#KEYOFF
        CHATCT=#CHAVAL                    ; Sets number of times of chattering to five
        CLR1    CHTENDF
        CLR1    KEYCHG
        ADM=#10000101B                    ; Selects ANI2 pin and starts operation
        EI
          :
          :
        if_bit(KEYCHG)                    ; Key changed?
                CLR1   KEYCHG
                ; Key input processing
        endif
          :
          :
;**********************************************
;       Watch timer interrupt processing
;              Interval:  1.95 ms
;**********************************************
INTTM3:                                   ; 1.95 ms interrupt processing
          :
          :
        DBNZ    CT5MS,$RTNTM3
        MOV     CT5MS,#3                  ; 1.95 ms × 3 elapses
        CALL    !AKEYIN
RTNTM3:
          :
          :
        RETI
```

(3) SPD chart


AKEYIN
Inputs and adjusts A/D conversion value (adds 8)
IF: overflow occurs
THEN
Sets no key input status
ELSE
Decodes key
IF: key input not changed
THEN
IF: chattering being absorbed
THEN
IF: chattering absorption ends
THEN
Sets chattering absorption status
Sets CHTENDF
IF: valid key changed
THEN
Updates key code
Sets key change status
Sets KEYCHG
ELSE
Updates comparison key code
Sets chattering absorption start status
Clears CHTENDF

(4) Program list

```
        PUBLIC  AKEYIN,KEYDAT,PASTDT
        PUBLIC  CHATCT,KEYOFF
        PUBLIC  KEYCHG,CHTENDF
AK_DAT  DSEG    SADDR
KEYDAT: DS      1                       ; Key data storage area
PASTDT: DS      1                       ; Chattering key data
CHATCT: DS      1                       ; Chattering counter

AK_FLG  BSEG
KEYCHG  DBIT                            ; Key changed
CHTENDF DBIT                            ; Chattering absorption end status

KEYOFF  EQU     10H                     ; OFF key data
CHAVAL  EQU     5                       ; Number of times of chattering absorption

AK_SEG  CSEG
;******************************
*       Analog key input
;******************************
AKEYIN:
        A=ADCR                          ; Inputs A/D conversion value
        A+=#8                           ; Corrects data
        if_bit(CY)
            A=#KEYOFF                   ; Sets no key input status
        else
            A>>=1                       ; Decodes key
            A>>=1
            A>>=1
            A>>=1
            A&=0FH
        endif
        if(A==PASTDT)                   ; No key change
            if_bit(!CHTENDF)            ; Chattering being absorbed
                CHATCT--                ; End of chattering absorption
                if(CHATCT==#0)
                    SET1    CHTENDF     ; Sets chattering absorption status
                    A=PASTDT
                    if(A!=KEYDAT)       ; Valid key changed
                        KEYDAT=A        ; Updates key data
                        SET1    KEYCHG  ; Sets key change status
                    endif
                endif
            endif
        endif
            PASTDT=A                    ; Updates previous key data
            CHATCT=#CHAVAL-1            ; Starts chattering absorption
            CLR1    CHTENDF
        endif
        RET
```

9.4 4-Channel Input A/D Conversion

This section describes the method to scan four channels for A/D conversion. The A/D conversion operation is started by the software.

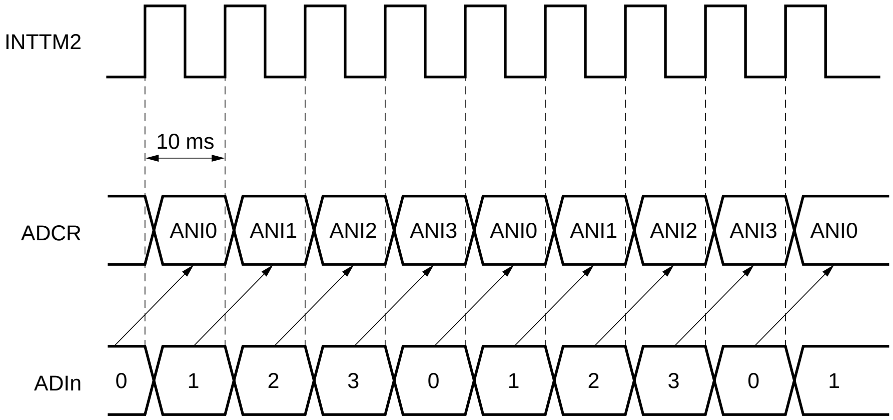
The analog voltages input to the selected four channels are converted into digital signals. The result of the A/D conversion of each channel is stored in RAM.

An interrupt request is generated by using 8-bit timer/event counter 1. The result of the conversion is loaded and channel is converted in the processing of this interrupt request. Because 8-bit timer/event counter 1 is set to 10 ms, it is not necessary to measure the wait time of the A/D conversion.

Caution To change the interrupt time, make the following setting:

- **Set timer longer than { A/D conversion end time + Interrupt entry return time + Interrupt processing time. }**
- **Test flags that indicate the end of the conversion.**

Figure 9-14. Timing Chart in 4-Channel Scan Mode



(1) Description of package

<Public declaration symbol>

- Output parameter

M_CH0	:	Stores conversion result of channel 0
M_CH1	:	Stores conversion result of channel 1
M_CH2	:	Stores conversion result of channel 2
M_CH3	:	Stores conversion result of channel 3

<Register used>

A

<RAM used>

Name	Usage	Attribute	Bytes
M_CH0	Channel 0 conversion result storage area	SADDR	1
M_CH1	Channel 1 conversion result storage area	SADDR	1
M_CH2	Channel 2 conversion result storage area	SADDR	1
M_CH3	Channel 3 conversion result storage area	SADDR	1
M_MODE	Mode storage area	SADDR	1

<Nesting>

1 level 3 bytes

<Hardware used>

- A/D converter
- 8-bit timer/event counter 1
- Port 1 (P10-P13)

<Initial setting>

• Selects A/D converter channel and starts operation	ADM = #1000××××B
• Selects number of A/D converter channels	ADIS = #00000100B

• Interval time of 8-bit timer/event counter 1: 10 ms	TCL1 = #00001110B
	TMC1 = #00000001B
	CR10 = #81

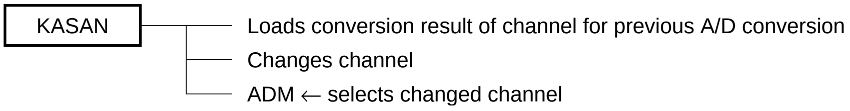
• Enables TMMK1 interrupt	

(2) Example of use

```
EXTRN   M_CH0,M_CH1,M_CH2,M_CH3,M_MODE
;*******************************************
;           Initialize
;*******************************************
M4          CSEG                           ;
RES_STA:
    SEL RB0                                ;
    DI                                     ;
          :
          :
          :
    ADM=#10000001B                         ; Starts A/D operation and selects external trigger channel 0
    ADIS=#00000100B                        ; Selects analog input channel 4
    CR10=#81                               ; Sets modulo register 81
    TCL1=#00001110B                        ; Count clock: 8.2 kHz
    TMC1=#00000001B                        ; Enables 8-bit timer/register 1 operation
    CLR1    TMIF1                          ; Clears timer 1 interrupt request flag
    CLR1    TMMK1                          ; Enables timer 1 interrupt
    EI                                     ;
    M_MODE=#0                              ; Sets initial value (0 channel) to mode area
          :
    while(forever)                         ;
          :
        A=M_CH0                            ; A ← data of channel 0
          :
        A=M_CH1                            ; A ← data of channel 1
          :
        A=M_CH2                            ; A ← data of channel 2
          :
        A=M_CH3                            ; A ← data of channel 3
          :
```

(3) SPD chart

[A/D conversion processing]



KASAN

Loads conversion result of channel for previous A/D conversion

Changes channel

ADM ← selects changed channel

(4) Program list

```
;********************************************
;              A/D conversion
;********************************************
;
$PC(054)                                    ;
;
PUBLIC  M_CH0,M_CH1,M_CH2,M_CH3,M_MODE      ;
;
VEINTM1 CSEG    AT 16H
        DW  KASAN
;********************************************
;              RAM definition
;********************************************
            DSEG    SADDR
M_CH0:      DS      1                       ; Area for channel 0 addition
M_CH1:      DS      1                       ; Area for channel 1 addition
M_CH2:      DS      1                       ; Area for channel 2 addition
M_CH3:      DS      1                       ; Area for channel 3 addition
M_MODE:     DS      1                       ; Mode storage area
;
        CSEG                                ;
KASAN:
        SEL RB2                             ; Selects bank 2
        switch(M_MODE)                      ; Channel currently selected?
        case 0:                             ;   Channel 0:
            M_CH0=ADCR (A)                  ;           Transfers conversion result to RAM
            M_MODE++                        ;
            ADM=#10000011B                  ;           Select channel 1:
            break                           ;
        case 1:                             ;   Channel 1:
            M_CH1=ADCR (A)                  ;           Transfers conversion result to RAM
            M_MODE++                        ;
            ADM=#10000101B                  ;           Selects channel 2
            break                           ;
        case 2:                             ;   Channel 2:
            M_CH2=ADCR (A)                  ;           Transfers conversion result to RAM
            M_MODE++                        ;
            ADM=#10000111B                  ;           Selects channel 3
            break                           ;
        case 3:                             ;   Channel 3:
            M_CH3=ADCR (A)                  ;           Transfers conversion result to RAM
            M_MODE=#0                       ;
            ADM=#10000001B                  ;           Selects channel 0
            break                           ;
        ends                                ;
        RETI                                ;
        END
```

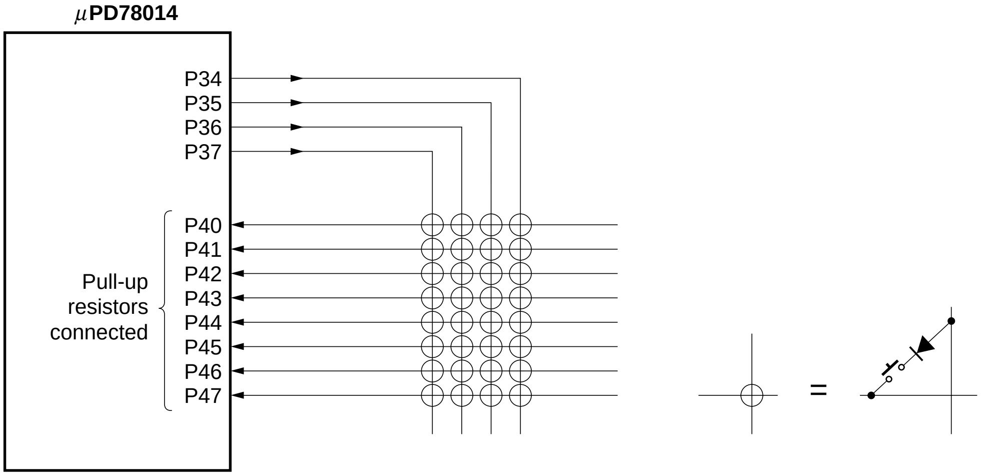
CHAPTER 10 APPLICATIONS OF KEY INPUT

This chapter introduces an example of a program that inputs signals from a key matrix of 4 × 8 keys. The key scan be pressed successively, and two or more keys can be pressed simultaneously. In the circuit shown in this section, the high-order 4 bits of port 3 (P34 through P37) are used as key scan signals, and port 4 is used as key return signals. As the pull-up resistor of port 4 for key return, the internal pull-up resistor set by software is used (refer to **Figure 10-1**).

Port 4 of the 78K/0 series has a function to detect the falling edges of the eight port pins in parallel. If port 4 is used for key return signals, therefore, the standby mode can be released through detection of a falling edge, i.e., by key input.

In this example, the μPD78014 subseries is used.

Figure 10-1. Key Matrix Circuit



The input keys are stored to RAM on a one key-to-1 bit basis. The RAM bit corresponding to a pressed key is set and the bit corresponding to a released key is cleared. By testing the RAM data on a 1-bit-by-1-bit basis starting from the first bit, the key status can be checked. To absorb chattering, the key is assumed to be valid when four successive key codes coincide with a given code. For example, if a key code is sampled every 5 ms, chattering of 15 ms to 20 ms can be absorbed. If the key input is changed, a key change flag (KEYCHG) is set.

(1) Description of package

<Public declaration symbol>

KEYIN	:	Key input subroutine name
KEYDATA	:	Key data storage area
CHATCT	:	Chattering counter
KEYCHG	:	Key change test flag

<Register used>

AX, DE, HL

<RAM used>

Name	Usage	Attribute	Bytes
KEYDATA	Stores valid key data	SADDR	4
WORK	Stores key data during chattering		
CHATCT	Chattering counter		1
WORKCT	Loop processing work counter		

<Flag used>

Name	Usage
CHGFG	Set if key input changes
KEYCHG	Set if valid key changes
CHTEND	Confirms end of chattering

<Nesting>

1 level 2 bytes

<Hardware used>

- P4
- P3 (P34-P37)

<Initial setting>

• Connects pull-up resistor to P4	PUO4 = 1
• Sets high-order 4 bits of P3 in output mode	PM3 = #0000××××B

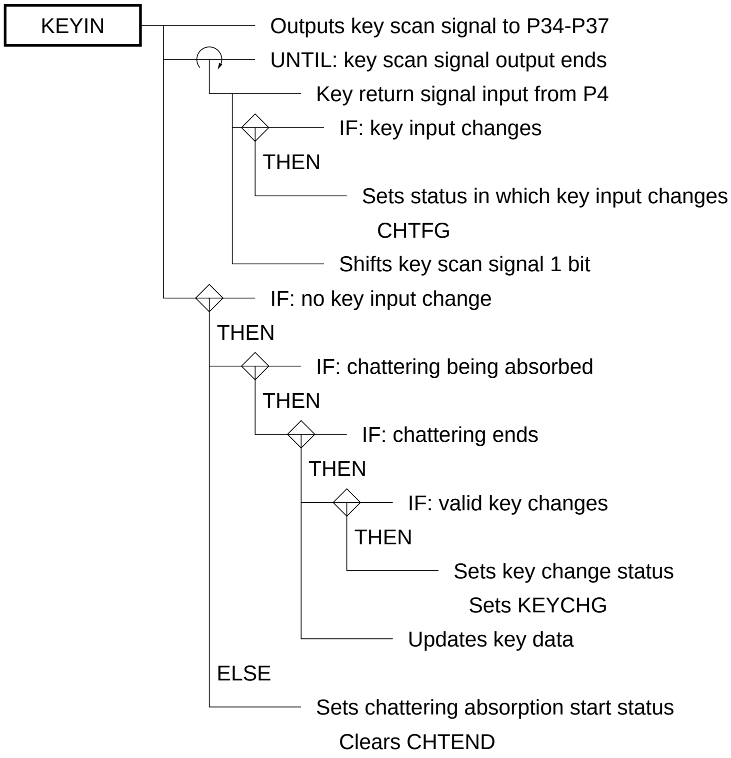
<Starting>

- Call KEYIN at specific intervals.
- Before inputting the key data, test the key change flag. The key change flag is not cleared by the subroutine. Clear the flag after it has been tested.

(2) Example of use

```
        EXTRN   AKEYIN,KEYDAT,PASTDT,CHATCT
        EXTRN   KEYOFF

        EXTBIT  KEYCHG,CHTENDF

VETM3   CSEG    AT 12H
        DW      INTTM3                          ; Sets vector address of watch timer

MAINDAT DSEG    SADDR
CT5MS:  DS      1
          :
          :
        TMC2=#00100110B
        CLR1    TMMK3
        CT5MS=#3

        KEYDAT=#KEYOFF                          ; Sets OFF data as key data
        PASTDT=#KEYOFF
        CHATCT=#CHAVAL                          ; Sets number of times of chattering to five
        CLR1    CHTENDF
        CLR1    KEYCHG
        ADM=#10000101B                          ; Selects ANI2 pin and starts operation
        EI
          :
          :
        if_bit(KEYCHG)                          ; Key changed?
                CLR1    KEYCHG
                ; Key input processing
        endif
          :
          :
;**********************************************
;       Watch timer interrupt processing
;       Interval time:  1.95 ms
;**********************************************
INTTM3:                                         ; 1.95 ms interrupt
          :
          :
        DBNZ    CT5MS,$RTNTM3
        MOV     CT5MS,#3                        ; 1.95 ms × 3 elapses
        CALL    !ANKEYIN
RTNTM3:
          :
          :
        RETI
```

(3) SPD chart



```
KEYIN
    Outputs key scan signal to P34-P37
    UNTIL: key scan signal output ends
        Key return signal input from P4
        IF: key input changes
        THEN
            Sets status in which key input changes
              CHTFG
        Shifts key scan signal 1 bit
    IF: no key input change
    THEN
        IF: chattering being absorbed
        THEN
            IF: chattering ends
            THEN
                IF: valid key changes
                THEN
                    Sets key change status
                      Sets KEYCHG
                Updates key data
    ELSE
        Sets chattering absorption start status
          Clears CHTEND
```

(4) Program list

```
        PUBLIC  KEYDATA,KEYCHG,KEYIN,CHATCT

KEY_DAT DSEG    SADDR
KEYDATA:DS      4                                     ; Key data storage area
WORK:   DS      4                                     ; Chattering key data
CHATCT: DS      1                                     ; Chattering counter
WORKCT: DS      1

KEY_FLG BSEG
CHGFG   DBIT                                          ; Key change status
KEYCHG  DBIT                                          ; Key changed
CHTEND  DBIT                                          ; Chattering absorption end status

KEY_SEG CSEG
;*******************************
*       Matrix key input
;*******************************
KEYIN:
        CLR1    CHGFG
        P3&=#00001111B
        P3|=#00010000B
        HL=#WORK                                      ; Sets address of key work area
        repeat
            A=P4
            A^=#11111111B                             ; Data inverted
            if(A!=[HL])                               ; Key changed?
                SET1    CHGFG
                [HL]=A
            endif
            HL++
            A=P3                                      ; Shifts key scan 1 bit
            A&=#11110000B
            X=A
            A=P3
            A+=X
            P3=A
        until_bit(CY)

        if_bit(!CHGFG)                                ; Key changed
            if_bit(!CHTEND)                           ; Chattering absorbed
                CHATCT--                              ; Chattering ends
                if(CHATCT==#0)
                    SET1    CHTEND
                    DE=#WORK
                    HL=#KEYDATA
                    for(WORKCT=#0;WORKCT<#4;WORKCT++)
                        if([DE]!=[HL]) (A)            ; Key changed
                            SET1    KEYCHG
                        endif
                        A<->[HL]                      ; Transfers WORK to KEYDATA
                        HL++
                        DE++
                    next
                endif
            endif
        else
            CHATCT=#3
            CLR1    CHTEND
        endif
        RET
```

[MEMO]

APPENDIX A DESCRIPTION OF SPD CHART

SPD stands for Structured Programming Diagrams.

Structuring means structuring the logical processing of a program, and designing and formulating the logic by using the basic structure of the logic elements.

All programs can be created by only combining the basic structure of logic elements, (sequentially, selectively, or repeatedly). (This is called a structured theorem). Through structuring, the flow of a program is clarified, and the reliability is improved. Although various methods are available for expressing the structuring of a program, NEC employs a diagram technique called SPD.

The following table describes the SPD symbols used for the SPD technique and compares them with flowchart symbols.

Table A-1. Comparison between SPD Symbols and Flowchart Symbol (1/2)

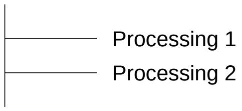
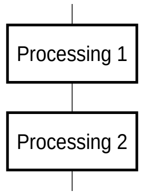
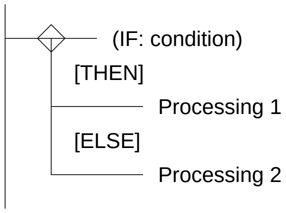
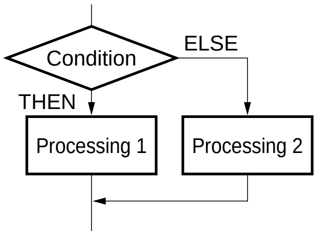
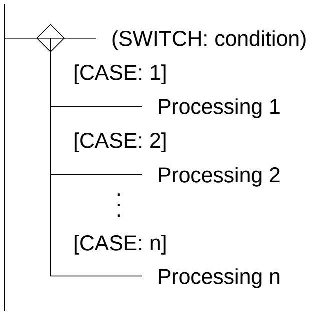
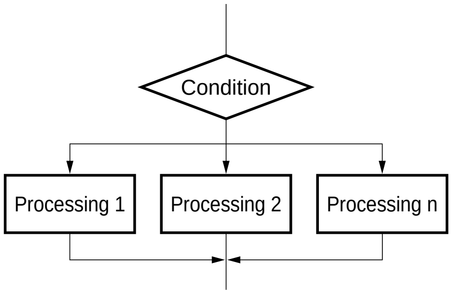
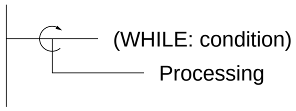
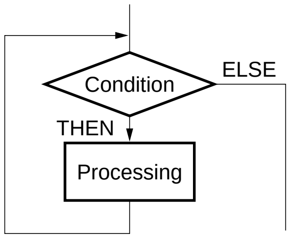
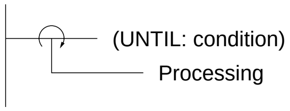
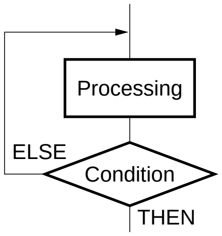
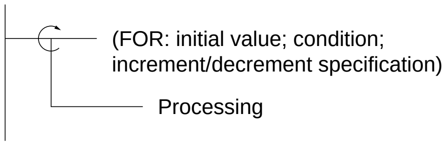
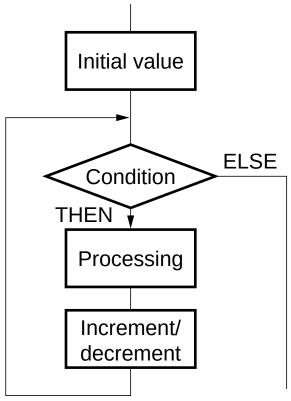
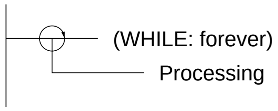
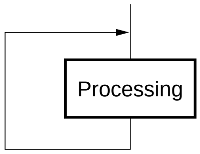
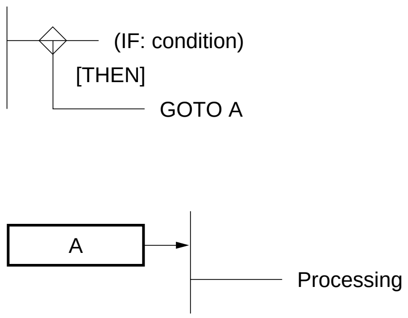
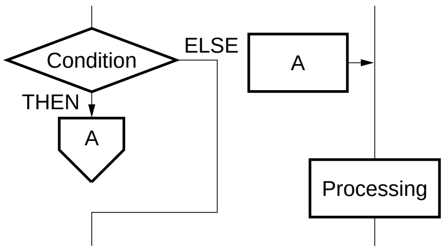
Processing Name	SPD Symbol	Flowchart Symbol
Sequential processing		
Conditional branch (IF)	 (IF: condition) [THEN] Processing 1 [ELSE] Processing 2	
Conditional branch (SWITCH)	 (SWITCH: condition) [CASE: 1] Processing 1 [CASE: 2] Processing 2 [CASE: n] Processing n	

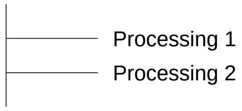
Table A-1. Comparison between SPD Symbols and Flowchart Symbol (2/2)

Processing Name	SPD Symbol	Flowchart Symbol
Conditional loop (WHILE)		
Conditional loop (UNTIL)		
Conditional loop (FOR)	 (FOR: initial value; condition; increment/decrement specification) Processing	 Initial value Condition ELSE THEN Processing Increment/ decrement
Infinite loop		
Connector		

1. Sequential processing

Sequential processing executes processing from top to bottom in the sequence in which processing appears.

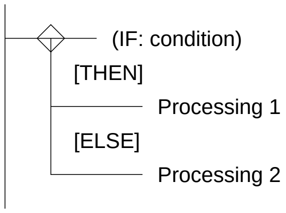
- **SPD chart**



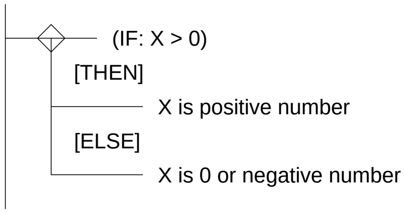
2. Conditional branch: 2 branch (IF)

Processing contents are selected according to the condition specified by IF is true or false (THEN/ELSE).

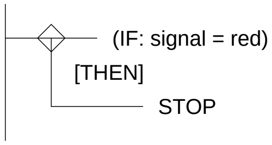
- **SPD chart**



Example 1. Identification of positive or negative of X



2. STOP if signal is red

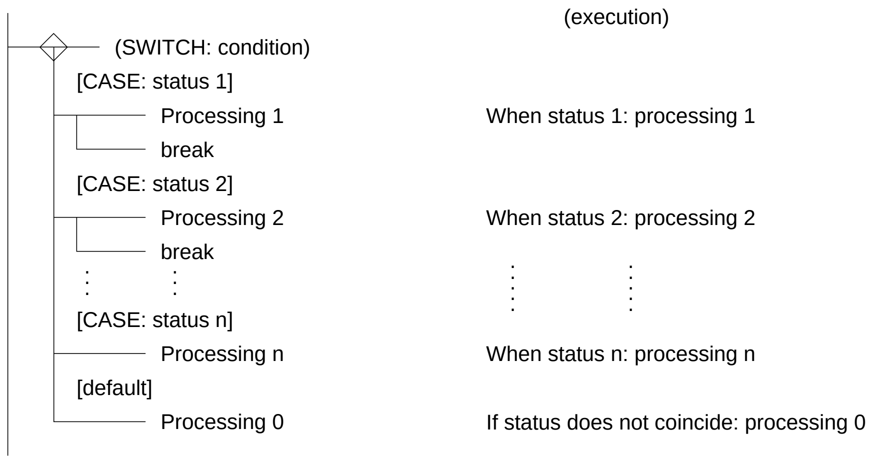


3. Conditional branch: multiple branch (SWITCH)

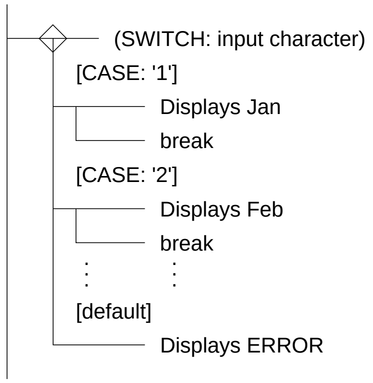
The condition specified by SWITCH is compared with the status indicated by CASE to select the processing. The processing of the SWITCH statement may be executed only when the given values coincide, or continued downward starting from when the given values coincide (if the processing is not continued downward, ‘break’ is described). If there is no coincide status, ‘default’ processing is executed (description of ‘default’ is arbitrary).

(1) Execution only on coincidence

- **SPD chart**

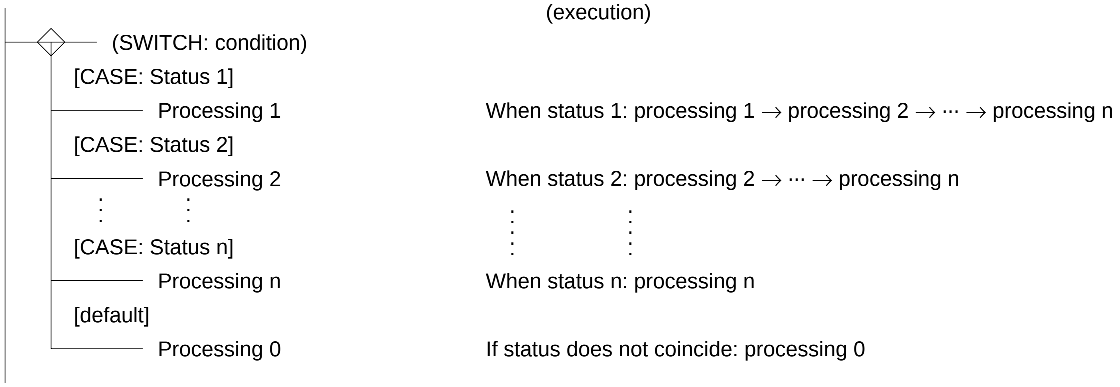


Example Displays name of month by input characters

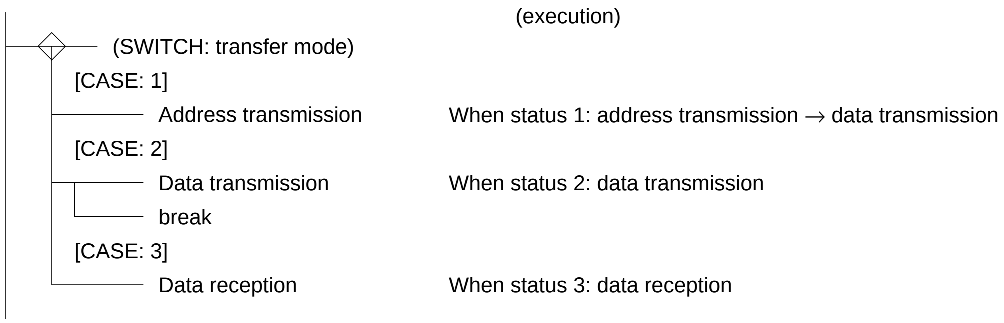


(2) If processing continues from coincidence status

- **SPD chart**


(execution)
(SWITCH: condition)
[CASE: Status 1]
Processing 1
When status 1: processing 1 → processing 2 → ··· → processing n
[CASE: Status 2]
Processing 2
When status 2: processing 2 → ··· → processing n
[CASE: Status n]
Processing n
When status n: processing n
[default]
Processing 0
If status does not coincide: processing 0

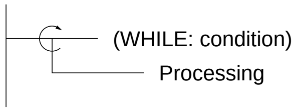
Example Transmission/reception of serial interface



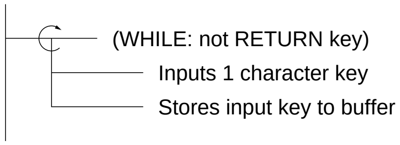
4. Conditional Loop (WHILE)

The condition indicated by WHILE is judged. If the condition is satisfied, processing is repeatedly executed (if the condition is not satisfied from the start, the processing is not executed).

- **SPD chart**



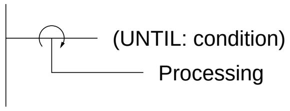
Example Buffers key until RETURN key is input



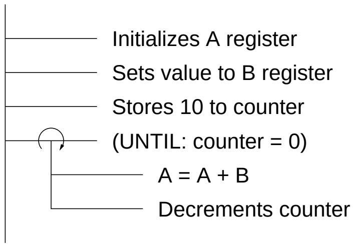
5. Conditional Loop (UNTIL)

The condition indicated by UNTIL is judged after processing has been executed, and the processing is repeatedly executed until a given condition is satisfied (even if the condition is not satisfied from the start, the processing is executed once).

- **SPD chart**



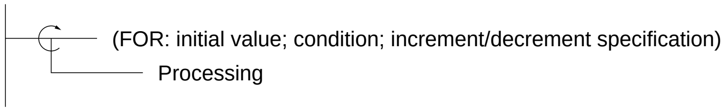
Example Multiplies value of B register by 10 and stores result to A register



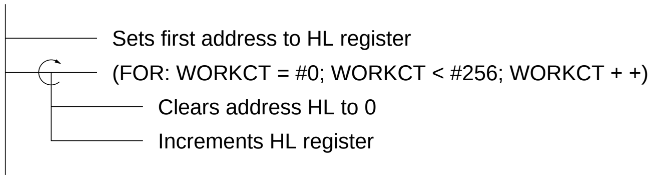
6. Conditional Loop (FOR)

While the condition of the parameter indicated by FOR is satisfied, processing is repeatedly executed.

- **SPD chart**


(FOR: initial value; condition; increment/decrement specification)
Processing

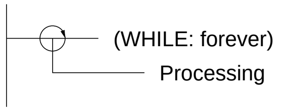
Example Clears 256 bytes to 0 starting from address HL


Sets first address to HL register
(FOR: WORKCT = #0; WORKCT < #256; WORKCT + +)
Clears address HL to 0
Increments HL register

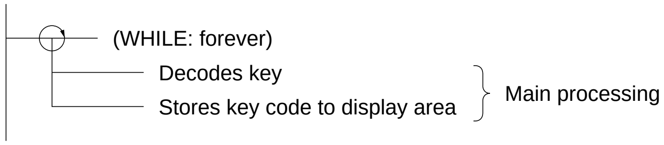
7. Infinite Loop

If ‘forever’ is set as the condition of WHILE, processing is infinitely executed.

- **SPD chart**



Example To execute main processing repeatedly

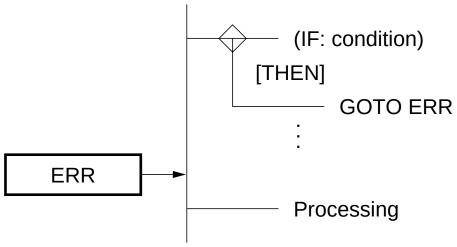


8. Connector (GOTO)

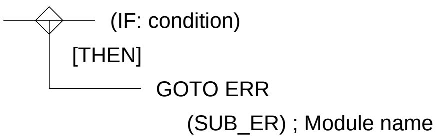
Unconditionally branches to a specified address.

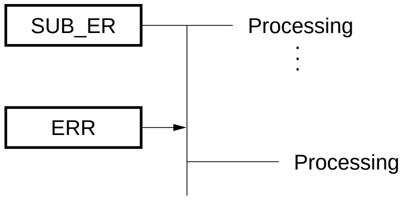
- **SPD chart**

(1) To branch to same module

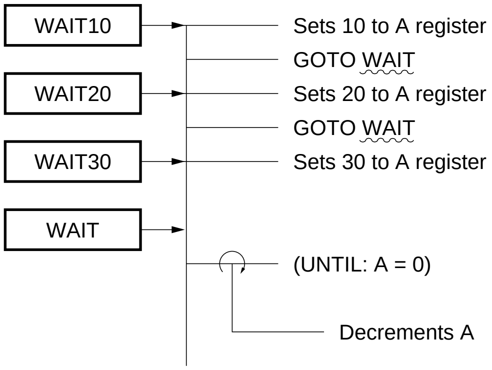


(2) To branch to different module





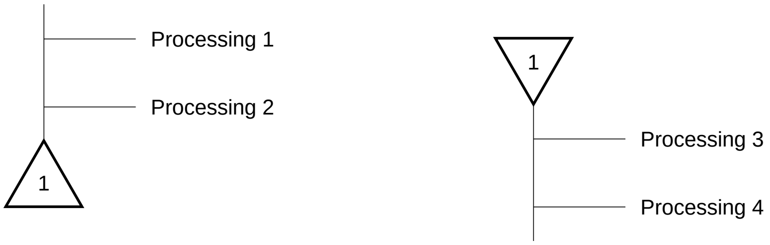
Example To select a parameter at the start address of a subroutine and set wait state



9. Connector (continuation)

Used when the SPD of one module requires two or more pages to indicate the flow of processing.

- **SPD chart**



★

APPENDIX B REVISION HISTORY

Here is the revision history of this document. “Chapter” indicates the location of the preceding edition which has been revised.

(1/3)

Edition	Major Revision from Previous Edition	Chapter
5th edition	Addition of following sections from 78K/0 Series Application Note (Basic II): 2.7 Binary Multiplication (16 bits × 16 bits) 2.8 Binary Division (32 bits ÷ 16 bits) 8.5 Half-Duplex Start-Stop Synchronization Communication 9.4 4-Channel Input A/D Conversion	Throughout
	Deletion of μPD78044, 78054, and 78064 series as applicable models (Deleted description is planned to be included in Basic II and III.)	
	Correction of count clock frequency of watchdog timer in format of timer clock select register	CHAPTER 4 APPLICATION OF WATCHDOG TIMER
		CHAPTER 7 APPLICATION OF WATCH TIMER
	Addition of note on using P30/TO0 pin as timer output pin to format of port mode register 3	CHAPTER 5 APPLICATION OF 16-BIT TIMER/EVENT COUNTER
	Addition of μPD78002 and 78002Y series to format of timer clock select register 3	CHAPTER 8 APPLICATION OF SERIAL INTERFACE
	Addition of note on using wake-up function to format of interrupt timing specification register	
	Addition of note on deletion of busy mode to format of serial bus interface control register	
	Correction of (2) Example in 8.1.1 Communication in 2-wire serial I/O mode	
	Correction of 8.2 Interface with OSD LSI (μPD6451A) as follows: (1) Description of package Deletion of <Flags used> Correction of <Nesting> to 1 level 2 bytes Deletion of enabling serial interface channel 1 interrupt from <Initial setting> Deletion of TREND from <Starting> (2) Example Deletion of TREND Deletion of enabling serial interface channel 1 interrupt (3) SPD chart, (4) Program list Deletion of setting of transfer completion status	
	Correction of program of transfer processing of SBI data in (4) Program list in 8.3.1 Application as master CPU	

(2/3)

Edition	Major Revision from Previous Edition	Chapter
6th edition	Addition of following products as applicable products: μPD78001B(A), 78002B(A) μPD78011B(A), 78012B(A), 78013(A), 78014(A) μPD78011F, 78012F, 78013F, 78014F, 78015F, 78016F, 78P018F μPD78011FY, 78012FY, 78013FY, 78014FY, 78015FY, 78016FY, 78P018FY	Throughout
	Addition of note on rewriting different data to format of timer clock select registers (TCL0, TCL1, TCL2, TCL3)	
	Addition of note on selecting mode to clear & start on coincidence between TM0 and CR00 to format of 16-bit timer mode control register	CHAPTER 5 APPLICATION OF 16-BIT TIMER/EVENT COUNTER
	Description of formats of following registers for each subseries: Format of timer clock select register 3 Format of serial operation mode register 0 Format of serial bus interface control register	CHAPTER 8 APPLICATION OF SERIAL INTERFACE
	Addition of APPENDIX B REVISION HISTORY	APPENDIX B REVISION HISTORY
7th edition	Addition of following products as applicable products: μPD780024, 780024Y, 780034, 780034Y, 78014H, 780924, 780964 subseries, μPD78018F, 78018FY, 780001, 78011F(A), 78012F(A), 78013F(A), 78014F(A), 78015F(A), 78016F(A), 78018F(A), 78P018F(A)	Throughout
	Addition of Figure 4-3 Format of Watchdog Timer Clock Select Register	CHAPTER 4 APPLICATIONS OF WATCHDOG TIMER
	Addition of Note 2 and Caution 2 to Figure 4-4 Format of Watchdog Timer Mode Register	
	Addition of Caution to Figure 5-7 Format of External Interrupt Mode Register	CHAPTER 5 APPLICATION OF 16-BIT TIMER/EVENT COUNTER
	Addition of following register formats: Figure 5-10 Format of Prescaler Mode Register 0 Figure 5-11 Format of Port Mode Register 7	
	Addition of following register formats: Figures 6-2 and 6-3 Format of Timer Clock Select Register 50 Figures 6-4 and 6-5 Format of Timer Clock Select Register 51 Figures 6-6 Format of Timer Clock Select Register 52 Figures 6-8 Format of 8-Bit Timer Mode Control Register 5n Figures 6-9 Format of 8-Bit Timer Mode Control Register 50 Figures 6-10 Format of 8-Bit Timer Mode Control Register 51 Figures 6-11 Format of 8-Bit Timer Mode Control Register 52 Figure 6-14 Format of Port Mode Register 7	CHAPTER 6 APPLICATIONS OF 8-BIT TIMER/EVENT COUNTER

(3/3)

Edition	Major Revision from Previous Edition	Chapter
7th edition	Addition of Table 8-2 Registers of Serial Interface	CHAPTER 8 APPLICATIONS OF SERIAL INTERFACE
	Addition of Caution to Figures 8-6 through 8-8 Format of Serial Operating Mode Register 0, and Note to Control of Wake-up Function	
	Addition of Caution to Figure 8-19 Format of Automatic Transmission/Reception Interval Specification Register	
	Change of μPD6252 as maintenance part in 8.1 Interface with EEPROM^{TM} (μPD6252)	
	Addition of (5) and (6) Limits when I^{2}C bus mode is used to 8.1.2 Communication in I^{2}C bus mode	
	Addition of HSC bit to Figure 9-2 Format of A/D Converter Mode Register	CHAPTER 9 APPLICATIONS OF A/D CONVERTER
	Addition of Figure 9-7 Format of Analog Input Channel Specification Register	

[MEMO]

Facsimile Message

From:

____
Name

____
Company

____
Tel. FAX

____
Address

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

Thank you for your kind support.

North America
NEC Electronics Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288
1-408-588-6130

Europe
NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

South America
NEC do Brasil S.A.
Fax: +55-11-6465-6829

Hong Kong, Philippines, Oceania
NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

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

Taiwan
NEC Electronics Taiwan Ltd.
Fax: 02-719-5951

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Japan
NEC Semiconductor Technical Hotline
Fax: 044-548-7900

I would like to report the following error/make the following suggestion:

Document title: ____

Document number: ______________________________ Page number: ______________

____

____

____

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	❑	❑	❑	❑
Technical Accuracy	❑	❑	❑	❑
Organization	❑	❑	❑	❑