

User's Manual

V850/SF1TM

32-Bit Single-Chip Microcontroller

Hardware

μ PD703078Y

μ PD703079Y

μ PD70F3079Y

[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.

V850 Family and V850/SF1 are trademarks of NEC Corporation.

Windows is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries.

Purchase of NEC I²C components conveys a license under the Philips I²C Patent Rights to use these components in an I²C system, provided that the system conforms to the I²C Standard Specification as defined by Philips.

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

License not needed: μ PD70F3079Y
The customer must judge the need for license: μ PD703078Y, 703079Y

• **The information in this document is current as of March, 2001. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC's data sheets or data books, etc., for the most up-to-date specifications of NEC semiconductor products. Not all products and/or types are available in every country. Please check with an NEC sales representative for availability and additional information.**

- No part of this document may be copied or reproduced in any form or by any means without prior written consent of NEC. NEC assumes no responsibility for any errors that may appear in this document.
- NEC does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC semiconductor products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of customer's equipment shall be done under the full responsibility of customer. NEC assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC endeavours to enhance the quality, reliability and safety of NEC semiconductor products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC semiconductor products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment, and anti-failure features.
- NEC semiconductor products are classified into the following three quality grades:
"Standard", "Special" and "Specific". The "Specific" quality grade applies only to semiconductor products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of a semiconductor product depend on its quality grade, as indicated below. Customers must check the quality grade of each semiconductor product before using it in a particular application.

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

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

"Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

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

(Note)

(1) "NEC" as used in this statement means NEC Corporation and also includes its majority-owned subsidiaries.

(2) "NEC semiconductor products" means any semiconductor product developed or manufactured by or for NEC (as defined above).

M8E 00.4

Regional Information

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

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

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

NEC Electronics Inc. (U.S.)

Santa Clara, California
Tel: 408-588-6000
800-366-9782
Fax: 408-588-6130
800-729-9288

NEC Electronics (Germany) GmbH

Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.

Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.l.

Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH

Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.

Velizy-Villacoublay, France
Tel: 01-3067-5800
Fax: 01-3067-5899

NEC Electronics (France) S.A.

Madrid Office
Madrid, Spain
Tel: 091-504-2787
Fax: 091-504-2860

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.

Novena Square, Singapore
Tel: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.

Taipei, Taiwan
Tel: 02-2719-2377
Fax: 02-2719-5951

NEC do Brasil S.A.

Electron Devices Division
Guarulhos-SP, Brasil
Tel: 11-6462-6810
Fax: 11-6462-6829

J01.2

Major Revisions in This Edition (1/2)

Page	Description
Throughout	Modification of regulator voltage
p.31	Modification of 1.4 Ordering Information
p.48	Modification of description in 2.3 (5) P40 to P47 (port 4)
p.49	Modification of description in 2.3 (6) P50 to P57 (port 5)
p.49	Modification of description in 2.3 (7) P60 to P65 (port 6)
p.50	Modification of description in 2.3 (9) P90 to P96 (port 9)
p.53	Addition of 2.3 (18) CLKOUT (Clock Out)
p.76	Modification of Figure 3-12 Application of Wraparound
p.99	Addition of Caution to 4.4.4 (1) Settings and operating states
p.102	Addition of 4.6 Cautions on Power Save Function
p.116	Modification of 5.2.4 (1) Function of P3 pins
p.134	Modification of Figure 5-12 Block Diagram of P110 and P114 to P117
p.188	Addition of 7.8 (1) Acknowledgement of interrupt servicing following EI instruction
p.195	Modification of Caution in 8.1.3 (2) Capture/compare register n0 (CR00, CR10, CR70)
p.196	Modification of Caution in 8.1.3 (3) Capture/compare register n1 (CR01, CR11, CR71)
p.221	Modification of Caution in Figure 8-21 Control Register Settings for One-Shot Pulse Output with Software Trigger
p.223	Modification of Caution in Figure 8-23 Control Register Settings for One-Shot Pulse Output with External Trigger
p.226	Modification of Figure 8-27 Data Hold Timing of Capture Register
p.226	Addition of 8.2.7 (6) (c) One-shot pulse output function
p.247	Modification of Caution in 9.3 (2) Watch timer clock selection register (WTNCS)
p.262	Addition of Remark to 11.2.2 (1) Serial operation mode register n (CSIMn)
p.264	Addition of Remark to Figure 11-3 CSIMn Setting (3-Wire Serial I/O Mode)
p.280	Addition of 11.3.2 (6) I²C0 transfer clock setting method
p.281	Modification of Table 11-3 Selection Clock Setting
p.334	Addition of Remark to 11.4.2 (4) Baud rate generator mode control registers n0, n1 (BRGMCn0, BRGMCn1)
p.339	Addition of Remark to Figure 11-29 BRGMCn0 and BRGMCn1 Settings (Asynchronous Serial Interface Mode)
p.364	Modification of 12.3 (1) A/D converter mode register 1 (ADM1)
p.373	Addition of description to 12.5 Low Power Consumption Mode
p.385	Modification of Figure 15-1 Regulator
p.386	Addition of Caution to 16.1 General
p.387	Deletion of Caution from 16.2.1 Correction control register (CORCN)
p.390	Addition of Caution to CHAPTER 17 FLASH MEMORY (μPD70F3079Y)
p.400	Modification of description in Table 18-1 Overview of Functions
p.402	Modification of Figure 18-1 Block Diagram of FCAN
p.418	Modification of figure in 18.4.1 CAN message data length registers 00 to 31 (M_DLC00 to M_DLC31)

Major Revisions in This Edition (2/2)

Page	Description
p.419	Modification of bit names in 18.4.2 CAN message control registers 00 to 31 (M_CTRL00 to M_CTRL31)
p.425	Modification of description in 18.4.5 CAN message ID registers L00 to L31 and H00 to H31 (M_IDL00 to M_IDL31 and M_IDH00 to M_IDH31)
p.429	Modification of figure in 18.4.7 CAN message status registers 00 to 31 (M_STAT00 to M_STAT31)
p.437	Addition of Caution to 18.4.12 CAN stop register (CSTOP)
p.439	Modification of GOM bit description in 18.4.13 CAN global status register (CGST)
p.442	Modification of Figure 18-2 FCAN Clocks
p.444	Modification of description of MFND4 to MFND0 bits in 18.4.17 CAN message search start/result register (CGMSS/CGMSR)
p.446	Addition of Caution to 18.4.18 CANn address mask a registers L and H (CnMASKLa and CnMASKHa)
p.446	Modification of bit names in 18.4.18 CANn address mask a registers L and H (CnMASKLa and CnMASKHa)
p.461	Modification of bit names in 18.4.25 CANn bit rate prescaler register (CnBRP)
p.464	Modification of Caution in 18.4.27 CANn synchronization control register (CnSYNC)
p.468	Modification of 18.6 Time Stamp Function
p.471	Modification of description in 18.7 Message Processing
p.472	Modification of 18.8 <2> Identifier bits set to message buffer 14 (example)
p.473	Modification of 18.8 <3> Mask setting for CAN module 1 (mask 1) (example)
p.488	Modification of description in 18.10.7 (1) Prescaler
p.488	Modification of 18.10.7 (2) Nominal bit time (8 to 25 time quantum)
p.492	Modification of Figure 18-28 Setting of CAN Global Interrupt Enable Register (CGIE)
p.493	Modification of Figure 18-30 Setting of CANn Bit Rate Prescaler Register (CnBRP)
p.503	Modification of 18.11.3 Receive setting
p.504	Modification of Figure 18-41 CAN Sleep Mode Setting
p.506	Modification of Figure 18-44 CAN Stop Mode Setting
p.506	Modification of Figure 18-45 Clearing of CAN Stop Mode
p.507	Modification of 18.12 Rules for Correct Setting of Baud Rate
p.513	Modification of 18.14.2 Burst read mode
p.514	Modification of 18.15.2 Interrupts that occur for global CAN interface
p.515	Addition of 18.17 Cautions on Use

The mark ★ shows major revised points.

PREFACE

Readers This manual is intended for users who wish to understand the functions of the V850/SF1 and design application systems using the V850/SF1.

Purpose This manual is intended to give users an understanding of the hardware functions described in Organization below.

Organization The V850/SF1 User's Manual is divided into two parts: hardware (this manual) and architecture (V850 Family™ User's Manual Architecture).

Hardware	Architecture
<ul style="list-style-type: none">• Pin functions• CPU function• Internal peripheral functions• Flash memory programming• FCAN controller	<ul style="list-style-type: none">• Data types• Register set• Instruction format and instruction set• Interrupts and exceptions• Pipeline operation

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

To find out the details of a register whose name is known:

→ Refer to **APPENDIX A REGISTER INDEX**.

To find out the details of a function, etc., whose name is known:

→ Refer to **APPENDIX C INDEX**.

To understand the details of an instruction function:

→ Refer to **V850 Family User's Manual Architecture** available separately.

How to read register formats:

→ Names of bits whose numbers are enclosed in a square are defined in the device file under reserved words.

To understand the overall functions of the V850/SF1:

→ Read this manual in accordance with the **CONTENTS**.

Conventions Data significance: Higher digits on the left and lower digits on the right
Active low representation: \overline{xxx} (overscore over pin or signal name)
Memory map addresses: Higher addresses at the top and lower addresses at the bottom
Note: Footnote for items marked with **Note** in the text
Caution: Information requiring particular attention
Remark: Supplementary information
Numerical representation: Binary ... xxxx or xxxxB
Decimal ... xxx
Hexadecimal ... xxxH
Prefixes indicating power of 2 (address space, memory capacity):
K (kilo) : 2^{10} ... 1024
M (mega) : 2^{20} ... 1024^2
G (giga) : 2^{30} ... 1024^3

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

Related documents for V850/SF1

Document Name	Document No.
V850 Family User's Manual Architecture	U10243E
V850/SF1 User's Manual Hardware	This manual
μ PD703078Y, 703079Y, 70F3079Y Data Sheet	U15183E

Related documents for development tool (user's manual)

Document Name		Document No.
IE-703002-MC (In-circuit emulator)		U11595E
IE-703079-MC-EM1 (In-circuit emulator option board)		To be prepared
CA850 (Ver. 2.30 or later) (C compiler package)	Operation	U14568E
	C Language	U14566E
	Project Manager	U14569E
	Assembly Language	U14567E
ID850 (Ver. 2.20 or later) (Integrated debugger)	Operation Windows™ Based	U14580E
SM850 (Ver.2.20 or later) (System simulator)	Operation Windows Based	U14782E
RX850 (Ver. 3.13 or later) (Real-time OS)	Basics	U13430E
	Installation	U13410E
	Technical	U13431E
RX850 Pro (Ver. 3.13) (Real-time OS)	Basics	U13773E
	Installation	U13774E
	Technical	U13772E
RD850 (Ver. 3.01) (Task debugger)		U13737E
RD850 Pro (Ver. 3.01) (Task debugger)		U13916E
AZ850 (Ver.3.0) (System performance analyzer)		U14410E
PG-FP3 (Flash memory programmer)		U13502E

CONTENTS

CHAPTER 1 INTRODUCTION	28
1.1 General	28
1.2 Features	29
1.3 Applications	31
1.4 Ordering Information	31
1.5 Pin Configuration (Top View)	32
1.6 Function Blocks	35
1.6.1 Internal block diagram.....	35
1.6.2 Internal units.....	36
CHAPTER 2 PIN FUNCTIONS	38
2.1 List of Pin Functions	38
2.2 Pin States	45
2.3 Description of Pin Functions	46
2.4 Pin I/O Circuit Types, I/O Buffer Power Supply and Connection of Unused Pins	54
2.5 I/O Circuit of Pins	56
CHAPTER 3 CPU FUNCTIONS	58
3.1 Features	58
3.2 CPU Register Set	59
3.2.1 Program register set.....	60
3.2.2 System register set.....	61
3.3 Operation Modes	63
3.4 Address Space	64
3.4.1 CPU address space.....	64
3.4.2 Images.....	65
3.4.3 Wraparound of CPU address space.....	66
3.4.4 Memory map.....	67
3.4.5 Area.....	68
3.4.6 External expansion mode.....	74
3.4.7 Recommended use of address space.....	75
3.4.8 Peripheral I/O registers.....	78
3.4.9 Specific registers.....	85
CHAPTER 4 CLOCK GENERATION FUNCTION	88
4.1 General	88
4.2 Configuration	88
4.3 Clock Output Function	89
4.3.1 Control registers.....	89
4.4 Power Save Functions	93
4.4.1 General.....	93
4.4.2 HALT mode.....	94

4.4.3	IDLE mode.....	97
4.4.4	Software STOP mode.....	99
4.5	Oscillation Stabilization Time.....	101
★ 4.6	Cautions on Power Save Function.....	102
CHAPTER 5 PORT FUNCTION.....		104
5.1	Port Configuration.....	104
5.2	Port Pin Functions.....	104
5.2.1	Port 0.....	104
5.2.2	Port 1.....	108
5.2.3	Port 2.....	112
5.2.4	Port 3.....	115
5.2.5	Ports 4 and 5.....	118
5.2.6	Port 6.....	121
5.2.7	Ports 7 and 8.....	124
5.2.8	Port 9.....	126
5.2.9	Port 10.....	129
5.2.10	Port 11.....	132
5.3	Setting When Port Pin Is Used for Alternate Function.....	136
CHAPTER 6 BUS CONTROL FUNCTION.....		140
6.1	Features.....	140
6.2	Bus Control Pins and Control Register.....	140
6.2.1	Bus control pins.....	140
6.3	Bus Access.....	141
6.3.1	Number of access clocks.....	141
6.3.2	Bus width.....	142
6.4	Memory Block Function.....	143
6.5	Wait Function.....	144
6.5.1	Programmable wait function.....	144
6.5.2	External wait function.....	145
6.5.3	Relationship between programmable wait and external wait.....	145
6.6	Idle State Insertion Function.....	146
6.7	Bus Hold Function.....	147
6.7.1	Outline of function.....	147
6.7.2	Bus hold procedure.....	148
6.7.3	Operation in power save mode.....	148
6.8	Bus Timing.....	149
6.9	Bus Priority.....	156
6.10	Memory Boundary Operation Condition.....	156
6.10.1	Program space.....	156
6.10.2	Data space.....	156
CHAPTER 7 INTERRUPT/EXCEPTION PROCESSING FUNCTION.....		157
7.1	Outline.....	157

7.1.1	Features	157
7.2	Non-Maskable Interrupt.....	160
7.2.1	Operation.....	161
7.2.2	Restore.....	163
7.2.3	NP flag.....	164
7.2.4	Noise eliminator of NMI pin	164
7.2.5	Edge detection function of NMI pin	165
7.3	Maskable Interrupts	166
7.3.1	Operation.....	166
7.3.2	Restore.....	168
7.3.3	Priorities of maskable interrupts.....	169
7.3.4	Interrupt control register (xxICn).....	172
7.3.5	In-service priority register (ISPR)	175
7.3.6	Interrupt disable flag (ID).....	175
7.3.7	Watchdog timer mode register (WDTM)	176
7.3.8	Noise elimination	176
7.3.9	Edge detection function.....	178
7.4	Software Exception.....	179
7.4.1	Operation.....	179
7.4.2	Restore.....	180
7.4.3	EP flag.....	181
7.5	Exception Trap.....	181
7.5.1	Illegal op code definition.....	181
7.5.2	Operation.....	181
7.5.3	Restore.....	182
7.6	Priority Control.....	184
7.6.1	Priorities of interrupts and exceptions	184
7.6.2	Multiple interrupt servicing.....	184
7.7	Response Time	187
7.8	Periods in Which Interrupts Are Not Acknowledged.....	187
7.9	Key Interrupt Function	190

CHAPTER 8 TIMER/COUNTER FUNCTION.....192

8.1	16-Bit Timer (TM0, TM1, TM7).....	192
8.1.1	Outline	192
8.1.2	Function.....	192
8.1.3	Configuration	194
8.1.4	Timer 0, 1, 7 control registers.....	197
8.2	16-Bit Timer (TM0, TM1, TM7) Operation.....	207
8.2.1	Operation as interval timer	207
8.2.2	PPG output operation.....	209
8.2.3	Pulse width measurement	210
8.2.4	Operation as external event counter	217
8.2.5	Operation as square-wave output	218
8.2.6	Operation as one-shot pulse output	220
8.2.7	Cautions	225
8.3	16-Bit Timer (TM2 to TM6).....	230

8.3.1	Functions	230
8.3.2	Configuration	231
8.3.3	Timer n control register.....	232
8.4	16-Bit Timer (TM2 to TM6) Operation.....	237
8.4.1	Operation as interval timer	237
8.4.2	Operation as external event counter.....	239
8.4.3	Operation as square-wave output.....	240
8.4.4	Operation as 16-bit PWM output	241
8.4.5	Cautions	243
CHAPTER 9 WATCH TIMER FUNCTION		244
9.1	Function	244
9.2	Configuration.....	245
9.3	Watch Timer Control Register	246
9.4	Operation	248
9.4.1	Operation as watch timer.....	248
9.4.2	Operation as interval timer	249
9.4.3	Cautions	250
CHAPTER 10 WATCHDOG TIMER FUNCTION		251
10.1	Functions	251
10.2	Configuration.....	253
10.3	Watchdog Timer Control Register.....	253
10.4	Operation	256
10.4.1	Operation as watchdog timer.....	256
10.4.2	Operation as interval timer	257
10.5	Standby Function Control Register.....	258
CHAPTER 11 SERIAL INTERFACE FUNCTION.....		259
11.1	Overview	259
11.2	3-Wire Serial I/O (CSI0, CSI1, CSI3)	259
11.2.1	Configuration	260
11.2.2	CSIn control registers	261
11.2.3	Operations	263
11.3	I²C Bus.....	266
11.3.1	Configuration	269
11.3.2	I ² C control registers	271
11.3.3	I ² C bus mode functions	283
11.3.4	I ² C bus definitions and control methods	284
11.3.5	I ² C interrupt request (INTIIC0).....	291
11.3.6	Interrupt request (INTIIC0) generation timing and wait control	309
11.3.7	Address match detection method	310
11.3.8	Error detection	310
11.3.9	Extension code	310
11.3.10	Arbitration	311

11.3.11	Wakeup function	313
11.3.12	Communication reservation.....	314
11.3.13	Cautions	318
11.3.14	Communication operations.....	319
11.3.15	Timing of data communication	321
11.4	Asynchronous Serial Interface (UART0, UART1).....	328
11.4.1	Configuration	328
11.4.2	UARTn control registers	330
11.4.3	Operations.....	335
11.4.4	Standby function.....	347
11.5	3-Wire Variable-Length Serial I/O (CSI4).....	348
11.5.1	Configuration	348
11.5.2	CSI4 control registers.....	351
11.5.3	Operations.....	355
CHAPTER 12	A/D CONVERTER.....	360
12.1	Function.....	360
12.2	Configuration	362
12.3	Control Registers.....	364
12.4	Operation	367
12.4.1	Basic operation.....	367
12.4.2	Input voltage and conversion result.....	369
12.4.3	A/D converter operation mode	370
12.5	Low Power Consumption Mode	373
12.6	Cautions.....	373
CHAPTER 13	DMA FUNCTIONS	377
13.1	Functions.....	377
13.2	Transfer Completion Interrupt Request.....	377
13.3	Control Registers.....	377
13.3.1	DMA peripheral I/O address registers 0 to 5 (DIOA0 to DIOA5)	377
13.3.2	DMA internal RAM address registers 0 to 5 (DRA0 to DRA5).....	378
13.3.3	DMA byte count registers 0 to 5 (DBC0 to DBC5).....	379
13.3.4	DMA start factor expansion register (DMAS).....	379
13.3.5	DMA channel control registers 0 to 5 (DCHC0 to DCHC5)	380
13.3.6	Start factor settings	381
CHAPTER 14	RESET FUNCTION.....	382
14.1	General.....	382
14.2	Pin Operations	382
14.3	Power-on-Clear Operation	383
CHAPTER 15	REGULATOR.....	385
15.1	Outline.....	385
15.2	Operation	385

CHAPTER 16 ROM CORRECTION FUNCTION	386
16.1 General	386
16.2 ROM Correction Peripheral I/O Registers	387
16.2.1 Correction control register (CORCN).....	387
16.2.2 Correction request register (CORRQ)	388
16.2.3 Correction address registers 0 to 3 (CORAD0 to CORAD3)	388
 CHAPTER 17 FLASH MEMORY (μPD70F3079Y)	 390
17.1 Features	390
17.1.1 Erasing unit.....	390
17.1.2 Write/read time	391
17.2 Writing with Flash Programmer.....	391
17.3 Programming Environment.....	391
17.4 Communication Mode	392
17.5 Pin Connection.....	394
17.5.1 V _{PP} pin	394
17.5.2 Serial interface pin.....	394
17.5.3 RESET pin	396
17.5.4 Port pin (including NMI).....	396
17.5.5 Other signal pins.....	396
17.5.6 Power supply	396
17.6 Programming Method.....	397
17.6.1 Flash memory control	397
17.6.2 Flash memory programming mode.....	397
17.6.3 Selection of communication mode.....	398
17.6.4 Communication command.....	398
17.6.5 Resources used.....	399
 CHAPTER 18 FCAN CONTROLLER.....	 400
18.1 Overview of Functions	400
18.2 Configuration.....	401
18.3 Internal Registers of FCAN Controller	403
18.3.1 Configuration of message buffers.....	403
18.3.2 List of FCAN registers	404
18.4 Control Registers	418
18.4.1 CAN message data length registers 00 to 31 (M_DLC00 to M_DLC31)	418
18.4.2 CAN message control registers 00 to 31 (M_CTRL00 to M_CTRL31).....	419
18.4.3 CAN message time stamp registers 00 to 31 (M_TIME00 to M_TIME31).....	421
18.4.4 CAN message data registers n0 to n7 (M_DATAn0 to M_DATAn7).....	423
18.4.5 CAN message ID registers L00 to L31 and H00 to H31 (M_IDL00 to M_IDL31 and M_IDH00 to M_IDH31)	425
18.4.6 CAN message configuration registers 00 to 31 (M_CONF00 to M_CONF31).....	427
18.4.7 CAN message status registers 00 to 31 (M_STAT00 to M_STAT31).....	429
18.4.8 CAN status set/clear registers 00 to 31 (SC_STAT00 to SC_STAT31).....	431
18.4.9 CAN interrupt pending register (CCINTP)	433
18.4.10 CAN global interrupt pending register (CGINTP).....	434

18.4.11	CANn interrupt pending register (CnINTP).....	435
18.4.12	CAN stop register (CSTOP)	437
18.4.13	CAN global status register (CGST)	438
18.4.14	CAN global interrupt enable register (CGIE).....	440
18.4.15	CAN main clock select register (CGCS).....	441
18.4.16	CAN time stamp count register (CGTSC).....	443
18.4.17	CAN message search start/result register (CGMSS/CGMSR).....	444
18.4.18	CANn address mask a registers L and H (CnMASKLa and CnMASKHa).....	446
18.4.19	CANn control register (CnCTRL).....	448
18.4.20	CANn definition register (CnDEF)	452
18.4.21	CANn information register (CnLAST).....	455
18.4.22	CANn error count register (CnERC).....	456
18.4.23	CANn interrupt enable register (CnIE).....	457
18.4.24	CANn bus active register (CnBA).....	459
18.4.25	CANn bit rate prescaler register (CnBRP).....	460
18.4.26	CANn bus diagnostic information register (CnDINF).....	463
18.4.27	CANn synchronization control register (CnSYNC)	464
18.5	Cautions Regarding Bit Set/Clear Function	466
18.6	Time Stamp Function	468
18.7	Message Processing	471
18.8	Mask Function.....	472
18.9	Protocol	474
18.9.1	Protocol mode function.....	474
18.9.2	Message formats.....	475
18.10	Functions	484
18.10.1	Determination of bus priority	484
18.10.2	Bit stuffing.....	484
18.10.3	Multiple masters	484
18.10.4	Multi-cast.....	484
18.10.5	CAN sleep mode/CAN stop mode function	485
18.10.6	Error control function	485
18.10.7	Baud rate control function	488
18.11	Operations	491
18.11.1	Initialization processing	491
18.11.2	Transmit setting.....	502
18.11.3	Receive setting.....	503
18.11.4	CAN sleep mode	504
18.11.5	CAN stop mode	506
18.12	Rules for Correct Setting of Baud Rate	507
18.13	Prioritization of Message Buffers During Receive Comparison	510
18.13.1	Reception of data frames	510
18.13.2	Reception of remote frames	511
18.14	Ensuring Data Consistency	512
18.14.1	Sequential data read	512
18.14.2	Burst read mode.....	513
18.15	Interrupt Conditions.....	514
18.15.1	Interrupts that occur for FCAN controller.....	514
18.15.2	Interrupts that occur for global CAN interface	514

	18.16 How to Shut down FCAN Controller.....	515
★	18.17 Cautions on Use	515
	APPENDIX A REGISTER INDEX.....	516
	APPENDIX B INSTRUCTION SET LIST.....	524
	APPENDIX C INDEX	531

LIST OF FIGURES (1/6)

Figure No.	Title	Page
3-1	CPU Register Set	59
3-2	CPU Address Space.....	64
3-3	Images on Address Space	65
3-4	Program Space.....	66
3-5	Data Space	66
3-6	Memory Map	67
3-7	Internal ROM/Flash Memory Area	68
3-8	Internal RAM Area	70
3-9	Internal Peripheral I/O Area	71
3-10	External Memory Area (When Expanded to 64 KB, 256 KB, or 1 MB).....	72
3-11	External Memory Area (When Expanded to 4 MB).....	73
3-12	Application of Wraparound	76
3-13	Recommended Memory Map (Flash Memory Version)	77
4-1	Clock Generator.....	88
4-2	Oscillation Stabilization Time.....	101
5-1	Block Diagram of P00 to P07.....	107
5-2	Block Diagram of P10 and P12.....	110
5-3	Block Diagram of P11 and P13 to P15	111
5-4	Block Diagram of P20 to P25.....	114
5-5	Block Diagram of P26 and P27.....	114
5-6	Block Diagram of P30 to P34.....	117
5-7	Block Diagram of P40 to P47 and P50 to P57.....	120
5-8	Block Diagram of P60 to P65.....	123
5-9	Block Diagram of P70 to P77 and P80 to P83.....	125
5-10	Block Diagram of P90 to P96.....	128
5-11	Block Diagram of P100 to P107.....	131
5-12	Block Diagram of P110 and P114 to P117	134
5-13	Block Diagram of P111 to P113.....	135
6-1	Byte Access (8 Bits).....	142
6-2	Halfword Access (16 Bits).....	142
6-3	Word Access (32 Bits).....	142
6-4	Memory Block	143
6-5	Wait Control.....	145
6-6	Example of Inserting Wait States	145
6-7	Bus Hold Procedure.....	148
6-8	Memory Read	149
6-9	Memory Write	153

LIST OF FIGURES (2/6)

Figure No.	Title	Page
6-10	Bus Hold Timing.....	155
7-1	Non-Maskable Interrupt Servicing.....	161
7-2	Acknowledging Non-Maskable Interrupt Requests	162
7-3	RETI Instruction Processing	163
7-4	NP Flag (NP).....	164
7-5	Maskable Interrupt Servicing	167
7-6	RETI Instruction Processing	168
7-7	Example of Interrupt Nesting	170
7-8	Example of Servicing Interrupt Requests Simultaneously Generated	172
7-9	Interrupt Disable Flag (ID).....	175
7-10	Software Exception Processing	179
7-11	RETI Instruction Processing	180
7-12	EP Flag (EP).....	181
7-13	Illegal Op Code	181
7-14	Exception Trap Processing	182
7-15	RETI Instruction Processing	183
7-16	Pipeline Operation at Interrupt Request Acknowledgement	187
7-17	Key Return Block Diagram.....	191
8-1	Block Diagram of TM0, TM1, and TM7	193
8-2	Control Register Settings When TMn Operates as Interval Timer.....	207
8-3	Configuration of Interval Timer.....	208
8-4	Timing of Interval Timer Operation	208
8-5	Control Register Settings in PPG Output Operation	209
8-6	Control Register Settings for Pulse Width Measurement with Free-Running Counter and One Capture Register	210
8-7	Configuration for Pulse Width Measurement with Free-Running Counter	211
8-8	Timing of Pulse Width Measurement with Free-Running Counter and One Capture Register (with Both Edges Specified).....	211
8-9	Control Register Settings for Measurement of Two Pulse Widths with Free-Running Counter	212
8-10	CRn1 Capture Operation with Rising Edge Specified.....	213
8-11	Timing of Pulse Width Measurement with Free-Running Counter (with Both Edges Specified).....	213
8-12	Control Register Settings for Pulse Width Measurement with Free-Running Counter and Two Capture Registers	214
8-13	Timing of Pulse Width Measurement with Free-Running Counter and Two Capture Registers (with Rising Edge Specified)	215
8-14	Control Register Settings for Pulse Width Measurement by Restarting	216
8-15	Timing of Pulse Width Measurement by Restarting (with Rising Edge Specified).....	216
8-16	Control Register Settings in External Event Counter Mode	217

LIST OF FIGURES (3/6)

Figure No.	Title	Page
8-17	Configuration of External Event Counter	218
8-18	Timing of External Event Counter Operation (with Rising Edge Specified)	218
8-19	Control Register Settings in Square-Wave Output Mode	219
8-20	Timing of Square-Wave Output Operation	220
8-21	Control Register Settings for One-Shot Pulse Output with Software Trigger	221
8-22	Timing of One-Shot Pulse Output Operation with Software Trigger	222
8-23	Control Register Settings for One-Shot Pulse Output with External Trigger	223
8-24	Timing of One-Shot Pulse Output Operation with External Trigger (with Rising Edge Specified)	224
8-25	Start Timing of 16-Bit Timer Register n	225
8-26	Timing After Changing Compare Register During Timer Count Operation	225
8-27	Data Hold Timing of Capture Register	226
8-28	Operation Timing of OVF _n Flag	227
8-29	Block Diagram of TM2 to TM6	230
8-30	Timing of Interval Timer Operation	237
8-31	Timing of External Event Counter Operation (with Rising Edge Specified)	239
8-32	Square-Wave Output Operation Timing	240
8-33	Timing of PWM Output	242
8-34	Start Timing of Timer n	243
9-1	Block Diagram of Watch Timer	244
9-2	Operation Timing of Watch Timer/Interval Timer	249
9-3	Watch Timer Interrupt Request (INTW _{TN}) Generation (Interrupt Period = 0.5 s)	250
10-1	Block Diagram of Watchdog Timer	251
11-1	Block Diagram of 3-Wire Serial I/O	260
11-2	CSIM _n Setting (Operation Stopped Mode)	263
11-3	CSIM _n Setting (3-Wire Serial I/O Mode)	264
11-4	Timing of 3-Wire Serial I/O Mode	265
11-5	Block Diagram of I ² C0	267
11-6	Serial Bus Configuration Example Using I ² C Bus	268
11-7	I ² C0 Transfer Clock Frequency (f _{SCL})	281
11-8	Pin Configuration Diagram	283
11-9	I ² C Bus's Serial Data Transfer Timing	284
11-10	Start Conditions	284
11-11	Address	285
11-12	Transfer Direction Specification	286
11-13	ACK Signal	287
11-14	Stop Condition	288

LIST OF FIGURES (4/6)

Figure No.	Title	Page
11-15	Wait Signal.....	289
11-16	Arbitration Timing Example	312
11-17	Communication Reservation Timing	315
11-18	Timing for Accepting Communication Reservations	316
11-19	Communication Reservation Flow Chart	317
11-20	Master Operation Flow Chart.....	319
11-21	Slave Operation Flow Chart.....	320
11-22	Example of Master to Slave Communication (When 9-Clock Wait Is Selected for Both Master and Slave)	322
11-23	Example of Slave to Master Communication (When 9-Clock Wait Is Selected for Both Master and Slave)	325
11-24	Block Diagram of UARTn.....	329
11-25	ASIMn Setting (Operation Stopped Mode).....	335
11-26	ASIMn Setting (Asynchronous Serial Interface Mode).....	336
11-27	ASISn Setting (Asynchronous Serial Interface Mode)	337
11-28	BRGCn Setting (Asynchronous Serial Interface Mode)	338
11-29	BRGMCn0 and BRGMCn1 Settings (Asynchronous Serial Interface Mode)	339
11-30	Error Tolerance (When k = 16), Including Sampling Errors	341
11-31	Format of Transmit/Receive Data in Asynchronous Serial Interface	342
11-32	Timing of Asynchronous Serial Interface Transmit Completion Interrupt.....	344
11-33	Timing of Asynchronous Serial Interface Receive Completion Interrupt.....	345
11-34	Receive Error Timing	346
11-35	Block Diagram of 3-Wire Variable-Length Serial I/O	349
11-36	When Transfer Bit Length Other Than 16 Bits Is Set.....	350
11-37	CSIM4 Setting (Operation Stopped Mode)	355
11-38	CSIM4 Setting (3-Wire Variable-Length Serial I/O Mode)	356
11-39	CSIB4 Setting (3-Wire Variable-Length Serial I/O Mode).....	357
11-40	Timing of 3-Wire Variable-Length Serial I/O Mode	358
11-41	Timing of 3-Wire Variable-Length Serial I/O Mode (When CSIB4 = 08H)	359
12-1	Block Diagram of A/D Converter.....	361
12-2	Basic Operation of A/D Converter.....	368
12-3	Relationship Between Analog Input Voltage and A/D Conversion Result.....	369
12-4	A/D Conversion by Hardware Start (with Falling Edge Specified)	371
12-5	A/D Conversion by Software Start	372
12-6	Handling of Analog Input Pin	374
12-7	A/D Conversion End Interrupt Generation Timing.....	375
12-8	Handling of ADCV _{DD} Pin	376

LIST OF FIGURES (5/6)

Figure No.	Title	Page
13-1	Correspondence Between DRAn Setting Value and Internal RAM	378
14-1	System Reset Timing.....	382
15-1	Regulator	385
16-1	Block Diagram of ROM Correction	386
16-2	ROM Correction Operation and Program Flow.....	389
17-1	Environment Required for Writing Programs to Flash Memory	391
17-2	Communication with Dedicated Flash Programmer (UART0)	392
17-3	Communication with Dedicated Flash Programmer (CSI0)	392
17-4	Communication with Dedicated Flash Programmer (CSI0 + HS)	393
17-5	V _{PP} Pin Connection Example	394
17-6	Conflict of Signals (Serial Interface Input Pin)	395
17-7	Malfunction of Other Device	395
17-8	Conflict of Signals (<u>RESET</u> Pin)	396
17-9	Procedure for Manipulating Flash Memory	397
17-10	Flash Memory Programming Mode.....	398
17-11	Communication Command	398
18-1	Block Diagram of FCAN.....	402
18-2	FCAN Clocks	442
18-3	Example of Bit Setting/Clearing Operations	466
18-4	16-Bit Data During Write Operation	467
18-5	Time Stamp Function Setting for Message Reception (When CxCTRL Register's TMR Bit = 0)	468
18-6	Time Stamp Function Setting for Message Reception (When CxCTRL Register's TMR Bit = 1)	469
18-7	Time Stamp Function Setting for Message Transmission (When M_CTRL Register's ATS Bit = 1)	470
18-8	Example of Message Processing	471
18-9	Composition of Layers	474
18-10	Data Frame	475
18-11	Remote Frame.....	476
18-12	Start of Frame (SOF)	476
18-13	Arbitration Field (in Standard Format Mode)	477
18-14	Arbitration Field (in Extended Format Mode).....	477
18-15	Control Field	478
18-16	Data Field	479
18-17	CRC Field	479
18-18	ACK Field.....	480
18-19	End of Frame (EOF)	480

LIST OF FIGURES (6/6)

Figure No.	Title	Page
18-20	Interframe Space	481
18-21	Error Frame.....	482
18-22	Overload Frame	483
18-23	Nominal Bit Time.....	488
18-24	Coordination of Data Bit Synchronization	489
18-25	Resynchronization	490
18-26	Initialization Processing	491
18-27	Setting of CAN Main Clock Select Register (CGCS)	492
18-28	Setting of CAN Global Interrupt Enable Register (CGIE).....	492
18-29	Setting of CAN Global Status Register (CGST).....	493
18-30	Setting of CANn Bit Rate Prescaler Register (CnBRP).....	493
18-31	Setting of CANn Synchronization Control Register (CnSYNC).....	494
18-32	Setting of CANn Interrupt Enable Register (CnIE).....	495
18-33	Setting of CANn Definition Register (CnDEF).....	496
18-34	Setting of CANn Control Register (CnCTRL).....	497
18-35	Setting of CANn Address Mask a Registers L and H (CnMASKLa and CnMASKHa).....	498
18-36	Message Buffer Setting.....	499
18-37	Setting of CAN Message Configuration Registers 00 to 31 (M_CONF00 to M_CONF31).....	500
18-38	Setting of CAN Message Control Registers 00 to 31 (M_CTRL00 to M_CTRL31)	501
18-39	Transmit Setting.....	502
18-40	Receive Setting.....	503
18-41	CAN Sleep Mode Setting	504
18-42	Clearing of CAN Sleep Mode by CAN Bus Active Status	504
18-43	Clearing of CAN Sleep Mode by CPU.....	505
18-44	CAN Stop Mode Setting	506
18-45	Clearing of CAN Stop Mode.....	506
18-46	CnSYNC Register Settings	510
18-47	Sequential Data Read.....	512

LIST OF TABLES (1/3)

Table No.	Title	Page
1-1	Product Lineup of V850/SF1.....	28
2-1	Pin I/O Buffer Power Supplies	38
2-2	Pin Operating State According to Operation Mode.....	45
3-1	Program Registers.....	60
3-2	System Register Numbers.....	61
3-3	Interrupt/Exception Table.....	69
4-1	Operating Statuses in HALT Mode	95
4-2	Operating Statuses in IDLE Mode	97
4-3	Operating Statuses in Software STOP Mode	100
5-1	Pin I/O Buffer Power Supplies	104
5-2	Port 0 Alternate Function Pins	105
5-3	Port 1 Alternate Function Pins	108
5-4	Port 2 Alternate Function Pins	112
5-5	Port 3 Alternate Function Pins	115
5-6	Alternate Function Pins of Ports 4 and 5	118
5-7	Port 6 Alternate Function Pins	121
5-8	Alternate Function Pins of Ports 7 and 8	124
5-9	Port 9 Alternate Function Pins	126
5-10	Port 10 Alternate Function Pins	129
5-11	Port 11 Alternate Function Pins	132
5-12	Setting When Port Pin Is Used for Alternate Function.....	136
6-1	Bus Control Pins	140
6-2	Number of Access Clocks	141
6-3	Bus Priority	156
7-1	Interrupt Source List	158
7-2	Interrupt Control Registers (xxICn).....	174
7-3	Priorities of Interrupts and Exceptions	184
7-4	Description of Key Return Detection Pin	190
8-1	Configuration of Timers 0, 1, and 7	194
8-2	Valid Edge of TIn0 Pin and Capture Trigger of CRn0.....	195
8-3	Valid Edge of TIn1 Pin and Capture Trigger of CRn0.....	195
8-4	TIn0 Pin Valid Edge and CRn1 Capture Trigger.....	196
8-5	Configuration of Timers 2 to 6	231

LIST OF TABLES (2/3)

Table No.	Title	Page
9-1	Interval Time of Interval Timer	245
9-2	Configuration of Watch Timer	245
9-3	Interval Time of Interval Timer	249
10-1	Loop Detection Time of Watchdog Timer	252
10-2	Interval Time of Interval Timer	252
10-3	Watchdog Timer Configuration	253
10-4	Loop Detection Time of Watchdog Timer	256
10-5	Interval Time of Interval Timer	257
11-1	Configuration of CSIn.....	260
11-2	Configuration of I ² C0.....	269
11-3	Selection Clock Setting	281
11-4	INTIIC0 Generation Timing and Wait Control	309
11-5	Extension Code Bit Definitions.....	310
11-6	Status During Arbitration and Interrupt Request Generation Timing	312
11-7	Wait Periods	314
11-8	Configuration of UARTn.....	328
11-9	Relationship Between Main Clock and Baud Rate.....	340
11-10	Receive Error Causes.....	346
11-11	Configuration of CSI4.....	348
12-1	Configuration of A/D Converter	362
13-1	Start Factor Settings	381
17-1	Signal Generation of Dedicated Flash Programmer (PG-FP3).....	393
17-2	Pins Used by Serial Interfaces.....	394
17-3	List of Communication Mode	398
17-4	Flash Memory Control Commands	399
17-5	Response Commands	399
18-1	Overview of Functions	400
18-2	Configuration of Message Buffers.....	403
18-3	Addresses of M_DLCn (n = 00 to 31)	418
18-4	Addresses of M_CTRLn (n = 00 to 31)	421
18-5	Addresses of M_TIMEn (n = 00 to 31)	422
18-6	Addresses of M_DATAnx (n = 00 to 31, x = 0 to 7).....	424
18-7	Addresses of M_IDLn and M_IDHn (n = 00 to 31).....	426
18-8	Addresses of M_CONFn (n = 00 to 31)	428

LIST OF TABLES (3/3)

Table No.	Title	Page
18-9	Addresses of M_STATn (n = 00 to 31)	430
18-10	Addresses of SC_STATn (n = 00 to 31)	432
18-11	Addresses of CnMASKLa and CnMASKHa (a = 0 to 3, n = 1, 2)	447
18-12	Example When Adding Captured Time Stamp Counter Value to Last 2 Bytes of Transmit Message	470
18-13	RTR Frame Settings	477
18-14	Protocol Mode Setting and Number of Identifier (ID) Bits	477
18-15	Data Length Code Settings	478
18-16	Operation When Third Bit of Intermission Is "Dominant (D)"	482
18-17	Determination of Bus Priority	484
18-18	Bit Stuffing	484
18-19	Types of Errors	485
18-20	Error Frame Output Timing	485
18-21	Types of Error Statuses	486
18-22	Error Counter	487
18-23	Prioritization of Message Buffers When Receiving Data Frames	510
18-24	Prioritization of Message Buffers When Receiving Remote Frames	511
B-1	Symbols in Operand Description	524
B-2	Symbols Used for Op Code	525
B-3	Symbols Used for Operation Description	525
B-4	Symbols Used for Flag Operation	526
B-5	Condition Codes	526

CHAPTER 1 INTRODUCTION

The V850/SF1 is a product in NEC's V850 Family of single-chip microcontrollers designed for low power operation.

1.1 General

The V850/SF1 is a 32-bit single-chip microcontroller that includes the V850 Family's CPU core, and peripheral functions such as ROM/RAM, a timer/counter, a serial interface, an A/D converter, a DMA controller, and features an automotive LAN (FCAN (Full Controller Area Network)).

In addition to high real-time response characteristics and 1-clock-pitch basic instructions, the V850/SF1 has multiplication, saturation operation, and bit manipulation instructions, realized by a hardware multiplier.

Table 1-1 shows the outline of the V850/SF1 product lineup.

Table 1-1. Product Lineup of V850/SF1

Product Name		ROM		RAM Size	I ² C	FCAN
Commercial Name	Part Number	Type	Size			
V850/SF1	μPD703078Y	Mask ROM	256 KB	16 KB	On-chip I ² C	1 channel
	μPD703079Y					2 channels
	μPD70F3079Y	Flash memory				

1.2 Features

- Number of instructions: 74
- ★ ○ Minimum instruction execution time
 - 62.5 ns (operating at 16 MHz, external power supply 5 V, regulator output 3.0 V)
- General-purpose registers 32 bits × 32 registers
- Instruction set
 - Signed multiplication (16 × 16 → 32): 125 ns (operating at 16 MHz)
 - (able to execute instructions in parallel continuously without creating any register hazards).
 - Saturation operations (overflow and underflow detection functions are included)
 - 32-bit shift instruction: 1 clock
 - Bit manipulation instructions
 - Load/store instructions with long/short format
- Memory space
 - 16 MB of linear address space (for programs and data)
 - External expandability: Expandable to 4 MB
 - Memory block allocation function: 2 MB per block
 - Programmable wait function
 - Idle state insertion function
- External bus interface
 - 16-bit data bus (address/data multiplexed)
 - 3 V to 5 V interface enabled
 - Bus hold function
 - External wait function
- Internal memory
 - μPD703078Y, 703079Y (mask ROM: 256 KB/RAM: 16 KB)
 - μPD70F3079Y (flash memory: 256 KB/RAM: 16 KB)
- Interrupts and exceptions
 - Non-maskable interrupts: 2 sources
 - Maskable interrupts: 41 sources (μPD703078Y)
 - 44 sources (μPD703079Y, 70F3079Y)
 - Software exceptions: 32 sources
 - Exception trap: 1 source
- I/O lines
 - Total: 84 (12 input ports and 72 I/O ports)
 - 3 V to 5 V interface enabled
- Timer/counter
 - 16-bit timer: 3 channels (TM0, TM1, TM7)
 - 16-bit timer: 5 channels (TM2 to TM6)
- Watch timer
 - When operating under subsystem or main system clock: 1 channel
 - Operation using the subsystem or main system clock is also possible in the IDLE mode.
- Watchdog timer 1 channel
- Serial interface (SIO)
 - Asynchronous serial interface (UART)
 - Clocked serial interface (CSI)
 - I²C bus interface (I²C)
 - 8-/16-bit variable-length serial interface
 - CSI/UART: 2 channels
 - CSI/I²C: 1 channel
 - CSI (8-/16-bit valuable): 1 channel
 - Dedicated baud rate generator: 3 channels

- A/D converter 10-bit resolution: 12 channels
- DMA controller Internal RAM ↔ internal peripheral I/O: 6 channels
- ROM correction 4 points modifiable
- ★ ○ Regulator 4.0 V to 5.5 V input → internal 3.0 V (μ PD703078Y, 703079Y)
4.5 V to 5.5 V input → internal 3.0 V (μ PD70F3079Y)
- Key return function 4 to 8 pins selectable, falling edge fixed
- Clock generator During main system clock or subsystem clock operation
5-level CPU clock (including slew rate and sub operations)
- Power-saving functions HALT/IDLE/STOP modes
- Automotive LAN 2 channels (μ PD703079Y, 70F3079Y)
1 channel (μ PD703078Y)
- Package 100-pin plastic LQFP (fine pitch, 14 × 14)
100-pin plastic QFP (14 × 20)
- CMOS structure All static circuits

1.3 Applications

AV equipment

Example: Car audio equipment

1.4 Ordering Information

	Part Number	Package	Internal ROM
★	μ PD703078YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	Mask ROM
	μ PD703078YGF-xxx-3BA ^{Note}	100-pin plastic QFP (14 × 20)	Mask ROM
★	μ PD703079YGC-xxx-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	Mask ROM
	μ PD703079YGF-xxx-3BA ^{Note}	100-pin plastic QFP (14 × 20)	Mask ROM
★	μ PD70F3079YGC-8EU	100-pin plastic LQFP (fine pitch) (14 × 14)	Flash memory
	μ PD70F3079YGF-3BA ^{Note}	100-pin plastic QFP (14 × 20)	Flash memory

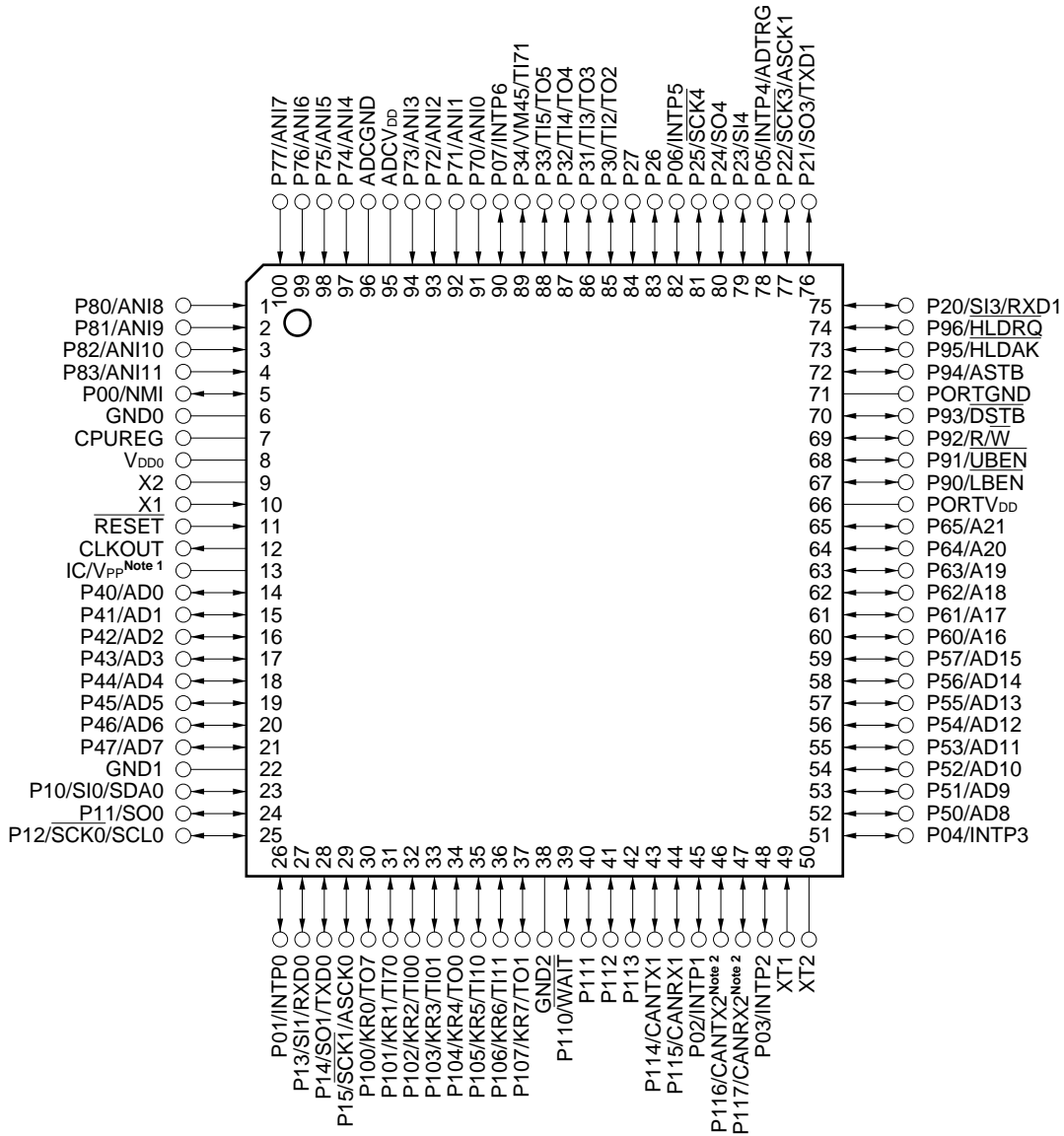
Note Under development

- Remarks**
1. xxx indicates ROM code suffix.
 2. ROMless devices are not provided.

1.5 Pin Configuration (Top View)

100-pin plastic LQFP (fine pitch) (14 × 14)

- μ PD703078YGC-xxx-8EU
- μ PD703079YGC-xxx-8EU
- μ PD70F3079YGC-8EU



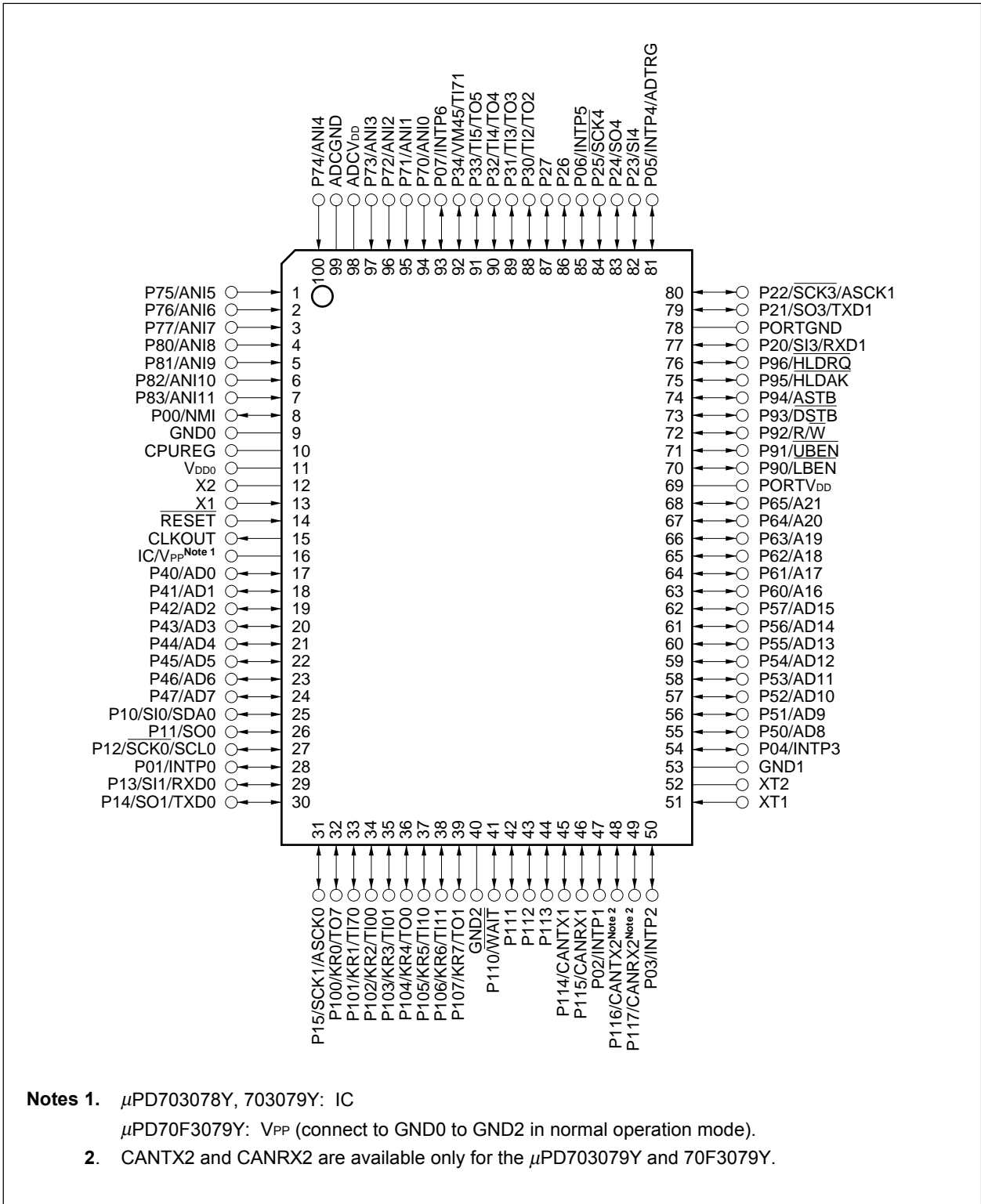
Notes 1. μ PD703078Y, 703079Y: IC

μ PD70F3079Y: V_{PP} (connect to GND0 to GND2 in normal operation mode).

2. CANTX2 and CANRX2 are available only for the μ PD703079Y and 70F3079Y.

100-pin plastic QFP (14 × 20)

- μ PD703078YGF-xxx-3BA
- μ PD703079YGF-xxx-3BA
- μ PD70F3079YGF-3BA



Notes 1. μ PD703078Y, 703079Y: IC

μ PD70F3079Y: V_{PP} (connect to GND0 to GND2 in normal operation mode).

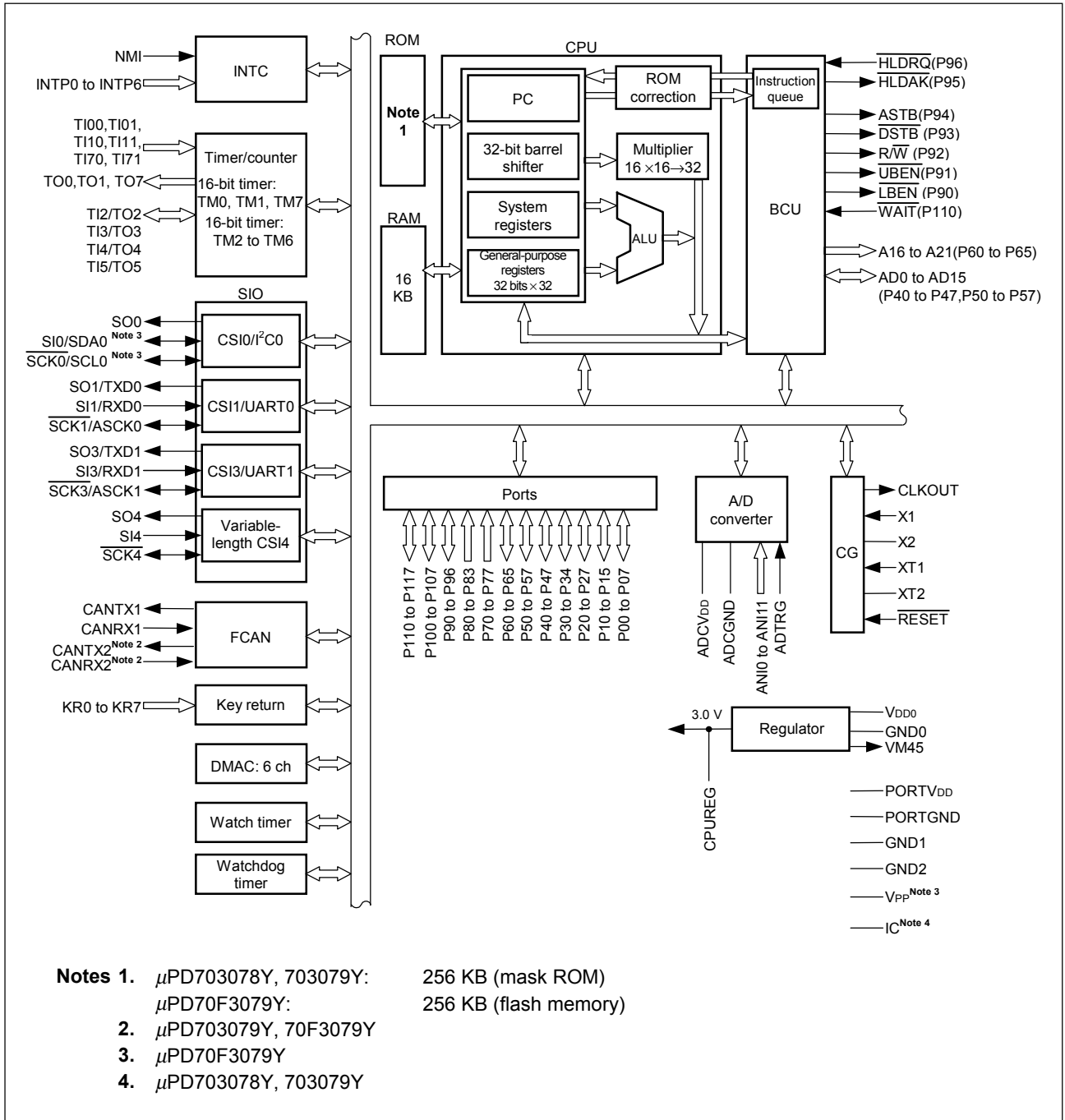
2. CANTX2 and CANRX2 are available only for the μ PD703079Y and 70F3079Y.

Pin names

A16 to A21:	Address bus	P50 to P57:	Port 5
AD0 to AD15:	Address/data bus	P60 to P65:	Port 6
ADCGND:	Ground for analog	P70 to P77:	Port 7
ADCV _{DD}	Power supply for analog	P80 to P83:	Port 8
ADTRG:	AD trigger input	P90 to P96:	Port 9
ANI0 to ANI11:	Analog input	P100 to P107:	Port 10
ASCK0, ASCK1:	Asynchronous serial clock	P110 to P117:	Port 11
ASTB:	Address strobe	$\overline{\text{RESET}}$:	Reset
CANRX1, CANRX2:	FCAN receive data	$\overline{\text{R/W}}$:	Read/write status
CANTX1, CANTX2:	FCAN transmit data	RXD0, RXD1:	Receive data
CLKOUT:	Clock output	$\overline{\text{SCK0}}$, $\overline{\text{SCK1}}$,	
CPUREG:	Regulator control	$\overline{\text{SCK3}}$, $\overline{\text{SCK4}}$:	Serial clock
$\overline{\text{DSTB}}$:	Data strobe	SCL0:	Serial clock
GND0, GND1,		SDA0:	Serial data
GND2:	Ground	SI0, SI1, SI3, SI4:	Serial input
$\overline{\text{HLDAK}}$:	Hold acknowledge	SO0, SO1, SO3,	
$\overline{\text{HLDRQ}}$:	Hold request	SO4:	Serial output
IC:	Internally connected	TI00, TI01, TI10,	
INTP0 to INTP6:	Interrupt request from peripherals	TI11, TI2 to TI5,	
KR0 to KR7:	Key return	TI70, TI71:	Timer input
$\overline{\text{LBEN}}$:	Lower byte enable	TO0 to TO5, TO7:	Timer output
NMI:	Non-maskable interrupt request	TXD0, TXD1:	Transmit data
PORTGND:	Ground for ports	$\overline{\text{UBEN}}$:	Upper byte enable
PORTV _{DD}	Power supply for ports	V _{DD0} :	Power supply
P00 to P07:	Port 0	VM45:	V _{DD} = 4.5 V monitor output
P10 to P15:	Port 1	V _{PP} :	Programming power supply
P20 to P27:	Port 2	$\overline{\text{WAIT}}$:	Wait
P30 to P34:	Port 3	X1, X2:	Crystal for main clock
P40 to P47:	Port 4	XT1, XT2:	Crystal for sub-clock

1.6 Function Blocks

1.6.1 Internal block diagram



1.6.2 Internal units

(1) CPU

The CPU uses five-stage pipeline control to enable single-clock execution of address calculations, arithmetic logic operations, data transfers, and almost all other instruction processing.

Other dedicated on-chip hardware, such as the multiplier (16 bits \times 16 bits \rightarrow 32 bits) and the barrel shifter (32 bits) help accelerate processing of complex instructions.

(2) Bus control unit (BCU)

The BCU starts a required external bus cycle based on the physical address obtained by the CPU. When an instruction is fetched from external memory space and the CPU does not send a bus cycle start request, the BCU generates a prefetch address and prefetches the instruction code. The prefetched instruction code is stored in an instruction queue.

(3) ROM

This consists of a 256 KB mask ROM or flash memory mapped to the address space starting at 00000000H. ROM can be accessed by the CPU in one clock cycle during instruction fetch.

(4) RAM

This consists of a 16 KB RAM mapped to the address space starting at FFFFB000H. RAM can be accessed by the CPU in one clock cycle during data access.

(5) Interrupt controller (INTC)

This controller handles hardware interrupt requests (NMI, INTP0 to INTP6) from on-chip peripheral hardware and external hardware. Eight levels of interrupt priorities can be specified for these interrupt requests, and multiplexed servicing control can be performed for interrupt sources.

(6) Clock generator (CG)

The clock generator includes two types of oscillators; each for main system clock (f_{xx}) and for subsystem clock (f_{XT}), generates five types of clocks (f_{xx} , $f_{xx}/2$, $f_{xx}/4$, $f_{xx}/8$, and f_{XT}), and supplies one of them as the operating clock for the CPU (f_{CPU}).

(7) Timer/counter

An eight-channel 16-bit timer/event counter is equipped, enabling measurement of pulse intervals and frequency as well as programmable pulse output.

(8) Watch timer

This timer counts the reference time period (0.5 second) for counting the clock (the 32.768 kHz subsystem clock or the 8.388 MHz main system clock). At the same time, the watch timer can be used as an interval timer for the main system clock.

(9) Watchdog timer

A watchdog timer is equipped to detect inadvertent program loops, system abnormalities, etc.

This timer can also be used as an interval timer.

When used as a watchdog timer, it generates a non-maskable interrupt request (INTWDT) after an overflow occurs, and when used as an interval timer, it generates a maskable interrupt request (INTWDTM) after an overflow occurs.

(10) Serial interface (SIO)

The V850/SF1 includes four kinds of serial interfaces: asynchronous serial interfaces (UART0, UART1), clocked serial interfaces (CSIn), an 8-/16-bit variable-length serial interface (CSI4), and an I²C bus interface (I²C0). Up to four channels can be used at the same time. Two of these channels are switchable between the UART and CSI and another one is switchable between CSI and I²C (n = 0, 1, 3).

For UART0 and UART1, data is transferred via the TXD0, TXD1, RXD0, and RXD1 pins.

For CSIn, data is transferred via the SOn, SIn, and \overline{SCKn} pins.

For CSI4, data is transferred via the SO4, SI4, and $\overline{SCK4}$ pins.

For I²C0, data is transferred via the SDA0 and SCL0 pins.

For UART and CSI4, a dedicated baud rate generator is provided.

(11) A/D converter

This high-speed, high-resolution 10-bit A/D converter includes 12 analog input pins. Conversion is performed using the successive approximation method.

(12) DMA controller

A six-channel DMA controller is equipped. This controller transfers data between the internal RAM and on-chip peripheral I/O devices in response to interrupt requests sent by on-chip peripheral I/O.

(13) Ports

As shown below, the following ports have general-purpose port functions and control pin functions.

Port	I/O	Port Function	Control Function
Port 0	8-bit I/O	General-purpose port	NMI, external interrupt, A/D converter trigger
Port 1	6-bit I/O		Serial interface
Port 2	8-bit I/O		Serial interface
Port 3	5-bit I/O		Timer I/O, V _{DD} = 4.5 V monitor output
Port 4	8-bit I/O		External address/data bus
Port 5	8-bit I/O		
Port 6	6-bit I/O		External address bus
Port 7	8-bit input		A/D converter analog input
Port 8	4-bit input		
Port 9	7-bit I/O		External bus interface control signal I/O
Port 10	8-bit I/O		Timer I/O, key return input
Port 11	8-bit I/O		Wait control, FCAN data I/O

(14) FCAN controller

The FCAN controller is a small-scale digital data transmission system for transferring data between units. A two-channel FCAN controller is incorporated in the μ PD703079Y and 70F3079Y (FCAN1, FCAN2), and a one-channel FCAN controller is incorporated in the μ PD703078Y (FCAN1).

CHAPTER 2 PIN FUNCTIONS

2.1 List of Pin Functions

The names and functions of pins of V850/SF1 are described below, divided into port pins and non-port pins.

There are two types of power supplies for the pin I/O buffers: $ADC_{V_{DD}}$ and $PORT_{V_{DD}}$. The relationship between these power supplies and the pins is described below.

Table 2-1. Pin I/O Buffer Power Supplies

Power Supply	Corresponding Pins
$ADC_{V_{DD}}$	Port 7, port 8
$PORT_{V_{DD}}$	Port 0, port 1, port 2, port 3, port 4, port 5, port 6, port 9, port 10, port 11, \overline{RESET} , CLKOUT

(1) Port pins

(1/3)

Pin Name	I/O	PULL	Function	Alternate Function
P00	I/O	No	Port 0 8-bit I/O port Input/output can be specified in 1-bit units.	NMI
P01				INTP0
P02				INTP1
P03				INTP2
P04				INTP3
P05				INTP4/ADTRG
P06				INTP5
P07				INTP6
P10	I/O	No	Port 1 6-bit I/O port Input/output can be specified in 1-bit units.	SI0/SDA0
P11				SO0
P12				SCK0/SCL0
P13				SI1/RXD0
P14				SO1/TXD0
P15				SCK1/ASCK0
P20	I/O	No	Port 2 8-bit I/O port Input/output can be specified in 1-bit units.	SI3/RXD1
P21				SO3/TXD1
P22				SCK3/ASCK1
P23				SI4
P24				SO4
P25				SCK4
P26				-
P27				-
P30	I/O	No	Port 3 5-bit I/O port Input/output can be specified in 1-bit units.	TI2/TO2
P31				TI3/TO3
P32				TI4/TO4
P33				TI5/TO5
P34				VM45/TI71
P40	I/O	No	Port 4 8-bit I/O port Input/output can be specified in 1-bit units.	AD0
P41				AD1
P42				AD2
P43				AD3
P44				AD4
P45				AD5
P46				AD6
P47				AD7

Remark PULL: On-chip pull-up resistor

Pin Name	I/O	PULL	Function	Alternate Function
P50	I/O	No	Port 5 8-bit I/O port Input/output can be specified in 1-bit units.	AD8
P51				AD9
P52				AD10
P53				AD11
P54				AD12
P55				AD13
P56				AD14
P57				AD15
P60	I/O	No	Port 6 6-bit I/O port Input/output can be specified in 1-bit units.	A16
P61				A17
P62				A18
P63				A19
P64				A20
P65				A21
P70	Input	No	Port 7 8-bit input port	ANI0
P71				ANI1
P72				ANI2
P73				ANI3
P74				ANI4
P75				ANI5
P76				ANI6
P77				ANI7
P80	Input	No	Port 8 4-bit input port	ANI8
P81				ANI9
P82				ANI10
P83				ANI11
P90	I/O	No	Port 9 7-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{LBEN}}$
P91				$\overline{\text{UBEN}}$
P92				$\overline{\text{R/W}}$
P93				$\overline{\text{DSTB}}$
P94				$\overline{\text{ASTB}}$
P95				$\overline{\text{HLDAK}}$
P96				$\overline{\text{HLDRQ}}$

Remark PULL: On-chip pull-up resistor

Pin Name	I/O	PULL	Function	Alternate Function
P100	I/O	Yes	Port 10 8-bit I/O port Input/output can be specified in 1-bit units.	KR0/TO7
P101				KR1/TI7
P102				KR2/TI00
P103				KR3/TI01
P104				KR4/TO0
P105				KR5/TI10
P106				KR6/TI11
P107				KR7/TO1
P110	I/O	No	Port 11 8-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{WAIT}}$
P111				–
P112				–
P113				–
P114				CANTX1
P115				CANRX1
P116				CANTX2 ^{Note}
P117				CANRX2 ^{Note}

Note Only for the μ PD703079Y, 70F3079Y

Remark PULL: On-chip pull-up resistor

(2) Non-port pins

(1/3)

Pin Name	I/O	PULL	Function	Alternate Function
A16 to A21	Output	No	Higher address bus used for external memory expansion	P60 to P65
AD0 to AD7	I/O	No	16-bit multiplexed address/data bus used for external memory expansion	P40 to P47
AD8 to AD15				P50 to P57
ADCGND	–	–	Ground potential for A/D converter	–
ADCV _{DD}	–	–	Power supply pin and reference voltage pin for A/D converter	–
ADTRG	Input	No	A/D converter external trigger input	P05/INTP4
ANI0 to ANI7	Input	No	Analog input to A/D converter	P70 to P77
ANI8 to ANI11				P80 to P83
ASCK0	Input	No	Baud rate clock input for UART0 and UART1	P15/ $\overline{\text{SCK1}}$
ASCK1				P22/ $\overline{\text{SCK3}}$
ASTB	Output	No	External address strobe signal output	P94
CANRX1	Input	No	CAN1 receive data input	P115
CANRX2	Input		CAN2 receive data input ($\mu\text{PD703079Y}$ and 70F3079Y only)	P117
CANTX1	Output		CAN1 transmit data output	P114
CANTX2	Output		CAN2 transmit data output ($\mu\text{PD703079Y}$ and 70F3079Y only)	P116
CLKOUT	Output	–	Internal system clock output	–
CPUREG	–	–	Connection of regulator output stabilizing capacitance	–
$\overline{\text{DSTB}}$	Output	No	External data strobe signal output	P93
GND0 to GND2	–	–	Ground potential	–
$\overline{\text{HLD}}\text{AK}$	Output	No	Bus hold acknowledge output	P95
$\overline{\text{HLD}}\text{RQ}$	Input	No	Bus hold request input	P96
IC	–	–	Internally connected ($\mu\text{PD703078Y}$ and $703079Y$ only)	–
INTP0 to INTP3	Input	Yes	External interrupt request input (analog noise elimination)	P01 to P04
INTP4			External interrupt request input (digital noise elimination)	P05/ADTRG
INTP5				P06
INTP6			External interrupt request input (digital noise elimination for remote control)	P07
KR0	Input	Yes	Key return input	P100/TO7
KR1				P101/TI70
KR2				P102/TO0
KR3				P103/TO1
KR4				P104/TO0
KR5				P105/TO10
KR6				P106/TO11
KR7				P107/TO1
LBEN	Output	No	External data bus's lower byte enable signal output	P90
NMI	Input	No	Non-maskable interrupt request input	P00

Remark PULL: On-chip pull-up resistor

Pin Name	I/O	PULL	Function	Alternate Function
PORTGND	–	–	Ground potential for port output	–
PORTV _{DD}	–	–	Positive power supply for port output	–
RESET	Input	–	System reset input	–
R/W	Output	No	External read/write status output	P92
RXD0	Input	No	Serial receive data input for UART0 and UART1	P13/SI1
RXD1				P20/SI3
SCK0	I/O	No	Serial clock I/O (3-wire type) for CSI0, CSI1, CSI3	P12/SCL0
SCK1				P15/ASCK0
SCK3				P22/ASCK1
SCK4			Serial clock I/O for variable-length CSI4 (3-wire type)	P25
SCL0	I/O	No	Serial clock I/O for I ² C0	P12/SCK0
SDA0	I/O	No	Serial transmit/receive data I/O for I ² C0	P10/SI0
SI0	Input	No	Serial receive data input (3-wire type) for CSI0, CSI1, CSI3	P10/SDA0
SI1				P13/RXD0
SI3				P20/RXD1
SI4			Serial receive data input (3-wire type) for variable-length CSI4	P23
SO0	Output	No	Serial transmit data output (3-wire type) for CSI0, CSI1, CSI3	P11
SO1				P14/TXD0
SO3				P21/TXD1
SO4			Serial transmit data output for variable-length CSI4 (3-wire type)	P24
TI00	Input	Yes	Shared as external capture trigger input and external count clock input for TM0	P102/KR2
TI01			External capture trigger input for TM0	P103/KR3
TI10			External count clock input/external capture trigger input for TM1	P105/KR5
TI11			External capture trigger input for TM1	P106/KR6
TI2		No	External count clock input for TM2	P30/TO2
TI3			External count clock input for TM3	P31/TO3
TI4			External count clock input for TM4	P32/TO4
TI5			External count clock input for TM5	P33/TO5
TI70		Yes	External count clock input/external capture trigger input for TM7	P101/KR1
TI71		No	External capture trigger input for TM7	P34/VM45
TO0		Output	Yes	Pulse signal output for TM0
TO1	Pulse signal output for TM1			P107/KR7
TO2	No		Pulse signal output for TM2	P30/TO2
TO3			Pulse signal output for TM3	P31/TO3
TO4			Pulse signal output for TM4	P32/TO4
TO5			Pulse signal output for TM5	P33/TO5
TO7	Yes		Pulse signal output for TM7	P100/KR0

Remark PULL: On-chip pull-up resistor

Pin Name	I/O	PULL	Function	Alternate Function
TXD0	Output	No	Serial transmit data output for UART0 and UART1	P14/SO1
TXD1				P21/SO3
$\overline{\text{UBEN}}$	Output	No	Higher byte enable signal output for external data bus	P91
V _{DD0}	–	–	Positive power supply pin	–
VM45	Output	No	V _{DD} = 4.5 V monitor output	P34/TI71
V _{PP}	–	–	High-voltage apply pin for program write/verify (μ PD70F3079Y only)	–
$\overline{\text{WAIT}}$	Input	Yes	Control signal input for inserting wait in bus cycle	P110
X1	Input	No	Resonator connection for main clock	–
X2	–			–
XT1	Input	No	Resonator connection for subsystem clock	–
XT2	–			–

Remark PULL: On-chip pull-up resistor

2.2 Pin States

The operating states of various pins are described below with reference to the operation mode.

Table 2-2. Pin Operating State According to Operation Mode

Pin \ Operation Mode	Reset	STOP Mode	IDLE Mode	HALT Mode	Bus Hold	Idle State
AD0 to AD15	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z
A16 to A21	Hi-Z	Hi-Z	Hi-Z	Held	Hi-Z	Held
$\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$	Hi-Z	Hi-Z	Hi-Z	Held	Hi-Z	Held
$\overline{\text{R/W}}$	Hi-Z	Hi-Z	Hi-Z	H	Hi-Z	H
$\overline{\text{DSTB}}$	Hi-Z	Hi-Z	Hi-Z	H	Hi-Z	H
ASTB	Hi-Z	Hi-Z	Hi-Z	H	Hi-Z	H
$\overline{\text{HLDRQ}}$	–	–	–	Operating	Operating	Operating
$\overline{\text{HLDAK}}$	Hi-Z	Hi-Z	Hi-Z	Operating	L	Operating
$\overline{\text{WAIT}}$	–	–	–	–	–	–
CLKOUT	Hi-Z	L	L	Operating ^{Note}	Operating ^{Note}	Operating ^{Note}

Note Low level (L) when in clock output inhibit mode

Remark

- Hi-Z: High impedance
- Held: State is held during previously set external bus cycle
- L: Low-level output
- H: High-level output
- : Input not sampled

2.3 Description of Pin Functions

(1) P00 to P07 (port 0) ... 3-state I/O

Port 0 is an 8-bit I/O port in which input and output can be specified in 1-bit units.

P00 to P07 can function as I/O port pins and can also function as NMI inputs, external interrupt request inputs, and external triggers for the A/D converter. Port/control mode can be selected for each bit, and the pin's valid edge is specified by the EGP0 and EGN0 registers.

(a) Port mode

P00 to P07 can be set to input or output in 1-bit units according to the contents of the port 0 mode register (PM0).

(b) Control mode

(i) NMI (non-maskable interrupt request) ... input

This is a non-maskable interrupt request signal input pin.

(ii) INTP0 to INTP6 (interrupt request from peripherals) ... input

These are external interrupt request input pins.

(iii) ADTRG (AD trigger input) ... input

This is the A/D converter's external trigger input pin. This pin is controlled by A/D converter mode register 1 (ADM1).

(2) P10 to P15 (port 1) ... 3-state I/O

Port 1 is a 6-bit I/O port in which input and output can be specified in 1-bit units.

P10 to P15 can function as I/O port pins and can also operate as input or output pins for the serial interface.

Port/control mode can be selected for each bit.

P10 and P12 can be selected as normal output or N-ch open-drain output.

(a) Port mode

P10 to P15 can be set to input or output in 1-bit units according to the contents of the port 1 mode register (PM1).

(b) Control mode

(i) SI0, SI1 (serial input 0, 1) ... input

These are the serial receive data input pins of CSI0 and CSI1.

(ii) SO0, SO1 (serial output 0, 1) ... output

These are the serial transmit data output pins of CSI0 and CSI1.

(iii) $\overline{\text{SCK0}}$, $\overline{\text{SCK1}}$ (serial clock 0, 1) ... 3-state I/O

These are the serial clock I/O pins for CSI0 and CSI1.

(iv) SDA0 (serial data 0) ... I/O

This is the serial transmit/receive data I/O pin for I²C0.

(v) SCL0 (serial clock 0) ... I/O

This is the serial clock I/O pin for I²C0.

(vi) RXD0 (receive data 0) ... input

This is the serial receive data input pin of UART0.

(vii) TXD0 (transmit data 0) ... output

This is the serial transmit data output pin of UART0.

(viii) ASCK0 (asynchronous serial clock 0) ... input

This is the serial baud rate clock input pin of UART0.

(3) P20 to P27 (port 2) ... 3-state I/O

Port 2 is an 8-bit I/O port in which input and output can be specified in 1-bit units.

P20 to P27 can function as I/O port pins and input or output pins for the serial interface.

Port/control mode can be selected for each bit.

(a) Port mode

P20 to P27 can be set to input or output in 1-bit units according to the contents of the port 2 mode register (PM2).

(b) Control mode**(i) SI3, SI4 (serial input 3, 4) ... input**

These are the serial receive data input pins of CSI3 and CSI4.

(ii) SO3, SO4 (serial output 3, 4) ... output

These are the serial transmit data output pins of CSI3 and CSI4.

(iii) $\overline{\text{SCK3}}$, $\overline{\text{SCK4}}$ (serial clock 3, 4) ... 3-state I/O

These are the serial clock I/O pins of CSI3 and CSI4.

(iv) RXD1 (receive data 1) ... input

This is the serial receive data input pin of UART1.

(v) TXD1 (transmit data 1) ... output

This is the serial transmit data output pin of UART1.

(vi) ASCK1 (asynchronous serial clock 1) ... input

This is the serial baud rate clock input pin of UART1.

(4) P30 to P34 (port 3) ... 3-state I/O

Port 3 is a 5-bit I/O port in which input and output can be specified in 1-bit units.

P30 to P34 can function as I/O port pins, input or output pins for the timer/counter, and $V_{DD} = 4.5\text{ V}$ monitor output.

Port/control mode can be selected for each bit.

(a) Port mode

P30 to P34 can be set to input or output in 1-bit units according to the contents of the port 3 mode register (PM3).

(b) Control mode**(i) TI2, TI3, TI4, TI5, TI71 (timer input 2, 3, 4, 5, 71) ... input**

These are the external count clock input pins of timers 2, 3, 4, 5, and 7.

(ii) TO2, TO3, TO4, TO5 (timer output 2, 3, 4, 5) ... output

These are the pulse signal output pins of timers 2, 3, 4, and 5.

(iii) VM45 ($V_{DD} = 4.5\text{ V}$ monitor output) ... output

This is the $V_{DD} = 4.5\text{ V}$ monitor output pin.

(5) P40 to P47 (port 4) ... 3-state I/O

Port 4 is an 8-bit I/O port in which input and output can be specified in 1-bit units.

P40 to P47 can function as I/O port pins and as a time division address/data bus (AD0 to AD7) when memory is expanded externally.

★

(a) Port mode

P40 to P47 can be set to input or output in 1-bit units according to the contents of the port 4 mode register (PM4).

(b) Control mode (external expansion mode)

P40 to P47 can be set as AD0 to AD7 according to the contents of the memory expansion mode register (MM).

(i) AD0 to AD7 (address/data 0 to 7) ... 3-state I/O

These pins comprise a multiplexed address/data bus that is used for external access. At the address timing (T1 state), these pins operate as the AD0 to AD7 (22-bit address) output pins. At the data timing (T2, TW, T3), they operate as the lower 8-bit I/O bus pins for 16-bit data. The output changes in synchronization with the rising edge of the clock in each state within the bus cycle. When the timing sets the bus cycle as inactive, these pins go into a high-impedance state.

(6) P50 to P57 (port 5) ... 3-state I/O

Port 5 is an 8-bit I/O port in which input and output can be specified in 1-bit units.

P50 to P57 can function as I/O port pins and as a time division address/data bus (AD8 to AD15) when memory is expanded externally.

(a) Port mode

P50 to P57 can be set to input or output in 1-bit units according to the contents of the port 5 mode register (PM5).

(b) Control mode (external expansion mode)

P50 to P57 can be specified as AD8 to AD15 according to the contents of the memory expansion mode register (MM).

(i) AD8 to AD15 (address/data 8 to 15) ... 3-state I/O

These pins comprise a multiplexed address/data bus that is used for external access. At the address timing (T1 state), these pins operate as the AD8 to AD15 (22-bit address) output pins. At the data timing (T2, TW, T3), they operate as the higher 8-bit I/O bus pins for 16-bit data. The output changes in synchronization with the rising edge of the clock in each state within the bus cycle. When the timing sets the bus cycle as inactive, these pins go into a high-impedance state.

(7) P60 to P65 (port 6) ... 3-state I/O

Port 6 is a 6-bit I/O port in which input and output can be specified in 1-bit units.

P60 to P65 can function as I/O port pins and as an address bus (A16 to A21) when memory is expanded externally. During 8-bit access of port 6, the higher 2 bits are ignored when writing, and are read as "00" when reading. Port/control mode can be selected in 2-bit units.

(a) Port mode

P60 to P65 can be set to input or output in 1-bit units according to the contents of the port 6 mode register (PM6).

(b) Control mode (external expansion mode)

P60 to P65 can be set as A16 to A21 according to the contents of the memory expansion mode register (MM).

(i) A16 to A21 (address 16 to 21) ... output

These pins comprise an address bus that is used for external access. These pins operate as the higher 6-bit address output pins within a 22-bit address. The output changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. When the timing sets the bus cycle as inactive, the previous bus cycle's address is retained.

(8) P70 to P77 (port 7), P80 to P83 (port 8) ... input

Port 7 is an 8-bit input-only port in which all pins are fixed to input. Port 8 is a 4-bit input-only port in which all pins are fixed to input.

P70 to P77 and P80 to P83 can function as input ports and as analog input pins for the A/D converter in control mode. However, they cannot be switched between input ports and analog input pins.

(a) Port mode

P70 to P77 and P80 to P83 are input-only pins.

(b) Control mode

P70 to P77 also function as pins ANI0 to ANI7 and P80 to P83 also function as ANI8 to ANI11, but these alternate functions are not switchable.

(i) ANI0 to ANI11 (analog input 0 to 11) ... input

These are analog input pins for the A/D converter.

Connect a capacitor between ADCV_{DD} and ADCGND to prevent noise-related operation faults. Also, do not apply voltage that is outside the range for ADCV_{DD} and ADCGND to pins that are being used as inputs for the A/D converter. If it is possible for noise above the ADCV_{DD} range or below the ADCGND to enter, clamp these pins using a diode that has a small V_F value.

(9) P90 to P96 (port 9) ... 3-state I/O

Port 9 is a 7-bit I/O port in which input and output can be specified in 1-bit units.

P90 to P96 can function as I/O port pins, control signal output pins, and bus hold control signal output pins when memory is expanded externally.

★ During 8-bit access of port 9, the MSB is ignored when writing and is read as “0” when reading.

(a) Port mode

P90 to P96 can be set to input or output in 1-bit units according to the contents of the port 9 mode register (PM9).

(b) Control mode (external expansion mode)

P90 to P96 can be set to operate as control signal outputs for external memory expansion according to the contents of the memory expansion mode register (MM).

(i) $\overline{\text{LBEN}}$ (lower byte enable) ... output

This is the lower byte enable signal output pin for an external 16-bit data bus. The output changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. When the timing sets the bus cycle as inactive, the previous bus cycle's status is retained.

(ii) $\overline{\text{UBEN}}$ (upper byte enable) ... output

This is the higher byte enable signal output pin for an external 16-bit data bus. During byte access of even-numbered addresses, these pins are set as inactive (high level). The output changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. When the timing sets the bus cycle as inactive, the previous bus cycle's status is retained.

Access		$\overline{\text{UBEN}}$	$\overline{\text{LBEN}}$	AD0
Word access		0	0	0
Halfword access		0	0	0
Byte access	Even-numbered address	1	0	0
	Odd-numbered address	0	1	1

(iii) $\overline{\text{R/W}}$ (read/write status) ... output

This is an output pin for the status signal that indicates whether the bus cycle is a read cycle or write cycle during external access. High level is set during the read cycle and low level is set during the write cycle. The output changes in synchronization with the rising edge of the clock in the T1 state of the bus cycle. High level is set when the timing sets the bus cycle as inactive.

(iv) $\overline{\text{DSTB}}$ (data strobe) ... output

This is an output pin for the external data bus's access strobe signal. Output becomes active (low level) during the T2 and TW states of the bus cycle. Output becomes inactive (high level) when the timing sets the bus cycle as inactive.

(v) $\overline{\text{ASTB}}$ (address strobe) ... output

This is an output pin for the external address bus's latch strobe signal. Output becomes active (low level) in synchronization with the falling edge of the clock during the T1 state of the bus cycle, and becomes inactive (high level) in synchronization with the falling edge of the clock during the T3 state of the bus cycle. Output becomes inactive when the timing sets the bus cycle as inactive.

(vi) $\overline{\text{HLDAK}}$ (hold acknowledge) ... output

This is an output pin for the acknowledge signal that indicates high impedance status for the address bus, data bus, and control bus when the V850/SF1 receives a bus hold request.

The address bus, data bus, and control bus are set to high impedance when this signal is active.

(vii) $\overline{\text{HLDRQ}}$ (hold request) ... input

This is the input pin by which an external device requests the V850/SF1 to release the address bus, data bus, and control bus. This pin can be input asynchronously to CLKOUT. When this pin is active, the address bus, data bus, and control bus are set to high impedance. This occurs either when the V850/SF1 completes execution of the current bus cycle or immediately if no bus cycle is being executed. The HLDAK signal is then set as active and the bus is released.

(10) P100 to P107 (port 10) ... 3-state I/O

Port 10 is an 8-bit I/O port in which input and output can be specified in 1-bit units.

P100 to P107 can function as I/O port pins, timer/counter I/O pins, and key return input pins.

(a) Port mode

P100 to P107 can be set to input or output in 1-bit units according to the contents of the port 10 mode register (PM10).

(b) Control mode**(i) KR0 to KR7 (key return 0 to 7) ... input**

These are key interrupt input pins. Their operations are specified by the key return mode register (KRM).

(ii) TI00, TI01, TI10, TI11, TI70 (timer input 00, 01, 10, 11, 70) ... input

These are external count clock input pins for timers 0, 1, and 7.

(iii) TO0, TO1, TO7 (timer output 0, 1, 7) ... output

These are pulse signal output pins for timers 0, 1, and 7.

(11) P110 to P117 (port 11) ... 3-state I/O

Port 11 is an 8-bit I/O port in which input and output can be specified in 1-bit units.

P110 to P117 can function as I/O port pins, FCAN data I/O pins, and the control signal ($\overline{\text{WAIT}}$) that inserts waits into the bus cycle.

(a) Port mode

P110 to P117 can be set to input or output in 1-bit units according to the contents of the port 11 mode register (PM11).

(b) Control mode**(i) $\overline{\text{WAIT}}$ (wait) ... input**

This is an input pin for the control signal used to insert waits into the bus cycle. This pin is sampled at the falling edge of the clock during the T2 or TW state of the bus cycle.

ON/OFF switching of the wait function is performed by the port alternate function control register (PAC).

(ii) CANRX1, CANRX2 (CAN receive data 1, 2) ... input

These are data input signals for CAN1 and CAN2. CANRX2 is available only for the $\mu\text{PD703079Y}$ and 70F3079Y.

(iii) CANTX1, CANTX2 (CAN transmit data 1, 2) ... output

These are data output signals for CAN1 and CAN2. CANTX2 is available only for the $\mu\text{PD703079Y}$ and 70F3079Y.

(12) $\overline{\text{RESET}}$ (reset) ... input

$\overline{\text{RESET}}$ input is an asynchronous input signal and has a constant low level width regardless of the operating clock's status. When this signal is input, a system reset is executed as the first priority ahead of all other operations.

In addition to being used for ordinary initialization/start operations, this pin can also be used to cancel a standby mode (HALT, IDLE, or STOP mode).

(13) X1, X2 (crystal)

These pins are used to connect the resonator that generates the main clock.

(14) XT1, XT2 (crystal for sub-clock)

These pins are used to connect the resonator that generates the subclock.

(15) ADCV_{DD} (analog power supply)

This is the analog positive power supply pin for the A/D converter and alternate-function ports.

(16) ADCGND (ground for analog)

This is the ground pin for the A/D converter and alternate-function ports.

(17) CPUREG (regulator control)

This is the regulator pin for the CPU power supply. Connect this pin to GND0 to GND2 via a capacitor of 1 μF (recommended value).

★ (18) CLKOUT (clock out) ... output

This pin outputs the bus clock generated internally.

(19) PORTV_{DD} (power supply for ports)

This is the positive power supply pin for I/O ports and alternate-function pins.

(20) PORTGND (ground for ports)

This is the ground pin for I/O ports and alternate-function pins (except for the alternate-function ports of the bus interface).

(21) V_{DD0} (power supply)

This is the positive power supply pin. All V_{DD0} pins should be connected to a positive power supply.

(22) GND0 to GND2 (ground)

These are the ground pins. All GND0 to GND2 pins should be grounded.

(23) V_{PP} (programming power supply)

This is the positive power supply pin used for flash memory programming mode.

This pin is used in the $\mu\text{PD70F3079Y}$. Connect to GND0 to GND2 in normal operation mode.

(24) IC (internally connected)

This is an internally connected pin used in the $\mu\text{PD703078Y}$ and 703079Y.

Connect directly to GND0 to GND2 in normal operation mode.

2.4 Pin I/O Circuit Types, I/O Buffer Power Supply and Connection of Unused Pins

(1/2)

Pin	Alternate Function	I/O Circuit Type	I/O Buffer Power Supply	Recommended Connection Method
P00	NMI	8	PORTV _{DD}	Input: Independently connect to PORTV _{DD} or PORTGND via a resistor. Output: Leave open.
P01 to P04	INTP0 to INTP3			
P05	INTP4/ADTRG			
P06, P07	INTP5, INTP6			
P10	SI0/SDA0	10	PORTV _{DD}	
P11	SO0	5		
P12	SCK0/SCL0	10		
P13	SI1/RXD0	8		
P14	SO1/TXD0	5		
P15	SCK1/ASCK0	8		
P20	SI3/RXD1	8		
P21	SO3/TXD1	5		
P22	SCK3/ASCK1	8		
P23	SI4			
P24	SO4	5		
P25	SCK4	8		
P26	–			
P27	–	5		
P30 to P33	TI2/TO2 to TI5/TO5	8	PORTV _{DD}	
P34	VM45/TI71			
P40 to P47	AD0 to AD7	5	PORTV _{DD}	
P50 to P57	AD8 to AD15	5	PORTV _{DD}	
P60 to P65	A16 to A21	5	PORTV _{DD}	
P70 to P77	ANI0 to ANI7	9	ADCV _{DD}	Independently connect to ADCV _{DD} or ADCGND via a resistor.
P80 to P83	ANI8 to ANI11	9	ADCV _{DD}	
P90	LBEN	5	PORTV _{DD}	Input: Independently connect to PORTV _{DD} or PORTGND via a resistor. Output: Leave open.
P91	UBEN			
P92	R/W			
P93	DSTB			
P94	ASTB			
P95	HLDKAK			
P96	HLDKAK			

Pin	Alternate Function	I/O Circuit Type	I/O Buffer Power Supply	Recommended Connection Method
P100	KR0/TO7	8-A	PORTV _{DD}	Input: Independently connect to PORTV _{DD} or PORTGND via a resistor. Output: Leave open.
P101	KR1/TI70			
P102	KR2/TI00			
P103	KR3/TI01			
P104	KR4/TO0			
P105	KR5/TI10			
P106	KR6/TI11			
P107	KR7/TO1			
P110	WAIT	5	PORTV _{DD}	
P111 to P113	–			
P114	CANTX1	5		
P115	CANRX1	8		
P116	CANTX2 ^{Note 1}	5		
P117	CANRX2 ^{Note 1}	8		
CLKOUT	–	4	PORTV _{DD}	Leave open.
RESET	–	2	PORTV _{DD}	–
X1	–	–	–	–
X2	–	–	–	–
XT1	–	–	–	Connect to GND0 to GND2 via a resistor.
XT2	–	–	–	Leave open.
V _{PP} ^{Note 2}	–	–	–	Connect to GND0 to GND2.
IC ^{Note 3}	–	–	–	Connect directly to GND0 to GND2.
CPUREG	–	–	–	–
V _{DD0}	–	–	–	–
GND0 to GND2	–	–	–	–
ADCV _{DD}	–	–	–	–
ADCGND	–	–	–	–
PORTV _{DD}	–	–	–	–
PORTGND	–	–	–	–

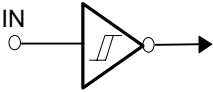
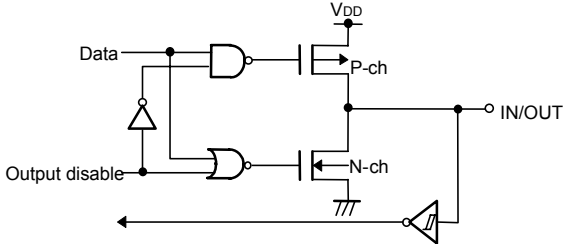
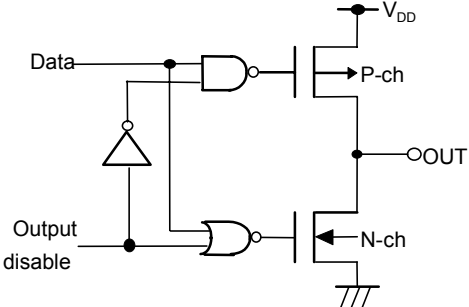
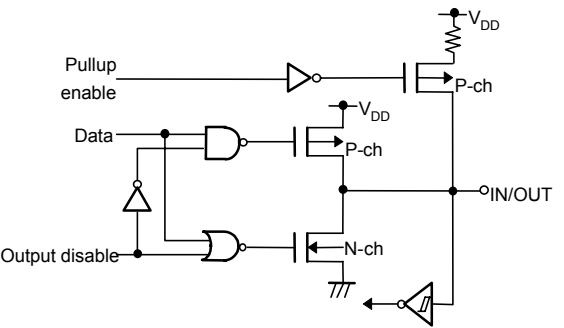
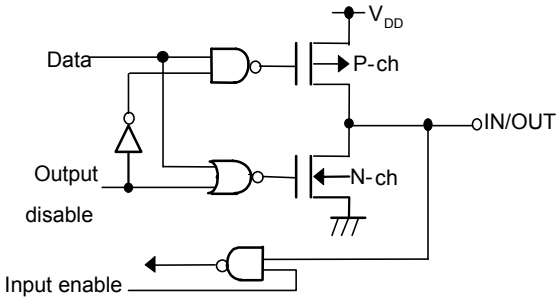
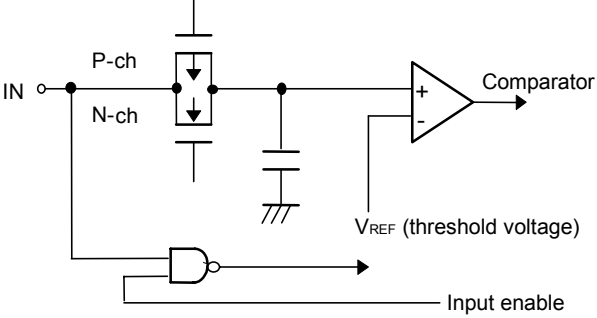
Notes 1. μ PD703079Y, 70F3079Y

2. μ PD70F3079Y

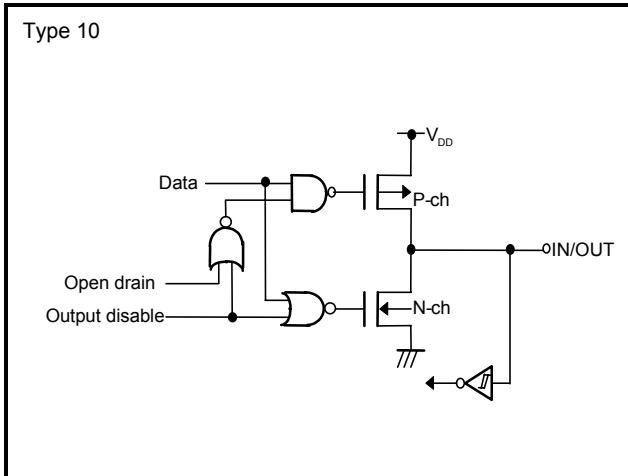
3. μ PD703078Y, 703079Y

2.5 I/O Circuit of Pins

(1/2)

<p>Type 2</p>  <p>Schmitt-triggered input with hysteresis characteristics</p>	<p>Type 8</p> 
<p>Type 4</p>  <p>Push-pull output that can be set for high impedance output (both P-ch and N-ch off).</p>	<p>Type 8-A</p> 
<p>Type 5</p> 	<p>Type 9</p> 

(2/2)



CHAPTER 3 CPU FUNCTIONS

The CPU of the V850/SF1 is based on RISC architecture and executes most instructions in one clock cycle by using a 5-stage pipeline.

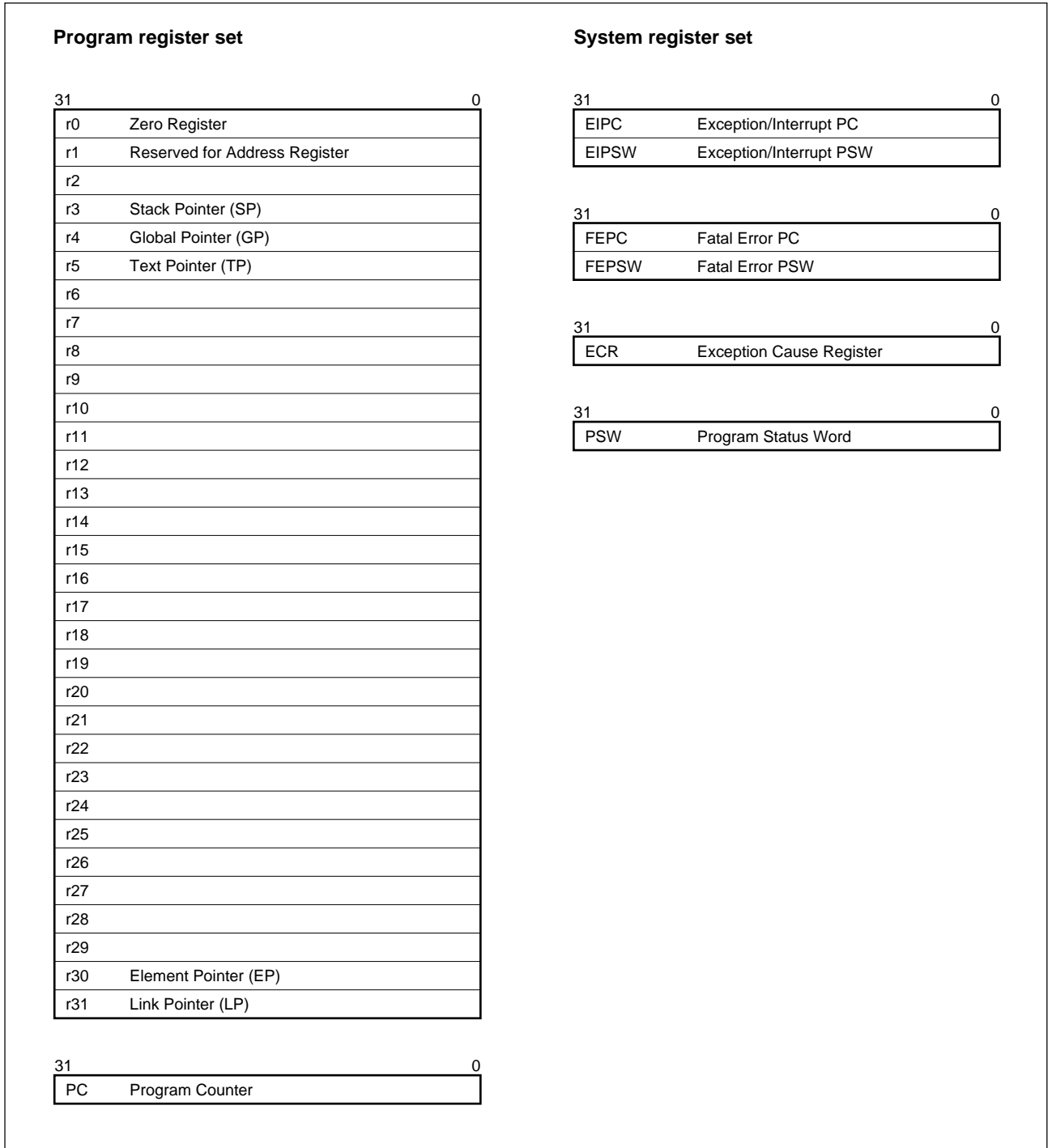
3.1 Features

- Minimum instruction execution time: 62.5 ns (@ 16 MHz internal operation)
- Address space: 16 MB linear
- Thirty-two 32-bit general-purpose registers
- Internal 32-bit architecture
- Five-stage pipeline control
- Multiplication/division instructions
- Saturated operation instructions
- One-clock 32-bit shift instruction
- Load/store instructions with long/short format
- Four types of bit manipulation instructions
 - SET1
 - CLR1
 - NOT1
 - TST1

3.2 CPU Register Set

The CPU registers in the V850/SF1 can be classified into two categories: a general-purpose program register set and a dedicated system register set. All the registers are 32 bits wide. For details, refer to **V850 Family User's Manual Architecture**.

Figure 3-1. CPU Register Set



3.2.1 Program register set

The program register set includes general-purpose registers and a program counter.

(1) General-purpose registers

Thirty-two general-purpose registers, r0 to r31, are available. Any of these registers can be used as a data variable or address variable.

However, r0 and r30 are implicitly used by instructions, and care must be exercised when using these registers. Also, r1, r3 to r5, and r31 are implicitly used by the assembler and C compiler. Therefore, before using these registers, their contents must be saved so that they are not lost. The contents must be restored to the registers after the registers have been used. r2 may be used by the real-time OS. r2 can be used as a variable register when the real-time OS that is used does not use r2.

Table 3-1. Program Registers

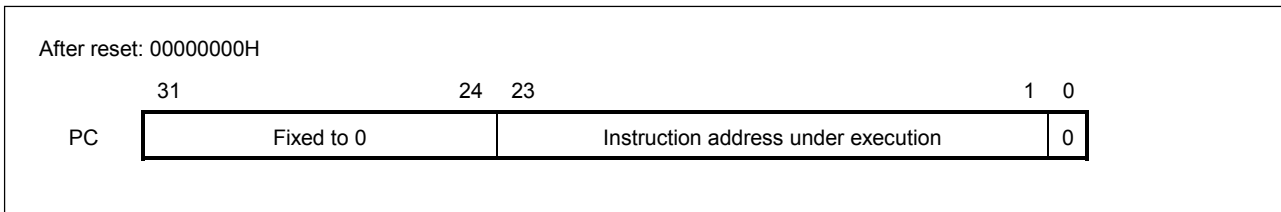
Name	Usage	Operation
r0	Zero register	Always holds 0
r1	Assembler-reserved register	Working register for generating 32-bit immediate
r2	Address/data variable register (when r2 is not used by the real-time OS being used)	
r3	Stack pointer	Used to generate stack frame when function is called
r4	Global pointer	Used to access global variable in data area
r5	Text pointer	Register to indicate the start of the text area ^{Note}
r6 to r29	Address/data variable registers	
r30	Element pointer	Base pointer when memory is accessed
r31	Link pointer	Used by compiler when calling function
PC	Program counter	Holds instruction address during program execution

Note Area in which program code is mapped.

(2) Program counter (PC)

This register holds the address of the instruction under execution. The lower 24 bits of this register are valid, and bits 31 to 24 are fixed to 0. If a carry occurs from bit 23 to 24, it is ignored.

Bit 0 is fixed to 0, and branching to an odd address cannot be performed.



3.2.2 System register set

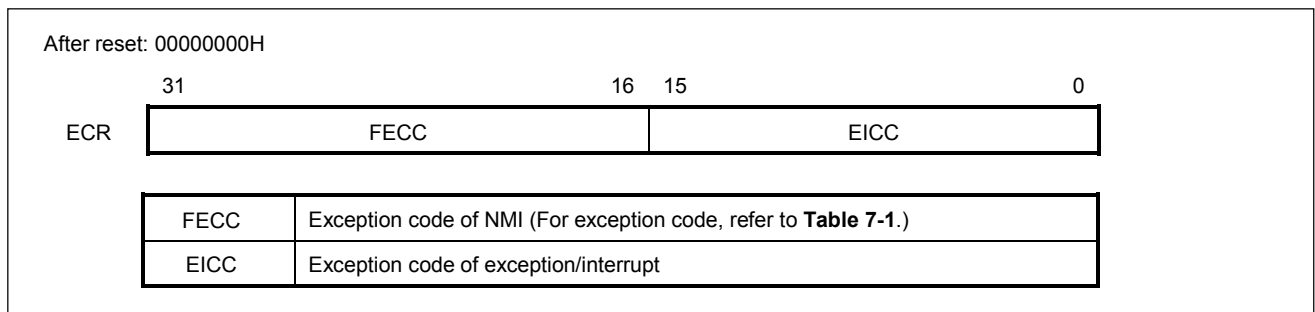
The system registers control the status of the CPU and hold interrupt information.

Table 3-2. System Register Numbers

No.	System Register Name	Usage	Operation
0	EIPC	Interrupt status saving registers	These registers save the PC and PSW when an exception or interrupt occurs. Because only one set of these registers is available, their contents must be saved when multiple interrupts are enabled.
1	EIPSW		
2	FEPC	NMI status saving registers	These registers save PC and PSW when NMI occurs.
3	FEPSW		
4	ECR	Interrupt source register	If exception, maskable interrupt, or NMI occurs, this register will hold information referencing the interrupt source. The higher 16 bits of this register are called FECC, to which the exception code of NMI is set. The lower 16 bits are called EICC, to which the exception code of exception/interrupt is set.
5	PSW	Program status word	The program status word is a collection of flags that indicate the program status (instruction execution result) and CPU status.
6 to 31	Reserved		

To read/write these system registers, specify a system register number, indicated by the system register load/store instruction (LDSR or STSR instruction).

(1) Interrupt source register (ECR)



(2) Program status word (PSW)



Bit position	Bit name	Function
31 to 8	RFU	Reserved field (fixed to 0).
7	NP	Indicates that non-maskable interrupt (NMI) servicing is in progress. This flag is set when an NMI is acknowledged, and disables multiple interrupts. 0: NMI servicing not under execution. 1: NMI servicing under execution.
6	EP	Indicates that exception processing is in progress. This flag is set when an exception is generated. Moreover, interrupt requests can be acknowledged when this bit is set. 0: Exception processing not under execution. 1: Exception processing under execution.
5	ID	Indicates whether a maskable interrupt request can be acknowledged or not. 0: Interrupt enabled. 1: Interrupt disabled.
4	SAT ^{Note}	Indicates that the operation result of a saturated operation processing instruction is saturated due to overflow. Due to the cumulative flag, if the operation result is saturated by the saturation operation instruction, this bit is set (1), but is not cleared (0) even if the operation results of subsequent instructions are not saturated. To clear (0) this bit, load data in PSW. Note that in a general arithmetic operation, this bit is neither set (1) nor cleared (0). 0: Not saturated. 1: Saturated.
3	CY	This flag is set if a carry or borrow occurs as the result of an operation (if a carry or borrow does not occur, it is reset). 0: Carry or borrow does not occur. 1: Carry or borrow occurs.
2	OV ^{Note}	This flag is set if an overflow occurs during operation (if an overflow does not occur, it is reset). 0: Overflow does not occur. 1: Overflow occurs.
1	S ^{Note}	This flag is set if the result of an operation is negative (it is reset if the result is positive). 0: The operation result was positive or 0. 1: The operation result was negative.
0	Z	This flag is set if the result of an operation is zero (if the result is not zero, it is reset). 0: The operation result was not 0. 1: The operation result was 0.

Note The result of a saturation-processed operation is determined by the contents of the OV and S flags in the saturation operation. Simply setting the OV flag (1) will set the SAT flag (1) in a saturation operation.

Status of operation result	Flag status			Saturation-processed operation result
	SAT	OV	S	
Maximum positive value exceeded	1	1	0	7FFFFFFFH
Maximum negative value exceeded	1	1	1	80000000H
Positive (not exceeding the maximum)	Retains the value before operation	0	0	Operation result itself
Negative (not exceeding the maximum)			1	

3.3 Operation Modes

The V850/SF1 has the following operation modes.

(1) Normal operation mode (single-chip mode)

After the system has been released from the reset status, the pins related to the bus interface are set to port mode, execution branches to the reset entry address of the internal ROM, and instruction processing written in the internal ROM is started. However, external expansion mode, in which an external device is connected to external memory area, is enabled by setting in the memory expansion mode register (MM) via an instruction.

(2) Flash memory programming mode

This mode is provided only in the μ PD70F3079Y. The internal flash memory is programmable or erasable when the V_{PP} voltage is applied to the V_{PP} pin.

V_{PP}	Operation Mode
0	Normal operation mode
7.8 V	Flash memory programming mode
V_{DD}	Setting prohibited

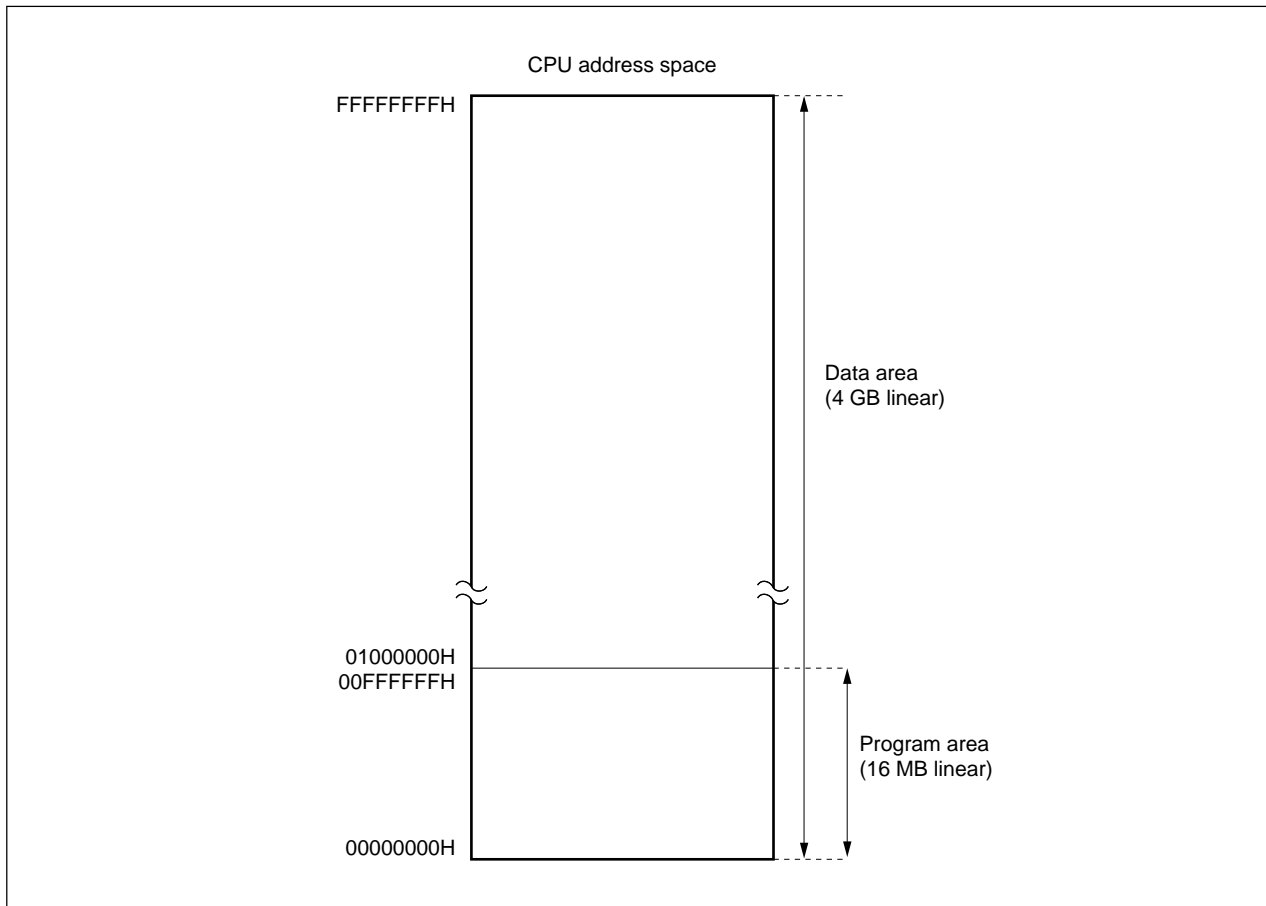
3.4 Address Space

3.4.1 CPU address space

The CPU of the V850/SF1 is of 32-bit architecture and supports up to 4 GB of linear address space (data space) during operand addressing (data access). When referencing instruction addresses, linear address space (program space) of up to 16 MB is supported.

The CPU address space is shown below.

Figure 3-2. CPU Address Space

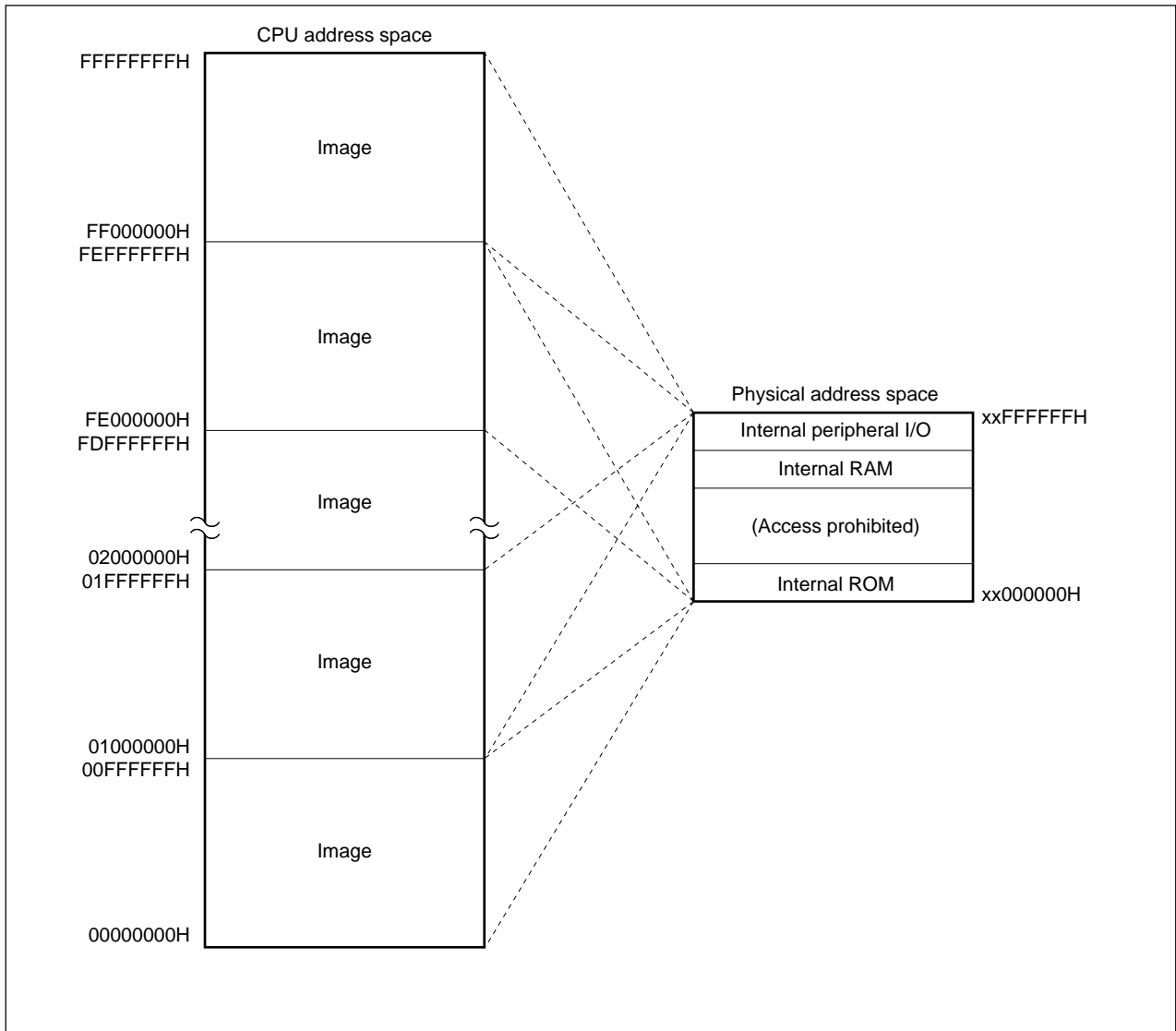


3.4.2 Images

A 16 MB physical address space is seen as 256 images in the 4 GB CPU address space. In other words, the same 16 MB physical address space is accessed regardless of the values of bits 31 to 24 of the CPU address. The images of the addressing space are shown below.

The physical address `xx000000H` can be seen as CPU address `00000000H`, and in addition, can be seen as addresses `01000000H`, `02000000H`, ... `FE000000H`, `FF000000H`. This is because the higher 8 bits of a 32-bit CPU address are ignored and the CPU address is only accessed as a 24-bit physical address.

Figure 3-3. Images on Address Space



3.4.3 Wraparound of CPU address space

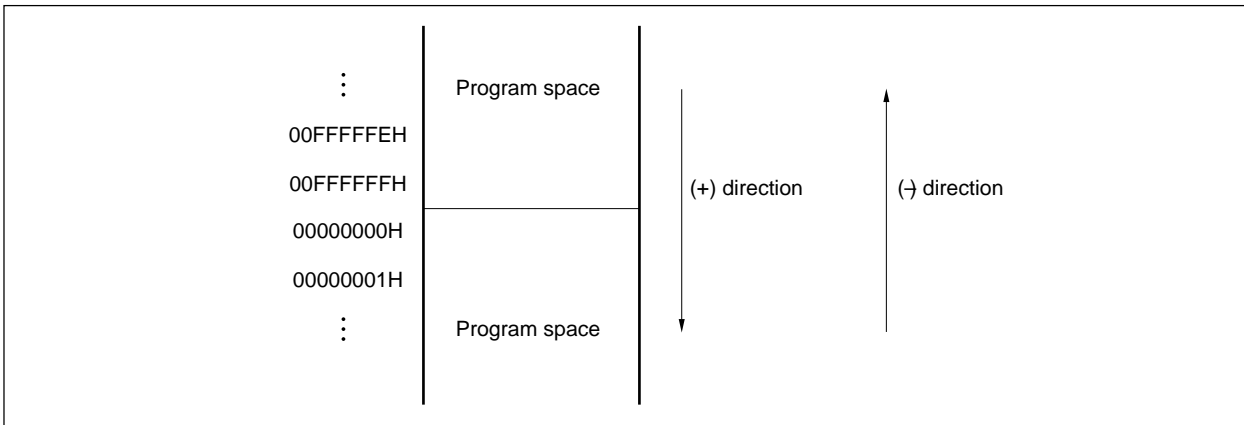
(1) Program space

Of the 32 bits of the PC (program counter), the higher 8 bits are fixed to 0, and only the lower 24 bits are valid. Even if a carry or borrow occurs from bit 23 to 24 as a result of a branch address calculation, the higher 8 bits ignore the carry or borrow and remain 0.

Therefore, the lower-limit address of the program space, address 00000000H, and the upper-limit address 00FFFFFFH are contiguous addresses, and the program space is wrapped around at the boundary of these addresses.

Caution No instruction can be fetched from the 4 KB area of 00FFF000H to 00FFFFFFH because this area is defined as peripheral I/O area. Therefore, do not execute any branch operation instructions in which the destination address will reside in any part of this area.

Figure 3-4. Program Space

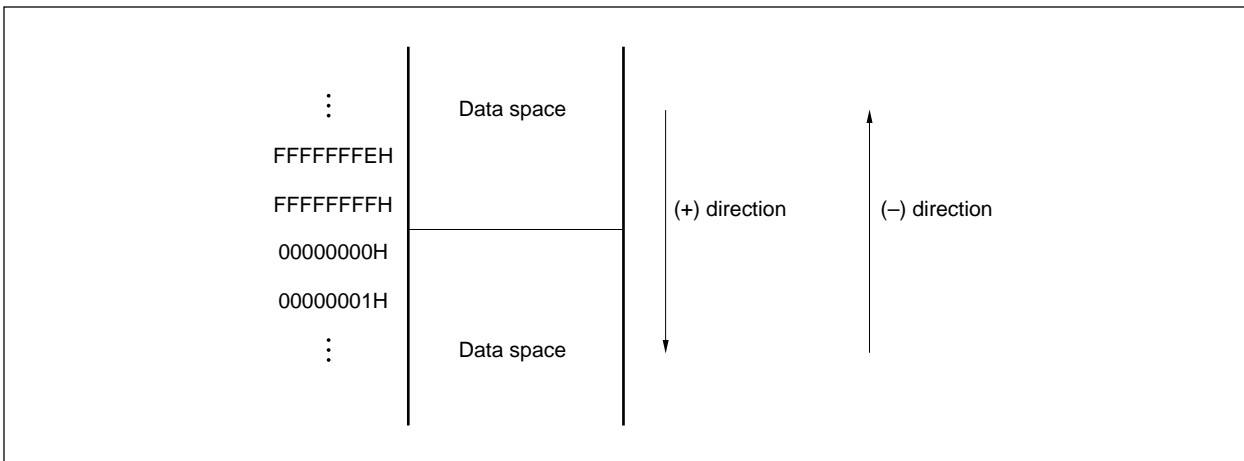


(2) Data space

The result of an operand address calculation that exceeds 32 bits is ignored.

Therefore, the lower-limit address of the program space, address 00000000H, and the upper-limit address FFFFFFFFH are contiguous addresses, and the data space is wrapped around at the boundary of these addresses.

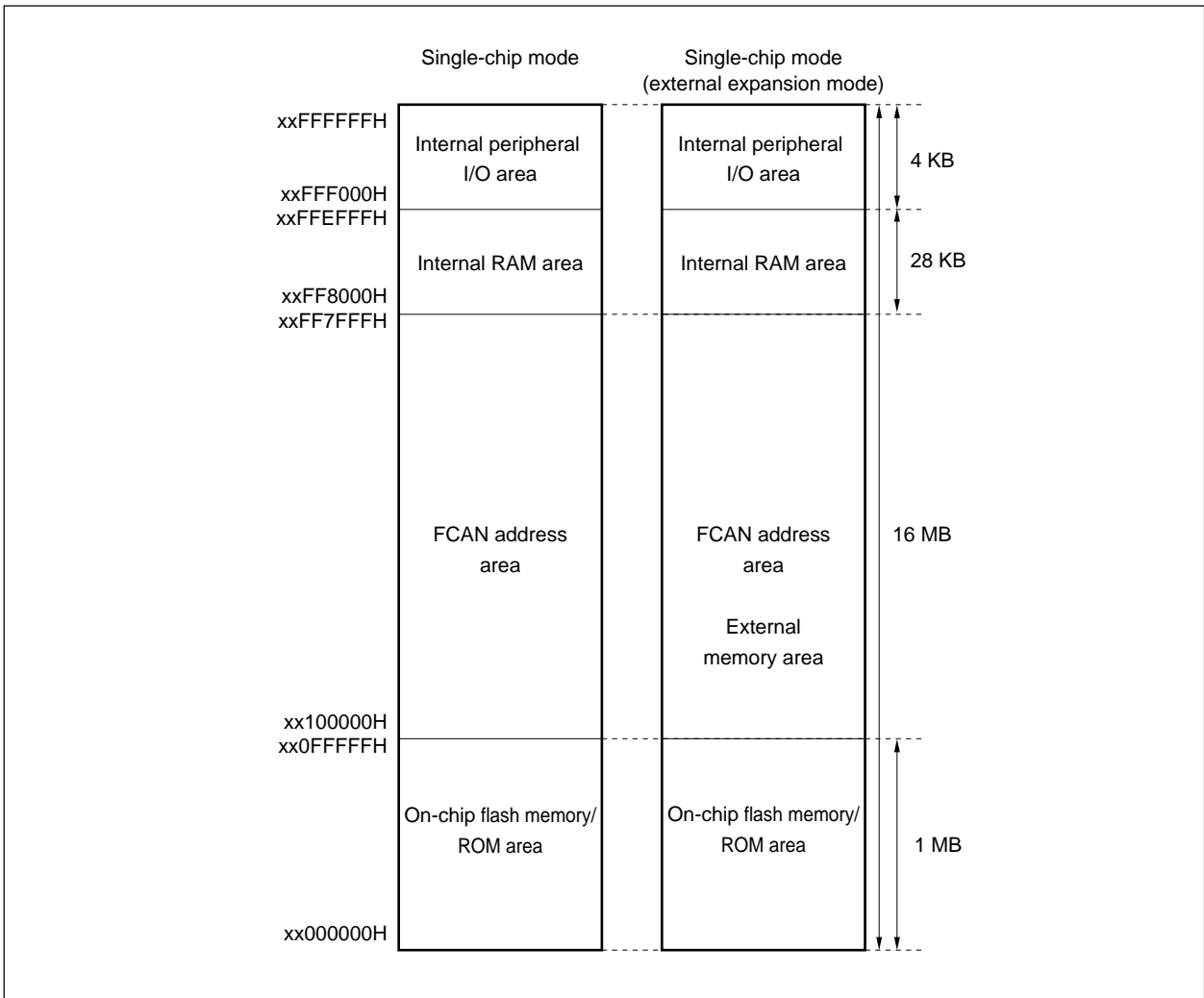
Figure 3-5. Data Space



3.4.4 Memory map

The V850/SF1 reserves areas as shown below.

Figure 3-6. Memory Map



3.4.5 Area

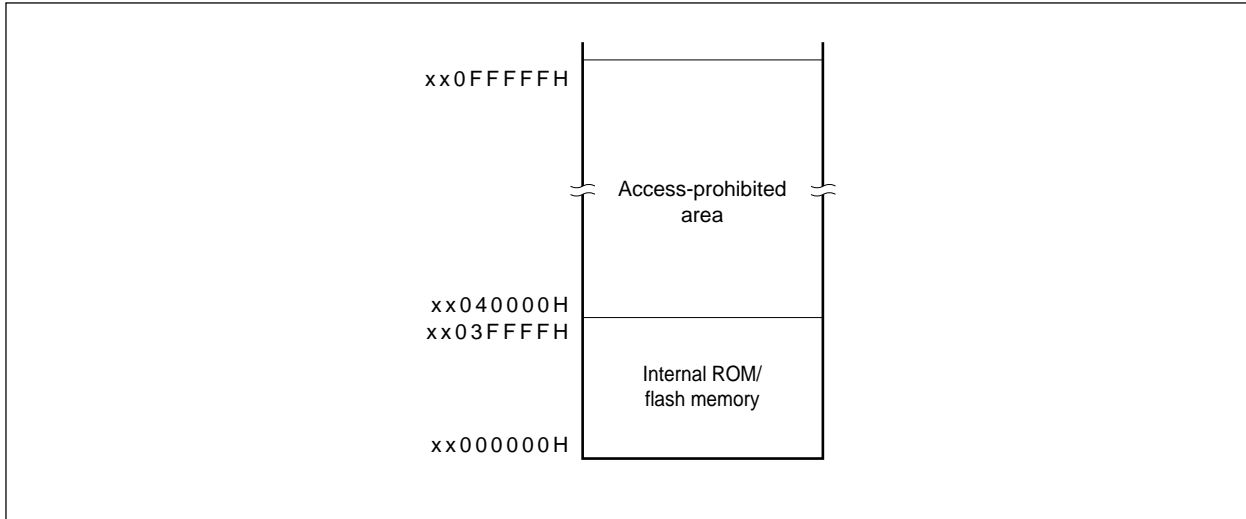
(1) Internal ROM/flash memory area

A maximum of 1 MB is reserved for the internal ROM/flash memory area.

256 KB are available at addresses xx000000H to xx03FFFFH.

Addresses xx040000H to xx0FFFFFFH are an access-prohibited area.

Figure 3-7. Internal ROM/Flash Memory Area



Interrupt/exception table

The V850/SF1 increases the interrupt response speed by assigning handler addresses corresponding to interrupts/exceptions.

The collection of these handler addresses is called an interrupt/exception table, which is located in the internal ROM area. When an interrupt/exception request is acknowledged, execution jumps to the handler address, and the program written at that memory address is executed. The sources of interrupts/exceptions, and the corresponding addresses are shown below.

Table 3-3. Interrupt/Exception Table

Start Address of Interrupt/Exception Table	Interrupt/Exception Source	Start Address of Interrupt/Exception Table	Interrupt/Exception Source
00000000H	RESET	000001B0H	INTTM4
00000010H	NMI	000001C0H	INTTM5
00000020H	INTWDT	000001D0H	INTWTM
00000040H	TRAP0n (n = 0 to F)	000001E0H	INTWTNI
00000050H	TRAP1n (n = 0 to F)	000001F0H	INTIIC0/INTCSI0
00000060H	ILGOP	00000200H	INTSER0
00000080H	INTWDTM	00000210H	INTSR0/INTCSI1
00000090H	INTP0	00000220H	INTST0
000000A0H	INTP1	00000230H	INTKR
000000B0H	INTP2	00000240H	INTCE1
000000C0H	INTP3	00000250H	INTCR1
000000D0H	INTP4	00000260H	INTCT1
000000E0H	INTP5	00000270H	INTICME
000000F0H	INTP6	00000280H	INTTM6
00000100H	INTCSI4	00000290H	INTTM70
00000110H	INTAD	000002A0H	INTTM71
00000120H	INTDMA0	000002B0H	INTSER1
00000130H	INTDMA1	000002C0H	INTSR1/INTCSI3
00000140H	INTDMA2	000002D0H	INTST1
00000150H	INTTM00	000002E0H	INTDMA3
00000160H	INTTM01	000002F0H	INTDMA4
00000170H	INTTM10	00000300H	INTDMA5
00000180H	INTTM11	00000310H	INTCE2 ^{Note}
00000190H	INTTM2	00000320H	INTCR2 ^{Note}
000001A0H	INTTM3	00000330H	INTCT2 ^{Note}

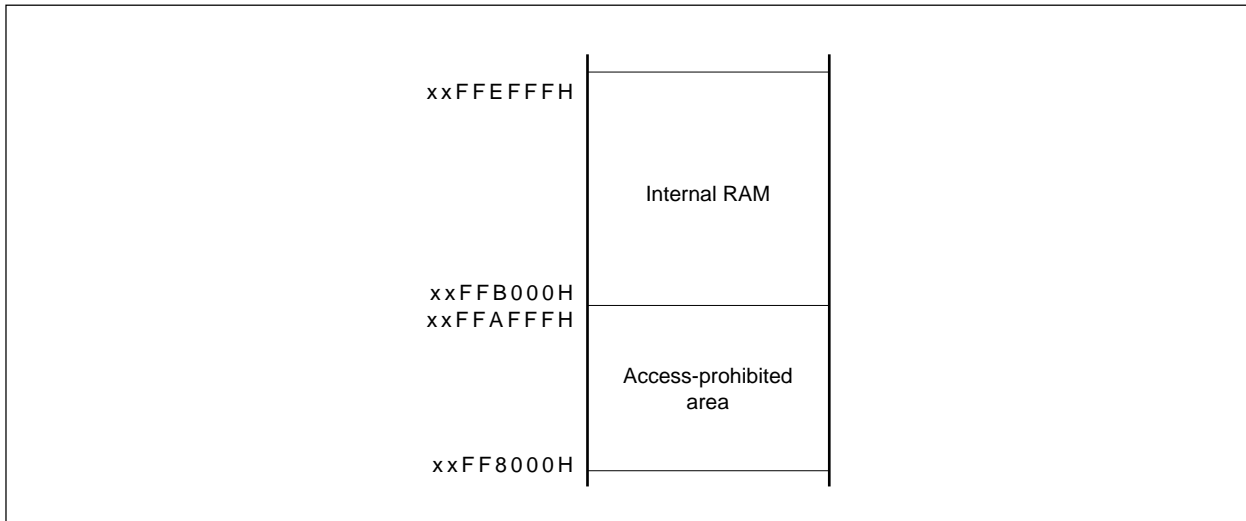
Note Available only for the μ PD703079Y and 70F3079Y.

(2) Internal RAM area

A maximum of 28 KB is reserved for the internal RAM area.

16 KB are available at addresses `xxFFB000H` to `xxFFEFFFH`.

Addresses `xxFF8000H` to `xxFFAFFFH` are an access-prohibited area.

Figure 3-8. Internal RAM Area

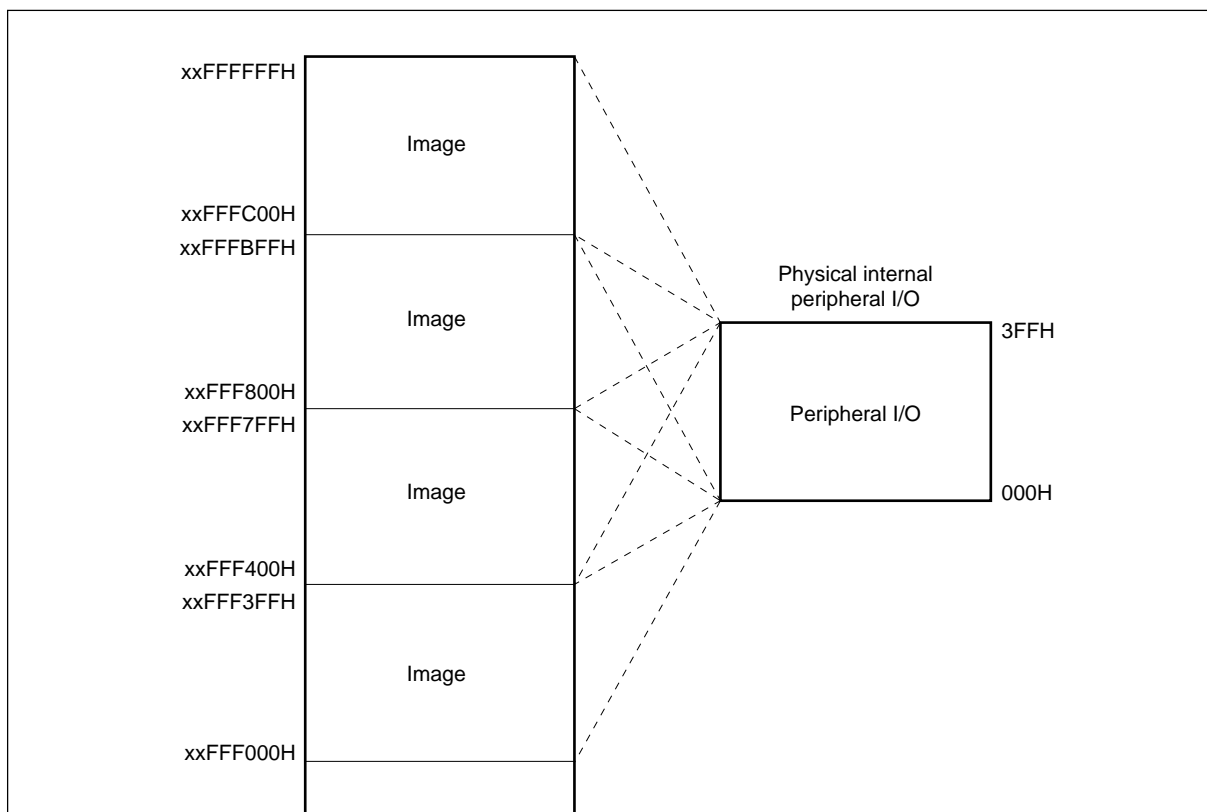
(3) Internal peripheral I/O area

The 4 KB area of addresses FFF000H to FFFFFFFH is reserved as an internal peripheral I/O area.

In the V850/SF1, the 1 KB area of addresses FFF000H to FFF3FFH is provided as a physical internal peripheral I/O area, and its image can be seen on the rest of the area (FFF400H to FFFFFFFH).

Peripheral I/O registers associated with functions such as operation mode specification and state monitoring for the internal peripherals are all memory-mapped to the internal peripheral I/O area. Program fetches are not allowed in this area.

Figure 3-9. Internal Peripheral I/O Area



- Cautions**
1. The least significant bit of an address is not decoded. If an odd address ($2n + 1$) in the peripheral I/O area is referenced (accessed in byte units), the register at an even address ($2n$) will be accessed.
 2. If a register that can be accessed in byte units is accessed in halfword units, the higher 8 bits become undefined, if the access is a read operation. If a write access is made, only the data in the lower 8 bits is written to the register.
 3. If a register with n address that can be accessed only in halfword units is accessed in word units, the operation is replaced with two halfword operations. The first operation (lower 16 bits) accesses the register with n address and the second operation (higher 16 bits) accesses the register with $n + 2$ address.
 4. If a register with n address that can be accessed in word units is accessed in word units, the operation is replaced with two halfword operations. The first operation (lower 16 bits) accesses the register with n address and the second operation (higher 16 bits) accesses the register with $n + 2$ address.
 5. Addresses that are not defined as registers are reserved for future expansion. If these addresses are accessed, the operation is undefined and not guaranteed.

(4) External memory

The V850/SF1 can use an area of up to 16 MB (xx100000H to xxFF7FFFH) for external memory area (in single-chip mode: external expansion).

64 KB, 256 KB, 1 MB, or 4 MB of physical external memory can be allocated when the external expansion mode is specified. In the area of other than the physical external memory, the image of the physical external memory can be seen.

The internal RAM area and internal peripheral I/O area are not subject to external memory access.

Caution Addresses xxnFF800H to xxnFFFFFFH (n = 3, 7, B) constitute an FCAN address area and are therefore access-prohibited.

Figure 3-10. External Memory Area (When Expanded to 64 KB, 256 KB, or 1 MB)

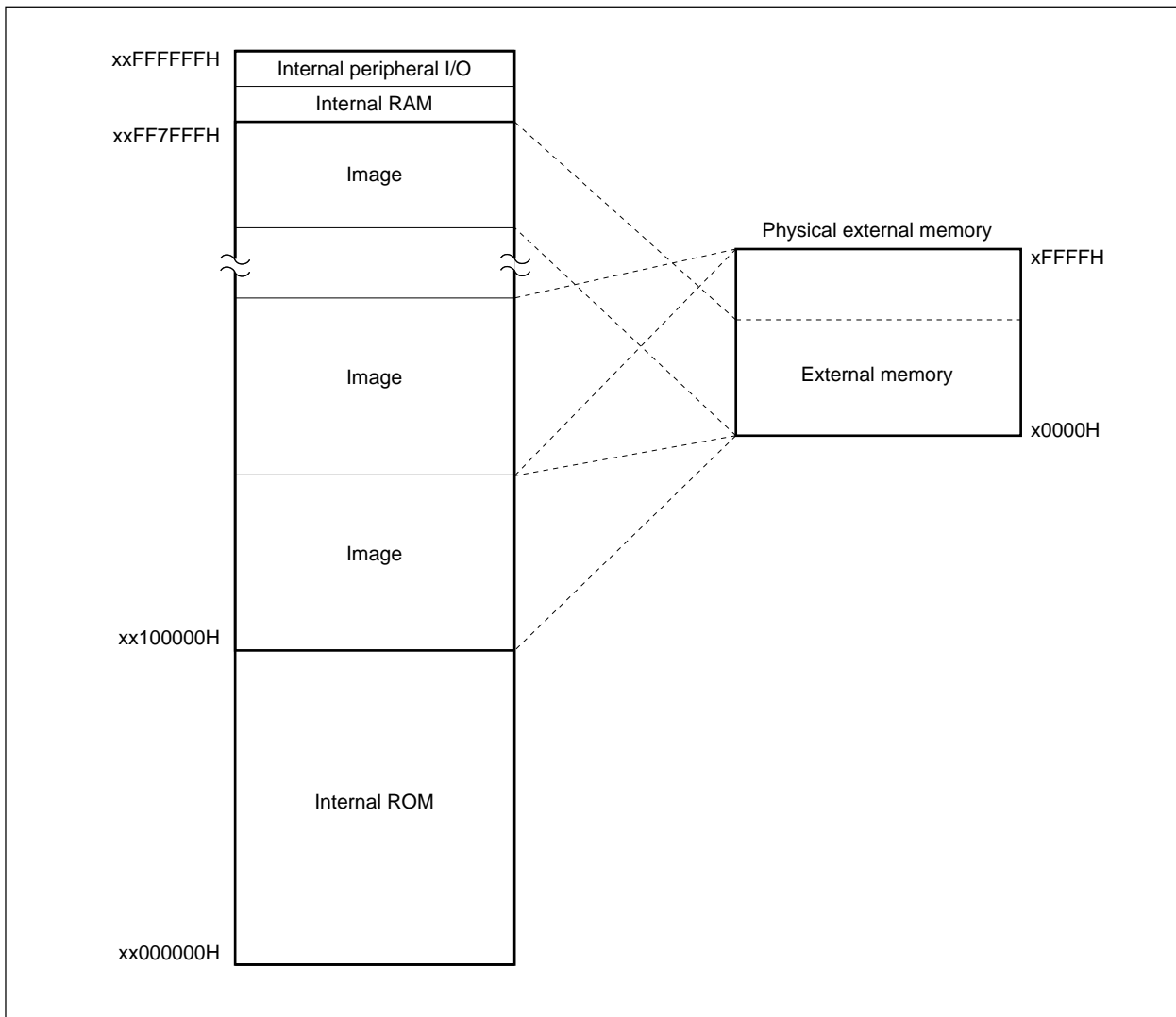
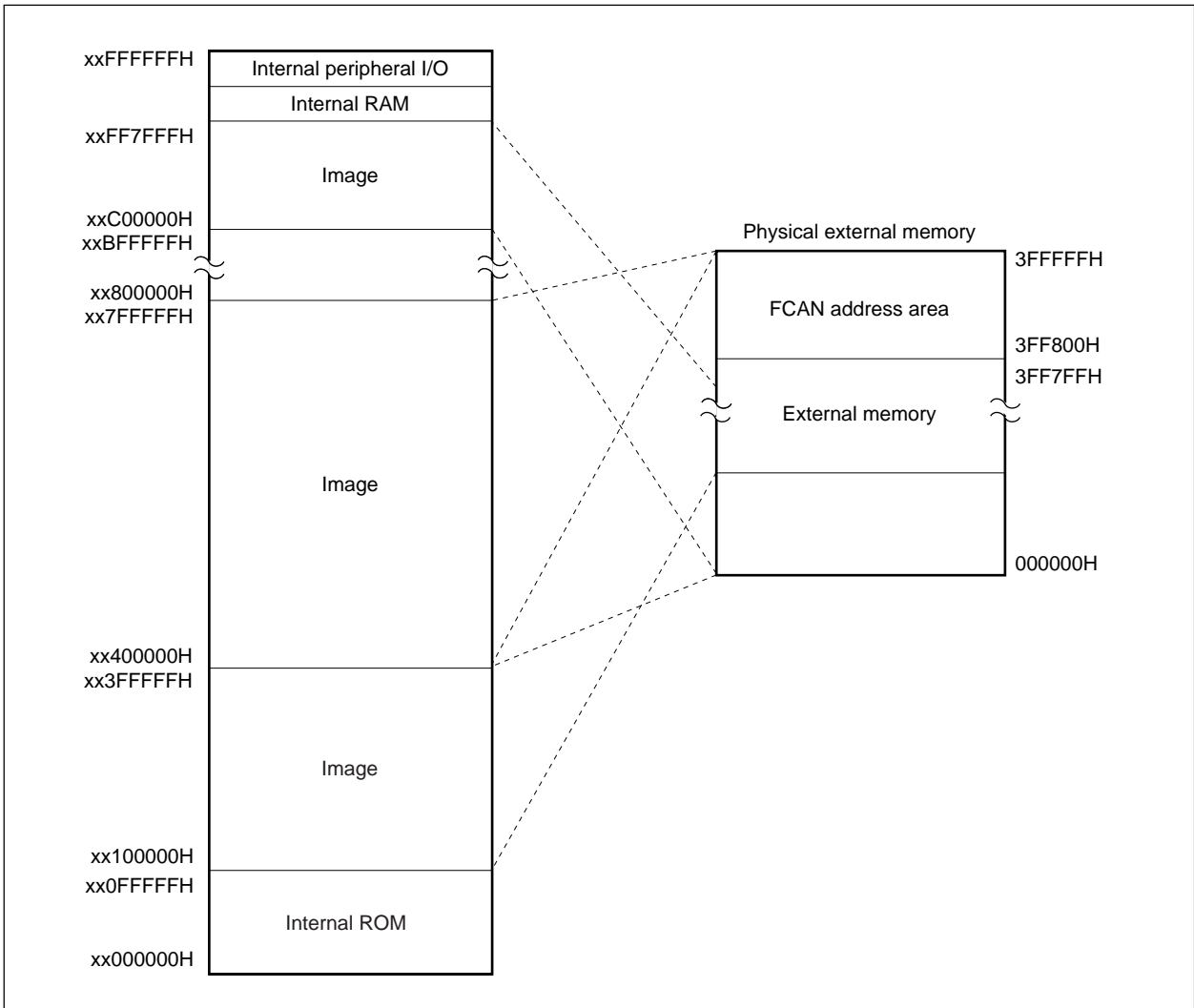


Figure 3-11. External Memory Area (When Expanded to 4 MB)



3.4.6 External expansion mode

The V850/SF1 allows external devices to be connected to the external memory space by using the pins of ports 4, 5, 6, and 9. To connect an external device, the port pins must be set to the external expansion mode by using the memory expansion mode register (MM).

Because the V850/SF1 is fixed to single-chip mode in the normal operation mode, the port alternate-function pins are in the port mode, and therefore the external memory cannot be used. When the external memory is used (external expansion mode), set the MM register by program.

(1) Memory expansion mode register (MM)

This register sets the mode of each pin of ports 4, 5, 6, and 9. In the external expansion mode, an external device can be connected to an external memory area of up to 4 MB. However, the external device cannot be connected to the internal RAM area, internal peripheral I/O area, and internal ROM area in the single-chip mode (and even if the external device is connected physically, it cannot be accessed).

The MM register can be read/written in 8- or 1-bit units. However, bits 4 to 7 are fixed to 0.

After reset: 00H R/W Address: FFFFF04CH

Symbol	7	6	5	4	3	2	1	0
MM	0	0	0	0	MM3	MM2	MM1	MM0

MM3	P95 and P96 operation modes
0	Port mode
1	External expansion mode (HLDAK: P95, HLDRQ: P96) ^{Note}

MM2	MM1	MM0	Address space	Port 4	Port 5	Port 6	Port 9
0	0	0	–	Port mode			
0	1	1	64 KB expansion mode	AD0 to AD7	AD8 to AD15	A16, A17	$\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$, $\overline{\text{R/W}}$, $\overline{\text{DSTB}}$, ASTB
1	0	0	256 KB expansion mode				
1	0	1	1 MB expansion mode				
1	1	×	4 MB expansion mode				
Other than above				RFU (reserved)			

Note Before switching to the external expansion mode, be sure to set P95 and P96 of Port 9 (P9) to 1.

Remark For details of the operation of each port pin, refer to **2.3 Description of Pin Functions**.

3.4.7 Recommended use of address space

The architecture of the V850/SF1 requires that a register that serves as a pointer be secured for address generation in operand data accessing for data space. The address in this pointer register ± 32 KB can be accessed directly from an instruction. However, the general-purpose registers that can be used as a pointer register are limited. Therefore, by minimizing the deterioration of the address calculation performance when changing the pointer value, the number of usable general-purpose registers for handling variables is maximized, and the program size can be saved because instructions for calculating pointer addresses are not required.

To enhance the efficiency of using the pointer in connection with the memory maps of the V850/SF1, the following points are recommended:

(1) Program space

Of the 32 bits of the PC (program counter), the higher 8 bits are fixed to 0, and only the lower 24 bits are valid. Therefore, a continuous 16 MB space, starting from address 00000000H, unconditionally corresponds to the memory map of the program space.

(2) Data space

For the efficient use of resources that utilize the wraparound feature of the data space, the continuous 8 MB address spaces 00000000H to 007FFFFFFH and FF800000H to FFFFFFFFH of the 4 GB CPU are used as the data space. With the V850/SF1, a 16 MB physical address space is seen as 256 images in the 4 GB CPU address space. The most significant bit (bit 23) of this 24-bit address is assigned as address sign-extended to 32 bits.

(a) Application of wraparound

For example, when R = r0 (zero register) is specified for the LD/ST disp16 [R] instruction, an addressing range of 00000000H ± 32 KB can be referenced with the sign-extended disp16. All resources including on-chip hardware can be accessed with one pointer.

The zero register (r0) is a register set to 0 by hardware, and eliminates the need for additional registers for the pointer.

Figure 3-12. Application of Wraparound

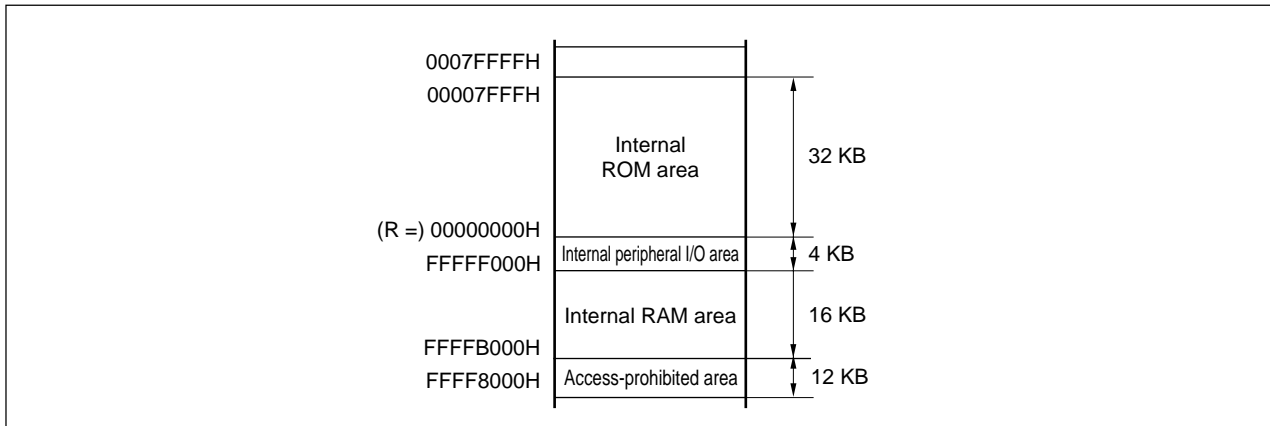
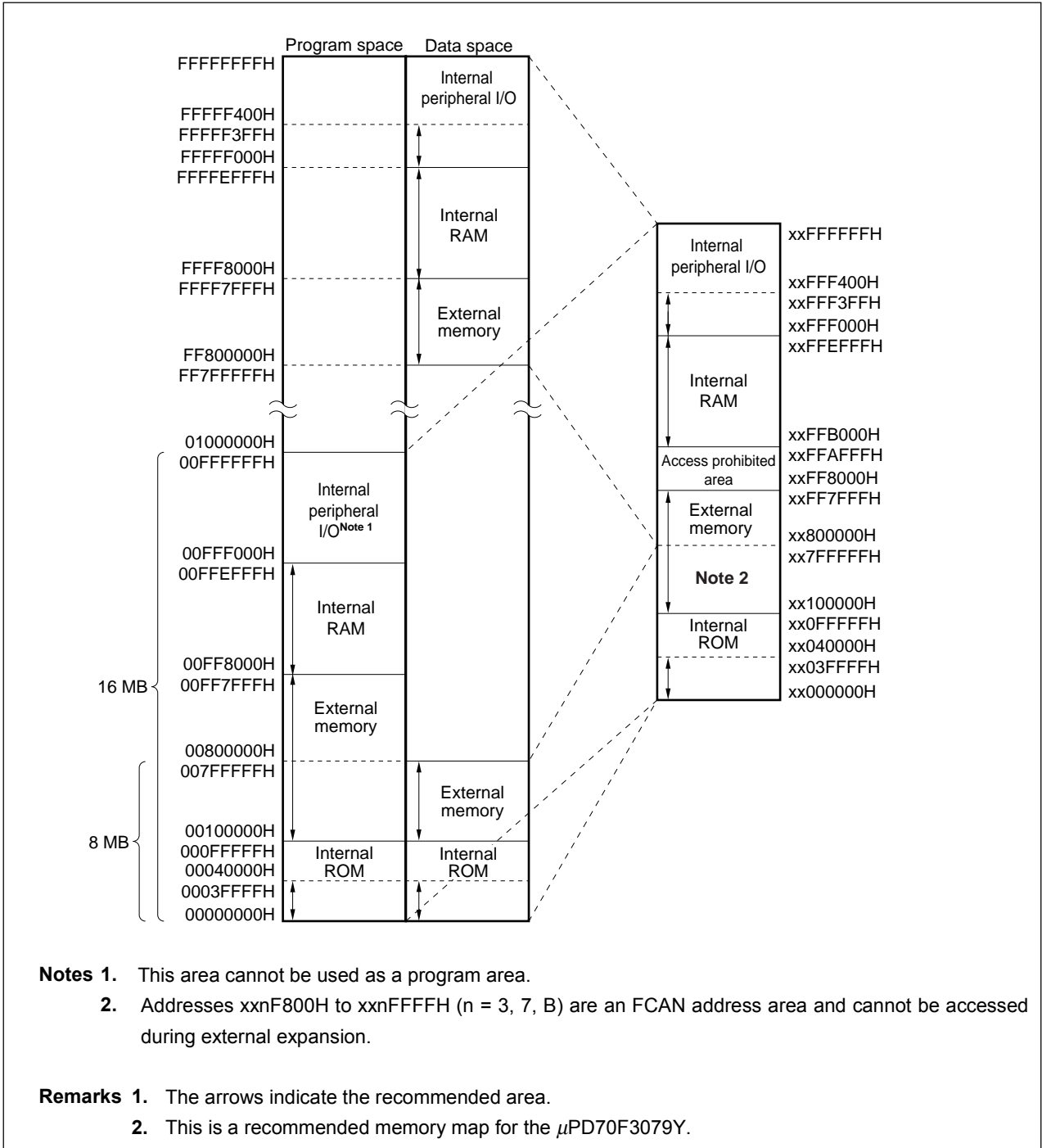


Figure 3-13. Recommended Memory Map (Flash Memory Version)



3.4.8 Peripheral I/O registers

(1/7)

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bits	16 Bits	32 Bits	
FFFFF000H	Port 0	P0	R/W	√	√			00H ^{Note}
FFFFF002H	Port 1	P1		√	√			
FFFFF004H	Port 2	P2		√	√			
FFFFF006H	Port 3	P3		√	√			
FFFFF008H	Port 4	P4		√	√			
FFFFF00AH	Port 5	P5		√	√			
FFFFF00CH	Port 6	P6		√	√			
FFFFF00EH	Port 7	P7	R	√	√			Undefined
FFFFF010H	Port 8	P8		√	√			
FFFFF012H	Port 9	P9	R/W	√	√			00H ^{Note}
FFFFF014H	Port 10	P10		√	√			
FFFFF016H	Port 11	P11		√	√			
FFFFF020H	Port 0 mode register	PM0		√	√			FFH
FFFFF022H	Port 1 mode register	PM1		√	√			3FH
FFFFF024H	Port 2 mode register	PM2		√	√			FFH
FFFFF026H	Port 3 mode register	PM3		√	√			
FFFFF028H	Port 4 mode register	PM4		√	√			
FFFFF02AH	Port 5 mode register	PM5		√	√			
FFFFF02CH	Port 6 mode register	PM6		√	√			3FH
FFFFF032H	Port 9 mode register	PM9		√	√			7FH
FFFFF034H	Port 10 mode register	PM10		√	√			FFH
FFFFF036H	Port 11 mode register	PM11		√	√			FFH
FFFFF040H	Port alternate function control register	PAC		√	√			00H
FFFFF04CH	Memory expansion mode register	MM		√	√			
FFFFF060H	Data wait control register	DWC				√		FFFFH
FFFFF062H	Bus cycle control register	BCC				√		AAAAH
FFFFF070H	Power save control register	PSC		√	√			C0H
FFFFF074H	Processor clock control register	PCC		√	√			03H
FFFFF078H	System status register	SYS		√	√			00H

Note Resetting initializes registers to input mode, so 00H cannot actually be read.

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset	
				1 Bit	8 Bits	16 Bits	32 Bits		
FFFFF07AH	POC status register	POCS	R		√			Held ^{Note 1}	
FFFFF07CH	VM45 control register	VM45C	R/W		√			00H	
FFFFF094H	Pull-up resistor option register 10	PU10		√	√				
FFFFF0A2H	Port 1 function register	PF1		√	√				
FFFFF0C0H	Rising edge specification register 0	EGP0		√	√				
FFFFF0C2H	Falling edge specification register 0	EGN0		√	√				
FFFFF0E4H	Timer clock selection register 30	TCL30			√				
FFFFF0E6H	Timer mode control register 30	TMC30		√	√				04H ^{Note 2}
FFFFF0EAH	16-bit counter 3	TM3		R			√		0000H
FFFFF0ECH	16-bit compare register 3	CR3	R/W			√			
FFFFF0EEH	Timer clock selection register 31	TLC31			√				00H
FFFFF100H	Interrupt control register	WDTIC		√	√				47H
FFFFF102H	Interrupt control register	PIC0		√	√				
FFFFF104H	Interrupt control register	PIC1		√	√				
FFFFF106H	Interrupt control register	PIC2		√	√				
FFFFF108H	Interrupt control register	PIC3		√	√				
FFFFF10AH	Interrupt control register	PIC4		√	√				
FFFFF10CH	Interrupt control register	PIC5		√	√				
FFFFF10EH	Interrupt control register	PIC6		√	√				
FFFFF110H	Interrupt control register	CSIC4		√	√				
FFFFF112H	Interrupt control register	ADIC		√	√				
FFFFF114H	Interrupt control register	DMAIC0		√	√				
FFFFF116H	Interrupt control register	DMAIC1		√	√				
FFFFF118H	Interrupt control register	DMAIC2		√	√				
FFFFF11AH	Interrupt control register	TMIC00		√	√				
FFFFF11CH	Interrupt control register	TMIC01		√	√				
FFFFF11EH	Interrupt control register	TMIC10		√	√				
FFFFF120H	Interrupt control register	TMIC11		√	√				
FFFFF122H	Interrupt control register	TMIC2		√	√				
FFFFF124H	Interrupt control register	TMIC3		√	√				
FFFFF126H	Interrupt control register	TMIC4		√	√				
FFFFF128H	Interrupt control register	TMIC5		√	√				
FFFFF12AH	Interrupt control register	WTNIC		√	√				
FFFFF12CH	Interrupt control register	WTNIC	√	√					
FFFFF12EH	Interrupt control register	CSIC0	√	√					

Notes 1. This value is 03H only after a power-on-clear reset. This cannot be reset by $\overline{\text{RESET}}$ signal input or watchdog timer.

2. Although the hardware status is initialized to 04H, 00H is read out if read.

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset	
				1 Bit	8 Bits	16 Bits	32 Bits		
FFFFF130H	Interrupt control register	SERIC0	R/W	√	√			47H	
FFFFF132H	Interrupt control register	CSIC1		√	√				
FFFFF134H	Interrupt control register	STIC0		√	√				
FFFFF136H	Interrupt control register	KRIC		√	√				
FFFFF138H	Interrupt control register	CANIC1		√	√				
FFFFF13AH	Interrupt control register	CANIC2		√	√				
FFFFF13CH	Interrupt control register	CANIC3		√	√				
FFFFF13EH	Interrupt control register	CANIC7		√	√				
FFFFF140H	Interrupt control register	TMIC6		√	√				
FFFFF142H	Interrupt control register	TMIC70		√	√				
FFFFF144H	Interrupt control register	TMIC71		√	√				
FFFFF146H	Interrupt control register	SERIC1		√	√				
FFFFF148H	Interrupt control register	CSIC3		√	√				
FFFFF14AH	Interrupt control register	STIC1		√	√				
FFFFF14CH	Interrupt control register	DMAIC3		√	√				
FFFFF14EH	Interrupt control register	DMAIC4		√	√				
FFFFF150H	Interrupt control register	DMAIC5		√	√				
FFFFF152H	Interrupt control register ^{Note}	CANIC4		√	√				
FFFFF154H	Interrupt control register ^{Note}	CANIC5		√	√				
FFFFF156H	Interrupt control register ^{Note}	CANIC6		√	√				
FFFFF166H	In-service priority register	ISPR	R	√	√			00H	
FFFFF170H	Command register	PRCMD	W		√			Undefined	
FFFFF180H	DMA peripheral I/O address register 0	DIOA0	R/W			√		Undefined	
FFFFF182H	DMA internal RAM address register 0	DRA0				√			
FFFFF184H	DMA byte count register 0	DBC0			√				
FFFFF186H	DMA channel control register 0	DCHC0		√	√				00H
FFFFF190H	DMA peripheral I/O address register 1	DIOA1				√			Undefined
FFFFF192H	DMA internal RAM address register 1	DRA1				√			
FFFFF194H	DMA byte count register 1	DBC1			√				
FFFFF196H	DMA channel control register 1	DCHC1		√	√				00H
FFFFF1A0H	DMA peripheral I/O address register 2	DIOA2				√			Undefined
FFFFF1A2H	DMA internal RAM address register 2	DRA2				√			
FFFFF1A4H	DMA byte count register 2	DBC2			√				
FFFFF1A6H	DMA channel control register 2	DCHC2		√	√				00H
FFFFF1B0H	DMA peripheral I/O address register 3	DIOA3				√			Undefined
FFFFF1B2H	DMA internal RAM address register 3	DRA3				√			

Note Available only for the μ PD703079Y and 70F3079Y.

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bits	16 Bits	32 Bits	
FFFFF1B4H	DMA byte count register 3	DBC3	R/W		√			Undefined
FFFFF1B6H	DMA channel control register 3	DCHC3		√	√			00H
FFFFF1C0H	DMA peripheral I/O address register 4	DIOA4				√		Undefined
FFFFF1C2H	DMA internal RAM address register 4	DRA4				√		
FFFFF1C4H	DMA byte count register 4	DBC4			√			00H
FFFFF1C6H	DMA channel control register 4	DCHC4		√	√			
FFFFF1D0H	DMA peripheral I/O address register 5	DIOA5				√		Undefined
FFFFF1D2H	DMA internal RAM address register 5	DRA5				√		
FFFFF1D4H	DMA byte count register 5	DBC5			√			00H
FFFFF1D6H	DMA channel control register 5	DCHC5		√	√			
FFFFF200H	16-bit timer register 0	TM0	R			√		0000H
FFFFF202H	16-bit capture/compare register 00	CR00	Note			√		
FFFFF204H	16-bit capture/compare register 01	CR01	Note			√		
FFFFF206H	Prescaler mode register 00	PRM00	R/W		√			00H
FFFFF208H	16-bit timer mode control register 0	TMC0		√	√			
FFFFF20AH	Capture/compare control register 0	CRC0		√	√			
FFFFF20CH	Timer output control register 0	TOC0		√	√			
FFFFF20EH	Prescaler mode register 01	PRM01			√			
FFFFF210H	16-bit timer register 1	TM1	R			√		0000H
FFFFF212H	16-bit capture/compare register 10	CR10	Note			√		
FFFFF214H	16-bit capture/compare register 11	CR11	Note			√		
FFFFF216H	Prescaler mode register 10	PRM10	R/W		√			00H
FFFFF218H	16-bit timer mode control register 1	TMC1		√	√			
FFFFF21AH	Capture/compare control register 1	CRC1		√	√			
FFFFF21CH	Timer output control register 1	TOC1		√	√			
FFFFF21EH	Prescaler mode register 11	PRM11			√			
FFFFF244H	Timer clock select register 20	TCL20			√			
FFFFF246H	Timer mode control register 20	TMC2		√	√			
FFFFF24AH	16-bit counter 2	TM2	R			√		0000H
FFFFF24CH	16-bit compare register 2	CR2	R/W			√		
FFFFF24EH	Timer clock selection register 21	TCL21			√			00H

Note In compare mode: R/W

In capture mode: R

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bits	16 Bits	32 Bits	
FFFFF264H	Timer clock selection register 40	TCL40	R/W		√			00H
FFFFF266H	Timer mode control register 40	TMC40		√	√			04H ^{Note}
FFFFF26AH	16-bit counter 4	TM4	R			√		0000H
FFFFF26CH	16-bit compare register 4	CR4	R/W			√		
FFFFF26EH	Timer clock selection register 41	TCL41			√			00H
FFFFF284H	Timer clock selection register 60	TCL60			√			
FFFFF286H	Timer mode control register 60	TMC60		√	√			04H ^{Note}
FFFFF28AH	16-bit counter 6	TM6	R			√		0000H
FFFFF28CH	16-bit compare register 6	CR6	R/W			√		
FFFFF28EH	Timer clock selection register 61	TCL61			√			00H
FFFFF2A0H	Serial I/O shift register 0	SIO0			√			
FFFFF2A2H	Serial operation mode register 0	CSIM0		√	√			
FFFFF2A4H	Serial clock selection register 0	CSIS0			√			
FFFFF2B0H	Serial I/O shift register 1	SIO1			√			
FFFFF2B2H	Serial operation mode register 1	CSIM1		√	√			
FFFFF2B4H	Serial clock selection register 1	CSIS1			√			
FFFFF2D0H	Serial I/O shift register 3	SIO3			√			
FFFFF2D2H	Serial operation mode register 3	CSIM3		√	√			
FFFFF2D4H	Serial clock selection register 3	CSIS3			√			
FFFFF2E0H	Variable-length serial I/O shift register 4	SIO4				√		0000H
FFFFF2E2H	Variable-length serial control register 4	CSIM4		√	√			00H
FFFFF2E4H	Variable-length serial setting register 4	CSIB4		√	√			
FFFFF2E6H	Baud rate generator source clock selection register 4	BRGCN4			√			
FFFFF2E8H	Baud rate generator output clock selection register 4	BRGCK4			√			7FH
FFFFF300H	Asynchronous serial interface mode register 0	ASIM0		√	√			00H
FFFFF302H	Asynchronous serial interface status register 0	ASIS0	R	√	√			
FFFFF304H	Baud rate generator control register 0	BRGC0	R/W		√			
FFFFF306H	Transmission shift register 0	TXS0	W		√			FFH
FFFFF308H	Reception buffer register 0	RXB0	R		√			
FFFFF30EH	Baud rate generator mode control register 00	BRGMC00	R/W		√			00H
FFFFF310H	Asynchronous serial interface mode register 1	ASIM1		√	√			
FFFFF312H	Asynchronous serial interface status register 1	ASIS1	R	√	√			
FFFFF314H	Baud rate generator control register 1	BRGC1	R/W		√			
FFFFF316H	Transmission shift register 1	TXS1	W		√			FFH
FFFFF318H	Reception buffer register 1	RXB1	R		√			

Note Although the hardware status is initialized to 04H, 00H is read out if read.

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset		
				1 Bit	8 Bits	16 Bits	32 Bits			
FFFFF31EH	Baud rate generator mode control register 10	BRGMC10	R/W		√			00H		
FFFFF320H	Baud rate generator mode control register 01	BRGMC01			√					
FFFFF322H	Baud rate generator mode control register 11	BRGMC11			√					
FFFFF334H	Timer clock selection register 50	TCL50			√					
FFFFF336H	Timer mode control register 50	TMC50		√	√					
FFFFF33AH	16-bit counter 5	TM5	R			√		0000H		
FFFFF33CH	16-bit compare register 5	CR5	R/W			√		00H		
FFFFF33EH	Timer clock selection register 51	TCL51				√				
FFFFF340H	IIC control register 0	IICC0	R/W	√	√			00H		
FFFFF342H	IIC state register 0	IICS0		R	√	√				
FFFFF344H	IIC clock selection register 0	IICCL0			√	√				
FFFFF346H	Slave address register 0	SVA0				√				
FFFFF348H	IIC shift register 0	IIC0				√				
FFFFF34AH	IIC function expansion register 0	IICX0			√	√				
FFFFF34CH	IIC clock expansion register 0	IICCE0				√				
FFFFF360H	Watch timer mode register	WTNM			√	√				
FFFFF364H	Watch timer clock selection register	WTNCS				√				
FFFFF36CH	Correction control register	CORCN			√	√				
FFFFF36EH	Correction request register	CORRQ			√	√				
FFFFF370H	Correction address register 0	CORAD0					√			00000000H
FFFFF374H	Correction address register 1	CORAD1					√			
FFFFF378H	Correction address register 2	CORAD2					√			
FFFFF37CH	Correction address register 3	CORAD3					√			
FFFFF380H	Oscillation stabilization time selection register	OSTS				√				04H
FFFFF382H	Watchdog timer clock selection register	WDCS				√				00H
FFFFF384H	Watchdog timer mode register	WDTM		√	√			0000H		
FFFFF38EH	DMA start factor expansion register	DMAS		√	√					
FFFFF3A0H	16-bit timer register 7	TM7				√				
FFFFF3A2H	16-bit capture/compare register 70	CR70				√				
FFFFF3A4H	16-bit capture/compare register 71	CR71				√				
FFFFF3A6H	Prescaler mode register 70	PRM70			√					
FFFFF3A8H	16-bit timer mode control register 7	TMC7		√	√					
FFFFF3AAH	Capture/compare control register 7	CRC7		√	√					
FFFFF3ACH	Timer output control register 7	TOC7		√	√					
FFFFF3AEH	Prescaler mode register 71	PRM71			√					
FFFFF3C0H	A/D converter mode register 1	ADM1		√	√			00H		
FFFFF3C2H	Analog input channel specification register	ADS		√	√					

(7/7)

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bits	16 Bits	32 Bits	
FFFFF3C4H	A/D conversion result register	ADCR	R			√		0000H
FFFFF3C6H	A/D conversion result register H (higher 8 bits)	ADCRH			√			
FFFFF3C8H	A/D converter mode register 2	ADM2	R/W	√	√			00H
FFFFF3D0H	Key return mode register	KRM		√	√			
FFFFF3D4H	Noise elimination control register	NCC			√			

3.4.9 Specific registers

Specific registers are registers that are protected from being written with illegal data due to erroneous program execution, etc. The write access of these specific registers is executed in a specific sequence, and the occurrence of abnormal store operations is reported by the system status register (SYS). The V850/SF1 has two specific registers, the power save control register (PSC) and processor clock control register (PCC). For details of the PSC register, refer to **4.3.1 (2) Power save control register (PSC)**, and for details of the PCC register, refer to **4.3.1 (1) Processor clock control register (PCC)**.

The following sequence shows the data setting of the specific registers.

- <1> Disable DMA operation.
- <2> Disable interrupts (set the PSW NP bit to 1).
- <3> Write any 8-bit data in the command register (PRCMD).
- <4> Write the data to be set in the specific registers (using the following instructions).
 - Store instruction (ST/SST instruction)
 - Bit manipulation instruction (SET1/CLR1/NOT1 instruction)
- <5> Cancel the interrupt-disabled state (return the PSW NP bit to 0).
- <6> Insert NOP instructions (2 or 5 instructions).
- <7> If necessary, enable DMA operation.

No special sequence is required when reading the specific registers.

Cautions 1. If an interrupt request or a DMA request is acknowledged between the time PRCMD is generated (<3>) and the specific register write operation (<4>) that follows immediately after, the write operation to the specific register is not performed and a protection error (PRERR bit of SYS register is 1) may occur. Therefore, set the NP bit of PSW to 1 (<2>) to disable the acknowledgement of INT/NMI or to disable DMA transfer.

The above also applies when a bit manipulation instruction is used to set a specific register. Moreover, to ensure that the execution routine following release of the STOP/IDLE mode is performed correctly, insert a NOP instruction as a dummy instruction (<6>). If the value of the ID bit of the PSW does not change as the result of execution of the instruction to return the NP bit to 0 (<5>), insert two NOP instructions, and if the value of the ID bit of the PSW changes, insert five NOP instructions.

A description example is given below.

[Description example]: In case of PSC register

```

LDSR rX,5          ; NP bit = 1
ST.B r0,PRCMD [r0] ; Write to PRCMD
ST.B rD,PSC [r0]   ; PSC register setting
LDSR rY,5          ; NP bit = 0
NOP                ; Dummy instruction (2 or 5 instructions)
    :
    :
NOP
(next instruction) ; Execution routine following cancellation of
STOP/IDLE mode
    :
    :

```

rX: Value to be written to PSW

rY: Value to be written back to PSW

rD: Value to be set to PSC

When saving the value of the PSW, the value of the PSW prior to setting the NP bit must be transferred to the rY register.

- Cautions**
2. The instructions (<5> interrupt disable cancel, <6> NOP instruction) following the store instruction for the PSC register for setting the software STOP mode and IDLE mode are executed before a power save mode is entered.
 3. Always stop DMA prior to accessing specific registers.

(1) Command register (PRCMD)

The command register (PRCMD) is used to prevent incorrect writing to the specific registers due to an inadvertent program loop when write-accessing the specific register.

This register can be written in 8-bit units. It becomes undefined in a read cycle.

The occurrence of illegal store operations can be checked by the PRERR bit of the SYS register.

After reset: Undefined	W	Address: FFFFF170H							
		7	6	5	4	3	2	1	0
PRCMD		REG7	REG6	REG5	REG4	REG3	REG2	REG1	REG0
	REGn	Registration code							
	0/1	Any 8-bit data							

(2) System status register (SYS)

This register is allocated with status flags showing the operating state of the entire system. This register can be read/written in 8- or 1-bit units.

After reset: 00H	R/W	Address: FFFFF078H							
		7	6	5	4	3	2	1	0
SYS		0	0	0	PRERR	0	0	0	0
	PRERR	Detection of protection error							
	0	Protection error did not occur							
	1	Protection error occurred							

The operation conditions of PRERR flag are shown below.

(a) Set conditions (PRERR = 1)

- (1) When a write operation to a specific register took place in a state where the store instruction operation for the recent peripheral I/O was not a write operation to the PRCMD register.
- (2) When the first store instruction operation following a write operation to the PRCMD register is to any peripheral I/O register apart from specific registers.

(b) Reset conditions: (PRERR = 0)

- (1) When 0 is written to the PRERR flag of the SYS register.
- (2) At system reset.

CHAPTER 4 CLOCK GENERATION FUNCTION

4.1 General

The clock generator is a circuit that generates the clock pulses that are supplied to the CPU and peripheral hardware. There are two types of system clock oscillators.

(1) Main system clock oscillator

The oscillator of V850/SF1 has an oscillation frequency of 2 to 16 MHz. Oscillation can be stopped by executing a STOP instruction or by setting the processor clock control register (PCC). Oscillation is also stopped during a reset.

In IDLE mode, supplying the peripheral clock to the clock timer only is possible. Therefore, in IDLE mode, it is possible to operate the clock timer without using the subsystem clock oscillator.

- Cautions 1. When the main oscillator is stopped by inputting a reset or executing a STOP instruction, oscillation stabilization time is secured after the stop mode is canceled. This oscillation stabilization time is set via the oscillation stabilization time selection register (OSTS). The watchdog timer is used to count the oscillation stabilization time.**
- 2. If the main system clock halt is released by clearing the MCK bit to 0 after the main system clock is stopped by setting the MCK bit in the PCC register to 1, the oscillation stabilization time is not secured.**
- 3. External clock input is disabled.**

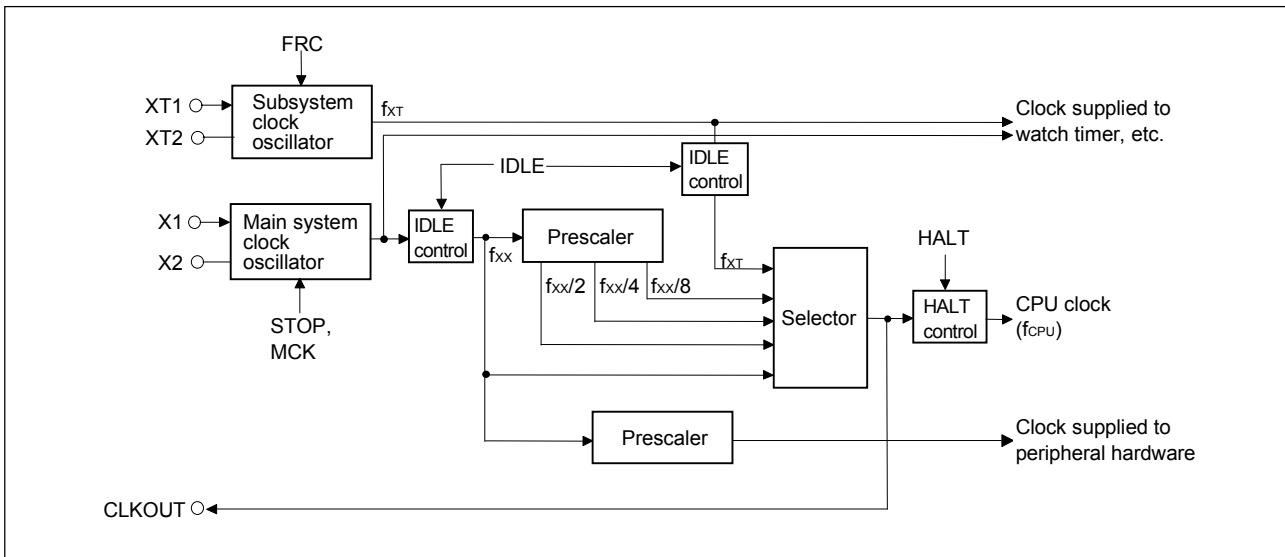
(2) Subsystem clock oscillator

This circuit has an oscillation frequency of 32.768 kHz. Its oscillation is not stopped when the STOP instruction is executed, nor when a reset is input.

When the subsystem clock oscillator is not used, the FRC bit in the processor clock control register (PCC) can be set to disable use of the internal feedback resistor. This enables the current consumption to be kept low in the STOP mode.

4.2 Configuration

Figure 4-1. Clock Generator



4.3 Clock Output Function

This function outputs the CPU clock via the CLKOUT pin.

When clock output is enabled, the CPU clock is output via the CLKOUT pin. When it is disabled, a low-level signal is output via the CLKOUT pin.

Output is stopped in the IDLE or STOP mode (fixed to low level).

This function is controlled via the DCLK1 and DCLK0 bits in the PSC register.

A high-impedance status is set during the reset period. After reset is canceled, a low level is output.

Caution While CLKOUT is being output, the CPU clock (CK2 to CK0 bits of PCC register) cannot be changed.

4.3.1 Control registers

(1) Processor clock control register (PCC)

This is a specific register. It can be written to only when a specified combination of sequences is used (see 3.4.9 Specific registers). This register can be read/written in 8- or 1-bit units.

After reset:	03H	R/W	Address: FFFFF074H					
	7	6	5	4	3	2	1	0
PCC	FRC	MCK	0	0	0	CK2	CK1	CK0
	FRC	Selection of internal feedback resistor for subsystem clock						
	0	Used						
	1	Not used						
	MCK	Operation of main system clock						
	0	Operating						
	1	Stopped						
	CK2 ^{Note}	CK1	CK0	Selection of CPU clock				
	0	0	0	f _{xx}				
	0	0	1	f _{xx} /2				
	0	1	0	f _{xx} /4				
	0	1	1	f _{xx} /8				
	1	X	X	f _{xT} (subsystem clock)				

Note It is recommended to manipulate CK2 in 1-bit units. However, when manipulating the PCC register in 8-bit units, be sure not to change the values of CK1 and CK0.

Cautions 1. Do not change the CPU clock (the value of the CK2 to CK0 in the PCC register) while CLKOUT is being output.

2. Even if the MCK bit is set to 1 during main clock operation, the main clock is not stopped. The CPU clock stops after the subsystem clock is selected.

Remark X: Either 0 or 1

(a) Example of main clock operation → subsystem clock operation setup

- <1> CK2 ← 1: Bit manipulation instructions are recommended. Do not change CK1 and CK0.
- <2> Subsystem clock operation: The maximum number of the following instructions is required before subsystem clock operation after the CK2 bit is set.
(CPU clock frequency before setting / subsystem clock frequency) × 2
Therefore, insert waits equivalent to this number by program.
- <3> MCK ← 1: Only when the main clock is stopped.

(b) Example of subsystem clock operation → main clock operation setup

- <1> MCK ← 0: Main clock oscillation start
- <2> Insert waits by program and wait until the main clock oscillation stabilization time elapses.
- <3> CK2, CK1, CH0 ← CPU clock
- <4> Main clock operation: If CK1 and CK0 are not changed from the CPU clock selection values set before the subsystem clock operation, a maximum of two instructions is required.
If CK1 and CK0 are changed, a maximum of ten instructions is required.

(2) Power save control register (PSC)

This is a specific register. It can be written to only when a specified combination of sequences is used. For details, see **3.4.9 Specific registers**.

This register can be read/written in 8- or 1-bit units.

After reset:	C0H	R/W	Address: FFFFF070H					
	7	6	5	4	3	2	1	0
PSC	DCLK1	DCLK0	0	0	0	IDLE	STP	0
	DCLK1	DCLK0	Specification of CLKOUT pin operation					
	0	0	Output enabled					
	0	1	Setting prohibited					
	1	0	Setting prohibited					
	1	1	Output disabled (when reset)					
	IDLE	IDLE mode setting						
	0	Normal mode						
	1	IDLE mode ^{Note 1}						
	STP	STOP mode setting						
	0	Normal mode						
	1	STOP mode ^{Note 2}						

Notes

1. When IDLE mode is canceled, this bit is automatically reset to 0.
2. When STOP mode is canceled, this bit is automatically reset to 0.

Caution The bits in DCLK0 and DCLK1 should be manipulated in 8-bit units.

(3) Oscillation stabilization time selection register (OSTS)

This register can be read/written in 8-bit units.

After reset: 04H R/W Address: FFFF380H

	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0

OSTS2	OSTS1	OSTS0	Selection of oscillation stabilization time		
			Clock	f _{xx}	
				16 MHz	8 MHz
0	0	0	$2^{16}/f_{xx}$	4.10 ms	8.19 ms
0	0	1	$2^{18}/f_{xx}$	16.4 ms	32.8 ms
0	1	0	$2^{19}/f_{xx}$	32.8 ms	65.5 ms
0	1	1	$2^{20}/f_{xx}$	65.5 ms	131 ms
1	0	0	$2^{21}/f_{xx}$	131 ms	262 ms
Other than above			Setting prohibited		

4.4 Power Save Functions

4.4.1 General

This product provides the following power saving functions.

These modes can be combined and switched to suit the target application, thus enabling the effective implementation of low-power systems.

(1) HALT mode

In this mode, the clock oscillator continues to operate but the CPU operating clock is stopped. A clock continues to be supplied for other on-chip peripheral functions to maintain operation of those functions. This enables the system's total power consumption to be reduced.

A special-purpose instruction (the HALT instruction) is used to switch to HALT mode.

(2) IDLE mode

This mode stops the entire system by stopping the CPU operating clock as well as the operating clock for on-chip peripheral functions while the clock oscillator is still operating. However, the subsystem clock continues to operate and supplies a clock to the on-chip peripheral functions.

When this mode is canceled, there is no need for the oscillator to wait for the oscillation stabilization time, so normal operation can be resumed quickly.

When the IDLE bit in the power saving control register (PSC) is set (1), the system switches to IDLE mode.

(3) Software STOP mode

This mode stops the entire system by stopping the clock oscillator for the main clock. The subsystem clock continues to be supplied to keep on-chip peripheral functions operating. If the subsystem clock is not used, ultra-low-power-consumption mode (leak current only) is set. STOP mode setting is prohibited if the CPU is operating via the subsystem clock.

If the STP bit of the PSC register is set (1), the system enters STOP mode.

(4) Subsystem clock operation

In this mode, the CPU clock is set to operate using the subsystem clock and the MCK bit of the PCC register is set (1) to set low-power-consumption mode in which the entire system operates using only the subsystem clock.

When HALT mode has been set, the CPU operating clock is stopped so that power consumption can be reduced.

When IDLE mode has been set, the CPU operating clock and some peripheral functions (DMAC and BCU) are stopped to enable an even greater reduction in power consumption than when in HALT mode.

4.4.2 HALT mode

(1) Settings and operating states

In this mode, the clock oscillator continues to operate but the CPU operating clock is stopped. A clock continues to be supplied for other on-chip peripheral functions to maintain operation of those functions. When HALT mode is set while the CPU is idle, it enables the system's total power consumption to be reduced.

When in HALT mode, execution of programs is stopped but the contents of all registers and on-chip RAM are retained as they were just before HALT mode was set. In addition, all on-chip peripheral functions that do not depend on instruction processing by the CPU continue operating.

HALT mode can be set by executing the HALT instruction. It can be set when the CPU is operating via either the main clock or subsystem clock.

The operating statuses in the HALT mode are listed in Table 4-1.

(2) Cancellation of HALT mode

HALT mode can be canceled by an NMI request, an unmasked maskable interrupt request, or $\overline{\text{RESET}}$ input.

(a) Cancellation by interrupt request

HALT mode is canceled regardless of the priority level when an NMI request or an unmasked maskable interrupt request occurs. However, the following occurs if HALT mode was set as part of an interrupt servicing routine.

- (i) When an interrupt request that has a lower priority level than the interrupt currently being serviced occurs, only HALT mode is canceled and the lower-priority interrupt request is not acknowledged. The interrupt request itself is retained.
- (ii) When an interrupt request (including NMI request) that has a higher priority level than the interrupt currently being serviced occurs, HALT mode is canceled and the interrupt request is acknowledged.

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

This is the same as for normal reset operations.

Table 4-1. Operating Statuses in HALT Mode (1/2)

HALT Mode Setting		When CPU Operates on Main Clock		When CPU Operates on Subsystem Clock	
		When Subsystem Clock Does Not Exist	When Subsystem Clock Exists	When Main Clock Oscillation Continues	When Main Clock Oscillation Is Stopped
CPU		Stopped			
ROM correction		Stopped			
Clock generator		Oscillation for main clock and subsystem clock Clock supply to CPU is stopped			
16-bit timer (TM0)		Operating			Operates when INTWT1 is selected as count clock (f _{XT} is selected for watch timer)
16-bit timer (TM1)		Operating			Stopped
16-bit timer (TM2)		Operating			Stopped
16-bit timer (TM3)		Operating			Stopped
16-bit timer (TM4)		Operates when other than f _{XT} is selected for count clock	Operating		Operates when f _{XT} is selected for count clock
16-bit timer (TM5)		Operates when other than f _{XT} is selected for count clock	Operating		Operates when f _{XT} is selected for count clock
16-bit timer (TM6)		Operating			Stopped
16-bit timer (TM7)		Operating			Stopped
Watch timer		Operates when main clock is selected for count clock	Operating		Operates when f _{XT} is selected for count clock
Watchdog timer		Operating (interval timer only)			
Serial interface	CSI0 , CSI1, CSI3	Operating			Operates when an external clock is selected as the serial clock
	I ² C0	Operating			Stopped
	UART0, UART1	Operating			Operates when an external clock is selected as the baud rate clock
	CSI4	Operating			Operates when an external clock is selected as the serial clock
FCAN1, FCAN2 ^{Note}		Operating			Stopped
A/D converter		Operating			Stopped
DMA0 to DMA5		Operating			

Note Available only for the μ PD703079Y, 70F3079Y.

Table 4-1. Operating Statuses in HALT Mode (2/2)

HALT Mode Setting		When CPU Operates on Main Clock		When CPU Operates on Subsystem Clock	
		When Subsystem Clock Does Not Exist	When Subsystem Clock Exists	When Main Clock Oscillation Continues	When Main Clock Oscillation Is Stopped
Item					
Port function		Held			
External bus interface		Only bus hold function operates			
External interrupt request	NMI	Operating			
	INTP0 to INTP3	Operating			
	INTP4 and INTP5	Operating			Stopped
	INTP6	Operates when other than f_{XT} is selected for noise eliminator	Operating		Operates when f_{XT} is selected for noise eliminator
Key return function		Operating			
In external expansion mode	AD0 to AD15	High impedance ^{Note}			
	A16 to A21				
	\overline{LBEN} , \overline{UBEN}	Held ^{Note} (high impedance when $\overline{HLDAK} = 0$)			
	$\overline{R/W}$	High level output ^{Note} (high impedance when $\overline{HLDAK} = 0$)			
	\overline{DSTB}				
	\overline{ASTB}				
	\overline{HLDAK}	Operating			

Note Even when the HALT instruction has been executed, the instruction fetch operation continues until the on-chip instruction prefetch queue becomes full. Once it is full, operation stops in the state shown in Table 4-1.

4.4.3 IDLE mode

(1) Settings and operating states

This mode stops the entire system except the watch timer by stopping the on-chip main clock supply while the clock oscillator is still operating. Supply to the subsystem clock continues. When this mode is canceled, there is no need for the oscillator to wait for the oscillation stabilization time, so normal operation can be resumed quickly.

When in IDLE mode, program execution is stopped and the contents of all registers and internal RAM are retained as they were just before IDLE mode was set. In addition, on-chip peripheral functions are stopped (except for peripheral functions that are operating with the subsystem clock). External bus hold requests (HLDRQ) are not acknowledged.

When the IDLE bit of the power save control register (PSC) is set (1), the system switches to IDLE mode.

The operating statuses in IDLE mode are listed in Table 4-2.

(2) Cancellation of IDLE mode

IDLE mode can be canceled by a non-maskable interrupt, an unmasked maskable interrupt request output from an operable on-chip peripheral I/O, or RESET input.

Table 4-2. Operating Statuses in IDLE Mode (1/2)

IDLE Mode Settings		When Subsystem Clock Exists	When Subsystem Clock Does Not Exist
Item			
CPU		Stopped	
ROM correction		Stopped	
Clock generator		Both a main clock and subsystem clock oscillator Clock supply to CPU and on-chip peripheral functions is stopped	
16-bit timer (TM0)		Operates when INTWTNI is selected as count clock (f _{XT} is selected for watch timer)	Stopped
16-bit timer (TM1)		Stopped	
16-bit timer (TM2)		Stopped	
16-bit timer (TM3)		Stopped	
16-bit timer (TM4)		Operates when f _{XT} is selected for count clock	Stopped
16-bit timer (TM5)		Operates when f _{XT} is selected for count clock	Stopped
16-bit timer (TM6)		Stopped	
16-bit timer (TM7)		Stopped	
Watch timer		Operating	
Watchdog timer		Stopped	
Serial interface	CSI0, CSI1, CSI3	Operates when an external clock is selected as the serial clock	
	I ² C0	Stopped	
	UART0, UART1	Operates when an external clock is selected as the baud rate clock	
	CSI4	Operates when an external clock is selected as the serial clock	
FCAN1, FCAN2 ^{Note}		Stopped	
A/D converter		Stopped	
DMA0 to DMA5		Stopped	
Port function		Held	

Note Available only for the μ PD703079Y, 70F3079Y.

Table 4-2. Operating Statuses in IDLE Mode (2/2)

IDLE Mode Settings		When Subsystem Clock Exists	When Subsystem Clock Does Not Exist
Item			
External bus interface		Stopped	
External interrupt request	NMI	Operating	
	INTP0 to INTP3	Operating	
	INTP4 and INTP5	Stopped	
	INTP6	Operates when f _{XT} is selected for sampling clock	Stopped
Key return function		Operating	
In external expansion mode	AD0 to AD15	High impedance	
	A16 to A21		
	$\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$		
	$\overline{\text{R/W}}$		
	$\overline{\text{DSTB}}$		
	$\overline{\text{ASTB}}$		
	$\overline{\text{HLDK}}$		

4.4.4 Software STOP mode

(1) Settings and operating states

This mode stops the entire system by stopping the main clock oscillator supplying the internal main clock. The subsystem clock oscillator continues operating and the internal subsystem clock supply is continued.

If the FRC bit in the processor clock control register (PCC) is set (1) when the subsystem clock oscillator is used, the subsystem clock oscillator's on-chip feedback resistor is cut. This sets ultra-low-power-consumption mode, in which the only current is the device's leak current.

In this mode, program execution is stopped and the contents of all registers and internal RAM are retained as they were just before software STOP mode was set.

This mode can be set only when the main clock is being used as the CPU clock. This mode is set when the STP bit in the power save control register (PSC) has been set to 1.

Do not set this mode when the subsystem clock has been selected as the CPU clock.

The operating statuses for software STOP mode are listed in Table 4-3.

★ **Caution** In order to reduce the current consumption in software STOP mode, be sure to initialize the FCAN settings as described below, regardless of the use of FCAN.

<1> Set the GOM bit of the CGST register to "1" (set GOM = 1, clear GOM = 0)

<2> Set the SMNO1 and SMNO0 bits of the CGMSS register to "01"

<3> Set the GOM bit of the CGST register to "0" (set GOM = 0, clear GOM = 1)

For details of FCAN settings, refer to CHAPTER 18 FCAN CONTROLLER.

(2) Cancellation of software STOP mode

Software STOP mode can be canceled by a non-maskable interrupt, an unmasked maskable interrupt request output from an operable on-chip peripheral I/O, or $\overline{\text{RESET}}$ input.

When the STOP mode is canceled, oscillation stabilization time must be secured.

Table 4-3. Operating Statuses in Software STOP Mode

STOP Mode Settings		When Subsystem Clock Exists	When Subsystem Clock Does Not Exist
Item			
CPU		Stopped	
ROM correction		Stopped	
Clock generator		Oscillation for main clock is stopped and oscillation for subsystem clock continues Clock supply to CPU and on-chip peripheral functions is stopped	
16-bit timer (TM0)		Operates when INTWTNI is selected for count clock (f _{XT} is selected as count clock for watch timer)	Stopped
16-bit timer (TM1)		Stopped	
16-bit timer (TM2)		Stopped	
16-bit timer (TM3)		Stopped	
16-bit timer (TM4)		Operates when f _{XT} is selected for count clock	Stopped
16-bit timer (TM5)		Operates when f _{XT} is selected for count clock	Stopped
16-bit timer (TM6)		Stopped	
16-bit timer (TM7)		Stopped	
Watch timer		Operates when f _{XT} is selected for count clock	Stopped (operation disabled)
Watchdog timer		Stopped	
Serial interface	CSI0, CSI1, CSI3	Operates when an external clock is selected as the serial clock	
	I ² C0	Stopped	
	UART0, UART1	Operates when an external clock is selected as the baud rate clock	
	CSI4	Operates when an external clock is selected as the serial clock	
FCAN1, FCAN2 ^{Note}		Stopped	
A/D converter		Stopped	
DMA0 to DMA5		Stopped	
Port function		Held	
External bus interface		Stopped	
External interrupt request	NMI	Operating	
	INTP0 to INTP3	Operating	
	INTP4 and INTP5	Stopped	
	INTP6	Operates when f _{XT} is selected for the sampling clock	Stopped
Key return function		Operating	
In external expansion mode	AD0 to AD15	High impedance	
	A16 to A21		
	LBEN, UBEN		
	R/W		
	DSTB		
	ASTB		
	HILDAK		

Note Available only for the μ PD703079Y and 70F3079Y.

4.5 Oscillation Stabilization Time

The following shows methods for specifying the length of oscillation stabilization time required to stabilize the oscillator following cancelation of STOP mode.

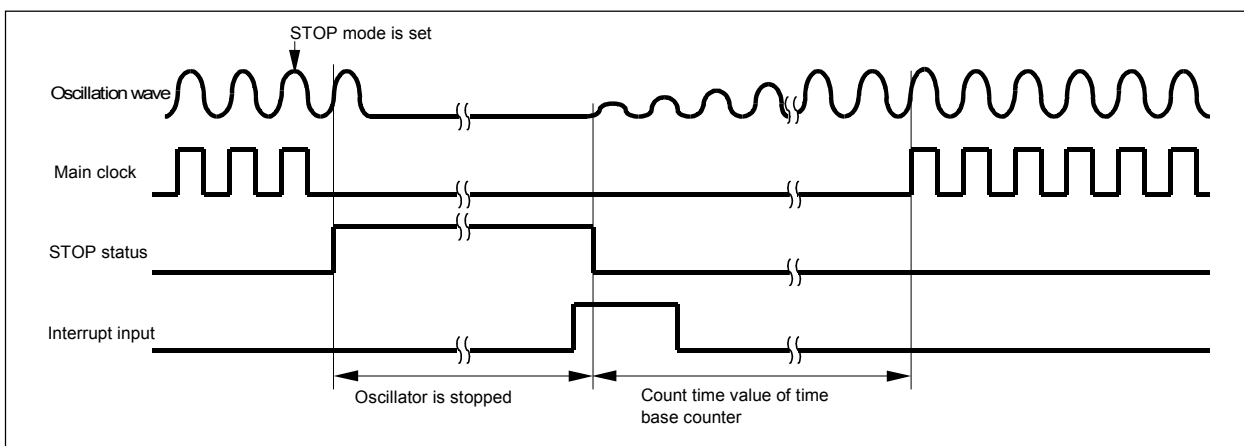
(1) Cancelation by non-maskable interrupt or by unmasked maskable interrupt request

STOP mode is canceled by a non-maskable interrupt or an unmasked maskable interrupt request. When an interrupt is input, the counter (watchdog timer) starts counting and the count time is the length of time that must elapse until the oscillator's clock output stabilizes.

Oscillation stabilization time \doteq WDT count time

After the specified amount of time has elapsed, system clock output starts and processing branches to the interrupt handler address.

Figure 4-2. Oscillation Stabilization Time



(2) Use of $\overline{\text{RESET}}$ pin to allocate time ($\overline{\text{RESET}}$ pin input)

For allocating time with the $\overline{\text{RESET}}$ pin, refer to **CHAPTER 14 RESET FUNCTION**.

★ 4.6 Cautions on Power Save Function

If the V850/SF1 is used under the following conditions, a discrepancy occurs between the address indicated by the program counter (PC) and the address at which an instruction is actually read after the power save mode is released.

This may result in the CPU ignoring a 4- or 8-byte instruction from between 4 bytes and 16 bytes after an instruction is executed to write to the PSC register, which could in turn result in the execution of an erroneous instruction.

[Conditions]

- (i) Setting of power save mode (IDLE mode or STOP mode) while an instruction is being executed on external ROM
- (ii) Cancellation of power save mode as the result of an interrupt request
- (iii) Execution of the next instruction when an interrupt request is held pending following cancellation of the power save mode

Conditions for interrupt request to be held pending:

- When NP flag of PSW register is "1" (NMI servicing in progress/set by software)
- When ID flag of PSW register is "1" (interrupt request servicing in progress/DI instruction/set by software)
- When an interrupt enable (EI) state occurs during interrupt request servicing, but this state is cleared by an interrupt request with the same or lower priority

Therefore, use the V850/SF1 under the following conditions.

[Usage Conditions]

- (i) Do not use a power save mode (IDLE mode or STOP mode) during instruction execution on external ROM.
- (ii) If it is necessary to use a power save mode during instruction execution on external ROM, implement the following software measures.
 - Insert 6 NOP instructions 4 bytes after an instruction that writes to the PSC register.
 - After the NOP instructions, insert a br\$+2 instruction to cancel the PC discrepancy.

[Workaround program example]

```
ldsr  rX,5           ;Sets rX value to PSW
st.b  r0,PRCMD[r0]   ;Writes to PRCMD
st.b  rD,PSC[r0]     ;Sets PSC register
ldsr  rY,5           ;Returns PSW value
nop                                ;6 or more NOP instructions
nop
nop
nop
nop
nop
nop
br  $+2              ;Cancels PC discrepancy
```

Remark It is assumed that `rD` (PSC setting value), `rX` (value written to PSW), and `rY` (value written back to PSW) have been set.

CHAPTER 5 PORT FUNCTION

5.1 Port Configuration

The V850/SF1 includes 84 port pins from ports 0 to 11, of which 72 are I/O pins and 12 are input only pins. There are two pin I/O buffer power supplies: ADCV_{DD} and PORTV_{DD}, which are described below.

Table 5-1. Pin I/O Buffer Power Supplies

Power Supply	Corresponding Pins
ADCV _{DD}	Port 7, port 8
PORTV _{DD}	Port 0, port 1, port 2, port 3, port 4, port 5, port 6, port 9, port 10, port 11, RESET, CLKOUT

5.2 Port Pin Functions

5.2.1 Port 0

Port 0 is an 8-bit I/O port for which I/O settings can be controlled in 1-bit units.

When using P00 to P04 as the NMI or INTP0 to INTP3 pins, noise is eliminated by an analog noise eliminator.

When using P05 to P07 as the INTP4/ADTRG, INTP5, and INTP6 pins, noise is eliminated by a digital noise eliminator.

After reset:	00H	R/W						Address: FFFFF00H
	7	6	5	4	3	2	1	0
P0	P07	P06	P05	P04	P03	P02	P01	P00
	P0n	Control of output data (in output mode) (n = 0 to 7)						
	0	Outputs 0						
	1	Outputs 1						

Remark In input mode: When the P0 register is read, the pin levels at that time are read. Writing to P0 writes the values to that register. This does not affect the input pins.

In output mode: When the P0 register is read, the values of P0 are read. Writing to P0 writes the values to that register, and those values are immediately output.

Port 0 includes the following alternate functions.

Table 5-2. Port 0 Alternate Function Pins

Pin Name		Alternate Function	I/O	PULL ^{Note}	Remark
Port 0	P00	NMI	I/O	No	Analog noise elimination
	P01	INTP0			
	P02	INTP1			
	P03	INTP2			
	P04	INTP3			Digital noise elimination
	P05	INTP4/ADTRG			
	P06	INTP5			
	P07	INTP6			

Note Software pull-up function

(1) Function of P0 pins

Port 0 is an 8-bit I/O port for which I/O settings can be controlled in 1-bit units. I/O settings are controlled via the port 0 mode register (PM0).

In output mode, the values set to each bit are output to port 0 (P0). When using this port in output mode, either the valid edge of each interrupt request should be made invalid or each interrupt request should be masked (except for NMI requests).

When using this port in input mode, the pin statuses can be read by reading P0. Also, the values of P0 (output latch) can be read by reading P0 while in output mode.

The valid edges of NMI and INTP0 to INTP6 are specified via rising edge specification register 0 (EGP0) and falling edge specification register 0 (EGN0).

When a reset is input, the settings are initialized to input mode. Also, the valid edge of each interrupt request becomes invalid (NMI and INTP0 to INTP6 do not function immediately after reset).

(2) Noise elimination

(a) Elimination of noise from NMI and INTP0 to INTP3 pins

An on-chip noise eliminator is provided that uses analog delay to eliminate noise. Consequently, if a signal having a constant level is input for longer than a specified time to these pins, it is detected as a valid edge. Such edge detection occurs only after the specified amount of time.

(b) Elimination of noise from INTP4 to INTP6 and ADTRG pins

A digital noise eliminator is provided on chip.

This circuit uses digital sampling. A pin's input level is detected using a sampling clock (f_{xx}), and noise elimination is performed for the INTP4, INTP5, and ADTRG pins if the same level is not detected three times consecutively. The noise-elimination width can be changed for the INTP6 pin (see **7.3.8 (3) Noise elimination of INTP6 pin**).

Cautions 1. If the input pulse width is 2 or 3 clocks, whether it will be detected as a valid edge or eliminated as noise is undefined.

To ensure correct detection of the valid edge, constant-level input is required for 3 clocks or more.

2. If noise is occurring in synchronization with the sampling clock, it may not be recognized as noise. In such cases, attach a filter to the input pins to eliminate the noise.
3. Noise elimination is not performed when these pins are used as an ordinary input port.

(3) Control registers

(a) Port 0 mode register (PM0)

PM0 can be read/written in 8- or 1-bit units.

After reset:	FFH	R/W						Address:	FFFF020H		
	7	6	5	4	3	2	1	0			
PM0	PM07	PM06	PM05	PM04	PM03	PM02	PM01	PM00			
	PM0n	Control of I/O mode (n = 0 to 7)									
	0	Output mode									
	1	Input mode									

(b) Rising edge specification register 0 (EGP0)

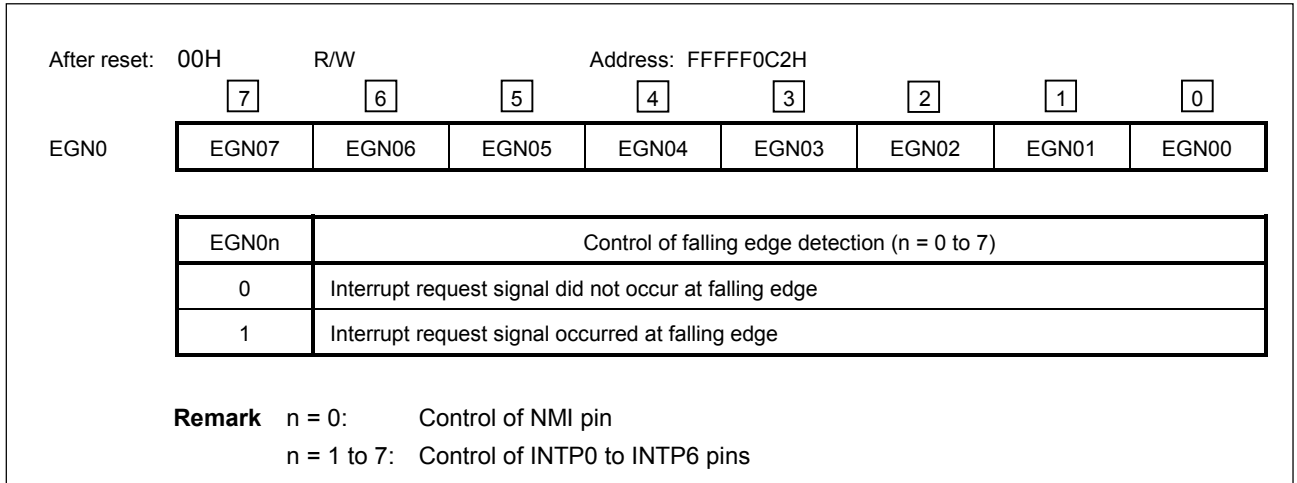
EGP0 can be read/written in 8- or 1-bit units.

After reset:	00H	R/W						Address:	FFFF0C0H		
	7	6	5	4	3	2	1	0			
EGP0	EGP07	EGP06	EGP05	EGP04	EGP03	EGP02	EGP01	EGP00			
	EGP0n	Control of rising edge detection (n = 0 to 7)									
	0	Interrupt request signal did not occur at rising edge									
	1	Interrupt request signal occurred at rising edge									

Remark n = 0: Control of NMI pin
n = 1 to 7: Control of INTP0 to INTP6 pins

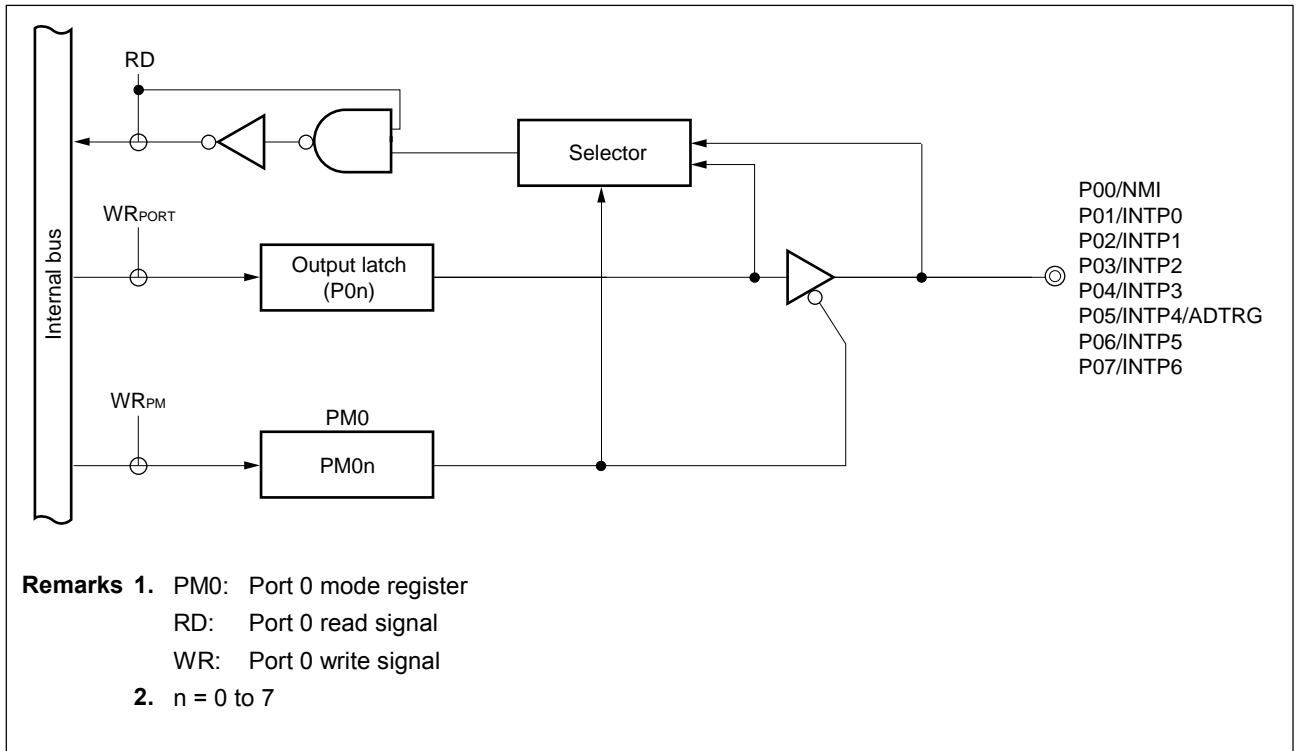
(c) Falling edge specification register 0 (EGN0)

EGN0 can be read/written in 8- or 1-bit units.



(4) Block diagram (Port 0)

Figure 5-1. Block Diagram of P00 to P07



5.2.2 Port 1

Port 1 is a 6-bit I/O port for which I/O settings can be controlled in 1-bit units. Bits 0 and 2 are selectable as normal outputs or N-ch open-drain outputs.

After reset:	00H	R/W	Address: FFFF002H					
	7	6	5	4	3	2	1	0
P1	0	0	P15	P14	P13	P12	P11	P10

P1n	Control of output data (in output mode) (n = 0 to 5)
0	Outputs 0
1	Outputs 1

Remark In input mode: When P1 is read, the pin levels at that time are read. Writing to P1 writes the values to that register. This does not affect the input pins.

In output mode: When P1 is read, the values of P1 are read. Writing to P1 writes the values to that register, and those values are immediately output.

Port 1 includes the following alternate functions.

Table 5-3. Port 1 Alternate Function Pins

Pin Name	Alternate Function	I/O	PULL ^{Note}	Remark	
Port 1	P10	SI0/SDA0	I/O	No	Selectable as N-ch open-drain output
	P11	SO0			–
	P12	SCK0/SCL0			Selectable as N-ch open-drain output
	P13	SI1/RXD0			–
	P14	SO1/TXD0			
	P15	SCK1/ASCK0			

Note Software pull-up function

(1) Function of P1 pins

Port 1 is a 6-bit I/O port for which I/O settings can be controlled in 1-bit units. I/O settings are controlled via the port 1 mode register (PM1).

In output mode, the values set to each bit are output to port 1 (P1). The port 1 function register (PF1) can be used to specify whether P10 and P12 are normal outputs or N-ch open-drain outputs.

When using this port in input mode, the pin statuses can be read by reading P1. Also, the values of P1 (output latch) can be read by reading P1 while in output mode.

Clear P1 and the PM1 register to 0 when using alternate-function pins as outputs. The logical sum (ORed result) of the port output and the alternate-function pin is output from the pins.

When a reset is input, the settings are initialized to input mode.

(2) Control registers

(a) Port 1 mode register (PM1)

PM1 can be read/written in 8- or 1-bit units.

After reset:	3FH	R/W	Address: FFFFF022H					
	7	6	5	4	3	2	1	0
PM1	0	0	PM15	PM14	PM13	PM12	PM11	PM10
	PM1n	Control of I/O mode (n = 0 to 5)						
	0	Output mode						
	1	Input mode						

(b) Port 1 function register (PF1)

PF1 can be read/written in 8- or 1-bit units.

After reset:	00H	R/W	Address: FFFFF0A2H					
	7	6	5	4	3	2	1	0
PF1	0	0	0	0	0	PF12	0	PF10
	PF1n	Control of normal output/N-ch open-drain output (n = 0, 2)						
	0	Normal output						
	1	N-ch open-drain output						

(3) Block diagram (Port 1)

Figure 5-2. Block Diagram of P10 and P12

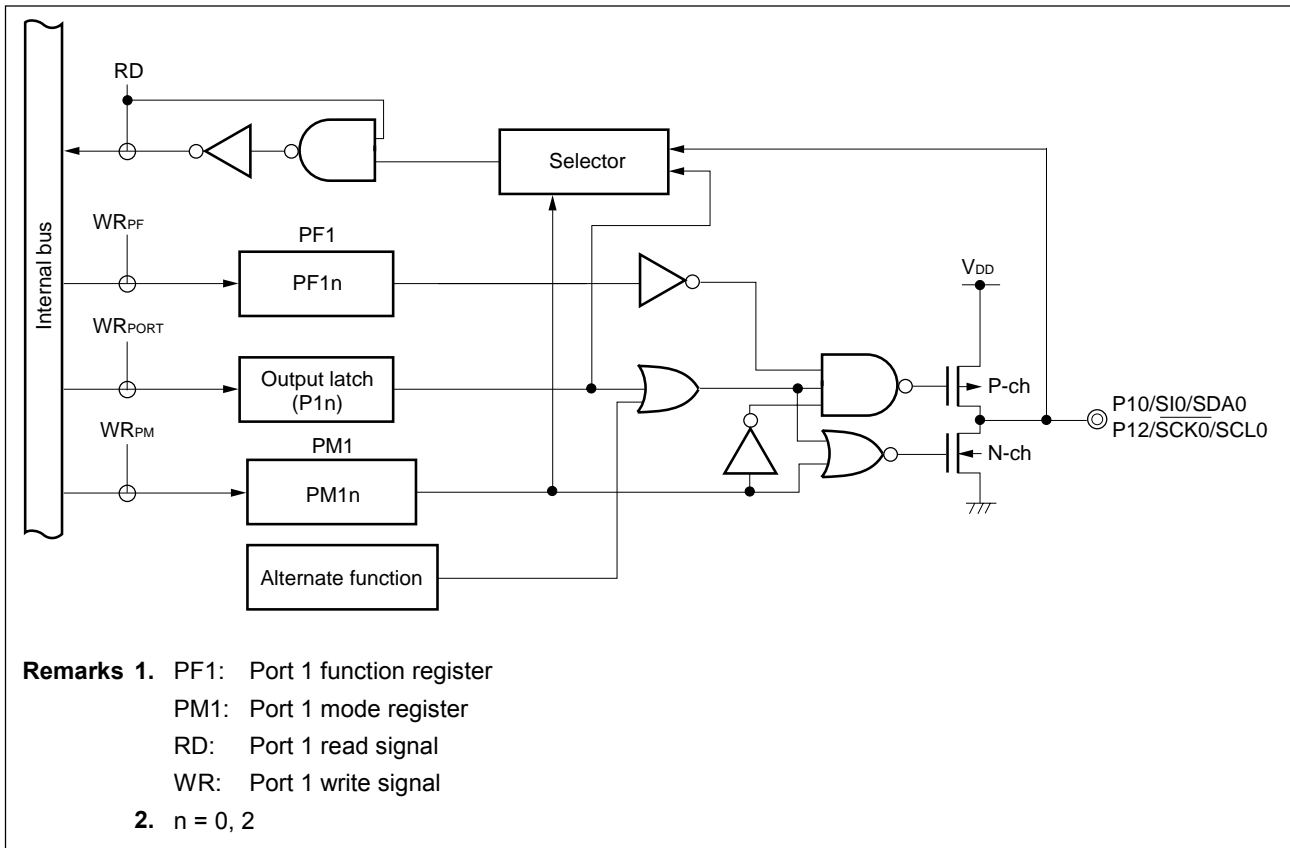
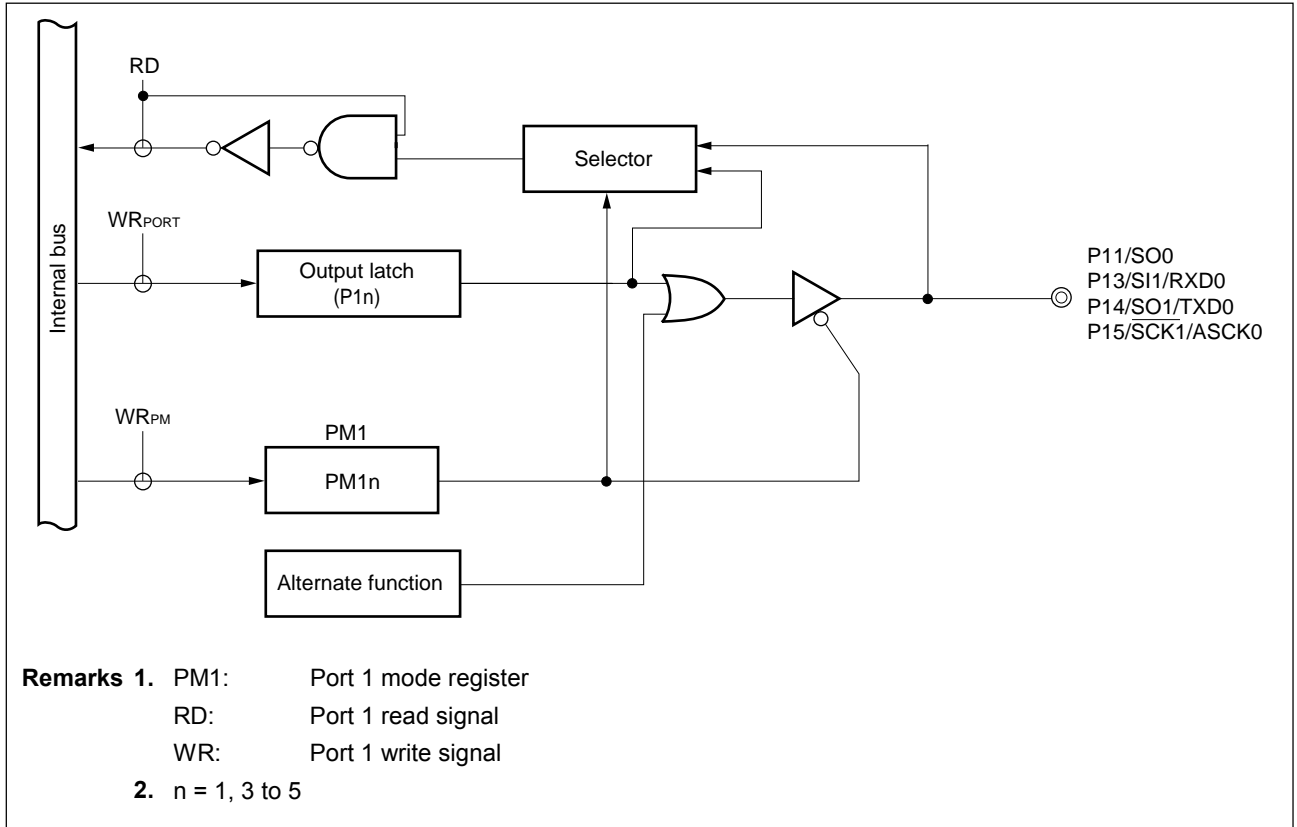


Figure 5-3. Block Diagram of P11 and P13 to P15



5.2.3 Port 2

Port 2 is an 8-bit I/O port for which I/O settings can be controlled in 1-bit units.

After reset: 00H R/W Address: FFFFF004H

	7	6	5	4	3	2	1	0
P2	P27	P26	P25	P24	P23	P22	P21	P20

P2n	Control of output data (in output mode) (n = 0 to 7)
0	Outputs 0
1	Outputs 1

Remark In input mode: When P2 is read, the pin levels at that time are read. Writing to P2 writes the values to that register. This does not affect the input pins.

In output mode: When P2 is read, the values of P2 are read. Writing to P2 writes the values to that register, and those values are immediately output.

Port 2 includes the following alternate functions.

Table 5-4. Port 2 Alternate Function Pins

Pin Name		Alternate Function	I/O	PULL ^{Note}	Remark
Port 2	P20	SI3/RXD1	I/O	No	-
	P21	SO3/TXD1			
	P22	SCK3/ASCK1			
	P23	SI4			
	P24	SO4			
	P25	SCK4			
	P26	-			
	P27	-			

Note Software pull-up function

(1) Function of P2 pins

Port 2 is an 8-bit I/O port for which I/O settings can be controlled in 1-bit units. I/O settings are controlled via the port 2 mode register (PM2).

In output mode, the values set to each bit are output to port 2 (P2).

When using this port in input mode, the pin statuses can be read by reading P2. Also, the values of P2 (output latch) can be read by reading P2 while in output mode.

Clear P2 and the PM2 register to 0 when using alternate-function pins as outputs. The logical sum (ORed result) of the port output and the alternate-function pin is output from the pins.

When a reset is input, the settings are initialized to input mode.

(2) Control register

(a) Port 2 mode register (PM2)

PM2 can be read/written in 8- or 1-bit units.

After reset:	FFH	R/W	Address: FFFFF024H					
	7	6	5	4	3	2	1	0
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20
	PM2n	Control of I/O mode (n = 0 to 7)						
	0	Output mode						
	1	Input mode						

Figure 5-4. Block Diagram of P20 to P25

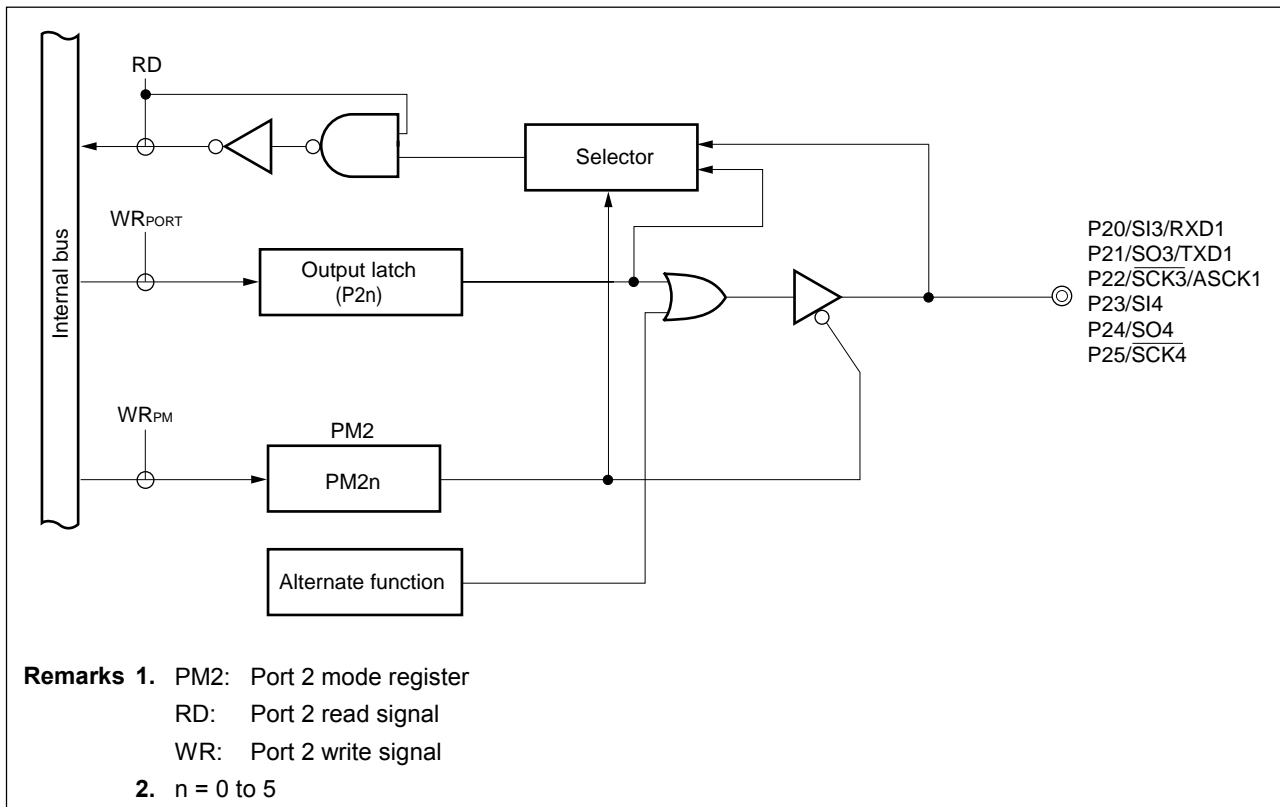
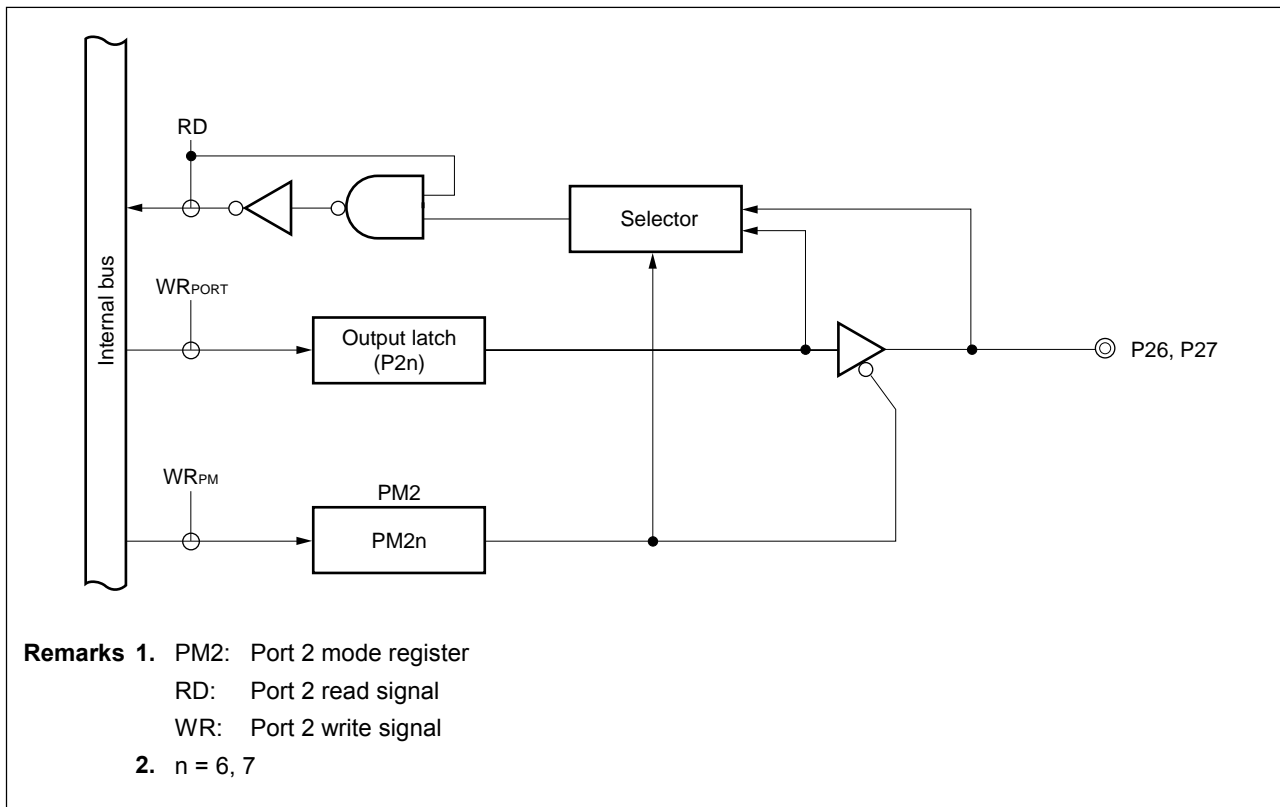


Figure 5-5. Block Diagram of P26 and P27



5.2.4 Port 3

Port 3 is a 5-bit I/O port for which I/O settings can be controlled in 1-bit units.

When using P30 to P33 as the TI2 to TI5 pins, noise is eliminated by a digital eliminator.

After reset:	00H	R/W	Address:	FFFFF006H				
	7	6	5	4	3	2	1	0
P3	0	0	0	P34	P33	P32	P31	P30

P3n	Control of output data (in output mode) (n = 0 to 4)
0	Outputs 0
1	Outputs 1

Remark In input mode: When P3 is read, the pin levels at that time are read. Writing to P3 writes the values to that register. This does not affect the input pins.

In output mode: When P3 is read, the values of P3 are read. Writing to P3 writes the values to that register, and those values are immediately output.

Port 3 includes the following alternate functions.

Table 5-5. Port 3 Alternate Function Pins

Pin Name	Alternate Function	I/O	PULL ^{Note}	Remark	
Port 3	P30	T12/TO2	I/O	No	Digital noise elimination
	P31	T13/TO3			
	P32	T14/TO4			
	P33	T15/TO5			
	P34	VM45/TI71			-

Note Software pull-up function

(1) Function of P3 pins

Port 3 is a 5-bit I/O port for which I/O settings can be controlled in 1-bit units. I/O settings are controlled via the port 3 mode register (PM3).

In output mode, the values set to each bit are output to port 3 (P3).

When using this port in input mode, the pin statuses can be read by reading P3. Also, the values of P3 (output latch) can be read by reading P3 while in output mode.

When using the alternate function as T12 to T15 pins, noise is eliminated by the digital noise eliminator (same as the digital noise eliminator for port 0).

Clear P3 and the PM3 register to 0 when using alternate-function pins as outputs. The logical sum (ORed result) of the port output and the alternate-function pin is output from the pins.

- ★ When using the alternate-function VM45 pin, set this port via the VM45 control register (VM45C). In this case, be sure to set P34 and PM34 to 0.

When a reset is input, the settings are initialized to input mode.

(2) Control register

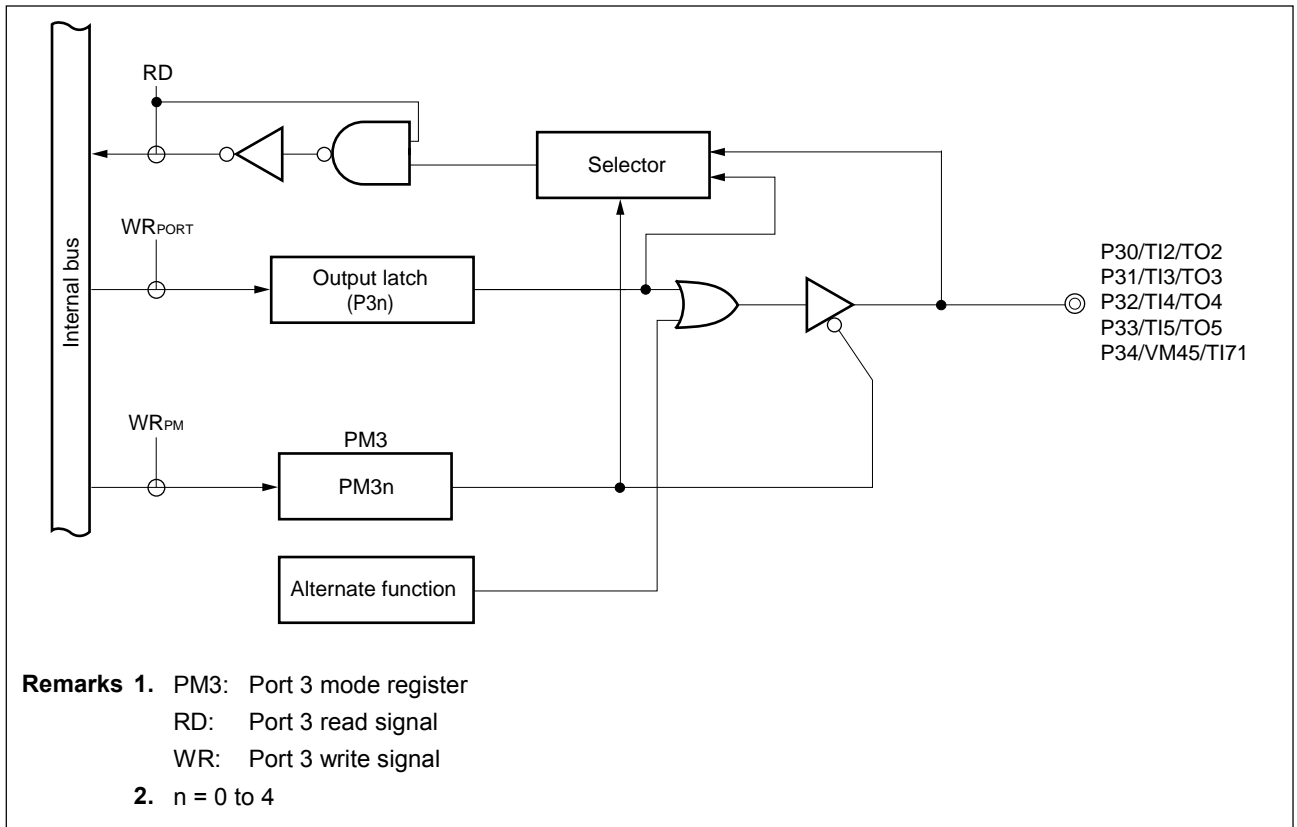
(a) Port 3 mode register (PM3)

PM3 can be read/written in 8- or 1-bit units.

After reset:	FFH	R/W	Address: FFFFF026H					
	7	6	5	4	3	2	1	0
PM3	0	0	0	PM34	PM33	PM32	PM31	PM30
	PM3n	Control of I/O mode (n = 0 to 4)						
	0	Output mode						
	1	Input mode						

(3) Block diagram (Port 3)

Figure 5-6. Block Diagram of P30 to P34



5.2.5 Ports 4 and 5

Ports 4 and 5 are 8-bit I/O ports for which I/O settings can be controlled in 1-bit units.

After reset: 00H R/W Address: FFFFF008H, FFFFF00AH

	7	6	5	4	3	2	1	0
Pn	Pn7	Pn6	Pn5	Pn4	Pn3	Pn2	Pn1	Pn0

(n = 4, 5)

Pnx	Control of output data (in output mode) (n = 4, 5, x = 0 to 7)
0	Outputs 0
1	Outputs 1

Remark In input mode: When P4 and P5 are read, the pin levels at that time are read. Writing to P4 and P5 writes the values to those registers. This does not affect the input pins.
 In output mode: When P4 and P5 are read, their values are read. Writing to P4 and P5 writes the values to those registers, and those values are immediately output.

Ports 4 and 5 include the following alternate functions.

Table 5-6. Alternate Function Pins of Ports 4 and 5

Pin Name		Alternate Function	I/O	PULL ^{Note}	Remark
Port 4	P40	AD0	I/O	No	-
	P41	AD1			
	P42	AD2			
	P43	AD3			
	P44	AD4			
	P45	AD5			
	P46	AD6			
	P47	AD7			
Port 5	P50	AD8	I/O	No	-
	P51	AD9			
	P52	AD10			
	P53	AD11			
	P54	AD12			
	P55	AD13			
	P56	AD14			
	P57	AD15			

Note Software pull-up function

(1) Functions of P4 and P5 pins

Ports 4 and 5 are 8-bit I/O ports for which I/O settings can be controlled in 1-bit units. I/O settings are controlled via port 4 mode register (PM4) and port 5 mode register (PM5).

In output mode, the values set to each bit are output to the port 4 and port 5 (P4 and P5).

When using these ports in input mode, the pin statuses can be read by reading P4 and P5. Also, the values of P4 and P5 (output latch) can be read by reading P4 and P5 while in output mode.

A software pull-up function is not implemented.

When using the P4 and P5 pins as AD0 to AD15, set the pin functions via the memory expansion mode register (MM). This does not affect the PM4 and PM5 registers.

When a reset is input, the settings are initialized to input mode.

(2) Control registers

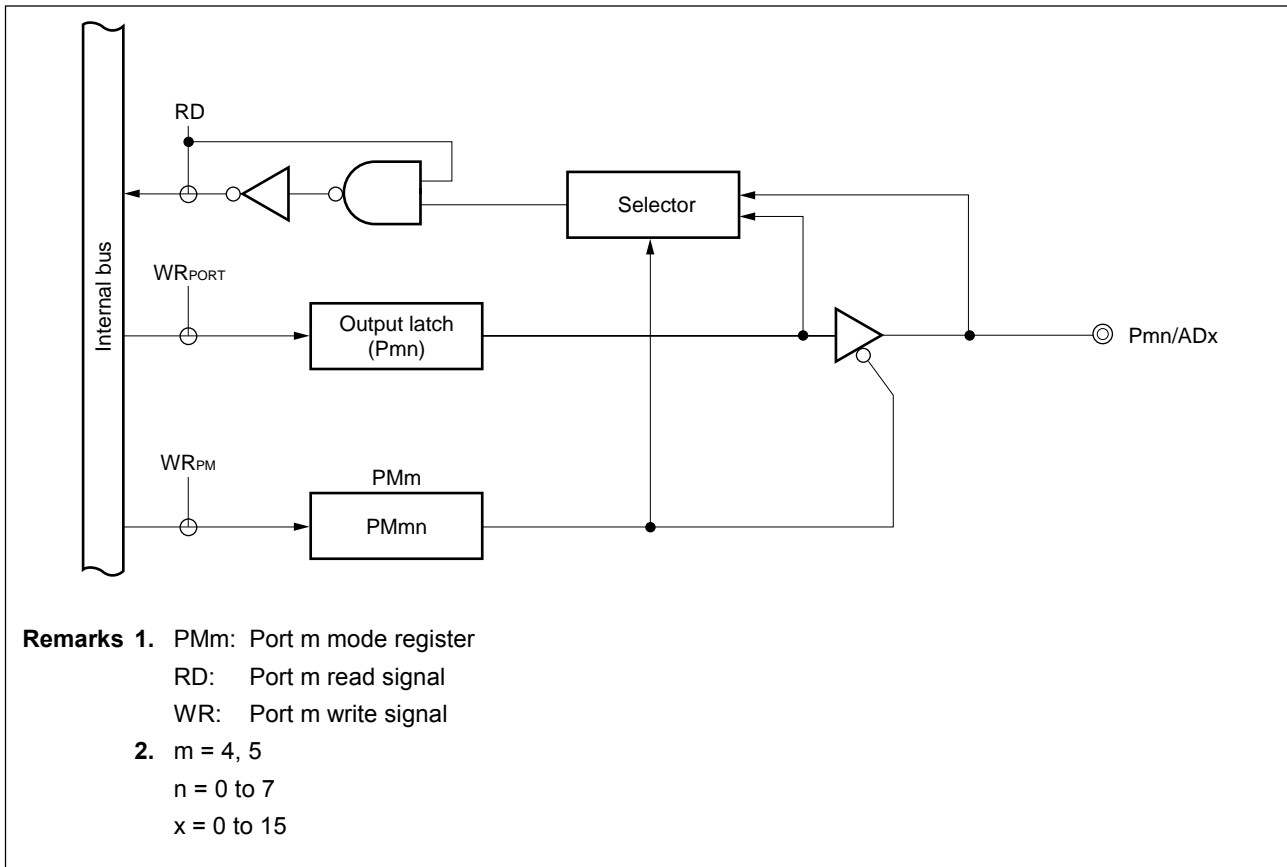
(a) Port 4 mode register and port 5 mode register (PM4 and PM5)

PM4 and PM5 can be read/written in 8- or 1-bit units.

After reset: FFH	R/W	Address: FFFFF028H, FFFFF02AH							
		7	6	5	4	3	2	1	0
PMn		PMn7	PMn6	PMn5	PMn4	PMn3	PMn2	PMn1	PMn0
(n = 4, 5)									
	PMnx	Control of I/O mode (n = 4, 5, x = 0 to 7)							
	0	Output mode							
	1	Input mode							

(3) Block diagram (Ports 4 and 5)

Figure 5-7. Block Diagram of P40 to P47 and P50 to P57



5.2.6 Port 6

Port 6 is a 6-bit I/O port for which I/O settings can be controlled in 1-bit units.

After reset:	00H	R/W	Address:	FFFFFF00CH				
	7	6	5	4	3	2	1	0
P6	0	0	P65	P64	P63	P62	P61	P60

P6n	Control of output data (in output mode) (n = 0 to 5)
0	Outputs 0
1	Outputs 1

Remark In input mode: When P6 is read, the pin levels at that time are read. Writing to P6 writes the values to that register. This does not affect the input pins.

In output mode: When P6 is read, the values of P6 are read. Writing to P6 writes the values to that register, and those values are immediately output.

Port 6 includes the following alternate functions.

Table 5-7. Port 6 Alternate Function Pins

Pin Name	Alternate Function	I/O	PULL ^{Note}	Remark	
Port 6	P60	A16	I/O	No	-
	P61	A17			
	P62	A18			
	P63	A19			
	P64	A20			
	P65	A21			

Note Software pull-up function

(1) Function of P6 pins

Port 6 is a 6-bit I/O port for which I/O settings can be controlled in 1-bit units. I/O settings are controlled via the port 6 mode register (PM6).

In output mode, the values set to each bit are output to port 6 (P6).

When using this port in input mode, the pin statuses can be read by reading P6. Also, the values of P6 (output latch) can be read by reading P6 while in output mode.

A software pull-up function is not implemented.

When using the alternate function as A16 to A21, set the pin functions via the memory expansion mode register (MM). This does not affect the PM6 register.

When a reset is input, the settings are initialized to input mode.

(2) Control register

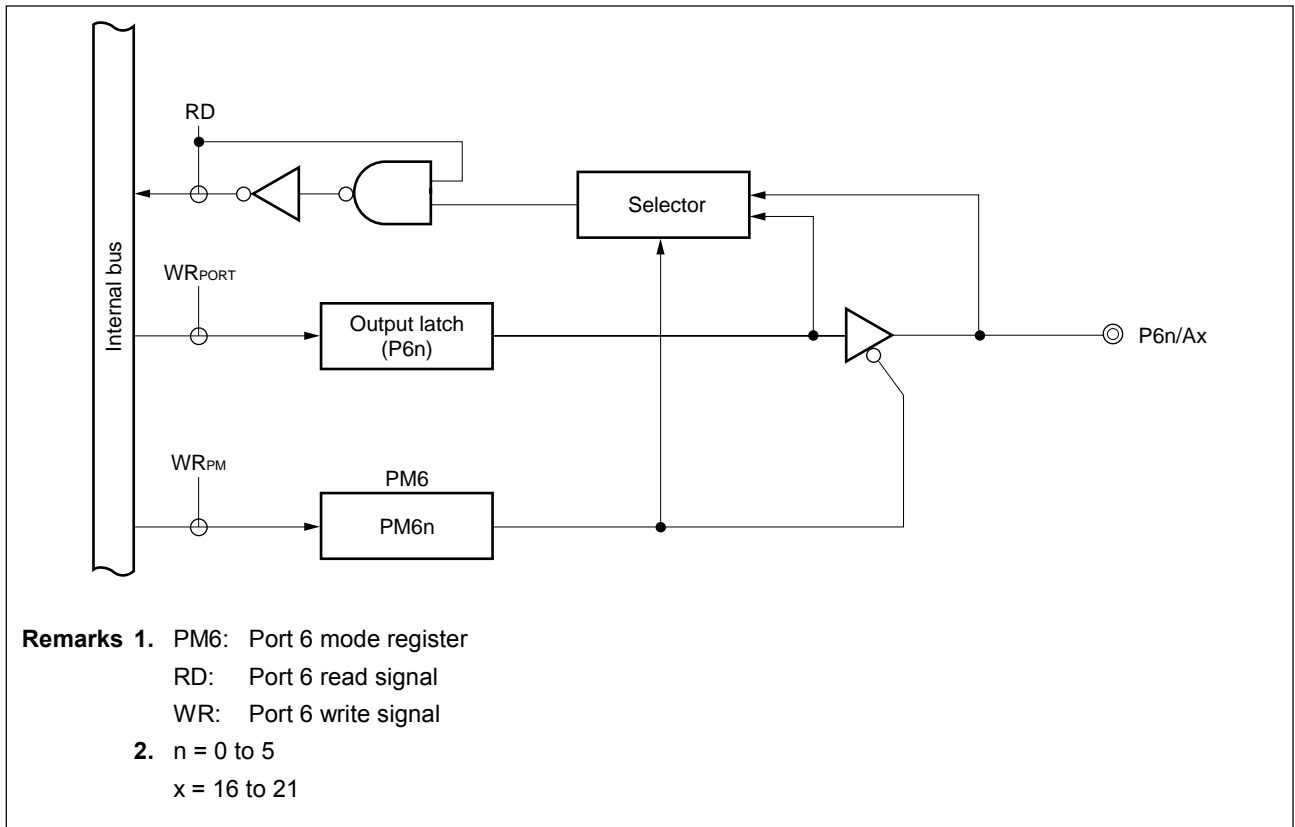
(a) Port 6 mode register (PM6)

PM6 can be read/written in 8- or 1-bit units.

After reset:	3FH	R/W	Address: FFFFF02CH					
	7	6	5	4	3	2	1	0
PM6	0	0	PM65	PM64	PM63	PM62	PM61	PM60
	PM6n	Control of I/O mode (n = 0 to 5)						
	0	Output mode						
	1	Input mode						

(3) Block diagram (Port 6)

Figure 5-8. Block Diagram of P60 to P65



5.2.7 Ports 7 and 8

Port 7 is an 8-bit input port and port 8 is a 4-bit input port. Both ports are read-only and are accessible in 8- or 1-bit units.

After reset:	Undefined	R	Address: FFFFF00EH						
	7	6	5	4	3	2	1	0	
P7	P77	P76	P75	P74	P73	P72	P71	P70	
	P7n	Pin level (n = 0 to 7)							
	0/1	Read pin level of bit n							
After reset:	Undefined	R	Address: FFFFF010H						
	7	6	5	4	3	2	1	0	
P8	0	0	0	0	P83	P82	P81	P80	
	P8n	Pin level (n = 0 to 3)							
	0/1	Read pin level of bit n							

Ports 7 and 8 include the following alternate functions.

Table 5-8. Alternate Function Pins of Ports 7 and 8

Pin Name		Alternate Function	I/O	PULL ^{Note}	Remark
Port 7	P70	ANI0	Input	No	-
	P71	ANI1			
	P72	ANI2			
	P73	ANI3			
	P74	ANI4			
	P75	ANI5			
	P76	ANI6			
	P77	ANI7			
Port 8	P80	ANI8	Input	No	-
	P81	ANI9			
	P82	ANI10			
	P83	ANI11			

Note Software pull-up function

(1) Functions of P7 and P8 pins

Port 7 is an 8-bit input-only port and port 8 is a 4-bit input-only port.

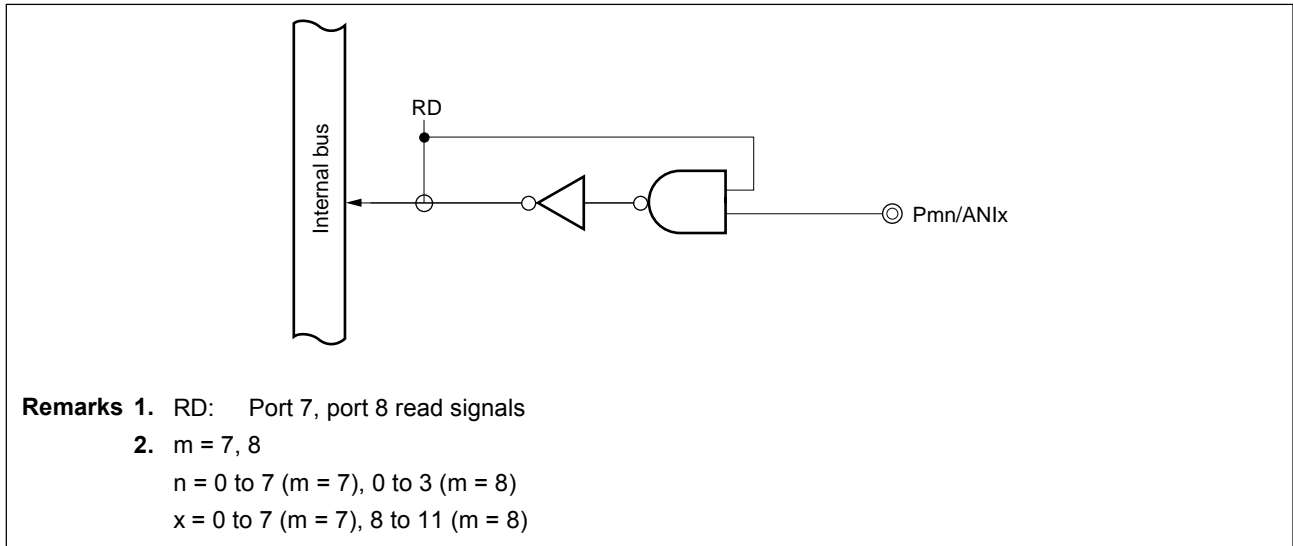
The pin statuses can be read by reading port 7 and port 8 (P7 and P8). Data cannot be written to P7 or P8.

A software pull-up function is not implemented.

Values read from pins specified as analog inputs are undefined values. Do not read values from P7 or P8 during A/D conversion.

(2) Block diagram (Ports 7 and 8)

Figure 5-9. Block Diagram of P70 to P77 and P80 to P83



5.2.8 Port 9

Port 9 is a 7-bit I/O port for which I/O settings can be controlled in 1-bit units.

After reset: 00H R/W Address: FFFFF012H

	7	6	5	4	3	2	1	0
P9	0	P96	P95	P94	P93	P92	P91	P90

P9n	Control of output data (in output mode) (n = 0 to 6)
0	Outputs 0
1	Outputs 1

Remark In input mode: When P9 is read, the pin levels at that time are read. Writing to P9 writes the values to that register. This does not affect the input pins.

 In output mode: When P9 is read, the values of P9 are read. Writing to P9 writes the values to that register, and those values are immediately output.

Port 9 includes the following alternate functions.

Table 5-9. Port 9 Alternate Function Pins

Pin Name		Alternate Function	I/O	PULL ^{Note}	Remark
Port 9	P90	$\overline{\text{LBEN}}$	I/O	No	-
	P91	$\overline{\text{UBEN}}$			
	P92	$\overline{\text{R/W}}$			
	P93	$\overline{\text{DSTB}}$			
	P94	$\overline{\text{ASTB}}$			
	P95	$\overline{\text{HLDK}}$			
	P96	$\overline{\text{HLDRQ}}$			

Note Software pull-up function

(1) Function of P9 pins

Port 9 is a 7-bit I/O port for which I/O settings can be controlled in 1-bit units. I/O settings are controlled via the port 9 mode register (PM9).

In output mode, the values set to each bit are output to port 9 (P9).

When using this port in input mode, the pin statuses can be read by reading P9. Also, the values of P9 (output latch) can be read by reading P9 while in output mode.

A software pull-up function is not implemented.

When using P9 for control signals in expansion mode, set the pin functions via the memory expansion mode register (MM).

When a reset is input, the settings are initialized to input mode.

(2) Control register

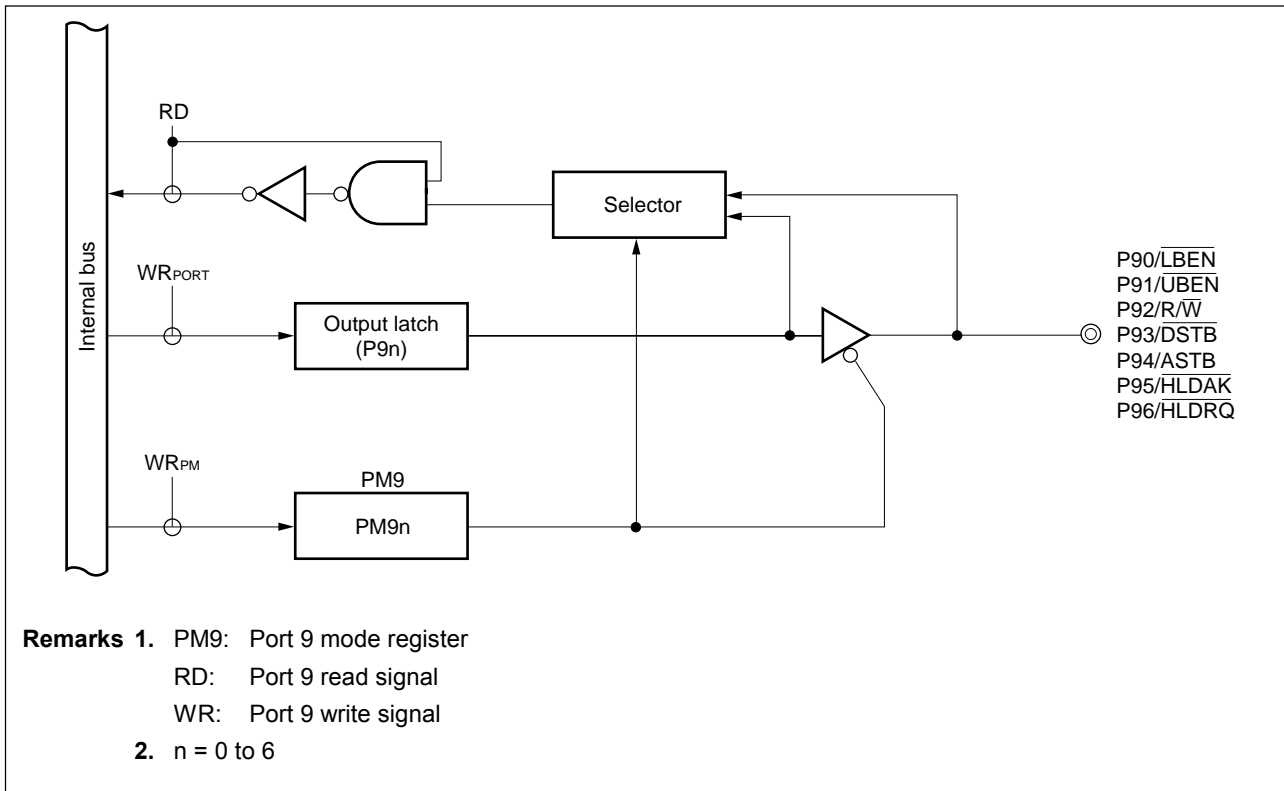
(a) Port 9 mode register (PM9)

PM9 can be read/written in 8- or 1-bit units.

After reset:	7FH	R/W	Address: FFFFF032H					
	7	6	5	4	3	2	1	0
PM9	0	PM96	PM95	PM94	PM93	PM92	PM91	PM90
	PM9n	Control of I/O mode (n = 0 to 6)						
	0	Output mode						
	1	Input mode						

(3) Block diagram (Port 9)

Figure 5-10. Block Diagram of P90 to P96



5.2.9 Port 10

Port 10 is an 8-bit I/O port for which I/O settings can be controlled in 1-bit units.

A pull-up resistor can be connected in 1-bit units (software pull-up function).

When using P100 to P107 as the KR0 to KR7 pins, noise is eliminated by an analog noise eliminator.

After reset:	00H	R/W	Address:	FFFFF014H				
	7	6	5	4	3	2	1	0
P10	P107	P106	P105	P104	P103	P102	P101	P100
	P10n	Control of output data (in output mode) (n = 0 to 7)						
	0	Outputs 0						
	1	Outputs 1						

Remark In input mode: When P10 is read, the pin levels at that time are read. Writing to P10 writes the values to that register. This does not affect the input pins.

In output mode: When P10 is read, the values of P10 are read. Writing to P10 writes the values to that register, and those values are immediately output.

Port 10 includes the following alternate functions.

Table 5-10. Port 10 Alternate Function Pins

Pin Name	Alternate Function	I/O	PULL ^{Note}	Remark
Port 10	P100	I/O	Yes	Analog noise elimination
	P101			
	P102			
	P103			
	P104			
	P105			
	P106			
	P107			

Note Software pull-up function

(1) Function of P10 pins

Port 10 is an 8-bit I/O port for which I/O settings can be controlled in 1-bit units. I/O settings are controlled via the port 10 mode register (PM10).

In output mode, the values set to each bit are output to port 10 (P10).

When using this port in input mode, the pin statuses can be read by reading P10. Also, the values of P10 (output latch) can be read by reading P10 while in output mode.

A pull-up resistor can be connected in 1-bit units when specified via pull-up resistor option register 10 (PU10).

When used as KR0 to KR7 pins, noise is eliminated by an analog noise eliminator.

Clear P10 and the PM10 register to 0 when using alternate-function pins as outputs. The logical sum (ORed result) of the port output and the alternate-function pin is output from the pins.

When a reset is input, the settings are initialized to input mode.

(2) Control registers

(a) Port 10 mode register (PM10)

PM10 can be read/written in 8- or 1-bit units.

After reset:	FFH	R/W	Address: FFFF034H					
	7	6	5	4	3	2	1	0
PM10	PM107	PM106	PM105	PM104	PM103	PM102	PM101	PM100
	PM10n	Control of I/O mode (n = 0 to 7)						
	0	Output mode						
	1	Input mode						

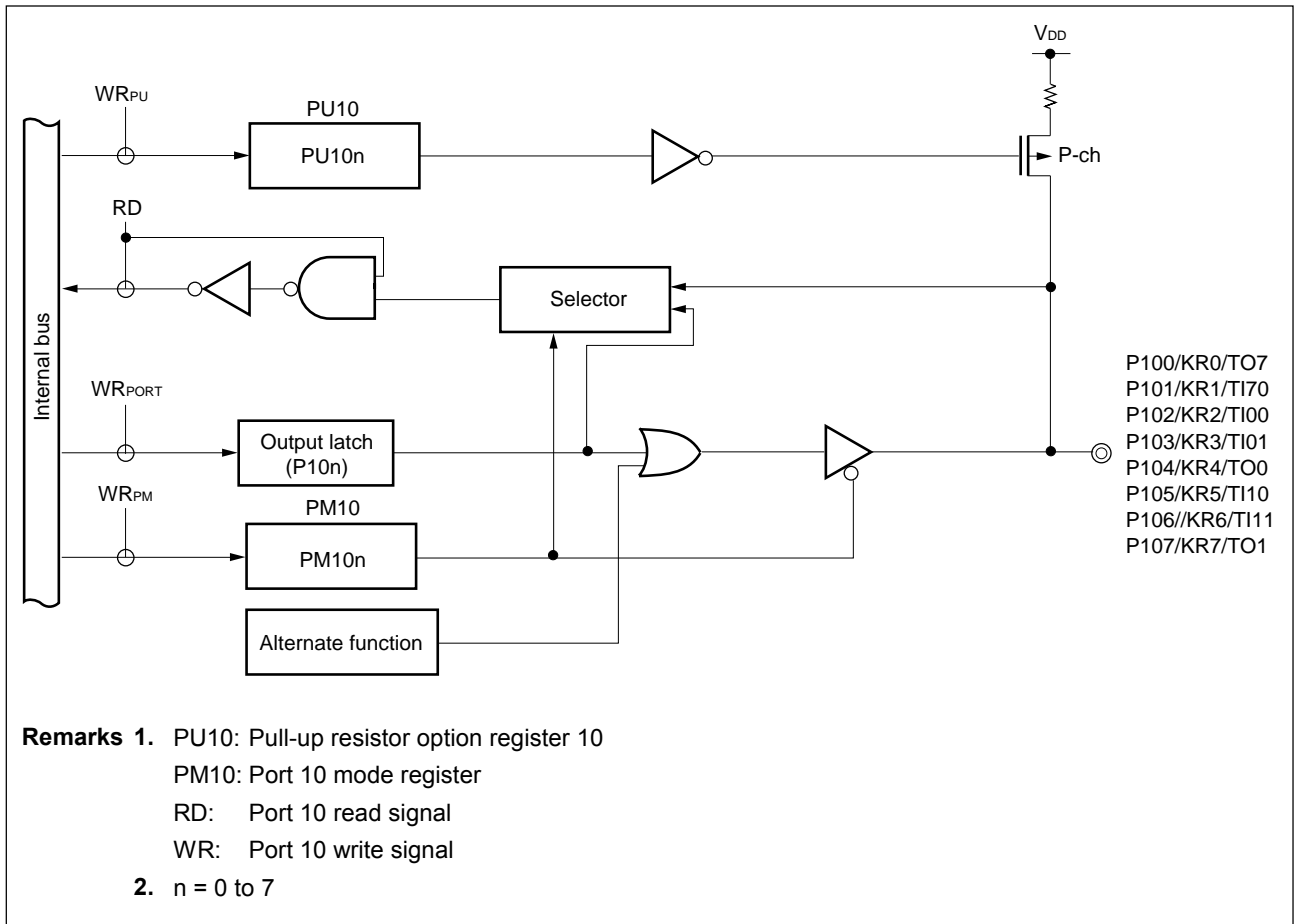
(b) Pull-up resistor option register 10 (PU10)

PU10 can be read/written in 8- or 1-bit units.

After reset:	00H	R/W	Address: FFFF094H					
	7	6	5	4	3	2	1	0
PU10	PU107	PU106	PU105	PU104	PU103	PU102	PU101	PU100
	PU10n	Control of on-chip pull-up resistor connection (n = 0 to 7)						
	0	Do not connect						
	1	Connect						

(3) Block diagram (Port 10)

Figure 5-11. Block Diagram of P100 to P107



5.2.10 Port 11

Port 11 is an 8-bit I/O port for which I/O settings can be controlled in 1-bit units.

P11 can be read/written in 8- or 1-bit units.

Turning on and off the wait function and switching between alternate pins and port pins can be performed via the port alternate-function control register (PAC) (CANTX2 and CANRX2 are available only for the μ PD703079Y and 70F3079Y).

After reset:	00H	R/W						Address:	FFFFF016H
	7	6	5	4	3	2	1	0	
P11	P117	P116	P115	P114	P113	P112	P111	P110	
	P11n	Control of output data (in output mode) (n = 0 to 7)							
	0	Outputs 0							
	1	Outputs 1							

Remark In input mode: When P11 is read, the pin levels at that time are read. Writing to P11 writes the values to that register. This does not affect the input pins.

In output mode: When P11 is read, the values of P11 are read. Writing to P11 writes the values to that register, and those values are immediately output.

Port 11 includes the following alternate functions.

Table 5-11. Port 11 Alternate Function Pins

Pin Name	Alternate Function	I/O	PULL ^{Note 1}	Remark
Port 11	P110	WAIT	No	-
	P111	-		
	P112	-		
	P113	-		
	P114	CANTX1		
	P115	CANRX1		
	P116	CANTX2 ^{Note 2}		
	P117	CANRX2 ^{Note 2}		

- Notes**
1. Software pull-up function
 2. Only for the μ PD703079Y and 70F3079Y

(1) Function of P11 pins

Port 11 is an 8-bit port for which I/O settings can be controlled in 1-bit units. I/O settings are controlled via the port 11 mode register (PM11).

In output mode, the values set to each bit are output to port 11 (P11).

When using this port in input mode, the pin statuses can be read by reading P11. Also, the values of P11 (output latch) can be read by reading P11 while in output mode.

Turning on and off the wait function and switching between alternate pins and port pins can be performed via the port alternate-function control register (PAC).

When a reset is input, the settings are initialized to input mode.

(2) Control registers

(a) Port 11 mode register (PM11)

PM11 can be read/written in 8- or 1-bit units.

After reset:	FFH	R/W	Address: FFFF036H					
	7	6	5	4	3	2	1	0
PM11	PM117	PM116	PM115	PM114	PM113	PM112	PM111	PM110
	PM11n	Control of I/O mode (n = 0 to 7)						
	0	Output mode						
	1	Input mode						

(b) Port alternate-function control register (PAC)

PAC can be read/written in 8- or 1-bit units.

After reset:	00H	R/W	Address: FFFF040H					
	7	6	5	4	3	2	1	0
PAC	PAC117 ^{Note}	PAC116 ^{Note}	PAC115	PAC114	0	0	0	WAC
	WAC	Control of wait function						
	0	Wait function OFF						
	1	Wait function ON						
	PAC11n	Control of port alternate function (n = 4 to 7)						
	0	Port function						
	1	Alternate function						

Note Bits PAC117 and PAC116 are available only for the μ PD703079Y and 70F3079Y. Set bits 7 and 6 to 0 when using the μ PD703078Y.

(3) Block diagram (Port 11)

★

Figure 5-12. Block Diagram of P110 and P114 to P117

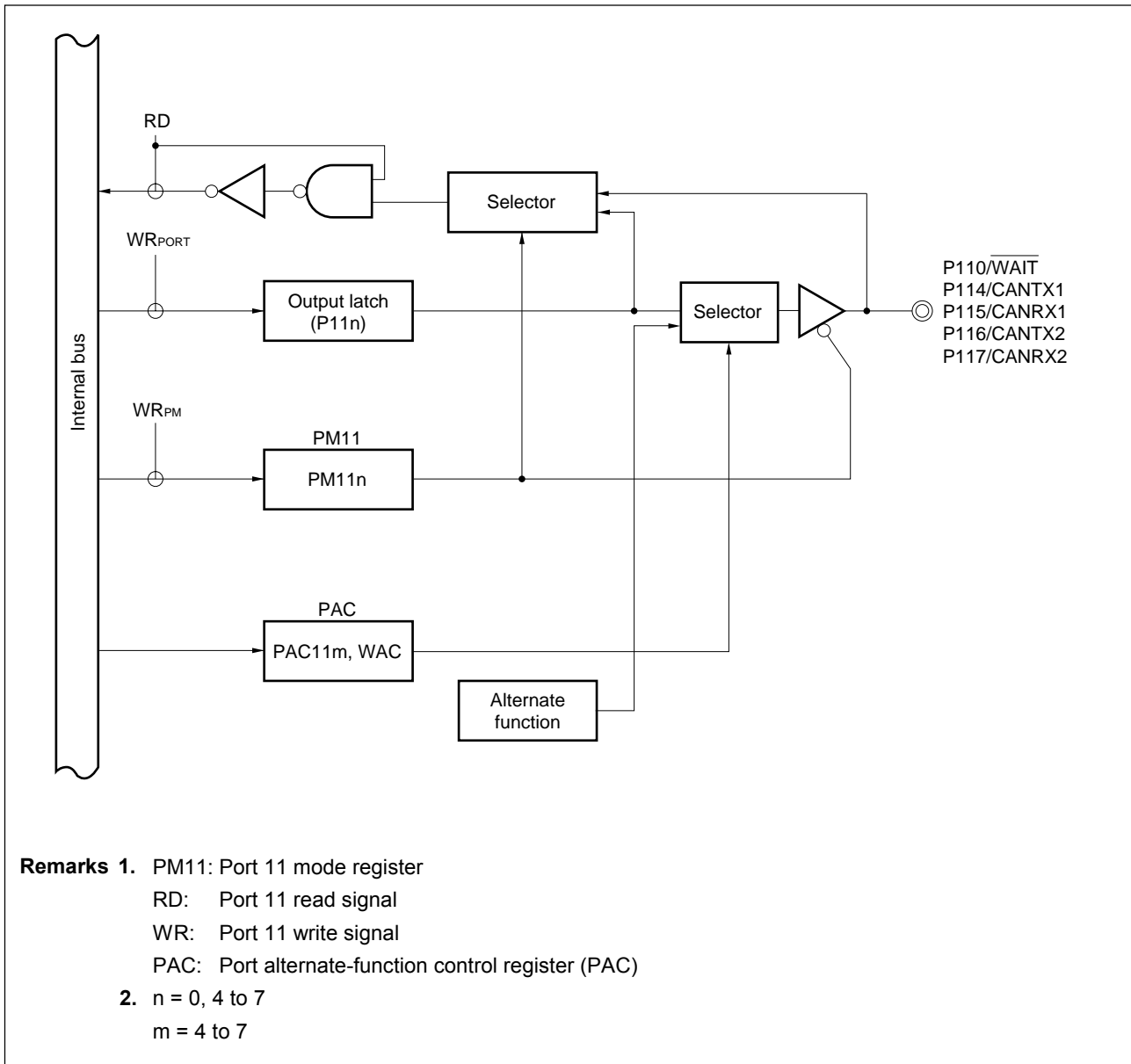
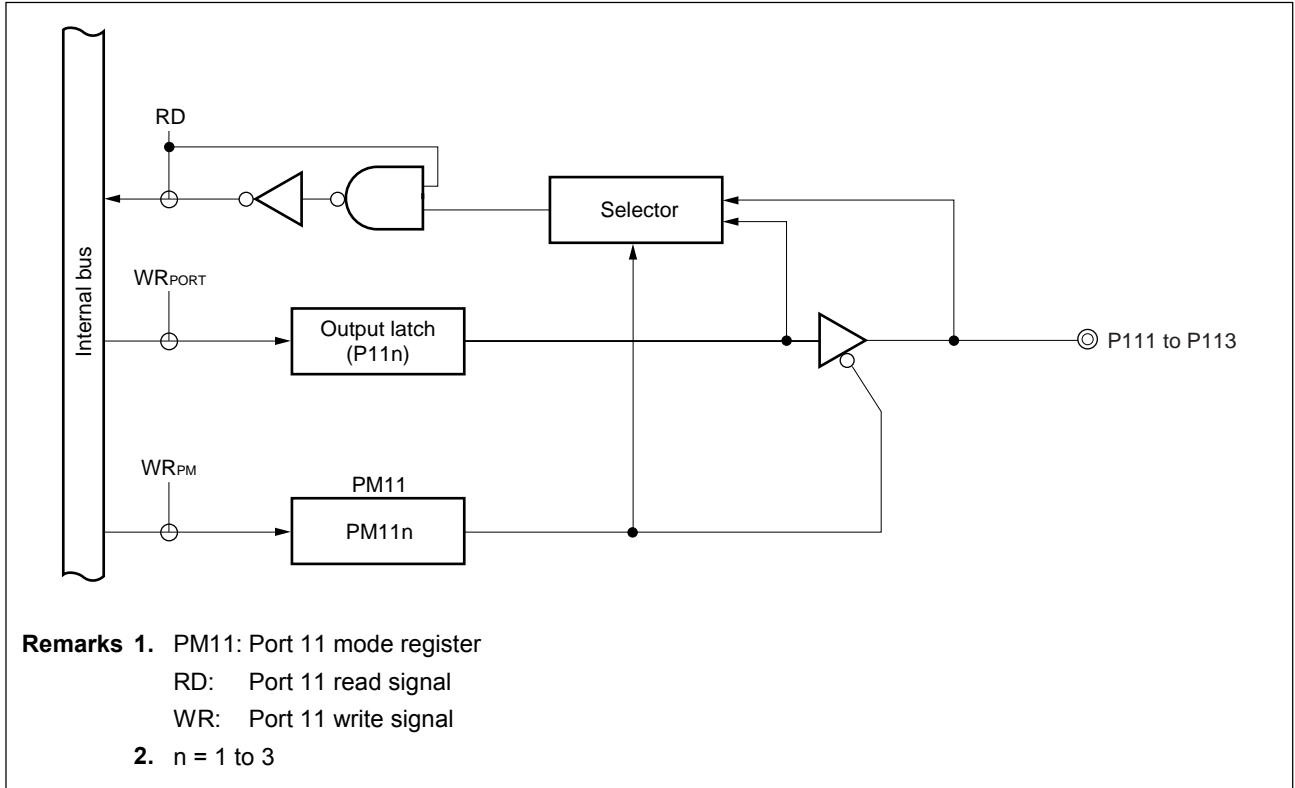


Figure 5-13. Block Diagram of P111 to P113



5.3 Setting When Port Pin Is Used for Alternate Function

When a port pin is used for an alternate function, set the port n mode register (PM0 to PM6 and PM9 to PM11) and output latch as shown in Table 5-12 below.

Table 5-12. Setting When Port Pin Is Used for Alternate Function (1/4)

Pin Name	Alternate Function		PMx Bit of PMn Register	Pnx Bit of Pn Register	Other Bits (Register)
	Function Name	I/O			
P00	NMI	Input	PM00 = 1	Setting not needed for P00	—
P01	INTP0	Input	PM01 = 1	Setting not needed for P01	—
P02	INTP1	Input	PM02 = 1	Setting not needed for P02	—
P03	INTP2	Input	PM03 = 1	Setting not needed for P03	—
P04	INTP3	Input	PM04 = 1	Setting not needed for P04	—
P05	INTP4	Input	PM05 = 1	Setting not needed for P05	—
	ADTRG	Input			
P06	INTP5	Input	PM06 = 1	Setting not needed for P06	—
P07	INTP6	Input	PM07 = 1	Setting not needed for P07	—
P10	SI0	Input	PM10 = 1	Setting not needed for P10	—
	SDA0	I/O	PM10 = 0	P10 = 0	
P11	SO0	Output	PM11 = 0	P11 = 0	—
P12	SCK0	Input	PM12 = 1	Setting not needed for P12	—
		Output	PM12 = 0	P12 = 0	
	SCL0	I/O			
P13	SI1	Input	PM13 = 1	Setting not needed for P13	—
	RXD0	Input			
P14	SO1	Output	PM14 = 0	P14 = 0	—
	TXD0	Output			
P15	SCK1	Input	PM15 = 1	Setting not needed for P15	—
		Output	PM15 = 0	P15 = 0	
	ASCK0	Input	PM15 = 1	Setting not needed for P15	

Table 5-12. Setting When Port Pin Is Used for Alternate Function (2/4)

Pin Name	Alternate Function		PMnx Bit of PMn Register	Pnx Bit of Pn Register	Other Bits (Register)
	Function Name	I/O			
P20	SI3	Input	PM20 = 1	Setting not needed for P20	—
	RXD1	Input			
P21	SO2	Output	PM21 = 0	P21 = 0	—
	TXD1	Output			
P22	SCK3	Input	PM22 = 1	Setting not needed for P22	—
		Output	PM22 = 0	P22 = 0	
	ASCK1	Input	PM22 = 1	Setting not needed for P22	
P23	SI4	Input	PM23 = 1	Setting not needed for P23	—
P24	SO4	Output	PM24 = 0	P24 = 0	—
P25	SCK4	Input	PM25 = 1	Setting not needed for P25	—
		Output	PM25 = 0	P25 = 0	
P30	TI2	Input	PM30 = 1	Setting not needed for P30	—
	TO2	Output	PM30 = 0	P30 = 0	
P31	TI3	Input	PM31 = 1	Setting not needed for P31	—
	TO3	Output	PM31 = 0	P31 = 0	
P32	TI4	Input	PM32 = 1	Setting not needed for P32	—
	TO4	Output	PM32 = 0	P32 = 0	
P33	TI5	Input	PM33 = 1	Setting not needed for P33	—
	TO5	Output	PM33 = 0	P33 = 0	
P34	TI71	Input	PM34 = 1	Setting not needed for P34	—
	VM45	Output	PM34 = 0	P34 = 0	

Note Refer to 14.3 (2) VM45 control register (VM45C).

Table 5-12. Setting When Port Pin Is Used for Alternate Function (3/4)

Pin Name	Alternate Function		PM _n x Bit of PM _n Register	P _n x Bit of P _n Register	Other Bits (Register)
	Function Name	I/O			
P40 to P47	AD0 to AD7	I/O	Setting not needed for PM40 to PM47	Setting not needed for P40 to P47	Note
P50 to P57	AD8 to AD15	I/O	Setting not needed for PM50 to PM57	Setting not needed for P50 to P57	Note
P60 to P65	A16 to A21	Output	Setting not needed for PM60 to PM65	Setting not needed for P60 to P65	Note
P70 to P77	ANI0 to ANI7	Input	None	Setting not needed for P70 to P77	—
P80 to P83	ANI8 to ANI11	Input	None	Setting not needed for P80 to P83	—
P90	$\overline{\text{LBEN}}$	Output	Setting not needed for PM90	Setting not needed for P90	Note
P91	$\overline{\text{UBEN}}$	Output	Setting not needed for PM91	Setting not needed for P91	Note
P92	$\overline{\text{R/W}}$	Output	Setting not needed for PM92	Setting not needed for P92	Note
P93	$\overline{\text{DSTB}}$	Output	Setting not needed for PM93	Setting not needed for P93	Note
P94	$\overline{\text{ASTB}}$	Output	Setting not needed for PM94	Setting not needed for P94	Note
P95	$\overline{\text{HLDK}}$	Output	Setting not needed for PM95	P95 = 1	Note
P96	$\overline{\text{HLDK}}$	Input	Setting not needed for PM96	P96 = 1	Note

Note Refer to 3.4.6 (1) Memory expansion mode register (MM).

Table 5-12. Setting When Port Pin Is Used for Alternate Function (4/4)

Pin Name	Alternate Function		PMnx Bit of PMn Register	Pnx Bit of Pn Register	Other Bits (Register)
	Function Name	I/O			
P100	KR0	Input	PM100 = 1	Setting not needed for P100 P100 = 0	–
	TO7	Output	PM100 = 0		
P101	KR1	Input	PM101 = 1	Setting not needed for P101	–
	TI70	Input			
P102	KR2	Input	PM102 = 1	Setting not needed for P102	–
	TI00	Input			
P103	KR3	Input	PM103 = 1	Setting not needed for P103	–
	TI01	Input			
P104	KR4	Input	PM104 = 1	Setting not needed for P104	–
	TO0	Output	PM104 = 0	P104 = 0	
P105	KR5	Input	PM105 = 1	Setting not needed for P105	–
	TI10	Input			
P106	KR6	Input	PM106 = 1	Setting not needed for P106	–
	TI11	Input			
P107	KR7	Input	PM107 = 1	Setting not needed for P107	–
	TO1	Output	PM107 = 0	P107 = 0	
P110	WAIT	Input	PM110 = 1	Setting not needed for P110	WAC = 1 (PAC)
P114	CANTX1	Output	PM114 = 0	P114 = 0	PAC114 = 1 (PAC)
P115	CANRX1	Input	PM115 = 1	P115 = 0	PAC115 = 1 (PAC)
P116	CANTX2 ^{Note}	Output	PM116 = 0	P116 = 0	PAC116 = 1 (PAC)
P117	CANRX2 ^{Note}	Input	PM117 = 1	P117 = 0	PAC117 = 1 (PAC)

Note Only for the μ PD703079Y and 70F3079Y.

- Cautions 1.** When changing the output level of port 0 by setting the port function output mode of port 0, the interrupt request flag will be set because port 0 also has an alternate function as external interrupt request input. Therefore, be sure to set the corresponding interrupt mask flag to 1 before using a port 0 pin as an output pin.
- 2.** When using the I²C bus mode, be sure to specify N-ch open-drain output for the SDA0/P10 and SCL0/P12 pins by setting the port 1 function register (PF1).

Remark PMnx bit of PMn register and Pnx bit of Pn register

n: 0 (x = 0 to 7) n: 1 (x = 0 to 5) n: 2 (x = 0 to 7) n: 3 (x = 0 to 4) n: 4 (x = 0 to 7)
n: 5 (x = 0 to 7) n: 6 (x = 0 to 5) n: 7 (x = 0 to 7) n: 8 (x = 0 to 3) n: 9 (x = 0 to 6)
n: 10 (x = 0 to 7) n: 11 (x = 0 to 7)

CHAPTER 6 BUS CONTROL FUNCTION

The V850/SF1 is provided with an external bus interface function by which external memories such as ROM and RAM, and I/O can be connected.

6.1 Features

- Address bus
- 16-bit data bus
- Able to be connected to external devices via pins with alternate-functions as ports
- Wait function
 - Programmable wait function, capable of inserting up to 3 wait states per 2 blocks
 - External wait control through $\overline{\text{WAIT}}$ input pin
- Idle state insertion function
- Bus mastership arbitration function
- Bus hold function

6.2 Bus Control Pins and Control Register

6.2.1 Bus control pins

The following pins are used for interfacing with external devices.

Table 6-1. Bus Control Pins

External Bus Interface Function	Corresponding Port (Pins)
Address/data bus (AD0 to AD7)	Port 4 (P40 to P47)
Address/data bus (AD8 to AD15)	Port 5 (P50 to P57)
Address bus (A16 to A21)	Port 6 (P60 to P65)
Read/write control ($\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$, $\overline{\text{R/W}}$, $\overline{\text{DSTB}}$)	Port 9 (P90 to P93)
Address strobe (ASTB)	Port 9 (P94)
Bus hold control ($\overline{\text{HLDRQ}}$, $\overline{\text{HLDAK}}$)	Port 9 (P95, P96)
External wait control ($\overline{\text{WAIT}}$)	Port 11 (P110)

The bus interface function of each pin is enabled by setting the memory expansion mode register (MM). For details of external bus interface operating mode specification, refer to **3.4.6 (1) Memory expansion mode register (MM)**.

6.3 Bus Access

6.3.1 Number of access clocks

The number of basic clocks necessary for accessing each resource is as follows.

Table 6-2. Number of Access Clocks

Bus Cycle Type	Peripheral I/O (Bus Width)			
	Internal ROM (32 Bits)	Internal RAM (32 Bits)	Peripheral I/O (16 Bits)	External Memory (16 Bits)
Instruction fetch	1	3	Disabled	3 + n
Operand data access	3	1	3	3 + n

- Remarks**
1. Unit: Clock/access
 2. n: Number of wait insertions

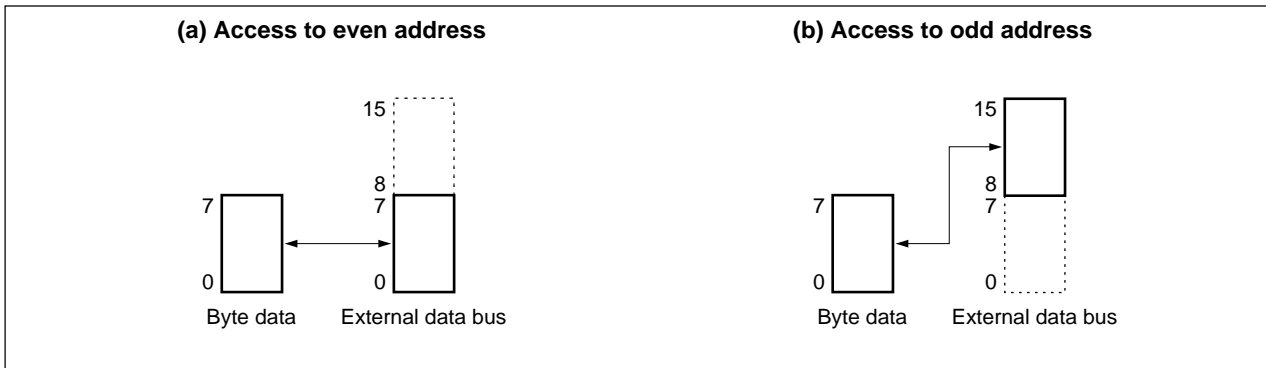
6.3.2 Bus width

The CPU carries out peripheral I/O access and external memory access in 8-bit, 16-bit, or 32-bit units. The following shows the operation for each access.

(1) Byte access (8 bits)

Byte access is divided into two types: access to even addresses and access to odd addresses.

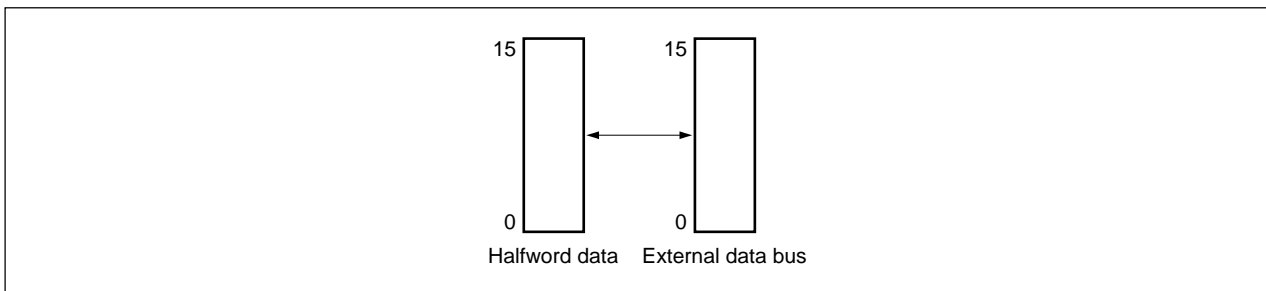
Figure 6-1. Byte Access (8 Bits)



(2) Halfword access (16 bits)

In halfword access to external memory, data is handled as is because the data bus is fixed to 16 bits.

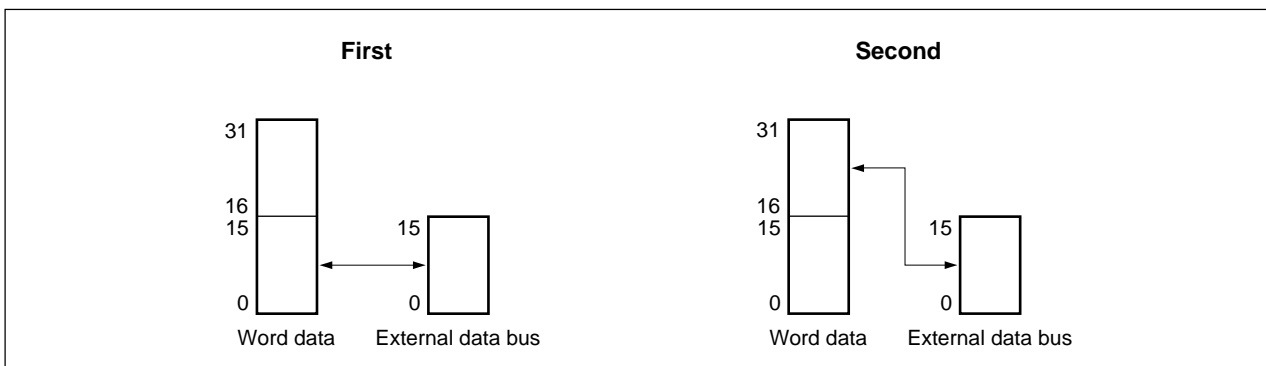
Figure 6-2. Halfword Access (16 Bits)



(3) Word access (32 bits)

In word access to external memory, the lower halfword is accessed first and then the higher halfword is accessed.

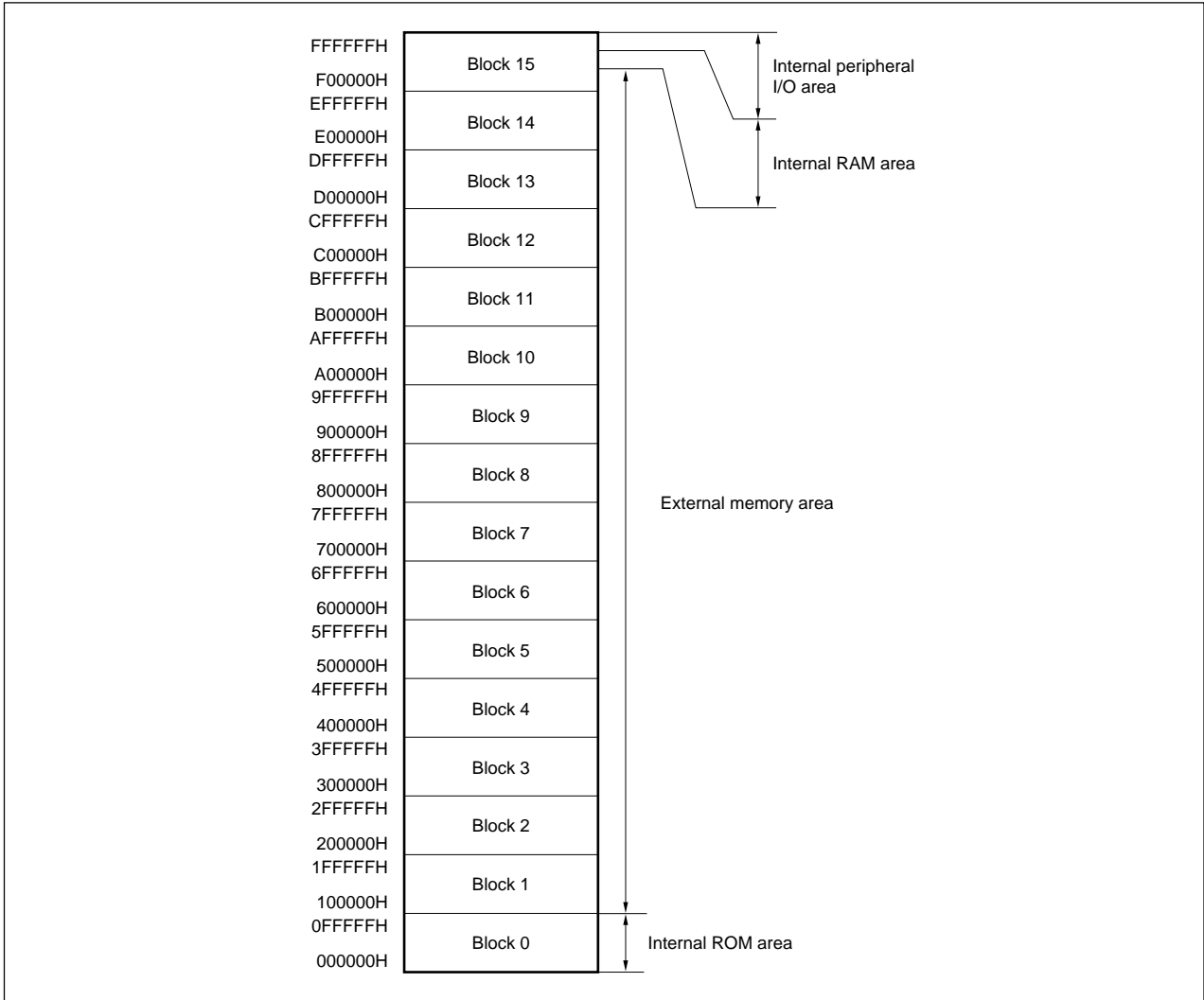
Figure 6-3. Word Access (32 Bits)



6.4 Memory Block Function

The 16 MB memory space is divided into memory blocks of 1 MB units. The programmable wait function and bus cycle operation mode can be independently controlled for every two memory blocks.

Figure 6-4. Memory Block



6.5 Wait Function

6.5.1 Programmable wait function

To facilitate interfacing with low-speed memories and I/O devices, up to 3 data wait states can be inserted in a bus cycle that starts every two memory blocks.

The number of wait states can be programmed by using the data wait control register (DWC). Immediately after the system has been reset, three data wait states are automatically programmed for insertion in all memory blocks.

(1) Data wait control register (DWC)

This register can be read/written in 16-bit units.

After reset: FFFFH		R/W		Address: FFFF060H												
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DWC	DW71	DW70	DW61	DW60	DW51	DW50	DW41	DW40	DW31	DW30	DW21	DW20	DW11	DW10	DW01	DW00
DWn1	DWn0	Number of wait states to be inserted														
0	0	0														
0	1	1														
1	0	2														
1	1	3														
n	Blocks into which wait states are inserted															
0	Blocks 0/1															
1	Blocks 2/3															
2	Blocks 4/5															
3	Blocks 6/7															
4	Blocks 8/9															
5	Blocks 10/11															
6	Blocks 12/13															
7	Blocks 14/15															

Block 0 is reserved for the internal ROM area. It is not subject to programmable wait control, regardless of the setting of DWC, and is always accessed without wait states.

The internal RAM area of block 15 is not subject to programmable wait control and is always accessed without wait states. The on-chip peripheral I/O area of this block is not subject to programmable wait control, either. The wait control is dependent upon the execution of each peripheral function.

6.5.2 External wait function

When an extremely slow device, I/O, or asynchronous system is connected, any number of wait states can be inserted in a bus cycle by sampling the external wait pin ($\overline{\text{WAIT}}$) to synchronize with the external device.

The external wait signal is for data wait only, and does not affect the access times of the internal ROM, internal RAM, and on-chip peripheral I/O areas, similar to programmable wait.

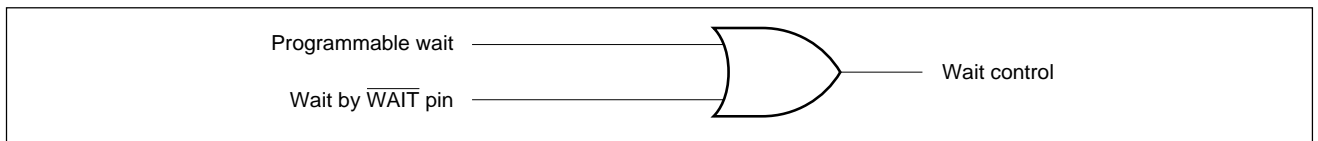
The external $\overline{\text{WAIT}}$ signal can be input asynchronously to CLKOUT and is sampled at the falling edge of the clock in the T2 and TW states of the bus cycle. If the setup/hold time at the sampling timing is not satisfied, the wait state may or may not be inserted in the next state.

Caution The P110 pin and $\overline{\text{WAIT}}$ pin are alternate-function pins. Set bit 0 (WAC) of the port alternate function control register (PAC) to 1 when these pins are used for the wait function.

6.5.3 Relationship between programmable wait and external wait

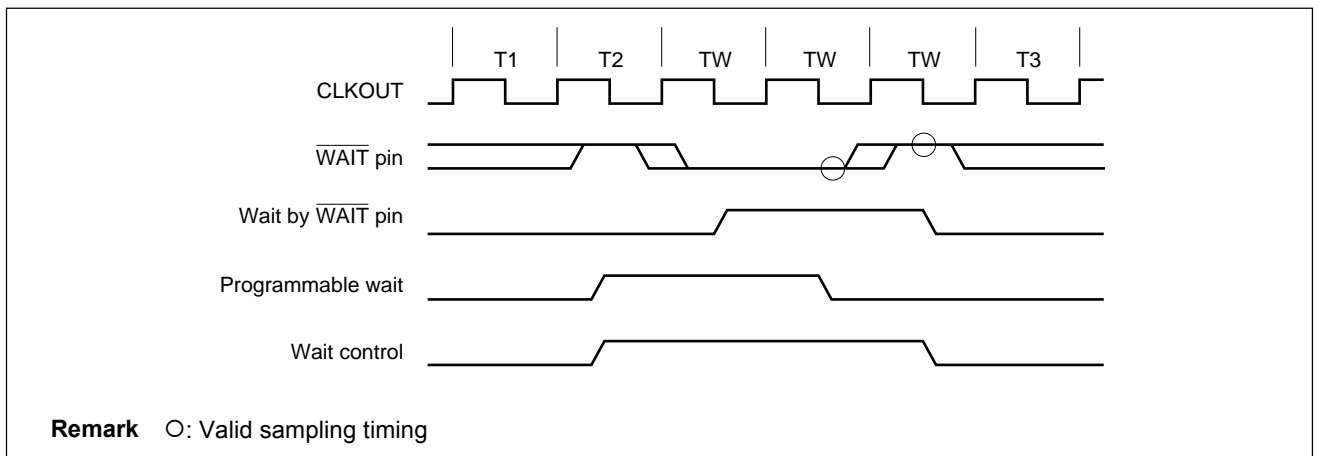
A wait cycle is inserted as a result of an OR operation between the wait cycle specified by the set value of a programmable wait and the wait cycle controlled by the $\overline{\text{WAIT}}$ pin. In other words, the number of wait cycles is determined by whichever has more cycles.

Figure 6-5. Wait Control



For example, if the number of programmable waits and the timing of the $\overline{\text{WAIT}}$ pin input signal are as illustrated below, three wait states will be inserted in the bus cycle.

Figure 6-6. Example of Inserting Wait States



6.6 Idle State Insertion Function

To facilitate interfacing with low-speed memory devices and meeting the data output float delay time on memory read accesses every two blocks, one idle state (TI) can be inserted into the current bus cycle after the T3 state. The bus cycle following continuous bus cycles starts after one idle state.

Specifying insertion of the idle state is programmable by using the bus cycle control register (BCC).

Immediately after the system has been reset, idle state insertion is automatically programmed for all memory blocks.

(1) Bus cycle control register (BCC)

This register can be read/written in 16-bit units.

After reset: AAAAH R/W Address: FFFF062H

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCC	BC71	0	BC61	0	BC51	0	BC41	0	BC31	0	BC21	0	BC11	0	BC01	0

BCn1	Idle state insert specification
0	Not inserted
1	Inserted

n	Blocks into which idle state is inserted
0	Blocks 0/1
1	Blocks 2/3
2	Blocks 4/5
3	Blocks 6/7
4	Blocks 8/9
5	Blocks 10/11
6	Blocks 12/13
7	Blocks 14/15

Block 0 is reserved for the internal ROM area and therefore no idle state can be specified.

The internal RAM area and on-chip peripheral I/O area of block 15 are not subject to insertion of an idle state.

Be sure to set bits 0, 2, 4, 6, 8, 10, 12, and 14 to 0. If these bits are set to 1, the operation is not guaranteed.

6.7 Bus Hold Function

6.7.1 Outline of function

When the MM3 bit of the memory expansion mode register (MM) is set (1), the $\overline{\text{HLDRQ}}$ and $\overline{\text{HLDK}}$ pin functions of P95 and P96 become valid.

When the $\overline{\text{HLDRQ}}$ pin becomes active (low) indicating that another bus master is requesting acquisition of the bus, the external address/data bus and strobe pins go into a high-impedance state, and the bus is released (bus hold status). When the $\overline{\text{HLDK}}$ pin becomes inactive (high) indicating that the request for the bus is cleared, these pins are driven again.

During the bus hold period, the internal operation continues until the next external memory access.

The bus hold status can be recognized by the $\overline{\text{HLDK}}$ pin becoming active (low).

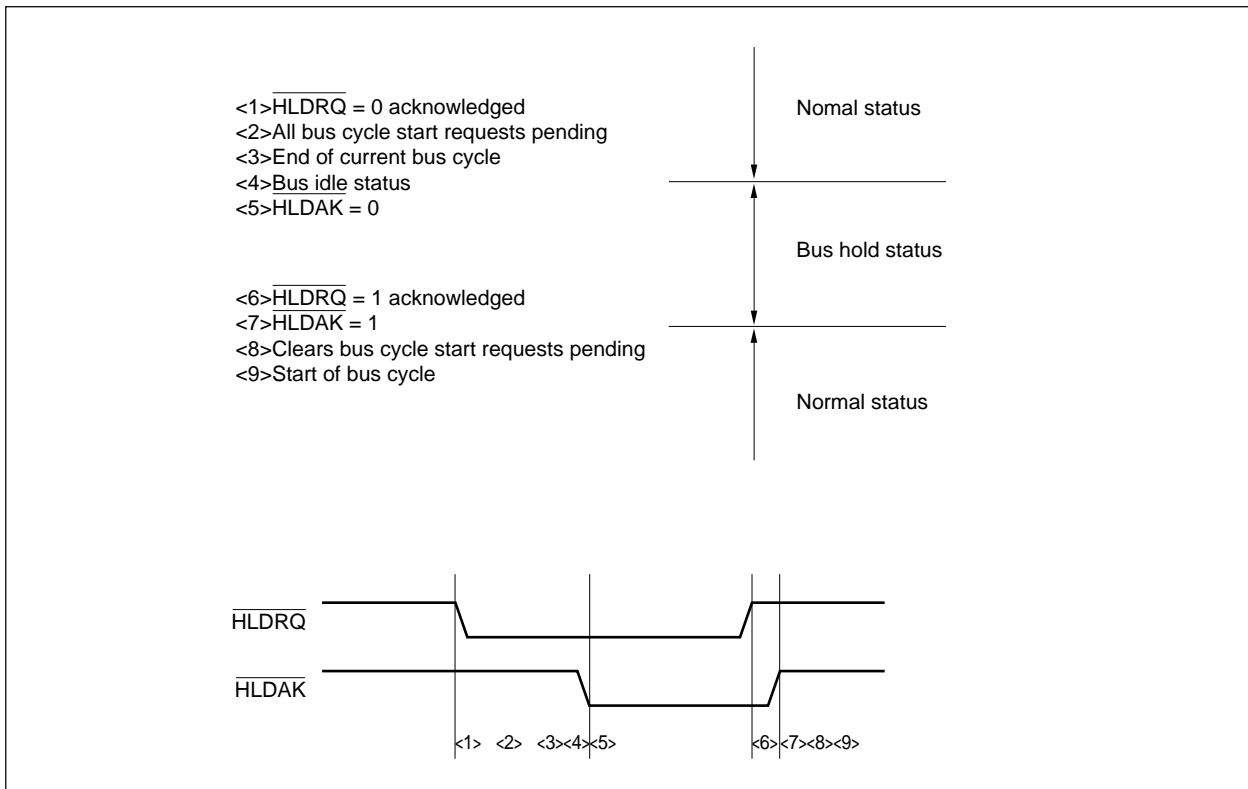
This feature can be used to design a system where two or more bus masters exist, such as when a multi-processor configuration is used and when a DMA controller is connected.

Bus hold requests are not acknowledged between the first and the second word access, nor between a read access and a write access in the read modify write access of a bit manipulation instruction.

6.7.2 Bus hold procedure

The procedure of the bus hold function is illustrated below.

Figure 6-7. Bus Hold Procedure



6.7.3 Operation in power save mode

In the STOP or IDLE mode, the system clock is stopped. Consequently, the bus hold status is not set even if the $\overline{\text{HLDQR}}$ pin becomes active.

In the HALT mode, the $\overline{\text{HLDAR}}$ pin immediately becomes active when the $\overline{\text{HLDQR}}$ pin becomes active, and the bus hold status is set. When the $\overline{\text{HLDQR}}$ pin becomes inactive, the $\overline{\text{HLDAR}}$ pin becomes inactive. As a result, the bus hold status is cleared, and the HALT mode is set again.

6.8 Bus Timing

The V850/SF1 can execute read/write control for an external device using the following mode.

- Mode using $\overline{\text{DSTB}}$, $\overline{\text{R}/\overline{\text{W}}}$, $\overline{\text{LBEN}}$, $\overline{\text{UBEN}}$, and ASTB signals

Figure 6-8. Memory Read (1/4)

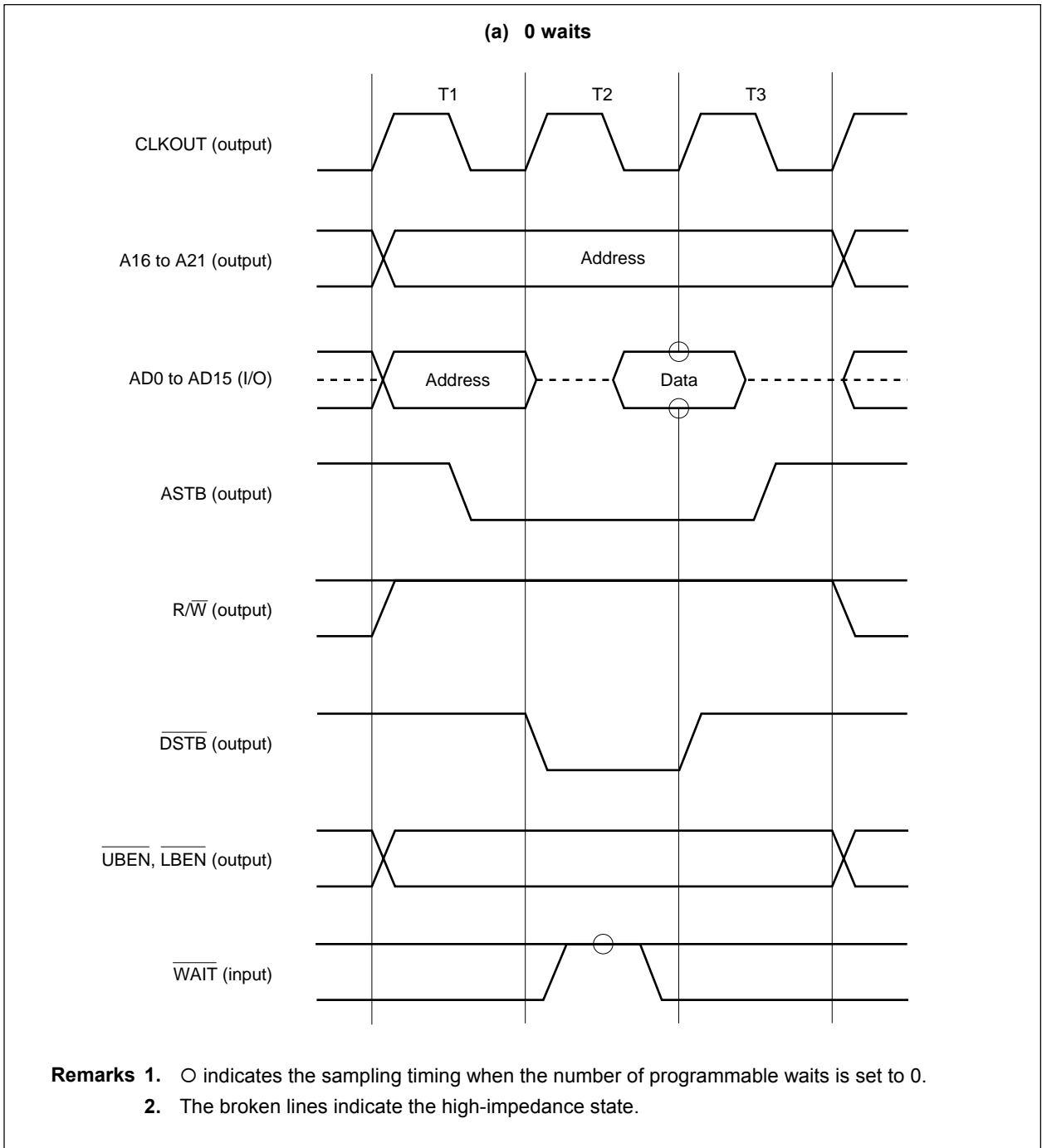


Figure 6-8. Memory Read (2/4)

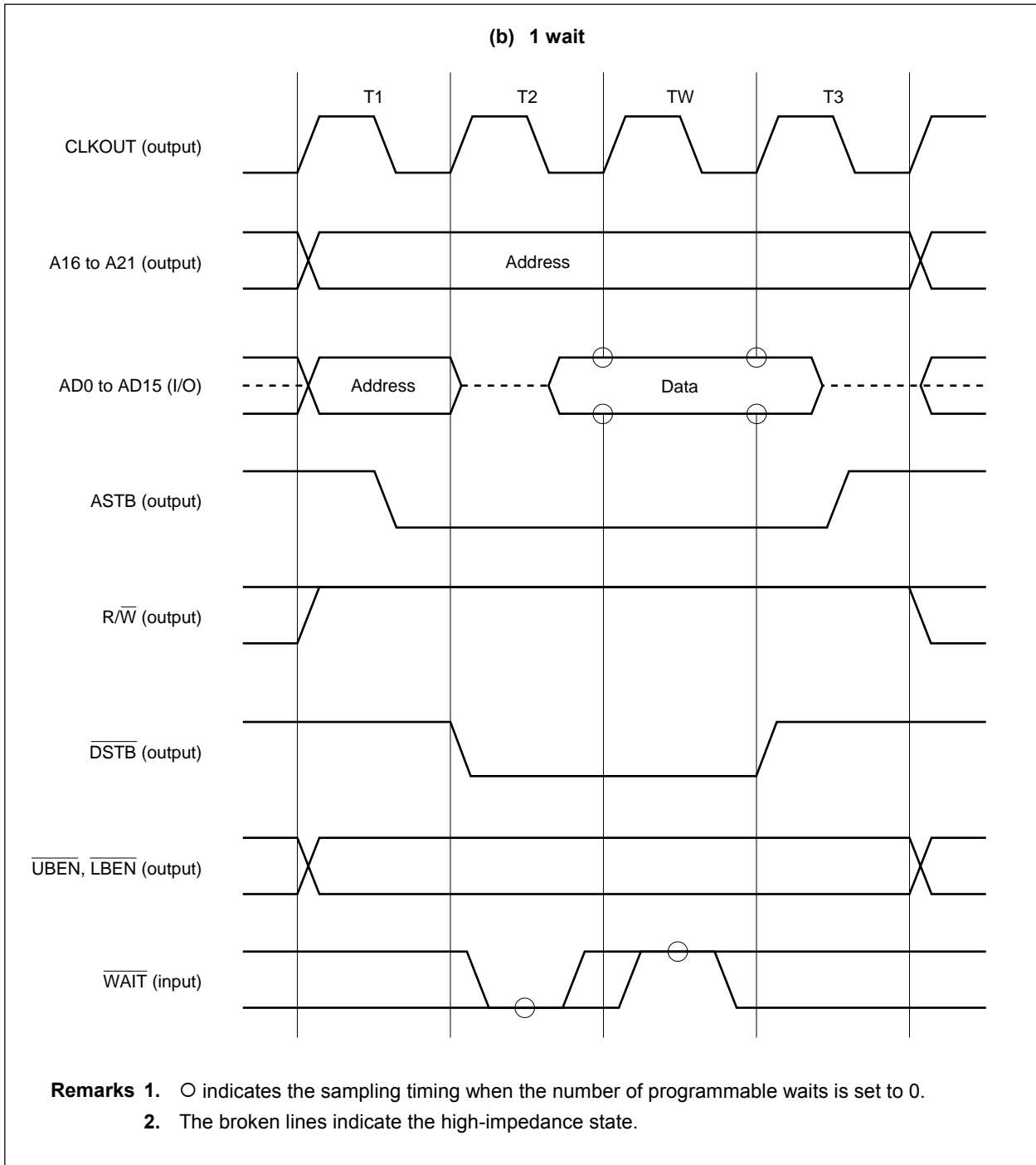


Figure 6-8. Memory Read (3/4)

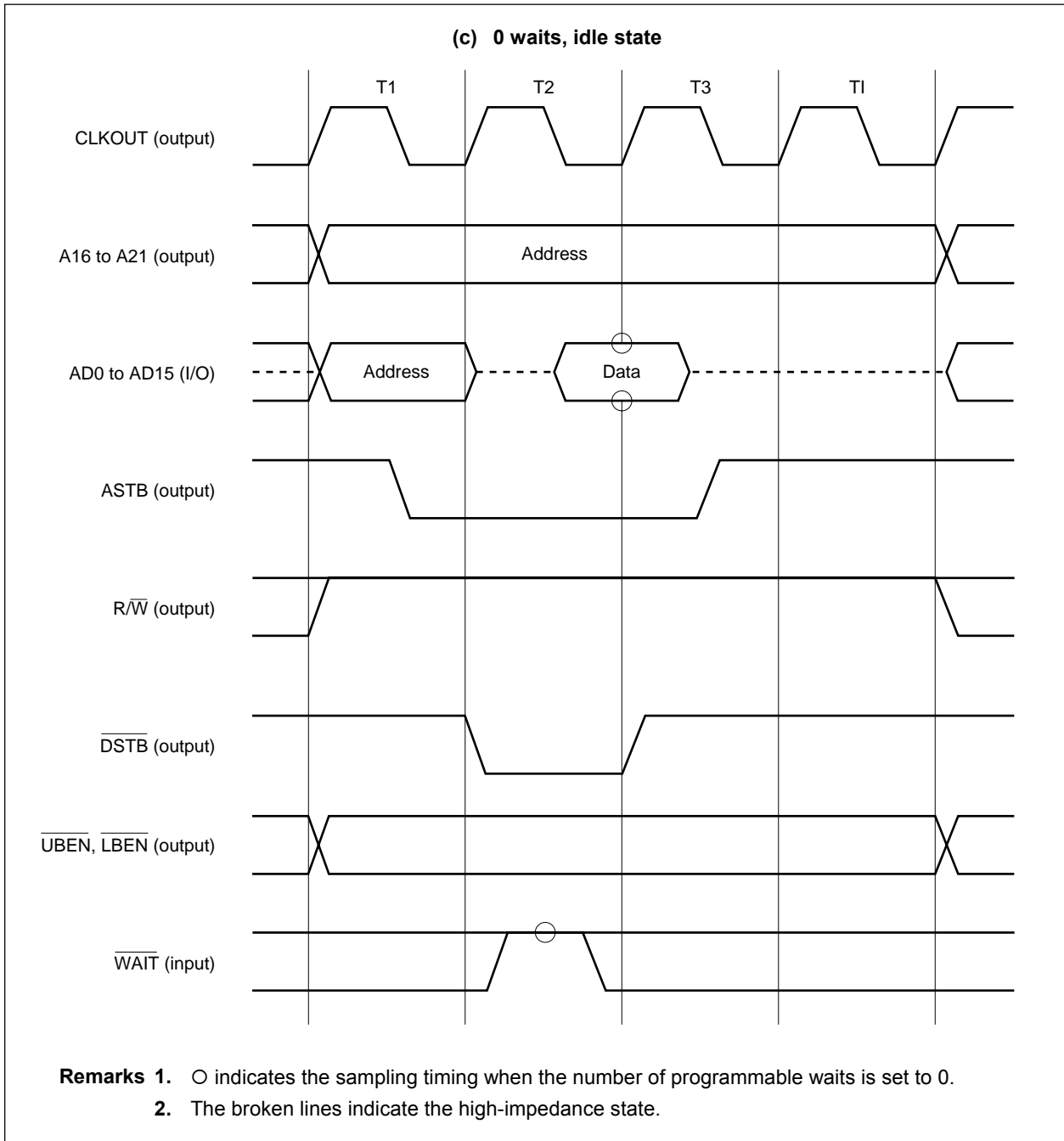


Figure 6-8. Memory Read (4/4)

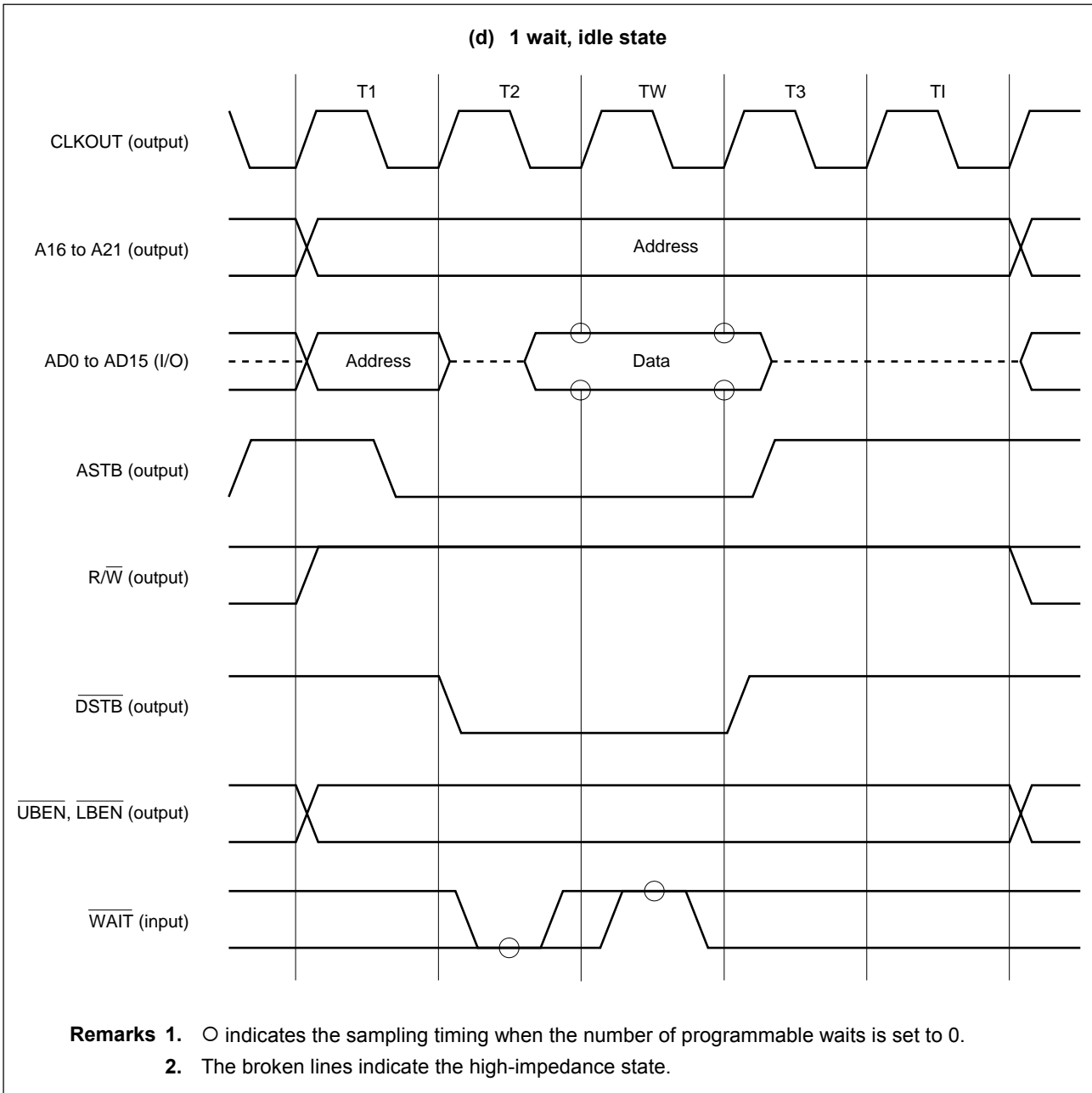


Figure 6-9. Memory Write (1/2)

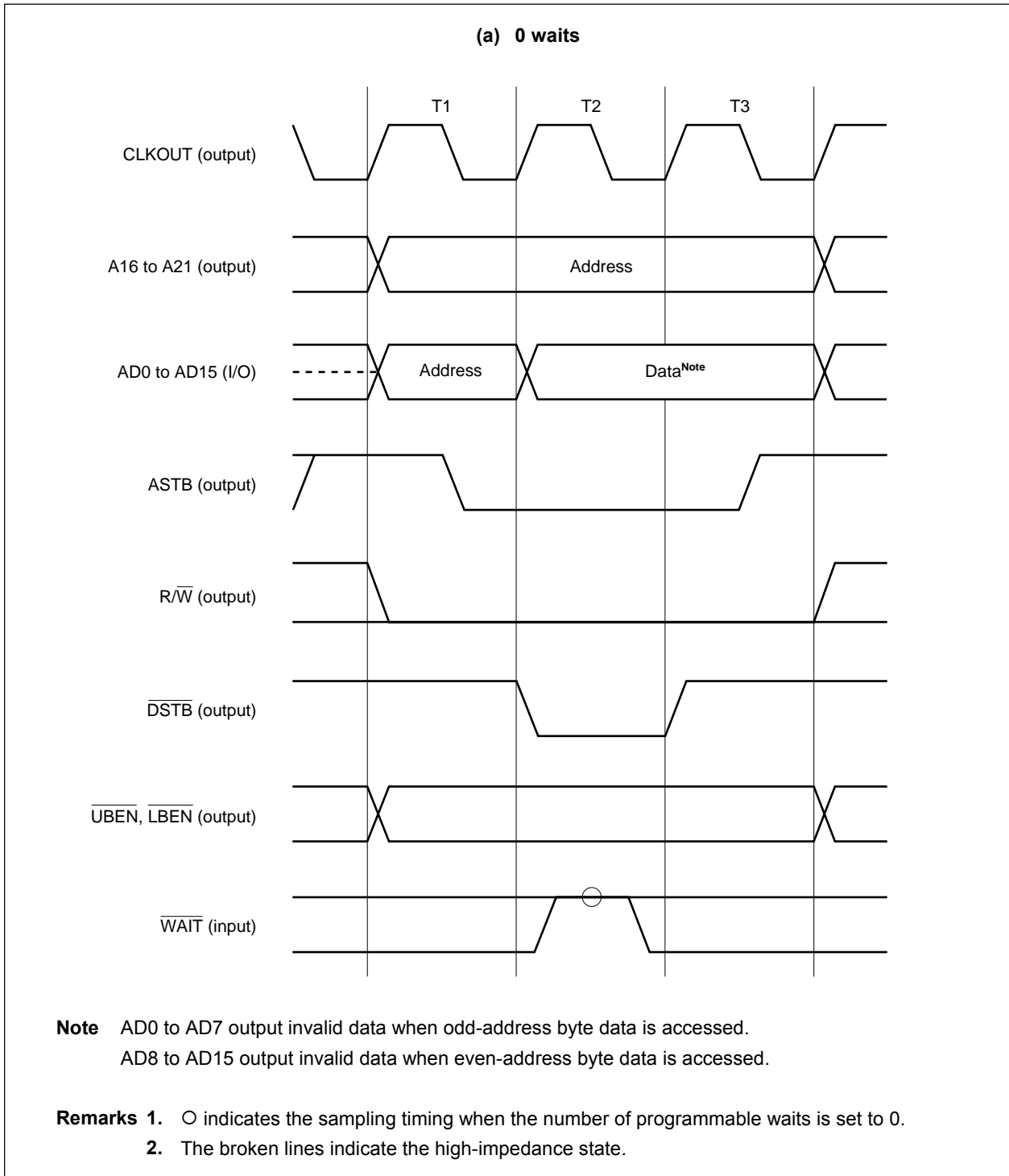


Figure 6-9. Memory Write (2/2)

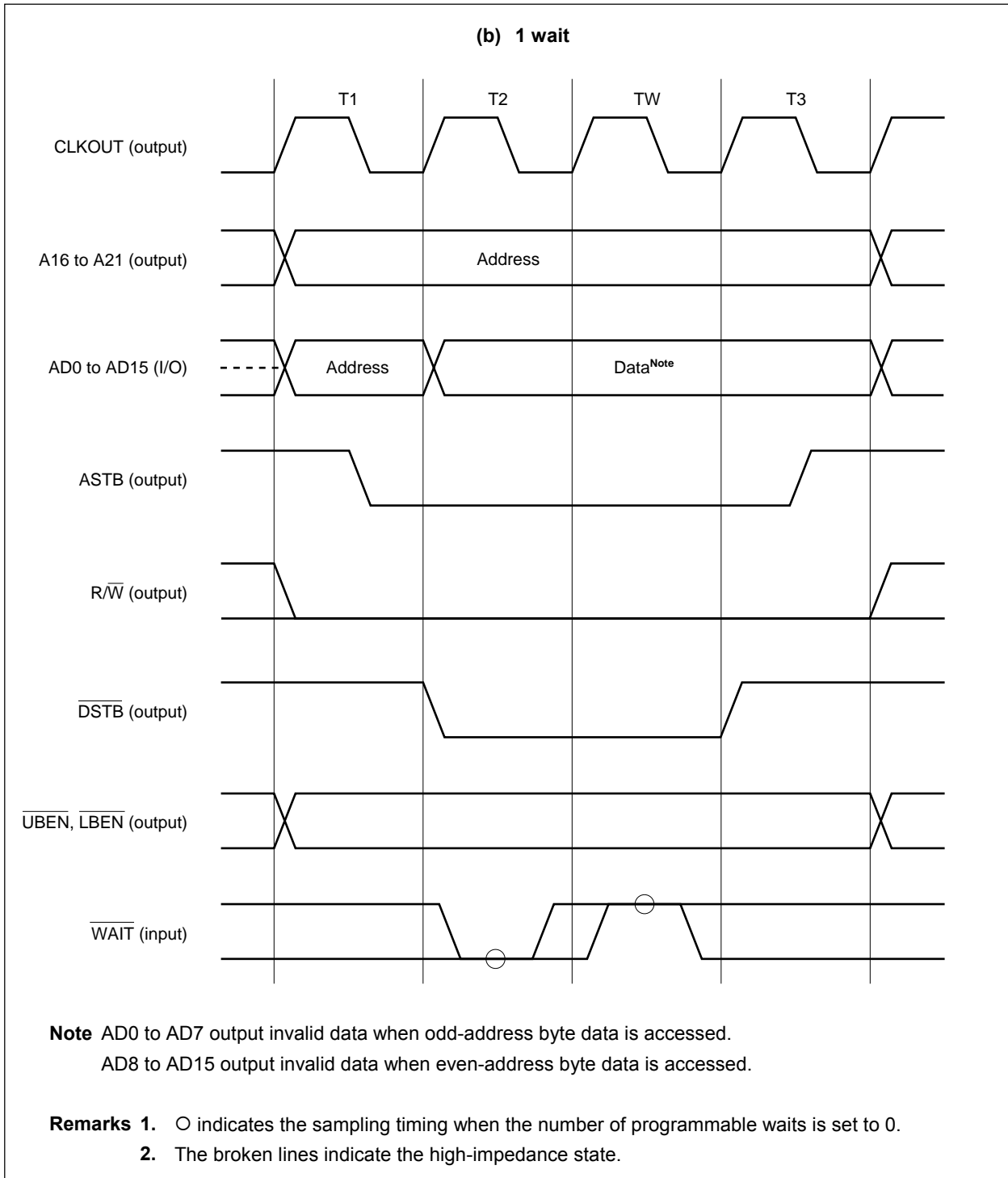
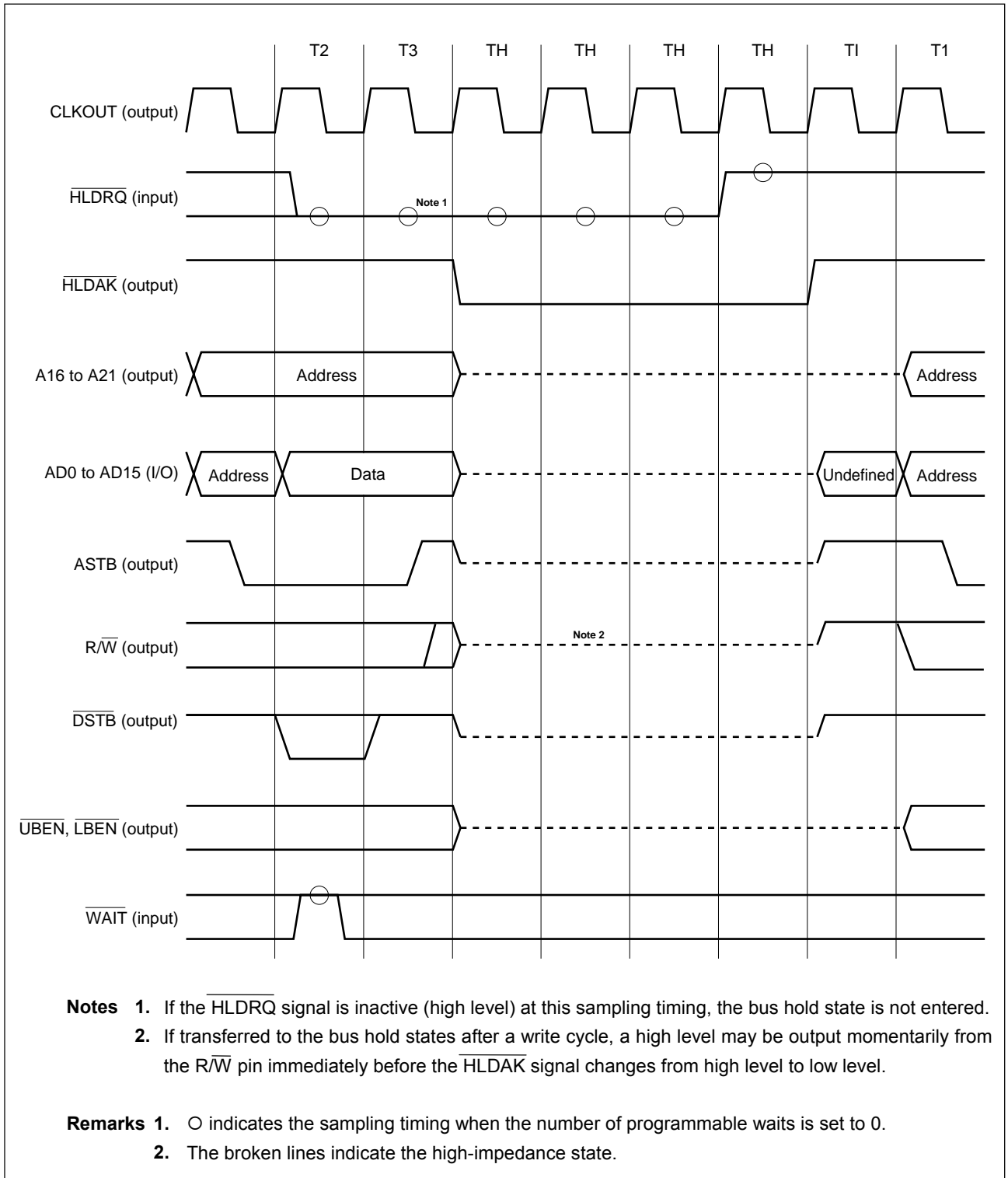


Figure 6-10. Bus Hold Timing



6.9 Bus Priority

There are four external bus cycles: bus hold, operand data access, instruction fetch (branch), and instruction fetch (continuous). The bus hold cycle is given the highest priority, followed by operand data access, instruction fetch (branch), and instruction fetch (continuous) in that order.

The instruction fetch cycle may be inserted in between a read access and a write access in a read-modify-write access.

No instruction fetch cycle and bus hold are inserted between the lower halfword access and the higher halfword access in word access operations.

Table 6-3. Bus Priority

External Bus Cycle	Priority
Bus hold	1
Operand data access	2
Instruction fetch (branch)	3
Instruction fetch (continuous)	4

6.10 Memory Boundary Operation Condition

6.10.1 Program space

- (1) Do not execute a branch to the on-chip peripheral I/O area or a continuous fetch from the internal RAM area to the peripheral I/O area. If a branch or instruction fetch is executed, the NOP instruction code is continuously fetched and fetching from external memory is not performed.
- (2) A prefetch operation extending over the on-chip peripheral I/O area (invalid fetch) does not take place if a branch instruction exists at the upper-limit address of the internal RAM area.

6.10.2 Data space

Only the address aligned at the halfword boundary (when the least significant bit of the address is "0")/word boundary (when the lowest 2 bits of the address are "0") is accessed by halfword (16 bits)/word (32 bits) access, respectively.

Therefore, access that extends over the memory or memory block boundary does not take place.

For details, refer to **V850 Family User's Manual Architecture**.

CHAPTER 7 INTERRUPT/EXCEPTION PROCESSING FUNCTION

7.1 Outline

The V850/SF1 is provided with a dedicated interrupt controller (INTC) for interrupt servicing and realizes a high-powered interrupt function that can service interrupt requests from a total of 41 sources (μ PD703078Y)/44 sources (μ PD70F3079Y, 703079Y).

An interrupt is an event that occurs independently of program execution, and an exception is an event that occurs dependent on program execution.

The V850/SF1 can service interrupt requests from the internal peripheral hardware and external sources. Moreover, exception processing can be started (exception trap) by the TRAP instruction (software exception) or by generation of an exception event (fetching of an illegal op code).

7.1.1 Features

- Interrupts
 - Non-maskable interrupts: 2 sources
 - Maskable interrupts (the number of maskable interrupt sources differs depending on product):
 - μ PD703078Y: 41 sources
 - μ PD703079Y, 70F3079Y: 44 sources
 - 8 levels of programmable priorities
 - Mask specification for interrupt requests according to priority
 - Masks can be specified for each maskable interrupt request.
 - Noise elimination, edge detection, and the valid edge of an external interrupt request signal can be specified.
- Exceptions
 - Software exceptions: 32 sources
 - Exception trap: 1 source (illegal op code exception)

The interrupt/exception sources are listed in Table 7-1.

Table 7-1. Interrupt Source List (1/2)

Type	Classification	Default Priority	Name	Trigger	Interrupt Source	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Reset	Interrupt	–	RESET	Reset input	–	0000H	00000000H	Undefined	–
Non-maskable	Interrupt	–	NMI	NMI pin input	–	0010H	00000010H	nextPC	–
	Interrupt	–	INTWDT	WDTOVF non-maskable	WDT	0020H	00000020H	nextPC	–
Software exception	Exception	–	TRAP0n ^{Note}	TRAP instruction	–	004nH ^{Note}	00000040H	nextPC	–
	Exception	–	TRAP1n ^{Note}	TRAP instruction	–	005nH ^{Note}	00000050H	nextPC	–
Exception trap	Exception trap	–	ILGOP	Illegal op code	–	0060H	00000060H	nextPC	–
Maskable	Interrupt	0	INTWDTM	WDTOVF maskable	WDT	0080H	00000080H	nextPC	WDTIC
		1	INTP0	INTP0 pin	Pin	0090H	00000090H	nextPC	PIC0
		2	INTP1	INTP1 pin	Pin	00A0H	000000A0H	nextPC	PIC1
		3	INTP2	INTP2 pin	Pin	00B0H	000000B0H	nextPC	PIC2
		4	INTP3	INTP3 pin	Pin	00C0H	000000C0H	nextPC	PIC3
		5	INTP4	INTP4 pin	Pin	00D0H	000000D0H	nextPC	PIC4
		6	INTP5	INTP5 pin	Pin	00E0H	000000E0H	nextPC	PIC5
		7	INTP6	INTP6 pin	Pin	00F0H	000000F0H	nextPC	PIC6
		8	INTCSI4	CSI4 transmit end	SIO4	0100H	00000100H	nextPC	CSIC4
		9	INTAD	A/D conversion end	A/D	0110H	00000110H	nextPC	ADIC
		10	INTDMA0	DMA0 transfer end	DMA0	0120H	00000120H	nextPC	DMA0
		11	INTDMA1	DMA1 transfer end	DMA1	0130H	00000130H	nextPC	DMA1
		12	INTDMA2	DMA2 transfer end	DMA2	0140H	00000140H	nextPC	DMA2
		13	INTTM00	TM0 and CR00 match/ TI00 pin valid edge	TM0	0150H	00000150H	nextPC	TMIC00
		14	INTTM01	TM1 and CR01 match/ TI01 pin valid edge	TM0	0160H	00000160H	nextPC	TMIC01
		15	INTTM10	TM1 and CR10 match/ TI00 pin valid edge	TM1	0170H	00000170H	nextPC	TMIC10
		16	INTTM11	TM1 and CR11 match/ TI11 pin valid edge	TM1	0180H	00000180H	nextPC	TMIC11
		17	INTTM2	TM2 compare match/OVF	TM2	0190H	00000190H	nextPC	TMIC2
		18	INTTM3	TM3 compare match/OVF	TM3	01A0H	000001A0H	nextPC	TMIC3
		19	INTTM4	TM4 compare match/OVF	TM4	01B0H	000001B0H	nextPC	TMIC4
		20	INTTM5	TM5 compare match/OVF	TM5	01C0H	000001C0H	nextPC	TMIC3
		21	INTWTN	Watch timer OVF	WT	01D0H	000001D0H	nextPC	WTNIC
		22	INTWTNI	Watch timer prescaler	WTN	01E0H	000001E0H	nextPC	WTNIC
		23	INTIIC0/ INTCSI0	I ² C interrupt/ CSI0 transmit end	I ² C/ CSI0	01F0H	000001F0H	nextPC	CSIC0
		24	INTSER0	UART0 serial error	UART0	0200H	00000200H	nextPC	SERIC0
		25	INTSR0/ INTCSI1	UART0 receive end/ CSI1 transmit end	UART0/ CSI1	0210H	00000210H	nextPC	CSIC1
26	INTST0	UART0 transmit end	UART0	0220H	00000220H	nextPC	STIC0		

Note n: 0 to FH

Table 7-1. Interrupt Source List (2/2)

Type	Classification	Default Priority	Name	Trigger	Interrupt Source	Exception Code	Handler Address	Restored PC	Interrupt Control Register
Maskable	Interrupt	27	INTKR	Key return interrupt	KR	0230H	00000230H	nextPC	KRIC
		28	INTCE1	FCAN1 serial error	FCAN1	0240H	00000240H	nextPC	CANIC1
		29	INTCR1	FCAN1 reception	FCAN1	0250H	00000250H	nextPC	CANIC2
		30	INTCT1	FCAN1 transmission	FCAN1	0260H	00000260H	nextPC	CANIC3
		31	INTCME	FCAN memory access error	FCAN1/2	0270H	00000270H	nextPC	CANIC7
		32	INTTM6	TM6 compare match/OVF	TM6	0280H	00000280H	nextPC	TMIC6
		33	INTTM70	TM7 and CR70 match/TI70 pin valid edge	TM7	0290H	00000290H	nextPC	TMIC70
		34	INTTM71	TM71 and CR71 match	TM7	02A0H	000002A0H	nextPC	TMIC71
		35	INTSER1	UART1 serial error	UART1	02B0H	000002B0H	nextPC	SERIC1
		36	INTSR1/ INTCSI3	UART1 receive end/ CSI3 transmit end	UART1/ CSI3	02C0H	000002C0H	nextPC	CSIC3
		37	INTST1	UART1 transmit end	UART1	02D0H	000002D0H	nextPC	STIC1
		38	INTDMA3	DMA3 transfer end	DMA3	02E0H	000002E0H	nextPC	DMAIC3
		39	INTDMA4	DMA4 transfer end	DMA4	02F0H	000002F0H	nextPC	DMAIC4
		40	INTDMA5	DMA5 transfer end	DMA5	0300H	00000300H	nextPC	DMAIC5
		41	INTCE2 ^{Note}	FCAN2 serial error	FCAN2	0310H	00000310H	nextPC	CANIC4
		42	INTCR2 ^{Note}	FCAN2 reception	FCAN2	0320H	00000320H	nextPC	CANIC5
43	INTCT2 ^{Note}	FCAN2 transmission	FCAN2	0330H	00000330H	nextPC	CANIC6		

Note Available only for the μ PD703079Y and 70F3079Y.

Remarks 1. Default Priority: The priority when two or more maskable interrupt requests occur at the same time. The highest priority is 0.

Restored PC: The value of the PC saved to EIPC or FEPC when interrupt/exception processing is started. However, the value of the PC saved when an interrupt is acknowledged during the DIVH (division) instruction execution is the value of the PC of the current instruction (DIVH).

- The execution address of the illegal instruction when an illegal op code exception occurs is calculated by (Restored PC – 4).
- The restored PC of an interrupt/exception other than RESET is the value of (the PC when an event occurred) + 1.
- Non-maskable interrupts (INTWDT) and maskable interrupts (INTWDTM) are set by the WDTM4 bit of the watchdog timer mode register (WDTM).

7.2 Non-Maskable Interrupt

A non-maskable interrupt is acknowledged unconditionally, even when interrupts are disabled (DI state). An NMI is not subject to priority control and takes precedence over all other interrupts.

The following two non-maskable interrupt requests are available in the V850/SF1.

- NMI pin input (NMI)
- Non-maskable watchdog timer interrupt request (INTWDT)

When the valid edge specified by the rising edge specification register (EGP0) and falling edge specification register (EGN0) is detected at the NMI pin, an interrupt occurs.

INTWDT functions as a non-maskable interrupt (INTWDT) only when the WDTM4 bit of the watchdog timer mode register (WDTM) is set to 1.

While the service routine of a non-maskable interrupt is being executed (PSW.NP = 1), the acknowledgement of another non-maskable interrupt request is held pending. The pending NMI is acknowledged after the original service routine of the non-maskable interrupt under execution has been terminated (by the RETI instruction), or when PSW.NP is cleared to 0 by the LDSR instruction. Note that if two or more NMI requests are input during the execution of the service routine for an NMI, only one NMI will be acknowledged after PSW.NP is cleared to 0.

Caution If PSW.NP is cleared to 0 by the LDSR instruction during non-maskable interrupt servicing, the interrupt afterwards cannot be acknowledged correctly.

7.2.1 Operation

If a non-maskable interrupt is generated, the CPU performs the following processing, and transfers control to the handler routine.

- (1) Saves the restored PC to FEPC.
- (2) Saves the current PSW to FEPSW.
- (3) Writes exception code 0010H to the higher half-word (FECC) of ECR.
- (4) Sets the NP and ID bits of the PSW and clears the EP bit.
- (5) Loads the handler address (00000010H, 00000020H) of the non-maskable interrupt routine to the PC, and transfers control.

Figure 7-1. Non-Maskable Interrupt Servicing

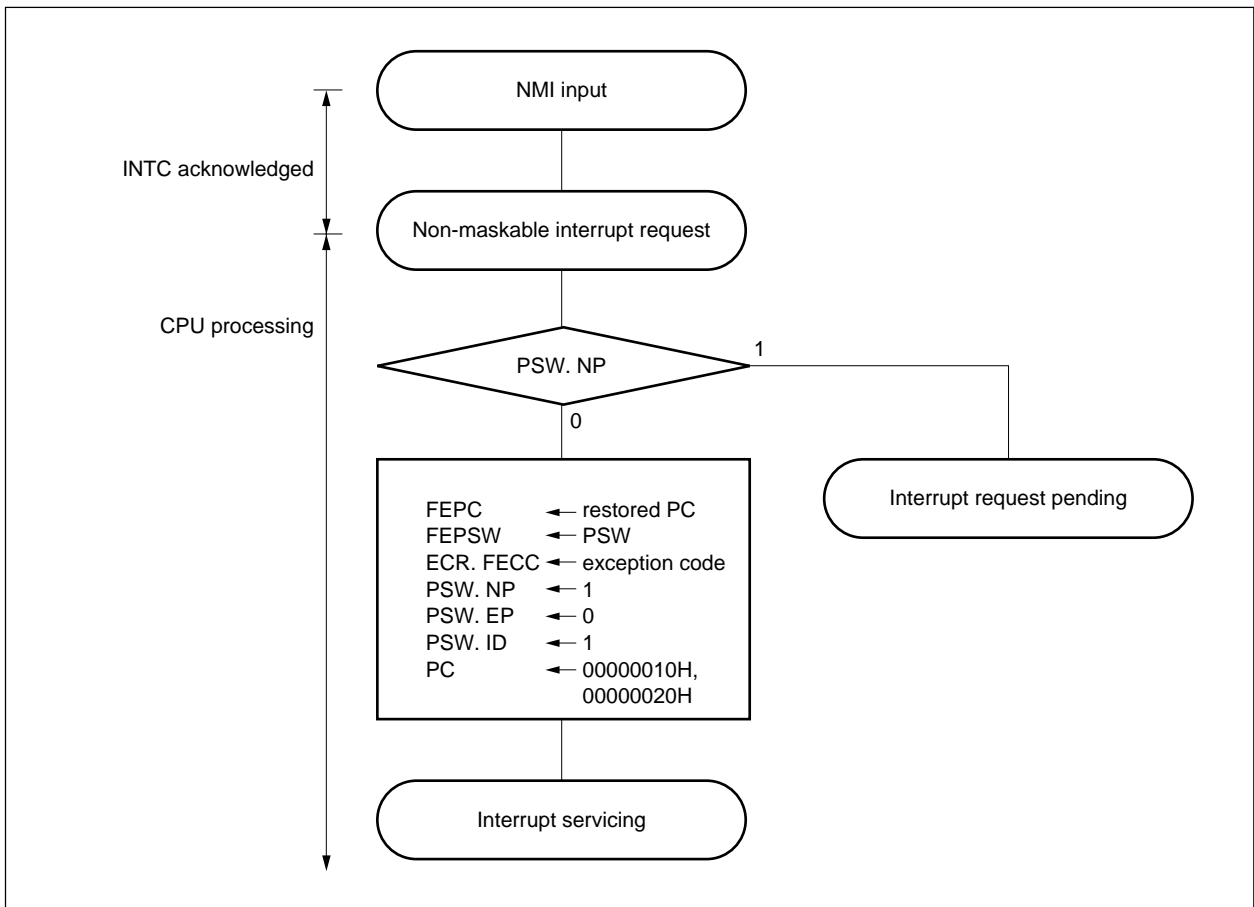
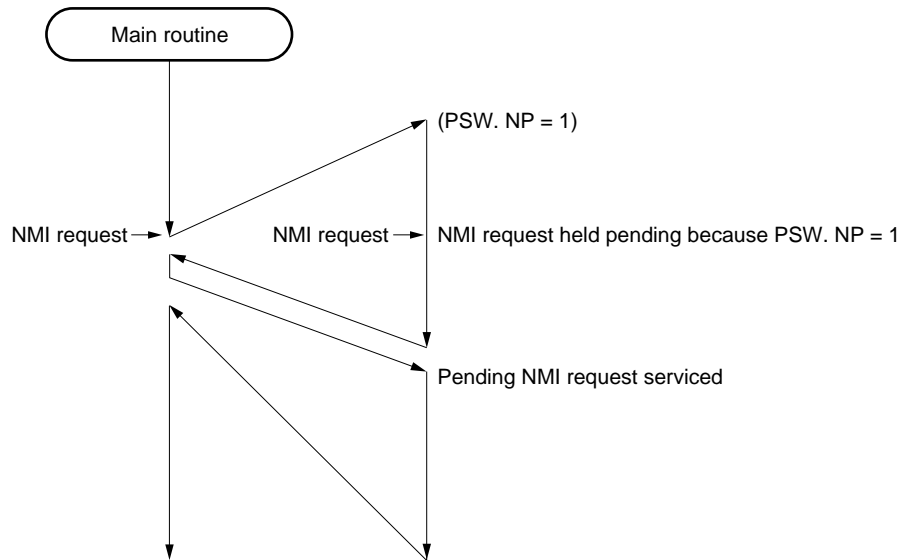
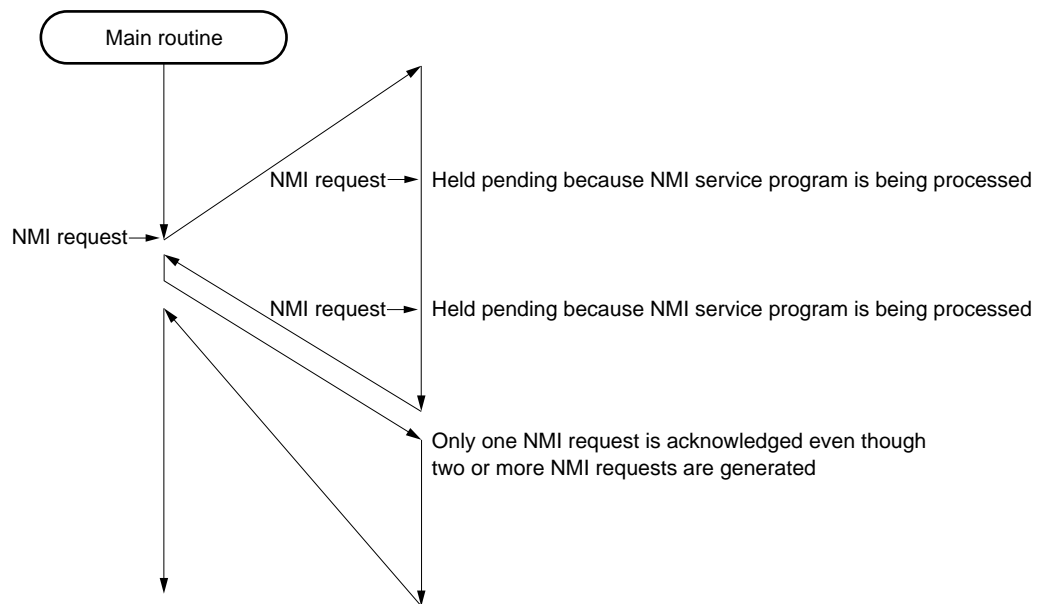


Figure 7-2. Acknowledging Non-Maskable Interrupt Requests

(a) If a new NMI request is generated while an NMI service routine is being executed



(b) If a new NMI request is generated twice while an NMI service routine is being executed



7.2.2 Restore

Execution is restored from non-maskable interrupt servicing by the RETI instruction.

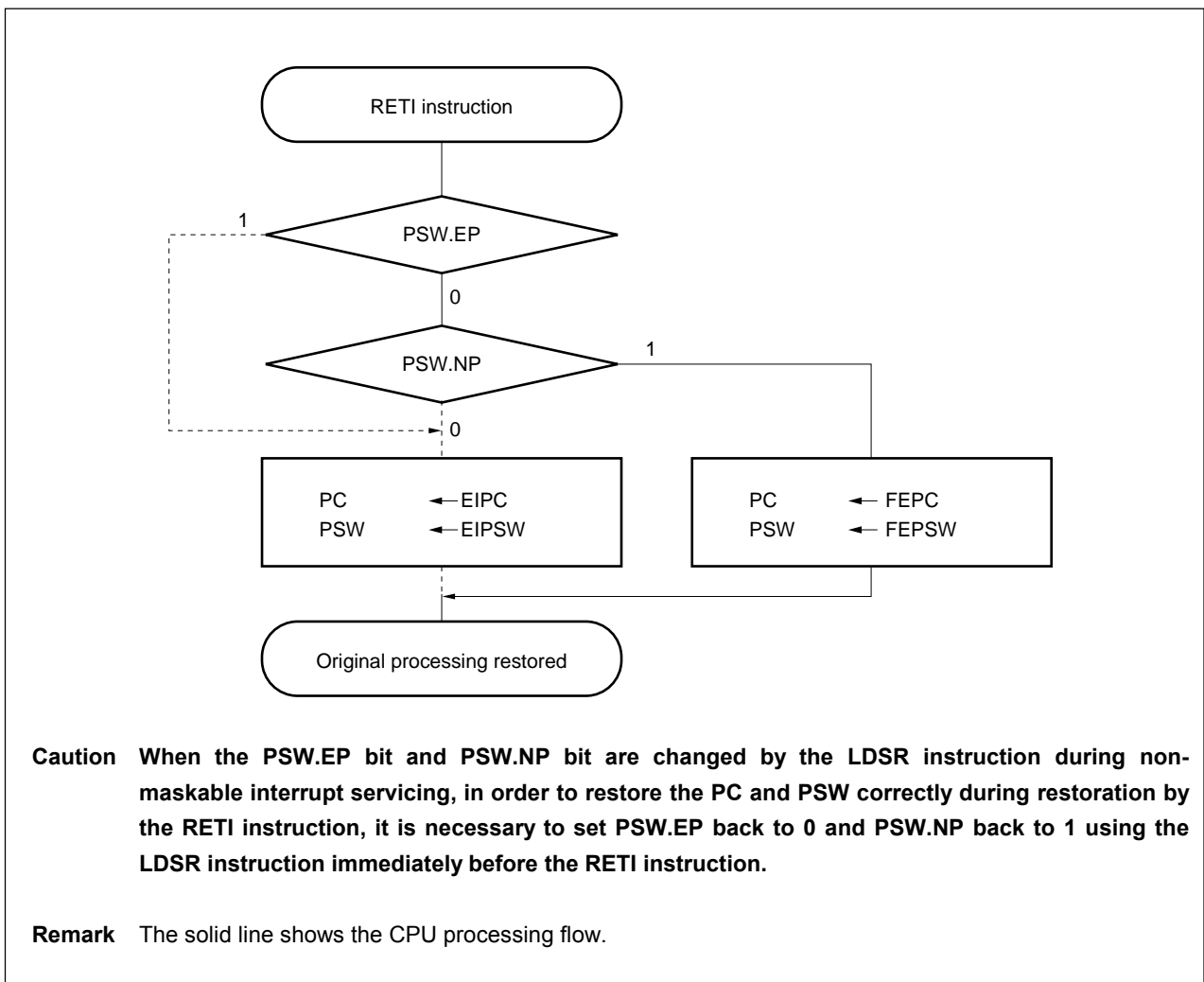
Operation of RETI instruction

When the RETI instruction is executed, the CPU performs the following processing, and transfers control to the address of the restored PC.

- (1) Restores the values of PC and PSW from FEPC and FEPSW, respectively, because the EP bit of the PSW is 0 and the NP bit of the PSW is 1.
- (2) Transfers control back to the address of the restored PC and PSW.

How the RETI instruction is processed is shown below.

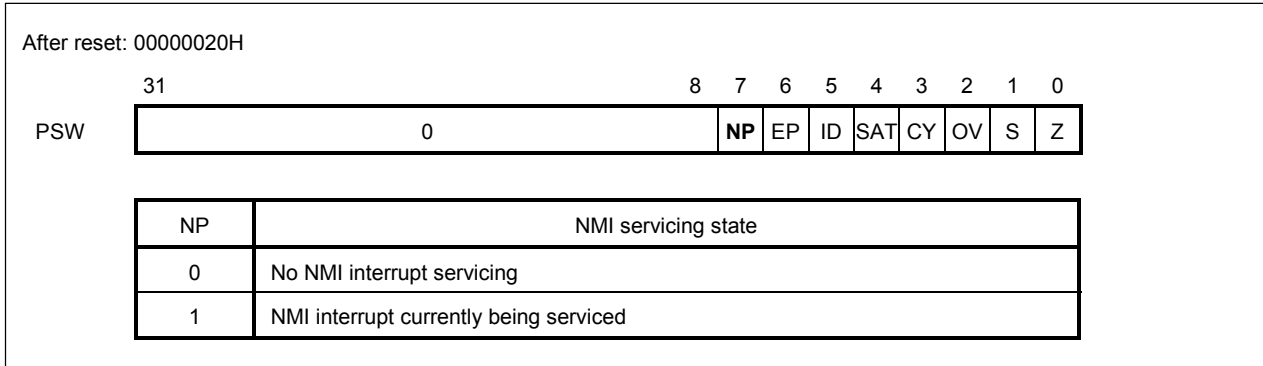
Figure 7-3. RETI Instruction Processing



7.2.3 NP flag

The NP flag is a status flag that indicates that non-maskable interrupt (NMI) servicing is under execution. This flag is set when an NMI interrupt request has been acknowledged, and masks all interrupt requests to prohibit multiple interrupts from being acknowledged.

Figure 7-4. NP Flag (NP)



7.2.4 Noise eliminator of NMI pin

NMI pin noise is eliminated by the noise eliminator using analog delay. Therefore, a signal input to the NMI pin is not detected as an edge, unless it maintains its input level for a certain period. The edge is detected after a certain period has elapsed.

The NMI pin is used for canceling the software stop mode. In the software stop mode, noise elimination using the system clock does not occur because the internal system clock is stopped.

7.2.5 Edge detection function of NMI pin

The NMI pin valid edge can be selected from the following four types: falling edge, rising edge, both edges, neither rising nor falling edge detected.

Rising edge specification register 0 (EGP0) and falling edge specification register 0 (EGN0) specify the valid edge of a non-maskable interrupt (NMI). These two registers can be read/written in 1-bit or 8-bit units.

After reset, the valid edge of the NMI pin is set to the “neither rising nor falling edge detected” state. Therefore, the NMI pin functions as a normal port and an interrupt request cannot be acknowledged, unless a valid edge is specified by using the EGP0 and EGN0 registers.

When using P00 as an output port, set the NMI valid edge to “neither rising nor falling edge detected”.

Rising edge specification register 0 (EGP0)

After reset: 00H R/W Address: FFFFF0C0H

	7	6	5	4	3	2	1	0
EGP0	EGP07	EGP06	EGP05	EGP04	EGP03	EGP02	EGP01	EGP00

EGP0n	Rising edge valid control
0	No interrupt request signal occurs at the rising edge
1	Interrupt request signal occurs at the rising edge

n = 0: NMI pin control
n = 1 to 7: INTP0 to INTP6 pins control

Falling edge specification register 0 (EGN0)

After reset: 00H R/W Address: FFFFF0C2H

	7	6	5	4	3	2	1	0
EGN0	EGN07	EGN06	EGN05	EGN04	EGN03	EGN02	EGN01	EGN00

EGN0n	Falling edge valid control
0	No interrupt request signal occurs at the falling edge
1	Interrupt request signal occurs at the falling edge

n = 0: NMI pin control
n = 1 to 7: INTP0 to INTP6 pins control

7.3 Maskable Interrupts

Maskable interrupt requests can be masked by interrupt control registers. The V850/SF1 has 41 (μ PD703078Y)/44 (μ PD703079Y, 70F3079Y) maskable interrupt sources.

If two or more maskable interrupt requests are generated at the same time, they are acknowledged according to the default priority. In addition to the default priority, eight levels of priorities can be specified by using the interrupt control registers, allowing programmable priority control.

When an interrupt request has been acknowledged, the acknowledgement of other maskable interrupts is disabled and the interrupt disabled (DI) status is set.

When the EI instruction is executed in an interrupt servicing routine, the interrupt enabled (EI) status is set, which enables interrupts having a higher priority to immediately interrupt the service routine in currently progress. Note that only interrupts with a higher priority will have this capability; interrupts with the same priority level cannot be nested.

To use multiple interrupts, it is necessary to save EIPC and EIPSW to memory or a register before executing the EI instruction, and restore EIPC and EIPSW to the original values by executing the DI instruction before the RETI instruction.

When the WDTM4 bit of the watchdog timer mode register (WDTM) is set to 0, the watchdog timer overflow interrupt functions as a maskable interrupt (INTWDTM).

7.3.1 Operation

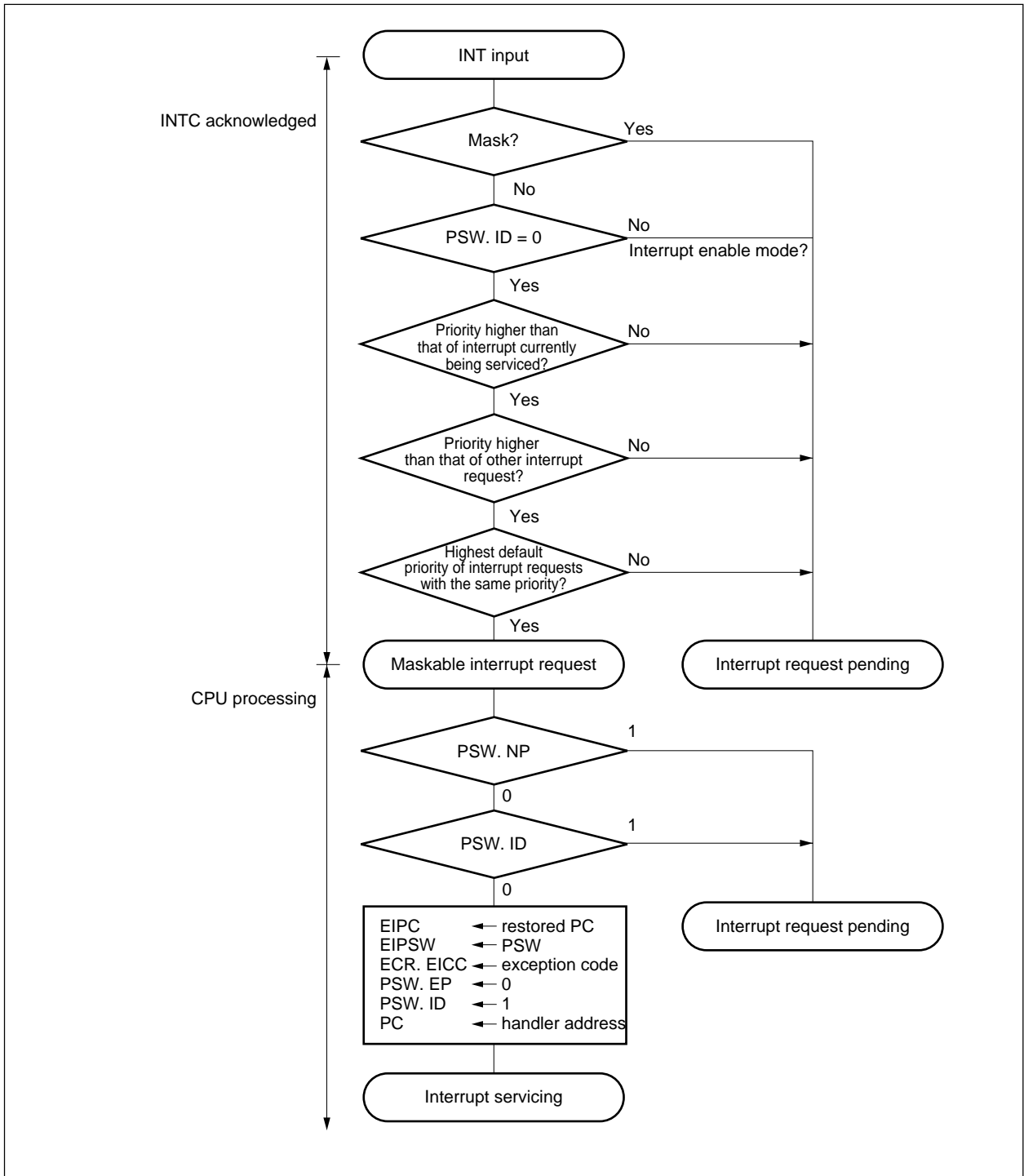
If a maskable interrupt occurs, the CPU performs the following processing, and transfers control to a handler routine.

- (1) Saves the restored PC to EIPC.
- (2) Saves the current PSW to EIPSW.
- (3) Writes an exception code to the lower halfword of ECR (EICC).
- (4) Sets the ID bit of the PSW and clears the EP bit.
- (5) Loads the corresponding handler address to the PC, and transfers control.

The INT input masked by INTC and the INT input that occurs while another interrupt is being serviced (when PSW.NP = 1 or PSW.ID = 1) are held pending internally. When the interrupts are unmasked, or when PSW.NP = 0 and PSW.ID = 0 by using the RETI and LDSR instructions, the pending INT is input to start new maskable interrupt servicing.

How maskable interrupts are serviced is shown below.

Figure 7-5. Maskable Interrupt Servicing



7.3.2 Restore

To restore execution from maskable interrupt servicing, the RETI instruction is used.

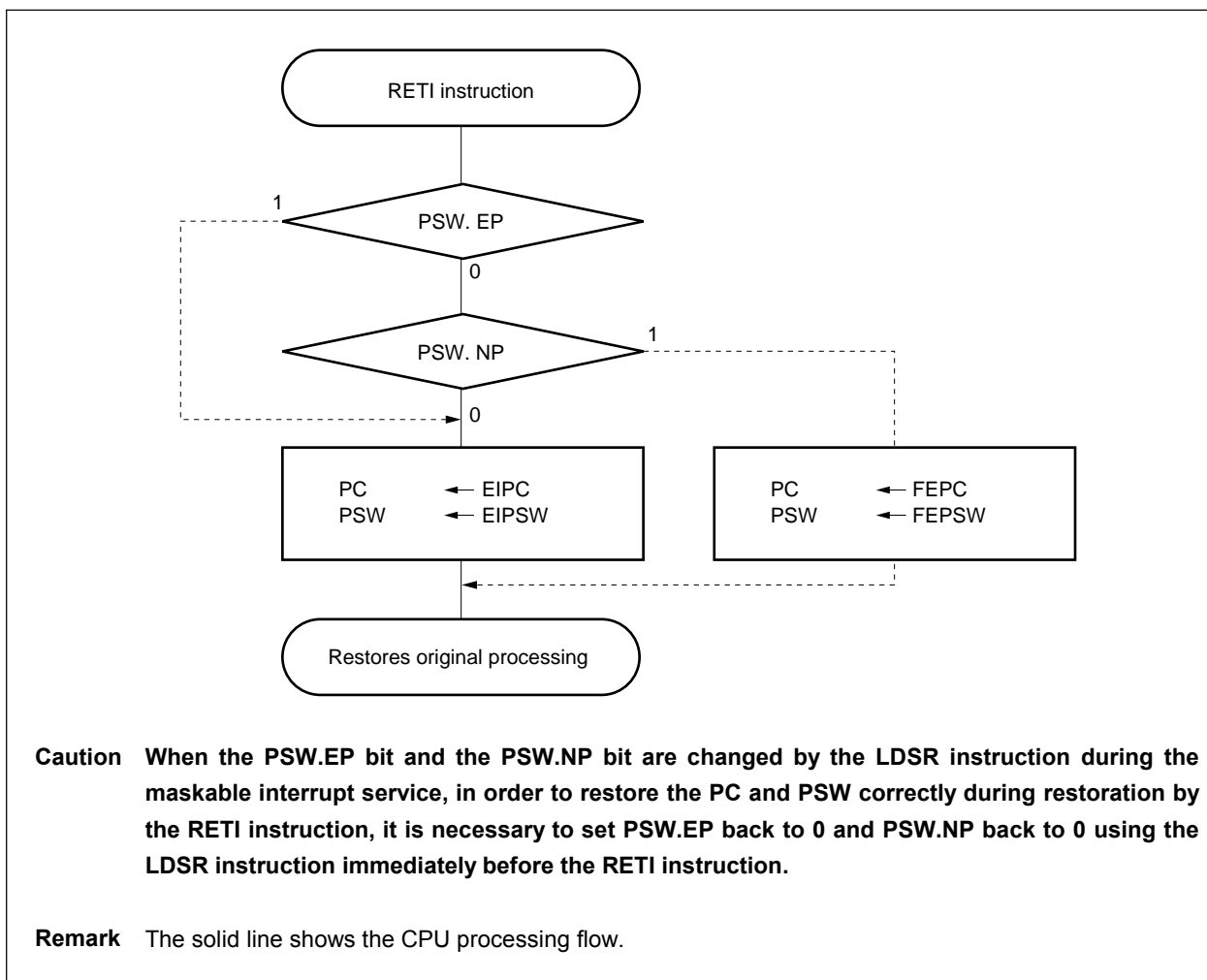
Operation of RETI instruction

When the RETI instruction is executed, the CPU performs the following steps, and transfers control to the address of the restored PC.

- (1) Restores the values of the PC and PSW from EIPC and EIPSW because the EP bit of the PSW is 0 and the NP bit of PSW is 0.
- (2) Transfers control to the address of the restored PC and PSW.

The processing of the RETI instruction is shown below.

Figure 7-6. RETI Instruction Processing



7.3.3 Priorities of maskable interrupts

The V850/SF1 provides multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being serviced. Multiple interrupts can be controlled by priority levels.

There are two types of priority level control: control based on the default priority levels, and control based on the programmable priority levels specified by the interrupt priority level specification bit (xxPRn). When two or more interrupts having the same priority level specified by xxPRn are generated at the same time, interrupts are serviced in order depending on the priority level allocated to each interrupt request type (default priority level) beforehand. For more information, refer to Table 7-1. Programmable priority control divides interrupt requests into eight levels by setting the priority level specification flag.

Note that when an interrupt request is acknowledged, the ID flag of the PSW is automatically set (1). Therefore, when multiple interrupts are to be used, clear (0) the ID flag beforehand (for example, by placing the EI instruction into the interrupt service program) to set the interrupt enable mode.

- Remark** xx: Identification name of each peripheral unit (WDT, P, WTNI, TM, CS, SER, ST, AD, DMA, WTN, CAN, KR)
n: Peripheral unit number (see **Table 7-2**)

Figure 7-7. Example of Interrupt Nesting (1/2)

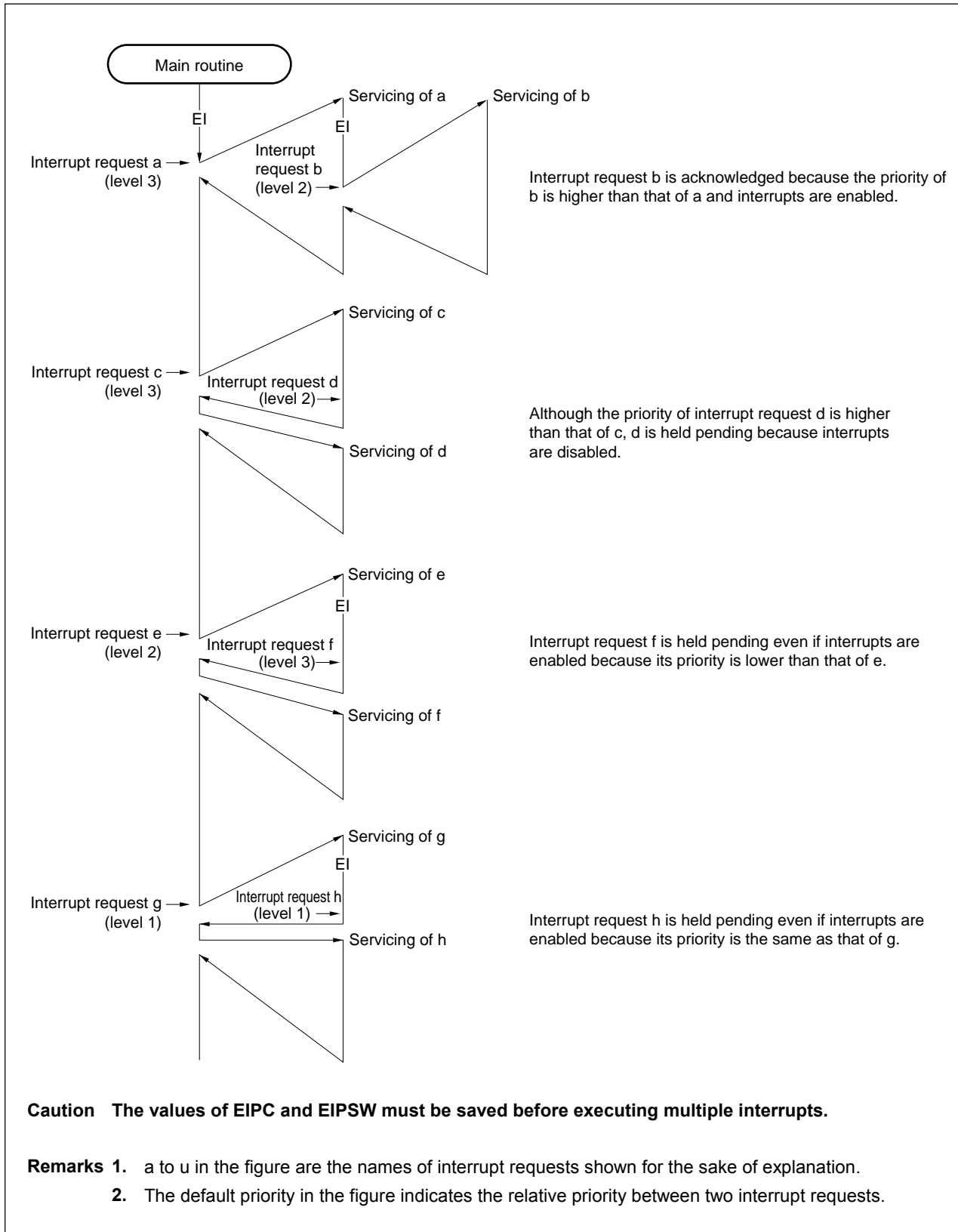


Figure 7-7. Example of Interrupt Nesting (2/2)

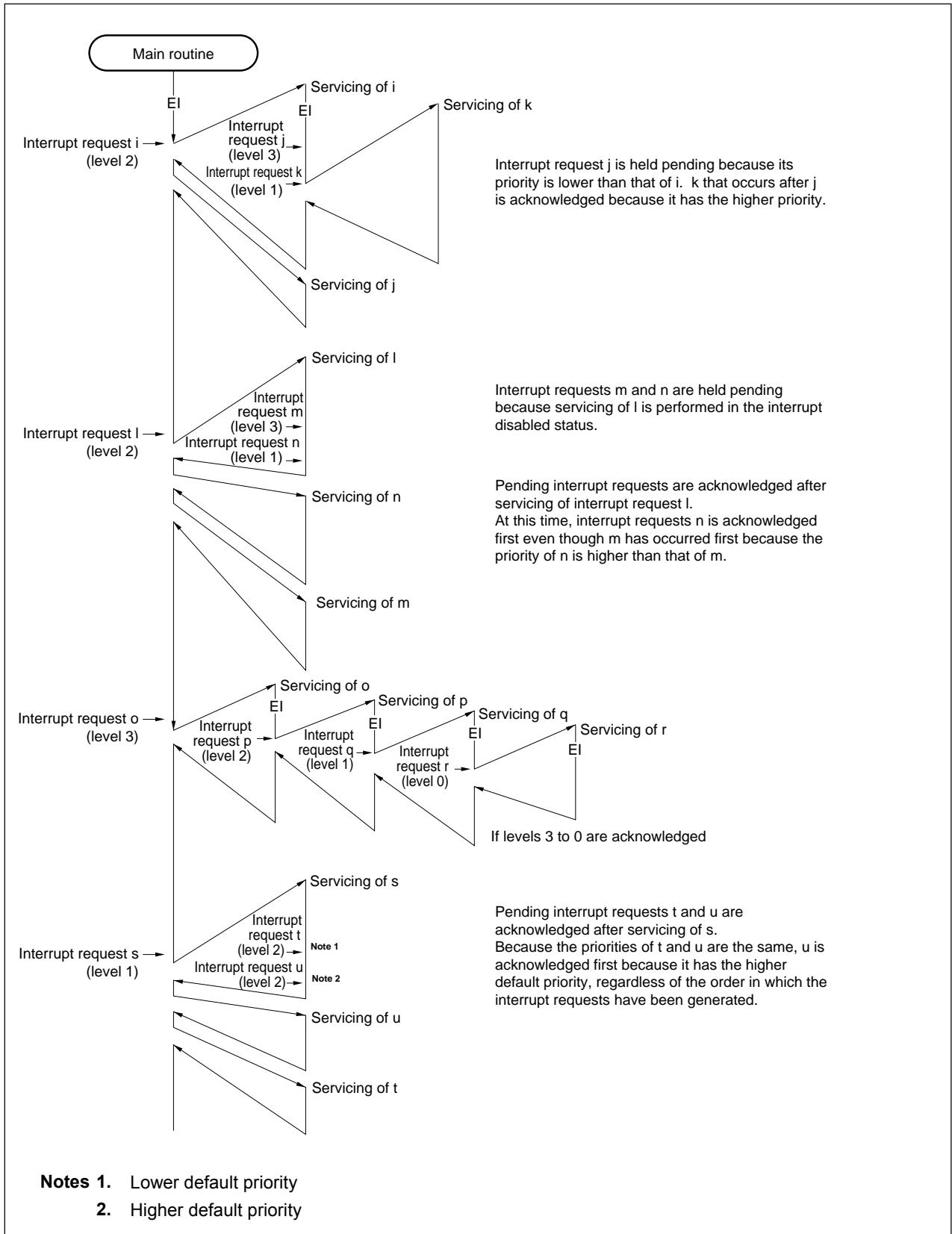
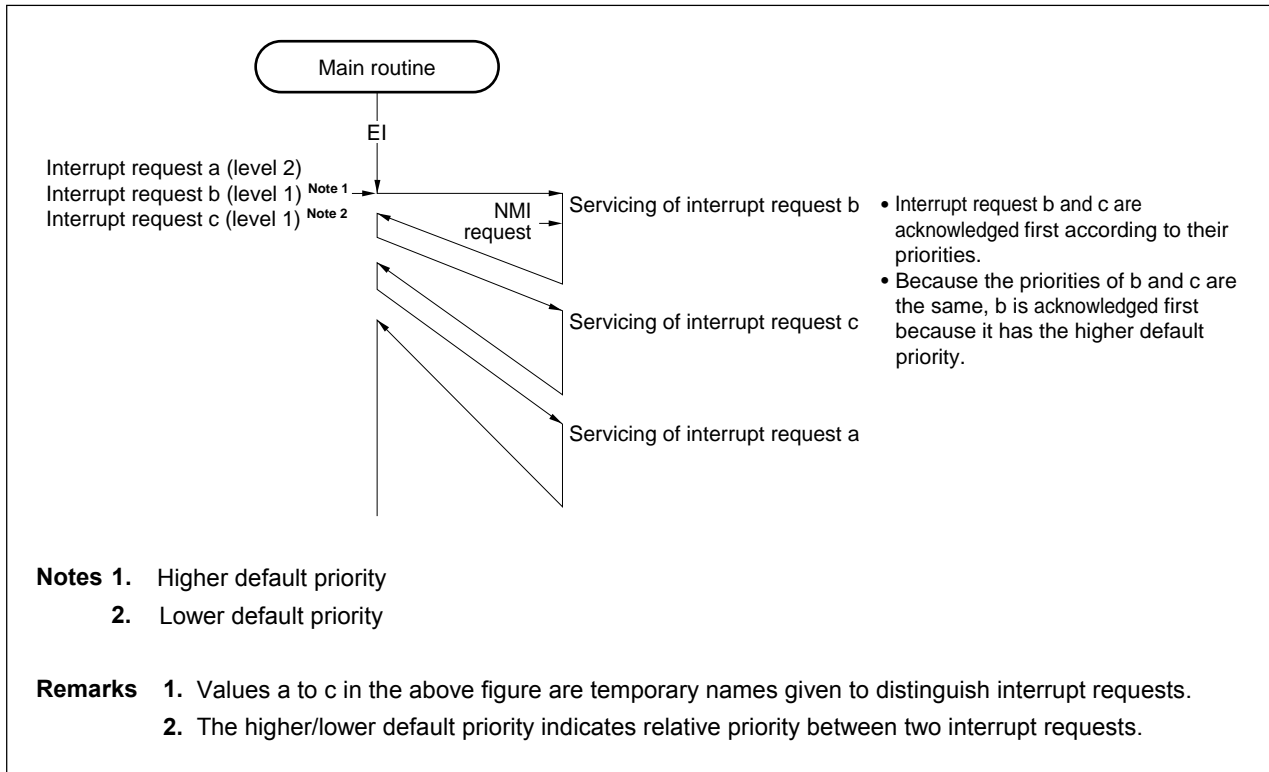


Figure 7-8. Example of Servicing Interrupt Requests Simultaneously Generated



7.3.4 Interrupt control register (xxICn)

An interrupt control register is assigned to each maskable interrupt and sets the control conditions for each maskable interrupt request.

The interrupt control register can be read/written in 8- or 1-bit units.

Caution If the following three conditions conflict, interrupt servicing is executed twice. However, when DMA is not used, interrupt servicing is not executed twice.

- Execution of a bit manipulation instruction corresponding to the interrupt request flag (xxIFn)
- An interrupt of the same interrupt control register (xxICn) as the interrupt request flag (xxIFn) is generated via hardware
- DMA is started during execution of a bit manipulation instruction corresponding to the interrupt request flag (xxIFn)

Two workarounds using software are shown below.

- Insert a DI instruction before the software-based bit manipulation instruction and an EI instruction after it, so that jumping to an interrupt immediately after the bit manipulation instruction execution does not occur.
- When an interrupt request is acknowledged, since the hardware becomes interrupt disabled (DI state), clear the interrupt request flag (xxIFn) before executing the EI instruction in each interrupt servicing routine.

After reset: 47H R/W Address: FFFFF100H to FFFFF156H

	7	6	5	4	3	2	1	0
xxICn	xxIFn	xxMKn	0	0	0	xxPRn2	xxPRn1	xxPRn0

xxIFn	Interrupt request flag ^{Note}
0	Interrupt request not generated
1	Interrupt request generated

xxMKn	Interrupt mask flag
0	Interrupt servicing enabled
1	Interrupt servicing disabled (pending)

xxPRn2	xxPRn1	xxPRn0	Interrupt priority specification bit
0	0	0	Specifies level 0 (highest)
0	0	1	Specifies level 1
0	1	0	Specifies level 2
0	1	1	Specifies level 3
1	0	0	Specifies level 4
1	0	1	Specifies level 5
1	1	0	Specifies level 6
1	1	1	Specifies level 7 (lowest)

Note Automatically reset by hardware when interrupt request is acknowledged.

Remark xx: Identification name of each peripheral unit (WDT, P, WTNI, TM, CS, SER, ST, AD, DMA, WTN, CAN, KR)
n: Peripheral unit number (see Table 7-2)

The addresses and bits of each interrupt control register are as follows.

Table 7-2. Interrupt Control Registers (xxICn)

Address	Register	Bit							
		7	6	5	4	3	2	1	0
FFFFF100H	WDTIC	WDTIF	WDTMK	0	0	0	WDTPR2	WDTPR1	WDTPR0
FFFFF102H	PIC0	PIF0	PMK0	0	0	0	PPR02	PPR01	PPR00
FFFFF104H	PIC1	PIF1	PMK1	0	0	0	PPR12	PPR11	PPR10
FFFFF106H	PIC2	PIF2	PMK2	0	0	0	PPR22	PPR21	PPR20
FFFFF108H	PIC3	PIF3	PMK3	0	0	0	PPR32	PPR31	PPR30
FFFFF10AH	PIC4	PIF4	PMK4	0	0	0	PPR42	PPR41	PPR40
FFFFF10CH	PIC5	PIF5	PMK5	0	0	0	PPR52	PPR51	PPR50
FFFFF10EH	PIC6	PIF6	PMK6	0	0	0	PPR62	PPR61	PPR60
FFFFF110H	CSIC4	CSIF4	CSMK4	0	0	0	CSPR42	CSPR41	CSPR40
FFFFF112H	ADIC	ADIF	ADMK	0	0	0	ADPR2	ADPR1	ADPR0
FFFFF114H	DMAIC0	DMAIF0	DMAMK0	0	0	0	DMAPR02	DMAPR01	DMAPR00
FFFFF116H	DMAIC1	DMAIF1	DMAMK1	0	0	0	DMAPR12	DMAPR11	DMAPR10
FFFFF118H	DMAIC2	DMAIF2	DMAMK2	0	0	0	DMAPR22	DMAPR21	DMAPR20
FFFFF11AH	TMIC00	TMIF00	TMMK00	0	0	0	TMPR002	TMPR001	TMPR000
FFFFF11CH	TMIC01	TMIF01	TMMK01	0	0	0	TMPR012	TMPR011	TMPR010
FFFFF11EH	TMIC10	TMIF10	TMMK10	0	0	0	TMPR102	TMPR101	TMPR100
FFFFF120H	TMIC11	TMIF11	TMMK11	0	0	0	TMPR112	TMPR111	TMPR110
FFFFF122H	TMIC2	TMIF2	TMMK2	0	0	0	TMPR22	TMPR21	TMPR20
FFFFF124H	TMIC3	TMIF3	TMMK3	0	0	0	TMPR32	TMPR31	TMPR30
FFFFF126H	TMIC4	TMIF4	TMMK4	0	0	0	TMPR42	TMPR41	TMPR40
FFFFF128H	TMIC5	TMIF5	TMMK5	0	0	0	TMPR52	TMPR51	TMPR50
FFFFF12AH	WTNIC	WTNIF	WTNMK	0	0	0	WTNPR2	WTNPR1	WTNPR0
FFFFF12CH	WTNIIC	WTNIIF	WTNIMK	0	0	0	WTNIPR2	WTNIPR1	WTNIPR0
FFFFF12EH	CSIC0	CSIF0	CSMK0	0	0	0	CSPR02	CSPR01	CSPR00
FFFFF130H	SERIC0	SERIF0	SERMK0	0	0	0	SERPR02	SERPR01	SERPR00
FFFFF132H	CSIC1	CSIF1	CSMK1	0	0	0	CSPR12	CSPR11	CSPR10
FFFFF134H	STIC0	STIF0	STMK0	0	0	0	STPR02	STPR01	STPR00
FFFFF136H	KRIC	KRIF	KRMK	0	0	0	KRPR2	KRPR1	KRPR0
FFFFF138H	CANIC1	CANIF1	CANMK1	0	0	0	CANPR12	CANPR11	CANPR10
FFFFF13AH	CANIC2	CANIF2	CANMK2	0	0	0	CANPR22	CANPR21	CANPR20
FFFFF13CH	CANIC3	CANIF3	CANMK3	0	0	0	CANPR32	CANPR31	CANPR30
FFFFF13EH	CANIC7	CANIF7	CANMK7	0	0	0	CANPR72	CANPR71	CANPR70
FFFFF140H	TMIC6	TMIF6	TMMK6	0	0	0	TMPR62	TMPR61	TMPR60
FFFFF142H	TMIC70	TMIF70	TMMK70	0	0	0	TMPR702	TMPR701	TMPR700
FFFFF144H	TMIC71	TMIF71	TMMK71	0	0	0	TMPR712	TMPR711	TMPR710
FFFFF146H	SERIC1	SERIF1	SERMK1	0	0	0	SERPR12	SERPR11	SERPR10
FFFFF148H	CSIC3	CSIF3	CSMK3	0	0	0	CSPR32	CSPR31	CSPR30
FFFFF14AH	STIC1	STIF1	STMK1	0	0	0	STPR12	STPR11	STPR10
FFFFF14CH	DMAIC3	DMAIF3	DMAMK3	0	0	0	DMAPR32	DMAPR31	DMAPR30
FFFFF14EH	DMAIC4	DMAIF4	DMAMK4	0	0	0	DMAPR42	DMAPR41	DMAPR40
FFFFF150H	DMAIC5	DMAIF5	DMAMK5	0	0	0	DMAPR52	DMAPR51	DMAPR50
FFFFF152H	CANIC4 ^{Note}	CANIF4	CANMK4	0	0	0	CANPR42	CANPR41	CANPR40
FFFFF154H	CANIC5 ^{Note}	CANIF5	CANMK5	0	0	0	CANPR52	CANPR51	CANPR50
FFFFF156H	CANIC6 ^{Note}	CANIF6	CANMK6	0	0	0	CANPR62	CANPR61	CANPR60

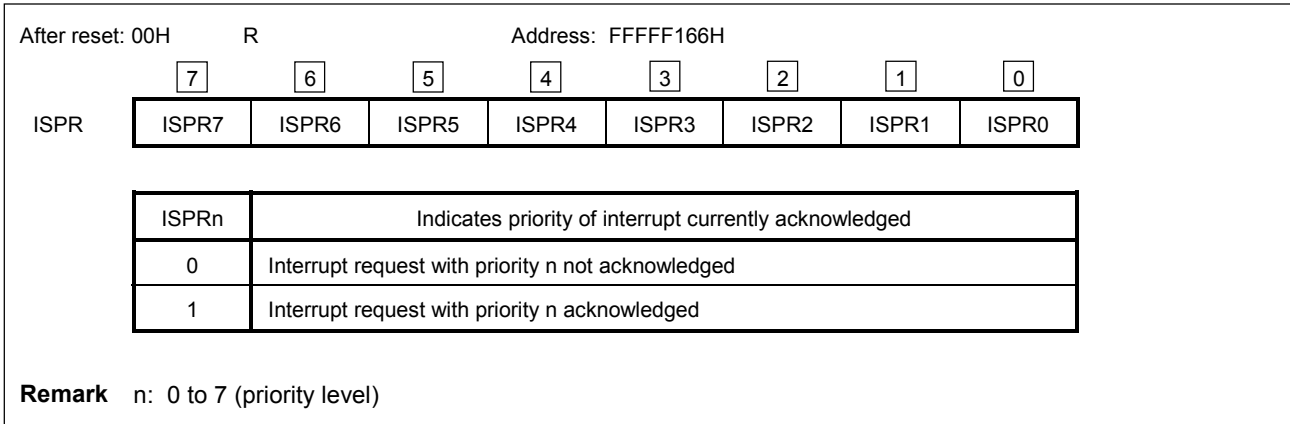
Note Available only for the μ PD703079Y and 70F3079Y.

7.3.5 In-service priority register (ISPR)

This register holds the priority level of the maskable interrupt currently acknowledged. When an interrupt request is acknowledged, the bit of this register corresponding to the priority level of that interrupt is set (1) and remains set while the interrupt is being serviced.

When the RETI instruction is executed, the bit corresponding to the interrupt request having the highest priority is automatically reset (0) by hardware. However, it is not reset (0) when execution is returned from non-maskable interrupt servicing or exception processing.

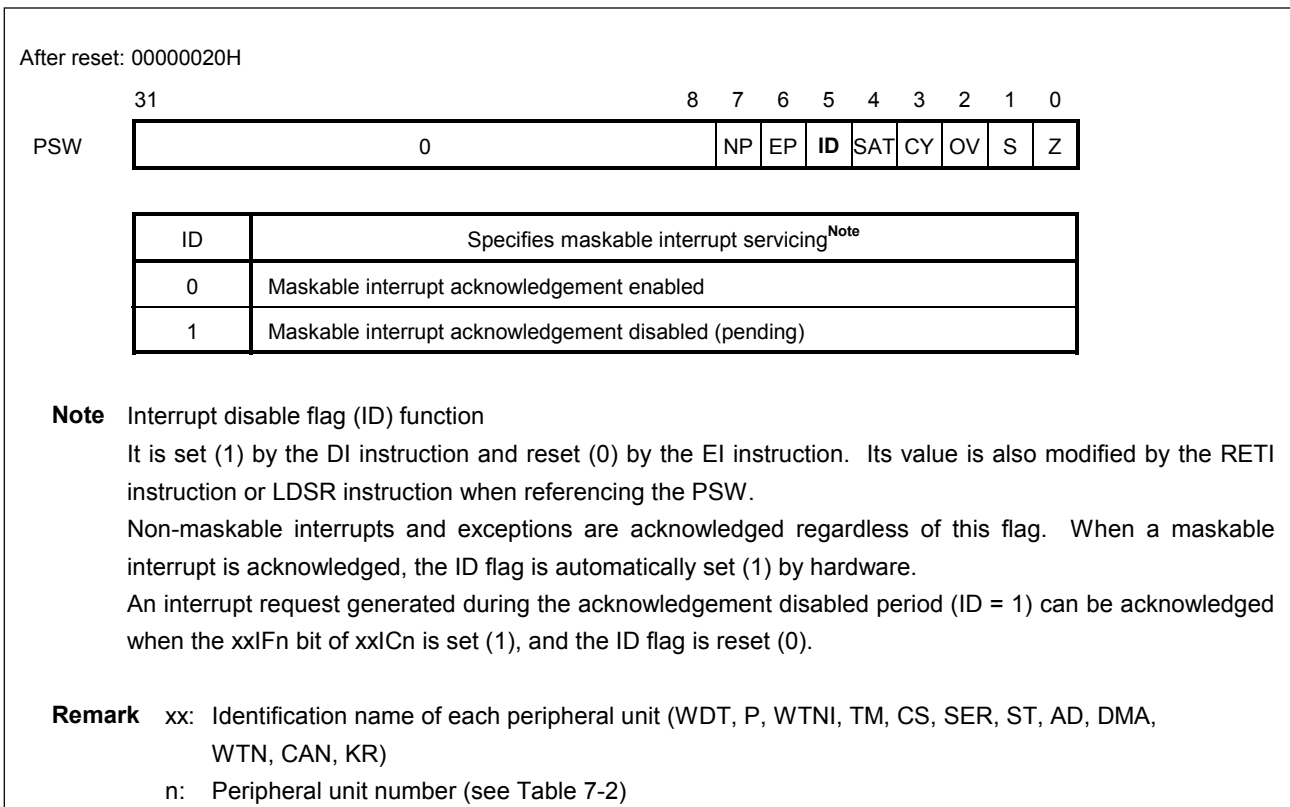
This register is read-only, in 8- or 1-bit units.



7.3.6 Interrupt disable flag (ID)

The interrupt disable flag (ID) controls the operating status of maskable interrupt requests and stores the enable/disable control information of interrupt requests. It is allocated in the PSW.

Figure 7-9. Interrupt Disable Flag (ID)



7.3.7 Watchdog timer mode register (WDTM)

This register can be read/written in 8- or 1-bit units (for details, refer to **CHAPTER 10 WATCHDOG TIMER FUNCTION**).

After reset: 00H R/W Address: FFFFF384H

	7	6	5	4	3	2	1	0
WDTM	RUN	0	0	WDTM4	0	0	0	0

RUN	Watchdog timer operation control
0	Count operation stopped
1	Count started after clearing

WDTM4	Timer mode selection/interrupt control by WDT
0	Interval timer mode
1	WDT mode

Caution If the RUN or WDTM4 bit is set to 1, that bit can only be cleared by reset input.

7.3.8 Noise elimination

(1) Noise elimination of INTP0 to INTP3 pins

The INTP0 to INTP3 pins incorporate a noise eliminator that functions via analog delay. Therefore, a signal input to each pin is not detected as an edge, unless it maintains its input level for a certain period.

An edge is detected after a certain period has elapsed.

(2) Noise elimination of INTP4 and INTP5 pins

The INTP4 and INTP5 pins incorporate a digital noise eliminator. If an input level of the INTP pin is detected by the sampling clock (f_{xx}) and the same level is not detected three successive times, the input pulse is eliminated as a noise. Note the following:

- If the input pulse width is 2 or 3 clocks, whether it is detected as a valid edge or eliminated as a noise is undetermined. To securely detect the valid edge, the same level input of 3 clocks or more is required.
- When noise is generated in synchronization with the sampling clock, this may not be recognized as noise. In this case, eliminate the noise by adding a filter to the input pin.

(3) Noise elimination of INTP6 pin

The INTP6 pin incorporates a digital noise eliminator. The sampling clock for digital sampling can be selected from among f_{xx} , $f_{xx}/64$, $f_{xx}/128$, $f_{xx}/256$, $f_{xx}/512$, $f_{xx}/1024$, and f_{XT} . Sampling is performed 3 times.

The noise elimination control register (NCC) selects the clock to be used. Remote control signals can be received effectively with this function.

f_{XT} can be used for the noise elimination clock. In this case, the INTP6 external interrupt function is enabled in the IDLE/STOP mode.

This register can be read/written in 8- or 1-bit units.

Caution After the sampling clock has been changed, it takes 3 sampling clocks to initialize the noise eliminator. For that reason, if an INTP6 valid edge was input within these 3 clocks, an interrupt request may occur. Therefore, observe the following points when using the interrupt and DMA functions.

- When using the interrupt function, after 3 sampling clocks have elapsed, enable interrupts after the interrupt request flag (bit 7 of PIC6) has been cleared.
- When using the DMA function, after 3 sampling clocks have elapsed, enable DMA by setting bit 0 of DCHCn.

After reset: 00H R/W Address: FFFF3D4H

	7	6	5	4	3	2	1	0
NCC	0	0	0	0	0	NCS2	NCS1	NCS0

NCS2	NCS1	NCS0	Sampling clock	Reliably eliminated noise width ^{Note}	
				$f_{xx} = 16 \text{ MHz}$	$f_{xx} = 8 \text{ MHz}$
0	0	0	f_{xx}	125.0 ns	250.0 ns
0	0	1	$f_{xx}/64$	8.0 μs	16.0 μs
0	1	0	$f_{xx}/128$	16.0 μs	32.0 μs
0	1	1	$f_{xx}/256$	32.0 μs	64.0 μs
1	0	0	$f_{xx}/512$	64.0 μs	128.0 μs
1	0	1	$f_{xx}/1024$	128.0 μs	256.0 μs
1	1	0	Setting prohibited		
1	1	1	f_{XT}	61 μs	

Note Since sampling is performed three times, the reliably eliminated noise width is $2 \times$ noise elimination clock.

7.3.9 Edge detection function

The valid edges of the INTP0 to INTP6 pins can be selected for each pin from the following four types.

- Rising edge
- Falling edge
- Both rising and falling edges
- Neither rising nor falling edge detected

The validity of the rising edge is controlled by rising edge specification register 0 (EGP0), and the validity of the falling edge is controlled by falling edge specification register 0 (EGN0). Refer to **7.2.5 Edge detection function of NMI pin** for details of EGP0 and EGN0.

After reset, the valid edge of the NMI pin is set to the “neither rising nor falling edge detected” state. Therefore, the NMI pin functions as a normal port and an interrupt request cannot be acknowledged, unless a valid edge is specified by using the EGP0 and EGN0 registers.

When using P01 to P07 as output ports, set the valid edges of INTP0 to INTP6 to “neither rising nor falling edge detected” or mask the interrupt request.

7.4 Software Exception

A software exception is generated when the CPU executes the TRAP instruction, and can always be acknowledged.

- TRAP instruction format: TRAP vector (where vector is 0 to 1FH)

For details of the instruction function, refer to the **V850 Family User's Manual Architecture**.

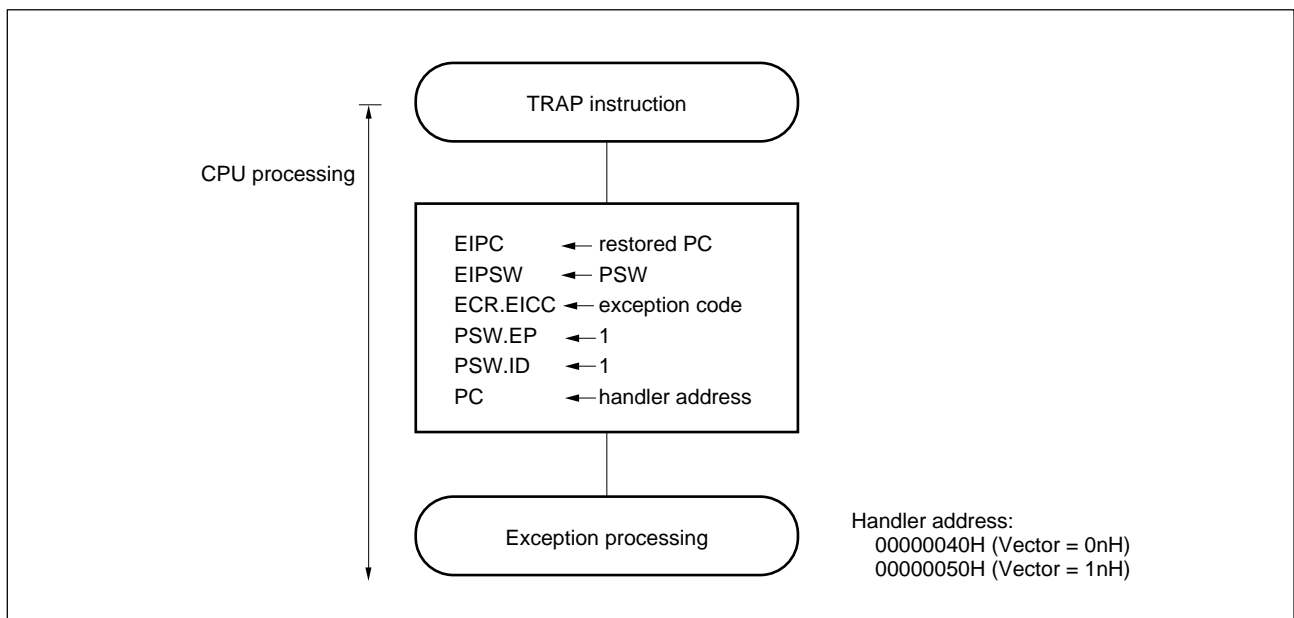
7.4.1 Operation

If a software exception occurs, the CPU performs the following processing, and transfers control to the handler routine.

- (1) Saves the restored PC to EIPC.
- (2) Saves the current PSW to EIPSW.
- (3) Writes an exception code to the lower 16 bits (EICC) of ECR (interrupt source).
- (4) Sets the EP and ID bits of the PSW.
- (5) Loads the handler address (00000040H or 00000050H) of the software exception routine in the PC, and transfers control.

How a software exception is processed is shown below.

Figure 7-10. Software Exception Processing



7.4.2 Restore

To restore or return execution from the software exception service routine, the RETI instruction is used.

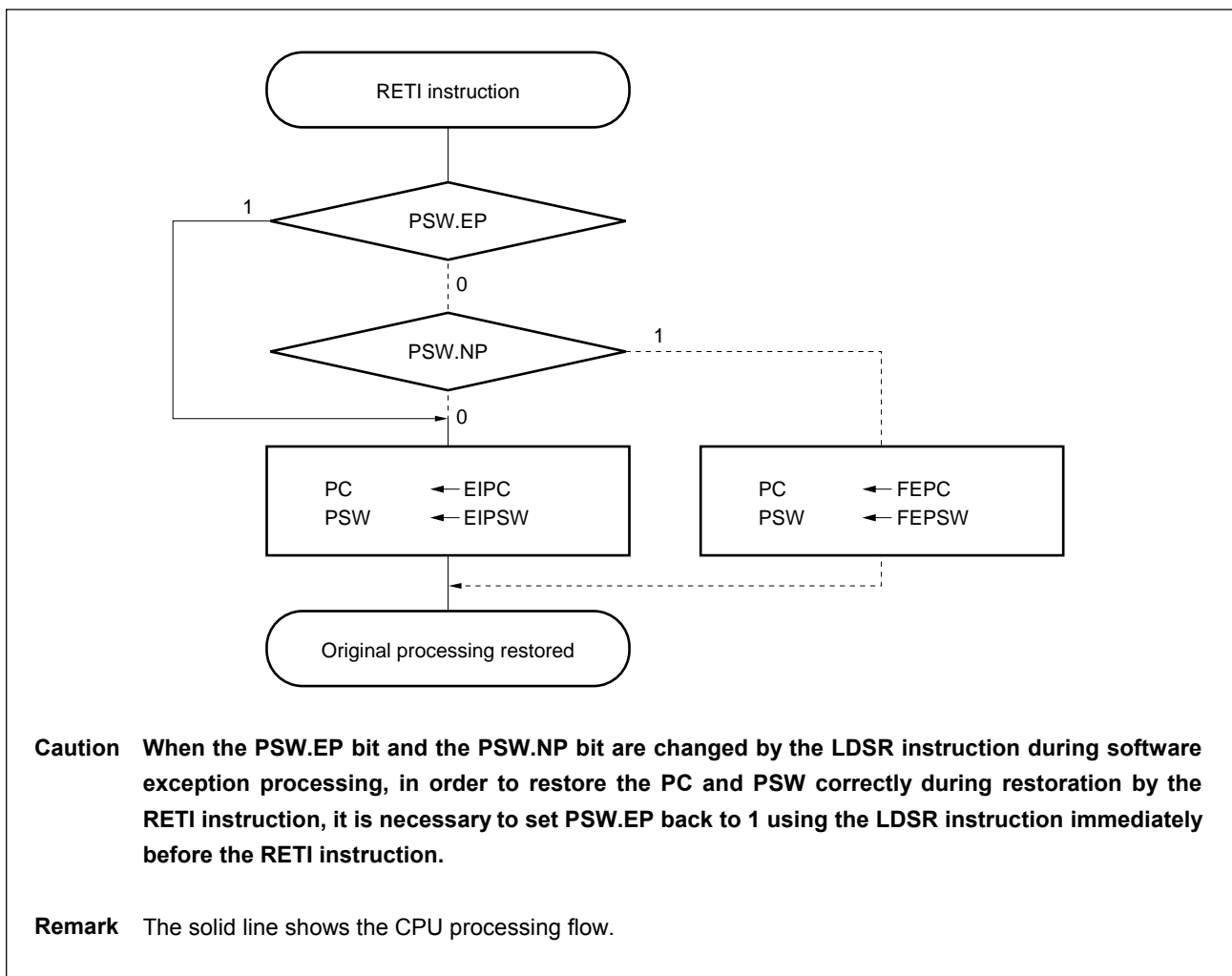
Operation of RETI instruction

When the RETI instruction is executed, the CPU performs the following steps, and transfers control to the address of the restored PC.

- (1) Restores the restored PC and PSW from EIPC and EIPSW because the EP bit of the PSW is 1.
- (2) Transfers control to the address of the restored PC and PSW.

The processing of the RETI instruction is shown below.

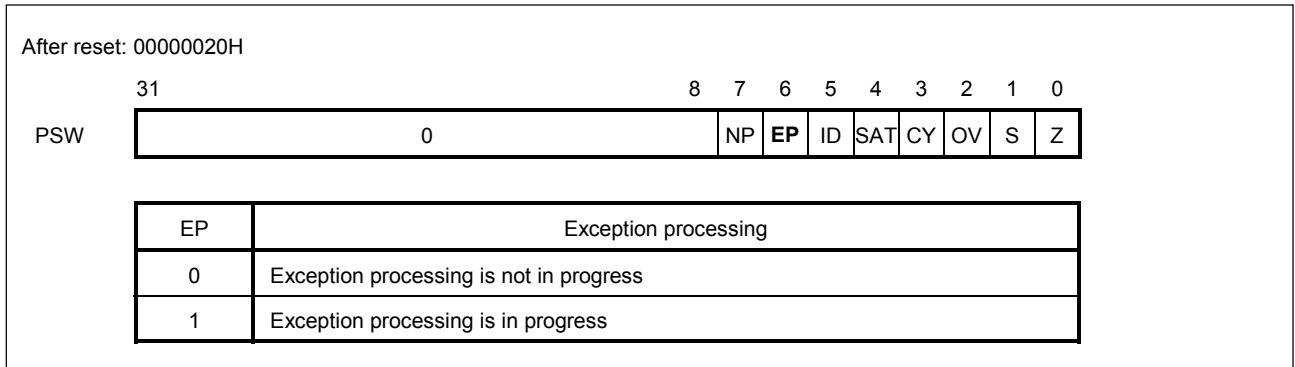
Figure 7-11. RETI Instruction Processing



7.4.3 EP flag

The EP flag in the PSW is a status flag used to indicate that exception processing is in progress. It is set when an exception occurs, and disables interrupts.

Figure 7-12. EP Flag (EP)



7.5 Exception Trap

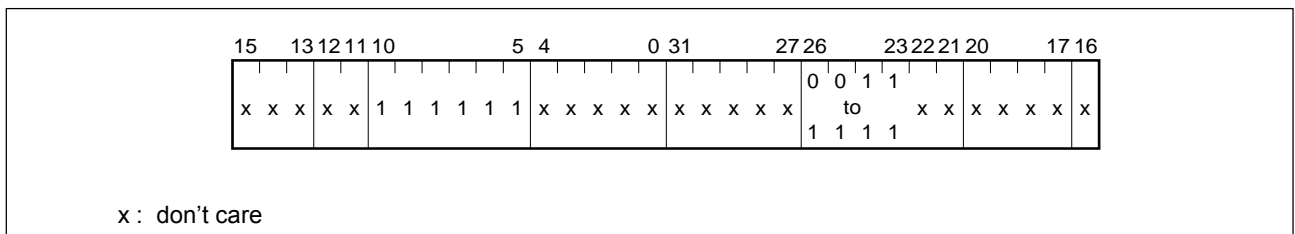
The exception trap is an interrupt that is requested when illegal execution of an instruction takes place. In the V850/SF1, an illegal op code exception (ILGOP: ILeGal OPcode trap) is considered as an exception trap.

- Illegal op code exception: Occurs if the sub op code field of an instruction to be executed next is not a valid op code.

7.5.1 Illegal op code definition

An illegal op code is defined to be a 32-bit word with bits 5 to 10 being 111111B and bits 23 to 26 being 0011B to 1111B.

Figure 7-13. Illegal Op Code



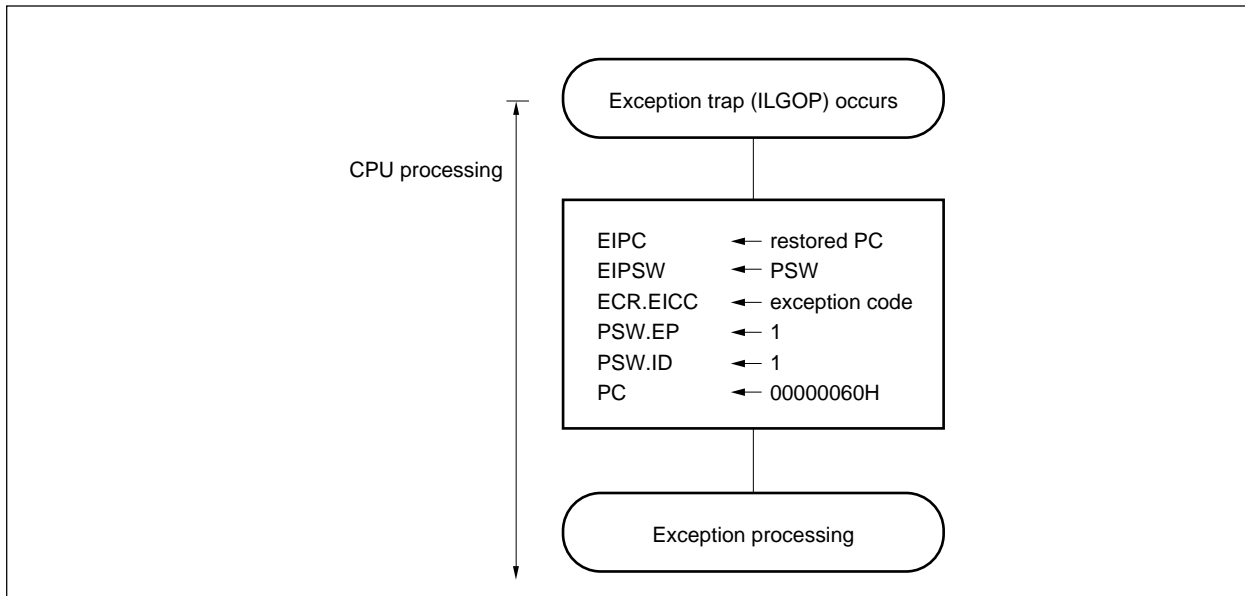
7.5.2 Operation

If an exception trap occurs, the CPU performs the following processing, and transfers control to the handler routine.

- (1) Saves the restored PC to EIPC.
- (2) Saves the current PSW to EIPSW.
- (3) Writes an exception code (0060H) to the lower 16 bits (EICC) of ECR.
- (4) Sets the EP and ID bits of the PSW.
- (5) Loads the handler address (00000060H) for the exception trap routine to the PC, and transfers control.

How the exception trap is processed is shown below.

Figure 7-14. Exception Trap Processing



7.5.3 Restore

To restore or return execution from the exception trap, the RETI instruction is used.

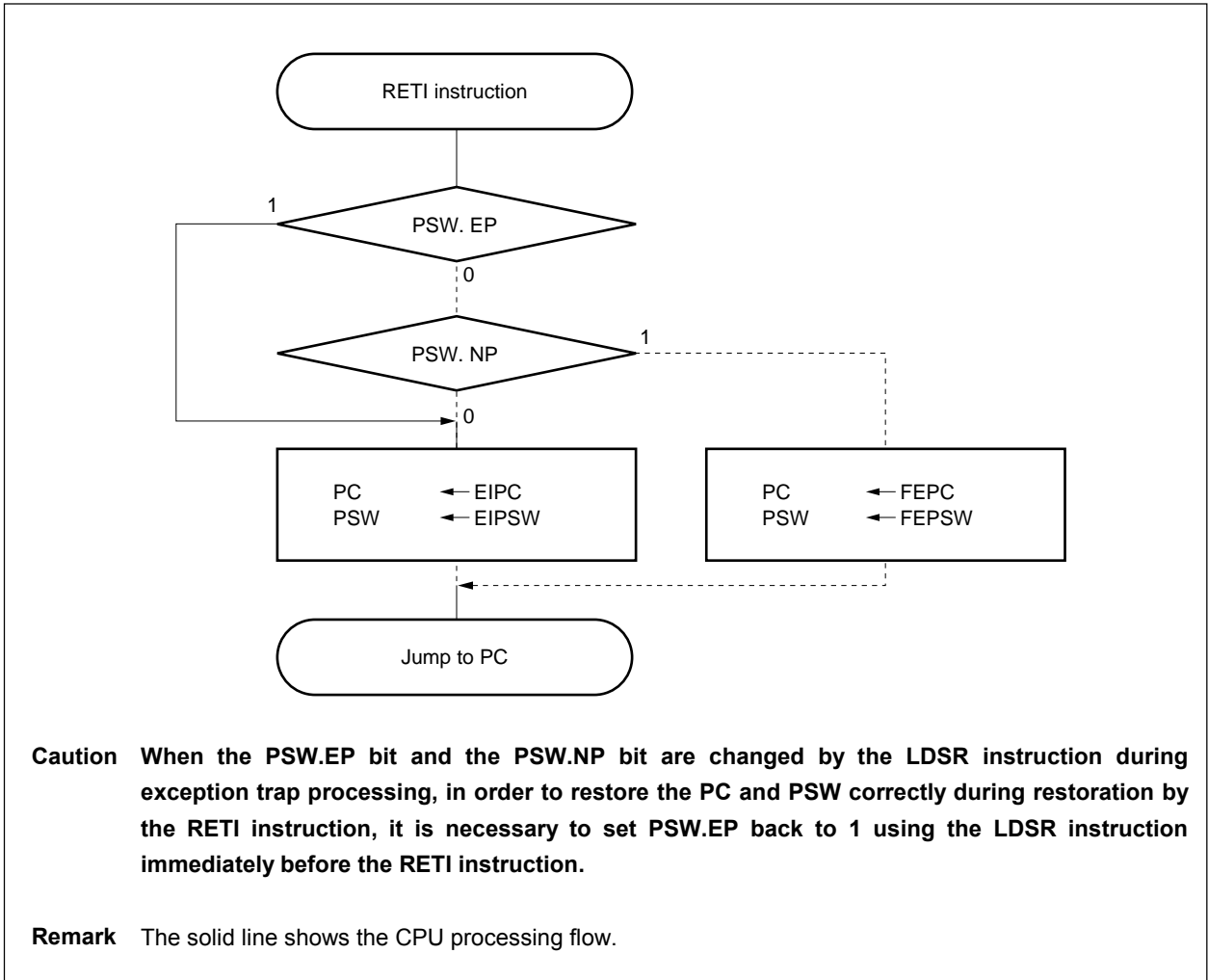
Operation of RETI instruction

When the RETI instruction is executed, the CPU performs the following processing, and transfers control to the address of the restored PC.

- (1) Restores the restored PC and PSW from EIPC and EIPSW because the EP bit of the PSW is 1.
- (2) Transfers control to the address of the restored PC and PSW.

The processing of the RETI instruction is shown below.

Figure 7-15. RETI Instruction Processing



7.6 Priority Control

7.6.1 Priorities of interrupts and exceptions

Table 7-3. Priorities of Interrupts and Exceptions

	RESET	NMI	INT	TRAP	ILGOP
RESET		*	*	*	*
NMI	×		←	←	←
INT	×	↑		←	←
TRAP	×	↑	↑		←
ILGOP	×	↑	↑	↑	

RESET: Reset

NMI: Non-maskable interrupt

INT: Maskable interrupt

TRAP: Software exception

ILGOP: Illegal op code exception

*: The item on the left ignores the item above.

×: The item on the left is ignored by the item above.

↑: The item above is higher than the item on the left in priority.

←: The item on the left is higher than the item above in priority.

7.6.2 Multiple interrupt servicing

Multiple interrupt servicing is a function that allows the nesting of interrupts. If a higher priority interrupt is generated and acknowledged, it will be allowed to stop the interrupt service routine currently in progress. Execution of the original routine will resume once the higher priority interrupt routine is completed.

If an interrupt with a lower or equal priority is generated and a service routine is currently in progress, the later interrupt will be held pending.

Multiple interrupt servicing control is performed when interrupts are enabled (ID = 0). Even in an interrupt servicing routine, multiple interrupt control must be performed when interrupts are enabled (ID = 0). If a maskable interrupt or exception is generated during the service program of maskable interrupt or exception, EIPC and EIPSW must be saved.

The following example shows the procedure of interrupt nesting.

(1) To acknowledge maskable interrupts in service program

Service program of maskable interrupt or exception

```

...
...
• EIPC saved to memory or register
• EIPSW saved to memory or register
• EI instruction (enables interrupt acknowledgement)
...
...
• DI instruction (disables interrupt acknowledgement)
• Saved value restored to EIPSW
• Saved value restored to EIPC
• RETI instruction

```

← Acknowledges interrupt such as INTP input.

(2) To generate exception in service program

Service program of maskable interrupt or exception

```

...
...
• EIPC saved to memory or register
• EIPSW saved to memory or register
• EI instruction (enables interrupt acknowledgement)
...
• TRAP instruction
• Illegal op code
...
• Saved value restored to EIPSW
• Saved value restored to EIPC
• RETI instruction

```

← Acknowledges exception such as TRAP instruction.

← Acknowledges exception such as illegal op code.

Priorities 0 to 7 (0 is the highest) can be programmed for each maskable interrupt request for multiple interrupt processing control. To set a priority level, write values to the xxPRn0 to xxPRn2 bits of the interrupt request control register (xxICn) corresponding to each maskable interrupt request. At reset, the interrupt request is masked by the xxMKn bit, and the priority level is set to 7 by the xxPRn0 to xxPRn2 bits.

Remark xx: Identification name of each peripheral unit (WDT, P, WTNI, TM, CS, SER, ST, AD, DMA, WTN, CAN, KR)
n: Peripheral unit number (see Table 7-2)

Priorities of maskable interrupts

(High) Level 0 > Level 1 > Level 2 > Level 3 > Level 4 > Level 5 > Level 6 > Level 7 (Low)

Interrupt servicing that has been suspended as a result of multiple interrupt servicing is resumed after the interrupt servicing of the higher priority has been completed and the RETI instruction has been executed.

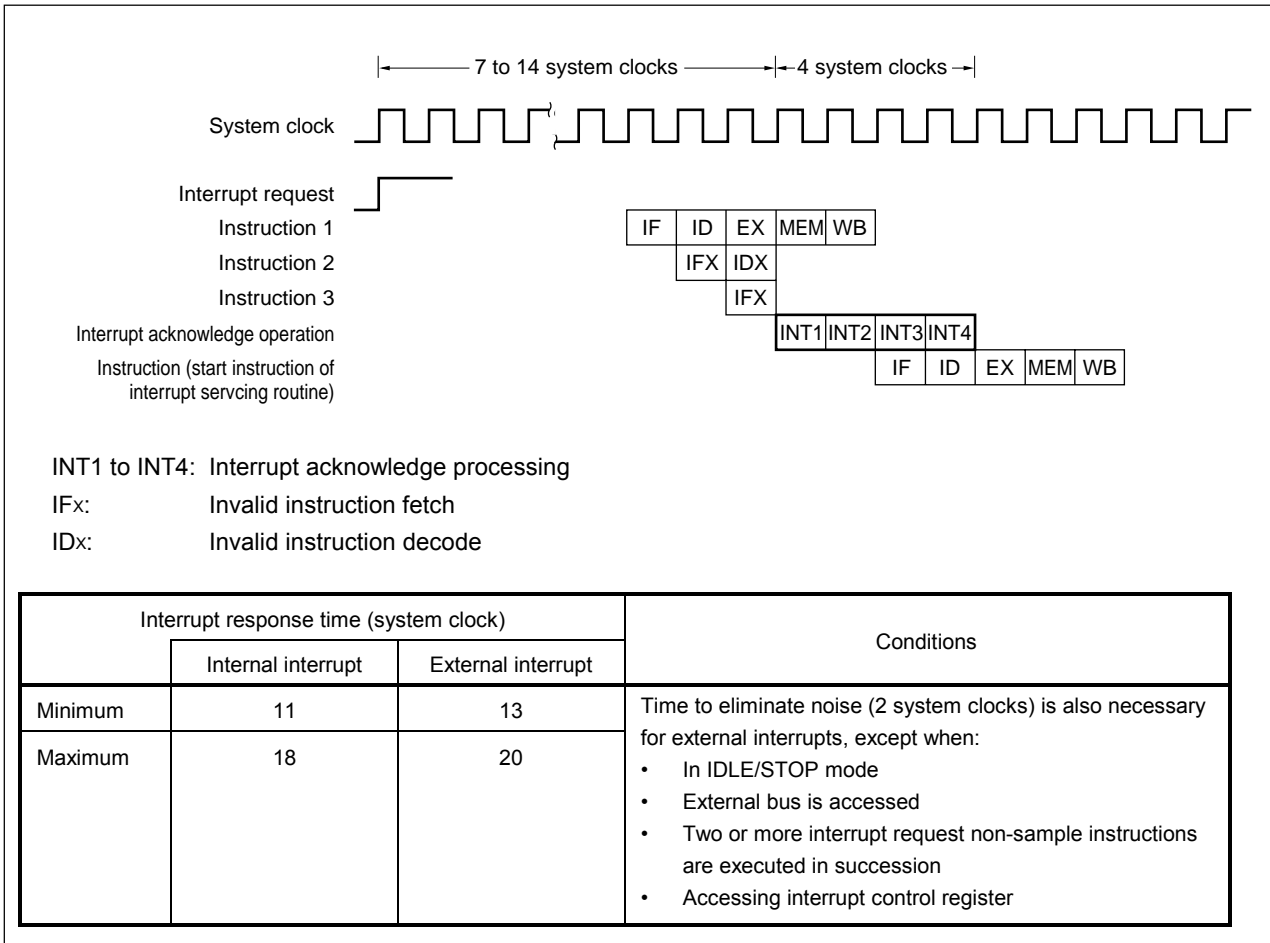
A pending interrupt request is acknowledged after the current interrupt servicing has been completed and the RETI instruction has been executed.

Caution In a non-maskable interrupt servicing routine (time until the RETI instruction is executed), maskable interrupts are held pending without being acknowledged.

7.7 Response Time

The following table describes the interrupt response time (from interrupt request generation to start of interrupt servicing).

Figure 7-16. Pipeline Operation at Interrupt Request Acknowledgement



7.8 Periods in Which Interrupts Are Not Acknowledged

An interrupt is acknowledged while an instruction is being executed. However, no interrupt will be acknowledged between an interrupt request non-sample instruction and the next instruction.

Interrupt request non-sample instruction

- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (vs. PSW)

★ (1) Acknowledgement of interrupt servicing following EI instruction

The V850/SF1 requires a minimum of 7 clocks from the occurrence of an interrupt request until the interrupt request is acknowledged as the interrupt request determination period. Since instructions can continue to be executed during this period, executing the DI (Disable Interrupt) instruction results in the interrupt disabled state. As a result, all interrupt requests are held pending until the EI (Enable Interrupt) instruction is executed.

Moreover, since an interrupt determination period is also required when the EI instruction is executed, at least 7 clocks are required between execution of the EI instruction and acknowledgement of the interrupt request. Therefore, if the DI instruction is executed before 7 clocks have elapsed following execution of the EI instruction, interrupts will be held pending. To reliably acknowledge interrupts, insert instructions whose execution clocks total 7 clocks or more between the EI instruction and DI instruction. This does not apply, however, to the following instructions.

- IDLE/STOP mode setting
- EI instruction, DI instruction
- RETI instruction
- LDSR instruction (for PSW register)
- Access to interrupt control register (xxICn)

Example: When EI instruction servicing is not enabled

[Program Example]

```

DI
  ⋮
  ⋮           ;MK flag = 0 (interrupt request enable)
  ⋮           ;Interrupt request occurrence (IF flag = 1)
EI
JR  LP1      ; 7 clocks have not elapsed between EI instruction and DI instruction (3 clocks)
  ⋮
  ⋮
LP1:
DI           ← ; Interrupt request not acknowledged
  ⋮

```

[Workaround Program Example]

```

DI
:           ;MK flag = 0 (interrupt request enable)
:           ; Interrupt request occurrence (IF flag = 1)
EI
NOP         ;1 system clock
NOP         ;1 system clock
NOP         ;1 system clock
NOP         ;1 system clock
JR LP1     ;3 system clocks (branching to LP1 routine)
:
:
LP1:
DI         ← ; Interrupt servicing executed at 8th system clock following execution of EI
:         instruction

```

7.9 Key Interrupt Function

A key interrupt can be generated by inputting a falling edge to the key input pins (KR0 to KR7) by setting the key return mode register (KRM). The key return mode register (KRM) includes 5 bits. The KRM0 bit controls the KR0 to KR3 signals in 4-bit units and the KRM4 to KRM7 bits control corresponding signals from KR4 to KR7 (arbitrary setting from 4 to 8 bits is possible).

This register can be read/written in 8- or 1-bit units.

After reset: 00H R/W Address: FFFF3D0H

	7	6	5	4	3	2	1	0
KRM	KRM7	KRM6	KRM5	KRM4	0	0	0	KRM0

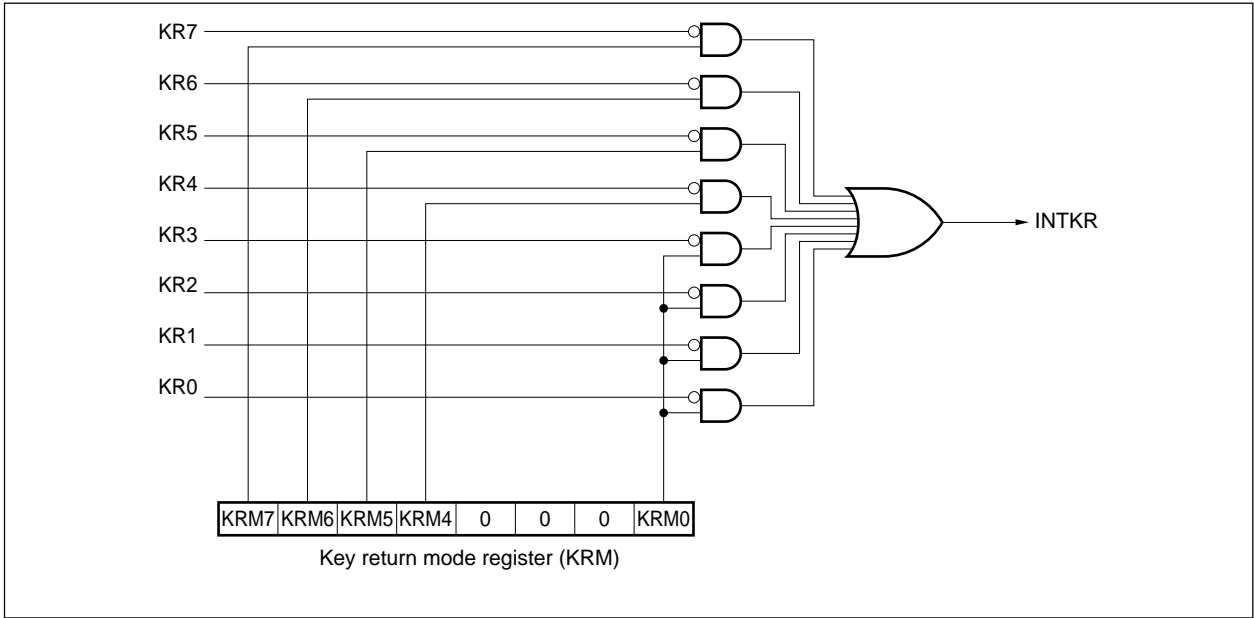
KRMn	Key return mode control
0	Key return signal not detected
1	Key return signal detected

Caution If the key return mode register (KRM) is changed, an interrupt request flag may be set. To avoid this flag being set, change the KRM register after disabling interrupts, and then enable interrupts after clearing the interrupt request flag.

Table 7-4. Description of Key Return Detection Pin

Flag	Pin Description
KRM0	Controls KR0 to KR3 signals in 4-bit units
KRM4	Controls KR4 signal in 1-bit units
KRM5	Controls KR5 signal in 1-bit units
KRM6	Controls KR6 signal in 1-bit units
KRM7	Controls KR7 signal in 1-bit units

Figure 7-17. Key Return Block Diagram



CHAPTER 8 TIMER/COUNTER FUNCTION

8.1 16-Bit Timer (TM0, TM1, TM7)

Remark n = 0, 1, 7 in section 8.1.

8.1.1 Outline

- 16-bit capture/compare registers: 2 (CRn0, CRn1)
- Independent capture/trigger inputs: 2 (TIn0, TIn1)
- Support of output of capture/match interrupt request signals (INTTMn0, INTTMn1)
- Event input (shared with TIn0) via digital noise eliminator and support of edge specifications
- Timer output operated by match detection: 1 each (TOn)

When using the P104/TO0, P107/TO1, and P100/TO7 pins as TO0, TO1, and TO7 (timer output), set the value of port 10 (P10) to 0 (port mode output) and the port 10 mode register (PM10) to 0. The logical sum (ORed) value of the output of a port and a timer is output.

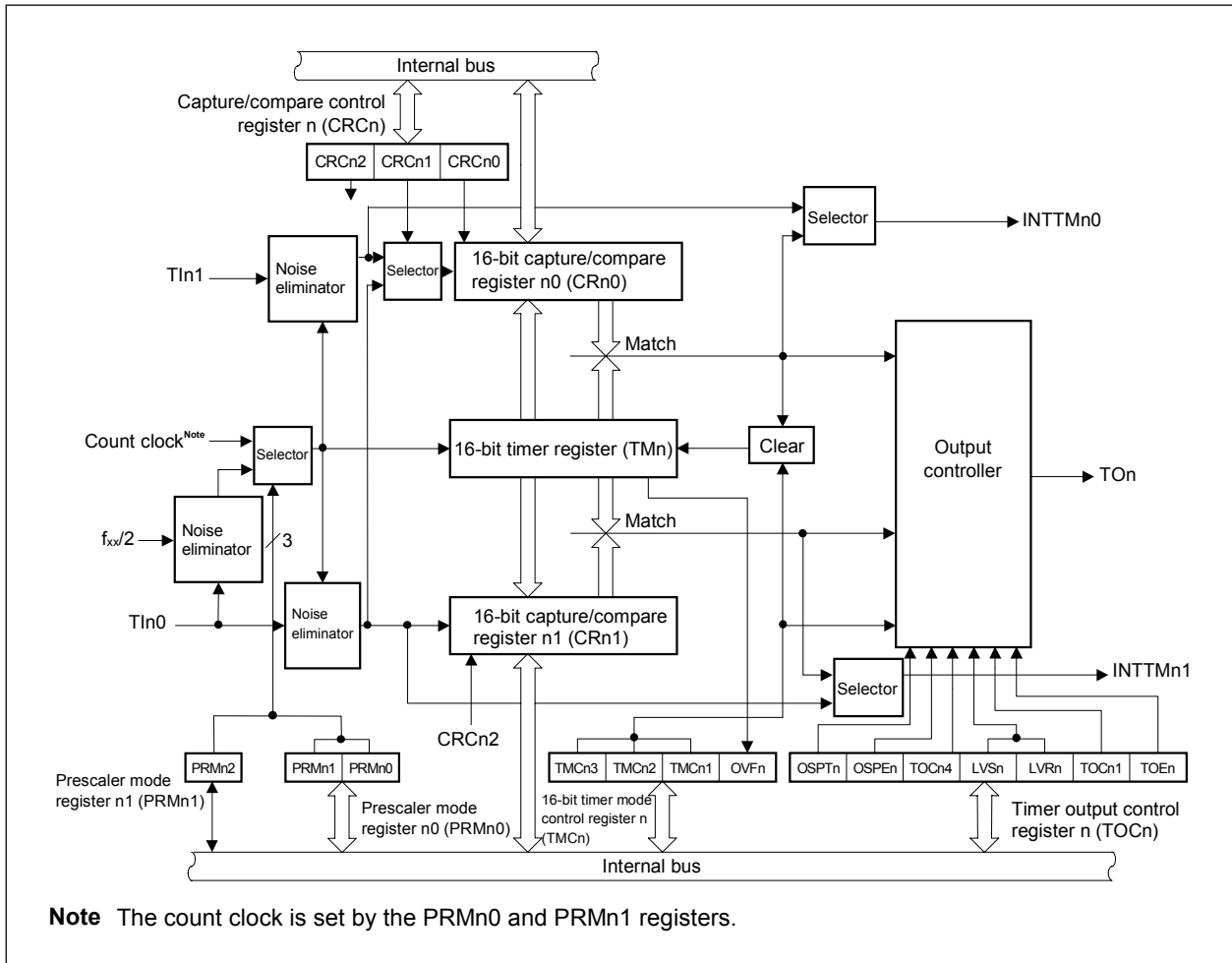
8.1.2 Function

TM0, TM1, and TM7 have the following functions.

- Interval timer
- PPG output
- Pulse width measurement
- External event counter
- Square wave output
- One-shot pulse output

Figure 8-1 shows the block diagram.

Figure 8-1. Block Diagram of TM0, TM1, and TM7

**(1) Interval timer**

Generates an interrupt at preset time intervals.

(2) PPG output

Can output a square wave with a frequency and output-pulse width that can be set arbitrarily.

(3) Pulse width measurement

Can measure the pulse width of a signal input from an external source.

(4) External event counter

Can measure the number of pulses of a signal input from an external source.

(5) Square wave output

Can output a square wave of any frequency.

(6) One-shot pulse output

Can output a one-shot pulse with any output pulse width.

8.1.3 Configuration

Timers 0, 1, and 7 include the following hardware.

Table 8-1. Configuration of Timers 0, 1, and 7

Item	Configuration
Timer registers	16 bits × 3 (TM0, TM1, TM7)
Registers	Capture/compare registers: 16 bits × 6 (CRn0, CRn1)
Timer outputs	3 (TO0, TO1, TO7)
Control registers	16-bit timer mode control register n (TMCn) Capture/compare control register n (CRCn) 16-bit timer output control register n (TOCn) Prescaler mode registers n0, n1 (PRMn0, PRMn1)

(1) 16-bit timer registers 0, 1, 7 (TM0, TM1, TM7)

TMn is a 16-bit read-only register that counts count pulses.

The counter is incremented in synchronization with the rising edge of the input clock. If the count value is read during operation, input of the count clock is temporarily stopped, and the count value at that point is read. The count value is reset to 0000H in the following cases:

- <1> At $\overline{\text{RESET}}$ input
- <2> If TMCn3 and TMCn2 are cleared
- <3> If the valid edge of TIn0 is input in the clear & start mode set by inputting the valid edge of TIn0
- <4> If TMn and CRn0 match in the clear & start mode set on a match between TMn and CRn0
- <5> If OSPTn is set or if the valid edge of TIn0 is input in the one-shot pulse output mode

(2) Capture/compare register n0 (CR00, CR10, CR70)

CRn0 is a 16-bit register that functions as a capture register and as a compare register. Whether this register functions as a capture or compare register is specified by bit 0 (CRn0) of the CRCn register.

(a) When using CRn0 as compare register

The value set to CRn0 is always compared with the count value of the TMn register. When the values of the two match, an interrupt request (INTTMn0) is generated. When TMn is used as an interval timer, CRn0 can also be used as a register that holds the interval time.

(b) When using CRn0 as capture register

The valid edge of the TIn0 or TIn1 pin can be selected as a capture trigger. The valid edge for TIn0 or TIn1 is set by using the PRMn0 register.

When the valid edge for TIn0 pin is specified as the capture trigger, refer to **Table 8-2**. When the valid edge for TIn1 pin is specified as the capture trigger, refer to **Table 8-3**.

Table 8-2. Valid Edge of TIn0 Pin and Capture Trigger of CRn0

ESn01	ESn00	Valid Edge of TIn0 Pin	CRn0 Capture Trigger
0	0	Falling edge	Rising edge
0	1	Rising edge	Falling edge
1	0	Setting prohibited	Setting prohibited
1	1	Both rising and falling edges	No capture operation

Table 8-3. Valid Edge of TIn1 Pin and Capture Trigger of CRn0

ESn11	ESn10	Valid Edge of TIn1 Pin	CRn0 Capture Trigger
0	0	Falling edge	Falling edge
0	1	Rising edge	Rising edge
1	0	Setting prohibited	Setting prohibited
1	1	Both rising and falling edges	Both rising and falling edges

CRn0 is set by a 16-bit memory manipulation instruction.

RESET input makes this register undefined.

★ **Caution** In the clear & start mode entered on a match between TMn and CRn0, set a value other than 0000H to CRn0. In the free running mode or the TIn0 valid edge clear mode, however, an interrupt request (INTTMn0) is generated after an overflow (FFFFH) when 0000H is set to CRn0.

(3) Capture/compare register n1 (CR01, CR11, CR71)

This is a 16-bit register that can be used as a capture register and a compare register. Whether it is used as a capture register or compare register is specified by bit 2 (CRCn2) of the CRCn register.

(a) When using CRn1 as compare register

The value set to CRn1 is always compared with the count value of TMn. When the values of the two match, an interrupt request (INTTMn1) is generated.

(b) When using CRn1 as capture register

The valid edge of the TIn1 pin can be selected as a capture trigger. The valid edge of TIn1 is specified using the PRMn0 register.

When the capture trigger is specified as the valid edge of TIn0, the relationship between the TIn0 valid edge and the CRn1 capture trigger is as follows.

Table 8-4. TIn0 Pin Valid Edge and CRn1 Capture Trigger

ESn01	ESn00	TIn0 Pin Valid Edge	CRn1 Capture Trigger
0	0	Falling edge	Falling edge
0	1	Rising Edge	Rising Edge
1	0	Setting prohibited	Setting prohibited
1	1	Both rising and falling edges	Both rising and falling edges

CRn1 is set by a 16-bit memory manipulation instruction.

RESET input makes this register undefined.

- ★ **Caution** In the clear & start mode entered on a match between TMn and CRn0, set a value other than 0000H to CRn1. In the free running mode or the TIn1 valid edge clear mode, however, an interrupt request (INTTMn1) is generated after an overflow (FFFFH) when 0000H is set to CRn1.

8.1.4 Timer 0, 1, 7 control registers

The registers to control timers 0, 1, 7 are shown below.

- 16-bit timer mode control register n (TMCn)
- Capture/compare control register n (CRCn)
- 16-bit timer output control register n (TOCn)
- Prescaler mode registers n0, n1 (PRMn0, PRMn1)

(1) 16-bit timer mode control registers 0, 1, 7 (TMC0, TMC1, TMC7)

TMCn specifies the operation mode of the 16-bit timer, and the clear mode, output timing, and overflow detection of the 16-bit timer register n.

TMCn is set by an 8- or 1-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input clears TMC0, TMC1, and TMC7 to 00H.

Caution 16-bit timer register n starts operating when a value other than 0, 0 (operation stop mode) is set to TMCn2 and TMCn3. To stop the operation, set 0, 0 to TMCn2 and TMCn3.

After reset: 00H R/W

Address: FFFF208H, FFFF218H, FFFF3A8H

7 6 5 4 3 2 1 0

TMCn

0	0	0	0	TMCn3	TMCn2	TMCn1	OVFn
---	---	---	---	-------	-------	-------	------

TMCn3	TMCn2	TMCn1	Selects operation mode and clear mode	Selects TOn output timing	Generation of interrupt
0	0	0	Operation stops (TMn is cleared to 0)	Not affected	Not generated
0	0	1			
0	1	0	Free running mode	Match between TMn and CRn0 or match between TMn and CRn1	Generated on match between TMn and CRn0 and match between TMn and CRn1
0	1	1		Match between TMn and CRn0, match between TMn and CRn1, or valid edge of TIn0	
1	0	0	Clears and starts at valid edge of TIn0	Match between TMn and CRn0 or match between TMn and CRn1	
1	0	1		Match between TMn and CRn0, match between TMn and CRn1, or valid edge of TIn0	
1	1	0	Clears and starts on match between TMn and CRn0	Match between TMn and CRn0 or match between TMn and CRn1	
1	1	1		Match between TMn and CRn0, match between TMn and CRn1, or valid edge of TIn0	

OVFn	Detection of overflow of 16-bit timer register n
0	Did not overflow
1	Overflow occurred

- Cautions**
1. When a bit other than the OVF_n flag is written, be sure to stop the timer operation.
 2. The valid edge of the TIn0 pin is set by using prescaler mode register n0 (PRMn0).
 3. When a mode in which the timer is cleared and started on a match between TM_n and CRn0 is selected, the OVF_n flag is set to 1 when the count value of TM_n changes from FFFFH to 0000H with CRn0 set to FFFFH.

Remark

TOn: Output pin of timer n
TIn0: Input pin of timer n
TMn: 16-bit timer register n
CRn0: Compare register n0
CRn1: Compare register n1

(2) Capture/compare control registers 0, 1, 7 (CRC0, CRC1, CRC7)

CRCn controls the operation of capture/compare register n (CRn0 and CRn1).

CRCn is set by an 8- or 1-bit memory manipulation instruction.

RESET input clears CRC0, CRC1, and CRC7 to 00H.

After reset: 00H R/W

Address: FFFFF20AH, FFFFF21AH, FFFFF3AAH

	7	6	5	4	3	2	1	0
CRCn	0	0	0	0	0	CRCn2	CRCn1	CRCn0

CRCn2	Selects operation mode of CRn1
0	Operates as compare register
1	Operates as capture register

CRCn1	Selects capture trigger of CRn0
0	Captured at valid edge of TIn1
1	Captured in reverse phase of valid edge of TIn0

CRCn0	Selects operation mode of CRn0
0	Operates as compare register
1	Operates as capture register

- Cautions**
1. Before setting CRCn, be sure to stop the timer operation.
 2. When the mode in which the timer is cleared and started on a match between TMn and CRn0 is selected by 16-bit timer mode control register n (TMCn), do not specify CRn0 as a capture register.
 3. When both the rising edge and falling edge are specified for the TIn0 valid edge, the capture operation does not work.
 4. For the capture trigger, a pulse longer than twice the count clock selected by prescaler mode registers 0n, 1n (PRM0n, PRM1n) is required for the signals from TIn0 and T2n1 to perform the capture operation correctly.

(3) 16-bit timer output control registers 0, 1, 7 (TOC0, TOC1, TOC7)

TOCn controls the operation of the timer n output controller by setting or resetting the R-S flip-flop (LV0), enabling or disabling inverse output, enabling or disabling output of timer n, enabling or disabling one-shot pulse output operation, and selecting an output trigger for a one-shot pulse by software.

TOCn is set by an 8- or 1-bit memory manipulation instruction.

RESET input clears TOC0, TOC1, and TOC7 to 00H.

After reset: 00H R/W

Address: FFFFF20CH, FFFFF21CH, FFFFF3ACH

	7	6	5	4	3	2	1	0
TOCn	0	OSPTn	OSPEn	TOCn4	LVSn	LVRn	TOCn1	TOEn

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

OSPEn	Controls one-shot pulse output operation
0	Successive pulse output
1	One-shot pulse output ^{Note}

TOCn4	Controls timer output F/F on match between CRn1 and TMn
0	Inverse timer output F/F disabled
1	Inverse timer output F/F enabled

LVSn	LVRn	Sets status of timer output F/F of timer 0
0	0	Not affected
0	1	Timer output F/F reset (0)
1	0	Timer output F/F set (1)
1	1	Setting prohibited

TOCn1	Controls timer output F/F on match between CRn0 and TMn
0	Inverse timer output F/F disabled
1	Inverse timer output F/F enabled

TOEn	Controls output of timer n
0	Output disabled (output is fixed to 0 level)
1	Output enabled

Note The one-shot pulse output operates only in the free-running mode and in the clear & start mode entered upon detection of TIn0 valid edge.

- Cautions**
1. Before setting TOCn, be sure to stop the timer operation.
 2. LVSn and LVRn are 0 when read after data has been set to them.
 3. OSPTn is 0 when read because it is automatically cleared after data has been set.
 4. Do not set OSPTn (to 1) for other than one-shot pulse output.

(4) Prescaler mode registers 00, 01 (PRM00, PRM01)

PRM00 and PRM01 select the count clock of the 16-bit timer (TM0) and the valid edge of TI00 or TI01 input. PRM00 and PRM01 are set by an 8-bit memory manipulation instruction. RESET input clears PRM00 and PRM01 to 00H.

After reset: 00H R/W

Address: FFFFF206H

	7	6	5	4	3	2	1	0
PRM00	ES011	ES010	ES001	ES000	0	0	PRM01	PRM00

After reset: 00H R/W

Address: FFFFF20EH

	7	6	5	4	3	2	1	0
PRM01	0	0	0	0	0	0	0	PRM02

ES011	ES010	Selects valid edge of TI01
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ES001	ES000	Selects valid edge of TI00
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

PRM02	PRM01	PRM00	Count clock selection		
			Count clock	f _{xx}	
				16 MHz	8 MHz
0	0	0	f _{xx} /2	125 ns	250 ns
0	0	1	f _{xx} /16	1 μs	2 μs
0	1	0	INTWTNI	–	–
0	1	1	TI00 valid edge ^{Note}	–	–
1	0	0	f _{xx} /4	250 ns	500 ns
1	0	1	f _{xx} /64	4 μs	8 μs
1	1	0	f _{xx} /256	16 μs	32 μs
1	1	1	Setting prohibited	–	–

Note An external clock requires a pulse longer than twice the internal clock (f_{xx}/2).

- Cautions**
1. When selecting the valid edge of TI00 or TI01 as the count clock, do not specify the valid edge of TI00 or TI01 to clear and start the timer and as a capture trigger.
 2. Before setting data to PRM00 and PRM01, always stop the timer operation.
 3. If the 16-bit timer (TM0) operation is enabled by specifying the rising edge or both edges as the valid edge of the TI00 or TI01 pin while the TI00 or TI01 pin is high level immediately after system reset, the rising edge is detected immediately after specification of the rising edge or both edges. Care is therefore needed when pulling up the TI00 or TI01 pin. However, the rising edge is not detected when operation is enabled after it has been stopped.

(5) Prescaler mode registers m0, m1 (PRMm0, PRMm1)

PRMm0 and PRMm1 select the count clock of the 16-bit timer (TM1, TM7) and the valid edge of the TIm0, TIm1 input. PRMm0 and PRMm1 are set by an 8-bit memory manipulation instruction (m = 1, 7).

RESET input clears PRMm0 and PRMm1 to 00H.

After reset: 00H R/W Address: FFFF216H, FFFF3A6H

	7	6	5	4	3	2	1	0
PRMm0	ESm11	ESm10	ESm01	ESm00	0	0	PRMm1	PRMm0

After reset: 00H R/W Address: FFFF21EH, FFFF3AEH

	7	6	5	4	3	2	1	0
PRMm1	0	0	0	0	0	0	0	PRMm2

ESm11	ESm10	Selects valid edge of TIm1
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

ESm01	ESm00	Selects valid edge of TIm0
0	0	Falling edge
0	1	Rising edge
1	0	Setting prohibited
1	1	Both rising and falling edges

PRMm2	PRMm1	PRMm0	Count clock selection		
			Count clock	f _{xx}	
				16 MHz	8 MHz
0	0	0	f _{xx} /2	125 ns	250 ns
0	0	1	f _{xx} /4	250 ns	500 ns
0	1	0	f _{xx} /16	1 μs	2 μs
0	1	1	TIm0 valid edge ^{Note}	–	–
1	0	0	f _{xx} /32	2 μs	4 μs
1	0	1	f _{xx} /128	8 μs	16 μs
1	1	0	f _{xx} /256	16 μs	32 μs
1	1	1	Setting prohibited	–	–

Note An external clock requires a pulse longer than twice the internal clock (f_{xx}/2).

- Cautions**
1. When selecting the valid edge of TIm0, TIm1 as the count clock, do not specify the valid edge of TIm0, TIm1 to clear and start the timer and as a capture trigger.
 2. Before setting data to PRMm0, PRMm1, always stop the timer operation.
 3. If the 16-bit timer (TM1, TM7) operation is enabled by specifying the rising edge or both edges for the valid edge of the TIm0, TIm1 pin while the TIm0, TIm1 pin is high level immediately after system reset, the rising edge is detected immediately after specification of the rising edge or both edges. Care is therefore needed when pulling up the TIm0, TIm1 pin. However, the rising edge is not detected when operation is enabled after it has been stopped.

Remark m = 1, 7

8.2 16-Bit Timer (TM0, TM1, TM7) Operation

Remark n = 0, 1, 7 in section 8.2.

8.2.1 Operation as interval timer

TMn operates as an interval timer when 16-bit timer mode control register n (TMCn) and capture/compare control register n (CRCn) are set as shown in Figure 8-2.

In this case, TMn repeatedly generates an interrupt at the time interval specified by the count value preset to 16-bit capture/compare register n (CRn0).

When the count value of TMn matches the set value of CRn0, the value of TMn is cleared to 0, and the timer continues counting. At the same time, an interrupt request signal (INTTMn0) is generated.

The count clock of the 16-bit timer/event counter can be selected by bits 0 and 1 (PRMn0 and PRMn1) of prescaler mode register n0 (PRMn0) and by bits 0 (PRMn2) of prescaler mode register n1 (PRMn1).

Figure 8-2. Control Register Settings When TMn Operates as Interval Timer

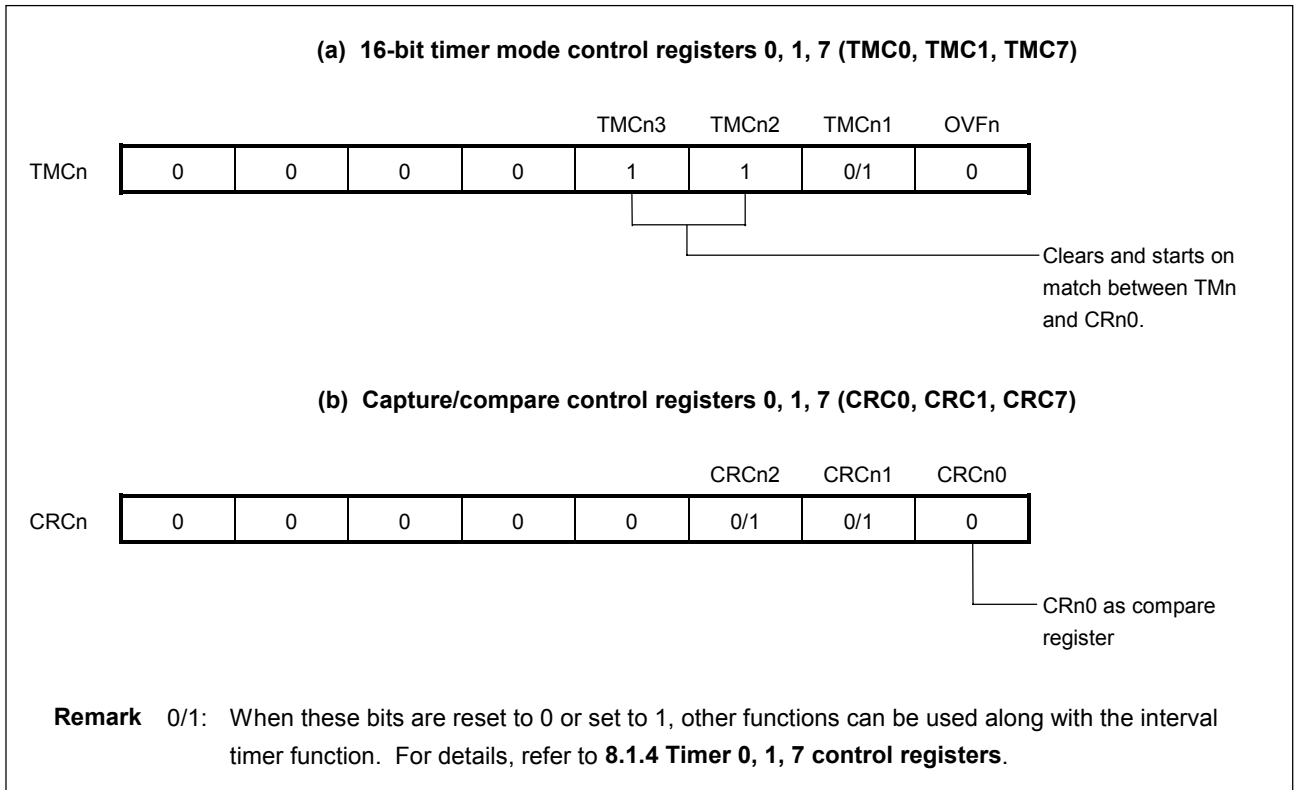


Figure 8-3. Configuration of Interval Timer

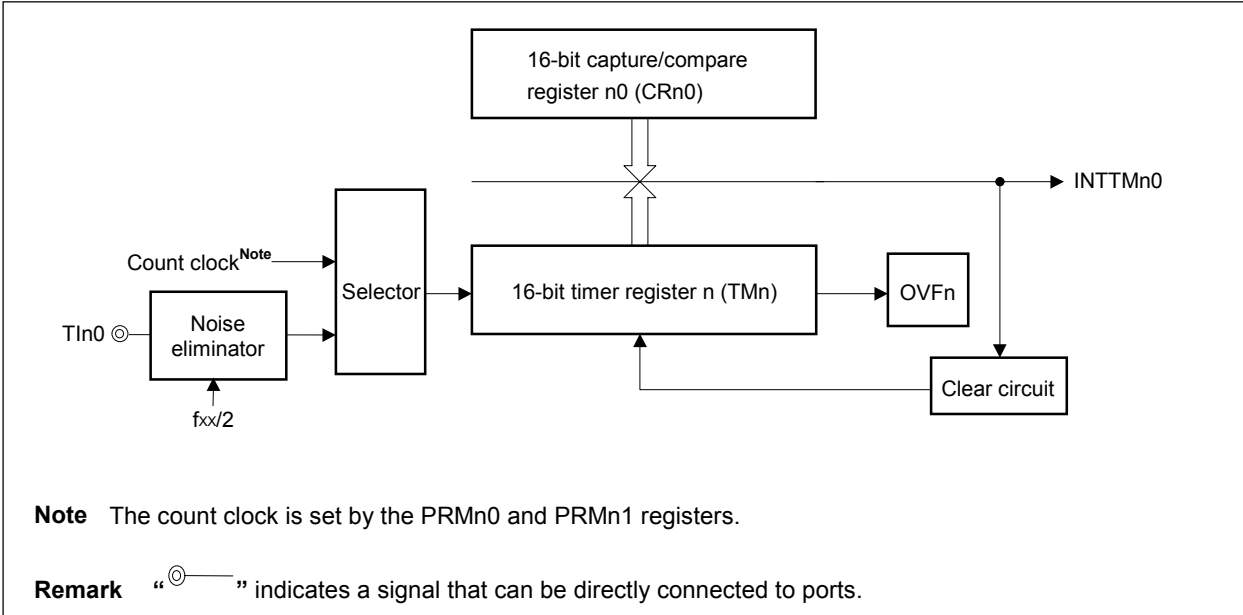
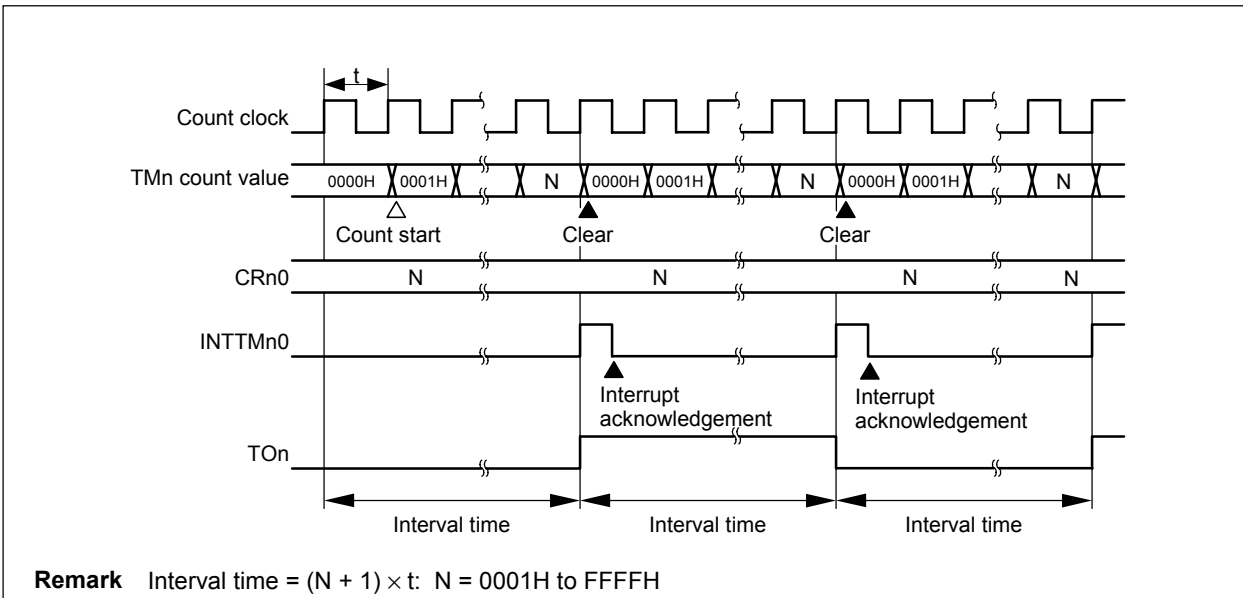


Figure 8-4. Timing of Interval Timer Operation

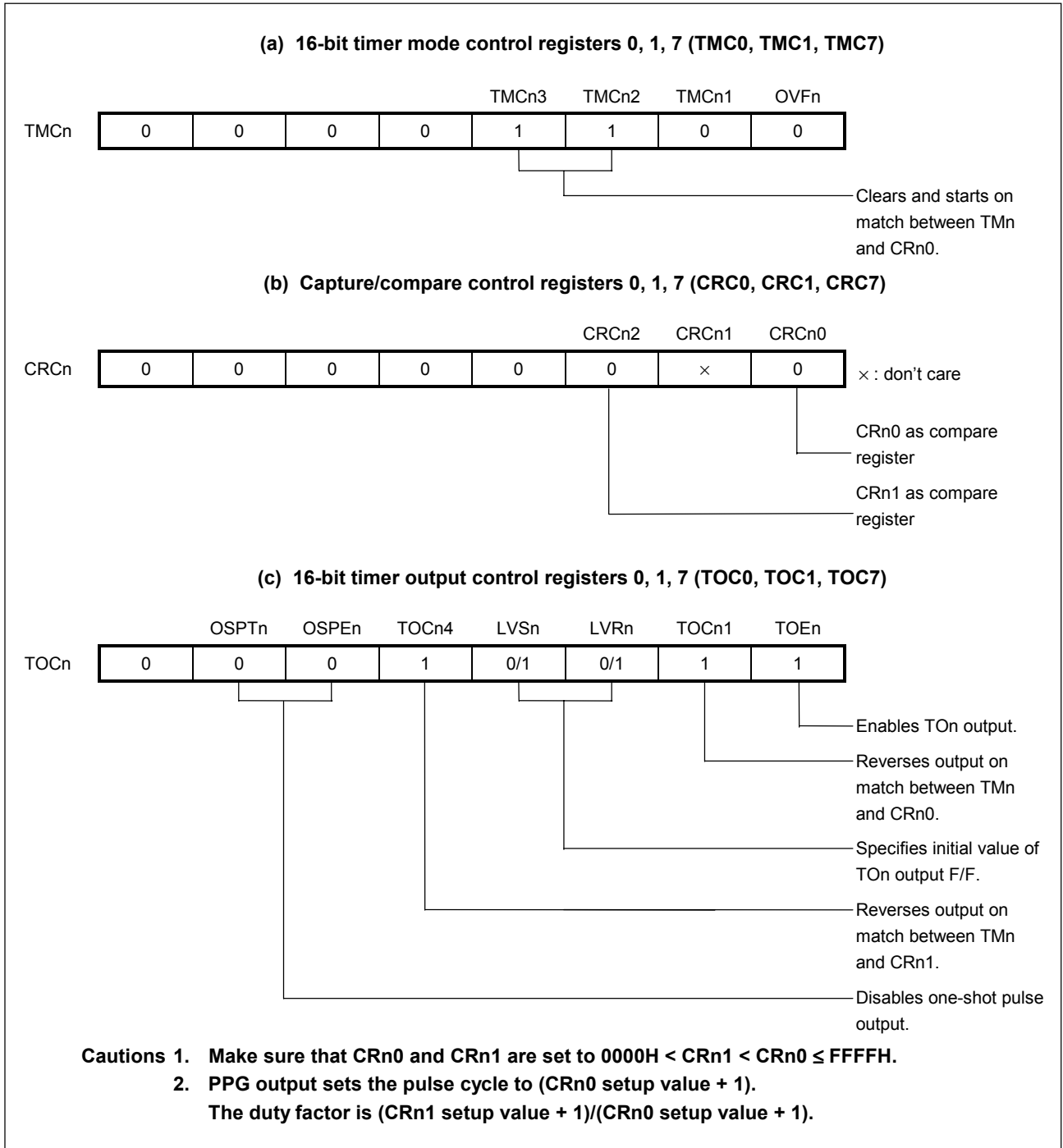


8.2.2 PPG output operation

TMn can be used for PPG (Programmable Pulse Generator) output by setting 16-bit timer mode control register n (TMCn) and capture/compare control register n (CRCn) as shown in Figure 8-5.

The PPG output function outputs a square wave from the TOn pin with a cycle specified by the count value preset to 16-bit capture/compare register n0 (CRn0) and a pulse width specified by the count value preset to 16-bit capture/compare register n1 (CRn1).

Figure 8-5. Control Register Settings in PPG Output Operation



8.2.3 Pulse width measurement

16-bit timer register n (TMn) can be used to measure the pulse widths of the signals input to the TIn0 and TIn1 pins.

Measurement can be carried out with TMn used as a free-running counter or by restarting the timer in synchronization with the edge of the signal input to the TIn0 pin.

(1) Pulse width measurement with free-running counter and one capture register

If the edge specified by prescaler mode register n0 (PRMn0) is input to the TIn0 pin when 16-bit timer register n (TMn) is used as a free-running counter (refer to **Figure 8-6**), the value of TMn is loaded to 16-bit capture/compare register n1 (CRn1), and an external interrupt request signal (INTTMn1) is set.

The edge is specified using bits 6 and 7 (ESn10 and ESn11) of prescaler mode register n0 (PRMn0). The rising edge, falling edge, or both edges can be selected.

The valid edge is detected by sampling with a count clock cycle selected by prescaler mode register n0 and n1 (PRMn0, PRMn1), and the capture operation is not performed until the valid level is detected two times, eliminating noise with a short pulse width.

Figure 8-6. Control Register Settings for Pulse Width Measurement with Free-Running Counter and One Capture Register

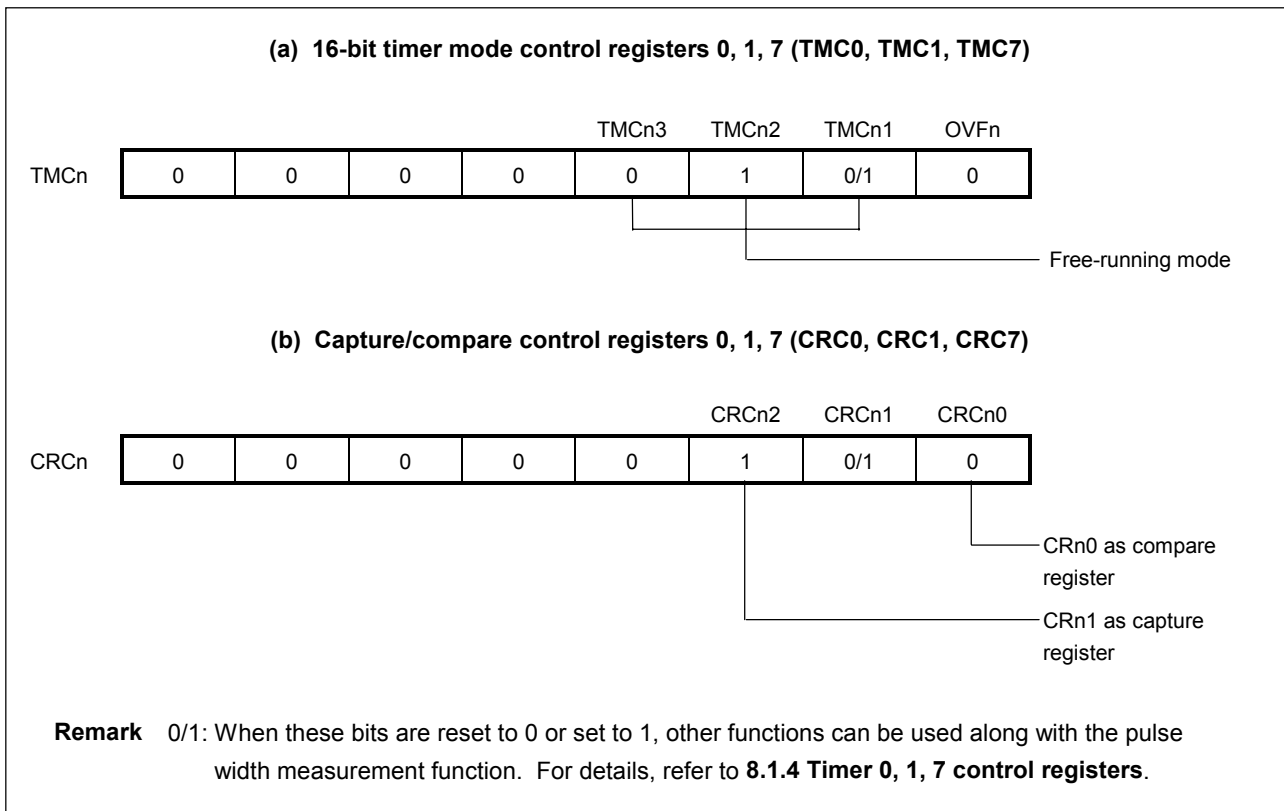


Figure 8-7. Configuration for Pulse Width Measurement with Free-Running Counter

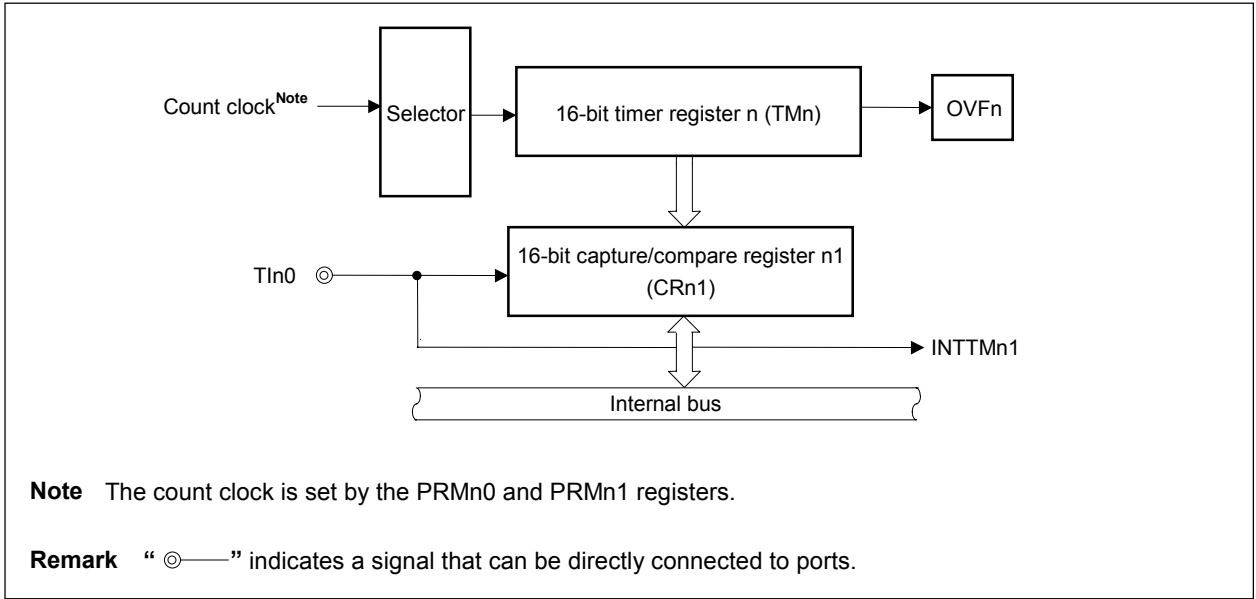
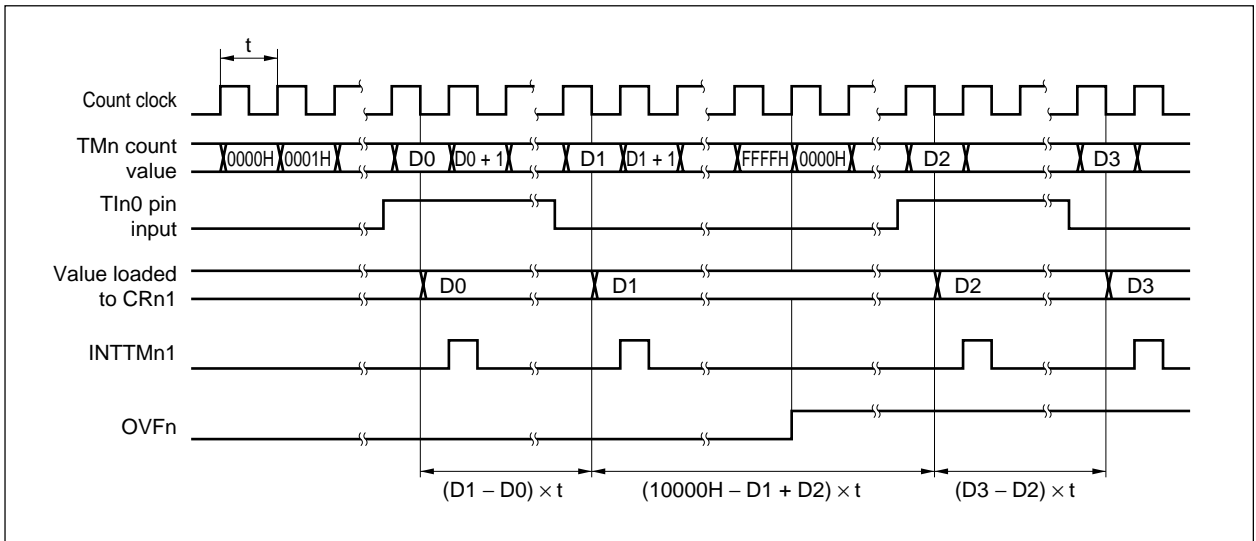


Figure 8-8. Timing of Pulse Width Measurement with Free-Running Counter and One Capture Register (with Both Edges Specified)



(2) Measurement of two pulse widths with free-running counter

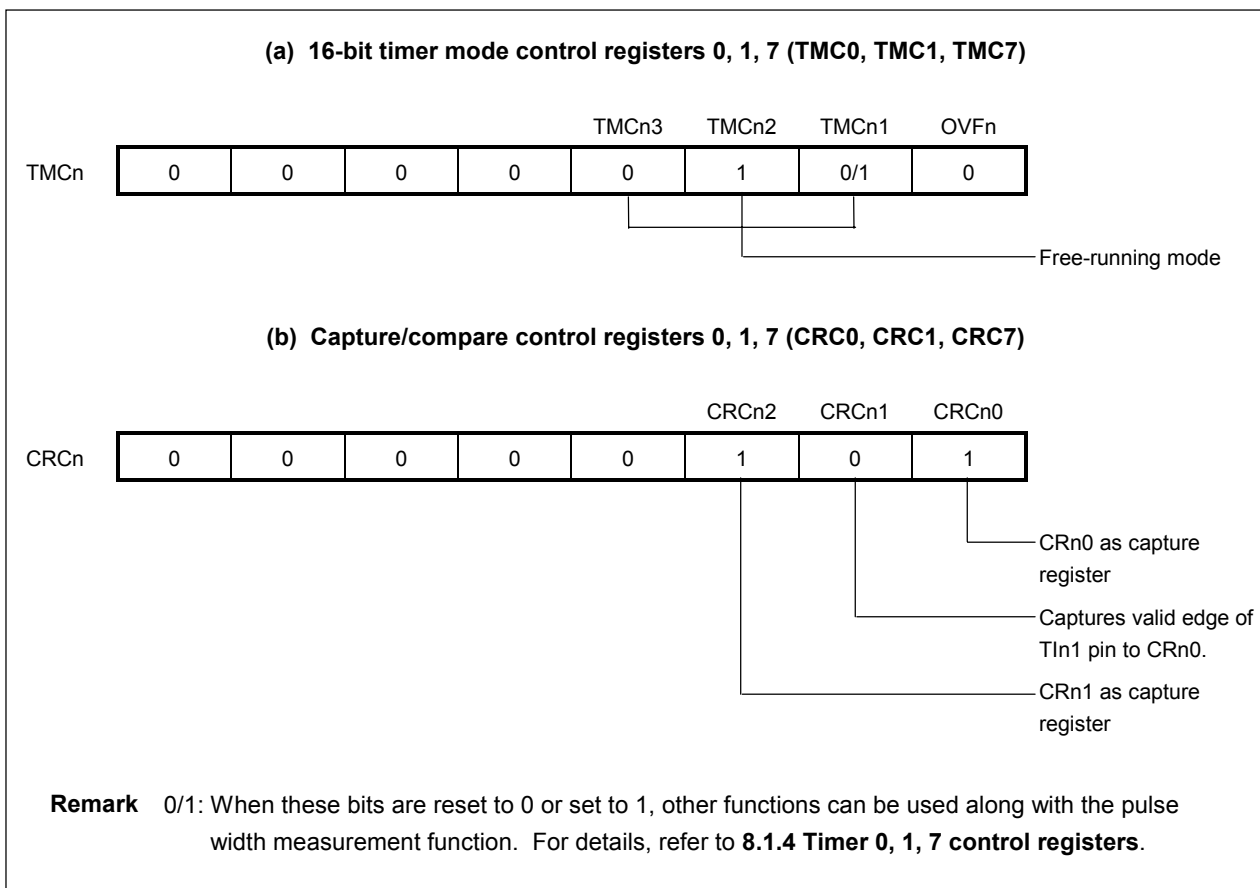
The pulse widths of the two signals respectively input to the TIn0 and TIn1 pins can be measured when 16-bit timer register n (TMn) is used as a free-running counter (refer to **Figure 8-9**).

When the edge specified by bits 4 and 5 (ESn00 and ESn01) of prescaler mode register n0 (PRMn0) is input to the TIn0 pin, the value of TMn is loaded to 16-bit capture/compare register n1 (CRn1) and an external interrupt request signal (INTTMn1) is set.

When the edge specified by bits 6 and 7 (ESn10 and ESn11) of PRMn0 is input to the TIn1 pin, the value of TMn is loaded to 16-bit capture/compare register n0 (CRn0), and an external interrupt request signal (INTTMn0) is set. The edges of the TIn0 and TIn1 pins are specified by bits 4 and 5 (ESn00 and ESn01) and bits 6 and 7 (ESn10 and ESn11) of PRMn0, respectively. The rising, falling, or both rising and falling edges can be specified.

The valid edge is detected by sampling with a count clock cycle selected by prescaler mode register n0 and n1 (PRMn0, PRMn1), and the capture operation is not performed until the valid level is detected two times, eliminating noise with a short pulse width.

Figure 8-9. Control Register Settings for Measurement of Two Pulse Widths with Free-Running Counter



- **Capture operation (free-running mode)**

The following figure illustrates the operation of the capture register when the capture trigger is input.

Figure 8-10. CRn1 Capture Operation with Rising Edge Specified

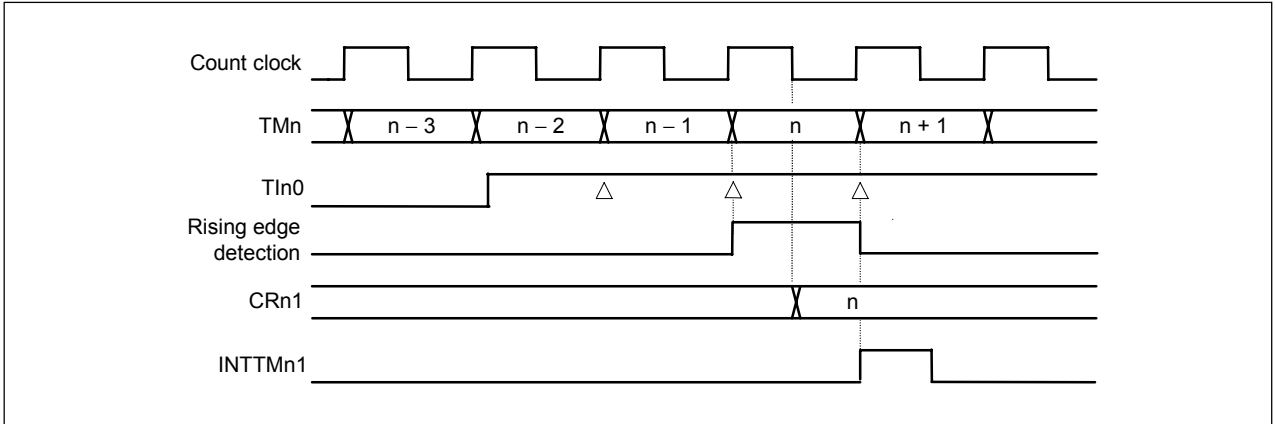
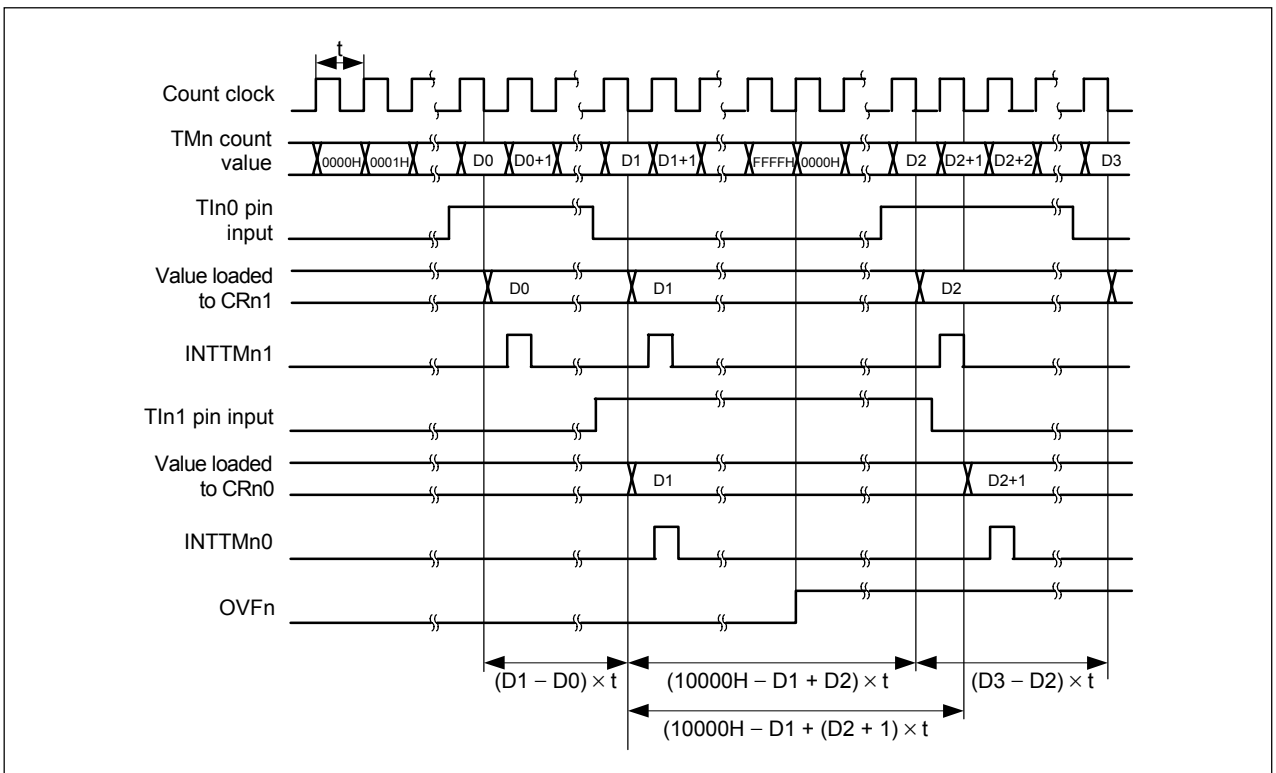


Figure 8-11. Timing of Pulse Width Measurement with Free-Running Counter (with Both Edges Specified)



(3) Pulse width measurement with free-running counter and two capture registers

When 16-bit timer register n (TMn) is used as a free-running counter (refer to **Figure 8-12**), the pulse width of the signal input to the TIn0 pin can be measured.

When the edge specified by bits 4 and 5 (ESn00 and ESn01) of prescaler mode register n0 (PRMn0) is input to the TIn0 pin, the value of TMn is loaded to 16-bit capture/compare register n1 (CRn1), and an external interrupt request signal (INTTMn1) is set.

The value of TMn is also loaded to 16-bit capture/compare register n0 (CRn0) when an edge that is the reverse of the one that triggers capturing to CRn1 is input.

The edge of the TIn0 pin is specified by bits 4 and 5 (ESn00 and ESn01) of prescaler mode register n0 (PRMn0). The rising or falling edge can be specified.

The valid edge of TIn0 is detected by sampling with a count clock cycle selected by prescaler mode register n0 and n1 (PRMn0, PRMn1), and the capture operation is not performed until the valid level is detected two times, eliminating noise with a short pulse width.

Caution If the valid edge of the TIn0 pin is specified to be both the rising and falling edges, capture/compare register n0 (CRn0) cannot perform a capture operation.

Figure 8-12. Control Register Settings for Pulse Width Measurement with Free-Running Counter and Two Capture Registers

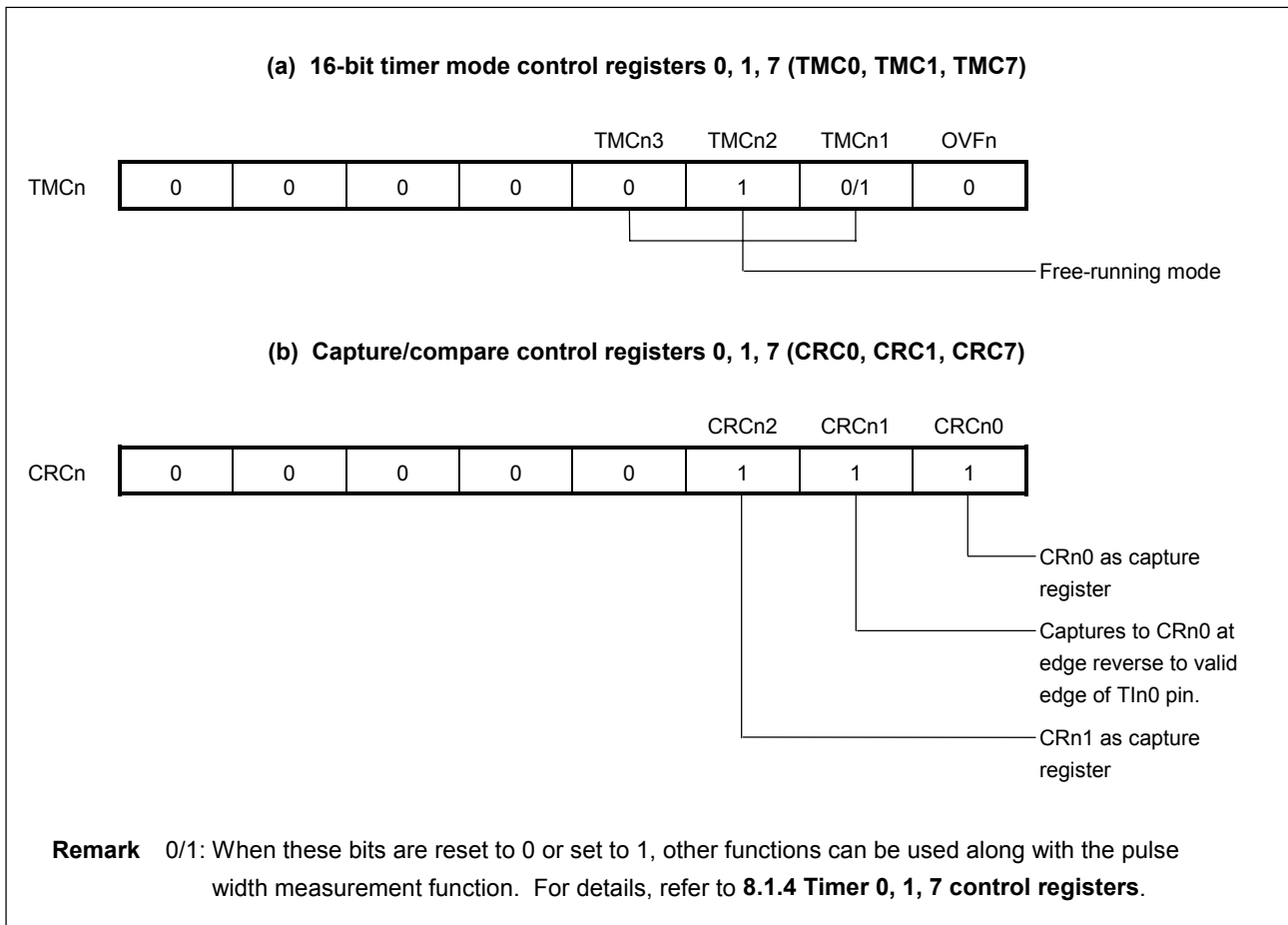
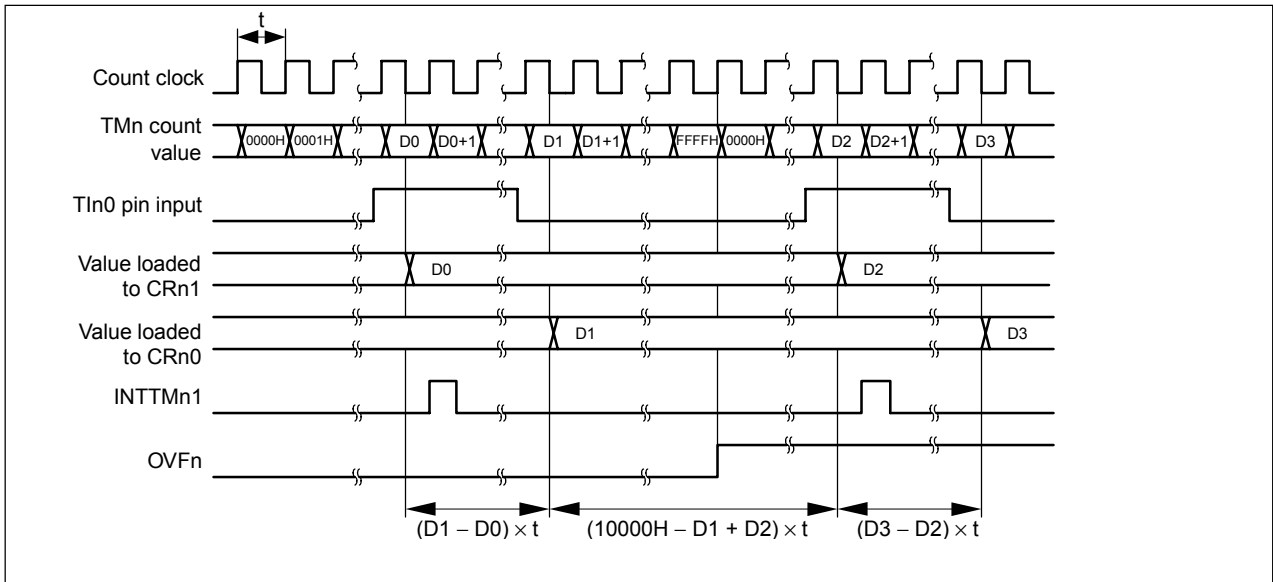


Figure 8-13. Timing of Pulse Width Measurement with Free-Running Counter and Two Capture Registers (with Rising Edge Specified)



(4) Pulse width measurement by restarting

When the valid edge of the TIn0 pin is detected, the pulse width of the signal input to the TIn0 pin can be measured by clearing 16-bit timer register n (TMn) once and then resuming counting after loading the count value of TMn to 16-bit capture/compare register n1 (CRn1). (See **Figure 8-15**)

The edge is specified by bits 4 and 5 (ESn00 and ESn01) of prescaler mode register n0 (PRMn0). The rising or falling edge can be specified.

The valid edge is detected by sampling with a count clock cycle selected by prescaler mode register n0 and n1 (PRMn0, PRMn1) and the capture operation is not performed until the valid level is detected two times, eliminating noise with a short pulse width.

Caution If the valid edge of the TIn0 pin is specified to be both the rising and falling edges, capture/compare register n0 (CRn0) cannot perform a capture operation.

Figure 8-14. Control Register Settings for Pulse Width Measurement by Restarting

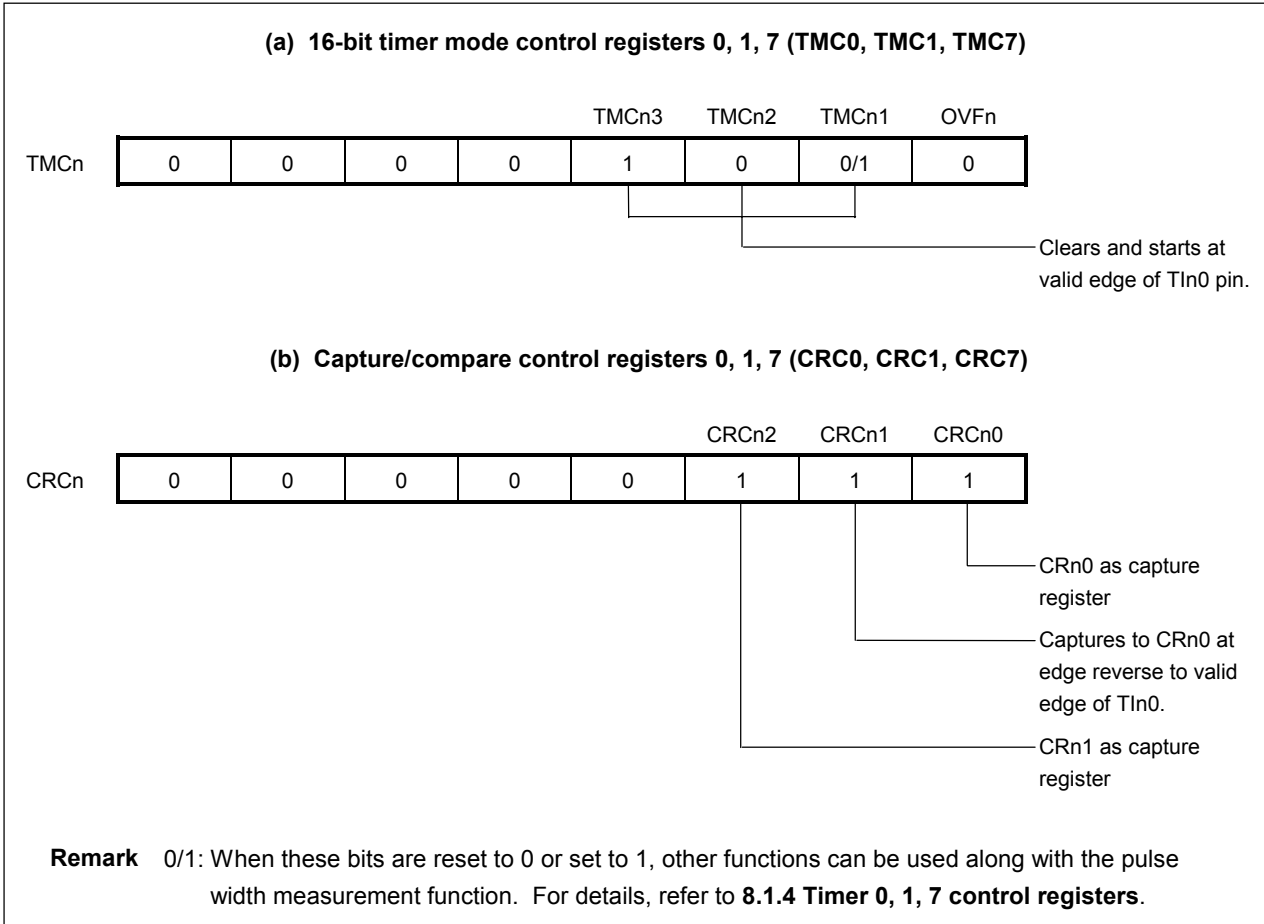
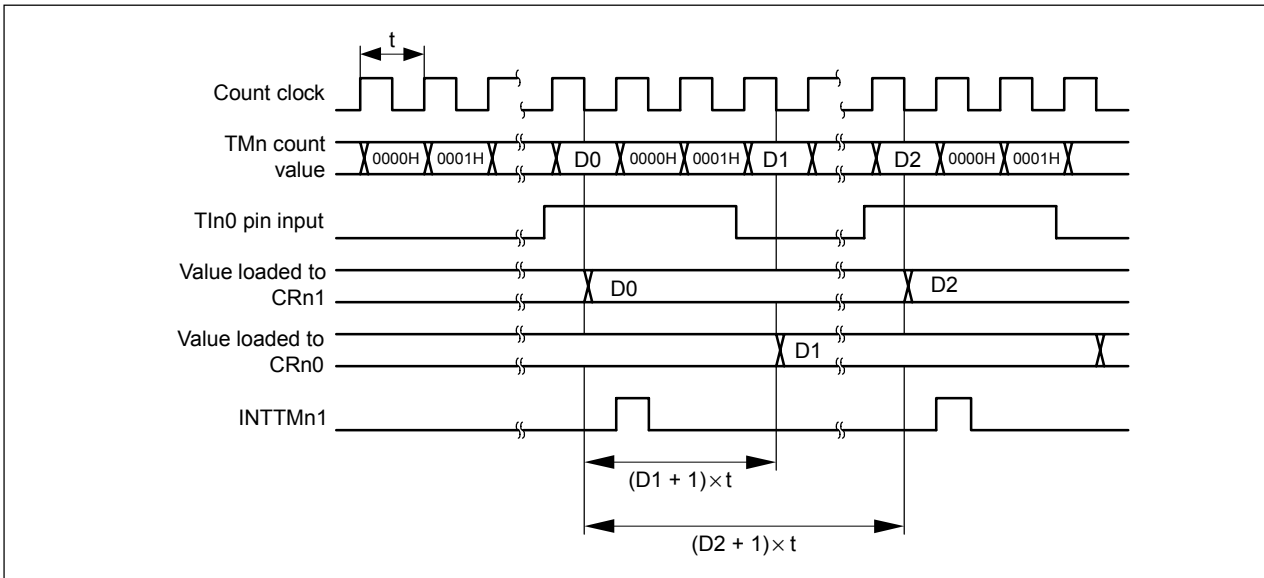


Figure 8-15. Timing of Pulse Width Measurement by Restarting (with Rising Edge Specified)



8.2.4 Operation as external event counter

TMn can be used as an external event counter that counts the number of clock pulses input to the TIn0 pin from an external source by using 16-bit timer register n (TMn).

Each time the valid edge specified by prescaler mode register n0 (PRMn0) has been input, TMn is incremented.

When the count value of TMn matches the value of 16-bit capture/compare register n0 (CRn0), TMn is cleared to 0, and an interrupt request signal (INTTMn0) is generated.

The edge is specified by bits 4 and 5 (ESn00 and ESn01) of prescaler mode register n0 (PRMn0). The rising, falling, or both the rising and falling edges can be specified.

The valid edge is detected by sampling with a count clock cycle of $f_{xx}/2$, and the capture operation is not performed until the valid level is detected two times, eliminating noise with a short pulse width.

Figure 8-16. Control Register Settings in External Event Counter Mode

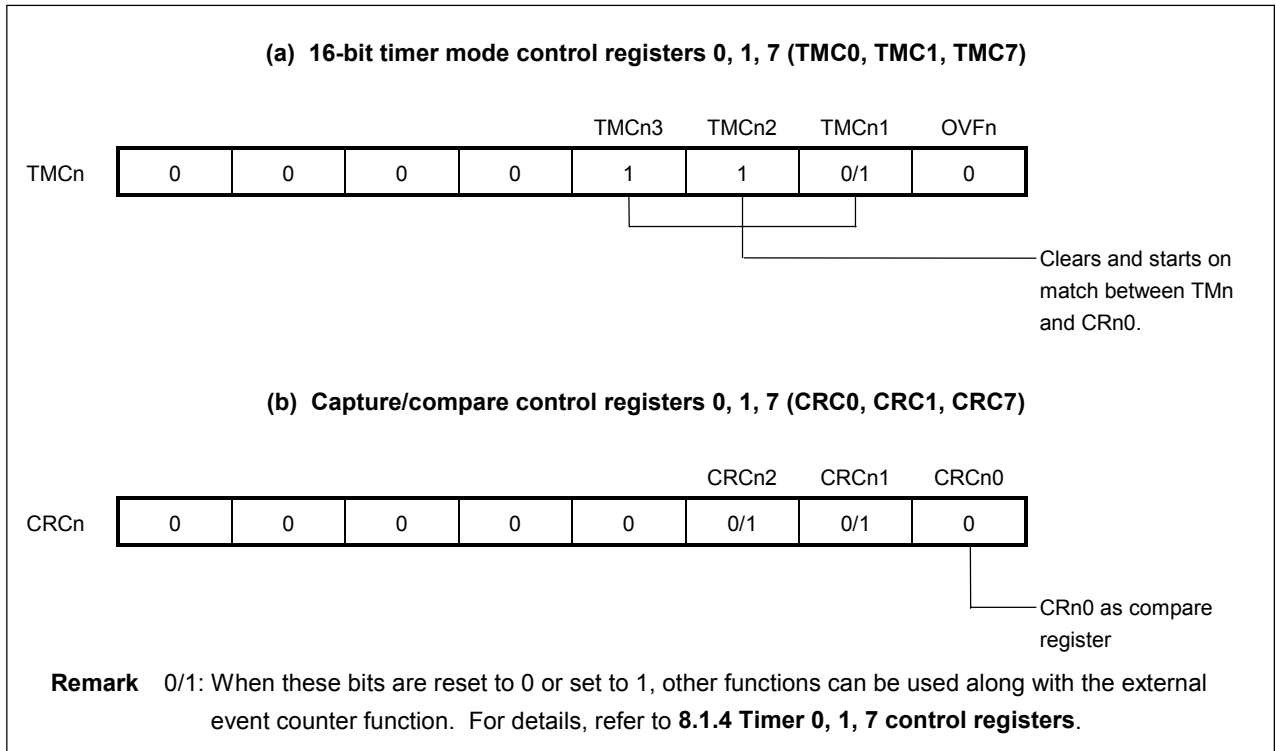


Figure 8-17. Configuration of External Event Counter

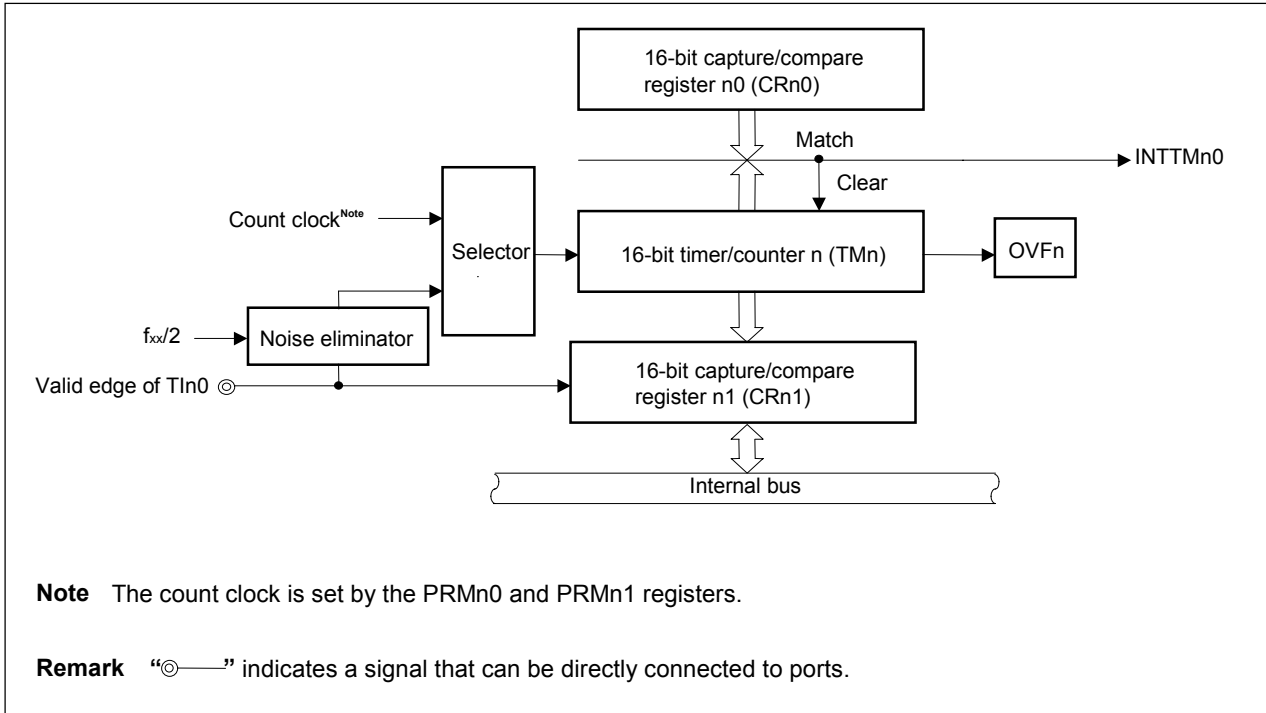
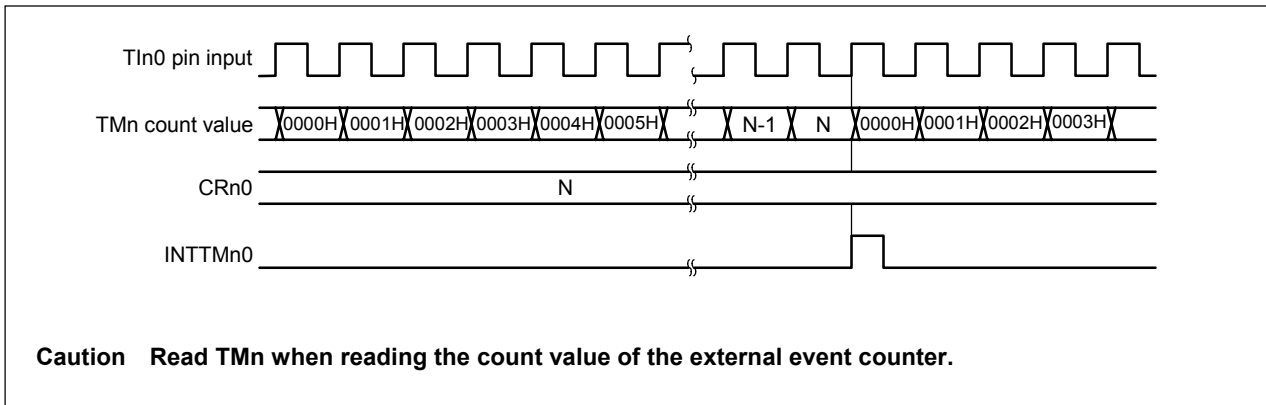


Figure 8-18. Timing of External Event Counter Operation (with Rising Edge Specified)



8.2.5 Operation as square-wave output

TMn can be used to output a square wave with any frequency at an interval specified by the count value preset to 16-bit capture/compare register n0 (CRn0).

By setting bits 0 (TOEn) and 1 (TOCn1) of 16-bit timer output control register n (TOCn) to 1, the output status of the TOn pin is reversed at an interval specified by the count value preset to CRn1. In this way, a square wave of any frequency can be output.

Figure 8-19. Control Register Settings in Square-Wave Output Mode

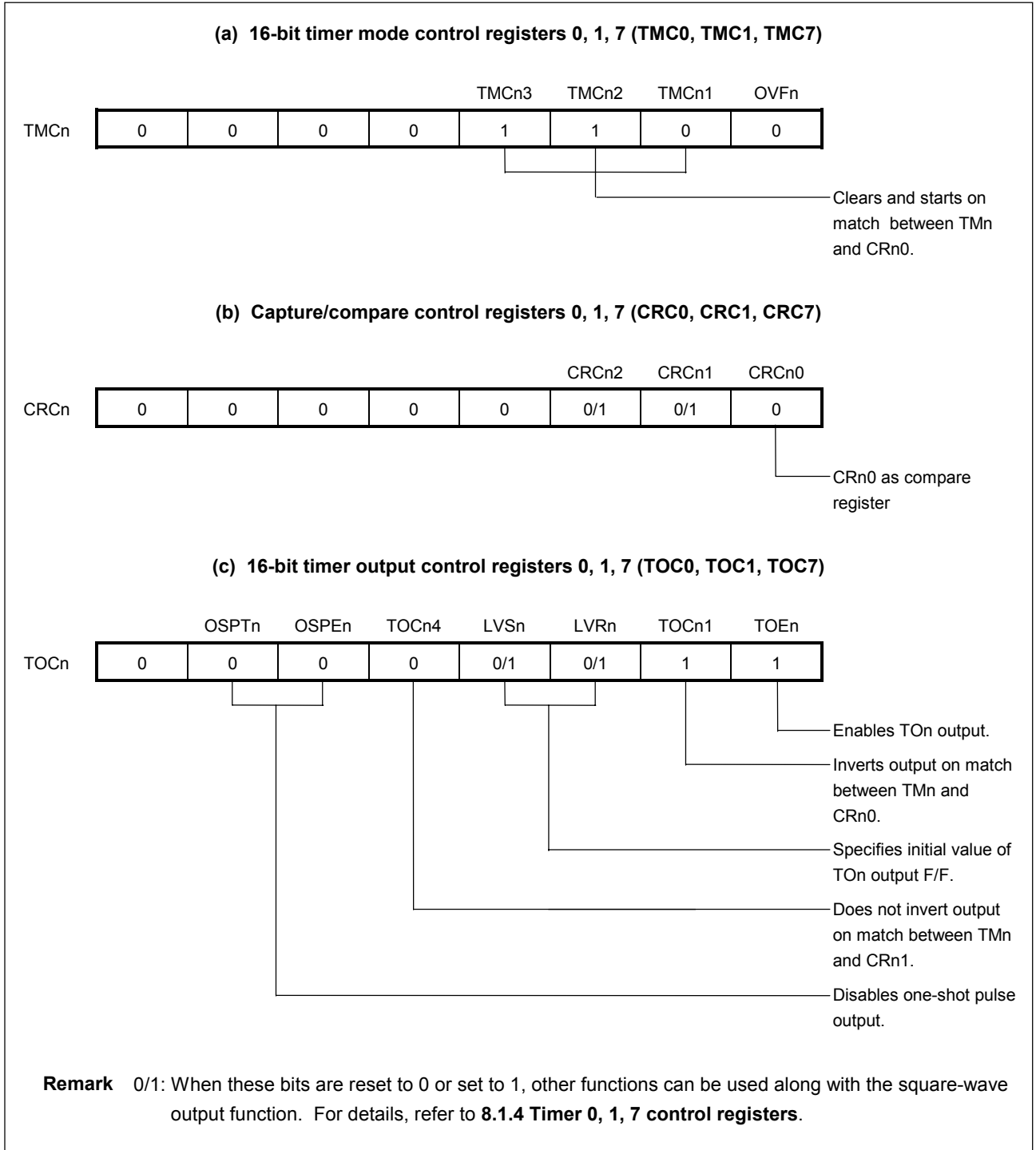
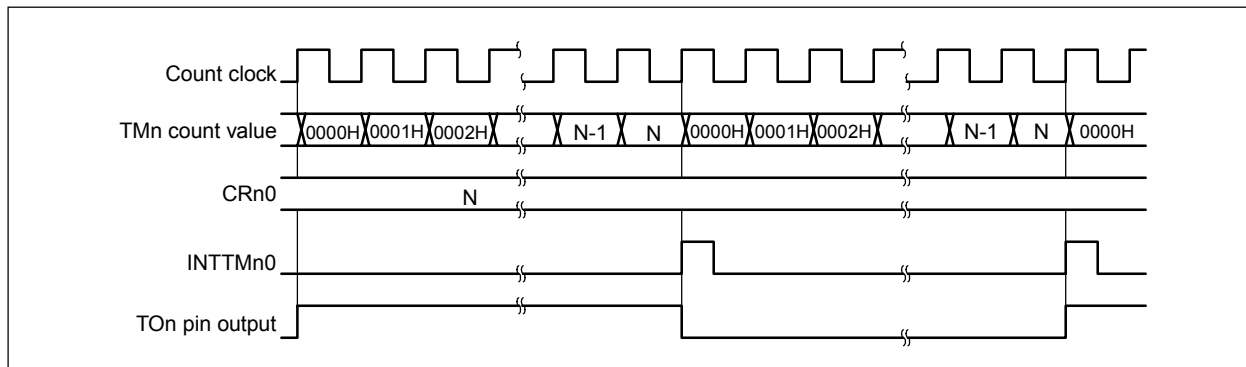


Figure 8-20. Timing of Square-Wave Output Operation



8.2.6 Operation as one-shot pulse output

TMn can output a one-shot pulse in synchronization with a software trigger and an external trigger (TIn0 pin input).

(1) One-shot pulse output with software trigger

A one-shot pulse can be output from the TOn pin by setting 16-bit timer mode control register n (TMCn), capture/compare control register n (CRCn), and 16-bit timer output control register n (TOCn) as shown in Figure 8-21, and by setting bit 6 (OSPTn) of TOCn by software.

By setting OSPTn to 1, the 16-bit timer/event counter is cleared and started, and its output is asserted at the count value (N) preset to 16-bit capture/compare register n1 (CRn1). After that, the output is deasserted at the count value (M) preset to 16-bit capture/compare register n0 (CRn0)^{Note}.

Even after a one-shot pulse has been output, TMn continues its operation. To stop TMn, TMCn must be reset to 00H.

Note This is an example when $N < M$. When $N > M$, the output becomes active at the CRn0 value and inactive at the CRn1 value.

Caution Do not set OSPTn to 1 while a one-shot pulse is being output. To output a one-shot pulse again, wait until the current one-shot pulse output ends.

Figure 8-21. Control Register Settings for One-Shot Pulse Output with Software Trigger

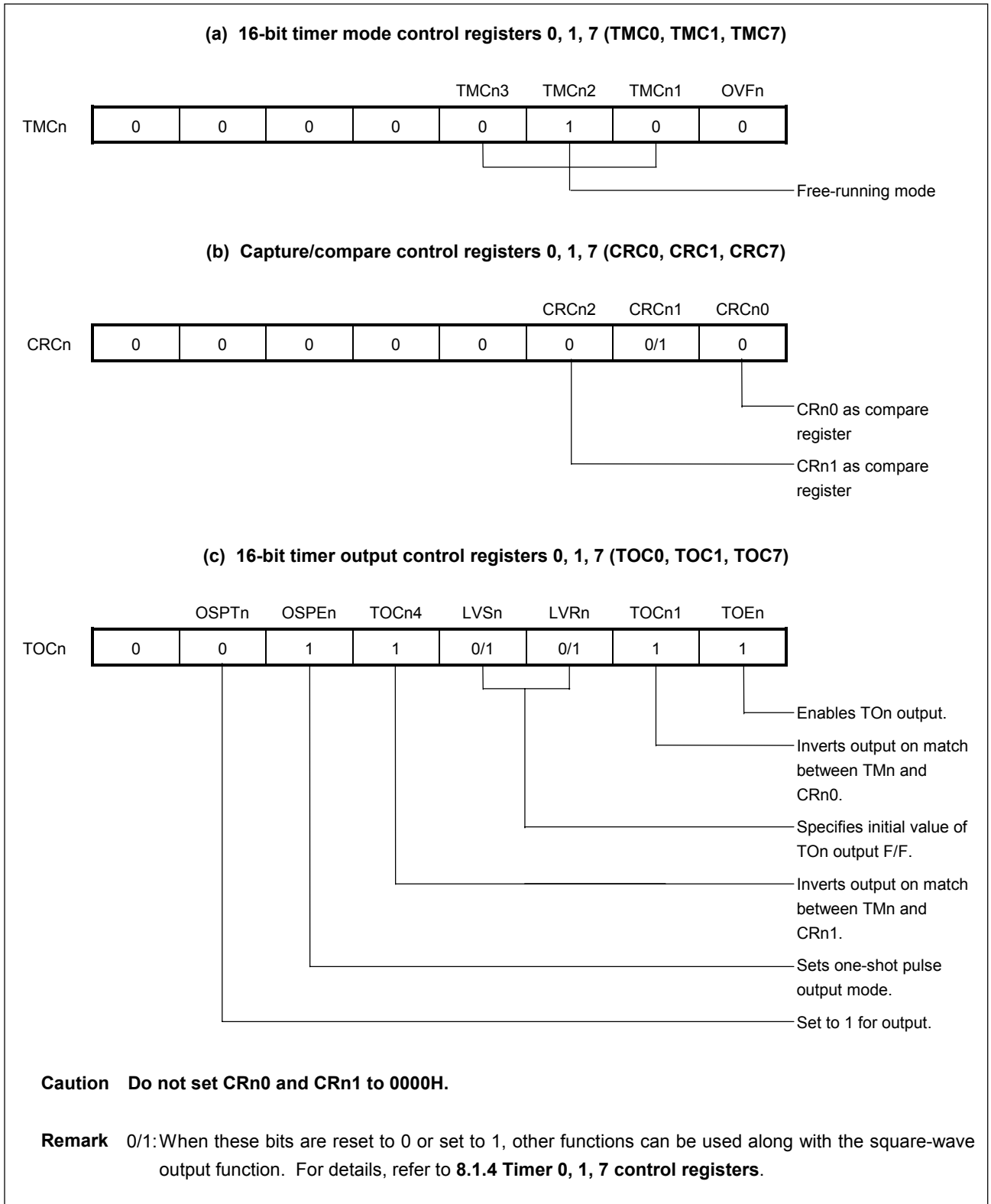
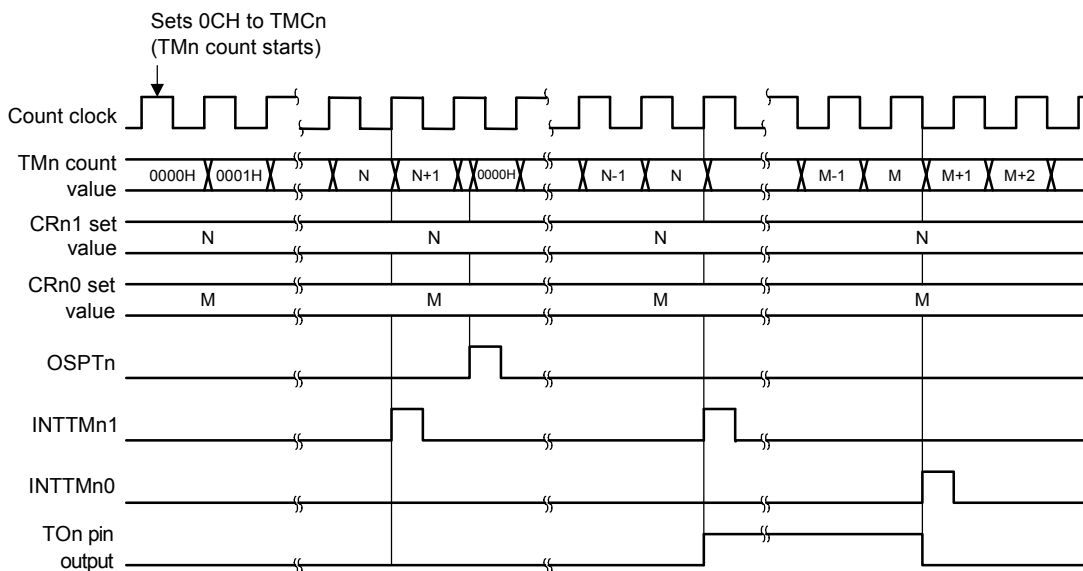


Figure 8-22. Timing of One-Shot Pulse Output Operation with Software Trigger



Caution 16-bit timer register n starts operating as soon as a value other than 0, 0 (operation stop mode) is set to TMCn2 and TMCn3.

Remark $N < M$

(2) One-shot pulse output with external trigger

A one-shot pulse can be output from the TOn pin by setting 16-bit timer mode control register n (TMCn), capture/compare control register n (CRCn), and 16-bit timer output control register n (TOCn) as shown in Figure 8-23, and by using the valid edge of the Tin0 pin as an external trigger.

The valid edge of the Tin0 pin is specified by bits 4 and 5 (ESn00 and ESn01) of prescaler mode register n0 (PRMn0). The rising, falling, or both the rising and falling edges can be specified.

When the valid edge of the Tin0 pin is detected, the 16-bit timer/event counter is cleared and started, and the output is asserted at the count value (N) preset to 16-bit capture/compare register n1 (CRn1).

After that, the output is deasserted at the count value (M) preset to 16-bit capture/compare register n0 (CRn0)^{Note}.

Note This is an example when $N < M$. When $N > M$, the output becomes active at the CRn0 value and inactive at the CRn1 value.

Caution Even if the external trigger is generated again while a one-shot pulse is being output, it is ignored.

Figure 8-23. Control Register Settings for One-Shot Pulse Output with External Trigger

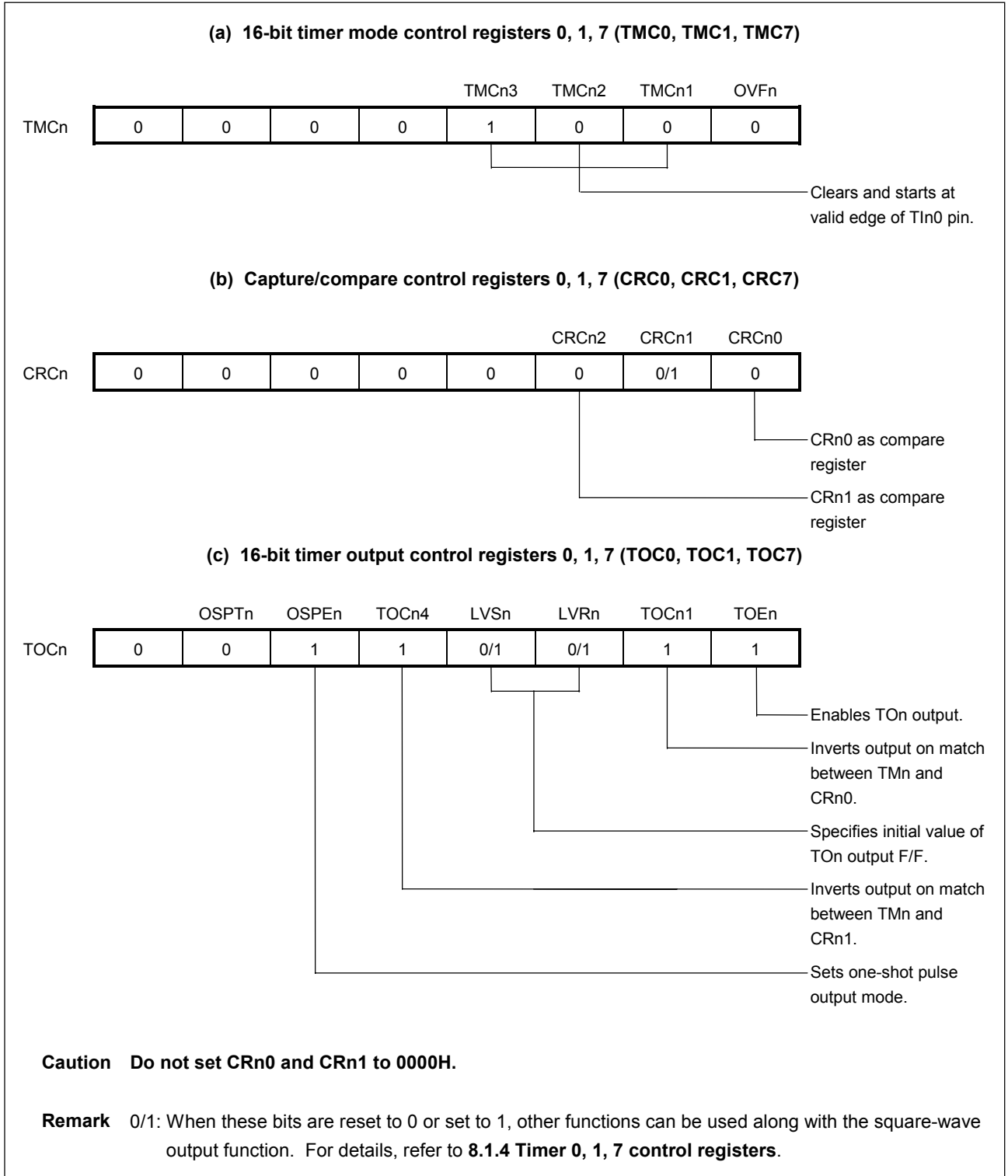
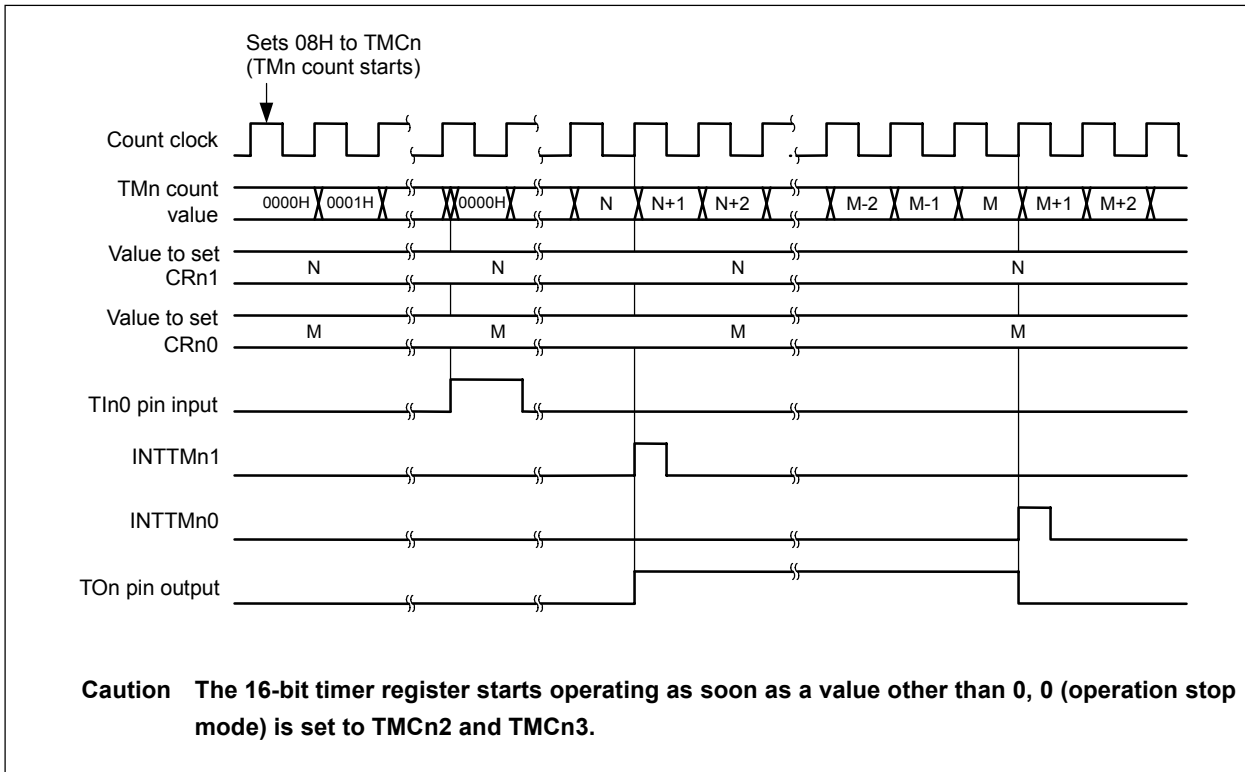


Figure 8-24. Timing of One-Shot Pulse Output Operation with External Trigger (with Rising Edge Specified)

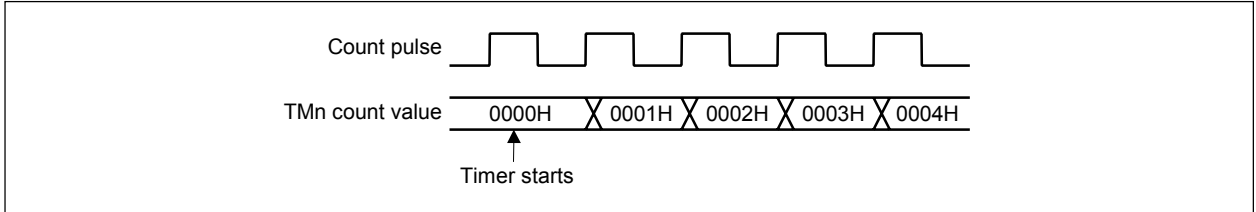


8.2.7 Cautions

(1) Error on starting timer

An error of up to 1 clock occurs before the match signal is generated after the timer has been started. This is because 16-bit timer register n (TMn) is started asynchronously to the count pulse.

Figure 8-25. Start Timing of 16-Bit Timer Register n



(2) 16-bit capture/compare register setting (in the clear & start mode entered on match between TMn and CRn0)

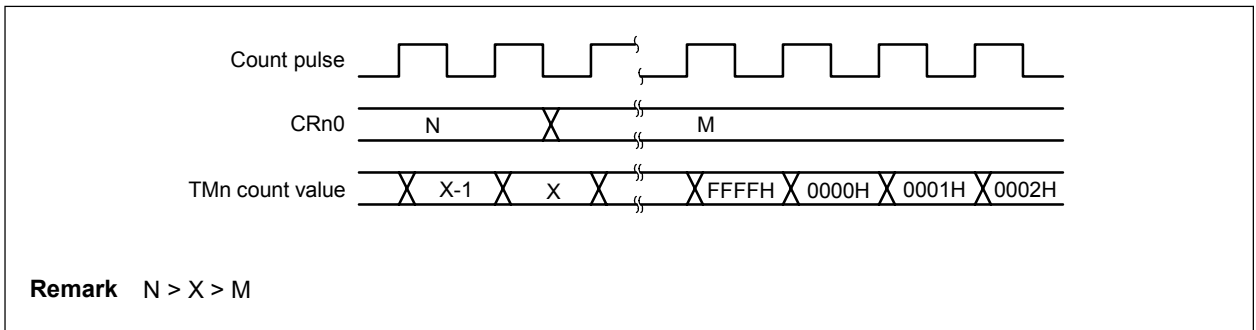
Set 16-bit capture/compare registers n0, n1 (CRn0, CRn1) to a value other than 0000H (a 1-pulse count operation is disabled when these registers are used as event counters).

(3) Setting compare register during timer count operation

If the value to which the current value of 16-bit capture/compare register n0 (CRn0) has been changed is less than the value of 16-bit timer register n (TMn), TMn continues counting, overflows, and starts counting again from 0.

If the new value of CRn0 (M) is less than the old value (N), the timer must be restarted after the value of CRn0 has been changed.

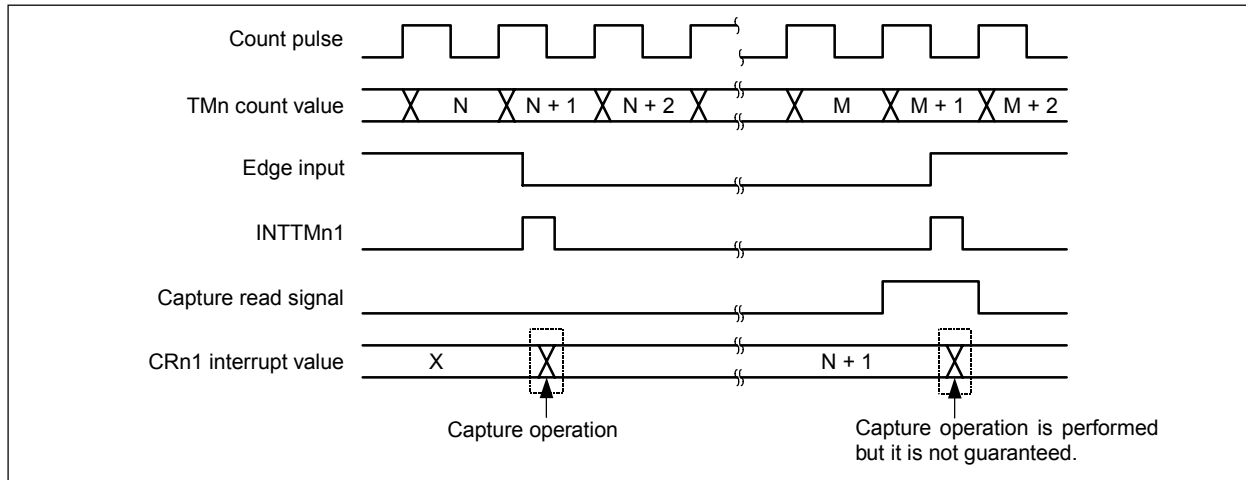
Figure 8-26. Timing After Changing Compare Register During Timer Count Operation



(4) Data hold timing of capture register

If the valid edge is input to the TIn0 pin while 16-bit capture/compare register n1 (CRn1) is read, CRn1 performs a capture operation, but this capture value is not guaranteed. However, the interrupt request signal (INTTMn1) is set as a result of detection of the valid edge.

★ **Figure 8-27. Data Hold Timing of Capture Register**

**(5) Setting valid edge**

Before setting the valid edge of the TIn0 pin, stop the timer operation by resetting bits 2 and 3 (TMCn2 and TMCn3) of 16-bit timer mode control register n to 0, 0. Set the valid edge by using bits 4 and 5 (ESn00 and ESn01) of prescaler mode register n0 (PRMn0).

(6) Re-triggering one-shot pulse**(a) One-shot pulse output via software**

When a one-shot pulse is output, do not set OSPTn to 1. Do not output the one-shot pulse again until the current one-shot pulse output ends.

(b) One-shot pulse output via external trigger

Even if the external trigger is generated again while a one-shot pulse is being output, it is ignored.

★ **(c) One-shot pulse output function**

When using a software trigger for one-shot pulse output of timers 0, 1, and 7, the level of the TIn0 pin or its alternate-function pin cannot be changed.

The reason for this is that the timer is inadvertently cleared and started at the level of the TIn0 pin or its alternate-function pin and pulses are output at an unintended timing because the external trigger is valid.

(7) Operation of OVF_n flag**(a) OVF_n flag set**

The OVF_n flag is set to 1 in the following case:

Selection of mode in which TM₀ is cleared and started on match between TM_n and CR_{n0}, or mode in which TM₀ is cleared and started at the TIn₀ valid edge, or free-running mode

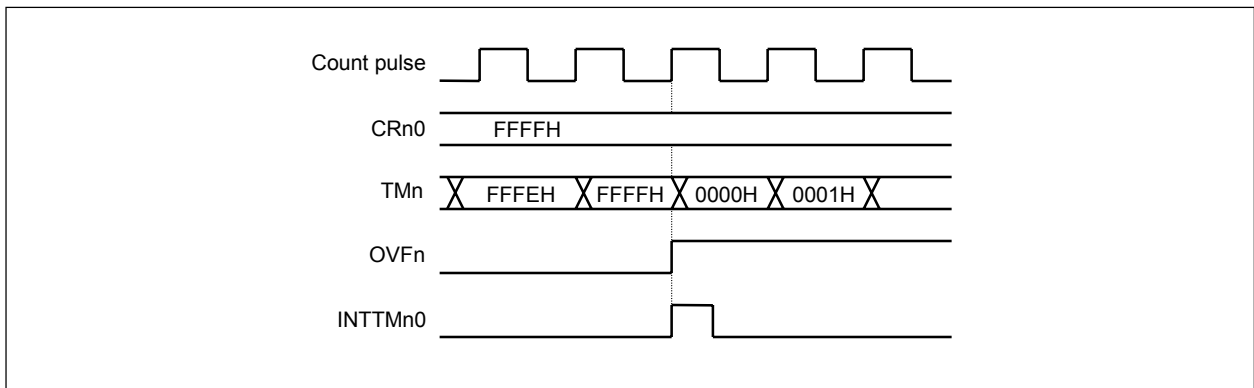
↓

CR_{n0} set to FFFFH

↓

When TM_n counts up from FFFFH to 0000H

Figure 8-28. Operation Timing of OVF_n Flag

**(b) Clear OVF_n flag**

Even if the OVF_n flag is cleared before the next count clock is counted (before TM_n becomes 0001H) after TM_n has overflowed, the OVF_n flag is set again and the clear becomes invalid.

(8) Conflicting operations**(a) If the read period and capture trigger input conflict**

When 16-bit capture/compare registers n0 and n1 (CR_{n0}, CR_{n1}) are used as capture registers, if the read period and capture trigger input conflict, the capture trigger has priority. The read data of CR_{n0} and CR_{n1} is undefined.

(b) If the match timings of the write period and TM_n conflict

When 16-bit capture/compare registers n0 and n1 (CR_{n0}, CR_{n1}) are used as capture registers, because match detection cannot be performed correctly if the match timings of the write period and 16-bit timer register n (TM_n) conflict, do not write to CR_{n0} and CR_{n1} close to the match timing.

(9) Timer operation**(a) CRn1 capture**

Even if 16-bit timer register n (TMn) is read, a capture to 16-bit capture/compare register n1 (CRn1) is not performed.

(b) Acknowledgement of TIn0 and TIn1 pins

When the timer is stopped, input signals to the TIn0 and TIn1 pins are not acknowledged, regardless of the CPU operation.

(c) One-shot pulse output

The one-shot pulse output operates correctly only in free-running mode or in clear & start mode set at the valid edge of the TIn0 pin. A one-shot pulse cannot be output in the clear & start mode set on a match of TMn and CRn0 because an overflow does not occur.

(10) Capture operation**(a) If the valid edge of TIn0 is specified for the count clock**

When the valid edge of TIn0 is specified for the count clock, the capture register with TIn0 specified as a trigger will not operate correctly.

(b) If both rising and falling edges are selected as the valid edge of TIn0

If both rising and falling edges are selected as the valid edge of TIn0, a capture operation is not performed.

(c) To capture the signals correctly from TIn0 and TIn1

The capture trigger needs a pulse longer than twice the count clock selected by prescaler mode registers n0 and n1 (PRMn0, PRMn1) in order to correctly capture the signals from TIn1 and TIn0.

(d) Interrupt request input

Although a capture operation is performed at the falling edge of the count clock, interrupt request inputs (INTTMn0, INTTMn1) are generated at the falling edge of the next count clock.

(11) Compare operation**(a) When rewriting CRn0 and CRn1 during timer operation**

When rewriting 16-bit timer capture/compare registers n0 and n1 (CRn0, CRn1), if the value is close to or larger than the timer value, the match interrupt request generation or clear operation may not be performed correctly.

(b) When CRn0 and CRn1 are set to compare mode

When CRn0 and CRn1 are set to compare mode, they do not perform a capture operation even if a capture trigger is input.

(12) Edge detection**(a) When the TIn0 or TIn1 pin is high level immediately after a system reset**

When the TIn0 or TIn1 pin is high level immediately after a system reset, if the valid edge of the TIn0 or TIn1 pin is specified as the rising edge or both rising and falling edges, and the operation of 16-bit timer/counter n (TMn) is then enabled, the rising edge will be detected immediately. Care is therefore needed when the TIn0 or TIn1 pin is pulled up. However, when operation is enabled after being stopped, the rising or falling edge is not detected.

(b) Sampling clock for noise elimination

The sampling clock for noise elimination differs depending on whether the TIn0 valid edge is used as a count clock or a capture trigger. The former is sampled by $fx/2$, and the latter is sampled by the count clock selected using prescaler mode registers n0 or n1 (PRMn0, PRMn1). Detecting the valid edge can eliminate short pulse width noise because a capture operation is performed only after the valid edge is sampled and a valid level is detected twice.

8.3 16-Bit Timer (TM2 to TM6)

Remark n = 2 to 6 in section 8.3.

8.3.1 Functions

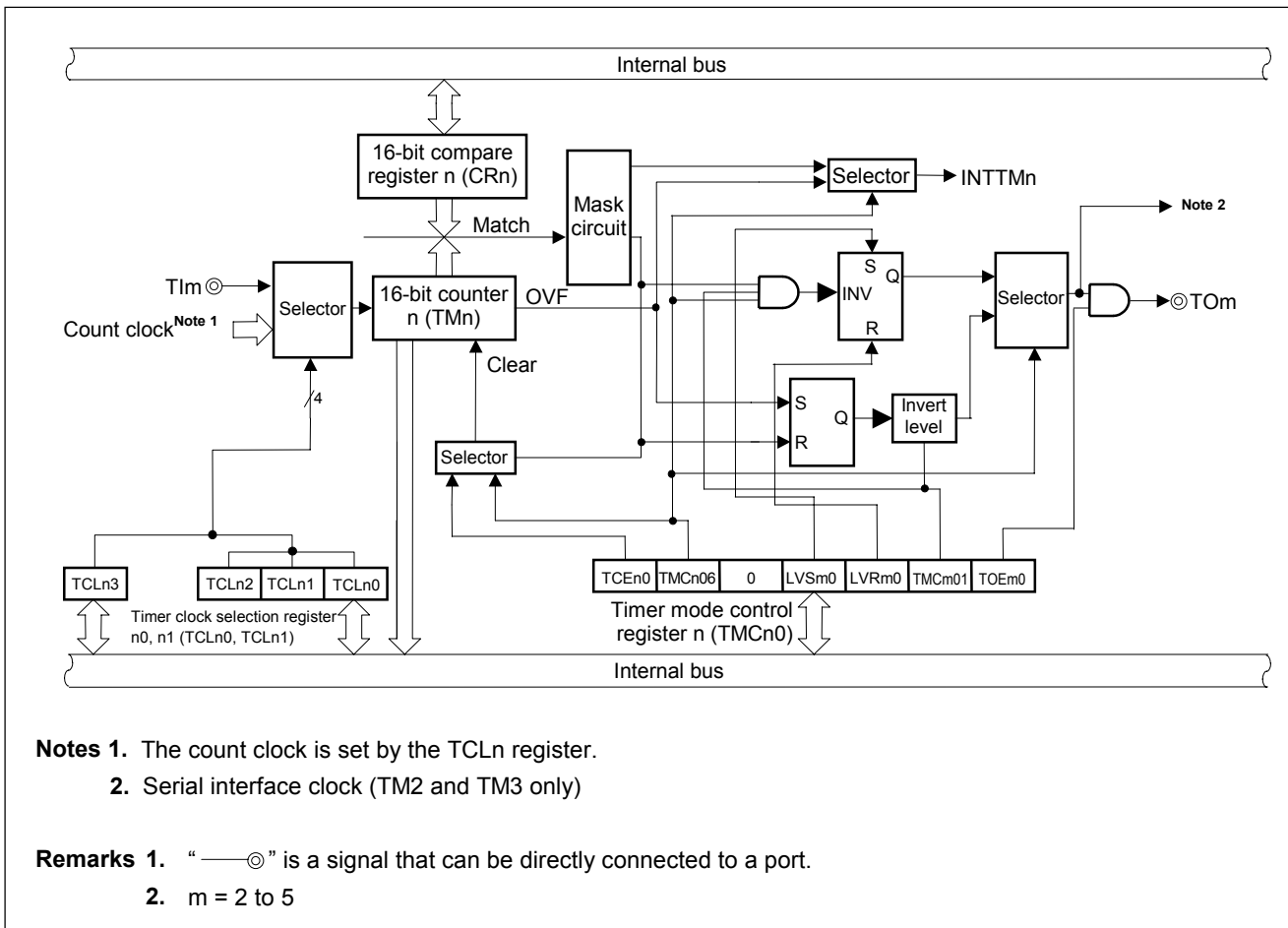
TM2 to TM5 have the following functions.

- PWM output with 16-bit resolution
- Interval timer with 16-bit resolution
- External event counter with 16-bit resolution
- Square-wave output with 16-bit resolution

TM6 has the following function.

- Interval timer with 16-bit resolution

Figure 8-29. Block Diagram of TM2 to TM6



8.3.2 Configuration

Timer n includes from the following hardware.

Table 8-5. Configuration of Timers 2 to 6

Item	Configuration
Timer registers	16-bit counters 2 to 6 (TM2 to TM6)
Registers	16-bit compare registers 2 to 6 (CR2 to CR6)
Timer outputs	TO2 to TO5
Control registers	Timer clock selection registers 20 to 60 and 21 to 61 (TCL20 to TCL60 and TCL21 to TCL61) 8-bit timer mode control registers 20 to 60 (TMC20 to TMC60)

(1) 16-bit counters 2 to 6 (TM2 to TM6)

TMn is a 16-bit read-only register that counts the count pulses.

The counter is incremented in synchronization with the rising edge of the count clock.

When the count is read out during operation, the count clock input temporarily stops and the count is read at that time. In the following cases, the count becomes 0000H.

- (1) $\overline{\text{RESET}}$ is input.
- (2) TCEn is cleared.
- (3) TMn and CRn match in the clear and start mode that occurs when TMn and CRn0 match.

(2) 16-bit compare registers 2 to 6 (CR2 to CR6)

The value set in CRn is always compared to the count in 16-bit counter n (TMn). If the two values match, an interrupt request (INTTMn) is generated (except in the PWM mode).

8.3.3 Timer n control register

The following two types of registers control timer n.

- Timer clock selection registers n0, n1 (TCLn0, TCLn1)
- 16-bit timer mode control register n (TMCn)

(1) Timer clock selection registers 20 to 60 and 21 to 61 (TCL20 to TCL60 and TCL21 to TCL61)

These registers set the count clock of timer n.

TCLn0 and TCLn1 are set by an 8-bit memory manipulation instruction.

RESET input sets these registers to 00H.

After reset: 00H R/W

Address: FFFFF244H, FFFFF0E4H

	7	6	5	4	3	2	1	0
TCLm0	0	0	0	0	0	TCLm2	TCLm1	TCLm0

(m = 2, 3)

After reset: 00H R/W

Address: FFFFF24EH, FFFFF2EEH

	7	6	5	4	3	2	1	0
TCLm1	0	0	0	0	0	0	0	TCLm3

(m = 2, 3)

TCLm3	TCLm2	TCLm1	TCLm0	Count clock selection		
				Count clock	f _{xx}	
					16 MHz	8 MHz
0	0	0	0	TIm falling edge	–	–
0	0	0	1	TIm rising edge	–	–
0	0	1	0	f _{xx} /4	250 ns	500 ns
0	0	1	1	f _{xx} /8	500 ns	1 μs
0	1	0	0	f _{xx} /16	1 μs	2 μs
0	1	0	1	f _{xx} /32	2 μs	4 μs
0	1	1	0	f _{xx} /128	8 μs	16 μs
0	1	1	1	f _{xx} /512	32 μs	64 μs
1	0	0	0	Setting prohibited	–	–
1	0	0	1	Setting prohibited	–	–
1	0	1	0	f _{xx} /64	4 μs	8 μs
1	0	1	1	f _{xx} /256	16 μs	32 μs
1	1	0	0	Setting prohibited	–	–
1	1	0	1	Setting prohibited	–	–
1	1	1	0	Setting prohibited	–	–
1	1	1	1	Setting prohibited	–	–

Cautions 1. When TCLm0 and TCLm1 are overwritten by different data, write after temporarily stopping the timer.

2. Always set bits 3 to 7 to in TCLm0 to 0, and bits 1 to 7 in TCLm1 to 0.

After reset: 00H R/W

Address: FFFFF264H, FFFFF334H

	7	6	5	4	3	2	1	0
TCLm0	0	0	0	0	0	TCLm2	TCLm1	TCLm0

(m = 4, 5)

After reset: 00H R/W

Address: FFFFF26EH, FFFFF33EH

	7	6	5	4	3	2	1	0
TCLm1	0	0	0	0	0	0	0	TCLm3

(m = 4, 5)

TCLm3	TCLm2	TCLm1	TCLm0	Count clock selection		
				Count clock	f _{xx}	
					16 MHz	8 MHz
0	0	0	0	T1m falling edge	–	–
0	0	0	1	T1m rising edge	–	–
0	0	1	0	f _{xx} /4	250 ns	500 ns
0	0	1	1	f _{xx} /8	500 ns	1 μs
0	1	0	0	f _{xx} /16	1 μs	2 μs
0	1	0	1	f _{xx} /32	2 μs	4 μs
0	1	1	0	f _{xx} /128	8 μs	16 μs
0	1	1	1	f _{XT} (subsystem clock)	30.5 μs	30.5 μs
1	0	0	0	Setting prohibited	–	–
1	0	0	1	Setting prohibited	–	–
1	0	1	0	f _{xx} /64	4 μs	8 μs
1	0	1	1	f _{xx} /256	16 μs	32 μs
1	1	0	0	Setting prohibited	–	–
1	1	0	1	Setting prohibited	–	–
1	1	1	0	Setting prohibited	–	–
1	1	1	1	Setting prohibited	–	–

Cautions 1. When TCLm0 and TCLm1 are overwritten by different data, write after temporarily stopping the timer.

2. Always set bits 3 to 7 of TCLm0 and bits 1 to 7 of TCLm1 to 0.

After reset: 00H R/W

Address: FFFFF284H

	7	6	5	4	3	2	1	0
TCL60	0	0	0	0	0	TCL62	TCL61	TCL60

After reset: 00H R/W

Address: FFFFF28EH

	7	6	5	4	3	2	1	0
TCL61	0	0	0	0	0	0	0	TCL63

TCL63	TCL62	TCL61	TCL60	Count clock selection		
				Count clock	f _{xx}	
					16 MHz	8 MHz
0	0	0	0	Setting prohibited	–	–
0	0	0	1	Setting prohibited	–	–
0	0	1	0	f _{xx} /4	250 ns	500 ns
0	0	1	1	f _{xx} /8	500 ns	1 μs
0	1	0	0	f _{xx} /16	1 μs	2 μs
0	1	0	1	f _{xx} /32	2 μs	4 μs
0	1	1	0	f _{xx} /64	4 μs	8 μs
0	1	1	1	f _{xx} /128	8 μs	16 μs
1	0	0	0	Setting prohibited	–	–
1	0	0	1	Setting prohibited	–	–
1	0	1	0	f _{xx} /256	16 μs	32 μs
1	0	1	1	f _{xx} /512	32 μs	64 μs
1	1	0	0	Setting prohibited	–	–
1	1	0	1	Setting prohibited	–	–
1	1	1	0	Setting prohibited	–	–
1	1	1	1	TM0 overflow signal	–	–

- Cautions**
1. When TCL60 and TCL61 are overwritten by different data, write after temporarily stopping the timer.
 2. Always set bits 3 to 7 of TCL60 and bits 1 to 7 of TCL61 to 0.

(2) 16-bit timer mode control registers 20 to 60 (TMC20 to TMC60)

The TMCn0 register makes the following five settings.

- (1) Controls the counting by 16-bit counter n (TMn)
- (2) Selects the operation mode of 16-bit counter n (TMn)
- (3) Sets the state of the timer output flip-flop
- (4) Controls the timer flip-flop or selects the active level in the PWM (free-running) mode
- (5) Controls timer output

TMCn0 is set by an 8- or 1-bit memory manipulation instruction.

RESET input sets these registers to 04H (although the state of hardware is initialized to 04H, 00H is read when reading).

After reset: 04H R/W Address: TMC20 FFFF246H TMC50 FFFF336H
 TMC30 FFFF0E6H TMC60 FFFF286H
 TMC40 FFFF266H

	7	6	5	4	3	2	1	0
TMCn0	TCEn0	TMCn06	0	0	LVS _m 0	LVR _m 0	TMC _m 01	TOE _m 0

(m = 2 to 5)

TCEn0	TM _n count operation control
0	Counting is disabled after the counter is cleared to 0 (prescaler disabled)
1	Start count operation

TMCn06	TM _n operating mode selection
0	Clear & start mode when TM _n and CR _n match
1	PWM (free-running) mode

LVS _m 0	LVR _m 0	Setting state of timer output flip-flop
0	0	Not change
0	1	Reset timer output flip-flop to 0
1	0	Set timer output flip-flop to 1
1	1	Setting prohibited

TMC _m 01	Other than PWM (free-running) mode (TMCn06 = 0)	PWM (free-running) mode (TMCn06 = 1)
	Controls timer F/F	
0	Inversion operation disabled	Active high
1	Inversion operation enabled	Active low

TOE _m 0	Timer output control
0	Output disabled (port mode)
1	Output enabled

- Cautions**
1. When using as the timer output pin (TOM), set the port value to 0 (port mode output). A logical sum (ORed) value of the timer output value is output.
 2. Since TOM and TIM are the same alternate-function pin, only one function can be used.

- Remarks**
1. In the PWM mode, the PWM output is set to the inactive level by TCE_m0 = 0.
 2. If LVS_m0 and LVR_m0 are read after setting data, 0 is read.

8.4 16-Bit Timer (TM2 to TM6) Operation

Remark n = 2 to 6 and m = 2 to 5 in section 8.4.

8.4.1 Operation as interval timer

The timer operates as an interval timer that repeatedly generates interrupts at the interval specified by the count value preset to 16-bit compare register n (CRn).

If the count value of 16-bit counter n (TMn) matches the set value of CRn, the value of TMn is cleared to 0 and the timer continues counting, and an interrupt request signal (INTTMn) is generated.

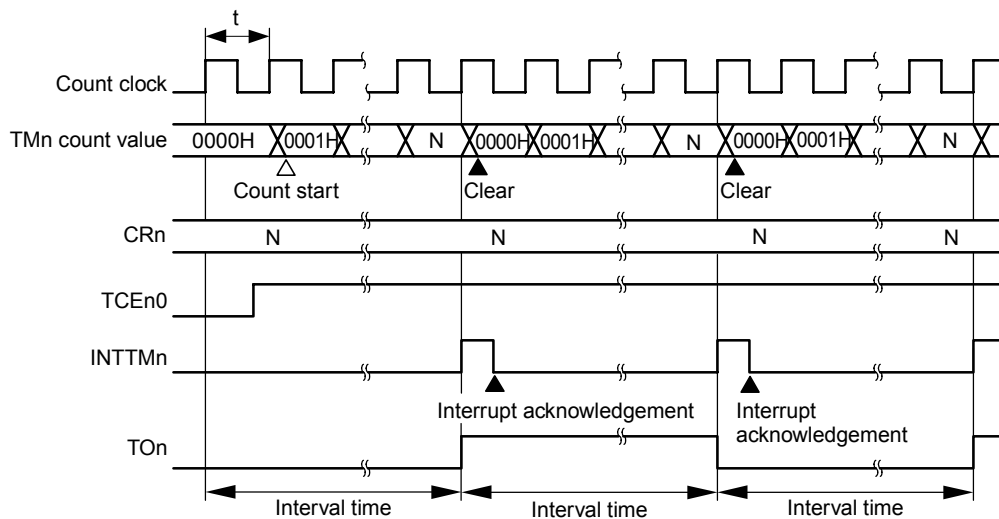
The TMn count clock can be selected by bits 0 to 2 (TCLn0 to TCLn2) of timer clock selection register n0 (TCLn0) and by bit 0 (TCLn3) of timer clock selection register n1 (TCLn1).

Setting method

- (1) Set each register.
 - TCLn0, TCLn1: Selects the count clock.
 - CRn: Compare value
 - TMCn0: Selects the clear and start mode when TMn and CRn match.
(TMCn0 = 0000xxx0B, x = don't care)
- (2) When TCEn0 = 1 is set, counting starts.
- (3) When the values of TMn and CRn match, INTTMn is generated (TMn is cleared to 0000H).
- (4) INTTMn is then repeatedly generated during the same interval. When counting stops, set TCEn0 = 0.

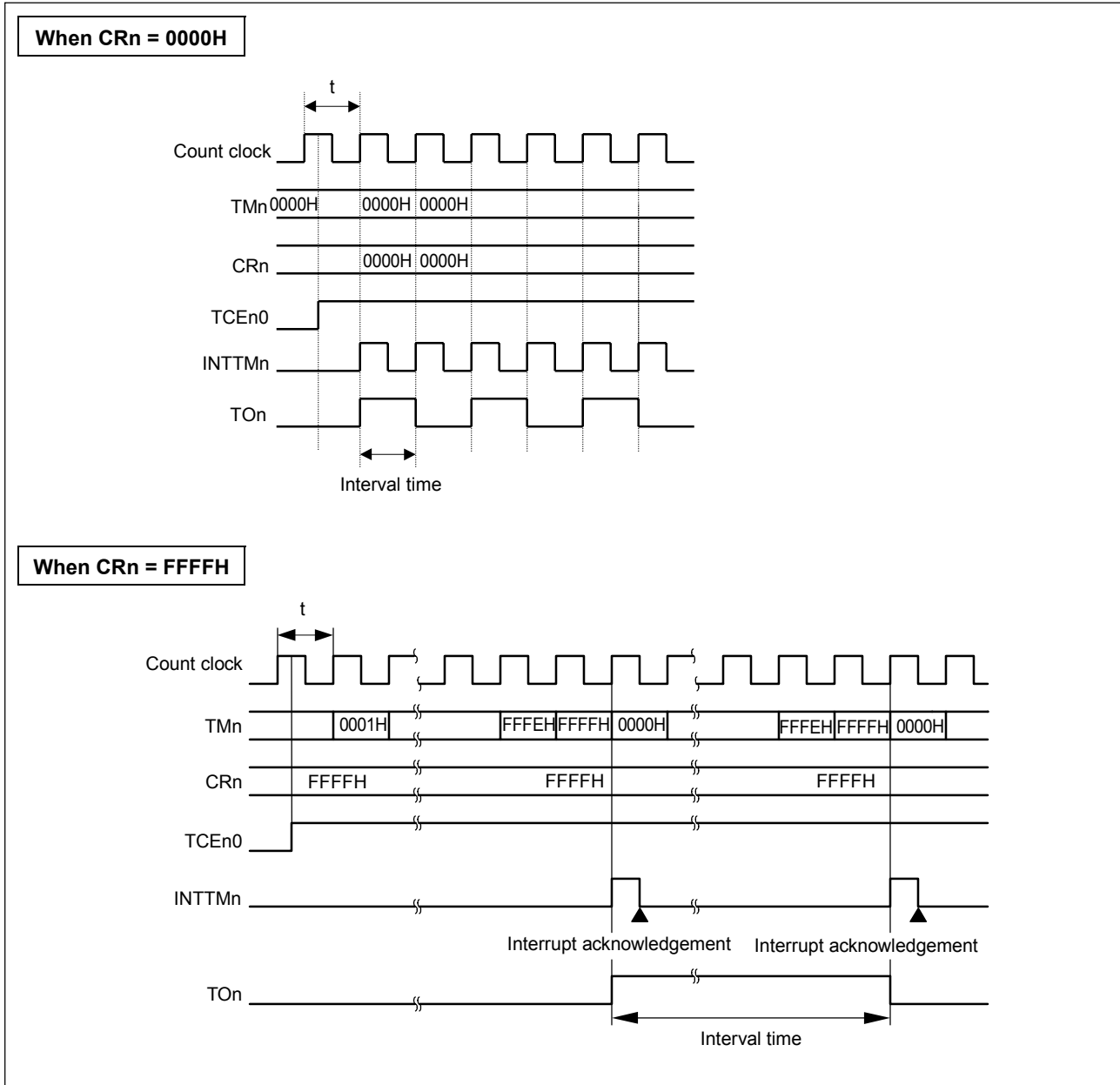
Figure 8-30. Timing of Interval Timer Operation (1/2)

Basic operation



Remark Interval time = $(N + 1) \times t$; N = 0000H to FFFFH

Figure 8-30. Timing of Interval Timer Operation (2/2)



8.4.2 Operation as external event counter

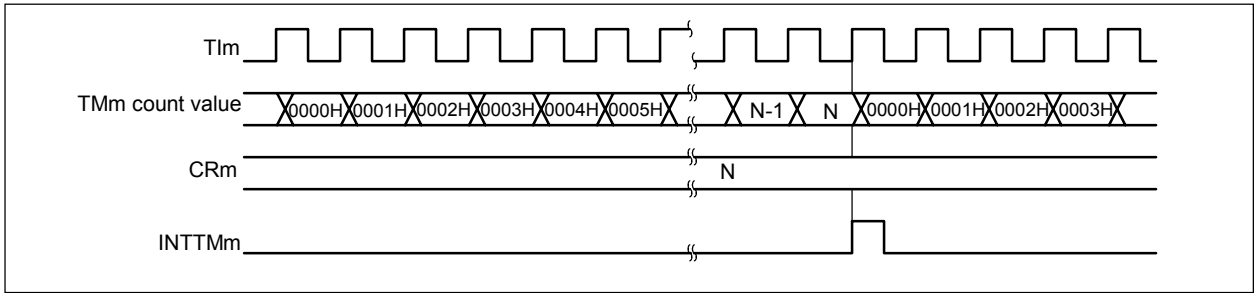
The external event counter counts the number of external clock pulses that are input to TIm.

Each time a valid edge specified by timer clock selection register m0, m1 (TCLm0, TCLm1) is input, TMm is incremented. The edge setting can be selected to be either a rising or falling edge.

If the total of TMm and the value of 16-bit compare register m (CRm) match, TMm is cleared to 0 and an interrupt request signal (INTTMm) is generated.

INTTMm is generated each time the TMm value matches the CRm value.

Figure 8-31. Timing of External Event Counter Operation (with Rising Edge specified)



8.4.3 Operation as square-wave output

A square-wave with any frequency is output at the interval preset to 16-bit compare register m (CRm).

By setting bit 0 (TOEm0) of 16-bit timer mode control register m0 (TMCm0) to 1, the output state of TOm is inverted at an interval specified by the count value preset to CRm. Therefore, a square-wave of any frequency (duty factor = 50%) can be output.

Setting method

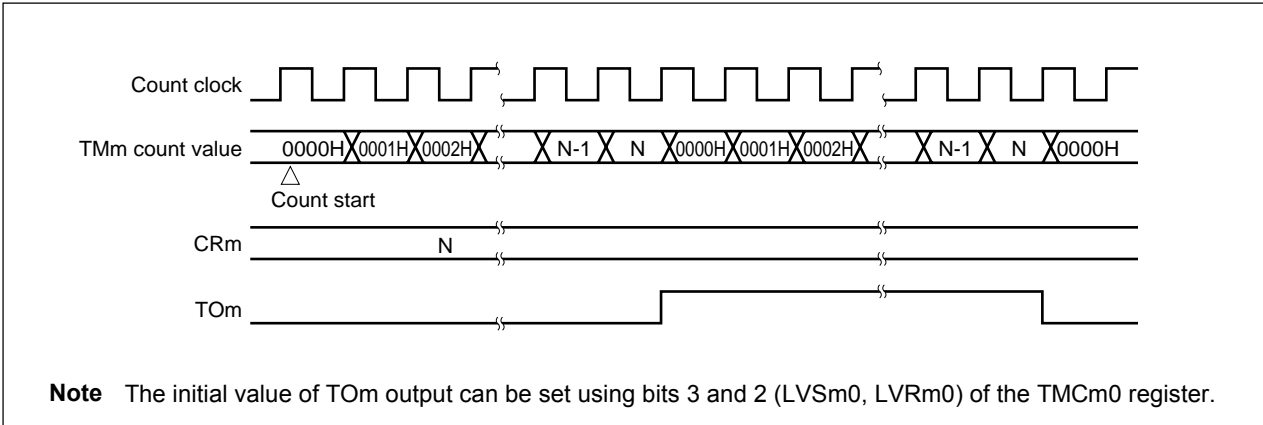
- (1) Set the registers.
 - Sets the port latch and port mode register to 0
 - TCLm0, TCLm1: Selects the count clock
 - CRm: Compare value
 - TMCm0: Clear and start mode when TMm and CRm match

LVS _m 0	LVR _m 0	Setting state of timer output flip-flop
1	0	High-level output
0	1	Low-level output

Inversion of timer output flip-flop enabled
 Timer output enabled → TOEm0 = 1

- (2) When TCEm0 = 1 is set, the counter starts operating.
- (3) If the values of TMm and CRm match, the timer output flip-flop inverts. Also, INTTMm is generated and TMm is cleared to 0000H.
- (4) The timer output flip-flop is then inverted at the same interval and a square-wave is output from TOm.

Figure 8-32. Square-Wave Output Operation Timing



8.4.4 Operation as 16-bit PWM output

By setting bit 6 (TMCm6) of 16-bit timer mode control register m0 (TMCm0) to 1, the timer operates as a PWM output.

Pulses with the duty factor determined by the value set in 16-bit compare register m (CRm) are output from T0m.

Set the width of the active level of the PWM pulse to CRm. The active level can be selected by bit 1 (TMCm01) of TMCm0.

The count clock can be selected by bits 0 to 2 (TCLm0 to TCLm2) of timer clock selection register m0 (TCLm0) and by bit 0 (TCLm3) of timer clock selection register m1 (TCLm1).

The PWM output can be enabled and disabled by bit 0 (TOEm0) of TMCm0.

Caution CRm can be rewritten only once in one period while in PWM mode.

(1) Basic operation of the PWM output

Setting method

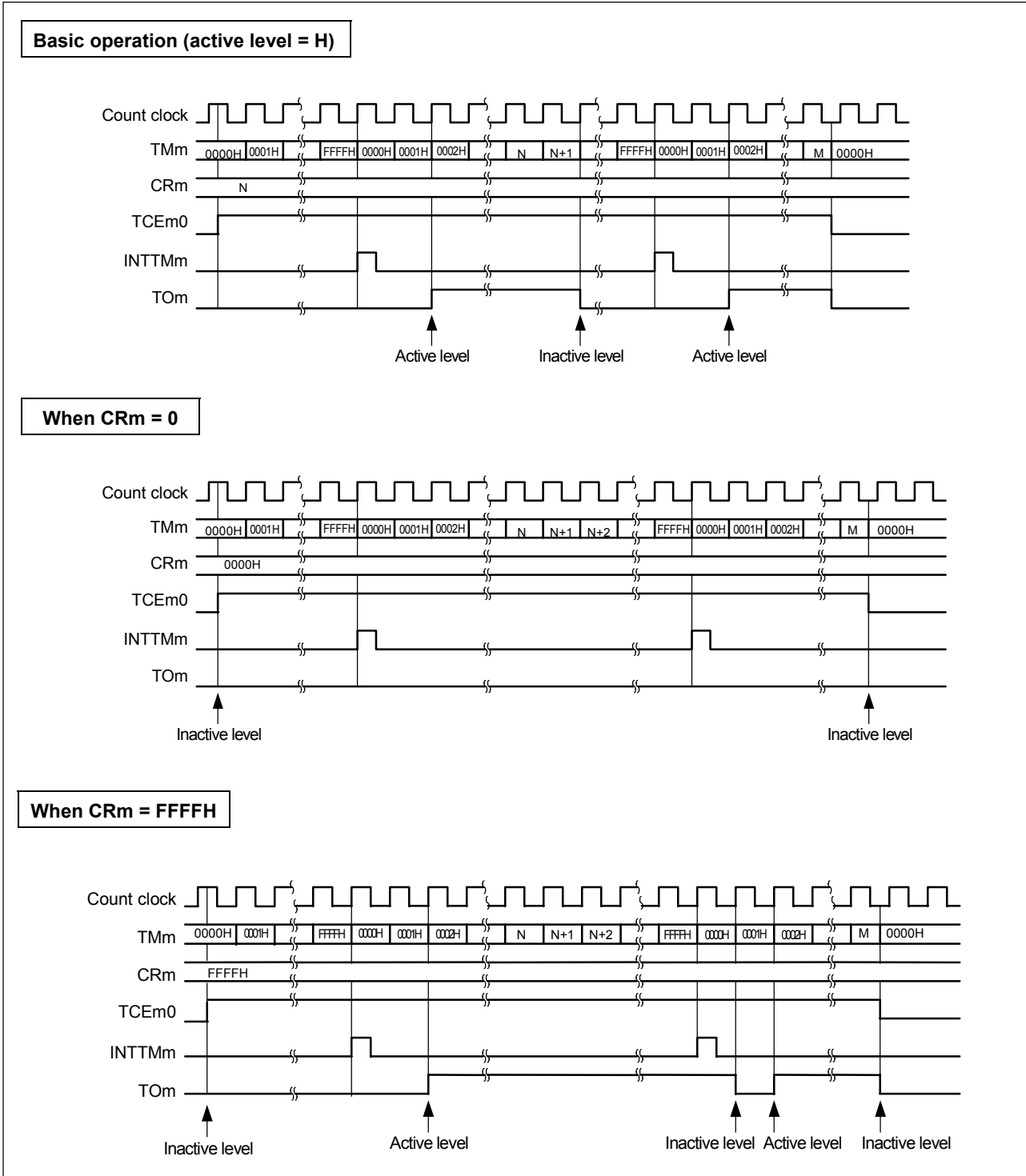
- (1) Set the port latch and port mode register n to 0.
- (2) Set the active level width in 16-bit compare register m (CRm).
- (3) Select the count clock using timer clock selection register m0, m1 (TCLm0, TCLm1).
- (4) Set the active level in bit 1 (TMCm01) of TMCm0.
- (5) If bit 7 (TCEm0) of TMCm0 is set to 1, counting starts. To stop counting, set TCEm0 to 0.

PWM output operation

- (1) When counting starts, the PWM output (output from T0m) outputs an inactive level until an overflow occurs.
- (2) When an overflow occurs, the active level specified in step (1) in the setting method is output. The active level is output until CRm and the count of 8-bit counter m (TMm) match.
- (3) The PWM output after CRm and the count match is the inactive level until an overflow occurs again.
- (4) Steps (2) and (3) repeat until counting stops.
- (5) If counting is stopped by TCEm0 = 0, PWM output goes to the inactive level.

(a) Basic operation of PWM output

Figure 8-33. Timing of PWM Output

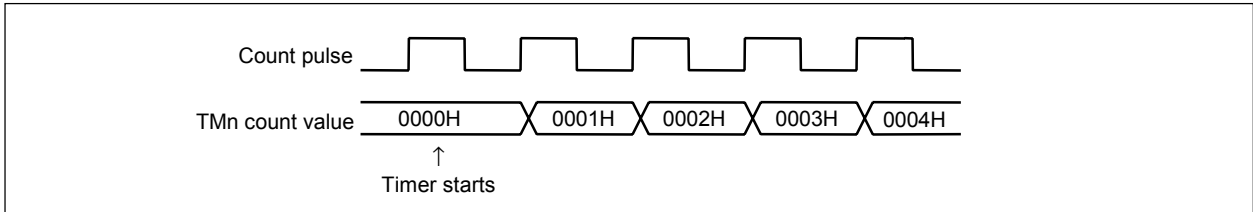


8.4.5 Cautions

(1) Error when the timer starts

An error of up to 1 clock occurs before the match signal is generated after the timer has been started. This is because 16-bit counter n (TMn) is started asynchronously to the count pulse.

Figure 8-34. Start Timing of Timer n



(2) TMn readout during timer operation

Since reading out TMn during operation occurs while the selected clock is temporarily stopped, select a high- or low-level waveform that is longer than the selected clock.

CHAPTER 9 WATCH TIMER FUNCTION

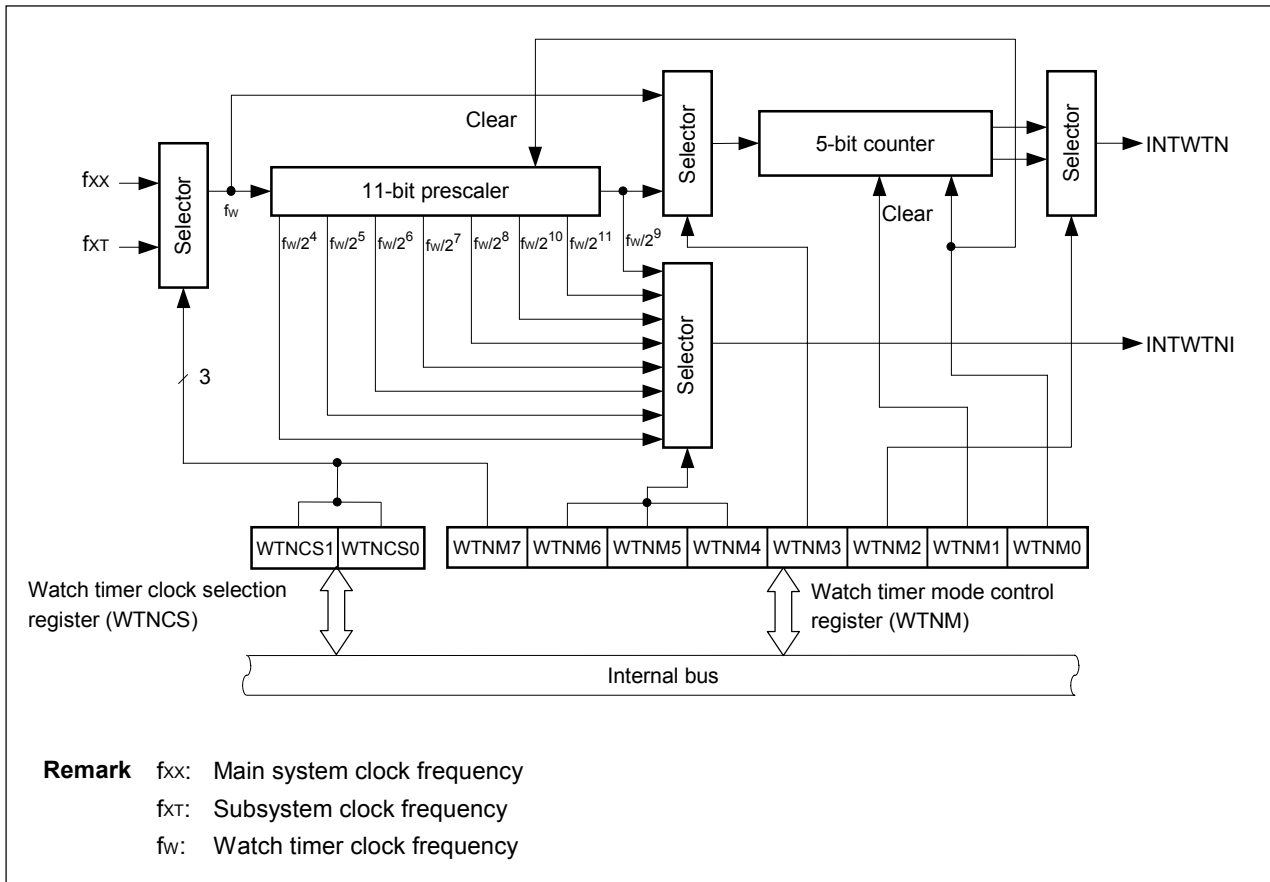
9.1 Function

The watch timer has the following functions.

- Watch timer
- Interval timer

The watch timer and interval timer functions can be used at the same time.

Figure 9-1. Block Diagram of Watch Timer



(1) Watch timer

The watch timer generates an interrupt request (INTWTN) at time intervals of 0.5 second or 0.25 second using the main system clock or subsystem clock.

(2) Interval timer

The watch timer generates an interrupt request (INTWTNI) at time intervals specified in advance.

Table 9-1. Interval Time of Interval Timer

Interval Time	$f_{XT} = 32.768 \text{ kHz}$
$2^4 \times 1/f_w$	488 μs
$2^5 \times 1/f_w$	977 μs
$2^6 \times 1/f_w$	1.95 ms
$2^7 \times 1/f_w$	3.91 ms
$2^8 \times 1/f_w$	7.81 ms
$2^9 \times 1/f_w$	15.6 ms
$2^{10} \times 1/f_w$	31.2 ms
$2^{11} \times 1/f_w$	62.4 ms

Remark Watch timer clock frequency

9.2 Configuration

The watch timer includes the following hardware.

Table 9-2. Configuration of Watch Timer

Item	Configuration
Counter	5 bits \times 1
Prescaler	11 bits \times 1
Control registers	Watch timer mode control register (WTNM) Watch timer clock selection register (WTNCS)

9.3 Watch Timer Control Register

The watch timer mode control register (WTNM) and watch timer clock selection register (WTNCS) control the watch timer. The watch timer should be operated after setting the count clock.

(1) Watch timer mode control register (WTNM)

This register enables or disables the count clock and operation of the watch timer, sets the interval time of the prescaler, controls the operation of the 5-bit counter, and sets the set time of the watch flag.

WTNM is set by an 8- or 1-bit memory manipulation instruction.

RESET input clears WTNM to 00H.

After reset: 00H R/W

Address: FFFFF360H

	7	6	5	4	3	2	1	0
WTNM	WTNM7	WTNM6	WTNM5	WTNM4	WTNM3	WTNM2	WTNM1	WTNM0

WTNM6	WTNM5	WTNM4	Selection of interval time of prescaler
0	0	0	$2^4/f_w$ (488 μ s)
0	0	1	$2^5/f_w$ (977 μ s)
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)
1	1	0	$2^{10}/f_w$ (31.2 ms)
1	1	1	$2^{11}/f_w$ (62.4 ms)

WTNM3	WTNM2	Selection of set time of watch flag
0	0	$2^{14}/f_w$ (0.5 s)
0	1	$2^{13}/f_w$ (0.25 s)
1	0	$2^5/f_w$ (977 μ s)
1	1	$2^4/f_w$ (488 μ s)

WTNM1	Operation of 5-bit counter
0	Cleared after operation stops
1	Operation starts

WTNM0	Operation of watch timer
0	Operation stopped (both prescaler and 5-bit counter cleared)
1	Operation enabled

- Remarks**
1. f_w : Watch timer clock frequency
 2. Values in parentheses apply when $f_w = 32.768$ kHz.
 3. For the settings of WTNM7, refer to **9.3 (2) Watch timer clock selection register (WTNCS)**.

(2) Watch timer clock selection register (WTNCS)

This register selects the count clock of the watch timer.

WTNCS is set by an 8-bit memory manipulation instruction.

RESET input clears WTNCS to 00H.

★ **Caution** Do not change the contents of the WTNM and WTNCS registers (interval time, watch flag set time, count clock) during a watch timer operation.

After reset: 00H R/W Address: FFFFF364H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	WTNCS1	WTNCS0

WTNCS1	WTNCS0	WTNM7	Selection of count clock	Main clock frequency
0	0	0	$f_{xx}/2^7$	4.194 MHz
0	0	1	f_{XT} (subsystem clock)	-
0	1	0	$f_{xx}/3 \times 2^6$	6.291 MHz
0	1	1	$f_{xx}/2^8$	8.388 MHz
1	0	0	Setting prohibited	-
1	0	1	Setting prohibited	-
1	1	0	$f_{xx}/3 \times 2^7$	12.582 MHz
1	1	1	Setting prohibited	-

Remark WTNM7 is bit 7 of the WTNM register

9.4 Operation

9.4.1 Operation as watch timer

The watch timer operates with time intervals of 0.5 second with the subsystem clock (32.768 kHz).

The watch timer generates an interrupt request at fixed time intervals.

The count operation of the watch timer is started when bits 0 (WTNM0) and 1 (WTNM1) of the watch timer mode control register (WTNM) are set to 1. When these bits are cleared to 0, the 11-bit prescaler and 5-bit counter are cleared, and the watch timer stops the count operation.

Setting the WTNM1 bit to 0 can clear the watch timer. An error of up to 15.6 ms may occur at this time.

Setting the WTNM0 bit to 0 can clear the interval timer. However, an error up to 0.5 sec. may occur after a watch timer overflow (INTWTN) because the 5-bit counter is also cleared.

9.4.2 Operation as interval timer

The watch timer can also be used as an interval timer that repeatedly generates an interrupt at intervals specified by a preset count value.

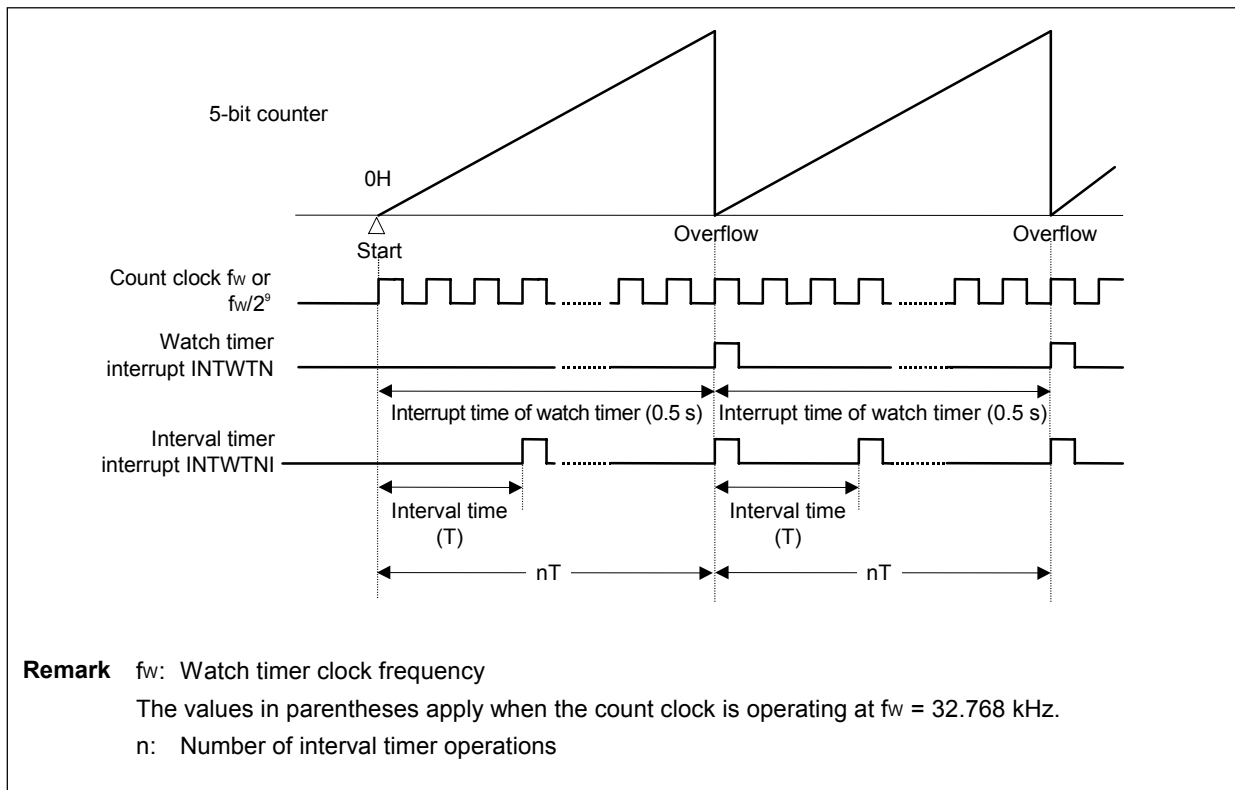
The interval time can be selected by bits 4 to 6 (WTNM4 to WTNM6) of the watch timer mode control register (WTNM).

Table 9-2. Interval Time of Interval Timer

WTNM6	WTNM5	WTNM4	Interval Time	$f_w = 32.768 \text{ kHz}$
0	0	0	$2^4 \times 1/f_w$	488 μs
0	0	1	$2^5 \times 1/f_w$	977 μs
0	1	0	$2^6 \times 1/f_w$	1.95 ms
0	1	1	$2^7 \times 1/f_w$	3.91 ms
1	0	0	$2^8 \times 1/f_w$	7.81 ms
1	0	1	$2^9 \times 1/f_w$	15.6 ms
1	1	0	$2^{10} \times 1/f_w$	31.2 ms
1	1	1	$2^{11} \times 1/f_w$	62.4 ms

Remark f_w : Watch timer clock frequency

Figure 9-2. Operation Timing of Watch Timer/Interval Timer

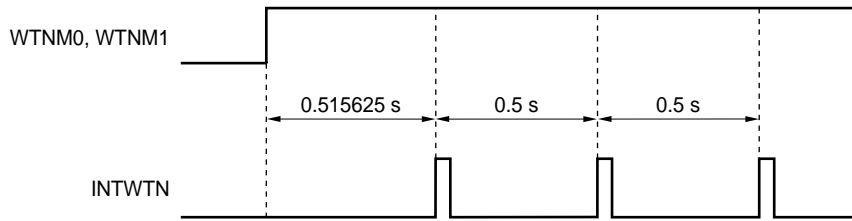


9.4.3 Cautions

It takes some time to generate the first watch timer interrupt request (INTWTN) after operation is enabled (WTNM1 and WTNM0 bits of WTNM register = 1).

Figure 9-3. Watch Timer Interrupt Request (INTWTN) Generation (Interrupt Period = 0.5 s)

It takes 0.515625 s to generate the first INTWTN ($2^9 \times 1/32.768 = 0.015625$ s longer). INTWTN is then generated every 0.5 s.



CHAPTER 10 WATCHDOG TIMER FUNCTION

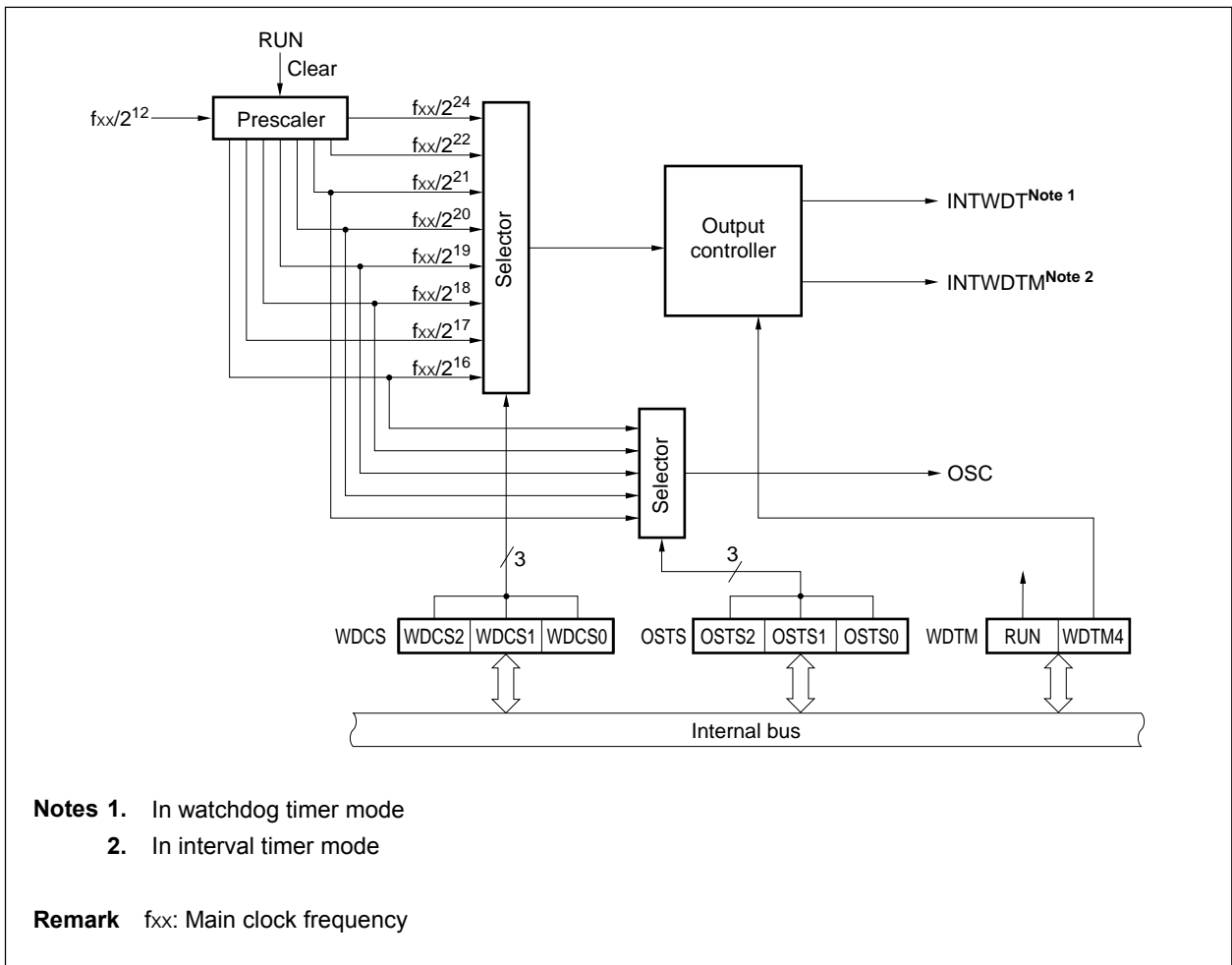
10.1 Functions

The watchdog timer has the following functions.

- Watchdog timer
- Interval timer
- Oscillation stabilization time selection

Caution Use the watchdog timer mode register (WDTM) to select the watchdog timer mode or the interval timer mode.

Figure 10-1. Block Diagram of Watchdog Timer



(1) Watchdog timer mode

This mode detects an inadvertent program loop. When a loop is detected, a non-maskable interrupt can be generated.

Table 10-1. Loop Detection Time of Watchdog Timer

Clock	Loop Detection Time	
	f _{xx} = 16 MHz	f _{xx} = 8 MHz
$2^{16}/f_{xx}$	4.1 ms	8.2 ms
$2^{17}/f_{xx}$	8.2 ms	16.4 ms
$2^{18}/f_{xx}$	16.4 ms	32.8 ms
$2^{19}/f_{xx}$	32.8 ms	65.5 ms
$2^{20}/f_{xx}$	65.5 ms	131.1 ms
$2^{21}/f_{xx}$	131.1 ms	262.1 ms
$2^{22}/f_{xx}$	262.1 ms	524.3 ms
$2^{24}/f_{xx}$	1.05 s	2.10 s

(2) Interval timer mode

Interrupts are generated at a preset time interval.

Table 10-2. Interval Time of Interval Timer

Clock	Interval Time	
	f _{xx} = 16 MHz	f _{xx} = 8 MHz
$2^{16}/f_{xx}$	4.1 ms	8.2 ms
$2^{17}/f_{xx}$	8.2 ms	16.4 ms
$2^{18}/f_{xx}$	16.4 ms	32.8 ms
$2^{19}/f_{xx}$	32.8 ms	65.5 ms
$2^{20}/f_{xx}$	65.5 ms	131.1 ms
$2^{21}/f_{xx}$	131.1 ms	262.1 ms
$2^{22}/f_{xx}$	262.1 ms	524.3 ms
$2^{24}/f_{xx}$	1.05 s	2.10 s

10.2 Configuration

The watchdog timer includes the following hardware.

Table 10-3. Watchdog Timer Configuration

Item	Configuration
Control registers	Oscillation stabilization time selection register (OSTS) Watchdog timer clock selection register (WDCS) Watchdog timer mode register (WDTM)

10.3 Watchdog Timer Control Register

The watchdog timer is controlled by the following registers.

- Oscillation stabilization time selection register (OSTS)
- Watchdog timer clock selection register (WDCS)
- Watchdog timer mode register (WDTM)

(1) Oscillation stabilization time selection register (OSTS)

This register selects the oscillation stabilization time after a reset is applied or the STOP mode is canceled until the oscillation is stable.

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

$\overline{\text{RESET}}$ input sets OSTS to 04H.

After reset: 04H		R/W	Address: FFFFF380H					
	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0
	OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection				
				Clock	f_{xx}			
					16 MHz	8 MHz		
	0	0	0	$2^{16}/f_{xx}$	4.1 ms	8.2 ms		
	0	0	1	$2^{18}/f_{xx}$	16.4 ms	32.8 ms		
	0	1	0	$2^{19}/f_{xx}$	32.8 ms	65.5 ms		
	0	1	1	$2^{20}/f_{xx}$	65.5 ms	131.1 ms		
	1	0	0	$2^{21}/f_{xx}$ (after reset)	131.1 ms	262.1 ms		
	Other than above			Setting prohibited				

(2) Watchdog timer clock selection register (WDCS)

This register selects the overflow times of the watchdog timer and the interval timer.

WDCS is set by an 8-bit memory manipulation instruction.

RESET input sets WDCS to 00H.

After reset: 00H R/W Address: FFFF382H

	7	6	5	4	3	2	1	0
WDCS	0	0	0	0	0	WDCS2	WDCS1	WDCS0

WDCS2	WDCS1	WDCS0	Watchdog timer/interval timer overflow time		
			Clock	f _{xx}	
				16 MHz	8 MHz
0	0	0	$2^{16}/f_{xx}$	4.1 ms	8.2 ms
0	0	1	$2^{17}/f_{xx}$	8.2 ms	16.4 ms
0	1	0	$2^{18}/f_{xx}$	16.4 ms	32.8 ms
0	1	1	$2^{19}/f_{xx}$	32.8 ms	65.5 ms
1	0	0	$2^{20}/f_{xx}$	65.5 ms	131.1 ms
1	0	1	$2^{21}/f_{xx}$	131.1 ms	262.1 ms
1	1	0	$2^{22}/f_{xx}$	262.1 ms	524.3 ms
1	1	1	$2^{24}/f_{xx}$	1.05 s	2.10 s

(3) Watchdog timer mode register (WDTM)

This register sets the operation mode of the watchdog timer, and enables and disables counting. WDTM is set by an 8- or 1-bit memory manipulation instruction. $\overline{\text{RESET}}$ input sets WDTM to 00H.

After reset: 00H	R/W	Address: FFFF384H						
	7	6	5	4	3	2	1	0
WDTM	RUN	0	0	WDTM4	0	0	0	0
	RUN	Operation mode selection for the watchdog timer ^{Note 1}						
	0	Count disabled						
	1	Count cleared and counting starts						
	WDTM4	Operation mode selection for the watchdog timer ^{Note 2}						
	0	Interval timer mode (If an overflow occurs, a maskable interrupt, INTWDTM, is generated.)						
	1	Watchdog timer mode 1 (If an overflow occurs, a non-maskable interrupt, INTWDT, is generated.)						

- Notes 1.** Once RUN is set (1), the register cannot be cleared (0) by software. Therefore, when the count starts, the count cannot be stopped except by $\overline{\text{RESET}}$ input.
- 2.** Once WDTM4 is set (1), the register cannot be cleared (0) by software.

Caution If RUN is set (1) and the watchdog timer is cleared, the actual overflow time may be up to $2^{12}/f_{\text{xx}}$ seconds shorter than the set time.

10.4 Operation

10.4.1 Operation as watchdog timer

Set bit 4 (WDTM4) of the watchdog timer mode register (WDTM) to 1 to operate as a watchdog timer to detect an inadvertent program loop.

Setting bit 7 (RUN) of WDTM to 1 starts the count. After counting starts, if RUN is set to 1 again within the set time interval for loop detection, the watchdog timer is cleared and counting starts again.

If RUN is not set to 1 and the loop detection time has elapsed, a non-maskable interrupt (INTWDT) is generated (no reset functions).

The watchdog timer stops running in the STOP mode and IDLE mode. Consequently, set RUN to 1 and clear the watchdog timer before entering the STOP mode or IDLE mode. Do not set the watchdog timer when using the HALT mode since the watchdog timer continues running in HALT mode.

- Cautions**
1. The actual loop detection time may be up to $2^{12}/f_{xx}$ seconds shorter than the set time.
 2. When the subsystem clock is selected for the CPU clock, the watchdog timer stops counting (pauses).

Table 10-4. Loop Detection Time of Watchdog Timer

Clock	Loop Detection Time	
	$f_{xx} = 16 \text{ MHz}$	$f_{xx} = 8 \text{ MHz}$
$2^{16}/f_{xx}$	4.1 ms	8.2 ms
$2^{17}/f_{xx}$	8.2 ms	16.4 ms
$2^{18}/f_{xx}$	16.4 ms	32.8 ms
$2^{19}/f_{xx}$	32.8 ms	65.5 ms
$2^{20}/f_{xx}$	65.5 ms	131.1 ms
$2^{21}/f_{xx}$	131.1 ms	262.1 ms
$2^{22}/f_{xx}$	262.1 ms	524.3 ms
$2^{24}/f_{xx}$	1.05 s	2.10 s

10.4.2 Operation as interval timer

Set bit 4 (WDTM4) to 0 in the watchdog timer mode register (WDTM) to operate the watchdog timer as an interval timer that repeatedly generates interrupts with a preset count value as the interval.

When operating as an interval timer, the interrupt mask flag (WDTMK) of the WDTIC register and the priority setting flag (WDTPR0 to WDTPR2) become valid, and a maskable interrupt (INTWDTM) can be generated. The default priority of INTWDTM has the highest priority setting of the maskable interrupts.

The interval timer continues operating in the HALT mode and stops in the STOP mode and IDLE mode. Therefore, before entering the STOP mode/IDLE mode, set the RUN bit of WDTM register to 1 to clear the interval timer, and then execute the STOP instruction.

- Cautions**
1. Once bit 4 (WDTM4) of WDTM is set to 1 (selecting the watchdog timer mode), the interval timer mode is not entered as long as **RESET** is not input.
 2. The interval time immediately after being set by WDTM may be up to $2^{12}/f_{xx}$ seconds shorter than the set time.
 3. When the subsystem clock is selected for the CPU clock, the watchdog timer stops counting (pauses).

Table 10-5. Interval Time of Interval Timer

Clock	Interval Time	
	$f_{xx} = 16 \text{ MHz}$	$f_{xx} = 8 \text{ MHz}$
$2^{16}/f_{xx}$	4.1 ms	8.2 ms
$2^{17}/f_{xx}$	8.2 ms	16.4 ms
$2^{18}/f_{xx}$	16.4 ms	32.8 ms
$2^{19}/f_{xx}$	32.8 ms	65.5 ms
$2^{20}/f_{xx}$	65.5 ms	131.1 ms
$2^{21}/f_{xx}$	131.1 ms	262.1 ms
$2^{22}/f_{xx}$	262.1 ms	524.3 ms
$2^{24}/f_{xx}$	1.05 s	2.10 s

10.5 Standby Function Control Register

The wait time from when the stop mode is canceled until the oscillation stabilizes is controlled by the oscillation stabilization time selection register (OSTS).

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

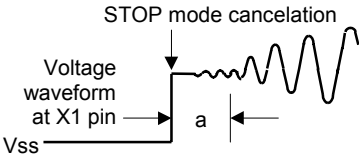
$\overline{\text{RESET}}$ input sets OSTS to 04H.

After reset: 04H R/W Address: FFFF380H

	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0

OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection		
			Clock	f _{xx}	
				16 MHz	8 MHz
0	0	0	$2^{16}/f_{xx}$	4.1 ms	8.2 ms
0	0	1	$2^{18}/f_{xx}$	16.4 ms	32.8 ms
0	1	0	$2^{19}/f_{xx}$	32.8 ms	65.5 ms
0	1	1	$2^{20}/f_{xx}$	65.5 ms	131.1 ms
1	0	0	$2^{21}/f_{xx}$ (after reset)	131.1 ms	262.1 ms
Other than above			Setting prohibited		

Caution The wait time at the cancellation of the STOP mode does not include the time (“a” in the figure below) until clock oscillation starts after STOP mode is canceled by $\overline{\text{RESET}}$ input or interrupt generation.



CHAPTER 11 SERIAL INTERFACE FUNCTION

11.1 Overview

The V850/SF1 incorporates the following serial interfaces.

- Channel 0: 3-wire serial I/O (CSI0)/I²C^{Note}
- Channel 1: 3-wire serial I/O (CSI1)/Asynchronous serial interface (UART0)
- Channel 3: 3-wire serial I/O (CSI3)/Asynchronous serial interface (UART1)
- Channel 4: 8 to 16-bit variable-length 3-wire serial I/O (CSI4)

Note I²C0 supports multiple masters.

Either 3-wire serial I/O or I²C can be used as a serial interface.

11.2 3-Wire Serial I/O (CSI0, CSI1, CSI3)

Remark n = 0, 1, 3 in section 11.2.

CSIn has the following two modes.

(1) Operation stop mode

This mode is used when serial transfers are not performed.

(2) 3-wire serial I/O mode (fixed as MSB first)

This is an 8-bit data transfer mode using three lines: a serial clock line ($\overline{\text{SCKn}}$), a serial output line (SOn), and a serial input line (SIn).

Since simultaneous transmit and receive operations are enabled in 3-wire serial I/O mode, the processing time for data transfer is reduced.

The first bit in the 8-bit data in serial transfers is fixed as the MSB.

3-wire serial I/O mode is useful for connection to a peripheral I/O device that includes a clocked serial interface, a display controller, etc.

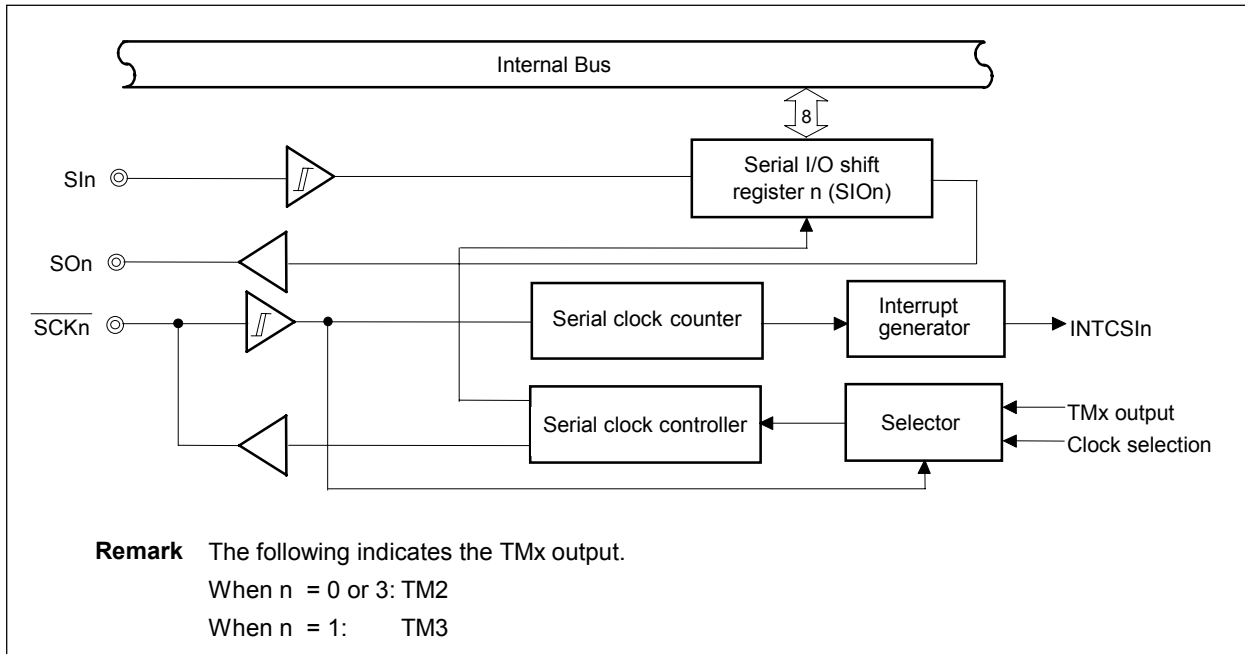
11.2.1 Configuration

CSIn includes the following hardware.

Table 11-1. Configuration of CSIn

Item	Configuration
Registers	Serial I/O shift register n (SIO _n)
Control registers	Serial operation mode register n (CSIM _n)
	Serial clock selection register n (CSIS _n)

Figure 11-1. Block Diagram of 3-Wire Serial I/O



(1) Serial I/O shift register n (SIO_n)

SIO_n is an 8-bit register that performs parallel-serial conversion and serial transmission/reception (shift operations) in synchronization with the serial clock.

SIO_n is set by an 8-bit memory manipulation instruction.

When "1" is set to bit 7 (CSIE_n) of serial operation mode register n (CSIM_n), a serial operation can be started by writing data to or reading data from SIO_n.

When transmitting, data written to SIO_n is output via the serial output (SOn).

When receiving, data is read from the serial input (SIn) and written to SIO_n.

RESET input resets these registers to 00H.

Caution Do not access SIO_n except via the transfer start trigger during a transfer operation (read is disabled when MODE_n = 0 and write is disabled when MODE_n = 1).

11.2.2 CSIn control registers

CSIn is controlled by the following registers.

- Serial operation mode register n (CSIMn)
- Serial clock selection register n (CSISn)

(1) Serial operation mode register n (CSIMn)

CSIMn is used to enable or disable the serial clock, operation modes, and specific operations of serial interface channel n.

CSIMn can be set by an 8- or 1-bit memory manipulation instruction.

RESET input sets these registers to 00H.

After reset: 00H R/W Address: CSIM0 FFFF2A2H
 CSIM1 FFFF2B2H
 CSIM3 FFFF2D2H

	7	6	5	4	3	2	1	0
CSIMn	CSIE _n	0	0	0	0	MODE _n	SCL _{n1}	SCL _{n0}

CSIE _n	SIO _n operation enable/disable specification		
	Shift register operation	Serial counter	Port
0	Operation disabled	Cleared	Port function ^{Note 1}
1	Operation enabled	Count operation enabled	Serial function + port function ^{Note 2}

MODE _n	Transfer operation mode flag		
	Operation mode	Transfer start trigger	SIO _n output
0	Transmit/receive mode	SIO _n write	Normal output
1	Receive-only mode	SIO _n read	Port function

SCL _{n2}	SCL _{n1}	SCL _{n0}	Clock selection
0	0	0	External clock input (SCK _n)
0	0	1	at n = 0, 3: TM2 output at n = 1: TM3 output
0	1	0	f _{xx} /8
0	1	1	f _{xx} /16
1	0	0	Setting prohibited
1	0	1	Setting prohibited
1	1	0	f _{xx} /32
1	1	1	f _{xx} /64

Notes 1. The SIO_n, SIO_n, and SCK_n pins are used as port function pins when CSIE_n = 0 (SIO_n operation stop status).

2. When CSIE_n = 1 (SIO_n operation enable status), the port function is available for the SIO_n pin when only using the transmit function and SIO_n pin when only using the receive function.

Caution Do not perform bit manipulation of SCL_{n1} and SCL_{n0}.

Remarks 1. Refer to 11.2.2 (2) Serial clock selection register n (CSIS_n) for the SCL_{n2} bit.

2. When the selection clock is output as a timer, pins P30/TO2/TI2 and P31/TO3/TI3 do not need to be set in timer output mode.

★

(2) Serial clock selection register n (CSISn)

CSISn is used to set the serial clock of serial interface channel n.

CSISn can be set by an 8-bit memory manipulation instruction.

RESET input sets these registers to 00H.

After reset : 00H	R/W	Address: CSIS0	FFFFF2A4H
		CSIS1	FFFFF2B4H
		CSIS3	FFFFF2D4H
		7	6
		5	4
		3	2
		1	0
CSISn		0	0
		0	0
		0	0
		0	0
		0	0
		0	0
		0	SCLn2

Remark Refer to 11.2.2 (1) Serial operation mode register n (CSIMn) for the setting of the SCLn2 bit.

11.2.3 Operations

The CSIn has the following two operation modes.

- Operation stopped mode
- 3-wire serial I/O mode

(1) Operation stopped mode

In this mode, serial transfers are not performed and therefore power consumption can be reduced.

When in operation stopped mode, SIn, SOn, and SCKn pins can be used as normal I/O port pins.

(a) Register settings

Operation stopped mode is set via the CSIEn bit of serial operation mode register n (CSIMn).

Figure 11-2. CSIMn Setting (Operation Stopped Mode)

After reset : 00H	R/W	Address: CSIM0	FFFFF2A2H
		CSIM1	FFFFF2B2H
		CSIM3	FFFFF2D2H
		7	6
		5	4
		3	2
		1	0
CSIMn		CSIEn	0
		0	0
		0	0
		0	0
		MODEn	SCLn1
			SCLn0

CSIEn	SIO n operation enable/disable specification		
	Shift register operation	Serial counter	Port
	0	Operation disabled	Cleared
			Port function

(2) 3-wire serial I/O mode

3-wire serial I/O mode is useful when connecting to a peripheral I/O device that includes a clocked serial interface, a display controller, etc.

This mode executes data transfers via three lines: a serial clock line (\overline{SCKn}), a serial output line (SOn), and a serial input line (SIn).

(a) Register settings

3-wire serial I/O mode is set by serial operation mode register n (CSIMn).

Figure 11-3. CSIMn Setting (3-Wire Serial I/O Mode)

After reset : 00H	R/W	Address: CSIM0	FFFFF2A2H
		CSIM1	FFFFF2B2H
		CSIM3	FFFFF2D2H

	7	6	5	4	3	2	1	0
CSIMn	CSIE _n	0	0	0	0	MODE _n	SCL _{n1}	SCL _{n0}

CSIE _n	SIO _n operation enable/disable specification		
	Shift register operation	Serial counter	Port
	1	Operation enabled	Count operation enabled

MODE _n	Transfer operation mode flag		
	Operation mode	Transfer start trigger	SIO _n output
	0	Transmit/receive mode	Write to SIO _n
1	Receive-only mode	Read from SIO _n	Port function

SCL _{n2}	SCL _{n1}	SCL _{n0}	Clock selection
0	0	0	External clock input (\overline{SCKn})
0	0	1	when n = 0, 3: TM2 output when n = 1: TM3 output
0	1	0	f _{xx} /8
0	1	1	f _{xx} /16
1	0	0	Setting prohibited
1	0	1	Setting prohibited
1	1	0	f _{xx} /32
1	1	1	f _{xx} /64

★ **Remark** When the selection clock is output as a timer, pins P30/TO2/TI2 and P31/TO3/TI3 do not need to be set in timer output mode.

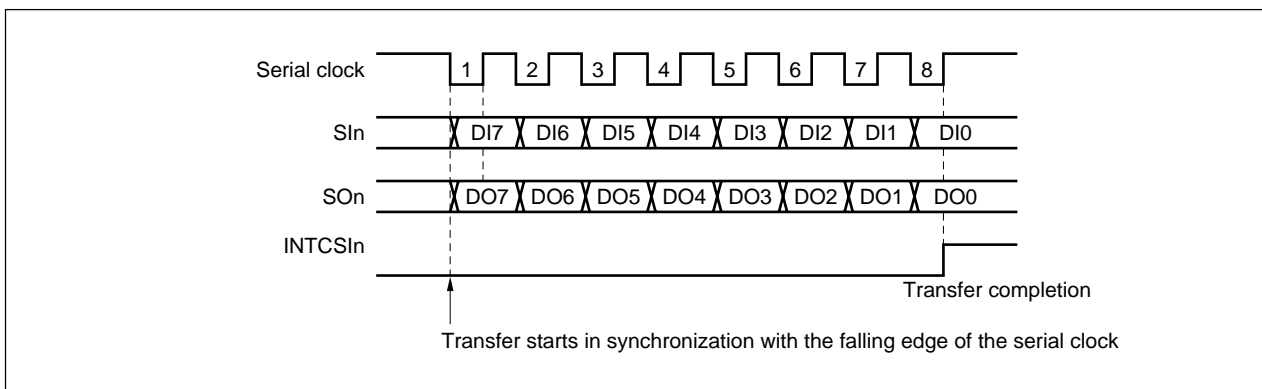
(b) Communication operations

In 3-wire serial I/O mode, data is transmitted and received in 8-bit units. Each bit of data is sent or received in synchronization with the serial clock.

Serial I/O shift register n (SIO_n) is shifted in synchronization with the falling edge of the serial clock. Transmission data is held in the SO_n latch and is output from the SO_n pin. Data that is received via the SIn pin in synchronization with the rising edge of the serial clock is latched to SIO_n.

Completion of an 8-bit transfer automatically stops operation of SIO_n and sets the interrupt request flag (INTCSIn).

Figure 11-4. Timing of 3-Wire Serial I/O Mode

**(c) Transfer start**

A serial transfer starts when the following two conditions have been satisfied and transfer data has been set to serial I/O shift register n (SIO_n).

- The SIO_n operation control bit (CSIE_n) = 1
- After an 8-bit serial transfer, the internal serial clock is either stopped or is set to high level.

The transfer data to SIO_n is set as follows.

- Transmit/receive mode
When CSIE_n = 1 and MODE_n = 0, transfer starts when writing to SIO_n.
- Receive-only mode
When CSIE_n = 1 and MODE_n = 1, transfer starts when reading from SIO_n.

Caution After data has been written to SIO_n, transfer will not start even if the CSIE_n bit value is set to 1.

Completion of an 8-bit transfer automatically stops the serial transfer operation and sets the interrupt request flag (INTCSIn).

11.3 I²C Bus

To use the I²C bus function, set the P10/SDA0 and P12/SCL0 pins to N-ch open drain output.

I²C0 has the following two modes.

- Operation stopped mode
- I²C (Inter IC) bus mode (multiple masters supported)

(1) Operation stopped mode

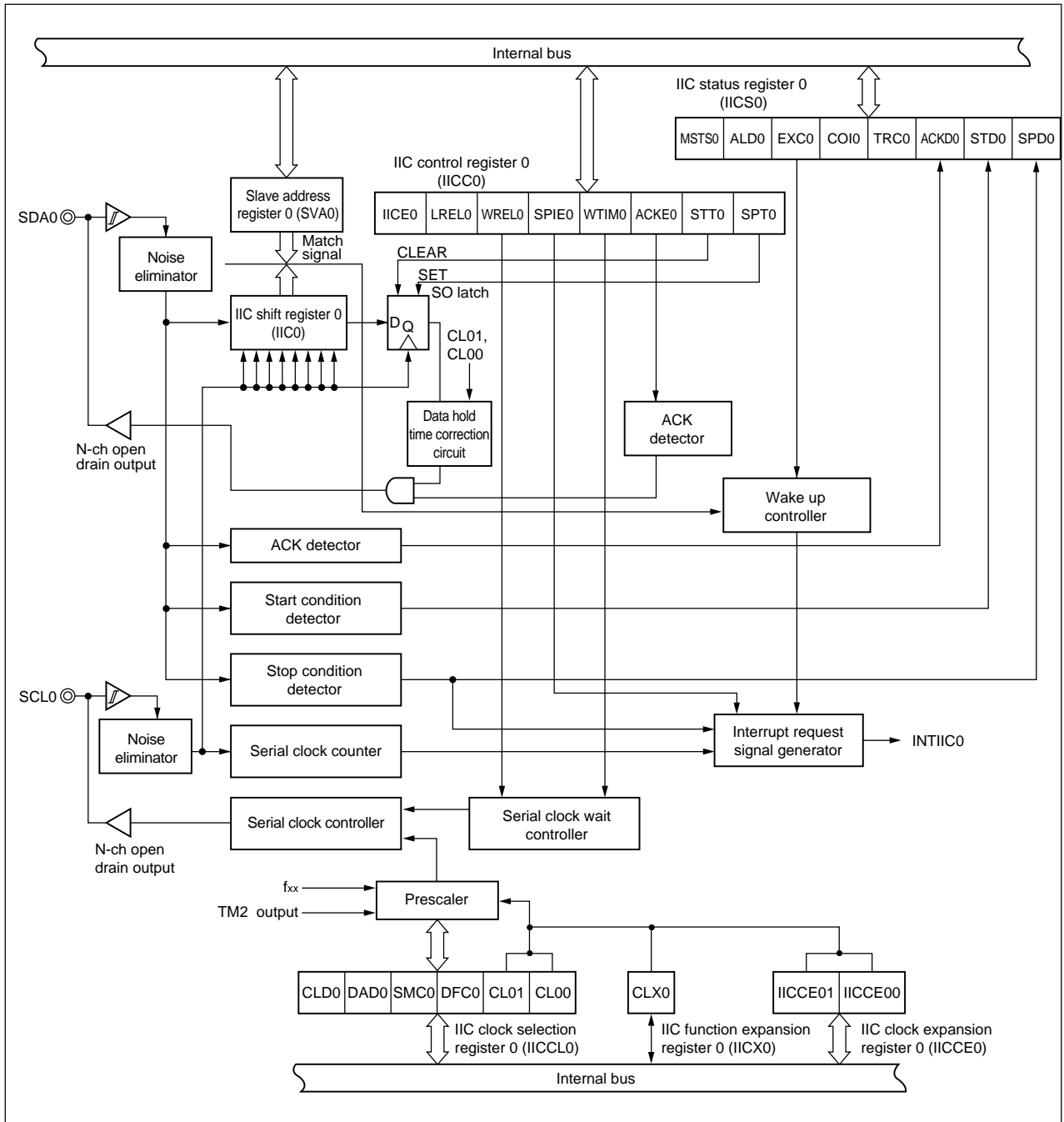
This mode is used when serial transfers are not performed. It can therefore be used to reduce power consumption.

(2) I²C bus mode (multiple masters support)

This mode is used for 8-bit data transfers with several devices via two lines: a serial clock line (SCL0) and a serial data bus line (SDA0).

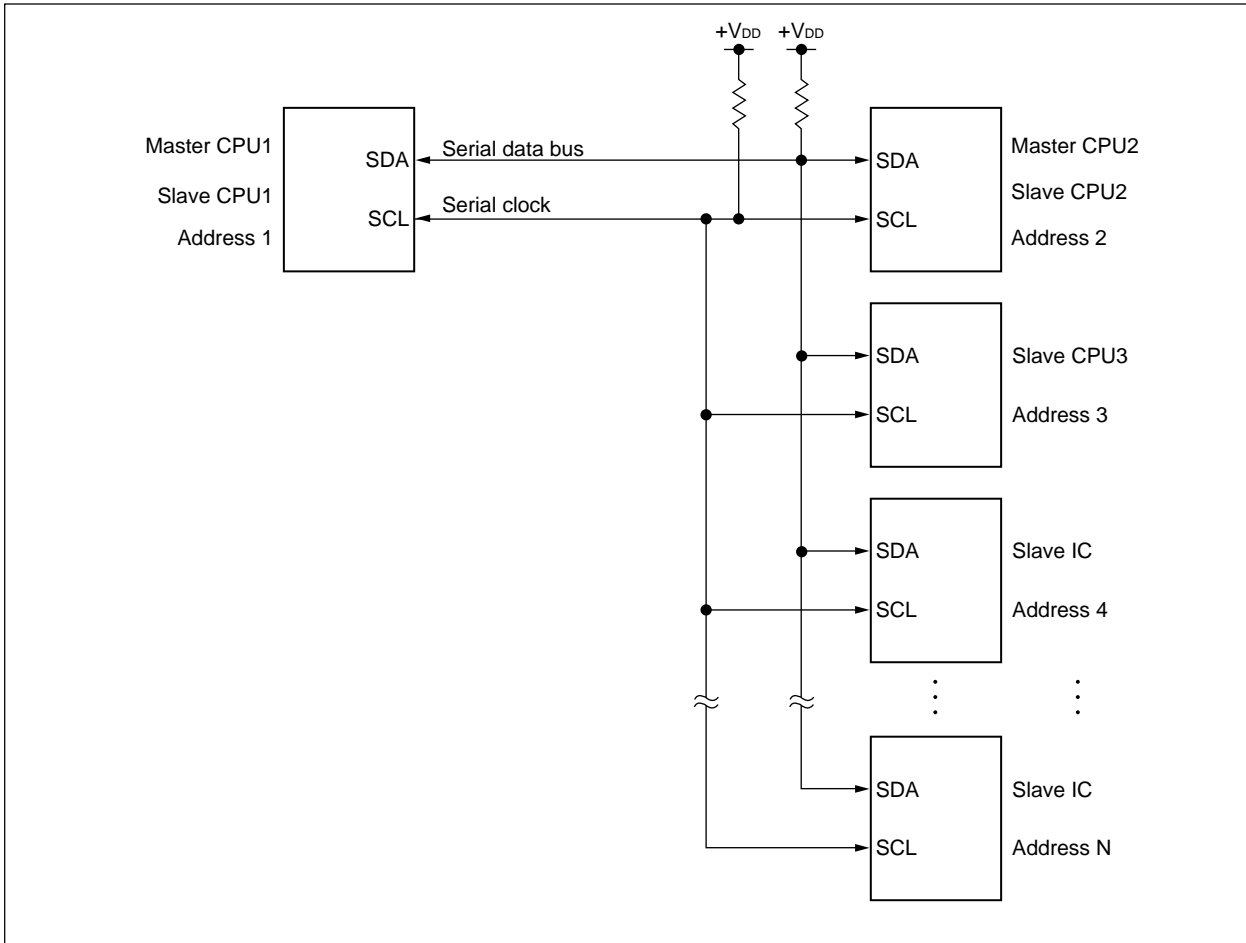
This mode complies with the I²C bus format and the master device can output “start condition”, “data”, and “stop condition” data to the slave device, via the serial data bus. The slave device automatically detects these received data by hardware. This function can simplify the part of application program that controls the I²C bus. Since SCL0 and SDA0 are open-drain outputs, the I²C0 requires pull-up resistors for the serial clock line and the serial data bus line.

Figure 11-5. Block Diagram of I²C0



A serial bus configuration example is shown below.

Figure 11-6. Serial Bus Configuration Example Using I²C Bus



11.3.1 Configuration

I²C0 includes the following hardware.

Table 11-2. Configuration of I²C0

Item	Configuration
Registers	IIC shift register 0 (IIC0) Slave address register 0 (SVA0)
Control registers	IIC control register 0 (IICC0) IIC status register 0 (IICS0) IIC clock selection register 0 (IICCL0) IIC clock expansion register 0 (IICCE0) IIC function expansion register 0 (IICX0)

(1) IIC shift register 0 (IIC0)

IIC0 is used to convert 8-bit serial data to 8-bit parallel data and to convert 8-bit parallel data to 8-bit serial data.

IIC0 can be used for both transmission and reception.

Write and read operations to IIC0 are used to control the actual transmit and receive operations.

IIC0 is set by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets the IIC0 to 00H.

(2) Slave address register 0 (SVA0)

SVA0 sets local addresses when in slave mode.

SVA0 is set by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets the SVA0 to 00H.

(3) SO latch

The SO latch is used to retain the SDA0 pin's output level.

(4) Wake-up controller

This circuit generates an interrupt request when the address received by this register matches the address value set to slave address register 0 (SVA0) or when an extension code is received.

(5) Clock selector

This selects the sampling clock to be used.

(6) Serial clock counter

This counter counts the serial clocks that are output and the serial clocks that are input during transmit/receive operations and is used to verify that 8-bit data was sent or received.

(7) Interrupt request signal generator

This circuit controls the generation of interrupt request signals (INTIIC0).

An I²C interrupt is generated following either of two triggers.

- Eighth or ninth clock of the serial clock (set by WTIM0 bit^{Note})
- Interrupt request generated when a stop condition is detected (set by SPIE0 bit^{Note})

Note WTIM0 bit: Bit 3 of IIC control register 0 (IICC0)

SPIE0 bit: Bit 4 of IIC control register 0 (IICC0)

(8) Serial clock controller

In master mode, this circuit generates the clock output via the SCL0 pin from a sampling clock.

(9) Serial clock wait controller

This circuit controls the wait timing.

(10) ACK output circuit, stop condition detector, start condition detector, and ACK detector

These circuits are used to output and detect various control signals.

(11) Data hold time correction circuit

This circuit generates the hold time for data corresponding to the falling edge of the serial clock.

11.3.2 I²C control registers

I²C0 is controlled by the following registers.

- IIC control register 0 (IICC0)
- IIC status register 0 (IICS0)
- IIC clock selection register 0 (IICCL0)
- IIC clock expansion register 0 (IICCE0)
- IIC function expansion register 0 (IICX0)

The following registers are also used.

- IIC shift register 0 (IIC0)
- Slave address register 0 (SVA0)

(1) IIC control register 0 (IICC0)

IICC0 is used to enable/disable I²C operations, set wait timing, and set other I²C operations.

IICC0 can be set by an 8- or 1-bit memory manipulation instruction.

RESET input sets IICC0 to 00H.

Caution In I²C0 bus mode, set the port 1 mode register (PM1) as follows. In addition, set each output latch to 0.

- Set P10 (SDA0) to output mode (PM10 = 0)
- Set P12 (SCL0) to output mode (PM12 = 0)

After reset: 00H R/W Address: FFFFF340H



IICE0	I ² C0 operation enable/disable specification
0	Operation stopped. IIC status register 0 (IICS0) preset. Internal operation stopped.
1	Operation enabled.
Condition for clearing (IICE0 = 0)	
<ul style="list-style-type: none"> • Cleared by instruction • When RESET is input 	
Condition for setting (IICE0 = 1)	
<ul style="list-style-type: none"> • Set by instruction 	

LRELO	Exit from communications
0	Normal operation
1	<p>This exits from the current communication operation and sets standby mode. This setting is automatically cleared after being executed. Its uses include cases in which a locally irrelevant extension code has been received. The SCL0 and SDA0 lines are set to high impedance.</p> <p>The following flags are cleared.</p> <ul style="list-style-type: none"> • STD0 • ACKD0 • TRC0 • COI0 • EXC0 • MST0 • STT0 • SPT0
<p>The standby mode following exit from communications remains in effect until the following communication entry conditions are met.</p> <ul style="list-style-type: none"> • After a stop condition is detected, restart is in master mode. • An address match or extension code reception occurs after the start condition. 	
Condition for clearing (LRELO = 0) ^{Note}	
<ul style="list-style-type: none"> • Automatically cleared after execution • When RESET is input 	
Condition for setting (LRELO = 1)	
<ul style="list-style-type: none"> • Set by instruction 	

Note This flag's signal is invalid when IICE0 = 0.

Remark

- STD0: Bit 1 of IIC status register 0 (IICS0)
- ACKD0: Bit 2 of IIC status register 0 (IICS0)
- TRC0: Bit 3 of IIC status register 0 (IICS0)
- COI0: Bit 4 of IIC status register 0 (IICS0)
- EXC0: Bit 5 of IIC status register 0 (IICS0)
- MST0: Bit 7 of IIC status register 0 (IICS0)

WRELO	Wait cancellation control	
0	Wait not canceled	
1	Wait canceled. This setting is automatically cleared after wait is canceled.	
Condition for clearing (WRELO = 0) ^{Note}		Condition for setting (WRELO = 1)
<ul style="list-style-type: none"> • Automatically cleared after execution • When RESET is input 		<ul style="list-style-type: none"> • Set by instruction

SPIE0	Enable/disable generation of interrupt request when stop condition is detected	
0	Disabled	
1	Enabled	
Condition for clearing (SPIE0 = 0) ^{Note}		Condition for setting (SPIE0 = 1)
<ul style="list-style-type: none"> • Cleared by instruction • When RESET is input 		<ul style="list-style-type: none"> • Set by instruction

WTIM0	Control of wait and interrupt request generation	
0	Interrupt request is generated at the eighth clock's falling edge. Master mode: After output of eight clocks, clock output is set to low level and wait is set. Slave mode: After input of eight clocks, the clock is set to low level and wait is set for master device.	
1	Interrupt request is generated at the ninth clock's falling edge. Master mode: After output of nine clocks, clock output is set to low level and wait is set. Slave mode: After input of nine clocks, the clock is set to low level and wait is set for master device.	
This bit's setting is invalid during an address transfer and is valid as the transfer is completed. When in master mode, a wait is inserted at the falling edge of the ninth clock during address transfers. For a slave device that has received a local address, a wait is inserted at the falling edge of the ninth clock after an ACK signal is issued. When the slave device has received an extension code, a wait is inserted at the falling edge of the eighth clock.		
Condition for clearing (WTIM0 = 0) ^{Note}		Condition for setting (WTIM0 = 1)
<ul style="list-style-type: none"> • Cleared by instruction • When RESET is input 		<ul style="list-style-type: none"> • Set by instruction

Note This flag's signal is invalid when IICE0 = 0.

ACKE0	Acknowledge control	
0	Acknowledgement disable.	
1	Acknowledgement enabled. During the ninth clock period, the SDA0 line is set to low level. However, ACK is invalid during address transfers and is valid when EXC0 = 1.	
Condition for clearing (ACKE0 = 0) ^{Note}		Condition for setting (ACKE0 = 1)
<ul style="list-style-type: none"> • Cleared by instruction • When RESET is input 		<ul style="list-style-type: none"> • Set by instruction

STT0	Start condition trigger	
0	Start condition not generated.	
1	<p>When bus is released (in STOP mode): Generates a start condition (for starting as master). The SDA0 line is changed from high level to low level and then the start condition is generated. Next, after the rated amount of time has elapsed, SCL0 is changed to low level.</p> <p>When bus is not used: This trigger functions as a start condition reserve flag. When set, it releases the bus and then automatically generates a start condition.</p> <p>In the wait state (when master device): Generates a restart condition after releasing the wait.</p>	
<p>Cautions concerning set timing</p> <ul style="list-style-type: none"> • For master reception: Cannot be set during transfer. Can be set only when ACKE0 has been set to 0 and slave has been notified of final reception. • For master transmission: A start condition cannot be generated normally during the ACK0 period. Set during the wait period. • Cannot be set at the same time as SPT0 		
Condition for clearing (STT0 = 0)		Condition for setting (STT0 = 1)
<ul style="list-style-type: none"> • Cleared by instruction • Cleared by loss in arbitration • Cleared after start condition is generated by master device • When LREL0 = 1 • When IICE0 = 0 • Cleared when RESET is input 		<ul style="list-style-type: none"> • Set by instruction

Note This flag's signal is invalid when IICE0 = 0.

Remark Bit 1 (STT0) is 0 if it is read immediately after data setting.

SPT0	Stop condition trigger				
0	Stop condition is not generated.				
1	Stop condition is generated (termination of master device's transfer). After the SDA0 line goes to low level, either set the SCL0 line to high level or wait until it goes to high level. Next, after the rated amount of time has elapsed, the SDA0 line is changed from low level to high level and a stop condition is generated.				
<p>Cautions concerning set timing</p> <ul style="list-style-type: none"> • For master reception: Cannot be set during transfer. Can be set only when ACKE0 has been set to 0 and during the wait period after slave has been notified of final reception. • For master transmission: A stop condition cannot be generated normally during the ACK0 period. Set during the wait period. • Cannot be set at the same time as STT0. • SPT0 can be set only when in master mode^{Note}. • When WTIM0 has been set to 0, if SPT0 is set during the wait period that follows output of eight clocks, note that a stop condition will be generated during the high-level period of the ninth clock. When a ninth clock must be output, WTIM0 should be changed from 0 to 1 during the wait period following output of eight clocks, and SPT0 should be set during the wait period that follows output of the ninth clock. 					
<table border="1"> <thead> <tr> <th>Condition for clearing (SPT0 = 0)</th> <th>Condition for setting (SPT0 = 1)</th> </tr> </thead> <tbody> <tr> <td> <ul style="list-style-type: none"> • Cleared by instruction • Cleared by loss in arbitration • Automatically cleared after stop condition is detected • When LREL0 = 1 • When IICE0 = 0 • Cleared when RESET is input </td> <td> <ul style="list-style-type: none"> • Set by instruction </td> </tr> </tbody> </table>		Condition for clearing (SPT0 = 0)	Condition for setting (SPT0 = 1)	<ul style="list-style-type: none"> • Cleared by instruction • Cleared by loss in arbitration • Automatically cleared after stop condition is detected • When LREL0 = 1 • When IICE0 = 0 • Cleared when RESET is input 	<ul style="list-style-type: none"> • Set by instruction
Condition for clearing (SPT0 = 0)	Condition for setting (SPT0 = 1)				
<ul style="list-style-type: none"> • Cleared by instruction • Cleared by loss in arbitration • Automatically cleared after stop condition is detected • When LREL0 = 1 • When IICE0 = 0 • Cleared when RESET is input 	<ul style="list-style-type: none"> • Set by instruction 				

Note Set SPT0 only in master mode. However, SPT0 must be set and a stop condition generated before the first stop condition is detected following the switch to the operation enabled status. For details, see **11.3.13 Cautions**.

Caution When bit 3 (TRC0) of IIC status register 0 (IICS0) is set to 1, WREL0 is set during the ninth clock and wait is canceled, after which TRC0 is cleared and the SDA0 line is set to high impedance.

Remark Bit 0 (SPT0) is 0 if it is read immediately after data setting.

(2) IIC status register 0 (IICS0)

IICS0 indicates the status of I²C0.

IICS0 can be set by an 8- or 1-bit memory manipulation instruction. IICS0 is a read-only register.

RESET input sets IICS0 to 00H.

(1/3)

After reset: 00H R Address: FFFFF342H

	7	6	5	4	3	2	1	0
IICS0	MSTS0	ALD0	EXC0	COI0	TRC0	ACKD0	STD0	SPD0

MSTS0	Master device status	
0	Slave device status or communication standby status	
1	Master device communication status	
Condition for clearing (MSTS0 = 0)		Condition for setting (MSTS0 = 1)
<ul style="list-style-type: none"> • When a stop condition is detected • When ALD0 = 1 • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When RESET is input 		<ul style="list-style-type: none"> • When a start condition is generated

ALD0	Detection of arbitration loss	
0	This status means either that there was no arbitration or that the arbitration result was a "win".	
1	This status indicates the arbitration result was a "loss". MSTS0 is cleared.	
Condition for clearing (ALD0 = 0)		Condition for setting (ALD0 = 1)
<ul style="list-style-type: none"> • Automatically cleared after IICS0 is read^{Note} • When IICE0 changes from 1 to 0 • When RESET is input 		<ul style="list-style-type: none"> • When the arbitration result is a "loss".

EXC0	Detection of extension code reception	
0	Extension code was not received.	
1	Extension code was received.	
Condition for clearing (EXC0 = 0)		Condition for setting (EXC0 = 1)
<ul style="list-style-type: none"> • When a start condition is detected • When a stop condition is detected • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When RESET is input 		<ul style="list-style-type: none"> • When the higher four bits of the received address data is either "0000" or "1111" (set at the rising edge of the eighth clock).

Note This register is also cleared when a bit manipulation instruction is executed for bits other than IICS0.

Remark LREL0: Bit 6 of IIC control register 0 (IICC0)
IICE0: Bit 7 of IIC control register 0 (IICC0)

COI0	Detection of matching addresses	
0	Addresses do not match.	
1	Addresses match.	
Condition for clearing (COI0 = 0)		Condition for setting (COI0 = 1)
<ul style="list-style-type: none"> • When a start condition is detected • When a stop condition is detected • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When RESET is input 		<ul style="list-style-type: none"> • When the received address matches the local address (SVA0) (set at the rising edge of the eighth clock).

TRC0	Detection of transmit/receive status	
0	Receive status (other than transmit status). The SDA0 line is set to high impedance.	
1	Transmit status. The value in the SO latch is enabled for output to the SDA0 line (valid starting at the falling edge of the first byte's ninth clock).	
Condition for clearing (TRC0 = 0)		Condition for setting (TRC0 = 1)
<ul style="list-style-type: none"> • When a stop condition is detected • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • Cleared by WREL0 = 1^{Note} • When ALD0 changes from 0 to 1 • When RESET is input <p>Master</p> <ul style="list-style-type: none"> • When "1" is output to the first byte's LSB (transfer direction specification bit) <p>Slave</p> <ul style="list-style-type: none"> • When a start condition is detected <p>When not used for communication</p>		<p>Master</p> <ul style="list-style-type: none"> • When a start condition is generated <p>Slave</p> <ul style="list-style-type: none"> • When "1" is input by the first byte's LSB (transfer direction specification bit)

Note TRC0 is cleared and the SDA0 line becomes high impedance when bit 5 (WREL0) of IIC control register 0 (IICC0) is set and the wait state is released at ninth clock by bit 3 (TRC0) of IIC status register 0 (IICS0) = 1.

Remark WREL0: Bit 5 of IIC control register 0 (IICC0)
 LREL0: Bit 6 of IIC control register 0 (IICC0)
 IICE0: Bit 7 of IIC control register 0 (IICC0)

ACKD0	Detection of ACK	
0	ACK was not detected.	
1	ACK was detected.	
Condition for clearing (ACKD0 = 0)		Condition for setting (ACKD0 = 1)
<ul style="list-style-type: none"> • When a stop condition is detected • At the rising edge of the next byte's first clock • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input 		<ul style="list-style-type: none"> • After the SDA0 line is set to low level at the rising edge of the SCL0's ninth clock

STD0	Detection of start condition	
0	Start condition was not detected.	
1	Start condition was detected. This indicates that the address transfer period is in effect	
Condition for clearing (STD0 = 0)		Condition for setting (STD0 = 1)
<ul style="list-style-type: none"> • When a stop condition is detected • At the rising edge of the next byte's first clock following address transfer • Cleared by LREL0 = 1 • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input 		When a start condition is detected

SPD0	Detection of stop condition	
0	Stop condition was not detected.	
1	Stop condition was detected. The master device's communication is terminated and the bus is released.	
Condition for clearing (SPD0 = 0)		Condition for setting (SPD0 = 1)
<ul style="list-style-type: none"> • At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition • When IICE0 changes from 1 to 0 • When $\overline{\text{RESET}}$ is input 		When a stop condition is detected

Remark LREL0: Bit 6 of IIC control register 0 (IICC0)

IICE0: Bit 7 of IIC control register 0 (IICC0)

(3) IIC clock selection register 0 (IICCL0)

IICCL0 is used to set the transfer clock for I²C0.

IICCL0 can be set by an 8- or 1-bit memory manipulation instruction. Bits SMC0, CL01 and CL00 are set using the CLX0 bit of IIC function expansion register 0 (IICX0) in combination with bits IICCE01 and IICCE00 of IIC clock expansion register 0 (IICCE0) (see **Table 11-3 Selection Clock Setting**).

RESET input sets IICCL0 to 00H.

After reset: 00H R/W^{Note} Address: FFFFF344H

	7	6	5	4	3	2	1	0
IICCL0	0	0	CLD0	DAD0	SMC0	DFC0	CL01	CL00

CLD0	Detection of SCL0 line level (valid only when IICE0 = 1)
0	SCL0 line was detected at low level.
1	SCL0 line was detected at high level.
Condition for clearing (CLD0 = 0)	
<ul style="list-style-type: none"> • When the SCL0 line is at low level • When IICE0 = 0 • When RESET is input 	
Condition for setting (CLD0 = 1)	
<ul style="list-style-type: none"> • When the SCL0 line is at high level 	

DAD0	Detection of SDA0 line level (valid only when IICE0 = 1)
0	SDA0 line was detected at low level.
1	SDA0 line was detected at high level.
Condition for clearing (DAD0 = 0)	
<ul style="list-style-type: none"> • When the SDA0 line is at low level • When IICE0 = 0 • When RESET is input 	
Condition for setting (DAD0 = 1)	
<ul style="list-style-type: none"> • When the SDA0 line is at high level 	

SMC0	Operation mode switching
0	Operates in standard mode.
1	Operates in high-speed mode.

DFC0	Digital filter operation control
0	Digital filter off.
1	Digital filter on.
The digital filter can be used only in high-speed mode.	
In high-speed mode, the transfer clock does not vary regardless of DFC0 switching (on/off).	

Note Bits 4 and 5 are read-only bits.

Remark IICE0: Bit 7 of IIC control register 0 (IICC0)

(4) IIC function expansion register 0 (IICX0)

This register sets the function expansion of I²C0 (valid only in high-speed mode).

IICX0 is set by an 8- or 1-bit memory manipulation instruction. Set the CLX0 bit in combination with the SMC0, DFC0, CL01, and the CL00 bits of IIC clock selection register 0 (IICCL0) and the IICCE01 and IICCE00 bits of IIC clock expansion register 0 (IICCE0) (see **Table 11-3 Selection Clock Setting**).

RESET input sets IICX0 to 00H.

After reset: 00H	R/W	Address: FFFFF34AH							
		7	6	5	4	3	2	1	0
IICX0		0	0	0	0	0	0	0	CLX0

(5) IIC clock expansion register 0 (IICCE0)

This register sets the clock selection expansion of I²C0.

IICCE0 is set by an 8-bit memory manipulation instruction. Set the IICCE01 and IICCE00 bits in combination with the SMC0, CL01, and CL00 bits of IIC clock selection register 0 (IICCL0) and the CLX0 bit of IIC function expansion register 0 (IICX0) (see **Table 11-3 Selection Clock Setting**).

RESET input sets IICCE0 to 00H.

After reset: 00H	R/W	Address: FFFFF34CH							
		7	6	5	4	3	2	1	0
IICCE0		0	0	0	0	0	0	IICCE01	IICCE00

★ **(6) I²C0 transfer clock setting method**

The I²C0 transfer clock frequency (f_{SCL}) is calculated using the following expression.

$$f_{SCL} = 1/(m \times T + t_r + t_f)$$

m = 12, 24, 48, 36, 54, 44, 86, 172, 132, 198 (see **Table 11-3 Selection Clock Setting**.)

T: 1/f_{xx}

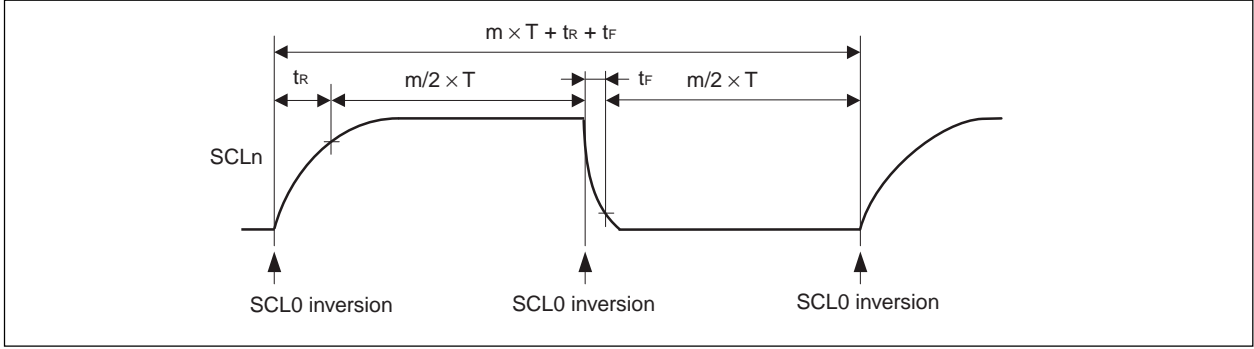
t_r: SCLn rise time

t_f: SCLn fall time

For example, the I²C0 transfer clock frequency (f_{SCL}) when f_{xx} = 16 MHz, m = 198, t_r = 200 ns, and t_f = 50 ns is calculated using the following expression.

$$f_{SCL} = 1/(198 \times 62.5 \text{ ns} + 200 \text{ ns} + 50 \text{ ns}) \cong 79.2 \text{ kHz}$$

Figure 11-7. I²C0 Transfer Clock Frequency (f_{SCL})



The selection clock is set using a combination of the SMC0, CL01, and CL00 bits of IIC clock select register 0 (IICCL0), the CLX0 bit of IIC function expansion register 0 (IICX0), and the IICCE01 and IICCE00 bits of IIC clock expansion register 0 (IICCE0).

★

Table 11-3. Selection Clock Setting

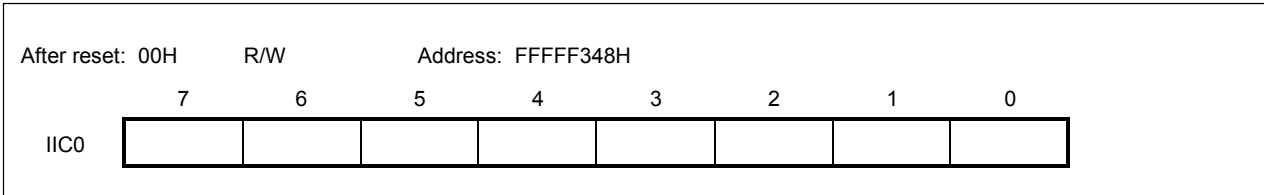
IICCE0		IICX0	IICCL0			Selection Clock (f _{xx} /m)	Settable Main Clock Frequency (f _{xx}) Range	Operation Mode
Bit 1	Bit 0	Bit 0	Bit 3	Bit 1	Bit 0			
IICCE01	IICCE00	CLX0	SMC0	CL01	CL00			
x	x	1	1	0	x	f _{xx} /12	4.0 MHz to 4.19 MHz	High-speed mode (SMC0 = 1)
x	x	0	1	0	x	f _{xx} /24	4.0 MHz to 8.38 MHz	
x	x	0	1	1	0	f _{xx} /48	8.0 MHz to 16.0 MHz	
0	1	0	1	1	1	f _{xx} /36	12.0 MHz to 13.4 MHz	
1	0	0	1	1	1	f _{xx} /54	16.0 MHz	
0	0	0	1	1	1	TM2 output/18	TM2 setting	Normal mode (SMC0 = 0)
x	x	0	0	0	0	f _{xx} /44	2.0 MHz to 4.19 MHz	
x	x	0	0	0	1	f _{xx} /86	4.19 MHz to 8.38 MHz	
x	x	0	0	1	0	f _{xx} /172	8.38 MHz to 16.0 MHz	
0	1	0	0	1	1	f _{xx} /132	12.0 MHz to 13.4 MHz	
1	0	0	0	1	1	f _{xx} /198	16.0 MHz	
0	0	0	0	1	1	TM2 output/66	TM2 setting	
Other than above						Setting prohibited		

Remarks 1. x: don't care

2. When the selection clock is output from a timer, the P30/TO2/TI2 pin does not need to be set in timer output mode.

(7) IIC shift register 0 (IIC0)

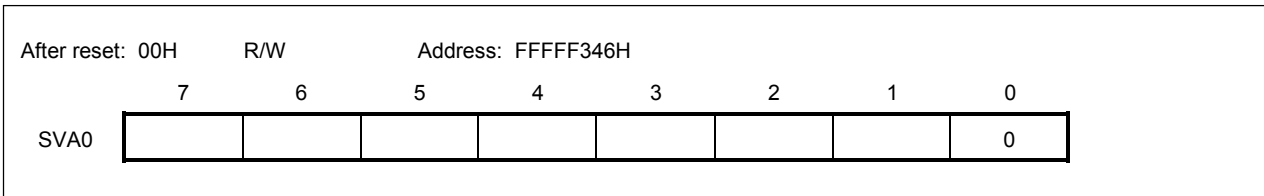
IIC0 is used for serial transmission/reception (shift operations) that are synchronized with the serial clock. It can be read from or written to in 8-bit units, but data should not be written to IIC0 during a data transfer.



(8) Slave address register 0 (SVA0)

SVA0 holds the I²C bus's slave addresses.

It can be read from or written to in 8-bit units, but bit 0 should be fixed to 0.



11.3.3 I²C bus mode functions

(1) Pin configuration

The serial clock pin (SCL0) and serial data bus pin (SDA0) are configured as follows.

SCL0This pin is used for serial clock I/O.

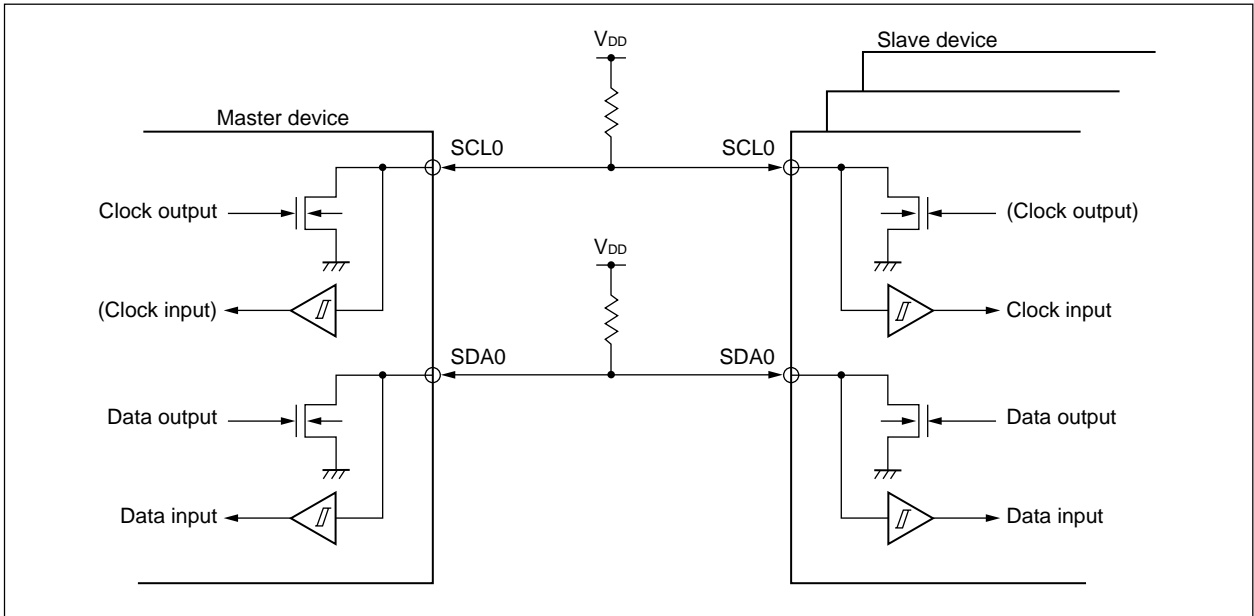
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

SDA0This pin is used for serial data I/O.

This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

Since outputs from the serial clock line and the serial data bus line are N-ch open-drain outputs, an external pull-up resistor is required.

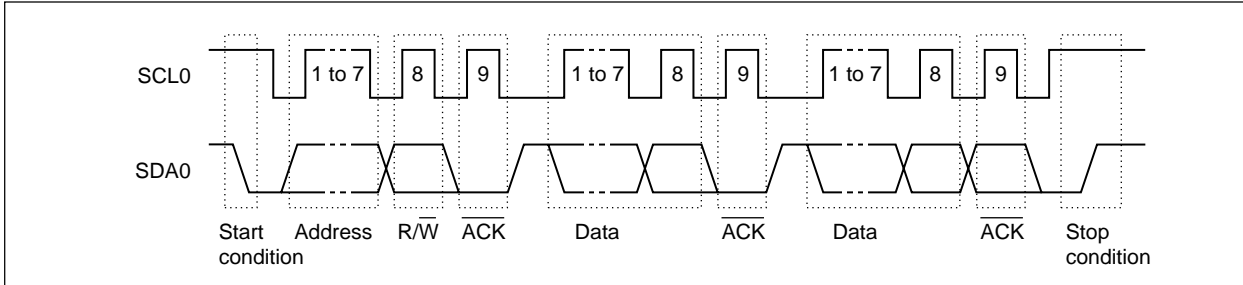
Figure 11-8. Pin Configuration Diagram



11.3.4 I²C bus definitions and control methods

The following section describes the I²C bus's serial data communication format and the signals used by the I²C bus. The transfer timing for the "start condition", "data", and "stop condition" output via the I²C bus's serial data bus is shown below.

Figure 11-9. I²C Bus's Serial Data Transfer Timing



The master device outputs the start condition, slave address, and stop condition.

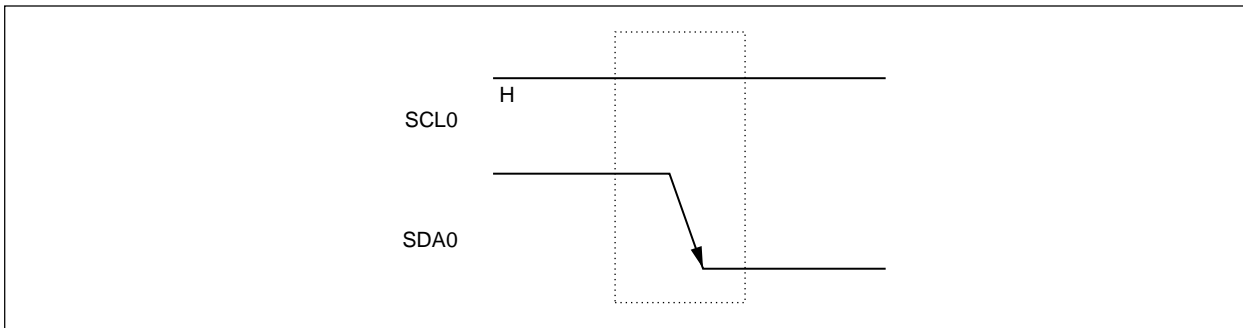
The acknowledge signal ($\overline{\text{ACK}}$) can be output by either the master or slave device (normally, it is output by the device that receives 8-bit data).

The serial clock (SCL0) is continuously output by the master device. However, in the slave device, SCL0's low-level period can be extended and a wait can be inserted.

(1) Start condition

A start condition is met when the SCL0 pin is at high level and the SDA0 pin changes from high level to low level. The start conditions for the SCL0 pin and SDA0 pin are signals that the master device outputs to the slave device when starting a serial transfer. The slave device includes hardware for detecting start conditions.

Figure 11-10. Start Conditions



A start condition is output when bit 1 (STT0) of IIC control register 0 (IICC0) is set to 1 after a stop condition has been detected (SPD0: Bit 0 = 1 in IIC status register 0 (IICS0)). When a start condition is detected, bit 1 (STD0) of IICS0 is set to 1.

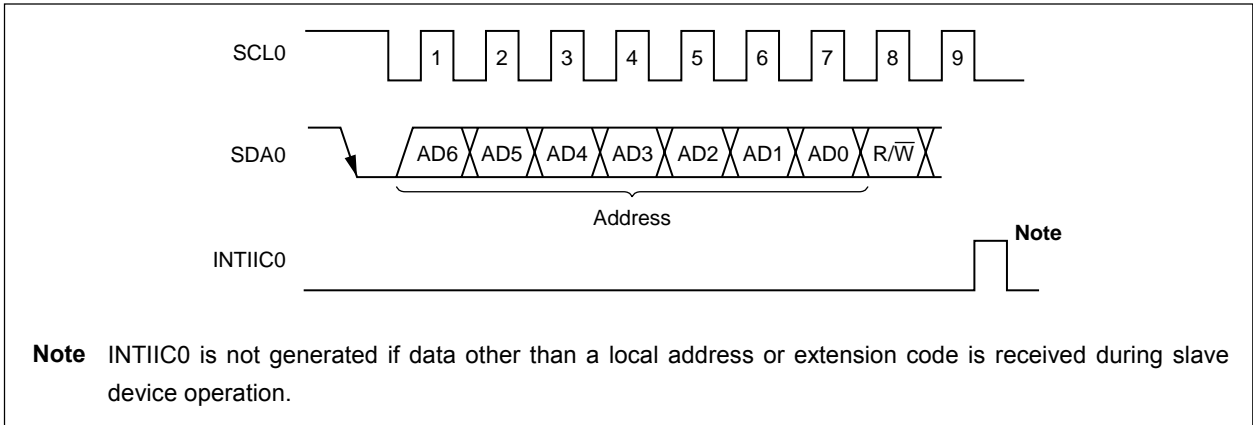
(2) Addresses

The 7 bits of data that follow the start condition are defined as an address.

An address is a 7-bit data segment that is output in order to select one of the slave devices that are connected to the master device via bus lines. Therefore, each slave device connected via the bus lines must have a unique address.

The slave devices include hardware that detects the start condition and checks whether or not the 7-bit data matches the data values stored in slave address register 0 (SVA0). If the 7-bit data matches the SVA0 values, the slave device is selected and communicates with the master device until the master device transmits a start condition or stop condition.

Figure 11-11. Address



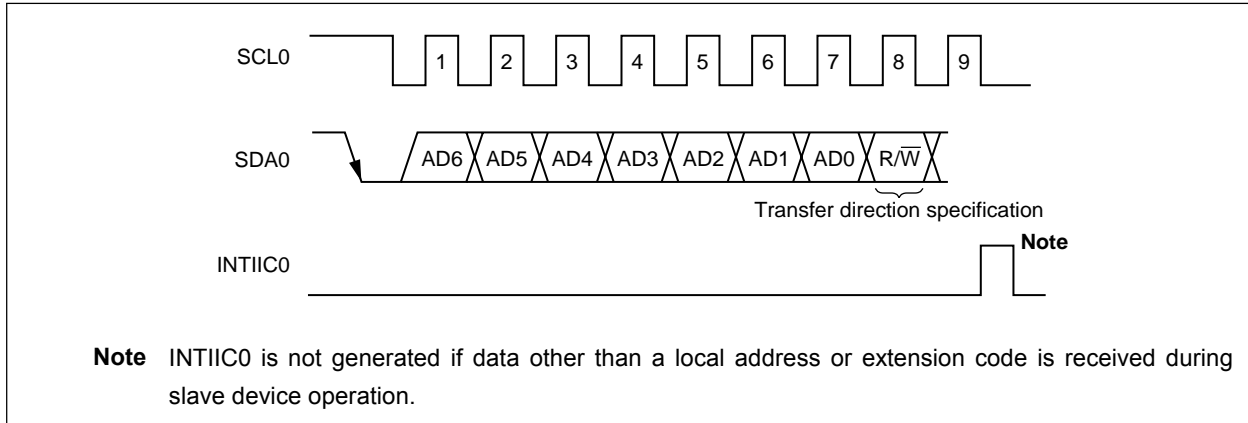
The slave address and the eighth bit, which specifies the transfer direction as described in **(3) Transfer direction specification** below, are written together to IIC shift register 0 (IIC0) and are then output. Received addresses are written to IIC0.

The slave address is assigned to the higher 7 bits of IIC0.

(3) Transfer direction specification

In addition to the 7-bit address data, the master device sends 1 bit that specifies the transfer direction. When this transfer direction specification bit has a value of 0, it indicates that the master device is transmitting data to a slave device. When the transfer direction specification bit has a value of 1, it indicates that the master device is receiving data from a slave device.

Figure 11-12. Transfer Direction Specification



(4) Acknowledge signal (\overline{ACK})

The acknowledge signal (\overline{ACK}) is used by the transmitting and receiving devices to confirm serial data reception.

The receiving device returns one \overline{ACK} signal for each 8 bits of data it receives. The transmitting device normally receives an \overline{ACK} signal after transmitting 8 bits of data. However, when the master device is the receiving device, it does not output an \overline{ACK} signal after receiving the final data to be transmitted. The transmitting device detects whether or not an \overline{ACK} signal is returned after it transmits 8 bits of data. When an \overline{ACK} signal is returned, the reception is judged as normal and processing continues. If the slave device does not return an \overline{ACK} signal, the master device outputs either a stop condition or a restart condition and then stops the current transmission. Failure to return an \overline{ACK} signal may be caused by the following two factors.

- (a) Reception was not correctly performed.
- (b) The final data was received.

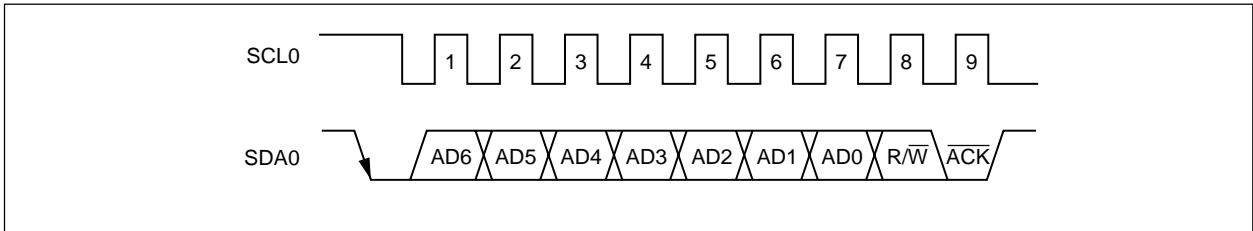
When the receiving device sets the SDA0 line to low level during the ninth clock, the \overline{ACK} signal becomes active (normal receive response).

When bit 2 (ACKE0) of IIC control register 0 (IICC0) is set to 1, automatic \overline{ACK} signal generation is enabled. Transmission of the eighth bit following the 7 address data bits causes bit 3 (TRC0) of IIC status register 0 (IICS0) to be set. When this TRC0 bit's value is 0, it indicates receive mode. Therefore, ACEK0 should be set to 1.

When the slave device is receiving (when TRC0 = 0), if the slave device does not need to receive any more data after receiving several bytes, setting ACEK0 to 0 will prevent the master device from starting transmission of the subsequent data.

Similarly, when the master device is receiving (when TRC0 = 0) and the subsequent data is not needed and when either a restart condition or a stop condition should therefore be output, setting ACEK0 to 0 will prevent the \overline{ACK} signal from being returned. This prevents the MSB data from being output via the SDA0 line (i.e., stops transmission) during transmission from the slave device.

Figure 11-13. \overline{ACK} Signal



When the local address is received, an $\overline{\text{ACK}}$ signal is automatically output in synchronization with the falling edge of the eighth clock of SCL0 regardless of the ACKE0 value. No $\overline{\text{ACK}}$ signal is output if the received address is not a local address.

The $\overline{\text{ACK}}$ signal output method during data reception is based on the wait timing setting, as described below.

When 8-clock wait is selected: The $\overline{\text{ACK}}$ signal is output at the falling edge of the eighth clock of SCL0 if ACKE0 is set to 1 before wait cancellation.

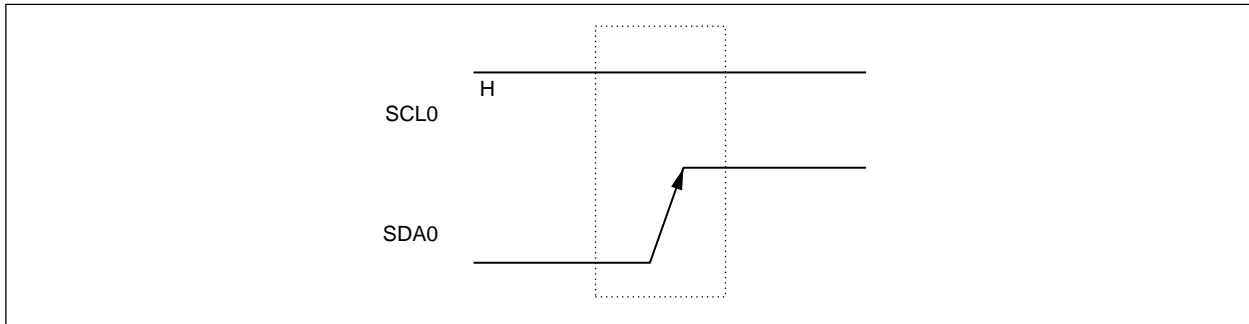
When 9-clock wait is selected: The $\overline{\text{ACK}}$ signal is automatically output at the falling edge of the eighth clock of SCL0 if ACKE0 has already been set to 1.

(5) Stop condition

When the SCL0 pin is at high level, changing the SDA0 pin from low level to high level generates a stop condition.

A stop condition is a signal that the master device outputs to the slave device when serial transfer has been completed. The slave device includes hardware that detects stop conditions.

Figure 11-14. Stop Condition



A stop condition is generated when bit 0 (SPT0) of IIC control register 0 (IICC0) is set to 1. When the stop condition is detected, bit 0 (SPD0) of IIC status register 0 (IICS0) is set to 1 and INTIIC0 is generated when bit 4 (SPIE0) of IICC0 is set to 1.

(6) Wait signal ($\overline{\text{WAIT}}$)

The wait signal ($\overline{\text{WAIT}}$) is used to notify the communication partner that a device (master or slave) is preparing to transmit or receive data (i.e., is in a wait state).

Setting the SCL0 pin to low level notifies the communication partner of the wait status. When the wait status has been canceled for both the master and slave devices, the next data transfer can begin.

Figure 11-15. Wait Signal (1/2)

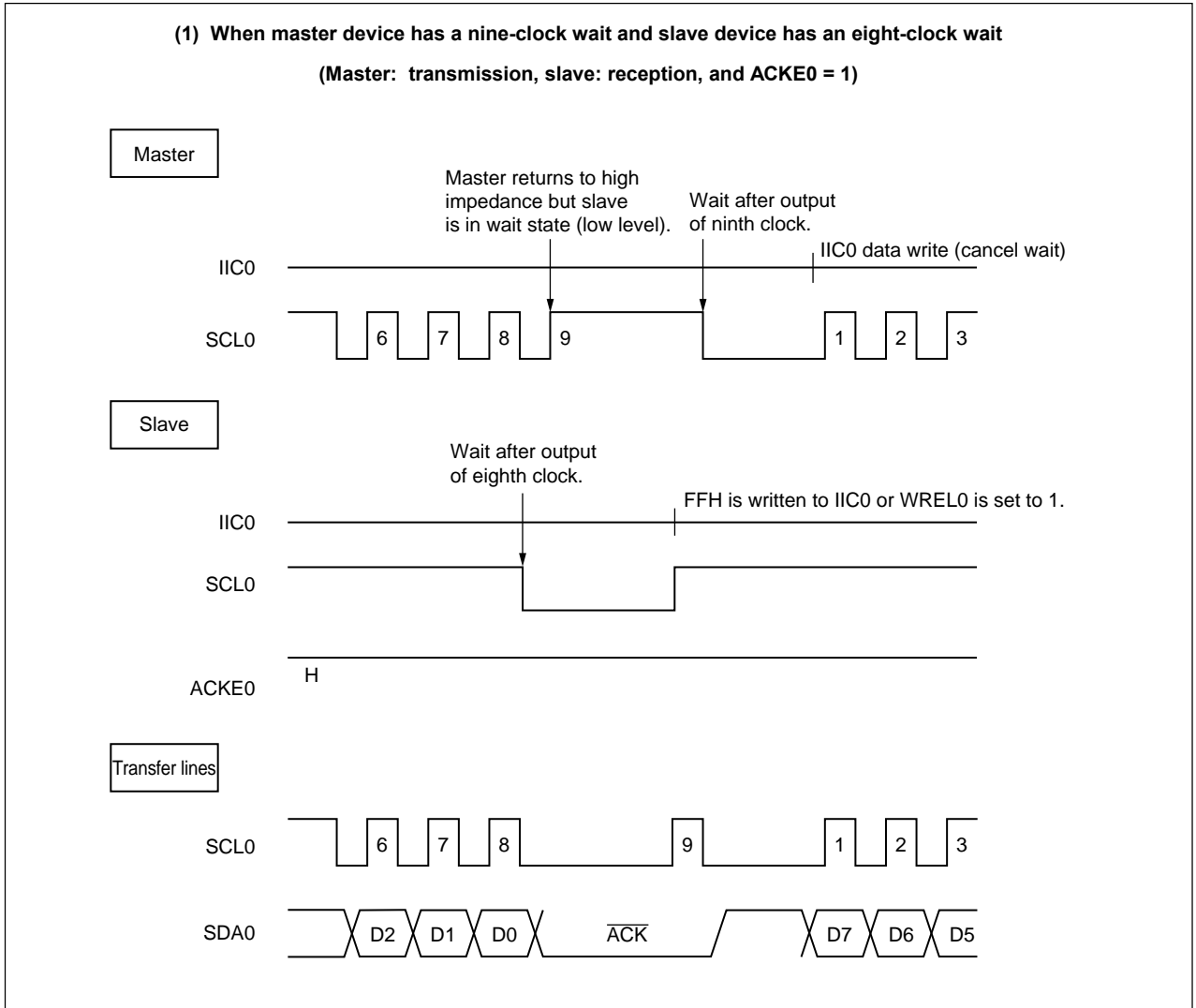
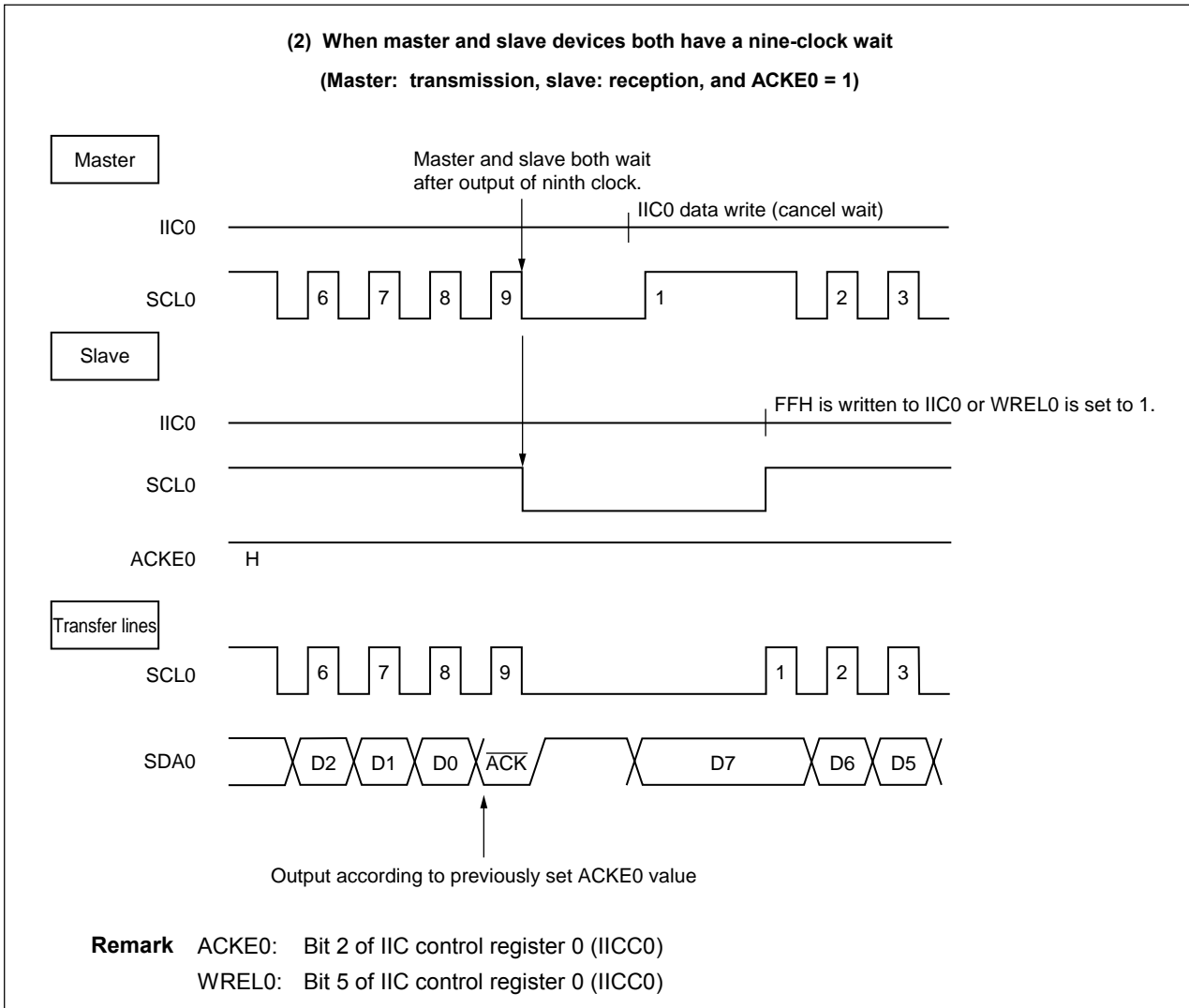


Figure 11-15. Wait Signal (2/2)



A wait may be automatically generated depending on the setting of bit 3 (WTIM0) of IIC control register 0 (IICC0). Normally, when bit 5 (WREL0) of IICC0 is set to 1 or when FFH is written to IIC shift register 0 (IIC0), the wait status is canceled and the transmitting side writes data to IIC0 to cancel the wait status. The master device can also cancel the wait status via either of the following methods.

- By setting bit 1 (STT0) of IICC0 to 1
- By setting bit 0 (SPT0) of IICC0 to 1

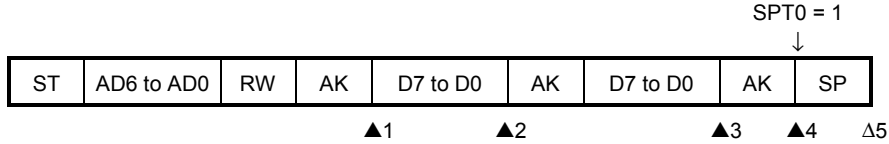
11.3.5 I²C interrupt request (INTIIC0)

The following shows the value of IIC status register 0 (IICS0) at the INTIIC0 interrupt request generation timing and at the INTIIC0 interrupt timing.

(1) Master device operation

(a) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)

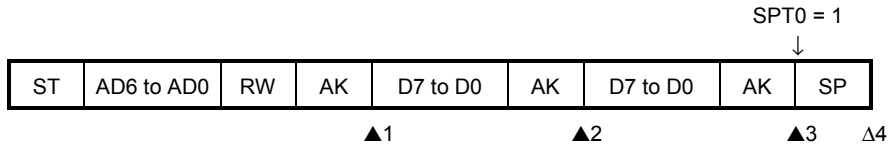
<1> When WTIM0 = 0



- ▲1: IICS0 = 10XXX110B
- ▲2: IICS0 = 10XXX000B
- ▲3: IICS0 = 10XXX000B (WTIM0 = 0)
- ▲4: IICS0 = 10XXXX00B
- Δ 5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1

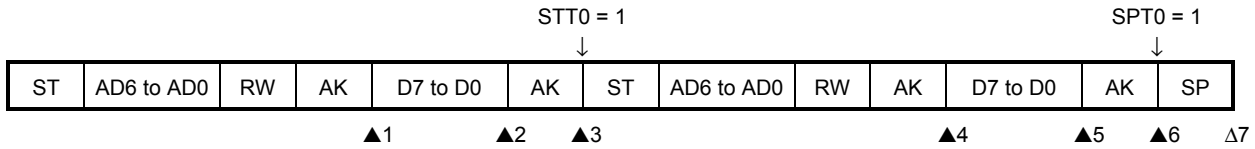


- ▲1: IICS0 = 10XXX110B
- ▲2: IICS0 = 10XXX100B
- ▲3: IICS0 = 10XXXX00B
- Δ 4: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

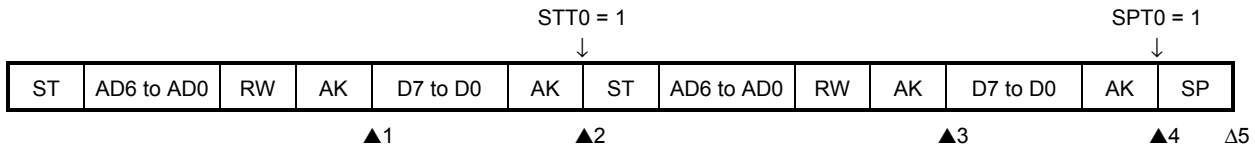
<1> When WTIM0 = 0



- ▲1: IICS0 = 10XXX110B
- ▲2: IICS0 = 10XXX000B (WTIM0 = 1)
- ▲3: IICS0 = 10XXX000B (WTIM0 = 0)
- ▲4: IICS0 = 10XXX110B (WTIM0 = 0)
- ▲5: IICS0 = 10XXX000B (WTIM0 = 1)
- ▲6: IICS0 = 10XXX000B
- Δ7: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1

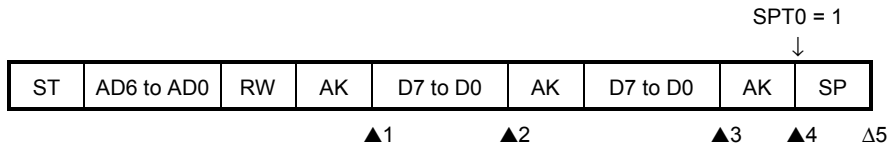


- ▲1: IICS0 = 10XXX110B
- ▲2: IICS0 = 10XXX000B
- ▲3: IICS0 = 10XXX110B
- ▲4: IICS0 = 10XXX000B
- Δ5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(c) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

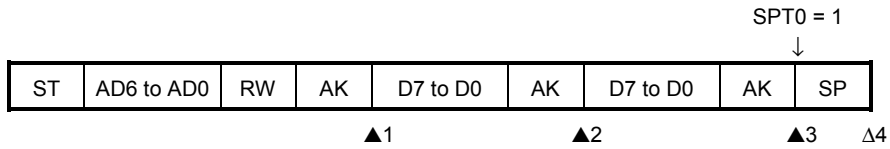
<1> When WTIM0 = 0



- ▲1: IICS0 = 1010X110B
- ▲2: IICS0 = 1010X000B
- ▲3: IICS0 = 1010X000B (WTIM0 = 1)
- ▲4: IICS0 = 1010XX00B
- Δ 5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1



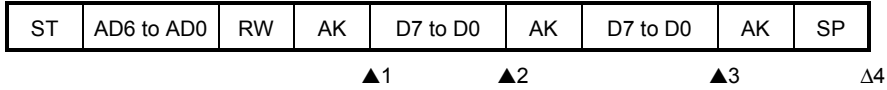
- ▲1: IICS0 = 1010X110B
- ▲2: IICS0 = 1010X100B
- ▲3: IICS0 = 1010XX00B
- Δ 4: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(2) Slave device operation (when receiving slave address data (matches with SVA0))

(a) Start ~ Address ~ Data ~ Data ~ Stop

<1> When WTIM0 = 0



▲1: IICS0 = 0001X110B

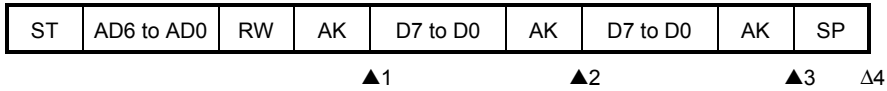
▲2: IICS0 = 0001X000B

▲3: IICS0 = 0001X000B

△4: IICS0 = 00000001B

Remark ▲: Always generated
 △: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1



▲1: IICS0 = 0001X110B

▲2: IICS0 = 0001X100B

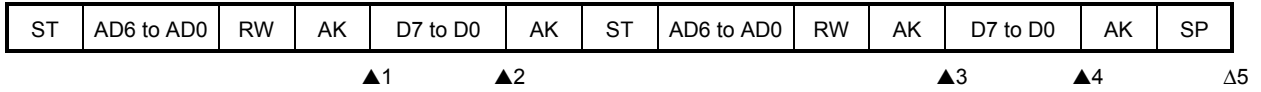
▲3: IICS0 = 0001XX00B

△4: IICS0 = 00000001B

Remark ▲: Always generated
 △: Generated only when SPIE0 = 1
 X: don't care

(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

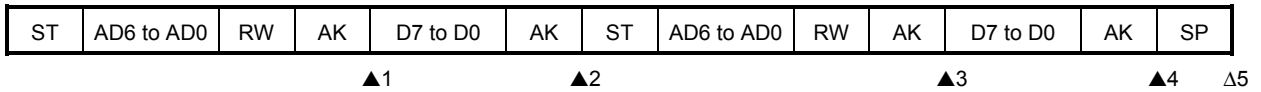
<1> When WTIM0 = 0 (after restart, matches with SVA0)



- ▲1: IICS0 = 0001X110B
- ▲2: IICS0 = 0001X000B
- ▲3: IICS0 = 0001X110B
- ▲4: IICS0 = 0001X000B
- Δ 5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1 (after restart, matches with SVA0)

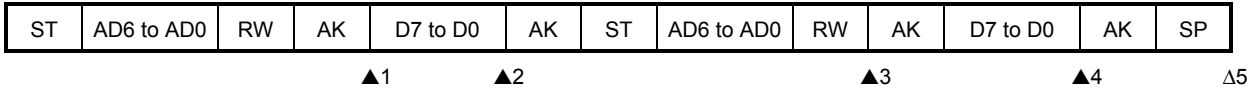


- ▲1: IICS0 = 0001X110B
- ▲2: IICS0 = 0001XX00B
- ▲3: IICS0 = 0001X110B
- ▲4: IICS0 = 0001XX00B
- Δ 5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(c) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIM0 = 0 (after restart, extension code reception)



▲1: IICS0 = 0001X110B

▲2: IICS0 = 0001X000B

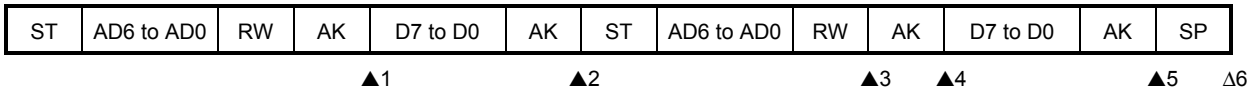
▲3: IICS0 = 0010X010B

▲4: IICS0 = 0010X000B

Δ 5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1 (after restart, extension code reception)



▲1: IICS0 = 0001X110B

▲2: IICS0 = 0001XX00B

▲3: IICS0 = 0010X010B

▲4: IICS0 = 0010X110B

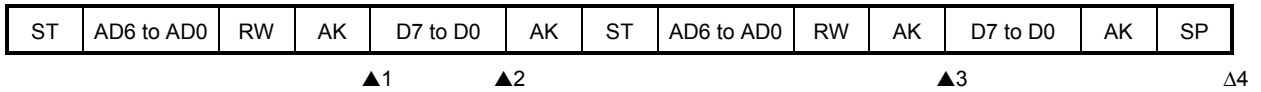
▲5: IICS0 = 0010XX00B

Δ 6: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(d) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

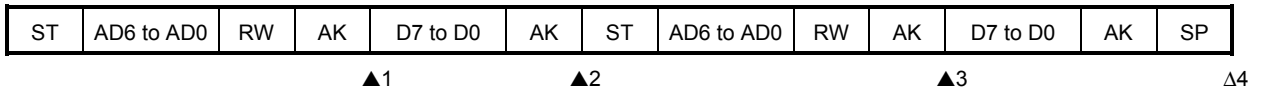
<1> When WTIM0 = 0 (after restart, mismatches with address (= not extension code))



- ▲1: IICS0 = 0001X110B
- ▲2: IICS0 = 0001X000B
- ▲3: IICS0 = 00000X10B
- Δ 4: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1 (after restart, mismatches with address (= not extension code))



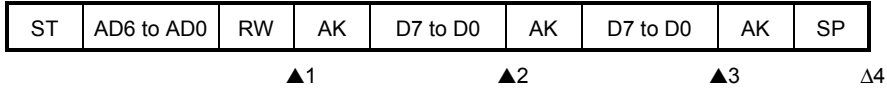
- ▲1: IICS0 = 0001X110B
- ▲2: IICS0 = 0001XX00B
- ▲3: IICS0 = 00000X10B
- Δ 4: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(3) Slave device operation (when receiving extension code)

(a) Start ~ Code ~ Data ~ Data ~ Stop

<1> When WTIM0 = 0



▲1: IICS0 = 0010X010B

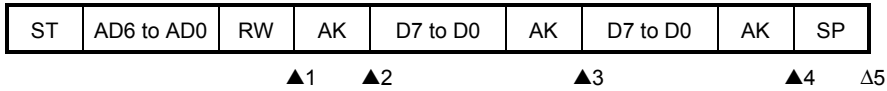
▲2: IICS0 = 0010X000B

▲3: IICS0 = 0010X000B

Δ 4: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1



▲1: IICS0 = 0010X010B

▲2: IICS0 = 0010X110B

▲3: IICS0 = 0010X100B

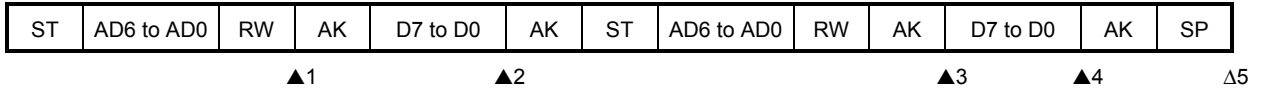
▲4: IICS0 = 0010XX00B

Δ 5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(b) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

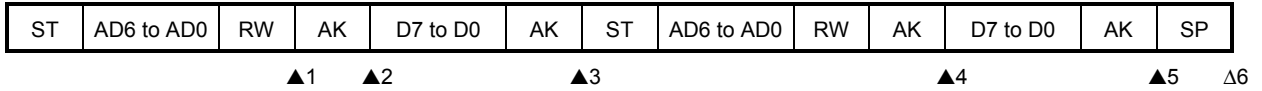
<1> When WTIM0 = 0 (after restart, matches with SVA0)



- ▲1: IICS0 = 0010X010B
- ▲2: IICS0 = 0010X000B
- ▲3: IICS0 = 0001X110B
- ▲4: IICS0 = 0001X000B
- Δ 5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1 (after restart, matches with SVA0)

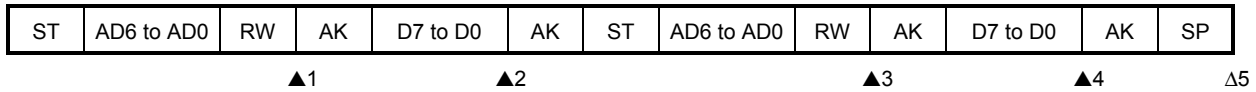


- ▲1: IICS0 = 0010X010B
- ▲2: IICS0 = 0010X110B
- ▲3: IICS0 = 0010XX00B
- ▲4: IICS0 = 0001X110B
- ▲5: IICS0 = 0001XX00B
- Δ 6: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(c) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When WTIM0 = 0 (after restart, extension code reception)



▲1: IICS0 = 0010X010B

▲2: IICS0 = 0010X000B

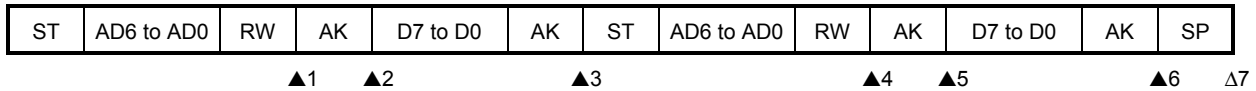
▲3: IICS0 = 0010X010B

▲4: IICS0 = 0010X000B

Δ5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1 (after restart, extension code reception)



▲1: IICS0 = 0010X010B

▲2: IICS0 = 0010X110B

▲3: IICS0 = 0010XX00B

▲4: IICS0 = 0010X010B

▲5: IICS0 = 0010X110B

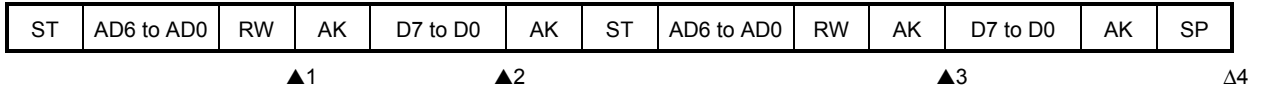
▲6: IICS0 = 0010XX00B

Δ7: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(d) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

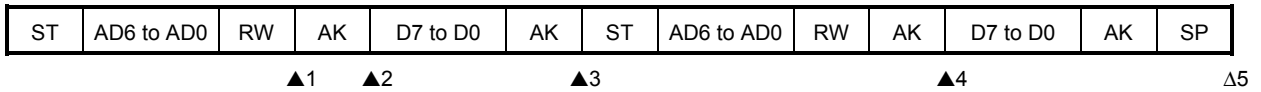
<1> When WTIM0 = 0 (after restart, mismatches with address (= not extension code))



- ▲1: IICS0 = 0010X010B
- ▲2: IICS0 = 0010X000B
- ▲3: IICS0 = 00000X10B
- Δ 4: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1 (after restart, mismatches with address (= not extension code))

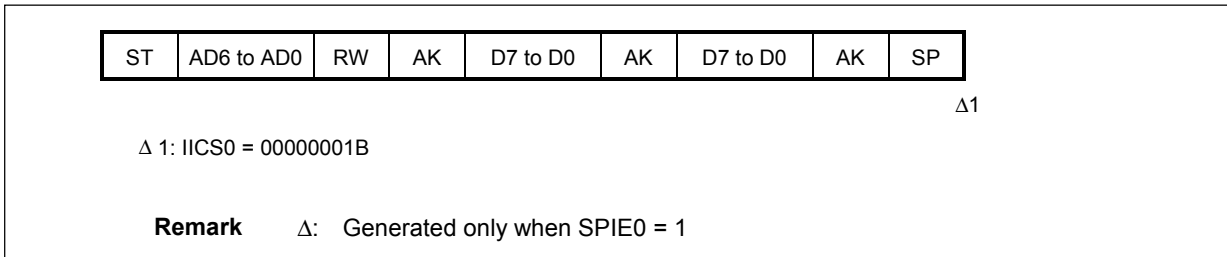


- ▲1: IICS0 = 0010X010B
- ▲2: IICS0 = 0010X110B
- ▲3: IICS0 = 0010XX00B
- ▲4: IICS0 = 00000X10B
- Δ 5: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

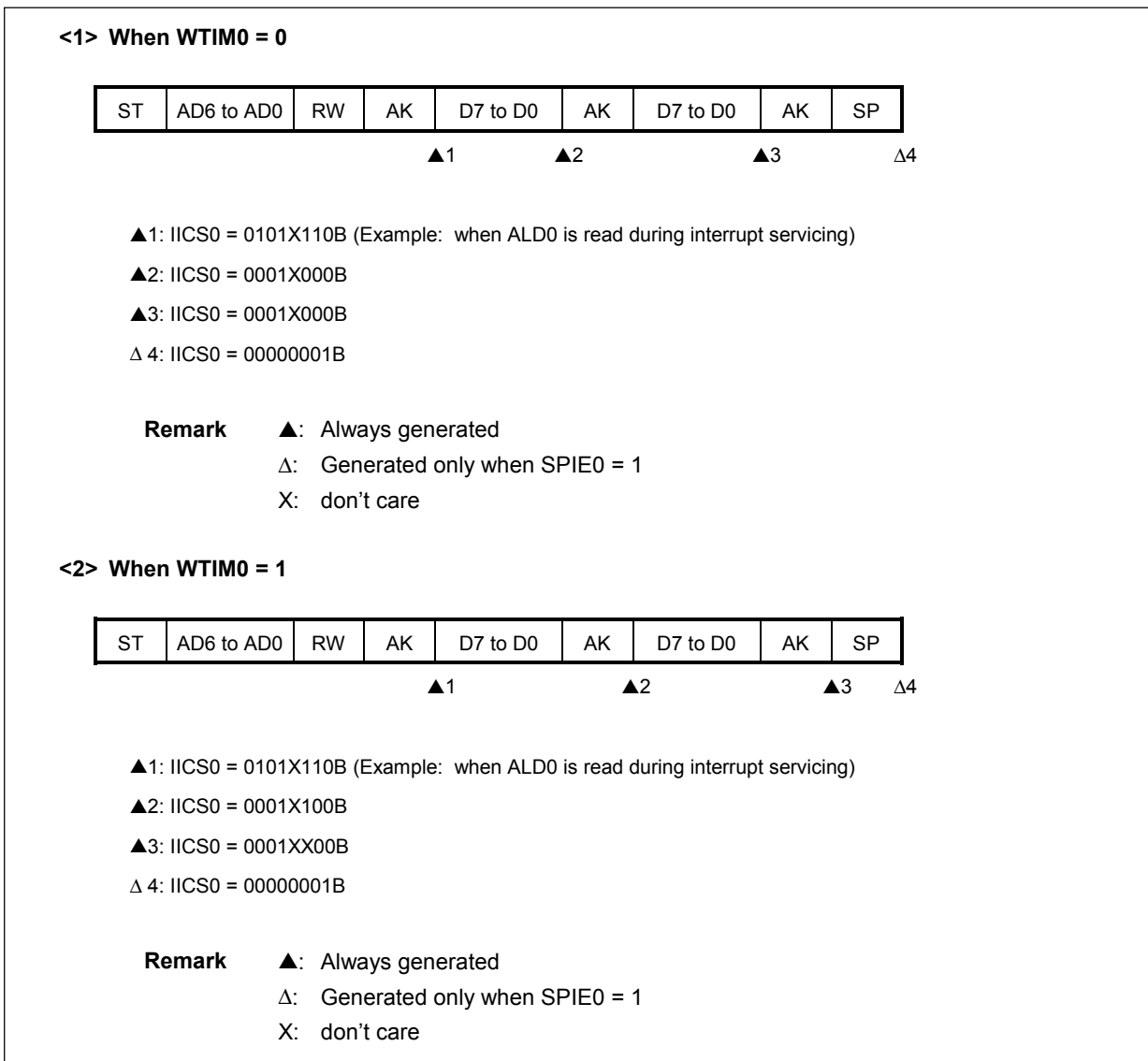
(4) Operation without communication

(a) Start ~ Code ~ Data ~ Data ~ Stop



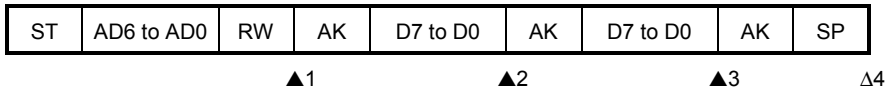
(5) Arbitration loss operation (operation as slave after arbitration loss)

(a) When arbitration loss occurs during transmission of slave address data



(b) When arbitration loss occurs during transmission of extension code

<1> When WTIM0 = 0



▲1: IICS0 = 0110X010B (Example: when ALD0 is read during interrupt servicing)

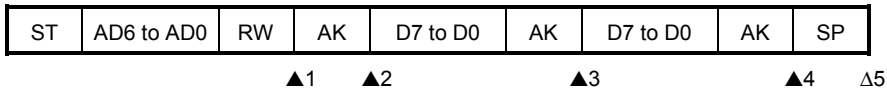
▲2: IICS0 = 0010X000B

▲3: IICS0 = 0010X000B

Δ 4: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

<2> When WTIM0 = 1



▲1: IICS0 = 0110X010B (Example: when ALD0 is read during interrupt servicing)

▲2: IICS0 = 0010X110B

▲3: IICS0 = 0010X100B

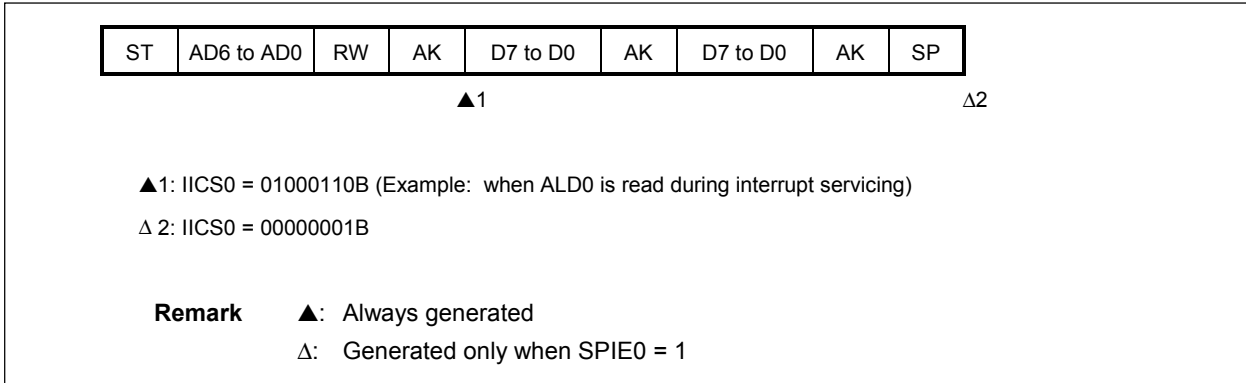
▲4: IICS0 = 0010XX00B

Δ 5: IICS0 = 00000001B

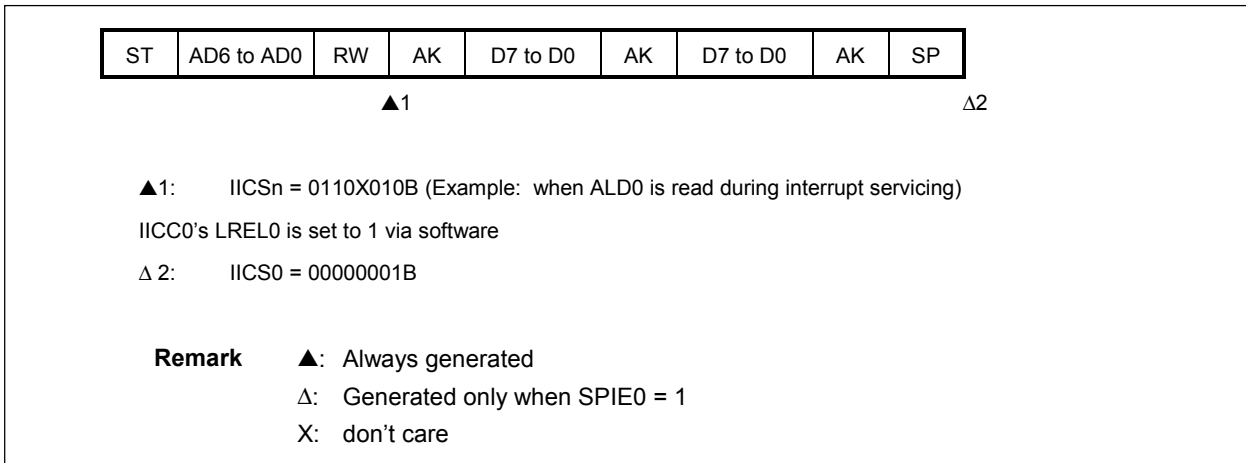
Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care

(6) Operation when arbitration loss occurs (no communication after arbitration loss)

(a) When arbitration loss occurs during transmission of slave address data

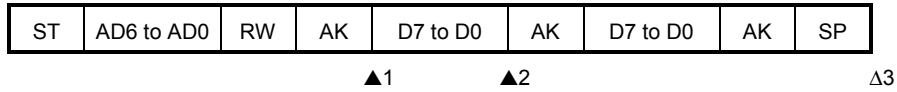


(b) When arbitration loss occurs during transmission of extension code



(c) When arbitration loss occurs during data transfer

<1> When **WTIM0 = 0**



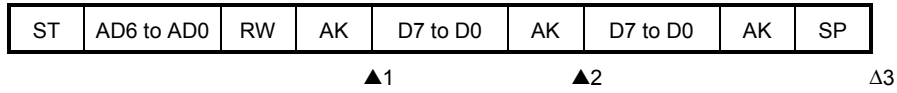
▲1: IICS0 = 10001110B

▲2: IICS0 = 01000000B (Example: when ALD0 is read during interrupt servicing)

Δ 3: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1

<2> When **WTIM0 = 1**



▲1: IICS0 = 10001110B

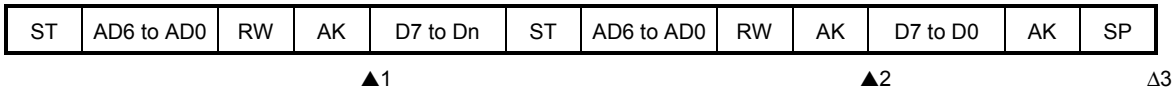
▲2: IICS0 = 01000100B (Example: when ALD0 is read during interrupt servicing)

Δ 3: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1

(d) When loss occurs due to restart condition during data transfer

<1> Not extension code (Example: mismatches with SVA0)



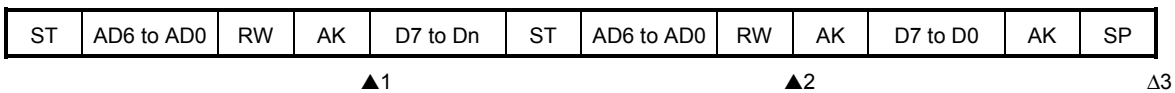
▲1: IICS0 = 1000X110B

▲2: IICS0 = 01000110B (Example: when ALD0 is read during interrupt servicing)

Δ 3: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care
 Dn = D6 to D0

<2> Extension code



▲1: IICS0 = 1000X110B

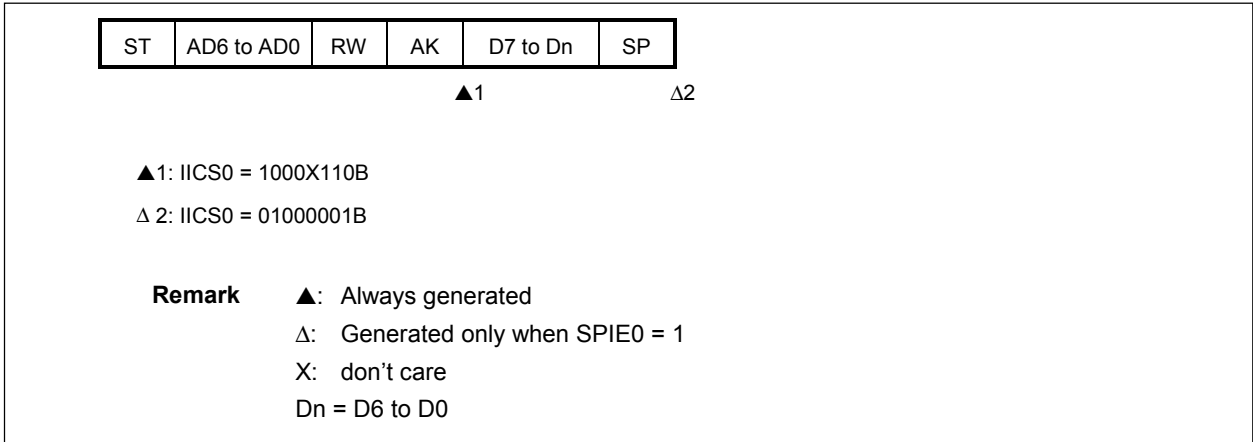
▲2: IICS0 = 0110X010B (Example: when ALD0 is read during interrupt servicing)

IICC0's LREL0 is set to 1 via software

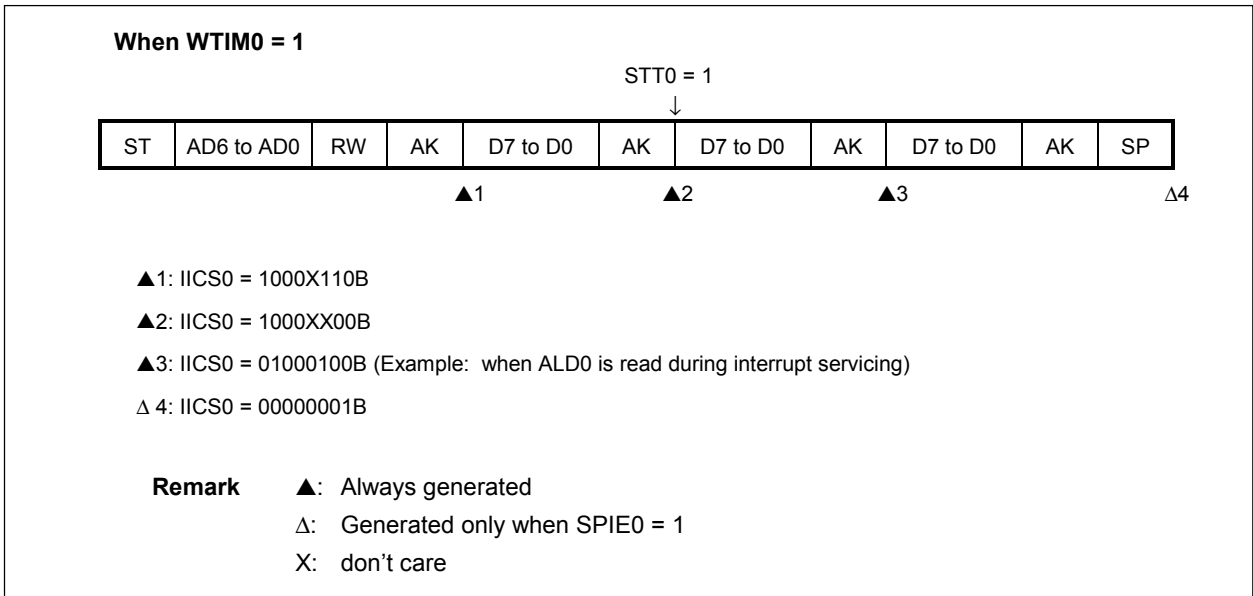
Δ 3: IICS0 = 00000001B

Remark ▲: Always generated
 Δ: Generated only when SPIE0 = 1
 X: don't care
 Dn = D6 to D0

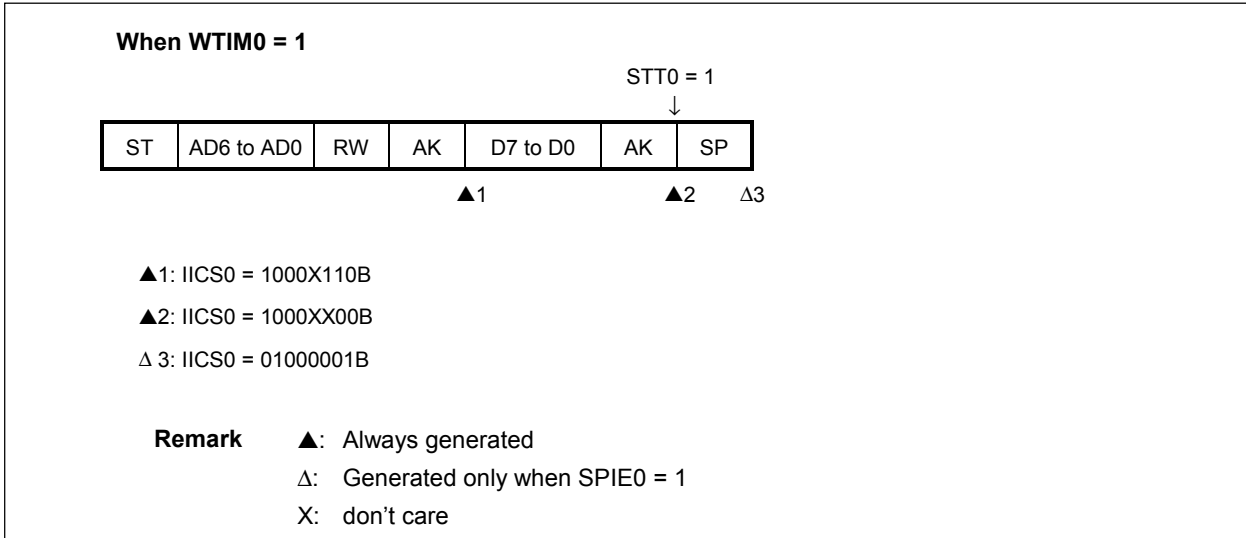
(e) When loss occurs due to stop condition during data transfer



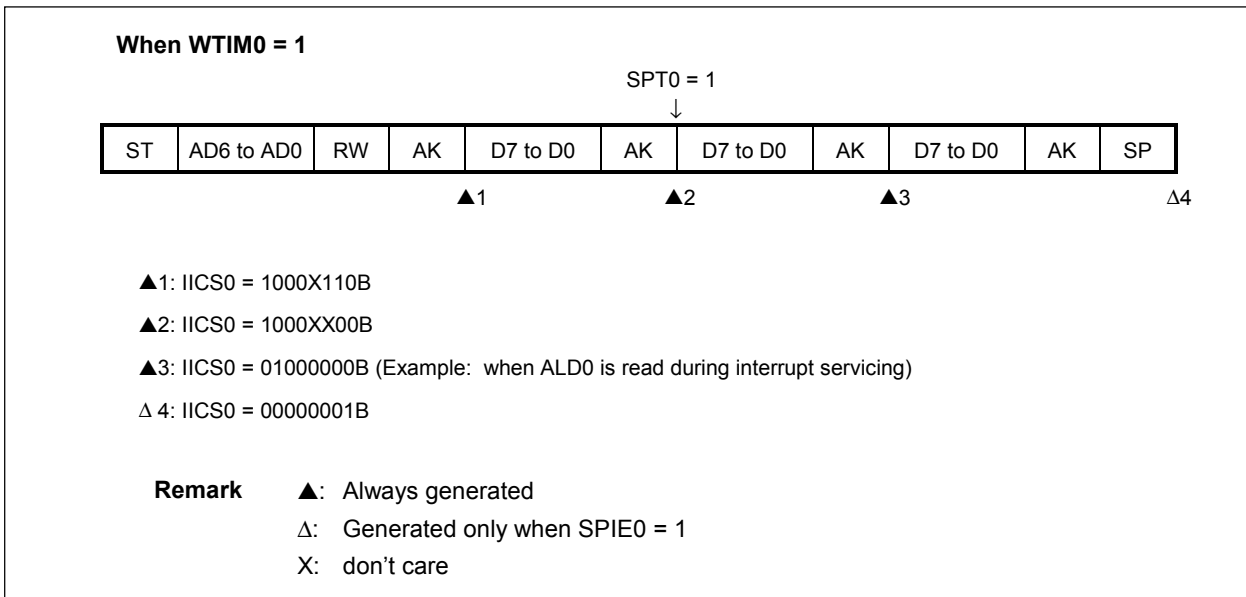
(f) When arbitration loss occurs due to low-level data when attempting to generate a restart condition



(g) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition



(h) When arbitration loss occurs due to low-level data when attempting to generate a stop condition



11.3.6 Interrupt request (INTIIC0) generation timing and wait control

The setting of bit 3 (WTIM0) in IIC control register 0 (IICC0) determines the timing by which INTIIC0 is generated and the corresponding wait control, as shown below.

Table 11-4. INTIIC0 Generation Timing and Wait Control

WTIM0	During Slave Device Operation			During Master Device Operation		
	Address	Data Reception	Data Transmission	Address	Data Reception	Data Transmission
0	9 ^{Notes 1, 2}	8 ^{Note 2}	8 ^{Note 2}	9	8	8
1	9 ^{Notes 1, 2}	9 ^{Note 2}	9 ^{Note 2}	9	9	9

- Notes 1.** The slave device's INTIIC0 signal and wait period occurs at the falling edge of the ninth clock only when there is a match with the address set to slave address register 0 (SVA0). At this point, \overline{ACK} is output regardless of the value set to bit 2 (ACKE0) of IICC0. For a slave device that has received an extension code, INTIIC0 occurs at the falling edge of the eighth clock.
- 2.** If the received address does not match the contents of slave address register 0 (SVA0), neither INTIIC0 nor a wait occurs.

Remark The numbers in the table indicate the number of the serial clock's clock signals. Interrupt requests and wait control are both synchronized with the falling edge of these clock signals.

(1) During address transmission/reception

- Slave device operation: The interrupt and wait timing are determined regardless of the WTIM0 bit.
- Master device operation: The interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIM0 bit.

(2) During data reception

- Master/slave device operation: The interrupt and wait timing are determined according to the WTIM0 bit.

(3) During data transmission

- Master/slave device operation: The interrupt and wait timing are determined according to the WTIM0 bit.

(4) Wait cancelation method

The four wait cancelation methods are as follows.

- By setting bit 5 (WREL0) of IIC control register 0 (IICC0) to 1
- By writing to the IIC shift register 0 (IIC0)
- By start condition setting (bit 1 (STT0) of IIC control register 0 (IICC0) = 1)
- By stop condition setting (bit 0 (SPT0) of IIC control register 0 (IICC0) = 1)

When an 8-clock wait has been selected (WTIM0 = 0), the output level of \overline{ACK} must be determined prior to wait cancelation.

(5) Stop condition detection

INTIIC0 is generated when a stop condition is detected.

11.3.7 Address match detection method

When in I²C bus mode, the master device can select a particular slave device by transmitting the corresponding slave address.

Address match detection is performed automatically by hardware. An interrupt request (INTIIC0) occurs when a local address has been set to slave address register 0 (SVA0) and when the address set to SVA0 matches the slave address sent by the master device, or when an extension code has been received.

11.3.8 Error detection

In I²C bus mode, the status of the serial data bus (SDA0) during data transmission is captured by IIC shift register 0 (IIC0) of the transmitting device, so the IIC0 data prior to transmission can be compared with the transmitted IIC0 data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match.

11.3.9 Extension code

(1) When the higher 4 bits of the receive address are either 0000 or 1111, the extension code flag (EXC0) is set for extension code reception and an interrupt request (INTIIC0) is issued at the falling edge of the eighth clock. The local address stored in slave address register 0 (SVA0) is not affected.

(2) If 11110xx0 is set to SVA0 by a 10-bit address transfer and 11110xx0 is transferred from the master device, the results are as follows. Note that INTIIC0 occurs at the falling edge of the eighth clock.

- Higher four bits of data match: EXC0 = 1^{Note}
- Seven bits of data match: COI0 = 1^{Note}

Note EXC0: Bit 5 of IIC status register 0 (IICS0)
 COI0: Bit 4 of IIC status register 0 (IICS0)

(3) Since the processing after the interrupt request occurs differs according to the data that follows the extension code, such processing is performed by software.

For example, when operation as a slave is not desired after the extension code is received, set bit 6 (LREL0) of IIC control register 0 (IICC0) to 1 and the CPU will enter the next communication wait state.

Table 11-5. Extension Code Bit Definitions

Slave address	R/W bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	X	CBUS address
0000 010	X	Address that is reserved for different bus format
1111 0xx	X	10-bit slave address specification

11.3.10 Arbitration

When several master devices simultaneously output a start condition (when STT0 is set to 1 before STD0 is set to 1^{Note}), communication among the master devices is performed as the number of clocks is adjusted until the data differs. This kind of operation is called arbitration.

When one of the master devices loses in arbitration, an arbitration loss flag (ALD0) in IIC status register 0 (IICS0) is set via the timing by which the arbitration loss occurred, and the SCL0 and SDA0 lines are both set to high impedance, which releases the bus.

The arbitration loss is detected based on the timing of the next interrupt request (the eighth or ninth clock, when a stop condition is detected, etc.) and the ALD0 = 1 setting that has been made by software.

For details of interrupt request timing, see **11.3.5 I²C interrupt request (INTIIC0)**.

Note STD0: Bit 1 of IIC status register 0 (IICS0)
STT0: Bit 1 of IIC control register 0 (IICC0)

Figure 11-16. Arbitration Timing Example

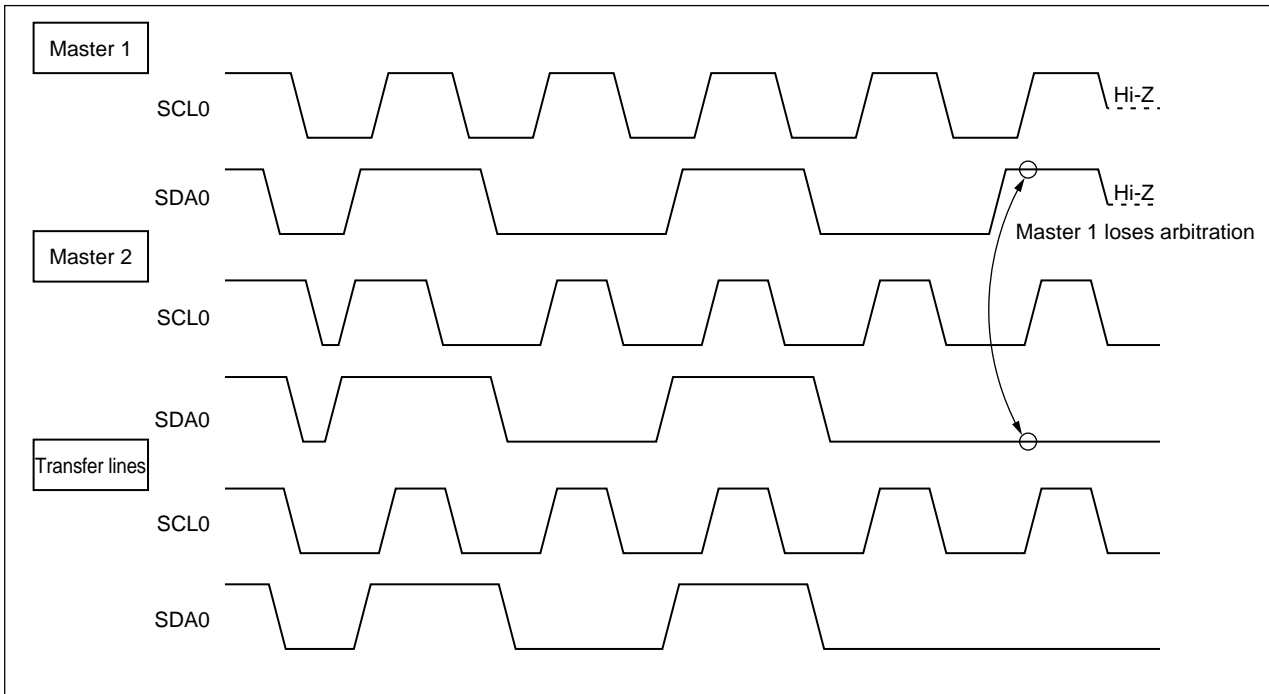


Table 11-6. Status During Arbitration and Interrupt Request Generation Timing

Status During Arbitration	Interrupt Request Generation Timing
During address transmission	At falling edge of eighth or ninth clock following byte transfer ^{Note 1}
Read/write data after address transmission	
During extension code transmission	
Read/write data after extension code transmission	
During data transmission	
During ACK signal transfer period after data transmission	
When restart condition is detected during data transfer	
When stop condition is detected during data transfer	When stop condition is output (when SPIE0 = 1) ^{Note 2}
When data is at low level while attempting to output a restart condition	At falling edge of eighth or ninth clock following byte transfer ^{Note 1}
When stop condition is detected while attempting to output a restart condition	When stop condition is output (when SPIE0 = 1) ^{Note 2}
When data is at low level while attempting to output a stop condition	At falling edge of eighth or ninth clock following byte transfer ^{Note 1}
When SCL0 is at low level while attempting to output a restart condition	

- Notes 1.** When WTIM0 (bit 3 of IIC control register 0 (IICC0)) = 1, an interrupt request occurs at the falling edge of the ninth clock. When WTIM0 = 0 and the extension code's slave address is received, an interrupt request occurs at the falling edge of the eighth clock.
- 2.** When there is a possibility that arbitration will occur, set SPIE0 = 1 for master device operation.

Remark SPIE0: Bit 5 of IIC control register 0 (IICC0)

11.3.11 Wakeup function

The I²C bus slave function is a function that generates an interrupt request (INTIIC0) when a local address and extension code have been received.

This function makes processing more efficient by preventing unnecessary interrupt requests from occurring when addresses do not match.

When a start condition is detected, wakeup standby mode is set. This wakeup standby mode is in effect while addresses are transmitted due to the possibility that an arbitration loss may change the master device (which has output a start condition) to a slave device.

However, when a stop condition is detected, bit 5 (SPIE0) of IIC control register 0 (IICC0) is set regardless of the wakeup function, and this determines whether interrupt requests are enabled or disabled.

11.3.12 Communication reservation

To start master device communications when not currently using a bus, a communication reservation can be made to enable transmission of a start condition when the bus is released. There are two modes under which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled (\overline{ACK} is not returned and the bus was released when bit 6 (LREL0) of IIC control register 0 (IICC0) was set to “1”).

If bit 1 (STT0) of IICC0 is set while the bus is not used, a start condition is automatically generated and the wait status is set after the bus is released (after a stop condition is detected).

When the bus release is detected (when a stop condition is detected), writing to IIC shift register 0 (IIC0) causes the master’s address transfer to start. At this point, bit 4 (SPIE0) of IICC0 should be set.

When STT0 has been set, the operation mode (as start condition or as communication reservation) is determined according to the bus status.

If the bus has been released a start condition is generated
 If the bus has not been released (standby mode)..... communication reservation

To detect which operation mode has been determined for STT0, set STT0, wait for the wait period, then check the MSTS0 (bit 7 of IIC status register 0 (IICS0)).

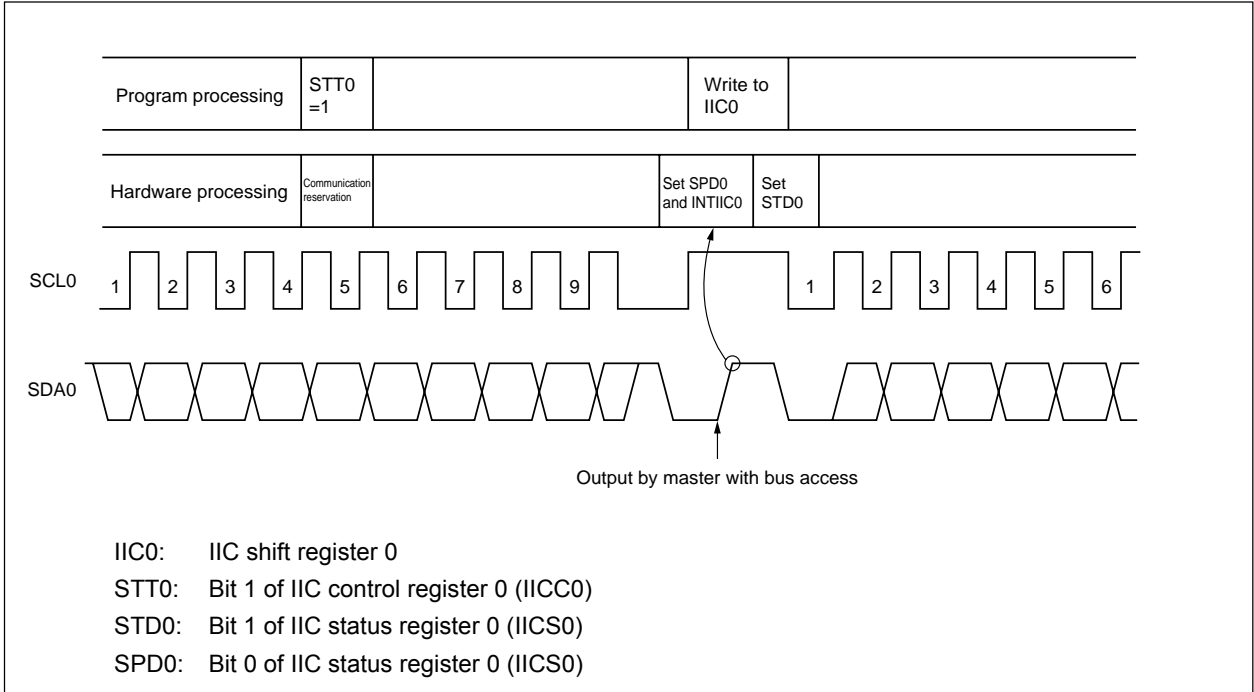
Wait periods, which should be set via software, are listed in Table 11-7. These wait periods can be set via the settings for bits 3, 1, and 0 (SMC0, CL01, and CL00) in IIC clock selection register 0 (IICCL0).

Table 11-7. Wait Periods

SMC0	CL01	CL00	Wait Period
0	0	0	26 clocks
0	0	1	46 clocks
0	1	0	92 clocks
0	1	1	37 clocks
1	0	0	16 clocks
1	0	1	
1	1	0	32 clocks
1	1	1	13 clocks

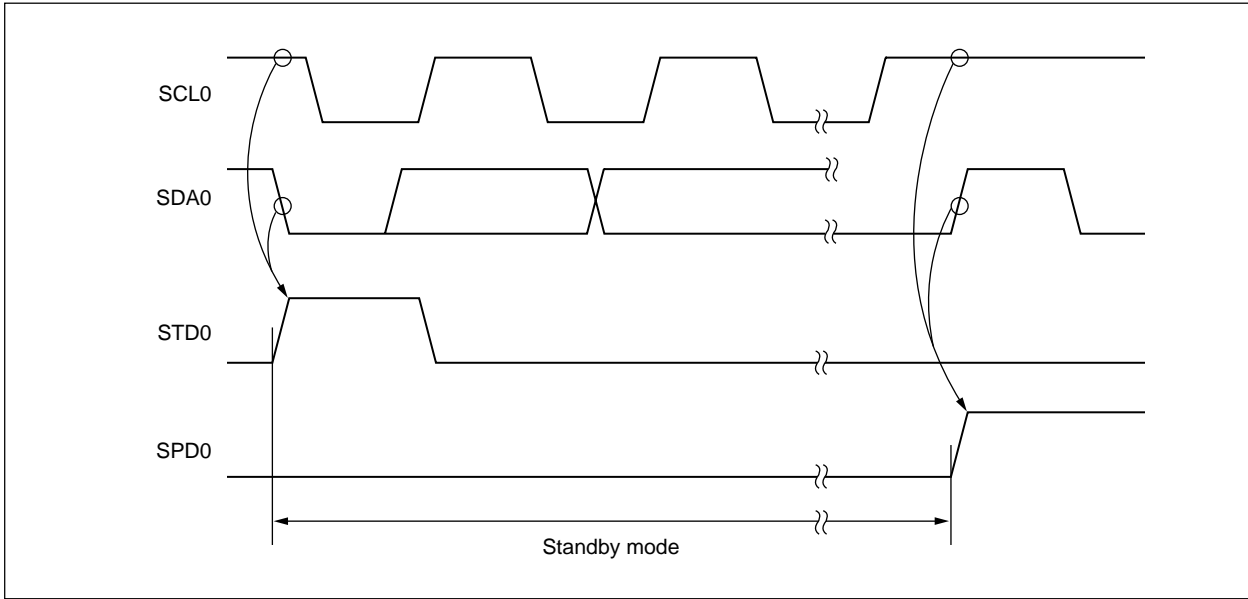
The communication reservation timing is shown below.

Figure 11-17. Communication Reservation Timing



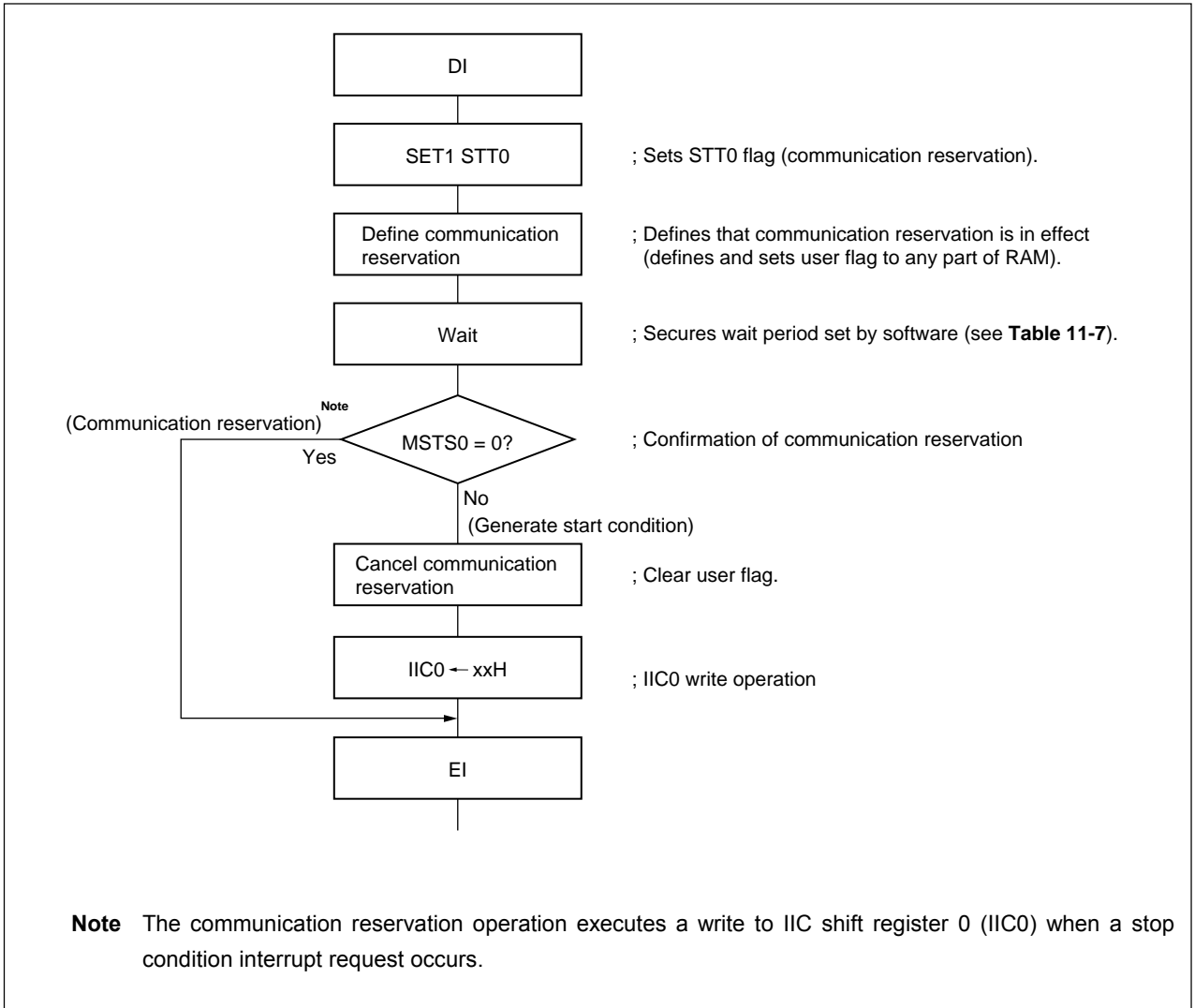
Communication reservations are accepted via the following timing. After bit 1 (STD0) of IIC status register 0 (IICS0) is set to 1, a communication reservation can be made by setting bit 1 (STT0) of IIC control register 0 (IICC0) to 1 before a stop condition is detected.

Figure 11-18. Timing for Accepting Communication Reservations



The communication reservation flow chart is illustrated below.

Figure 11-19. Communication Reservation Flow Chart



11.3.13 Cautions

After a reset, when changing from a mode in which no stop condition has been detected (the bus has not been released) to a master device communication mode, first generate a stop condition to release the bus, then perform master device communication.

When using multiple masters, it is not possible to perform master device communication when the bus has not been released (when a stop condition has not been detected).

Use the following sequence for generating a stop condition.

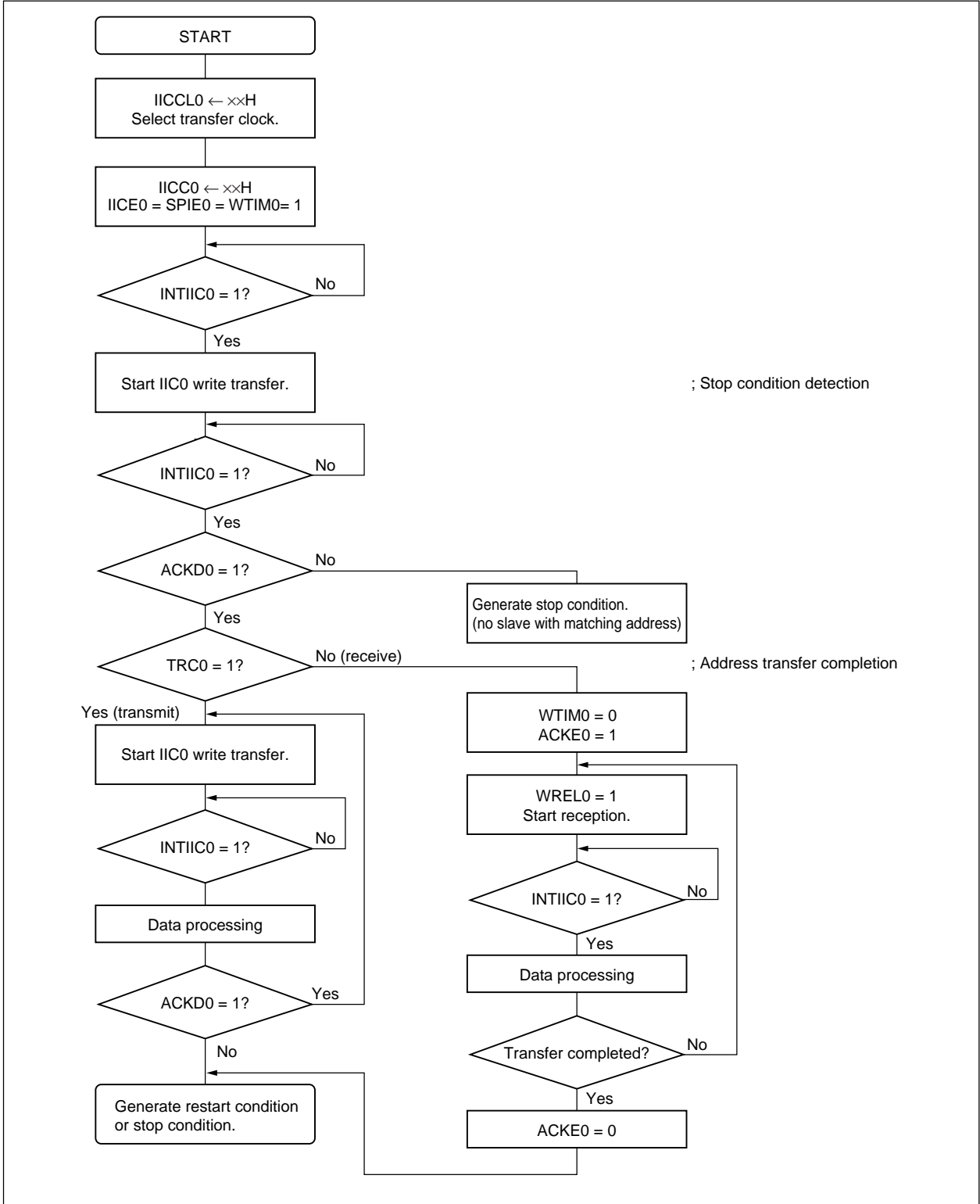
- (a) Set IIC clock selection register 0 (IICCL0).
- (b) Set bit 7 (IICE0) of IIC control register 0 (IICC0).
- (c) Set bit 0 of IICC0.

11.3.14 Communication operations

(1) Master operations

The following is a flow chart of the master operations.

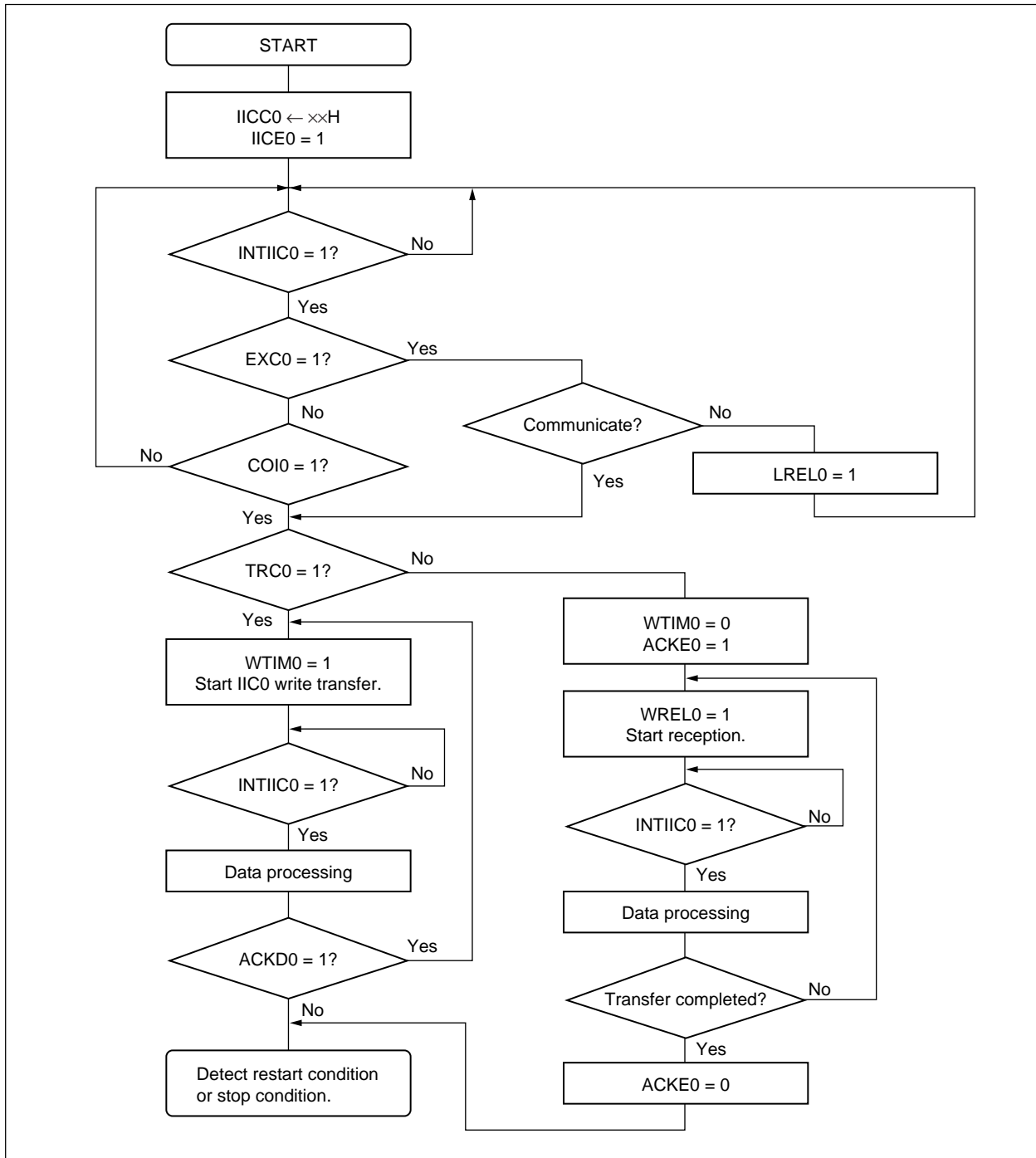
Figure 11-20. Master Operation Flow Chart



(2) Slave operation

An example of slave operation is shown below.

Figure 11-21. Slave Operation Flow Chart



11.3.15 Timing of data communication

When using I²C bus mode, the master device outputs an address via the serial bus to select one of several slave devices as its communication partner.

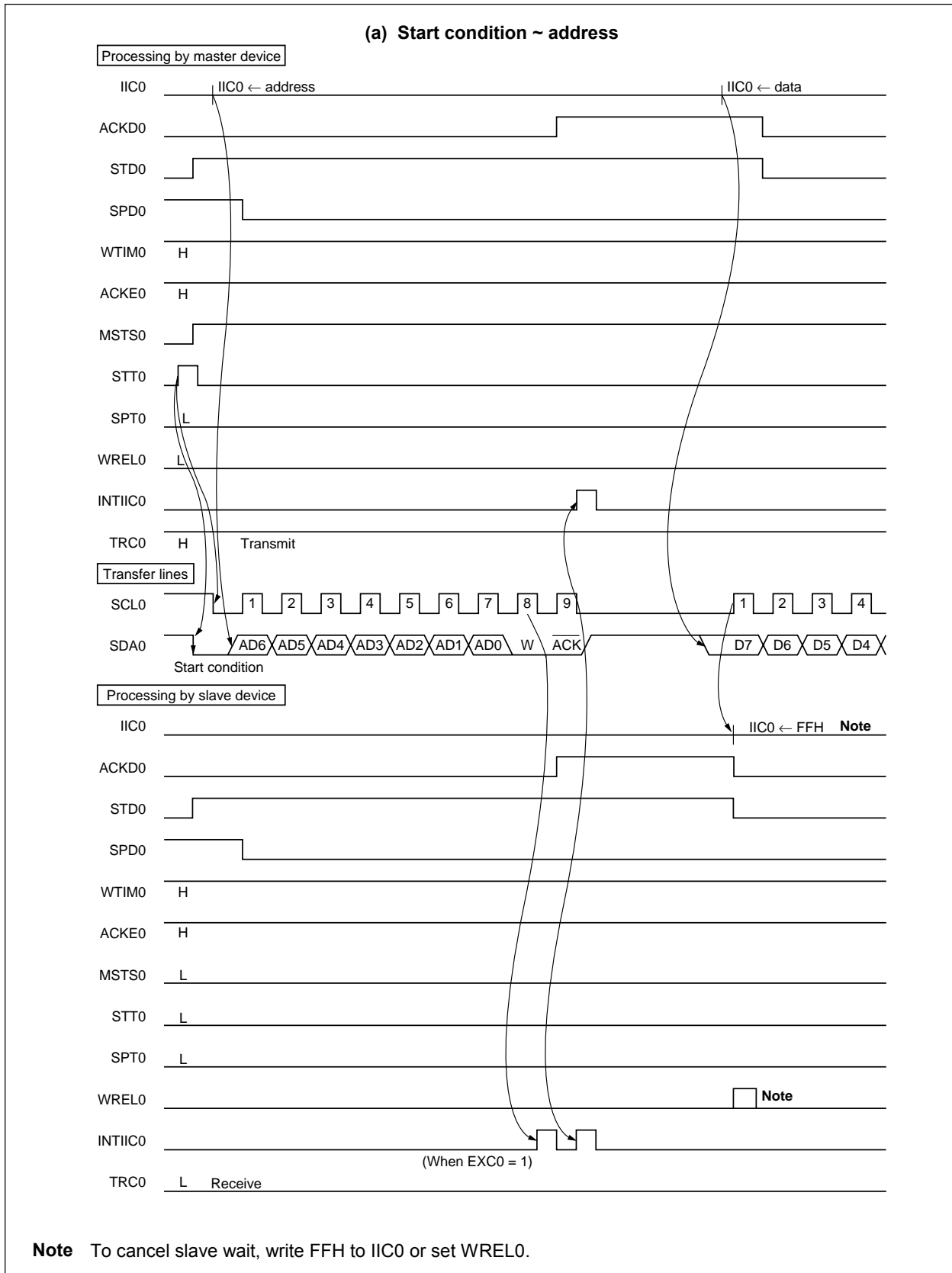
After outputting the slave address, the master device transmits the TRC0 bit (bit 3 of IIC status register 0 (IICS0)), which specifies the data transfer direction and then starts serial communication with the slave device.

The shift operation of IIC bus shift register 0 (IIC0) is synchronized with the falling edge of the serial clock (SCL0). The transmit data is transferred to the SO latch and is output (MSB first) via the SDA0 pin.

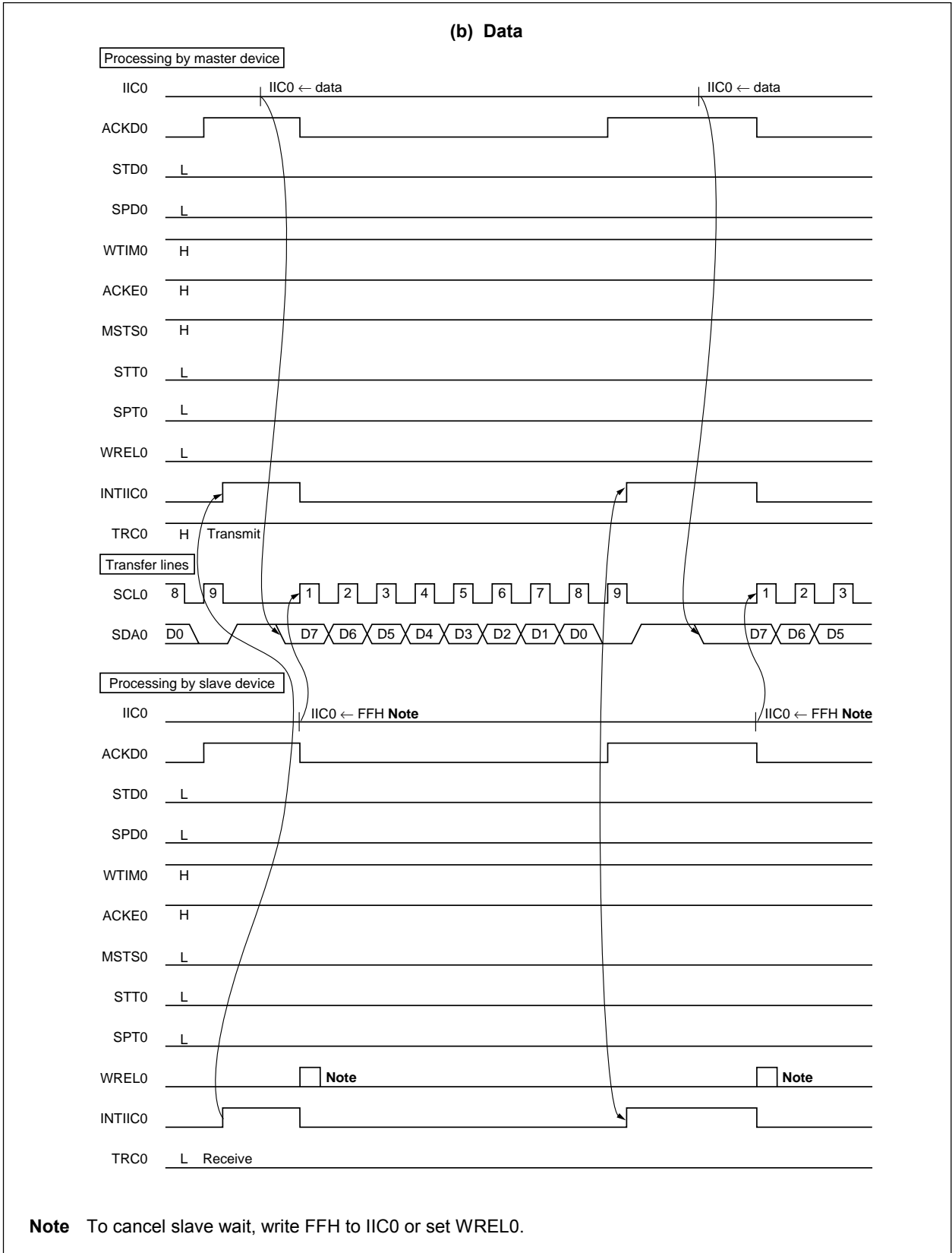
Data input via the SDA0 pin is captured by IIC0 at the rising edge of SCL0.

The data communication timing is shown below.

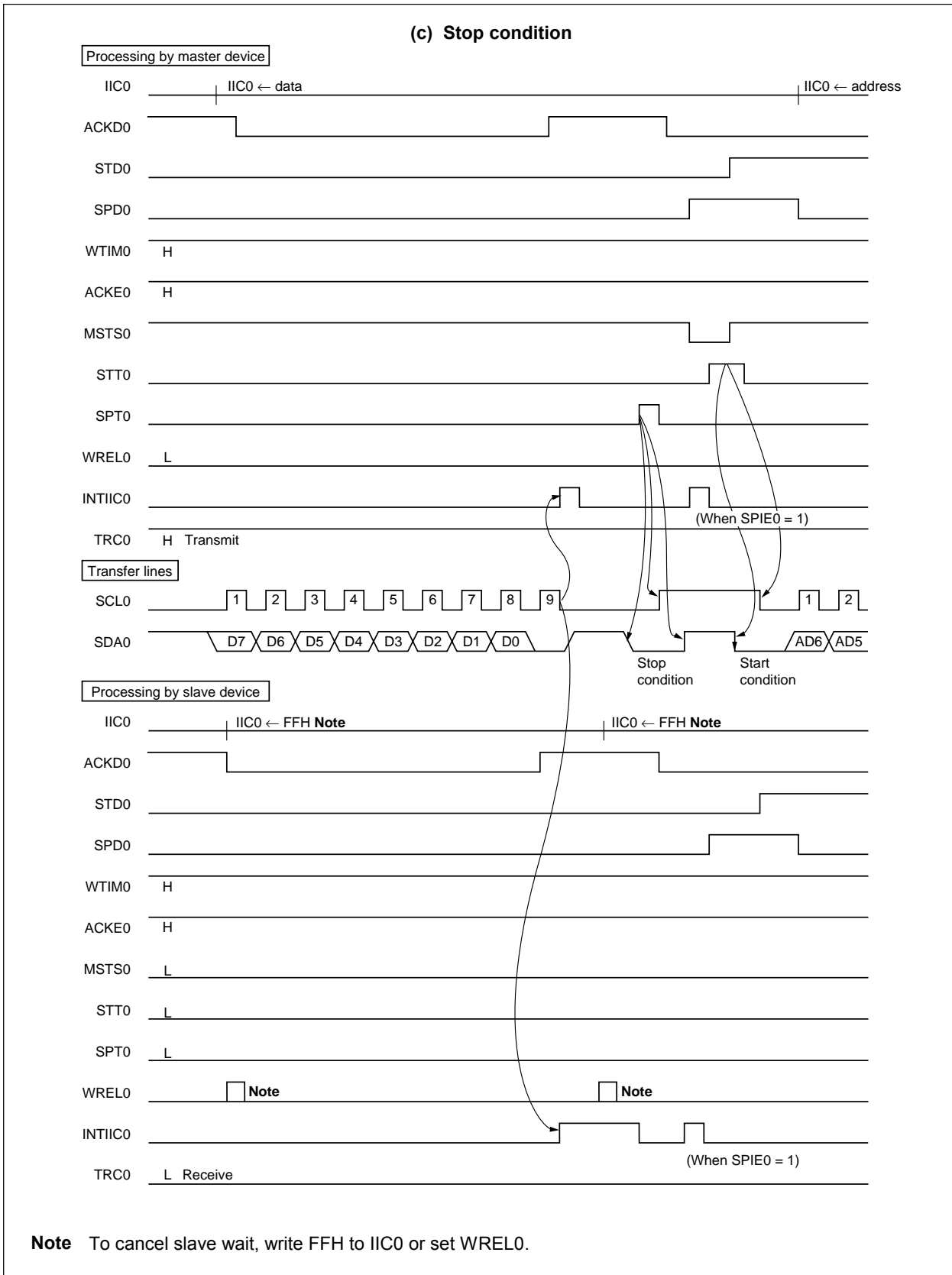
**Figure 11-22. Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)**



**Figure 11-22. Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)**



**Figure 11-22. Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)**



**Figure 11-23. Example of Slave to Master Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)**

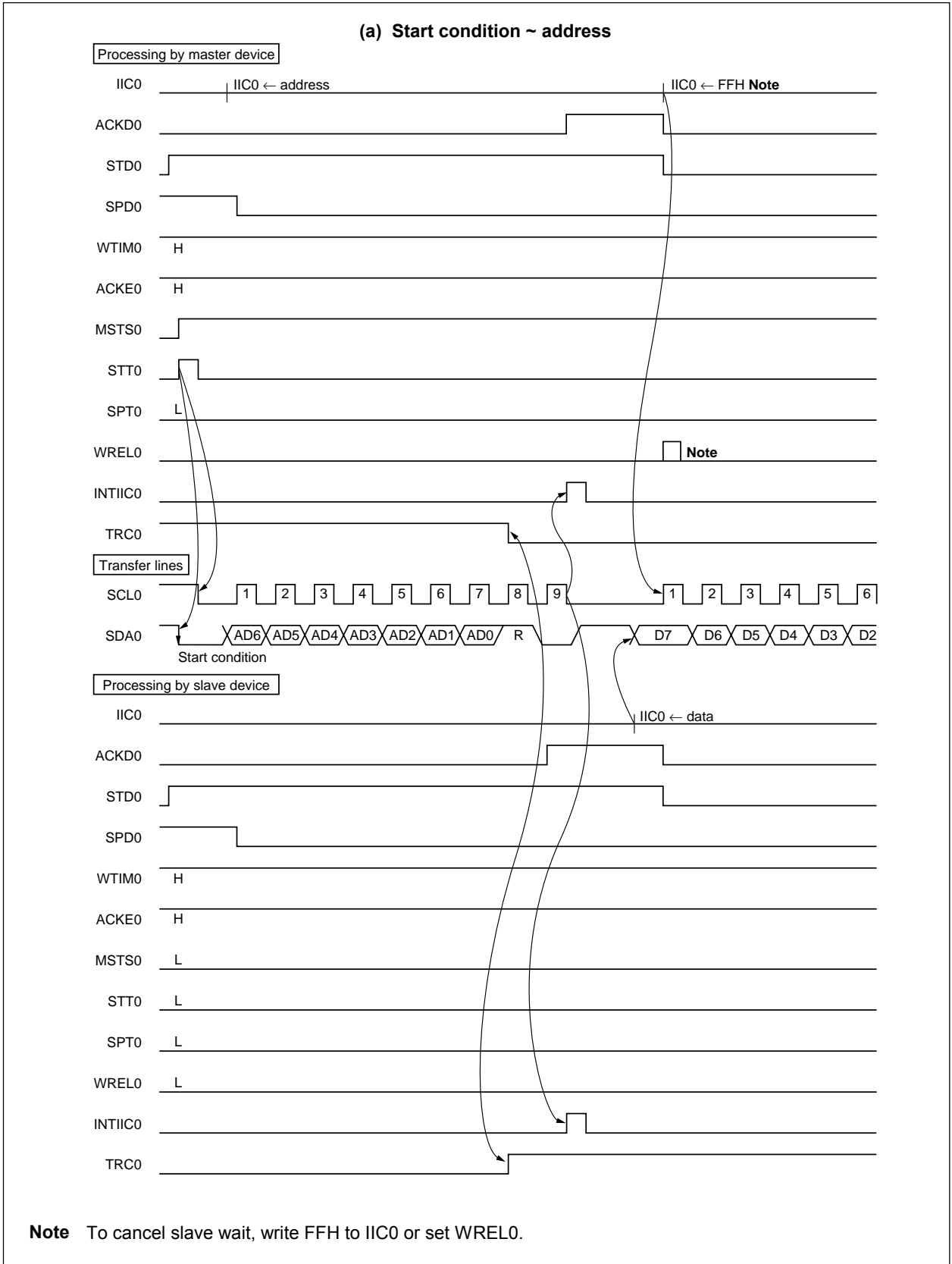
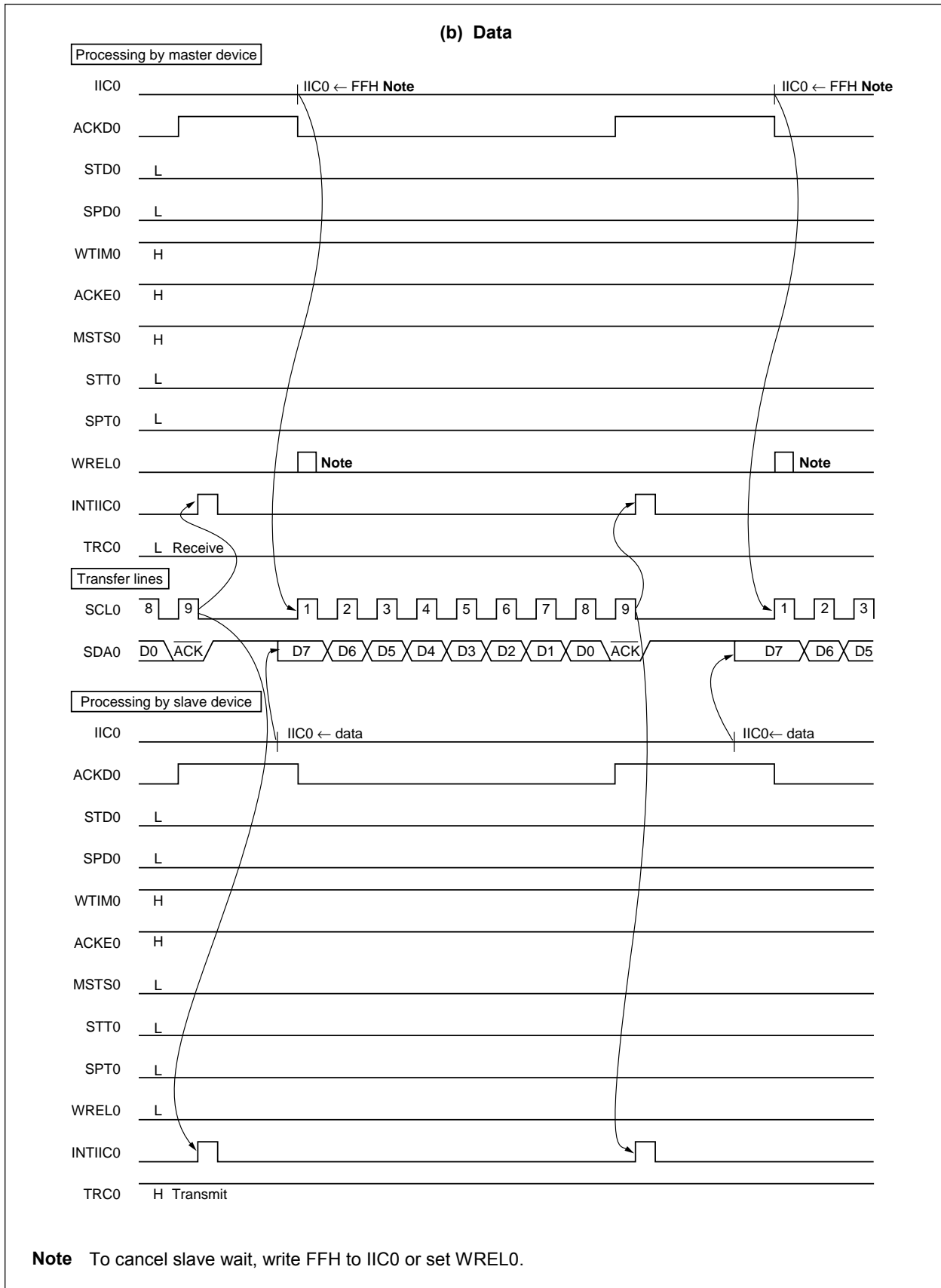
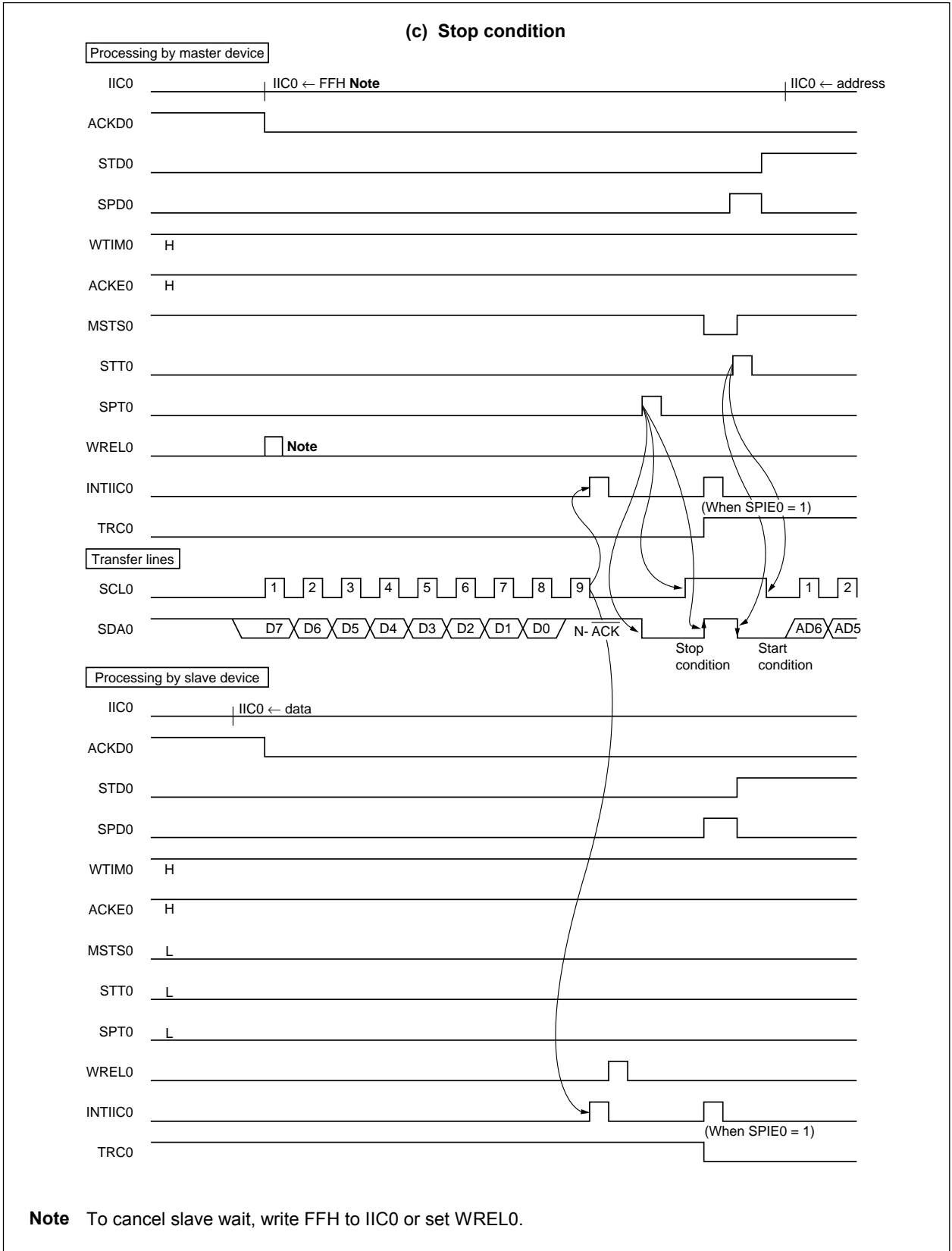


Figure 11-23. Example of Slave to Master Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)



**Figure 11-23. Example of Slave to Master Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)**



11.4 Asynchronous Serial Interface (UART0, UART1)

Remark n = 0, 1 in section 11.4.

UARTn has the following two operation modes.

(1) Operation stopped mode

This mode is used when serial transfers are not performed. It can therefore be used to reduce power consumption.

(2) Asynchronous serial interface mode

This mode enables full-duplex operation in which one byte of data is transmitted and received after the start bit. The on-chip dedicated UARTn baud rate generator enables communications using a wide range of selectable baud rates. In addition, a baud rate based on divided clock input to the ASCKn pin can also be defined. The UARTn baud rate generator can also be used to generate a MIDI-standard baud rate (31.25 kbps).

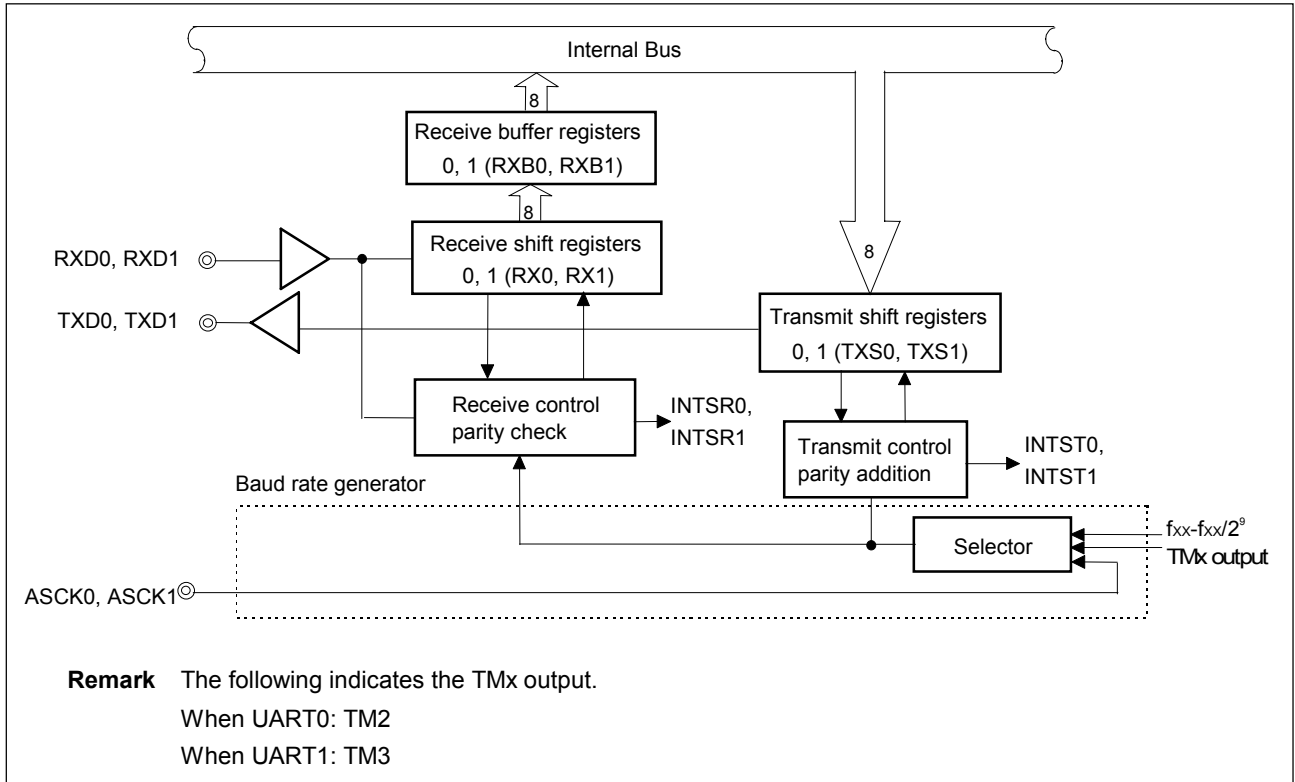
11.4.1 Configuration

UARTn includes the following hardware.

Table 11-8. Configuration of UARTn

Item	Configuration
Registers	Transmit shift registers 0, 1 (TXS0, TXS1) Receive buffer registers 0, 1 (RXB0, RXB1)
Control registers	Asynchronous serial interface mode registers 0, 1 (ASIM0, ASIM1) Asynchronous serial interface status registers 0, 1 (ASIS0, ASIS1) Baud rate generator control registers 0, 1 (BRGC0, BRGC1) Baud rate generator mode control registers 00, 01 (BRGMC00, BRGMC01) Baud rate generator mode control registers 10, 11 (BRGMC10, BRGMC11)

Figure 11-24. Block Diagram of UARTn

**(1) Transmit shift registers 0, 1 (TXS0, TXS1)**

TXSn is the register for setting transmit data. Data written to TXSn is transmitted as serial data.

When the data length is set as 7 bits, bit 0 to bit 6 of the data written to TXSn is transmitted as serial data.

Writing data to TXSn starts the transmit operation.

TXSn can be written to by an 8-bit memory manipulation instruction. It cannot be read.

RESET input sets these registers to FFH.

Caution Do not write to TXSn during a transmit operation.

(2) Receive shift registers 0, 1 (RX0, RX1)

RXn register converts serial data input via the RXD0, RXD1 pins to parallel data. When one byte of data is received at RXn, the received data is transferred to receive buffer registers 0, 1 (RXB0, RXB1).

RX0, RX1 cannot be manipulated directly by a program.

(3) Receive buffer registers 0, 1 (RXB0, RXB1)

RXBn is used to hold receive data. When one byte of data is received, one byte of new receive data is transferred.

When the data length is set as 7 bits, received data is sent to bit 0 to bit 6 of RXBn. In RXBn, the MSB must be set to "0".

RXBn can be read by an 8-bit memory manipulation instruction. It cannot be written.

RESET input sets RXBn to FFH.

(4) Transmission controller

The transmission controller controls transmit operations, such as adding a start bit, parity bit, and stop bit to data that is written to transmit shift register n (TXSn), based on the values set to asynchronous serial interface mode register n (ASIMn).

(5) Reception controller

The reception controller controls receive operations based on the values set to asynchronous serial interface mode register n (ASIMn). During a receive operation, it performs error checking, such as for parity errors, and sets various values to asynchronous serial interface status register n (ASISn) according to the type of error that is detected.

11.4.2 UARTn control registers

UARTn is controlled by the following registers.

- Asynchronous serial interface mode register n (ASIMn)
- Asynchronous serial interface status register n (ASISn)
- Baud rate generator control register n (BRGCn)
- Baud rate generator mode control registers n0, n1 (BRGMCn0, BRGMCn1)

(1) Asynchronous serial interface mode registers 0, 1 (ASIM0, ASIM1)

ASIMn is an 8-bit register that controls UARTn's serial transfer operations.

ASIMn can be set by an 8- or 1-bit memory manipulation instruction.

RESET input sets these registers to 00H.

After reset: 00H R/W Address: FFFF300H, FFFF310H

	7	6	5	4	3	2	1	0
ASIMn	TXEn	RXEn	PS1n	PS0n	UCLn	SLn	ISRMn	0

TXEn	RXEn	Operation mode	RxDn/Pxx pin function	TxDn/Pxx pin function
0	0	Operation stopped	Port function	Port function
0	1	UARTn mode (receive only)	Serial function	Port function
1	0	UARTn mode (transmit only)	Port function	Serial function
1	1	UARTn mode (transmit and receive)	Serial function	Serial function

PS1n	PS0n	Parity bit specification
0	0	No parity
0	1	Zero parity always added during transmission No parity detection during reception (parity errors do not occur)
1	0	Odd parity
1	1	Even parity

UCLn	Character length specification
0	7 bits
1	8 bits

SLn	Stop bit length specification for transmit data
0	1 bit
1	2 bits

ISRMn	Receive completion interrupt control when error occurs
0	Receive completion interrupt is issued when an error occurs
1	Receive completion interrupt is not issued when an error occurs

Caution Do not switch the operation mode until after the current serial transmit/receive operation has been stopped.

(2) Asynchronous serial interface status registers 0, 1 (ASIS0, ASIS1)

When a receive error occurs in asynchronous serial interface mode, these registers indicate the type of error.

ASISn can be read by an 8- or 1-bit memory manipulation instruction.

RESET input sets these registers to 00H.

After reset: 00H R Address: FFFF302H, FFFF312H

	7	6	5	4	3	2	1	0
ASISn	0	0	0	0	0	PEn	FEn	OVer

PEn	Parity error flag
0	No parity error
1	Parity error (Transmit data parity does not match)

FEn	Framing error flag
0	No framing error
1	Framing error ^{Note 1} (Stop bit not detected)

OVer	Overrun error flag
0	No overrun error
1	Overrun error ^{Note 2} (Next receive operation was completed before data was read from receive buffer register)

- Notes**
1. Even if a stop bit length has been set as two bits by setting bit 2 (SLn) in asynchronous serial interface mode register n (ASIMn), stop bit detection during a receive operation only applies to a stop bit length of 1 bit.
 2. Be sure to read the contents of receive buffer register n (RXBn) when an overrun error has occurred.
Until the contents of RXBn are read, further overrun errors will occur when receiving data.

(3) Baud rate generator control registers 0, 1 (BRGC0, BRGC1)

These registers set the serial clock for UARTn.

BRGCn can be set by an 8-bit memory manipulation instruction.

RESET input sets these registers to 00H.

After reset: 00H

R/W

Address: FFFF304H, FFFF314H

	7	6	5	4	3	2	1	0
BRGCn	MDLn7	MDLn6	MDLn5	MDLn4	MDLn3	MDLn2	MDLn1	MDLn0

MD Ln7	MD Ln6	MD Ln5	MD Ln4	MD Ln3	MD Ln2	MD Ln1	MD Ln0	Selection of input clock	k
0	0	0	0	0	×	×	×	Setting prohibited	–
0	0	0	0	1	0	0	0	f _{sck} /8	8
0	0	0	0	1	0	0	1	f _{sck} /9	9
0	0	0	0	1	0	1	0	f _{sck} /10	10
0	0	0	0	1	0	1	1	f _{sck} /11	11
0	0	0	0	1	1	0	0	f _{sck} /12	12
0	0	0	0	1	1	0	1	f _{sck} /13	13
0	0	0	0	1	1	1	0	f _{sck} /14	14
0	0	0	0	1	1	1	1	f _{sck} /15	15
0	0	0	1	0	0	0	0	f _{sck} /16	16
•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	1	1	1	f _{sck} /255	255

- Cautions**
1. The value of BRGCn becomes 00H after reset. Before starting operation, select a setting other than “Setting prohibited”. Selecting the “Setting prohibited” setting in stop mode does not cause any problems.
 2. If write is performed to BRGCn during communication processing, the output of the baud rate generator will be disturbed and communication will not be performed normally. Therefore, do not write to BRGCn during communication processing.

Remark f_{sck}: Source clock of 8-bit counter

(4) Baud rate generator mode control registers n0, n1 (BRGMCn0, BRGMCn1)

These registers set the UARTn source clock.

BRGMCn0 and BRGMCn1 are set by an 8-bit memory manipulation instruction.

RESET input sets these registers to 00H.

After reset: 00H R/W Address: FFFFF30EH, FFFFF31EH

	7	6	5	4	3	2	1	0
BRGMCn0	0	0	0	0	0	TPSn2	TPSn1	TPSn0

After reset: 00H R/W Address: FFFFF320H, FFFFF322H

	7	6	5	4	3	2	1	0
BRGMCn1	0	0	0	0	0	0	0	TPSn3

TPSn3	TPSn2	TPSn1	TPSn0	8-bit counter source clock selection	m
0	0	0	0	External clock (ASCKn)	–
0	0	0	1	f_{xx}	0
0	0	1	0	$f_{xx}/2$	1
0	0	1	1	$f_{xx}/4$	2
0	1	0	0	$f_{xx}/8$	3
0	1	0	1	$f_{xx}/16$	4
0	1	1	0	$f_{xx}/32$	5
0	1	1	1	at n = 0: TM2 output at n = 1: TM3 output	–
1	0	0	0	$f_{xx}/64$	6
1	0	0	1	$f_{xx}/128$	7
1	0	1	0	$f_{xx}/256$	8
1	0	1	1	$f_{xx}/512$	9
1	1	0	0	Setting prohibited	–
1	1	0	1		–
1	1	1	0		–
1	1	1	1		–

Caution If write is performed to BRGMCn0, n1 during communication processing, the output of the baud rate generator will be disturbed and communication will not be performed normally. Therefore, do not write to BRGMCn0, n1 during communication processing.

Remarks 1. f_{sck} : Source clock of 8-bit counter

2. When the selection clock is output from the timer, pins P30/TO2/TI2 and P31/TO3/TI3 do not need to be set to timer output mode.

★

11.4.3 Operations

UARTn has the following two operation modes.

- Operation stopped mode
- Asynchronous serial interface mode

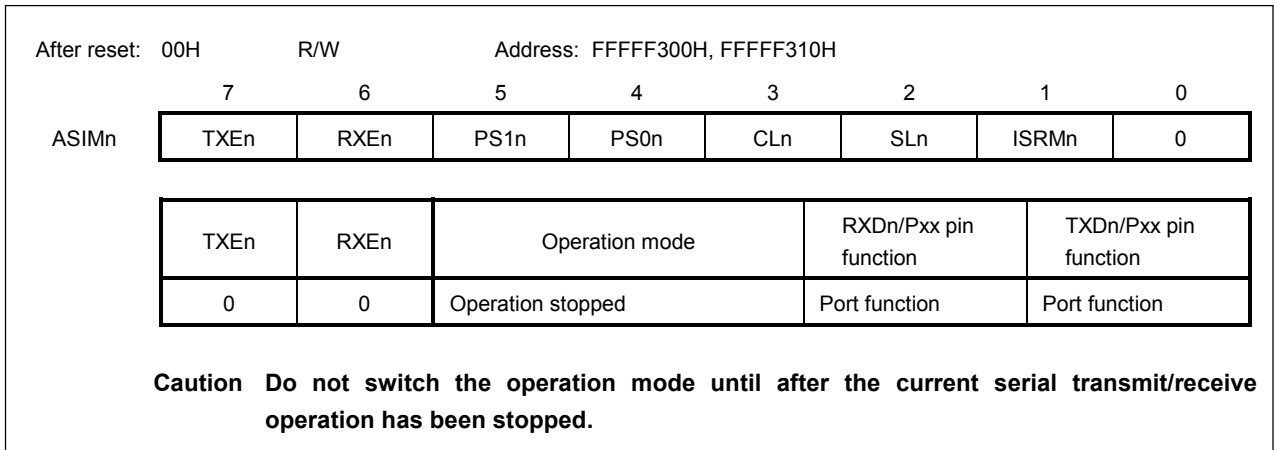
(1) Operation stopped mode

In this mode, serial transfers are not performed and therefore power consumption can be reduced. When in operation stopped mode, pins can be used as ordinary ports.

(a) Register settings

Operation stopped mode settings are made via bits TXEn and RXEn of asynchronous serial interface mode register n (ASIMn).

Figure 11-25. ASIMn Setting (Operation Stopped Mode)



(2) Asynchronous serial interface mode

This mode enables full-duplex operation in which one byte of data after the start bit is transmitted and received.

The on-chip dedicated UARTn baud rate generator enables communications using a wide range of selectable baud rates.

The UARTn baud rate generator can also be used to generate a MIDI-standard baud rate (31.25 kbps).

(a) Register settings

The asynchronous serial interface mode settings are made via ASIMn, BRGCn, BRGMCn0, and BRGMCn1.

Figure 11-26. ASIMn Setting (Asynchronous Serial Interface Mode)

After reset: 00H	R/W	Address: FFFF300H, FFFF310H							
		7	6	5	4	3	2	1	0
ASIMn		TXEn	RXEn	PS1n	PS0n	CLn	SLn	ISRMn	0

TXEn	RXEn	Operation mode	RxDn/Pxx pin function	TxDn/Pxx pin function
0	1	UARTn mode (receive only)	Serial function	Port function
1	0	UARTn mode (transmit only)	Port function	Serial function
1	1	UARTn mode (transmit and receive)	Serial function	Serial function

PS1n	PS0n	Parity bit specification
0	0	No parity
0	1	Zero parity always added during transmission No parity detection during reception (parity errors do not occur)
1	0	Odd parity
1	1	Even parity

CLn	Character length specification
0	7 bits
1	8 bits

SLn	Stop bit length specification for transmit data
0	1 bit
1	2 bits

ISRMn	Receive completion interrupt control when error occurs
0	Receive completion interrupt is issued when an error occurs
1	Receive completion interrupt is not issued when an error occurs

Caution Do not switch the operation mode until after the current serial transmit/receive operation has been stopped.

Figure 11-27. ASISn Setting (Asynchronous Serial Interface Mode)

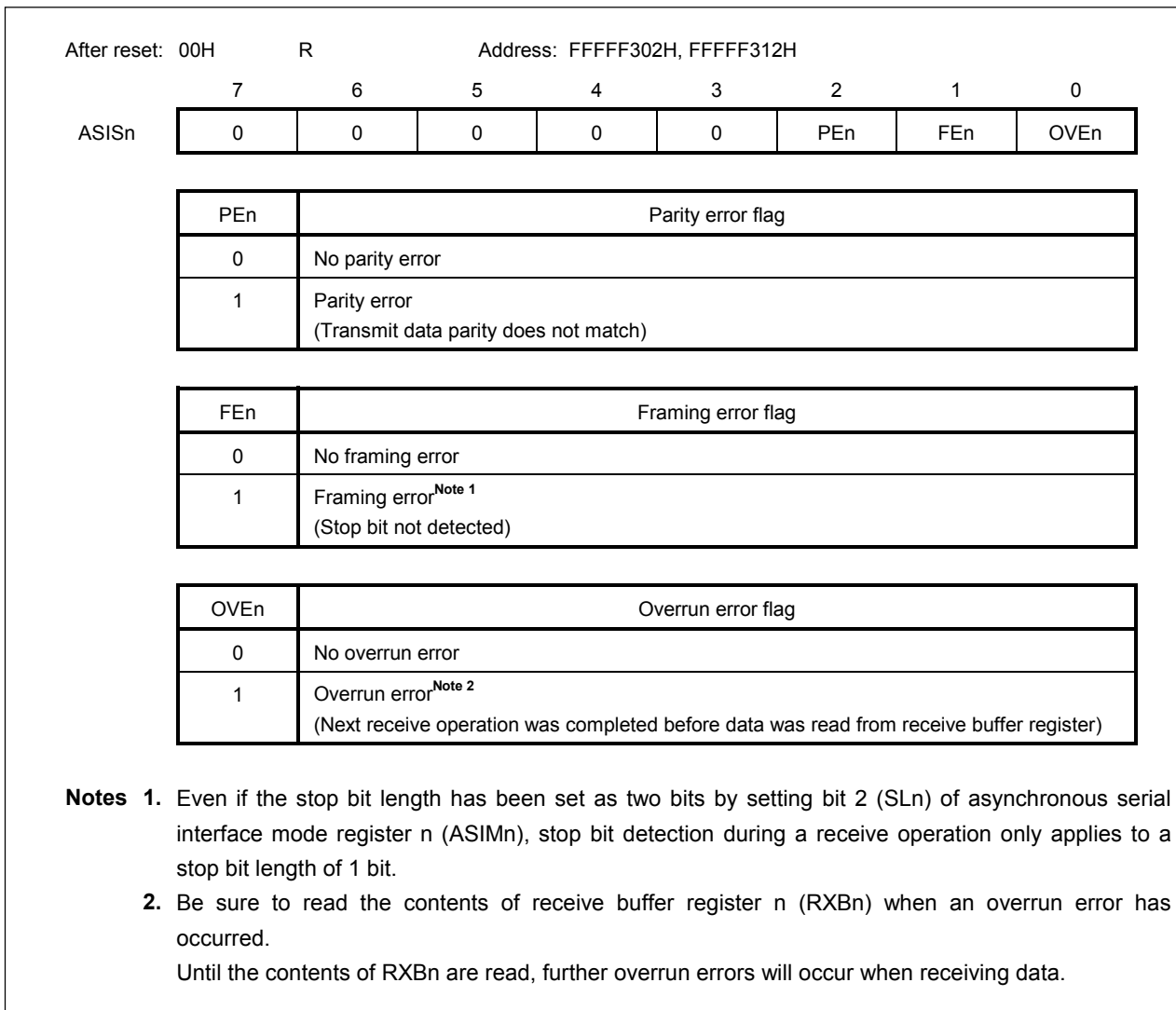


Figure 11-28. BRGCn Setting (Asynchronous Serial Interface Mode)

After reset: 00H R/W Address: FFFF304H, FFFF314H

	7	6	5	4	3	2	1	0
BRGCn	MDLn7	MDLn6	MDLn5	MDLn4	MDLn3	MDLn2	MDLn1	MDLn0

MD Ln7	MD Ln6	MD Ln5	MD Ln4	MD Ln3	MD Ln2	MD Ln1	MD Ln0	Input clock selection	k
0	0	0	0	0	×	×	×	Setting prohibited	–
0	0	0	0	1	0	0	0	f _{sck} /8	8
0	0	0	0	1	0	0	1	f _{sck} /9	9
0	0	0	0	1	0	1	0	f _{sck} /10	10
0	0	0	0	1	0	1	1	f _{sck} /11	11
0	0	0	0	1	1	0	0	f _{sck} /12	12
0	0	0	0	1	1	0	1	f _{sck} /13	13
0	0	0	0	1	1	1	0	f _{sck} /14	14
0	0	0	0	1	1	1	1	f _{sck} /15	15
0	0	0	1	0	0	0	0	f _{sck} /16	16
•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	1	1	1	f _{sck} /255	255

- Cautions**
1. Before starting operation, select a setting other than “Setting prohibited”. Selecting “Setting prohibited” setting in stop mode does not cause any problems.
 2. If write is performed to BRGCn during communication processing, the output of the baud rate generator is disturbed and communication will not be performed normally. Therefore, do not write to BRGCn during communication processing.

Remark f_{sck}: Source clock of 8-bit counter

Figure 11-29. BRGMCn0 and BRGMCn1 Settings (Asynchronous Serial Interface Mode)

After reset: 00H R/W Address: FFFF30EH, FFFF31EH

	7	6	5	4	3	2	1	0
BRGMCn0	0	0	0	0	0	TPSn2	TPSn1	TPSn0

After reset: 00H R/W Address: FFFF320H, FFFF322H

	7	6	5	4	3	2	1	0
BRGMCn1	0	0	0	0	0	0	0	TPSn3

TPSn3	TPSn2	TPSn1	TPSn0	8-bit counter source clock selection	m
0	0	0	0	External clock (ASCKn)	–
0	0	0	1	f_{xx}	0
0	0	1	0	$f_{xx}/2$	1
0	0	1	1	$f_{xx}/4$	2
0	1	0	0	$f_{xx}/8$	3
0	1	0	1	$f_{xx}/16$	4
0	1	1	0	$f_{xx}/32$	5
0	1	1	1	at n = 0: TM3 output at n = 1: TM2 output	–
1	0	0	0	$f_{xx}/64$	6
1	0	0	1	$f_{xx}/128$	7
1	0	1	0	$f_{xx}/256$	8
1	0	1	1	$f_{xx}/512$	9
1	1	0	0	Setting prohibited	–
1	1	0	1		–
1	1	1	0		–
1	1	1	1		–

Caution If write is performed to BRGMCn0, n1 during communication processing, the output of the baud rate generator is disturbed and communication will not be performed normally. Therefore, do not write to BRGMCn0 and BRGMCn1 during communication processing.

Remarks

1. f_{xx} : Main clock oscillation frequency
2. When the selection clock is output from the timer, pins P30/TO2/TI2 and P31/TO3/TI3 do not need to be set in timer output mode.

★

(b) Baud rate

The baud rate transmit/receive clock that is generated is obtained by dividing the main clock.

- **Generation of baud rate transmit/receive clock using main clock**

The transmit/receive clock is obtained by dividing the main clock. The following equation is used to obtain the baud rate from the main clock.

<When $8 \leq k \leq 255$ >

$$[\text{Baud rate}] = \frac{f_{xx}}{2^{m+1} \times k} \text{ [Hz]}$$

f_{xx} : Main clock oscillation frequency

m: Value set by TPSn3 to TPSn0 ($0 \leq m \leq 9$)

k: Value set by MDLn7 to MDLn0 ($8 \leq k \leq 255$)

- **Baud rate tolerance**

The baud rate tolerance depends on the number of bits in a frame and the counter division ratio $[1/(16+k)]$.

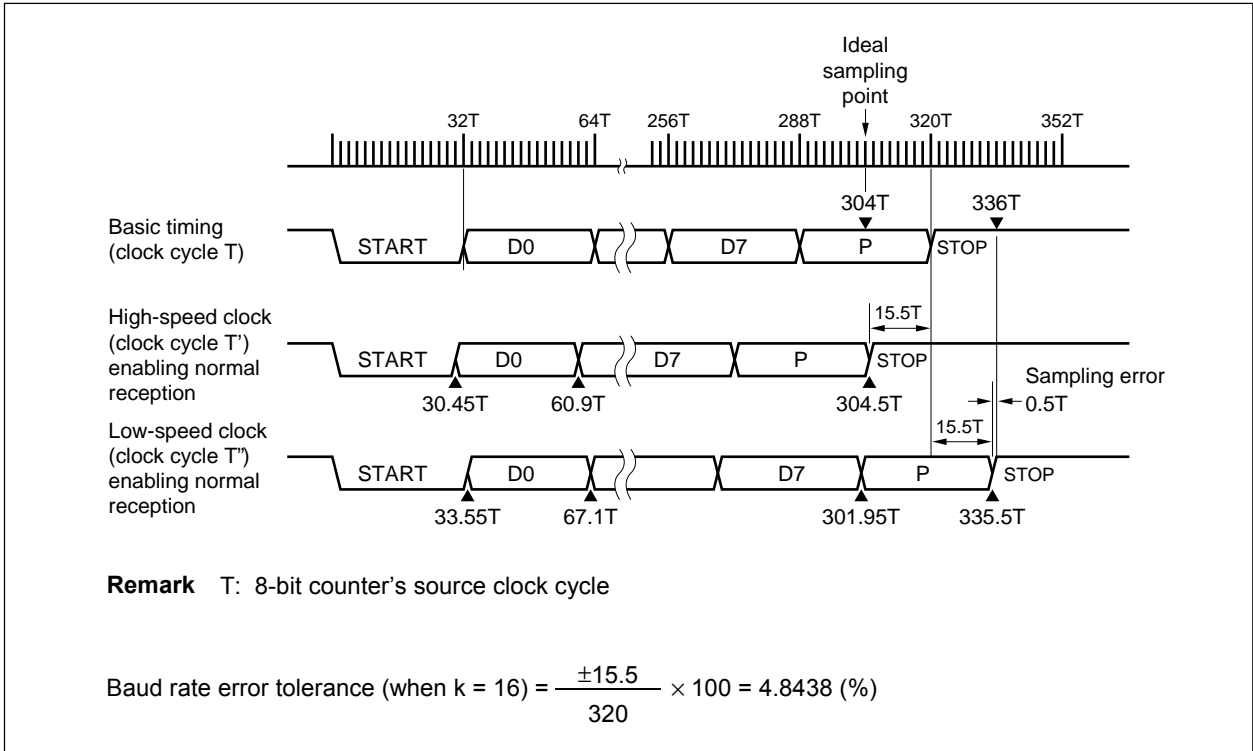
Table 11-9 shows the relationship between the main clock and the baud rate, and Figure 11-30 shows an example of the baud rate tolerance.

Table 11-9. Relationship Between Main Clock and Baud Rate

Baud Rate (bps)	$f_{xx} = 16 \text{ MHz}$			$f_{xx} = 8 \text{ MHz}$		
	k	m	Error (%)	k	m	Error (%)
32	–	–	–	244	9	0.06
64	244	9	0.06	244	8	0.06
128	244	8	0.06	244	7	0.06
300	208	7	0.16	208	6	0.16
600	208	6	0.16	208	5	0.16
1200	208	5	0.16	208	4	0.16
2400	208	4	0.16	208	3	0.16
4800	208	3	0.16	208	2	0.16
9600	208	2	0.16	208	1	0.16
19200	208	1	0.16	208	0	0.16
38400	208	0	0.16	104	0	0.16
76800	104	0	0.16	52	0	0.16
150000	53	0	0.63	27	0	-1.24
300000	27	0	-1.24	13	0	2.56

Remark f_{xx} : Main clock oscillation frequency

Figure 11-30. Error Tolerance (When k = 16), Including Sampling Errors



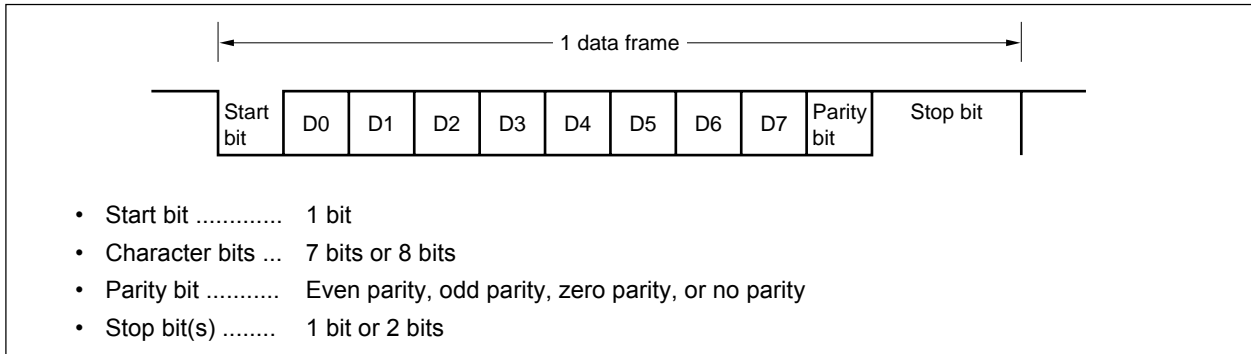
(3) Communication operations

(a) Data format

As shown in Figure 11-31, the format of the transmit/receive data consists of a start bit, character bits, a parity bit, and one or more stop bits.

Asynchronous serial interface mode register n (ASIMn) is used to set the character bit length, parity selection, and stop bit length within each data frame.

Figure 11-31. Format of Transmit/Receive Data in Asynchronous Serial Interface



When 7 bits is selected as the number of character bits, only the lower 7 bits (from bit 0 to bit 6) are valid, so during a transmission the most significant bit (bit 7) is ignored and during reception the most significant bit (bit 7) must be set to 0.

Asynchronous serial interface mode register n (ASIMn) and baud rate generator control register n (BRGCn) are used to set the serial transfer rate.

If a receive error occurs, information about the receive error can be ascertained by reading asynchronous serial interface status register n (ASISn).

(b) Parity types and operations

The parity bit is used to detect bit errors in transfer data. Usually, the same type of parity bit is used by the transmitting and receiving sides. When odd parity or even parity is set, errors in the parity bit (the odd-number bit) can be detected. When zero parity or no parity is set, errors are not detected.

(i) Even parity**• During transmission**

The number of bits in transmit data including a parity bit is controlled so that the number is set an even number of “1” bits. The value of the parity bit is as follows.

If the transmit data contains an odd number of “1” bits: The parity bit value is “1”

If the transmit data contains an even number of “1” bits: The parity bit value is “0”

• During reception

The number of “1” bits is counted among the receive data including a parity bit, and a parity error is generated when the result is an odd number.

(ii) Odd parity**• During transmission**

The number of bits in transmit data including a parity bit is controlled so that the number is set an odd number of “1” bits. The value of the parity bit is as follows.

If the transmit data contains an odd number of “1” bits: The parity bit value is “0”

If the transmit data contains an even number of “1” bits: The parity bit value is “1”

• During reception

The number of “1” bits is counted among the receive data including a parity bit, and a parity error is generated when the result is an even number.

(iii) Zero parity

During transmission, the parity bit is set to “0” regardless of the transmit data.

During reception, the parity bit is not checked. Therefore, no parity errors will be generated regardless of whether the parity bit is a “0” or a “1”.

(iv) No parity

No parity bit is added to the transmit data.

During reception, receive data is regarded as having no parity bit. Since there is no parity bit, no parity errors will be generated.

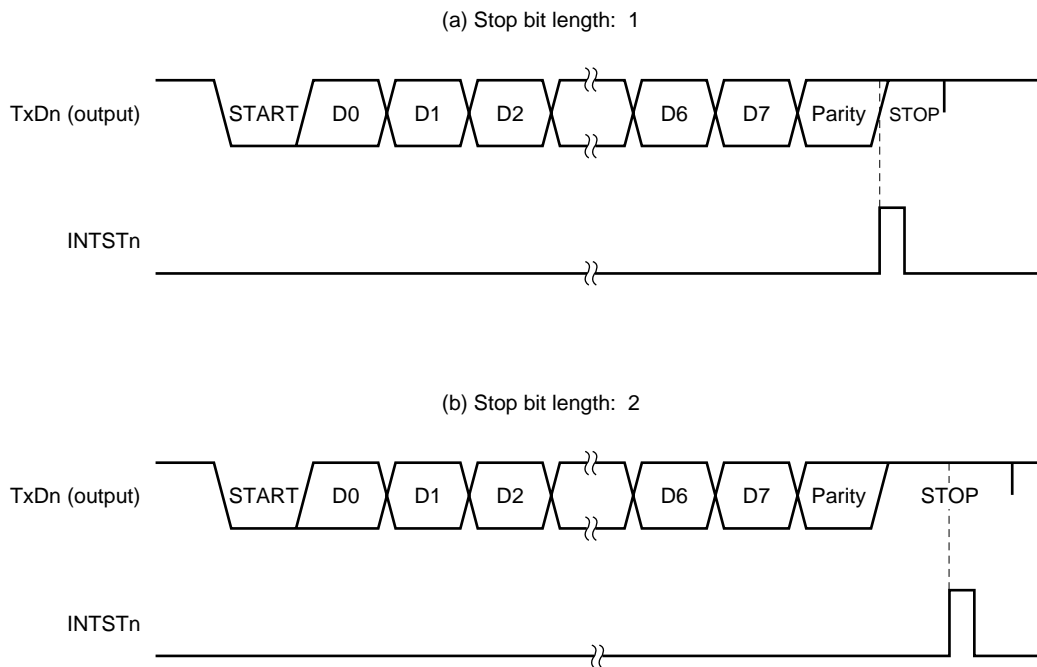
(c) Transmission

A transmit operation is started when transmit data is written to transmit shift register n (TXSn). A start bit, parity bit, and stop bit(s) are automatically added to the data.

Starting a transmit operation shifts out the data in TXSn, thereby emptying TXSn, after which a transmit completion interrupt (INTSTn) is issued.

The timing of the transmit completion interrupt is shown below.

Figure 11-32. Timing of Asynchronous Serial Interface Transmit Completion Interrupt



Caution Do not write to asynchronous serial interface mode register n (ASIMn) during a transmit operation. Writing to ASIMn during a transmit operation may disable further transmit operations (in such cases, enter a **RESET** to restore normal operation).

Whether or not a transmit operation is in progress can be determined via software using the transmit completion interrupt (INTSTn) or the interrupt request flag (STIFn) that is set by INTSTn.

(d) Reception

A receive operation is enabled when bit 6 (RXEn) of asynchronous serial interface mode register n (ASIMn) is set to 1, and input via the RXDn pin is sampled.

The serial clock specified by ASIMn is used when sampling the RXDn pin.

When the RXDn pin goes low, the 8-bit counter begins counting and the start timing signal for data sampling is output when half of the specified baud rate time has elapsed. If sampling the RXDn pin input with this start timing signal yields a low-level result, the start bit is recognized, after which the 8-bit counter is initialized and starts counting and data sampling begins. After the start bit is recognized, the character data, parity bit, and one-bit stop bit are detected, at which point reception of one data frame is completed.

Once reception of one data frame is complete, the receive data in the shift register is transferred to receive buffer register n (RXBn) and a receive completion interrupt (INTSRn) occurs.

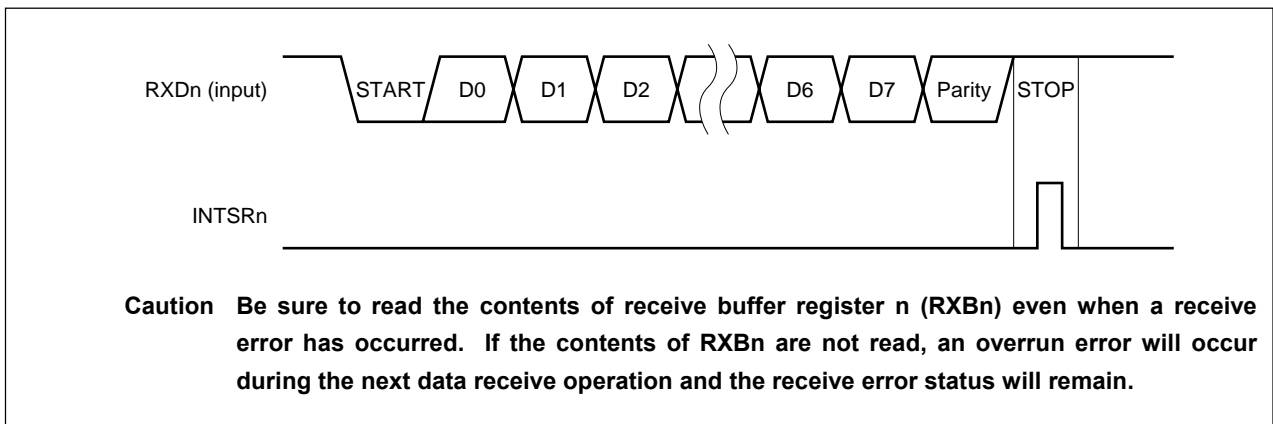
Even if an error has occurred, the receive data in which the error occurred is still transferred to RXBn.

When an error occurs, INSTRn is generated if bit 1 (ISRMn) of ASIMn is cleared (0). On the other hand, INTSRn is not generated if the ISRMn bit is set (1).

If the RXEn bit is reset to 0 during a receive operation, the receive operation is stopped immediately. At this time, the contents of RXBn and ASISn do not change, nor does INTSRn or INTSERn occur.

The timing of the asynchronous serial interface receive completion interrupt is shown below.

Figure 11-33. Timing of Asynchronous Serial Interface Receive Completion Interrupt



(e) Receive error

There are three types of errors during a receive operation: a parity error, a framing error, and an overrun error. When, as the result of data reception, an error flag is set in asynchronous serial interface status register n (ASISn), the receive error interrupt request (INTSERn) is generated. The receive error interrupt request is generated prior to the receive completion interrupt request (INTSRn).

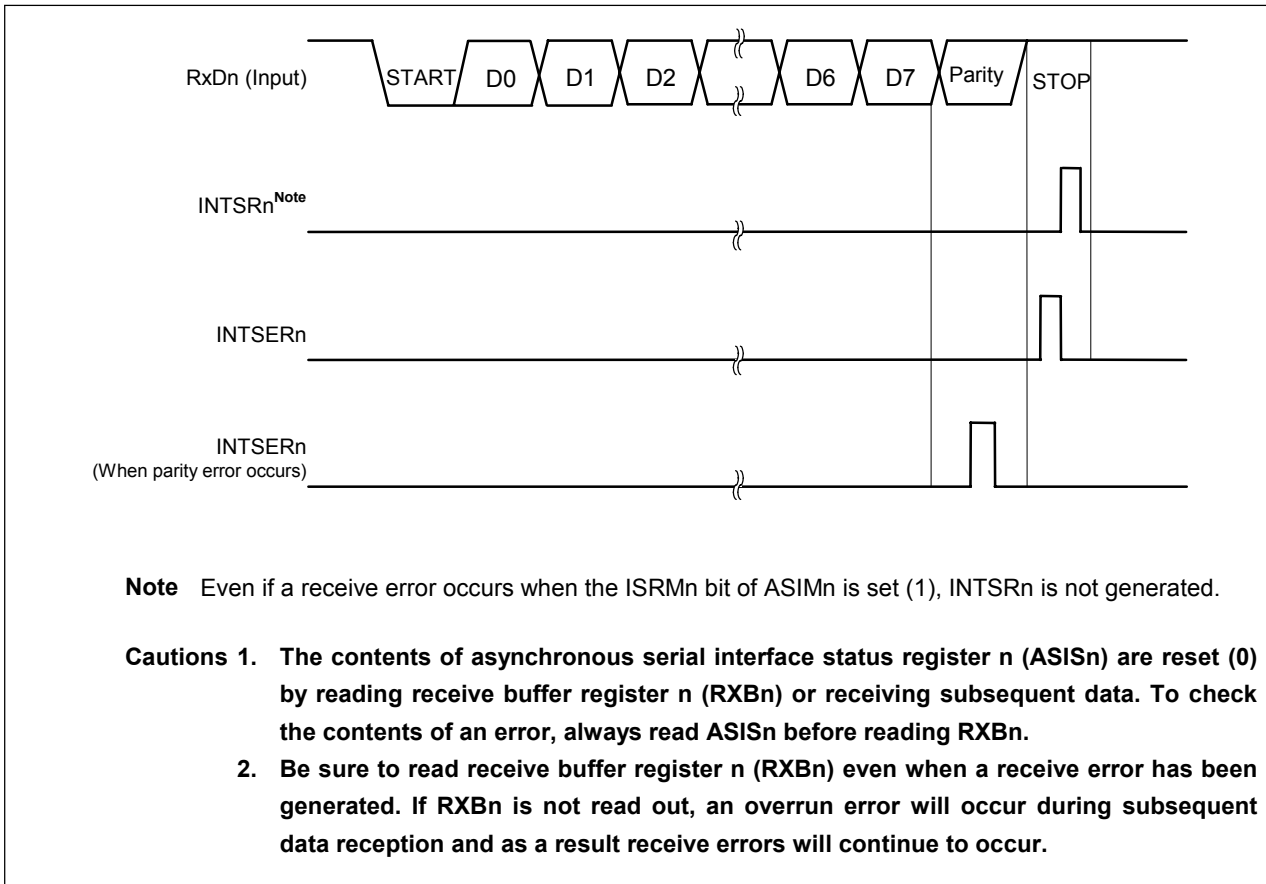
By reading the contents of ASISn during receive error interrupt servicing (INTSERn), it is possible to detect which error has occurred at reception.

The contents of ASISn are reset (0) by reading receive buffer register n (RXBn) or receiving subsequent data (if there is an error in the subsequent data, the error flag is set).

Table 11-10. Receive Error Causes

Receive Error	Cause	ASISn Value
Parity error	Parity specification at transmission and receive data parity do not match.	04H
Framing error	Stop bit is not detected.	02H
Overrun error	Reception of subsequent data was completed before data was read from the receive buffer register.	01H

Figure 11-34. Receive Error Timing



11.4.4 Standby function

(1) Operation in HALT mode

Serial transfer operations are performed normally.

(2) Operation in STOP and IDLE modes

(a) When internal clock is selected as serial clock

The operations of asynchronous serial interface mode register n (ASIMn), transmit shift register n (TXSn), and receive buffer register n (RXBn) are stopped and their values immediately before the clock stopped are held.

The TXDn pin output holds the data immediately before the clock is stopped (in STOP mode) during transmission. When the clock is stopped during reception, the receive data until the clock stopped is stored and subsequent receive operations are stopped. Reception resumes upon clock restart.

(b) When external clock is selected as serial clock

Serial transfer operations are performed normally.

11.5 3-Wire Variable-Length Serial I/O (CSI4)

CSI4 has the following two operation modes.

(1) Operation stopped mode

This mode is used when serial transfers are not performed.

(2) 3-wire variable-length serial I/O mode (MSB/LSB first switchable)

This mode transfers variable data of 8 to 16 bits via three lines: a serial clock line ($\overline{\text{SCK4}}$), a serial output line (SO4), and a serial input line (SI4).

Since data can be transmitted and received simultaneously in 3-wire variable-length serial I/O mode, the processing time of data transfer is shortened.

MSB and LSB can be switched for the first bit of data to be transferred in serial.

3-wire variable-length serial I/O mode is useful when connecting to a peripheral I/O device that includes a clocked serial interface, a display controller, etc.

Caution The serial clock ($\overline{\text{SCK4}}$) must be used within the range of $\overline{\text{SCK4}} \leq 2.5 \text{ MHz}$.

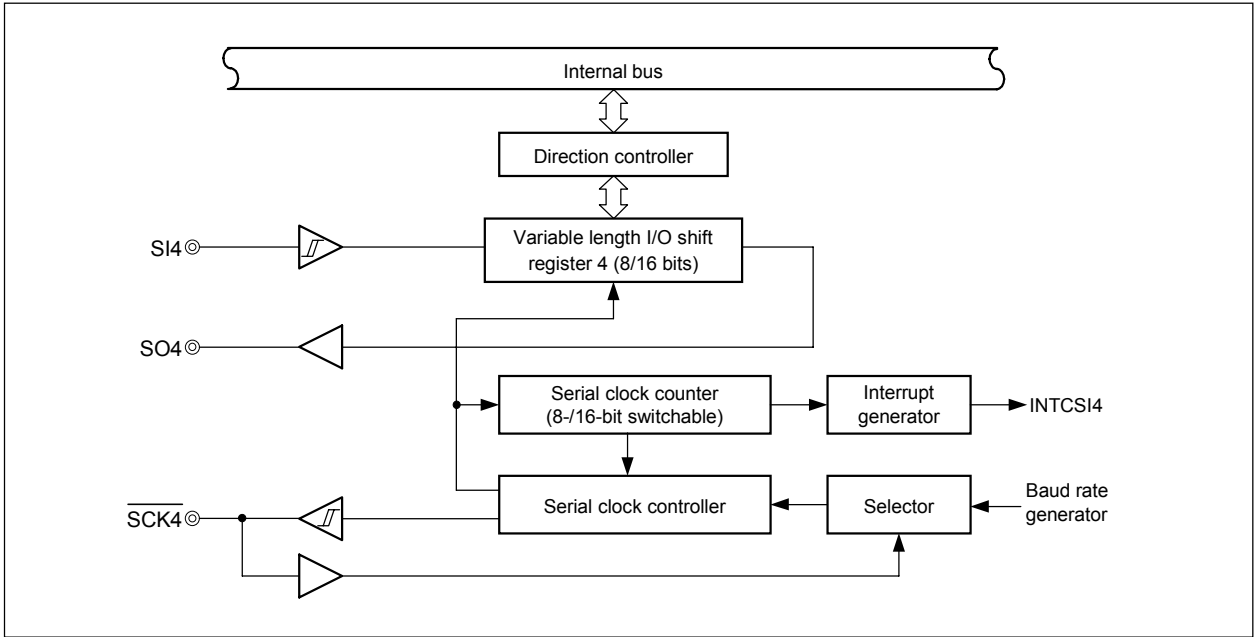
11.5.1 Configuration

CSI4 includes the following hardware.

Table 11-11. Configuration of CSI4

Item	Configuration
Register	Variable-length serial IO shift register 4 (SIO4)
Control registers	Variable-length serial control register 4 (CSIM4) Variable-length serial setting register 4 (CSIB4) Baud rate generator source clock selection register 4 (BRGCN4) Baud rate generator output clock selection register 4 (BRGCK4)

Figure 11-35. Block Diagram of 3-Wire Variable-Length Serial I/O



(1) Variable-length serial I/O shift register 4 (SIO4)

SIO4 is a 16-bit variable register that performs parallel-serial conversion and transmission/reception (shift operations) in synchronization with the serial clock.

SIO4 is set by a 16-bit memory manipulation instruction.

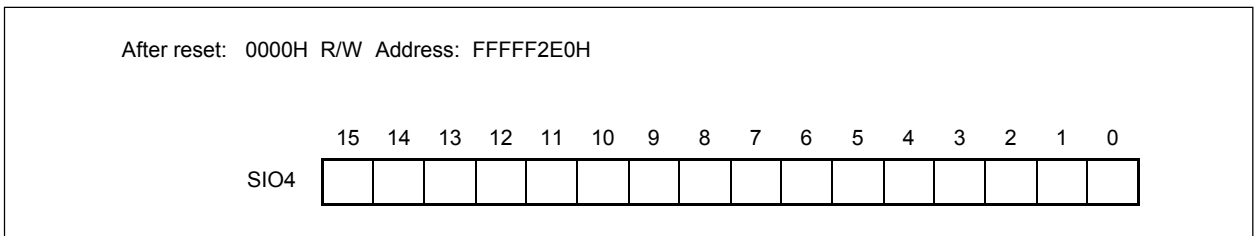
A serial operation starts when data is written to or read from SIO4, while bit 7 (CSIE4) of variable-length serial control register 4 (CSIM4) is 1.

When transmitting, data written to SIO4 is output via the serial output (SO4).

When receiving, data is read from the serial input (SI4) and written to SIO4.

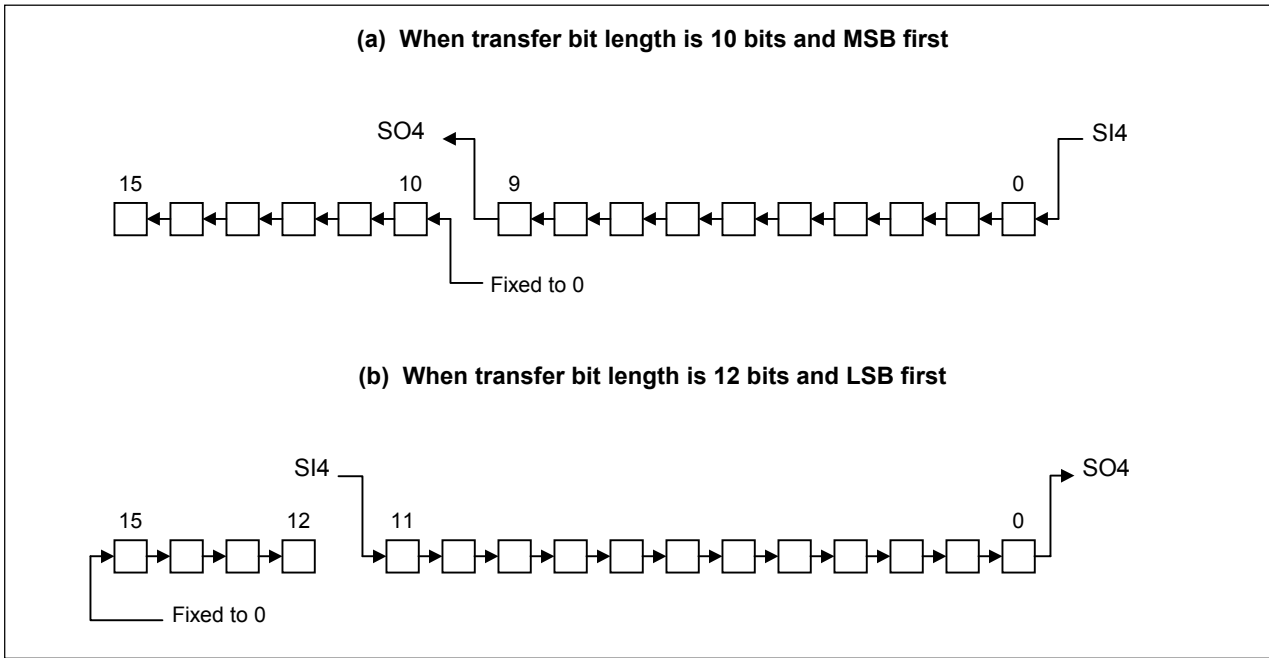
RESET input resets SIO4 to 0000H.

Caution Do not access SIO4 except via the transfer start trigger during a transfer operation (read is disabled when MODE4 = 0 and write is disabled when MODE4 = 1).



When the transfer bit length is set to other than 16 bits and data is set to the shift register, data should be aligned from the lowest bit of the shift register, regardless of whether MSB or LSB is set for the first transfer bit. Any data can be set to the unused higher bits, however, in this case the received data after a serial transfer operation becomes 0.

Figure 11-36. When Transfer Bit Length Other Than 16 Bits Is Set



11.5.2 CSI4 control registers

CSI4 is controlled by the following registers.

- Variable-length serial control register 4 (CSIM4)
- Variable-length serial setting register 4 (CSIB4)
- Baud rate generator source clock selection register 4 (BRGCN4)
- Baud rate generator output clock selection register 4 (BRGCK4)

(1) Variable-length serial control register 4 (CSIM4)

This register is used to enable or disable the serial clock, operation modes, and specific operations of serial interface channel 4.

CSIM4 can be set by an 8- or 1-bit memory manipulation instruction.

RESET input sets CSIM4 to 00H.

After reset:	00H	R/W	Address: FFFFF2E2H					
	7	6	5	4	3	2	1	0
CSIM4	CSIE4	0	0	0	0	MODE4	0	SCL4

CSIE4	SIO4 operation enable/disable specification		
	Shift register operation	Serial counter	Port
0	Operation disabled	Cleared	Port function ^{Note 1}
1	Operation enabled	Count operation enabled	Serial function + port function ^{Note 2}

MODE4	Transfer operation mode flag		
	Operation mode	Transfer start trigger	SO4 output
0	Transmit/receive mode	SIO4 write	Normal output
1	Receive-only mode	SIO4 read	Port function

SCL4	Clock selection
0	External clock input ($\overline{SCK4}$)
1	BRG (Baud rate generator)

Notes 1. When CSIE4 = 0 (SIO4 operation disabled status), the port function is available for the SI4, SO4, and $\overline{SCK4}$ pins.

2. When CSIE4 = 1 (SIO4 operation enabled status), the port function is available for the SI4 pin when using the transmit function only and for the SO4 pin when using the dedicated receive function.

(2) Variable-length serial setting register 4 (CSIB4)

CSIB4 is used to set the operation format of serial interface channel 4.

The bit length of a variable register is set by setting bits 3 to 0 (BSEL3 to BSEL0) of variable-length serial setting register 4. Data is transferred MSB first while bit 4 (DIR) is 1, and is transferred LSB first while DIR is 0.

CSIB4 can be set by an 8- or 1-bit memory manipulation instruction.

RESET input sets CSIB4 to 00H.

After reset : 00H R/W Address: FFFF2E4H

	7	6	5	4	3	2	1	0
CSIB4	0	CMODE	DMODE	DIR	BSEL3	BSEL2	BSEL1	BSEL0

CMODE	DMODE	SCK4 active level	SI4 interrupt timing	SO4 output timing
0	0	Low level	Rising edge of SCK4	Falling edge of SCK4
0	1	Low level	Falling edge of SCK4	Rising edge of SCK4
1	0	High level	Falling edge of SCK4	Rising edge of SCK4
1	1	High level	Rising edge of SCK4	Falling edge of SCK4

DIR	Serial transfer direction
0	LSB first
1	MSB first

BSEL3	BSEL2	BSEL1	BSEL0	Bit length of serial register
0	0	0	0	16 bits
1	0	0	0	8 bits
1	0	0	1	9 bits
1	0	1	0	10 bits
1	0	1	1	11 bits
1	1	0	0	12 bits
1	1	0	1	13 bits
1	1	1	0	14 bits
1	1	1	1	15 bits
Other than above				Setting prohibited

(3) Baud rate generator source clock selection register 4 (BRGCN4)

BRGCN4 can be set by an 8-bit memory manipulation instruction.

RESET input sets BRGCN4 to 00H.

After reset :	00H	R/W	Address: FFFFF2E6H					
	7	6	5	4	3	2	1	0
BRGCN4	0	0	0	0	0	BRGN2	BRGN1	BRGN0

BRGN2	BRGN1	BRGN0	Source clock (fscx)	m
0	0	0	fxx	0
0	0	1	fxx/2	1
0	1	0	fxx/4	2
0	1	1	fxx/8	3
1	0	0	fxx/16	4
1	0	1	fxx/32	5
1	1	0	fxx/64	6
1	1	1	fxx/128	7

(4) Baud rate generator output clock selection register 4 (BRGCK4)

BRGCK4 is set by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets BRGCK4 to 7FH.

After reset : 7FH R/W Address: FFFFF2E8H

	7	6	5	4	3	2	1	0
BRGCK4	0	BRGK6	BRGK5	BRGK4	BRGK3	BRGK2	BRGK1	BRGK0

BRGK6	BRGK5	BRGK4	BRGK3	BRGK2	BRGK1	BRGK0	Baud rate output clock	k
0	0	0	0	0	0	0	Setting prohibited	0
0	0	0	0	0	0	1	f _{sck} /2	1
0	0	0	0	0	1	0	f _{sck} /4	2
0	0	0	0	0	1	1	f _{sck} /6	3
•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•
1	1	1	1	1	1	0	f _{sck} /252	126
1	1	1	1	1	1	1	f _{sck} /254	127

The baud rate transmit/receive clock that is generated is obtained by dividing the main clock.

• **Generation of baud rate transmit/receive clock using main clock**

The transmit/receive clock is obtained by dividing the main clock. The following equation is used to obtain the baud rate from the main clock.

<When 1 ≤ k ≤ 127>

$$[\text{Baud rate}] = \frac{f_{xx}}{2^m \times k \times 2} \text{ [Hz]}$$

- f_{xx}: Main clock oscillation frequency
- m: Value set by BRGN2 to BRGN0 (0 ≤ m ≤ 7)
- k: Value set by BRGK6 to BRGK0 (1 ≤ k ≤ 127)

Caution Do not use the baud rate transmit/receive clock of the variable-length serial I/O (CSI4) with a transfer rate higher than the CPU operation clock. If used with a transfer rate higher than the CPU operation clock, transfer cannot be performed correctly.

11.5.3 Operations

CSI4 has the following two operation modes.

- Operation stopped mode
- 3-wire variable-length serial I/O mode

(1) Operation stopped mode

In this mode, serial transfers are not performed, and therefore power consumption can be reduced. When in operation stopped mode, SI4, SO4, and $\overline{\text{SCK4}}$ can be used as normal I/O ports.

(a) Register settings

Operation stopped mode is set via the CSIE4 bit of variable-length serial control register 4 (CSIM4). While CSIE4 = 0 (SIO4 operation stopped state), the pins connected to SI4, SO4, or $\overline{\text{SCK4}}$ function as port pins.

Figure 11-37. CSIM4 Setting (Operation Stopped Mode)

After reset : 00H	R/W	Address: FFFFF2E2H							
		7	6	5	4	3	2	1	0
CSIM4		CSIE4	0	0	0	0	MODE4	0	SCL4
		SIO4 operation enable/disable specification							
	CSIE4	Shift register operation		Serial counter			Port		
	0	Operation disabled		Cleared			Port function		

(2) 3-wire variable-length serial I/O mode

3-wire variable-length serial I/O mode is useful when connecting to a peripheral I/O device that includes a clocked serial interface, a display controller, etc.

This mode executes data transfers via three lines: a serial clock line ($\overline{\text{SCK4}}$), a serial output line (SO4), and a serial input line (SI4).

(a) Register settings

3-wire variable-length serial I/O mode is set using variable-length serial control register 4 (CSIM4).

Figure 11-38. CSIM4 Setting (3-Wire Variable-Length Serial I/O Mode)

After reset : 00H	R/W	Address: FFFF2E2H							
		7	6	5	4	3	2	1	0
CSIM4		CSIE4	0	0	0	0	MODE4	0	SCL4
		SIO4 operation enable/disable specification							
		Shift register operation		Serial counter		Port			
		1	Operation enabled		Count operation enabled		Serial function + port function		
		Transfer operation mode flag							
		Operation mode		Transfer start trigger		SO4 output			
		0	Transmit/receive mode		Write to SIO4		Normal output		
		1	Receive-only mode		Read from SIO4		Port function		
		Serial clock selection							
		0	External clock input ($\overline{\text{SCK4}}$)						
		1	BRG (Baud rate generator)						

The bit length of a variable-length register is set by setting bits 3 to 0 (BSEL3 to BSEL0) of CSIB4. Data is transferred MSB first while bit 4 (DIR) is 1, and is transferred LSB first while DIR is 0.

Figure 11-39. CSIB4 Setting (3-Wire Variable-Length Serial I/O Mode)

After reset : 00H R/W Address: FFFFF2E4H

	7	6	5	4	3	2	1	0
CSIB4	0	CMODE	DMODE	DIR	BSEL3	BSEL2	BSEL1	BSEL0

CMODE	DMODE	$\overline{\text{SCK4}}$ active level	SI4 interrupt timing	SO4 output timing
0	0	Low level	Rising edge of $\overline{\text{SCK4}}$	Falling edge of $\overline{\text{SCK4}}$
0	1	Low level	Falling edge of $\overline{\text{SCK4}}$	Rising edge of $\overline{\text{SCK4}}$
1	0	High level	Falling edge of $\overline{\text{SCK4}}$	Rising edge of $\overline{\text{SCK4}}$
1	1	High level	Rising edge of $\overline{\text{SCK4}}$	Falling edge of $\overline{\text{SCK4}}$

DIR	Serial transfer direction
0	LSB first
1	MSB first

BSEL3	BSEL2	BSEL1	BSEL0	Bit length of serial register
0	0	0	0	16 bits
1	0	0	0	8 bits
1	0	0	1	9 bits
1	0	1	0	10 bits
1	0	1	1	11 bits
1	1	0	0	12 bits
1	1	0	1	13 bits
1	1	1	0	14 bits
1	1	1	1	15 bits
Other than above				Setting prohibited

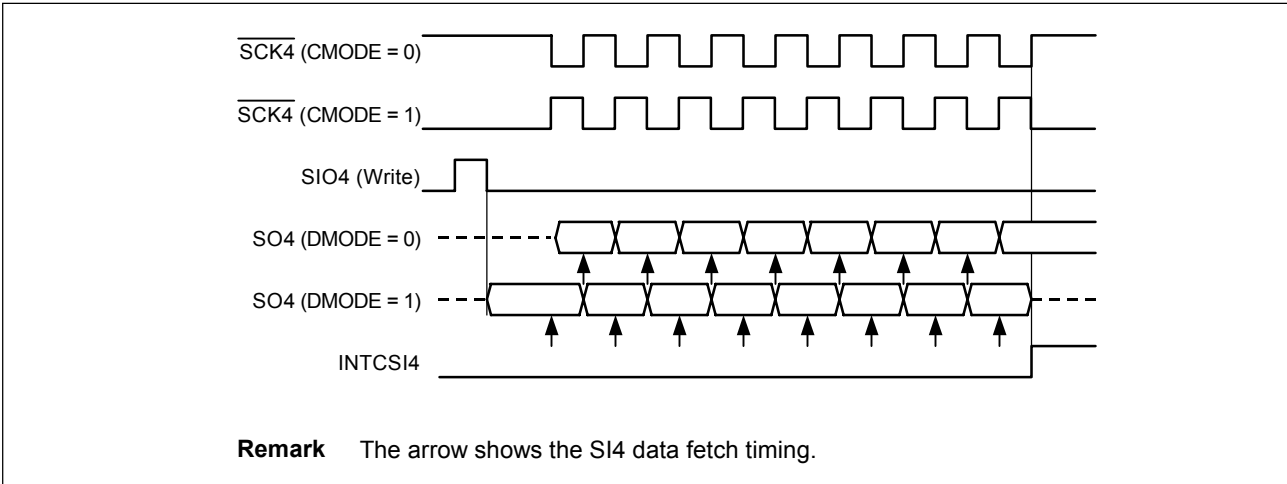
(b) Communication operations

In 3-wire variable-length serial I/O mode, data is transmitted and received in 8- to 16-bit units, and is specified by setting bits 3 to 0 (BSEL3 to BSEL0) of variable-length serial setting register 4 (CSIB4). Each bit of data is transmitted or received in synchronization with the serial clock.

After transfer of all bits is complete, SIO4 stops operation automatically and the interrupt request flag (INTCSI4) is set.

Bits 6 and 5 (CMODE and DMODE) of variable-length serial setting register 4 (CSIB4) can change the attribute of the serial clock ($\overline{SCK4}$) and the phases of serial data (SI4 and SO4).

Figure 11-40. Timing of 3-Wire Variable-Length Serial I/O Mode



When CMODE = 0, the serial clock ($\overline{SCK4}$) stops at a high level while the operation is stopped, and outputs a low level during a data transfer operation. When CMODE = 1, on the other hand, $\overline{SCK4}$ stops at a low level while the operation is stopped and outputs a high level during a data transfer operation.

The phases of the SO4 output timing and the S14 fetch timing can be shifted half a clock by setting DMODE.

However, the interrupt signal (INTCSI4) is generated at the final edge of the serial clock ($\overline{SCK4}$), regardless of the setting of CSIB4.

(c) Transfer start

A serial transfer becomes possible when the following two conditions have been satisfied.

- The SIO4 operation control bit (CSIE4) = 1
- After a serial transfer, the internal serial clock is stopped.

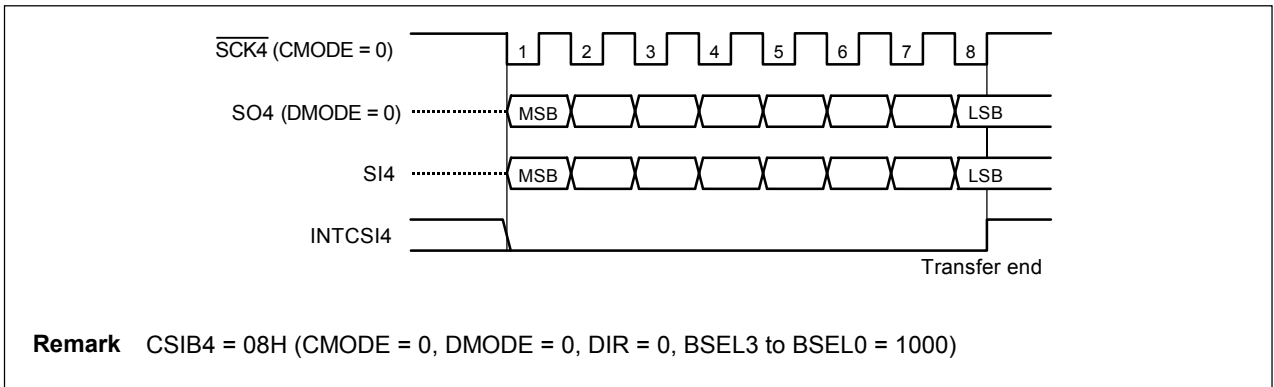
Serial transfer starts when the following operation is performed after the above two conditions have been satisfied.

- Transmit/transmit and receive mode (MODE4 = 0)
Transfer starts when writing to SIO4.
- Receive-only mode (MODE4 = 1)
Transfer starts when reading from SIO4.

Caution After data has been written to SIO4, transfer will not start even if the CSIE4 bit is set to 1.

Completion of the final-bit transfer automatically stops the serial transfer operation and sets the interrupt request flag (INTCSI4).

Figure 11-41. Timing of 3-Wire Variable-Length Serial I/O Mode (When CSIB4 = 08H)



CHAPTER 12 A/D CONVERTER

12.1 Function

The A/D converter converts analog input signals into digital values, has a resolution of 10 bits, and can handle 12 channels of analog input signals (ANI0 to ANI11).

The V850/SF1 supports low-speed conversion and a low-power consumption mode.

(1) Hardware start

Conversion is started by trigger input (ADTRG) (rising edge, falling edge, or both rising and falling edges can be specified).

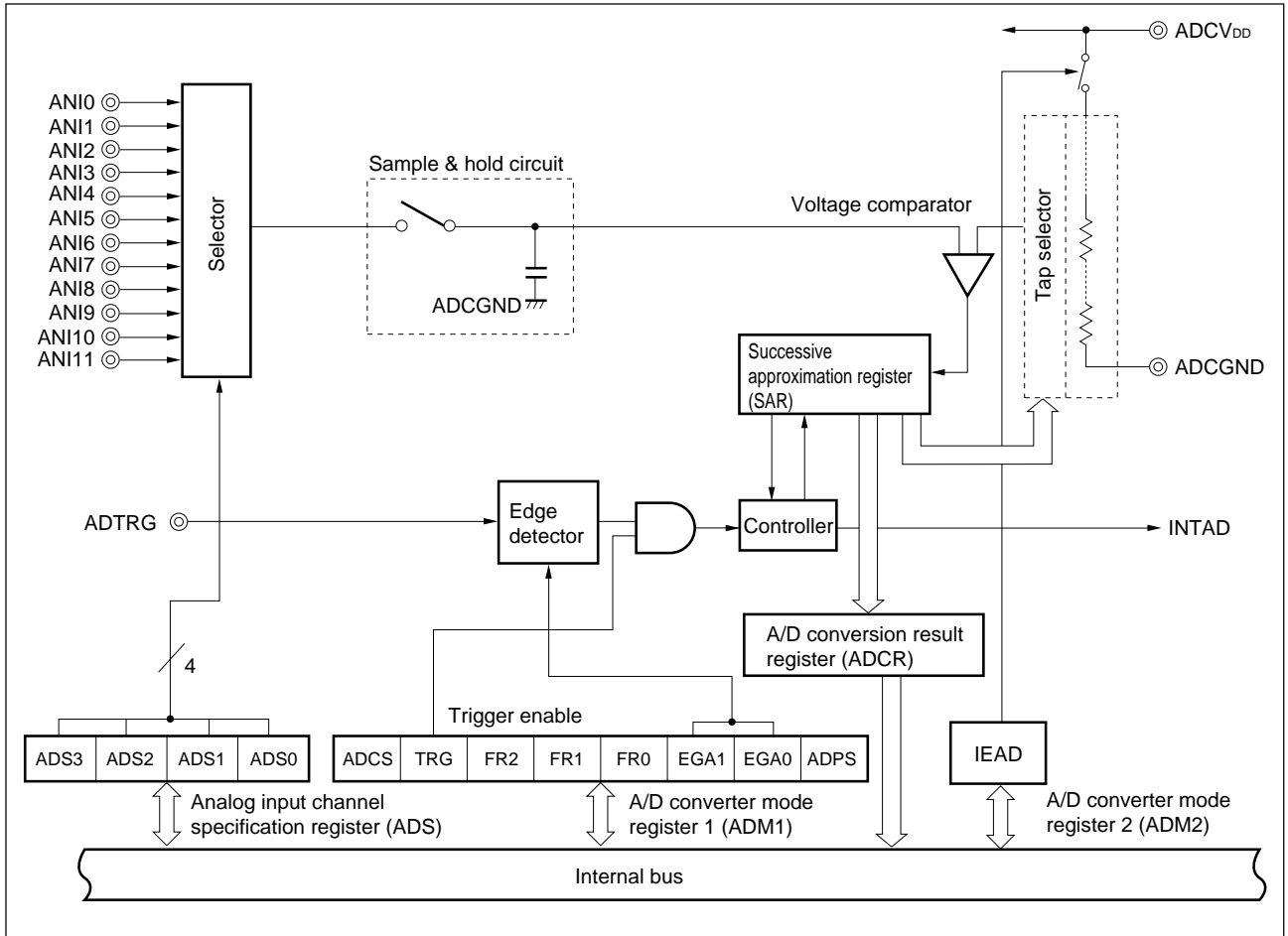
(2) Software start

Conversion is started by setting A/D converter mode register 1 (ADM1).

One analog input channel is selected from ANI0 to ANI11, and A/D conversion is performed. If A/D conversion has been started by means of a hardware start, conversion stops once it has been completed, and an interrupt request (INTAD) is generated. If conversion has been started by means of a software start, conversion is performed repeatedly. Each time conversion has been completed, INTAD is generated.

The block diagram is shown below.

Figure 12-1. Block Diagram of A/D Converter



12.2 Configuration

The A/D converter includes the following hardware units.

Table 12-1. Configuration of A/D Converter

Item	Configuration
Analog input	12 channels (ANI0 to ANI11)
Registers	Successive approximation register (SAR) A/D conversion result register (ADCR) A/D conversion result register H (ADCRH): Only higher 8 bits can be read
Control registers	A/D converter mode register 1 (ADM1) A/D converter mode register 2 (ADM2) Analog input channel specification register (ADS)

(1) Successive approximation register (SAR)

This register compares the voltage value of the analog input signal with the voltage tap (compare voltage) value from the series resistor string, and holds the result of the comparison starting from the most significant bit (MSB). When the comparison result has been obtained down to the least significant bit (LSB) (i.e., when A/D conversion is complete), the contents of the SAR are transferred to the A/D conversion result register.

(2) A/D conversion result register (ADCR), A/D conversion result register H (ADCRH)

Each time A/D conversion has been completed, the result of the conversion is loaded to this register from the successive approximation register. The higher 10 bits of this register hold the result of the A/D conversion (the lower 6 bits are fixed to 0).

This register is read by a 16-bit memory manipulation instruction. $\overline{\text{RESET}}$ input sets ADCR to 0000H.

When using only the higher 8 bits of the result of the A/D conversion, ADCRH is read by an 8-bit memory manipulation instruction. $\overline{\text{RESET}}$ input sets ADCRH to 00H.

Caution A write operation to A/D converter mode register 1 (ADM1) and the analog input channel specification register (ADS) may cause the ADCR contents to be undefined. After the conversion, read out the conversion result before writing to ADM1 and ADS. Correct conversion results may not be read if the timing is other than the above.

(3) Sample & hold circuit

The sample & hold circuit samples each of the analog input signals sequentially sent from the input circuit, and sends the sampled data to the voltage comparator. This circuit also holds the sampled analog input signal voltage during A/D conversion.

(4) Voltage comparator

The voltage comparator compares the analog input signal with the output voltage of the series resistor string.

(5) Series resistor string

The series resistor string is connected between ADCV_{DD} and ADCGND and generates a voltage for comparison with the analog input signal.

(6) ANI0 to ANI11 pins

These are analog input pins for the 12 channels of the A/D converter, and are used to input the analog signals to be converted into digital signals. Pins other than ones selected for analog input using the analog input channel specification register (ADS) can be used as input ports.

Caution Make sure that the voltages input to ANI0 through ANI11 do not exceed the rated values. If a voltage higher than ADC_{VDD} or lower than $ADCGND$ (even within the range of the absolute maximum ratings) is input to a channel, the conversion value of the channel becomes undefined, and the conversion values of the other channels may also be affected.

(7) ADCGND pin

This is the ground pin of the A/D converter. Always make the potential at this pin the same as that at the GND0 pin even when the A/D converter is not in use.

(8) ADC_{VDD} pin

This is the analog power supply pin of the A/D converter. Always make the potential at this pin the same as that at the V_{DD0} pin even when the A/D converter is not in use.

12.3 Control Registers

The A/D converter is controlled by the following registers.

- A/D converter mode register 1 (ADM1)
- Analog input channel specification register (ADS)
- A/D converter mode register 2 (ADM2)

(1) A/D converter mode register 1 (ADM1)

This register specifies the conversion time of the input analog signal to be converted into a digital signal, starting or stopping the conversion, and an external trigger.

ADM1 is set by an 8- or 1-bit memory manipulation instruction.

RESET input sets ADM1 to 00H.

(1/2)

After reset: 00H	R/W	Address: FFFFF3C0H							
		7	6	5	4	3	2	1	0
ADM1		ADCS	TRG	FR2	FR1	FR0	EGA1	EGA0	ADPS
	ADCS	A/D conversion control							
	0	Conversion stopped							
	1	Conversion enabled							
	TRG	Software start or hardware start selection							
	0	Software start							
	1	Hardware start							

After reset: 00H R/W Address: FFFF3C0H

	7	6	5	4	3	2	1	0
ADM1	ADCS	TRG	FR2	FR1	FR0	EGA1	EGA0	ADPS

ADPS	FR2	FR1	FR0	Selection of conversion time		
				Conversion time ^{Note 1} + stabilization time ^{Note 2}	fxx	
					16 MHz	8 MHz
0	0	0	0	168/fxx	Setting prohibited	Setting prohibited
0	0	0	1	120/fxx	7.5 μs	Setting prohibited
0	0	1	0	84/fxx	5.25 μs	Setting prohibited
0	0	1	1	60/fxx	Setting prohibited	7.5 μs
0	1	0	0	48/fxx	Setting prohibited	5.25 μs
0	1	0	1	36/fxx	Setting prohibited	Setting prohibited
0	1	1	0	Setting prohibited	Setting prohibited	Setting prohibited
0	1	1	1	12/fxx	Setting prohibited	Setting prohibited
1	0	0	0	168/fxx + 64/fxx	Setting prohibited	Setting prohibited
1	0	0	1	120/fxx + 60/fxx	7.5 + 4.2 μs	Setting prohibited
1	0	1	0	84/fxx + 42/fxx	5.25 + 3.0 μs	Setting prohibited
1	0	1	1	60/fxx + 30/fxx	Setting prohibited	7.5 + 4.2 μs
1	1	0	0	48/fxx + 24/fxx	Setting prohibited	5.25 + 3.0 μs
1	1	0	1	36/fxx + 18/fxx	Setting prohibited	Setting prohibited
1	1	1	0	Setting prohibited	Setting prohibited	Setting prohibited
1	1	1	1	12/fxx + 6/fxx	Setting prohibited	Setting prohibited

EGA1	EGA0	Valid edge specification for external trigger signal
0	0	No edge detection
0	1	Detects at falling edge
1	0	Detects at rising edge
1	1	Detects at both rising and falling edges

ADPS	Comparator control while A/D conversion is stopped (ADCS = 0)
0	Comparator ON
1	Comparator OFF

- ★
- Notes**
1. Conversion time (actual A/D conversion time).
Always set the time to $5 \mu s \leq \text{Conversion time} \leq 10 \mu s$.
 2. Stabilization time (setup time of A/D converter)
Each A/D conversion requires “conversion time + stabilization time”. There is no stabilization time when ADPS = 0.

- ★
- Cautions**
1. The A/D converter cannot be used when the operation frequency is 2.4 to 3.6 MHz.
 2. Cut the current consumption by setting the ADPS bit to 1 when the ADCS bit is set to 0.

(2) Analog input channel specification register (ADS)

ADS specifies the port for inputting the analog voltage to be converted into a digital signal.

ADS is set by an 8- or 1-bit memory manipulation instruction.

RESET input sets ADS to 00H.

After reset:	00H	R/W	Address: FFFF3C2H					
	7	6	5	4	3	2	1	0
ADS	0	0	0	0	ADS3	ADS2	ADS1	ADS0
	ADS3	ADS2	ADS1	ADS0	Analog input channel specification			
	0	0	0	0	ANI0			
	0	0	0	1	ANI1			
	0	0	1	0	ANI2			
	0	0	1	1	ANI3			
	0	1	0	0	ANI4			
	0	1	0	1	ANI5			
	0	1	1	0	ANI6			
	0	1	1	1	ANI7			
	1	0	0	0	ANI8			
	1	0	0	1	ANI9			
	1	0	1	0	ANI10			
	1	0	1	1	ANI11			
	Other than above				Setting prohibited			

(3) A/D converter mode register 2 (ADM2)

ADM2 specifies connection/disconnection of ADCV_{DD} and the series resistor string.

ADM2 is set by an 8- or 1-bit memory manipulation instruction.

RESET input sets ADM2 to 00H.

After reset:	00H	R/W	Address: FFFF3C8H					
	7	6	5	4	3	2	1	0
ADM2	0	0	0	0	0	0	0	IEAD
	IEAD	A/D current cut control						
	0	Cut between ADCV _{DD} and series resistor string						
	1	Connect between ADCV _{DD} and series resistor string						

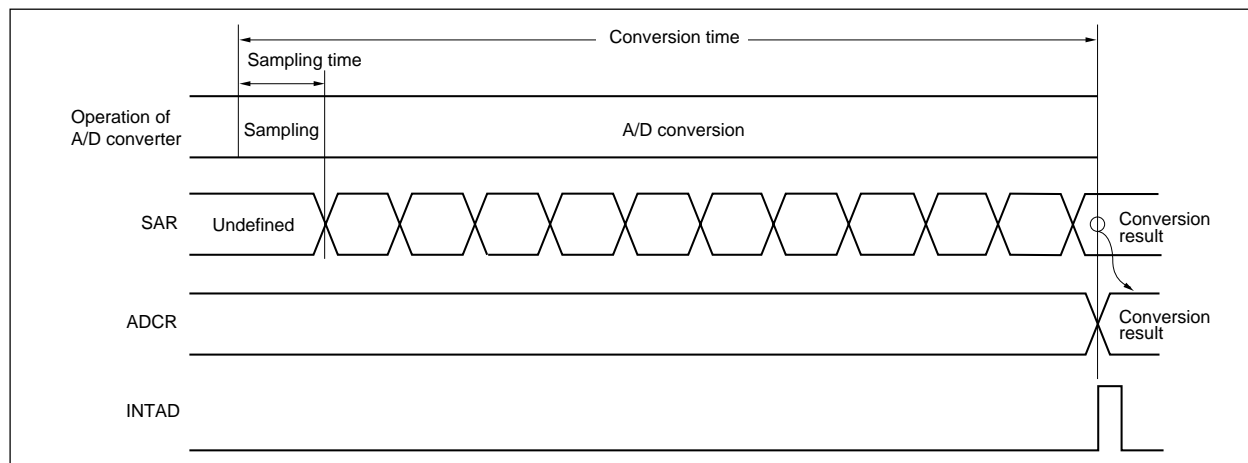
12.4 Operation

12.4.1 Basic operation

- <1> Select one channel whose analog signal is to be converted into a digital signal by using the analog input channel specification register (ADS).
- <2> The sample & hold circuit samples the voltage input to the selected analog input channel.
- <3> After sampling for a specific time, the sample & hold circuit enters the hold status, and holds the input analog voltage until it has been converted into a digital signal.
- <4> Set bit 9 of the successive approximation register (SAR). The tap selector sets the voltage tap of the series resistor string to $(1/2) ADCV_{DD}$.
- <5> The voltage difference between the voltage tap of the series resistor string and the analog input voltage is compared by the voltage comparator. If the analog input voltage is greater than $(1/2) ADCV_{DD}$, the MSB of the SAR remains set. If the analog input voltage is less than $(1/2) ADCV_{DD}$, the MSB is reset.
- <6> Next, bit 8 of the SAR is automatically set, and the analog input voltage is compared again. Depending on the value of bit 9 to which the result of the preceding comparison has been set, the voltage tap of the series resistor string is selected as follows:
 - Bit 9 = 1: $(3/4) ADCV_{DD}$
 - Bit 9 = 0: $(1/4) ADCV_{DD}$The analog input voltage is compared with one of these voltage taps, and bit 8 of the SAR is manipulated as follows depending on the result of the comparison.
 - Analog input voltage \geq voltage tap: Bit 8 = 1
 - Analog input voltage \leq voltage tap: Bit 8 = 0
- <7> The above steps are repeated until bit 0 of the SAR has been manipulated.
- <8> When comparison of all 10 bits of the SAR has been completed, the valid digital value remains in the SAR, and the value of the SAR is transferred and latched to the A/D conversion result register (ADCR). At the same time, an A/D conversion end interrupt request (INTAD) can be generated.

Caution The first conversion value immediately after starting the A/D conversion may not satisfy the ratings.

Figure 12-2. Basic Operation of A/D Converter



A/D conversion is successively executed until bit 7 (ADCS) of A/D converter mode register 1 (ADM1) is reset to 0 by software.

If ADM1 and the analog input channel specification register (ADS) are written during A/D conversion, the conversion is initialized. If ADCS is set to 1 at this time, conversion is started from the beginning.

$\overline{\text{RESET}}$ input sets the A/D conversion result register (ADCR) to 0000H.

12.4.2 Input voltage and conversion result

The analog voltages input to the analog input pins (ANI0 to ANI11) and the result of the A/D conversion (contents of the A/D conversion result register (ADCR)) are related as follows:

$$ADCR = \text{INT}\left(\frac{V_{IN}}{ADCV_{DD}} \times 1024 + 0.5\right)$$

Or,

$$(ADCR - 0.5) \times \frac{ADCV_{DD}}{1024} \leq V_{IN} < (ADCR + 0.5) \times \frac{ADCV_{DD}}{1024}$$

INT (): Function that returns integer of value enclosed in parentheses

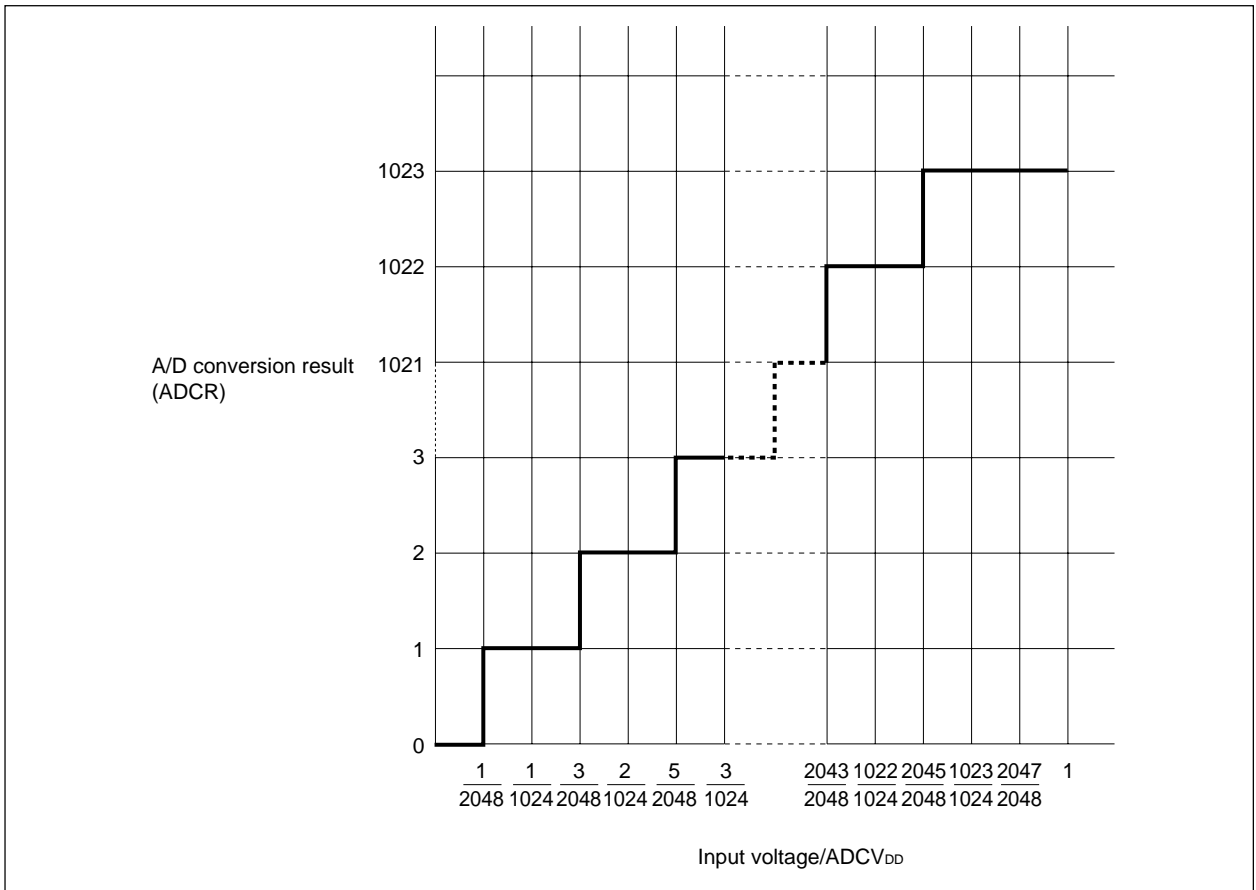
V_{IN}: Analog input voltage

ADCV_{DD}: A/D converter reference voltage

ADCR: Value of the A/D conversion result register (ADCR)

The relationship between the analog input voltage and A/D conversion result is shown below.

Figure 12-3. Relationship Between Analog Input Voltage and A/D Conversion Result



12.4.3 A/D converter operation mode

In this mode one of the analog input channels ANI0 to ANI11 is selected by the analog input channel specification register (ADS) and A/D conversion is executed.

A/D conversion can be started in the following two ways.

- Hardware start: Started by trigger input (ADTRG) (rising edge, falling edge, or both rising and falling edges can be specified)
- Software start: Started by setting A/D converter mode register 1 (ADM1)

The result of the A/D conversion is stored in the A/D conversion result register (ADCR) and an interrupt request signal (INTAD) is generated at the same time.

(1) A/D conversion by hardware start

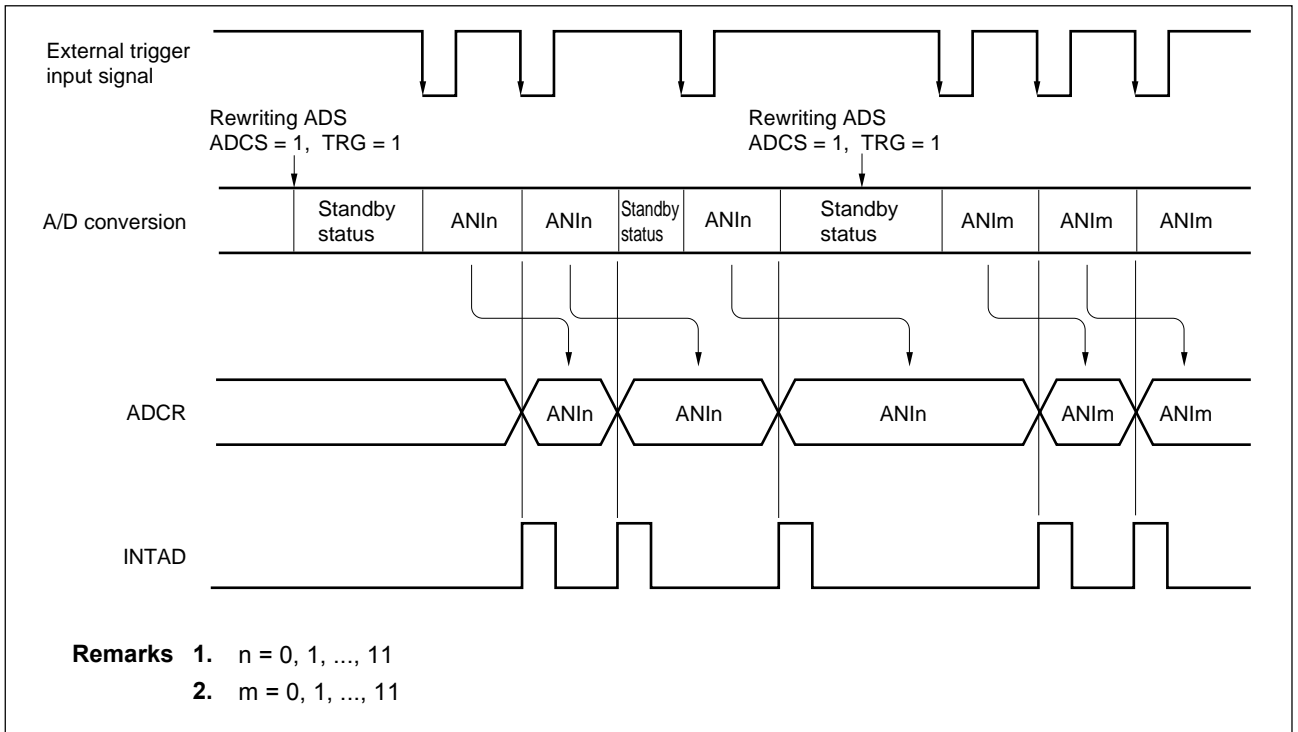
A/D conversion is on standby if bit 6 (TRG) and bit 7 (ADCS) of A/D converter mode register 1 (ADM1) are set to 1. When an external trigger signal is input, the A/D converter starts converting the voltage applied to the analog input pin specified by the analog input channel specification register (ADS) into a digital signal.

When the A/D conversion has been completed, the result of the conversion is stored in the A/D conversion result register (ADCR), and an interrupt request signal (INTAD) is generated. Once the A/D conversion has been started and completed, conversion is not started again unless a new external trigger signal is input.

If data with ADCS set to 1 is written to ADM during A/D conversion, the conversion under execution is stopped, and the A/D converter stands by until a new external trigger signal is input. If the external trigger signal is input, A/D conversion is executed again from the beginning.

If data with ADCS set to 0 is written to ADM1 during A/D conversion, the conversion is immediately stopped.

Figure 12-4. A/D Conversion by Hardware Start (with Falling Edge Specified)



(2) A/D conversion by software start

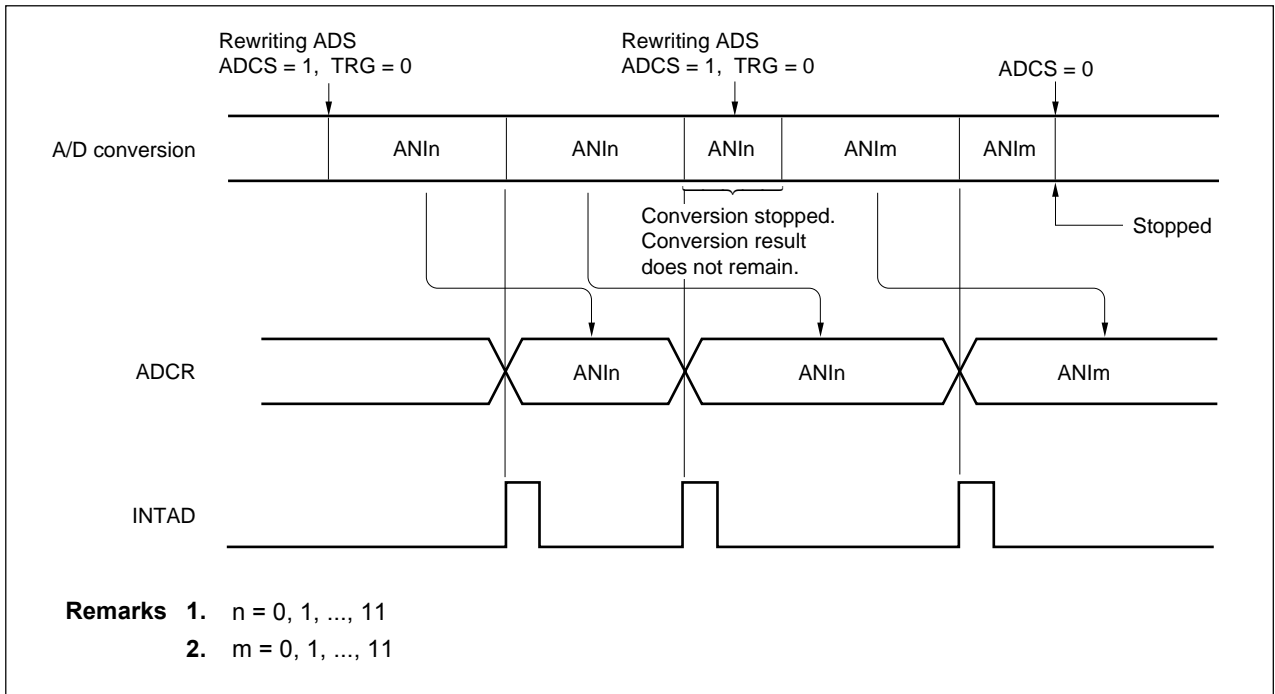
If bit 6 (TRG) and bit 7 (ADCS) of A/D converter mode register 1 (ADM1) are set to 1, the A/D converter starts converting the voltage applied to the analog input pin specified by the analog input channel specification register (ADS) into a digital signal.

When the A/D conversion has been completed, the result of the conversion is stored in the A/D conversion result register (ADCR), and an interrupt request signal (INTAD) is generated. Once A/D conversion has been started and completed, the next conversion is started immediately. A/D conversion is repeated until new data is written to ADS.

If ADS is rewritten during A/D conversion, the conversion under execution is stopped, and conversion of the newly selected analog input channel is started.

If data with ADCS set to 0 is written to ADM1 during A/D conversion, the conversion is immediately stopped.

Figure 12-5. A/D Conversion by Software Start



12.5 Low Power Consumption Mode

The V850/SF1 features a function that can cut or connect the current between ADCV_{DD} and the series resistor string. Switching can be performed by setting A/D converter mode register 2 (ADM2).

★ When not using the A/D converter, cut off the tap selector (a function to reduce current) from the voltage supply block (ADCV_{DD}) while A/D conversion is stopped (ADCS = 0) to cut the current consumption.

- Set the ADPS bit of A/D converter mode register 1 (ADM1) to 1.
- Set the IEAD bit of A/D converter mode register 2 (ADM2) to 0.

When the ADPS bit is reset to 0 (comparator on), stabilization time (5 μ s max.) is required before starting A/D conversion. Therefore, secure a wait of at least 5 μ s by software.

12.6 Cautions

(1) Current consumption in standby mode

The A/D converter stops operation in the STOP and IDLE modes (it can be operated in the HALT mode). At this time, the current consumption of the A/D converter can be reduced by stopping the conversion (by resetting bit 7 (ADCS) of A/D converter mode register 1 (ADM1) to 0).

(2) Input range of ANI0 to ANI11

Keep the input voltage of the ANI0 through ANI11 pins to within the rated range. If a voltage greater than ADCV_{DD} or lower than ADCGND (even within the range of the absolute maximum ratings) is input to a channel, the converted value of the channel becomes undefined. Moreover, the values of the other channels may also be affected.

(3) Conflict

<1> Conflict between writing A/D conversion result register (ADCR) and reading ADCR at end of conversion

Reading ADCR takes precedence. After ADCR has been read, a new conversion result is written to ADCR.

<2> Conflict between writing ADCR and external trigger signal input at end of conversion

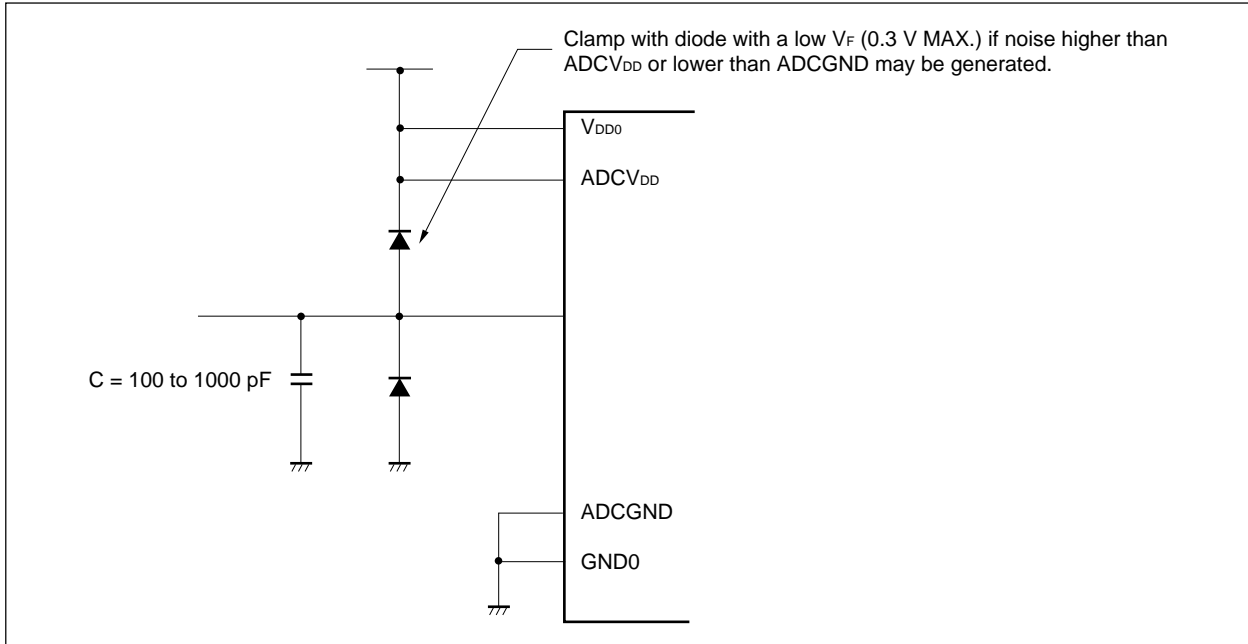
The external trigger signal is not input during A/D conversion. Therefore, the external trigger signal is not accepted while writing ADCR.

<3> Conflict between writing of ADCR and writing A/D converter mode register 1 (ADM1) or analog input channel specification register (ADS)

When ADM1 or ADS is written immediately after ADCR is written following the end of A/D conversion, the conversion result is written to the ADCR register, but the timing is such that INTAD is not generated.

(4) Countermeasures against noise

To keep the resolution of 10 bits, prevent noise from being superimposed on the ANI0 to ANI11 pins. The higher the output impedance of the analog input source, the heavier the influence of noise. To lower noise, connecting an external capacitor as shown in Figure 12-6 is recommended.

Figure 12-6. Handling of Analog Input Pin**(5) ANI0 to ANI11**

The analog input (ANI0 to ANI11) pins are also used as port pins.

To execute A/D conversion with any of ANI0 to ANI11 selected, do not execute an instruction that inputs data to a port during conversion; otherwise, the resolution may drop.

If a digital pulse is applied to pins adjacent to the pin whose input signal is converted into a digital signal, the expected A/D conversion result may not be obtained because of the influence of coupling noise. Therefore, do not apply a pulse to adjacent pins.

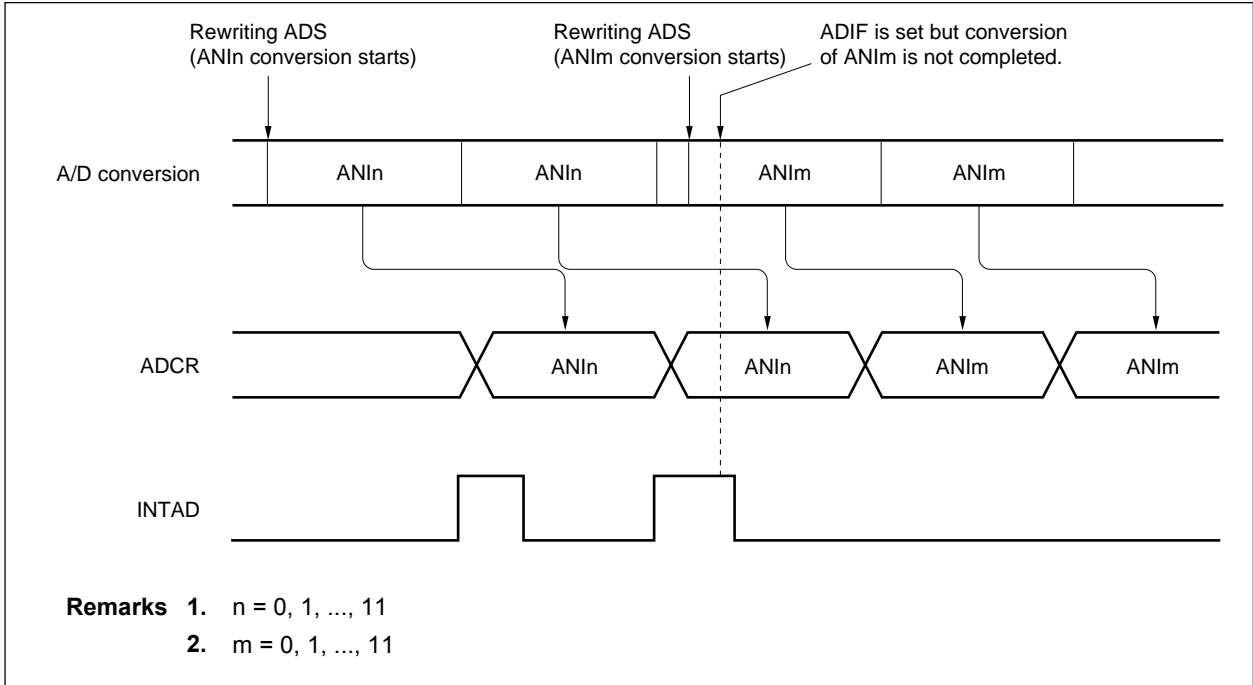
(6) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even if the contents of the analog input channel specification register (ADS) are changed.

If the analog input pin is changed during conversion, therefore, the result of the A/D conversion of the preceding analog input signal and the conversion end interrupt request flag may be set immediately before ADS is rewritten. If ADIF is read immediately after ADS has been rewritten, it may be set despite the fact that conversion of the newly selected analog input signal has not been completed yet.

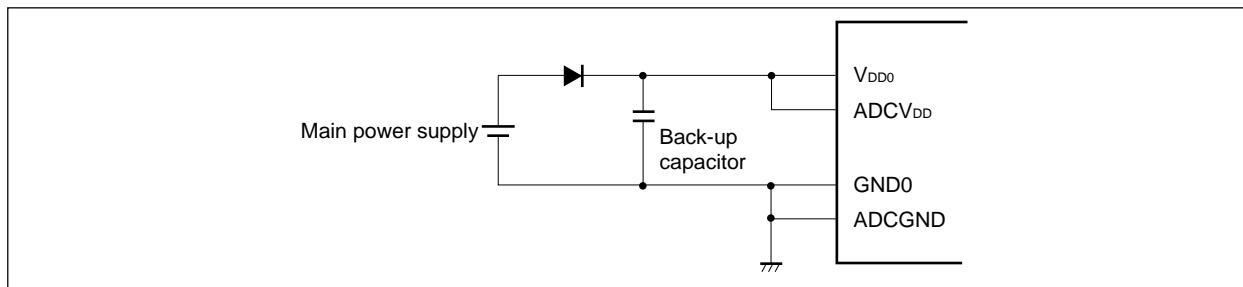
When stopping A/D conversion and then resuming, clear ADIF before resuming conversion.

Figure 12-7. A/D Conversion End Interrupt Generation Timing



(7) ADCV_{DD} pin

The ADCV_{DD} pin is the power supply pin of the analog circuit, and also supplies power to the input circuit of ANI0 to ANI11. Even in an application where a back-up power supply is used, therefore, be sure to apply the same voltage as the V_{DD0} pin to the ADCV_{DD} pin as shown in Figure 12-8.

Figure 12-8. Handling of ADCV_{DD} Pin**(8) Reading A/D converter result register (ADCR)**

Writing to A/D converter mode register 1 (ADM1) and analog input channel specification register (ADS) may cause the ADCR contents to be undefined. After the conversion, read out the conversion result before writing to ADM1 and ADS. Incorrect conversion results may be read out at timings other than the above.

CHAPTER 13 DMA FUNCTIONS

13.1 Functions

The DMA (Direct Memory Access) controller transfers data between memory and peripheral I/Os based on DMA requests sent from on-chip peripheral hardware (such as a serial interface, timer, or A/D converter).

This product includes six independent DMA channels that can transfer data in 8-bit and 16-bit units. The maximum number of transfers is 256 (when transferring data in 8-bit units).

After a DMA transfer has occurred a specified number of times, DMA transfer completion interrupt (INTDMA0 to INTDMA5) requests are output individually from the various channels.

The priority levels of the DMA channels are fixed as follows for simultaneous generation of multiple DMA transfer requests.

DMA0 > DMA1 > DMA2 > DMA3 > DMA4 > DMA5

13.2 Transfer Completion Interrupt Request

After a DMA transfer has occurred a specified number of times and the TCn bit in corresponding DMA channel control registers 0 to 5 (DCHC0 to DCHC5) has been set to 1, a DMA transfer completion interrupt request (INTDMA0 to INTDMA5) to the interrupt controller occurs in each channel.

13.3 Control Registers

Remark n = 0 to 5 in section 13.3.

13.3.1 DMA peripheral I/O address registers 0 to 5 (DIOA0 to DIOA5)

These registers are used to set the peripheral I/O register address for DMA channel n.

These registers can be read/written in 16-bit units.

After reset:	Undefined	R/W	Address:	DIOA0	FFFFF180H	DIOA3	FFFFF1B0H		
				DIOA1	FFFFF190H	DIOA4	FFFFF1C0H		
				DIOA2	FFFFF1A0H	DIOA5	FFFFF1D0H		
	15	14	13	12	11	10	9	1	0
DIOAn	0	0	0	0	0	0	IOAn9 to IOAn1	0	

Caution The following peripheral I/O registers must not be set.
P4, P5, P6, P9, P11, PM4, PM5, PM6, PM9, PM11, MM, DWC, BCC, PSC, PCC, SYS, PRCMD, DIOAn, DRAn, DBCn, DCHCn, CORCN, CORRQ, CORADn, interrupt control register (xxICn), ISPR, POCS, VM45, FCAN register (see CHAPTER 18)

13.3.2 DMA internal RAM address registers 0 to 5 (DRA0 to DRA5)

These registers set DMA channel n internal RAM addresses.

An address is incremented after each transfer is completed, when the DADn bit of the DCHCn register is 0. The incrementation value is “1” for 8-bit transfer and “2” for 16-bit transfer.

These registers can be read/written in 16-bit units.

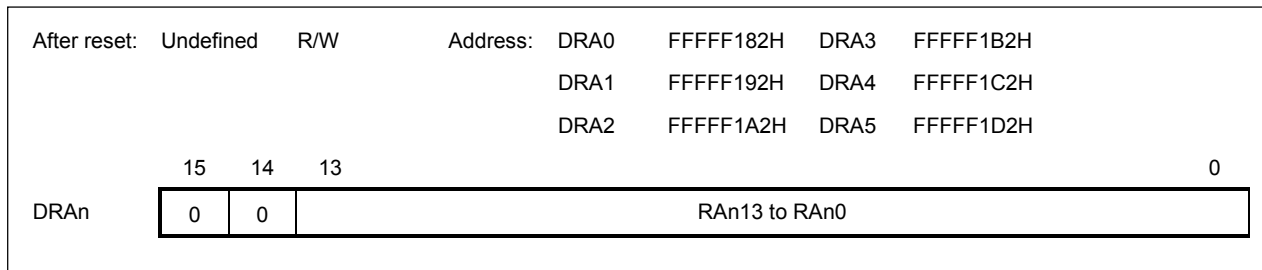
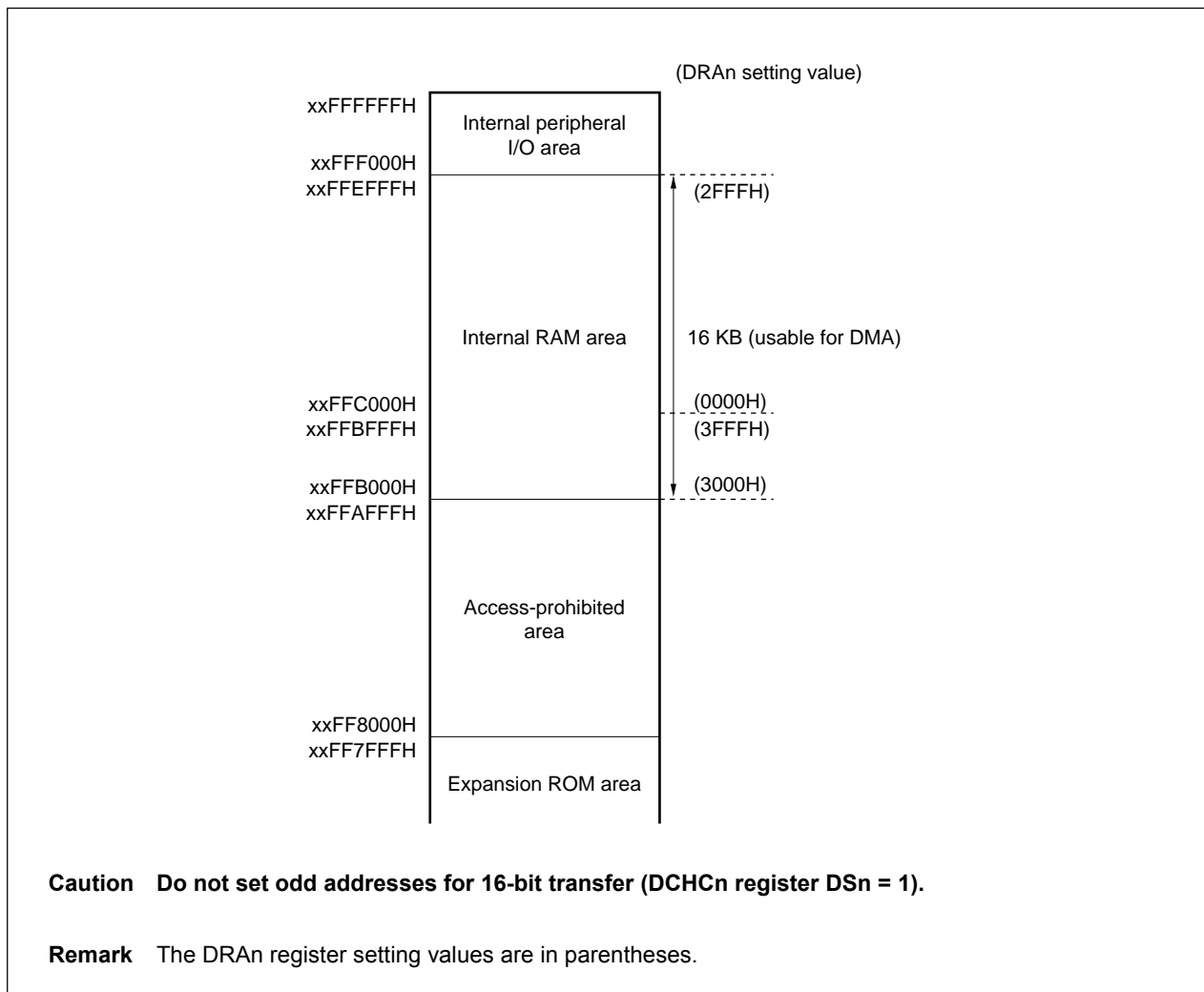


Figure 13-1. Correspondence Between DRAn Setting Value and Internal RAM



Caution Do not set odd addresses for 16-bit transfer (DCHCn register DS_n = 1).

Remark The DRAn register setting values are in parentheses.

13.3.5 DMA channel control registers 0 to 5 (DCHC0 to DCHC5)

These registers are used to control the DMA transfer operation mode for DMA channel n.

Refer to 13.3.6 Start factor settings for the TTYPn1 and TTYPn0 bit settings.

These registers can be read/written in 8- or 1-bit units.

After reset:	00H	R/W	Address:	DCHC0	FFFFF186H	DCHC3	FFFFF1B6H	
				DCHC1	FFFFF196H	DCHC4	FFFFF1C6H	
				DCHC2	FFFFF1A6H	DCHC5	FFFFF1D6H	
	7	6	5	4	3	2	1	0
DCHCn	TCn	0	DDADn	TTYPn1	TTYPn0	TDIRn	DSn	ENn

TCn	DMA transfer completed/not completed ^{Note 1}
0	Not completed
1	Completed

DDADn	Internal RAM address count direction control
0	Incremented
1	Address is fixed

TDIRn	Transfer direction control between peripheral I/Os and internal RAM ^{Note 2}
0	From internal RAM to peripheral I/Os
1	From peripheral I/Os to internal RAM

DSn	Control of transfer data size for DMA transfer ^{Note 2}
0	8-bit transfer
1	16-bit transfer

ENn	Control of DMA transfer enable/disable status ^{Note 3}
0	Disabled
1	Enabled (reset to 0 after DMA transfer is complete)

- Notes**
1. TCn (n = 0 to 5) is set to 1 when a specified number of transfers are complete, and is cleared to 0 when a write instruction is executed.
 2. Make sure that the transfer format conforms to the peripheral I/O register specifications (access-enabled data size, read/write, etc.) for the DMA peripheral I/O address register (DIOAn).
 3. After the specified number of transfers is complete, this bit is cleared to 0.

13.3.6 Start factor settings

The DMA start factor is set using bits 2 to 0 (DMAS2 to DMAS0) of the DMA start factor expansion register (DMAS) in combination with bits 4 and 3 (TTYPn1, TTYPn0) of DMA channel control registers 0 to 5 (DCHC0 to DCHC5).

Table 13-1 shows the DMA start factor settings.

Table 13-1. Start Factor Settings

Channel n	DMAS2	DMAS1	DMAS0	TTYPn1	TTYPn0	DMA Transfer Start Factor Settings
0	x	x	x	0	0	INTCSI0/INTIIC0
				0	1	INTCSI1/INTSR0
				1	0	INTAD
				1	1	INTTM00
1	x	x	0	0	0	INTCSI0/INTIIC0
			1	0	0	INTCSI1/INTSR0
			x	0	1	INTST0
			1	0	0	INTP0
			1	1	1	INTTM10
2	x	0	x	0	0	INTCSI4
		1		0	0	INTCSI3/INTSR1
		x		0	1	INTP6
		1		0	0	Setting prohibited
		1		1	1	INTAD
3	0	x	x	0	0	INTTM3
	1			0	0	INTCSI3/INTSR1
	x			0	1	INTTM5
	1			0	0	Setting prohibited
	1			1	1	INTTM4
4	x	x	x	0	0	INTST1
				0	1	INTCSI4
				1	0	INTAD
				1	1	INTTM2
5	x	x	x	0	0	INTCSI3/INTSR1
				0	1	INTCSI4
				1	0	INTTM70
				1	1	INTTM6

- Remarks**
1. DMAS2 to DMAS0: Bits 2 to 0 of the DMA start factor expansion register (DMAS)
 2. TTYPn1, TTYPn0: Bits 4 and 3 of DMA channel control register n (DCHCn)
 3. x: don't care

CHAPTER 14 RESET FUNCTION

14.1 General

When a low-level input occurs at the $\overline{\text{RESET}}$ pin, a system reset is performed and the various on-chip hardware devices are reset to their initial settings. In addition, oscillation of the main clock is stopped during the reset period, although oscillation of the subsystem clock continues.

When the input at the $\overline{\text{RESET}}$ pin changes from low level to high level, the reset status is canceled and the CPU resumes program execution. The contents of the various registers should be initialized within the program as necessary.

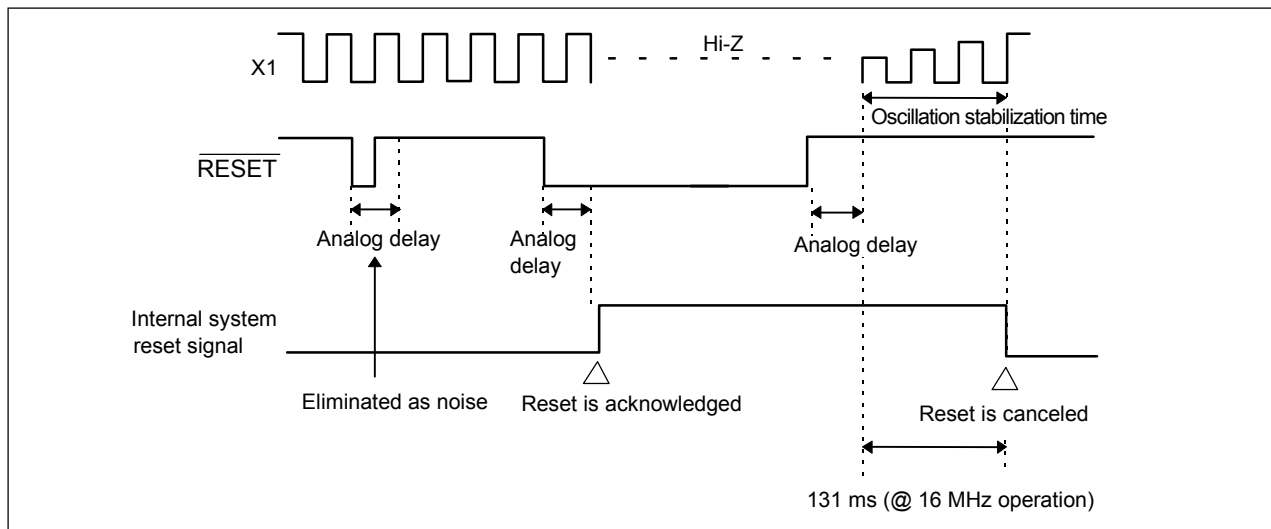
An on-chip noise eliminator uses analog delay to prevent noise-related malfunction at the $\overline{\text{RESET}}$ pin.

14.2 Pin Operations

During the system reset period, almost all pins are set to high impedance (except for $\overline{\text{RESET}}$, X2, CPUREG, V_{DD0} , $\text{ADC}V_{DD}$, ADCGND , $\text{PORT}V_{DD}$, PORTGND , GND0 , GND1 , GND2 , and $V_{PP/IC}$).

Accordingly, if connected to an external memory device, be sure to attach a pull-up (or pull-down) resistor at each pin. If such a resistor is not attached, high impedance will be set for these pins, which could damage the data in memory devices. Likewise, make sure the pins are handled so as to prevent such effects at the signal outputs of on-chip peripheral I/O functions and output ports.

Figure 14-1. System Reset Timing



14.3 Power-on-Clear Operation

The V850/SF1 includes a power-on-clear circuit (POC), through which low-voltage detection and V_{DD0} pin voltage detection (4.2 ± 0.3 V) can be performed using the POC status register (POCS).

(1) POC status register (POCS)

When a power-on-clear is generated, bit 0 of the POCS register is set to 1.

In addition, if the voltage level at the V_{DD0} pin is less than 4.2 ± 0.3 V, bit 1 of the POCS register is set to 1, thus enabling detection of a voltage level of less than 4.2 ± 0.3 V at the V_{DD0} pin.

In the case of a reset generated by the $\overline{\text{RESET}}$ pin, however, the POCM and VM45 bits retain their previous statuses. A low voltage state can be detected by reading the POCS register following reset cancellation.

The POCS register is read-only, using an 8-bit memory manipulation instruction. This register is reset when read.

After reset: Retained ^{Note R}		Address: FFFFF07AH						
	7	6	5	4	3	2	1	0
POCS	0	0	0	0	0	0	VM45	POCM
POCM	Detection of power-on-clear generation status							
0	Power-on-clear not generated							
1	Power-on-clear reset generated							
VM45	Detection of V_{DD0} pin voltage level							
0	V_{DD0} pin voltage of less than 4.5 V not detected							
1	V_{DD0} pin voltage of less than 4.5 V detected							
Note This value is 03H only after a power-on-clear reset; it is not initialized by a reset from the $\overline{\text{RESET}}$ pin.								

(2) VM45 control register (VM45C)

The detection status (detected/undetected) according to the POCS register's VM45 bit can be output (monitored) at the VM45/P34 pin via control by the VM45C register.

After reset: 00H	R/W	Address: FFFF07CH							
		7	6	5	4	3	2	1	0
POCS		0	0	0	0	0	0	VM45C1	VM45C0
VM45C1	VM45 (V _{DD0} 4.5 V monitor) output enabled/disabled								
0	VM45 output at VM45/P34 pin disabled (port function)								
1	VM45 output at VM45/P34 pin enabled ^{Note}								
VM45C0	VM45 (V _{DD0} 4.5 V monitor) output selection								
0	High-level output when VM45 detected								
1	Low-level output when VM45 detected								

Note When using P34 as an alternate function pin, it is necessary to set the PM34 bit of the port 3 mode register (PM3) to 0 (output mode), or the P34 bit of port 3 (P3) to 0 (0 output).

CHAPTER 15 REGULATOR

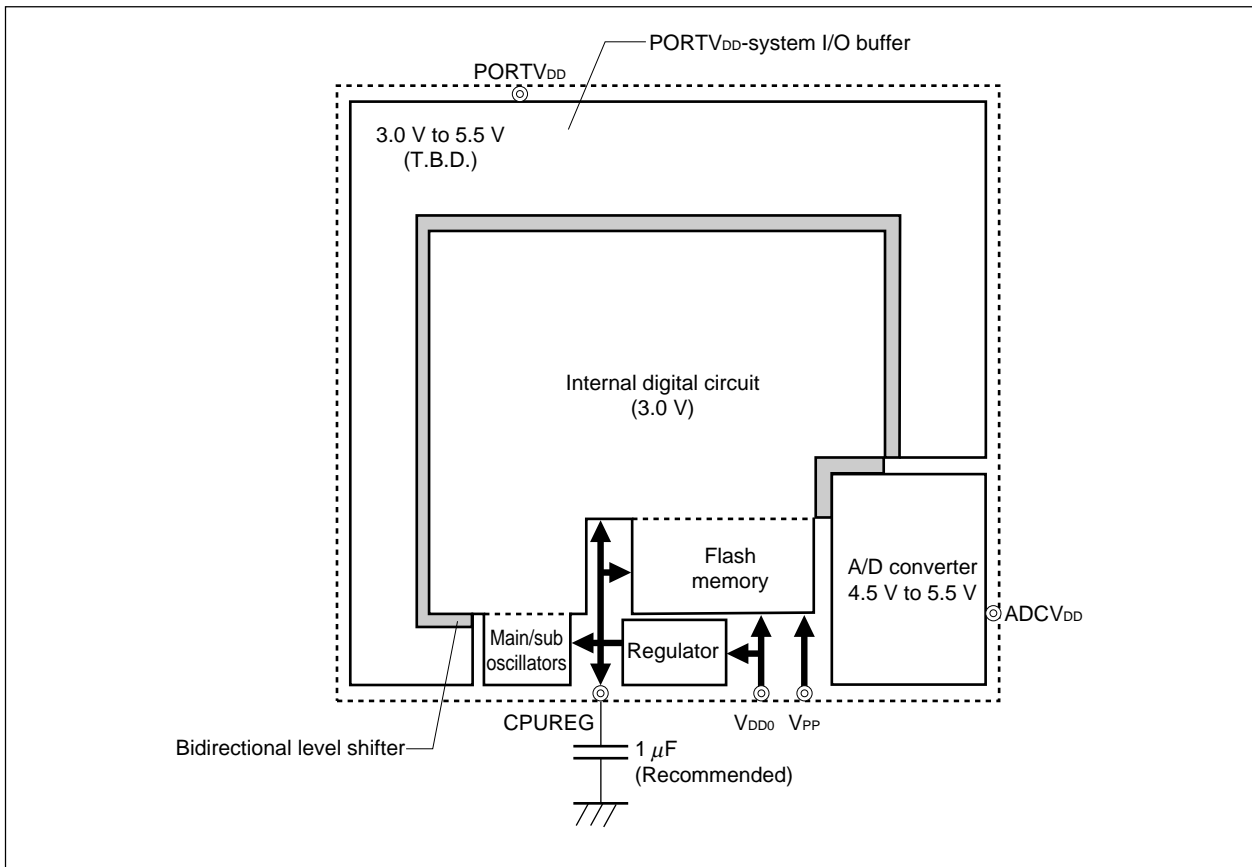
15.1 Outline

The V850/SF1 incorporates a regulator to realize a 5 V single power supply, low power consumption, and to reduce noise.

This regulator supplies a voltage obtained by stepping down the V_{DD} power supply voltage to oscillation blocks and on-chip logic circuits (excluding the A/D converter and output buffers). The regulator output voltage is set to 3.0 V.

Refer to **2.4 Pin I/O Circuit Types, I/O Buffer Power Supply and Connection of Unused Pins** for the power supply corresponding to each pin.

★ **Figure 15-1. Regulator**



15.2 Operation

The regulator of the V850/SF1 operates in every mode (STOP, IDLE, HALT).

For stabilization of regulator outputs, connect an electrolytic capacitor of about 1 μ F to the CPUREG pin.

CHAPTER 16 ROM CORRECTION FUNCTION

Remark n = 0 to 3 in CHAPTER 16.

16.1 General

The ROM correction function provided in the V850/SF1 is a function that replaces part of a program in the mask ROM with a program in the internal RAM.

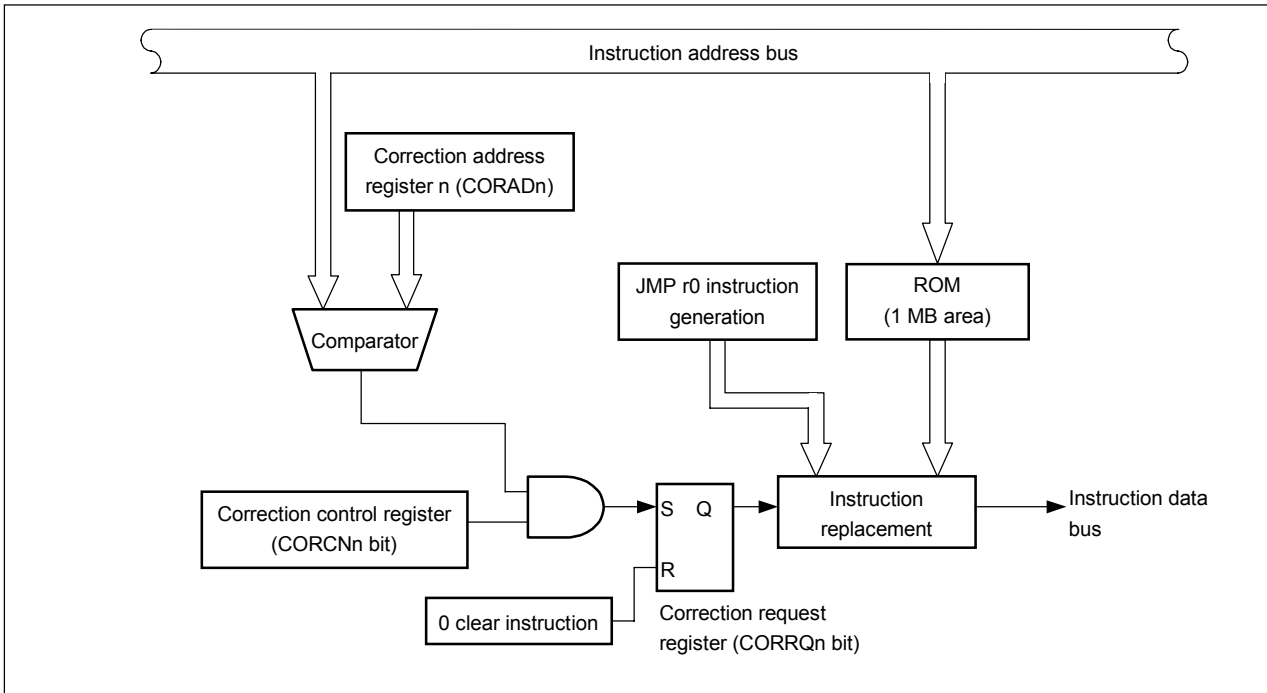
First, the instruction of the address where the program replacement should start is replaced with the JMP r0 instruction and the program is instructed to jump to 00000000H. The correction request register (CORRQ) is then checked. At this time, if the CORRQn flag is set (1), program control shifts to the internal RAM after being made to jump to the internal RAM area by an instruction such as a jump instruction.

Instruction bugs found in the mask ROM can be avoided, and program flow can be changed by using the ROM correction function.

Up to four correction addresses can be specified.

- Cautions**
1. The ROM correction function cannot be used for the data in the internal ROM; it can only be used for instruction codes. If the ROM correction is carried out on data, that data will replace the instruction code of the JMP r0 instruction.
 2. ROM correction for instructions that access the CORCN, CORRQ, or CORAD0 to CORAD3 registers is prohibited.

Figure 16-1. Block Diagram of ROM Correction



16.2 ROM Correction Peripheral I/O Registers

★ 16.2.1 Correction control register (CORCN)

CORCN controls whether or not the instruction of the correction address is replaced with the JMP r0 instruction when the correction address matches the fetch address.

Whether match detection by a comparator is enabled or disabled can be set for each channel.

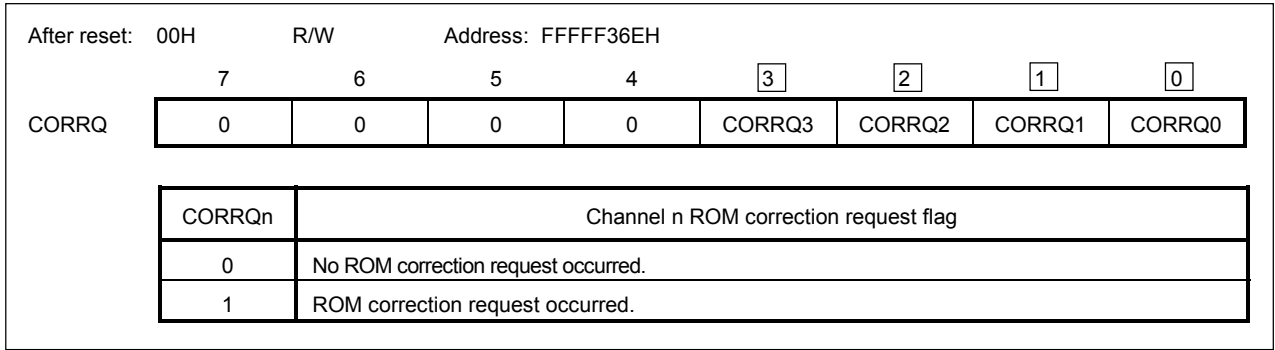
CORCN can be set by an 8- or 1-bit memory manipulation instruction.

After reset:	00H	R/W	Address: FFFF36CH								
	7	6	5	4	3	2	1	0			
CORCN	0	0	0	0	COREN3	COREN2	COREN1	COREN0			
	CORENn	CORADn register and fetch address match detection control									
	0	Match detection disabled									
	1	Match detection enabled									

16.2.2 Correction request register (CORRQ)

CORRQ saves the channel in which ROM correction occurred. The JMP r0 instruction makes the program jump to 00000000H after the correction address matches the fetch address. At this time, the program can judge the following cases by reading CORRQ.

- Reset input: CORRQ = 00H
- ROM correction generation: CORRQn bit = 1
- Branch to 00000000H by user program: CORRQ = 00H



16.2.3 Correction address registers 0 to 3 (CORAD0 to CORAD3)

CORADn sets the start address of an instruction to be corrected (correction address) in the ROM.

Up to four points of the program can be corrected at once since the V850/SF1 has four correction address registers (CORADn).

Set 00000000H to 0003FFFEH since the V850/SF1 incorporates a 256 KB ROM.

Bits 0 and 18 to 31 should be fixed to 0.

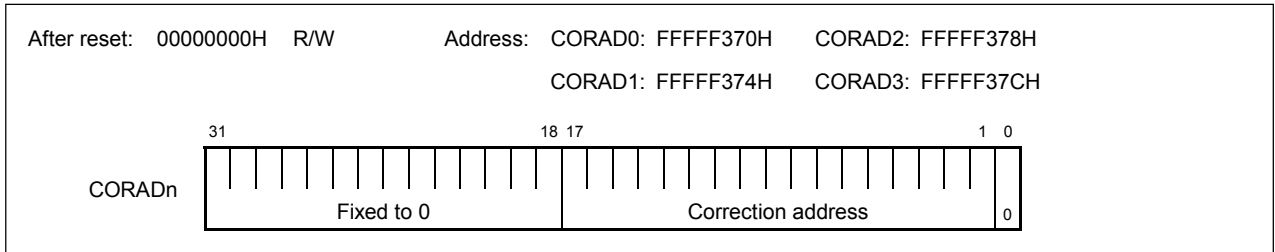
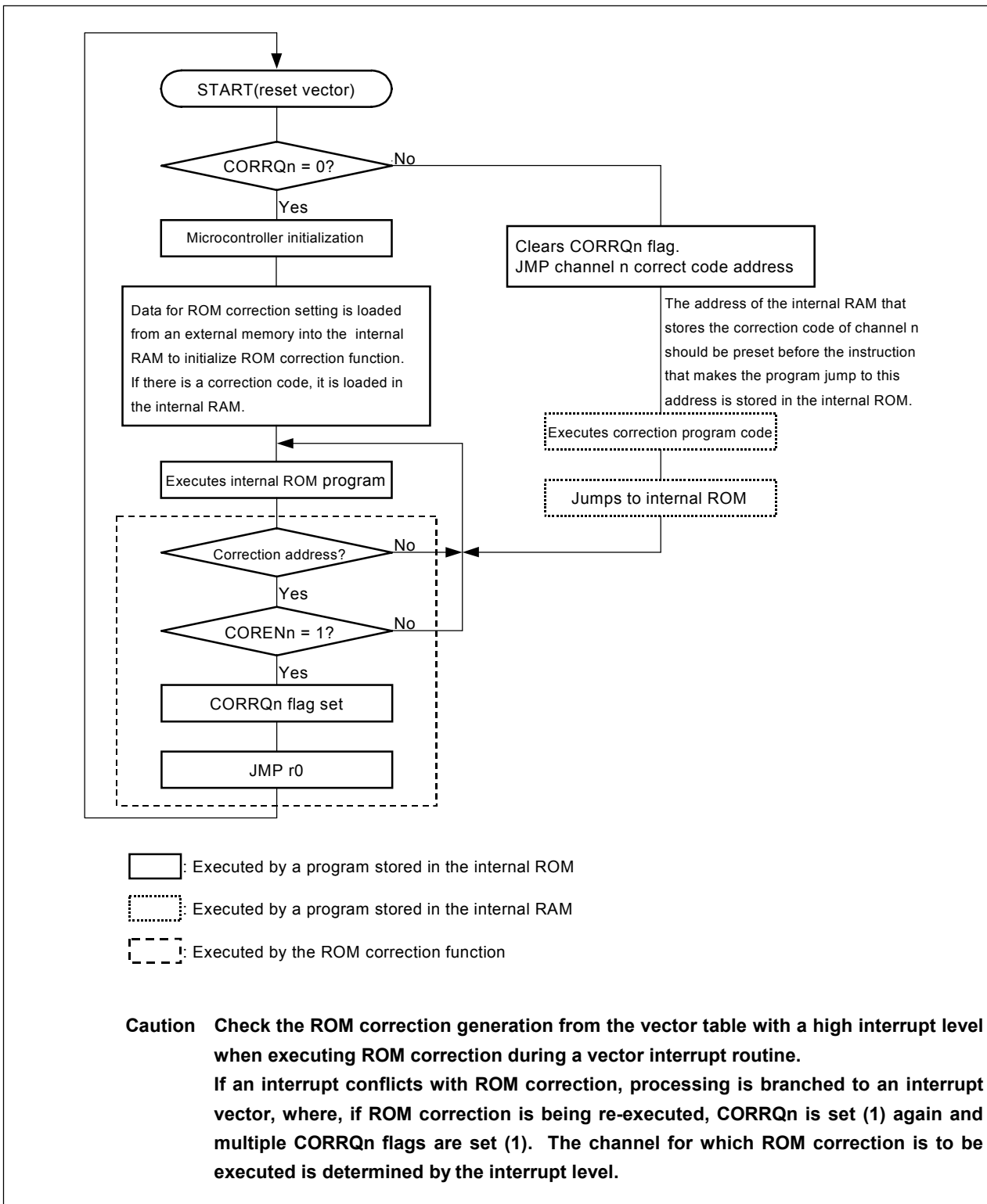


Figure 16-2. ROM Correction Operation and Program Flow



CHAPTER 17 FLASH MEMORY (μ PD70F3079Y)

The μ PD70F3079Y is the flash memory version of the V850/SF1 and incorporates a 256 KB flash memory. In the instruction fetch to this flash memory, 4 bytes can be accessed by a single clock in the same way as in the mask ROM version.

- ★ **Caution** There are differences in noise immunity and noise radiation between flash memory versions and mask ROM versions. When pre-producing an application set with the flash memory version and then mass-producing it with the mask ROM version, be sure to conduct sufficient evaluations for the commercial samples (CS) (not engineering samples (ES)) of the mask ROM versions.

Writing to flash memory can be performed with memory mounted on the target system (on board). A dedicated flash programmer is connected to the target system to perform writing.

The following can be considered the development environment and applications in which flash memory is used.

- Software can be altered after the V850/SF1 is solder-mounted on the target system.
- Small scale production of various models is made easier by differentiating software.
- Data adjustment in starting mass production is made easier.

17.1 Features

- 4-byte/1-clock access (in instruction fetch access)
- All area batch erase/area unit erase
- Communication via serial interface with the dedicated flash programmer
- Erase/write voltage: $V_{PP} = 7.8 \text{ V}$
- On-board programming
- Flash memory programming via self-rewrite in area (128 KB) units is possible

17.1.1 Erasing unit

This product has following two erasure units.

(a) All area batch erase

The area of xx000000H to xx03FFFFH can be erased at the same time. The erasure time is 4.0 s.

(b) Area erase

Erasure can be performed in area units (there are two 128 KB unit areas). The erasure time is 2.0 s for each area.

Area 0: The area of xx000000H to xx01FFFFH (128 KB) is erased

Area 1: The area of xx020000H to xx03FFFFH (128 KB) is erased

17.1.2 Write/read time

The write/read time is shown below.

Write time: 20 μ s/byte

Read time: 62.5 ns (cycle time)

17.2 Writing with Flash Programmer

Writing can be performed either on-board or off-board with the dedicated flash programmer.

(1) On-board programming

The contents of the flash memory are rewritten after the V850/SF1 is mounted on the target system. Mount connectors, etc., on the target system to connect the dedicated flash programmer.

(2) Off-board programming

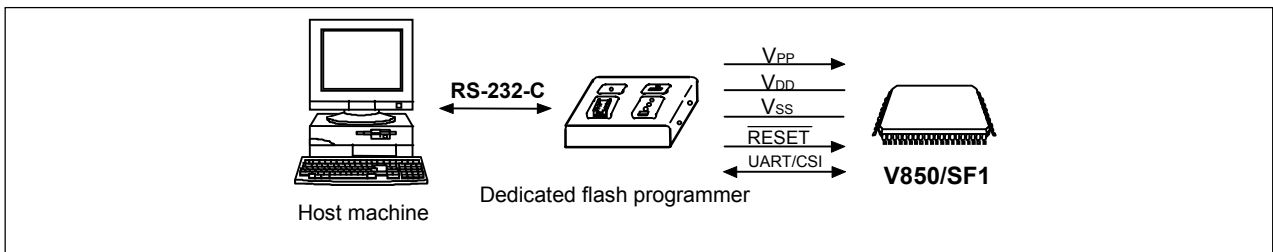
Writing to flash memory is performed by the dedicated program adapter (FA Series), etc., before mounting the V850/SF1 on the target system.

Remark The FA Series is a product of Naito Densai Machida Mfg. Co., Ltd.

17.3 Programming Environment

The following shows the environment required for writing programs to the flash memory of the V850/SF1.

Figure 17-1. Environment Required for Writing Programs to Flash Memory



A host machine is required for controlling the dedicated flash programmer.

UART0 or CSI0 is used as the interface between the dedicated flash programmer and the V850/SF1 to perform writing, erasing, etc. A dedicated program adapter (FA Series) required for off-board writing.

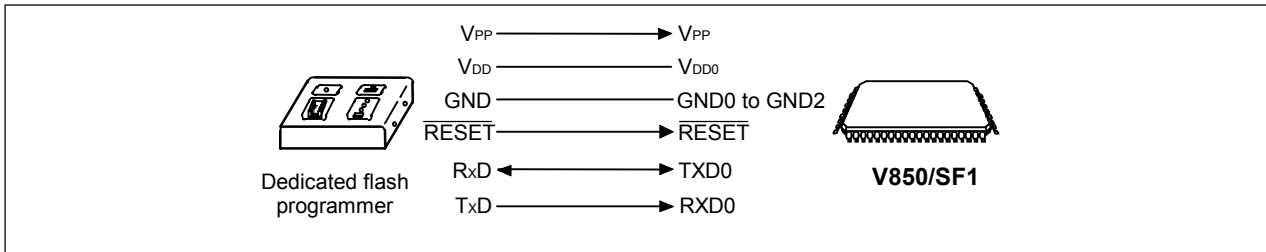
17.4 Communication Mode

Communication between the dedicated flash programmer and the V850/SF1 is performed by serial communication using UART0 or CSI0 of the V850/SF1.

(1) UART0

Transfer rate: 4800 to 76800 bps

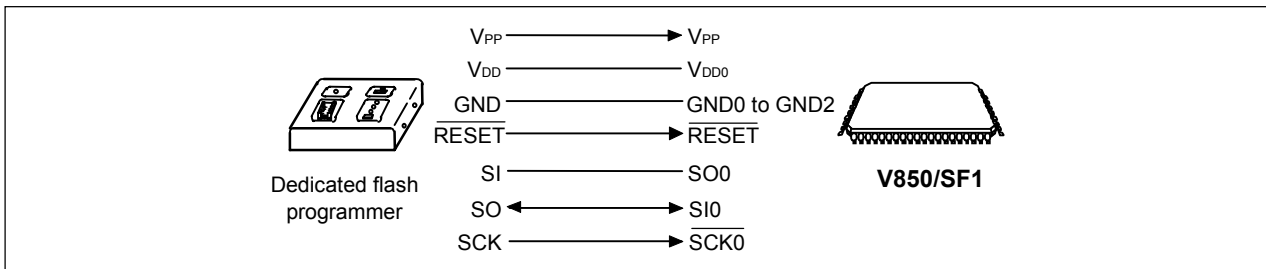
Figure 17-2. Communication with Dedicated Flash Programmer (UART0)



(2) CSI0

Serial clock: Up to 1 MHz (MSB first)

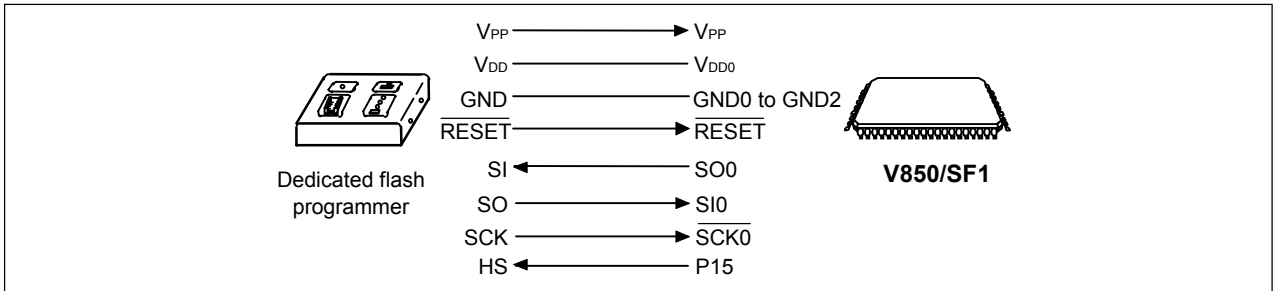
Figure 17-3. Communication with Dedicated Flash Programmer (CSI0)



(3) CSIO + HS

Serial clock: Up to 1 MHz (MSB first)

Figure 17-4. Communication with Dedicated Flash Programmer (CSI0 + HS)



The dedicated flash programmer outputs the transfer clock, and the V850/SF1 operates as a slave.

When the PG-FP3 is used as the dedicated flash programmer, it generates the following signals to the V850/SF1. For the details, refer to the PG-FP3 manual.

Table 17-1. Signal Generation of Dedicated Flash Programmer (PG-FP3)

PG-FP3			V850/SF1	Measures When Connected		
Signal Name	I/O	Pin Function	Pin Name	CSI0	UART0	CSI0 + HS
V _{PP}	Output	Writing voltage	V _{PP}	⊙	⊙	⊙
V _{DD}	I/O	V _{DD} voltage generation/ voltage monitoring	V _{DD0}	⊙	⊙	⊙
GND	–	Ground	GND0 to GND2	⊙	⊙	⊙
CLK ^{Note}	Output	Clock output to V850/SF1	X1	×	×	×
RESET	Output	Reset signal	RESET	⊙	⊙	⊙
SI/RxD	Input	Receive signal	SO0/TXD0	⊙	⊙	⊙
SO/TxD	Output	Transmit signal	SI0/RXD0	⊙	⊙	⊙
SCK	Output	Transfer clock	SCK0	⊙	×	⊙
HS	Input	Handshake signal of CSI0 + HS	P15	×	×	⊙

Note Supply clock on the target board.

Remark ⊙: Always connected

○: Does not need to be connected, if generated on the target board

×: Does not need to be connected

17.5 Pin Connection

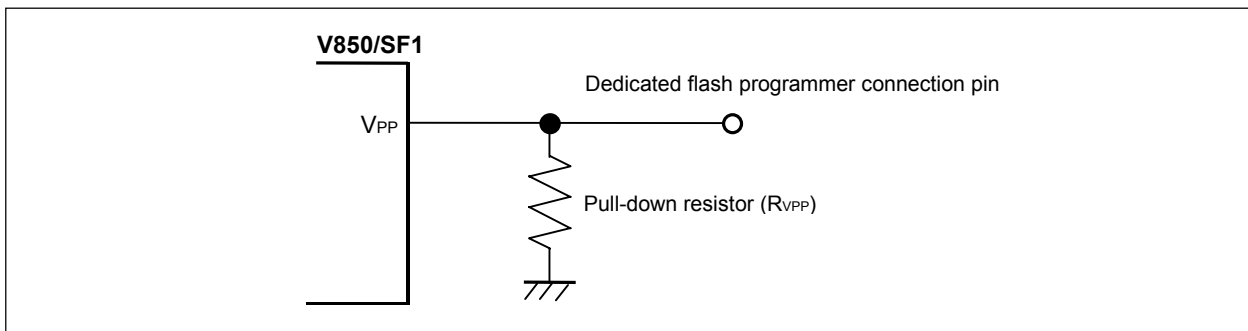
When performing on-board writing, install a connector on the target system to connect to the dedicated flash programmer. Also, install a function on-board to switch from the normal operation mode to the flash memory programming mode.

When switched to the flash memory programming mode, all the pins not used for the flash memory programming become the same status as that immediately after reset. Therefore, all the ports become output high-impedance, making pin handling necessary if the external device does not acknowledge the output high-impedance status.

17.5.1 V_{PP} pin

In the normal operation mode, 0 V is input to the V_{PP} pin. In the flash memory programming mode, a 7.8 V writing voltage is supplied to the V_{PP} pin. The following shows an example of the connection of the V_{PP} pin.

Figure 17-5. V_{PP} Pin Connection Example



17.5.2 Serial interface pin

The following shows the pins used by each serial interface.

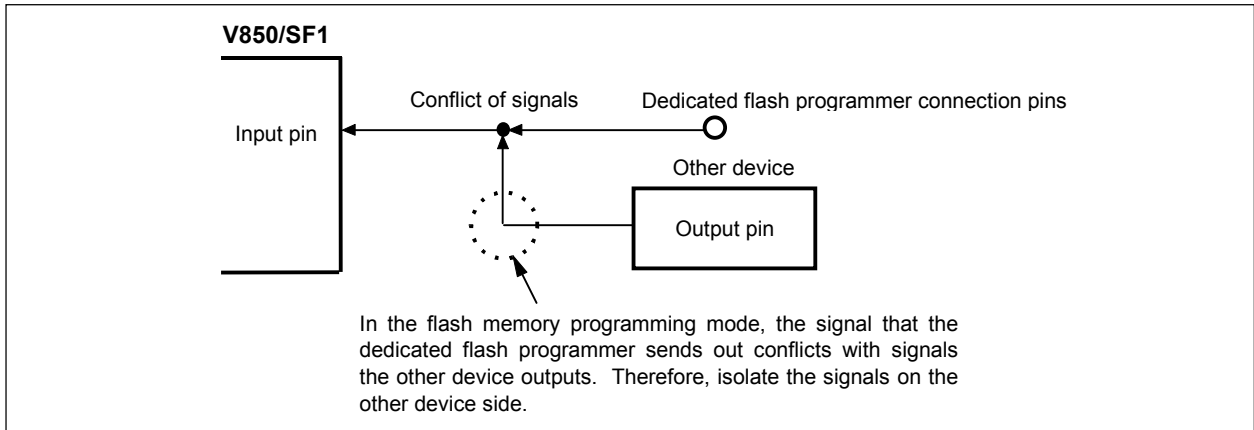
Table 17-2. Pins Used by Serial Interfaces

Serial Interface	Pins Used
CSI0	SO0, SI0, $\overline{\text{SCK0}}$
CSI0 + HS	SO0, SI0, $\overline{\text{SCK0}}$, P15
UART0	TXD0, RXD0

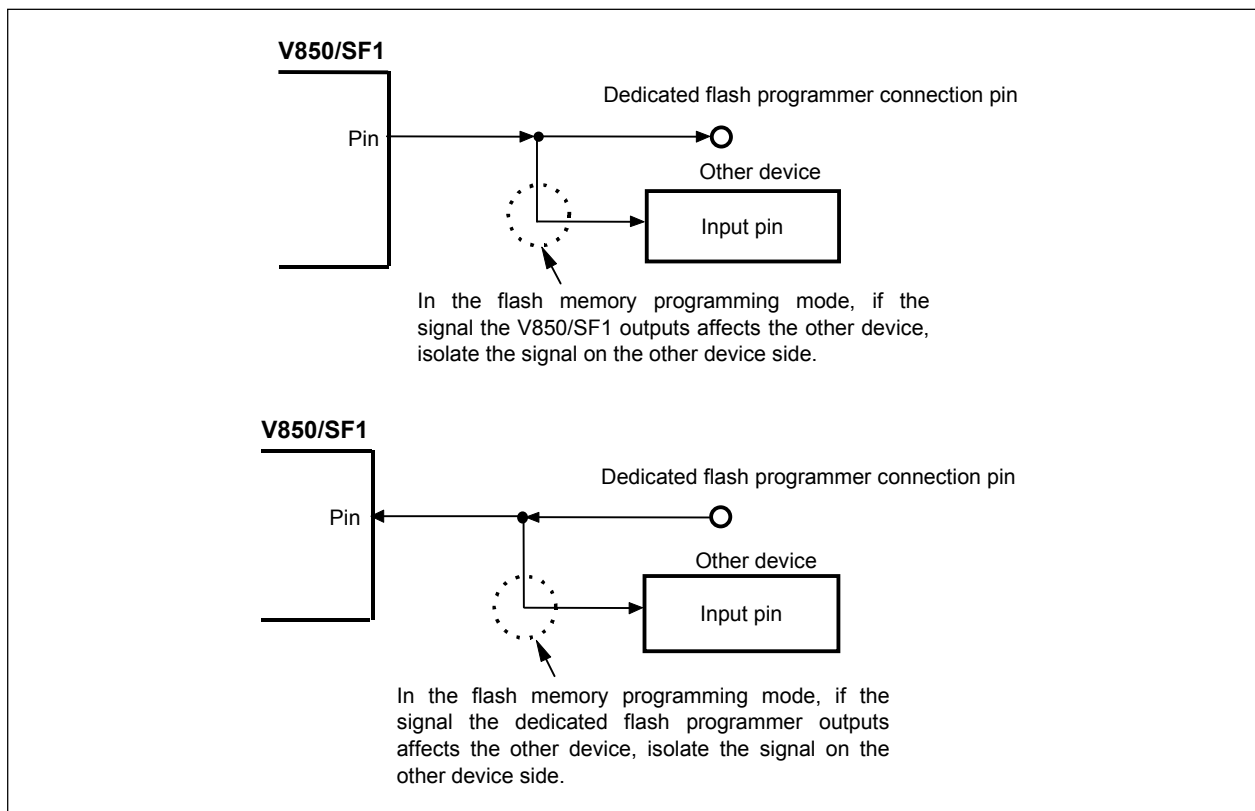
When connecting a dedicated flash programmer to a serial interface pin that is connected to other devices on-board, care should be taken to the conflict of signals and the malfunction of other devices, etc.

(1) Conflict of signals

When connecting the dedicated flash programmer (output) to a serial interface pin (input) that is connected to another device (output), a conflict of signals occurs. To avoid the conflict of signals, isolate the connection to the other device or set the other device to output high-impedance.

Figure 17-6. Conflict of Signals (Serial Interface Input Pin)**(2) Malfunction of other device**

When connecting dedicated flash programmer (output or input) to a serial interface pin (input or output) that is connected to another device (input), the signal output to the other device may cause the device to malfunction. To avoid this, isolate the connection to the other device or make the setting so that the input signal to the other device is ignored.

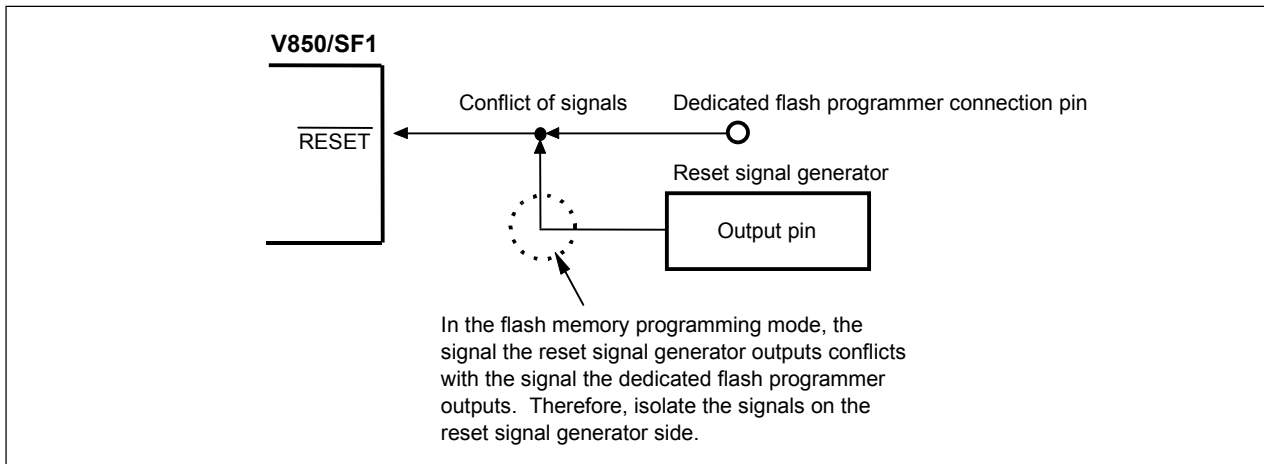
Figure 17-7. Malfunction of Other Device

17.5.3 $\overline{\text{RESET}}$ pin

When connecting the reset signals of the dedicated flash programmer to the $\overline{\text{RESET}}$ pin that is connected to the reset signal generator on-board, a conflict of signals occurs. To avoid the conflict of signals, isolate the connection to the reset signal generator.

When a reset signal is input from the user system in the flash memory programming mode, the programming operation will not be performed correctly. Therefore, do not input signals other than the reset signals from the dedicated flash programmer.

Figure 17-8. Conflict of Signals ($\overline{\text{RESET}}$ Pin)



17.5.4 Port pin (including NMI)

When the flash memory programming mode is set, all the port pins except the pins that communicate with the dedicated flash programmer become output high-impedance. If problems such as disabling the output high-impedance status should occur in the external devices connected to the port, connect them to V_{DD0} or GND0 to GND2 via resistors.

17.5.5 Other signal pins

Connect X1, X2, XT1, and XT2 in the same status as that in the normal operation mode.

17.5.6 Power supply

Supply the power as follows:

$$V_{DD0} = \text{PORTV}_{DD}$$

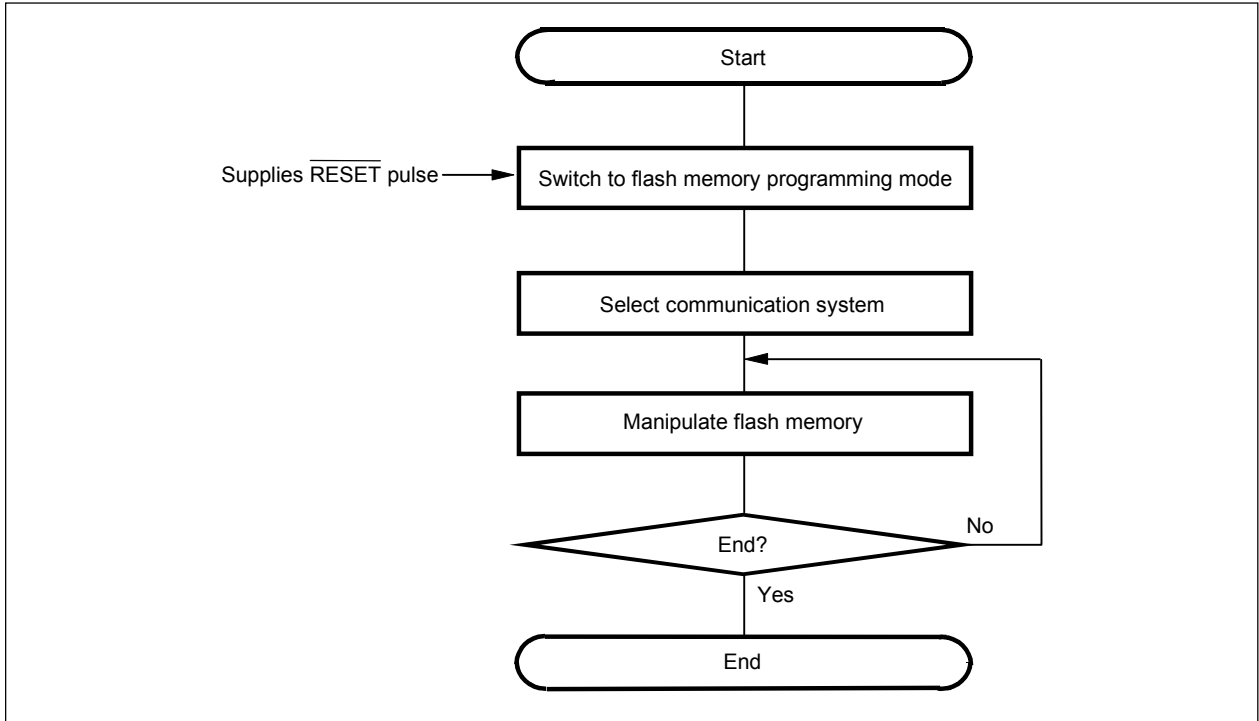
Supply the power (ADCV_{DD} , ADCGND , GND0 to GND2 , and PORTGND) in the same way as in normal operation mode.

17.6 Programming Method

17.6.1 Flash memory control

The following shows the procedure for manipulating the flash memory.

Figure 17-9. Procedure for Manipulating Flash Memory

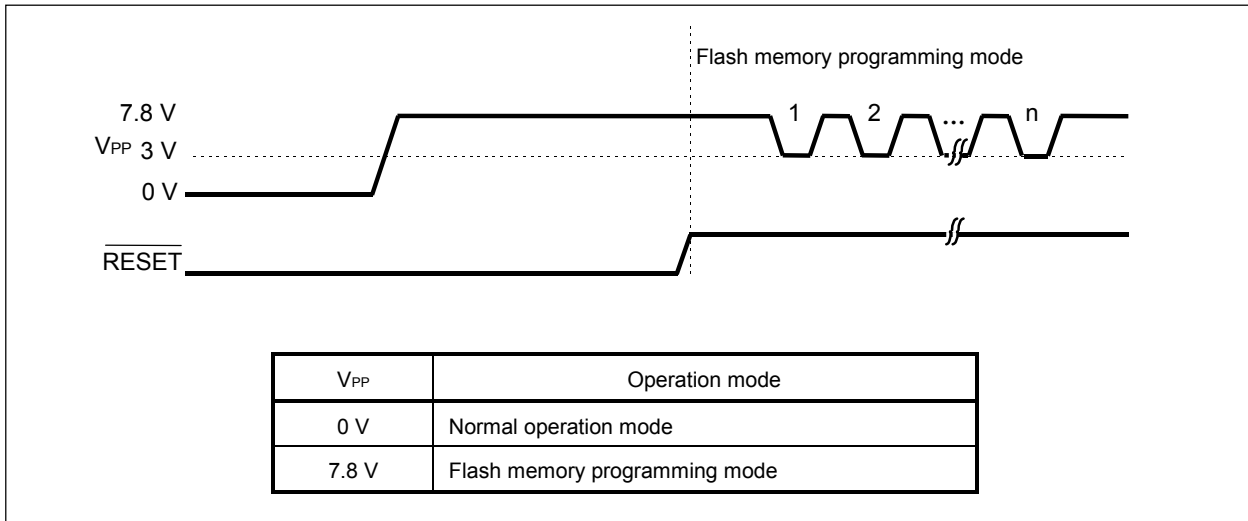


17.6.2 Flash memory programming mode

When rewriting the contents of flash memory using the dedicated flash programmer, set the V850/SF1 in the flash memory programming mode. When switching modes, set the V_{PP} pin before canceling reset.

When performing on-board writing, switch modes using a jumper, etc.

Figure 17-10. Flash Memory Programming Mode



17.6.3 Selection of communication mode

In the V850/SF1, a communication mode is selected by inputting pulses (16 pulses max.) to the V_{PP} pin after switching to the flash memory programming mode. The V_{PP} pulse is generated by the dedicated flash programmer.

The following shows the relationship between the number of pulses and the communication mode.

Table 17-3. List of Communication Mode

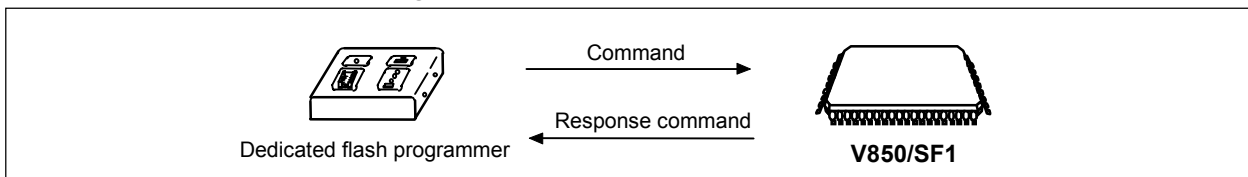
V _{PP} Pulse	Communication Mode	Remarks
0	CSI0	V850/SF1 performs slave operation, MSB first
3	CSI0 + HS	V850/SF1 performs slave operation, MSB first
8	UART0	Communication rate: 9600 bps (after reset), LSB first
Others	RFU	Setting prohibited

Caution When UART is selected, the receive clock is calculated based on the reset command sent from the dedicated flash programmer after receiving the V_{PP} pulse.

17.6.4 Communication command

The V850/SF1 communicates with the dedicated flash programmer by means of commands. The command sent from the dedicated flash programmer to the V850/SF1 is called a “command”. The response signal sent from the V850/SF1 to the dedicated flash programmer is called a “response command”.

Figure 17-11. Communication Command



The following shows the commands for flash memory control of the V850/SF1. All of these commands are issued from the dedicated flash programmer, and the V850/SF1 performs the various processing corresponding to the commands.

Table 17-4. Flash Memory Control Commands

Category	Command Name	Function
Verify	Batch verify command	Compares the contents of the entire memory and the input data.
Erase	Batch erase command	Erases the contents of the entire memory.
	Write back command	Writes back the contents which is overerased.
Blank check	Batch blank check command	Checks the erase state of the entire memory.
Data write	High-speed write command	Writes data by the specification of the write address and the number of bytes to be written, and executes verify check.
	Continuous write command	Writes data from the address following the high-speed write command executed immediately before, and executes verify check.
System setting and control	Status read out command	Acquires the status of operations.
	Oscillating frequency setting command	Sets the oscillating frequency.
	Erasure time setting command	Sets the erasure time of batch erase.
	Writing time setting command	Sets the writing time of data write.
	Write back time setting command	Sets the write back time.
	Baud rate setting command	Sets the baud rate when using UART.
	Silicon signature command	Reads out the silicon signature information.
	Reset command	Escapes from each state.

The V850/SF1 sends back response commands to the commands issued from the dedicated flash programmer. The following shows the response commands the V850/SF1 sends out.

Table 17-5. Response Commands

Response Command Name	Function
ACK (acknowledge)	Acknowledges command/data, etc.
NAK (not acknowledge)	Acknowledges illegal command/data, etc.

17.6.5 Resources used

The resources used in the flash memory programming mode are all the FFE000H to FFE7FFH area of the internal RAM and all the registers. The FFE800H to FFEFFFH area of the internal RAM retains data as long as the power is on. The registers that are initialized by reset are changed to default values.

CHAPTER 18 FCAN CONTROLLER

The V850/SF1 features an on-chip FCAN (Full Controller Area Network) controller that complies with CAN specification Ver. 2.0, Part B. (The V850/SF1 product line includes the μ PD703079Y and μ PD70F3079Y as two-channel devices and the μ PD703078Y as a single-channel device.)

18.1 Overview of Functions

Table 18-1 presents an overview of V850/SF1 functions.

Table 18-1. Overview of Functions

Function	Description
Protocol	CAN Protocol Ver. 2.0, Part B (standard and extended frame transmission/reception)
Baud rate	Maximum 1 Mbps (during 16 MHz clock input)
Data storage	<ul style="list-style-type: none"> • Allocated to common access-enabled RAM area • RAM that is mapped to an unused message byte can be used for CPU processing or other processing
Mask functions	<ul style="list-style-type: none"> • Four • Global masks and local masks can be used without distinction
Message configuration	Can be declared as transmit message or receive message
No. of messages	32 messages
Message storage method	<ul style="list-style-type: none"> • Storage to receive buffer corresponding to each ID • Storage to buffer specified by receive mask function
Remote reception	<ul style="list-style-type: none"> • Remote frames can be received in either the receive message buffer or the transmit message buffer • If a remote frame is received by a transmit message buffer, there is a choice between having the remote request processed by the CPU or starting the auto transmit function.
Remote transmission	The remote frame can be sent either by setting the transmit message's RTR bit (M_CTRLn register) or by setting the receive message's send request.
Time stamp function	A time stamp function can be set for receive messages and transmit messages.
Diagnostic functions	<ul style="list-style-type: none"> • Read-enabled error counter provided. • "Valid protocol operation flag" provided for verification of bus connections. • Receive-only mode (with auto baud rate detection) provided. • Diagnostic processing mode provided.
Low-power mode	<ul style="list-style-type: none"> • CAN sleep mode (wake up function using CAN bus enabled) • CAN stop mode (wake up function using CAN bus disabled)

Remark n = 00 to 31

18.2 Configuration

FCAN is composed of the following four blocks.

(1) NPB interface

This functional block provides an NPB (NEC peripheral I/O bus) interface as a means of transmitting and receiving signals.

(2) MAC (Memory Access Controller)

This functional block controls access to the CAN module within the FCAN and to the CAN RAM.

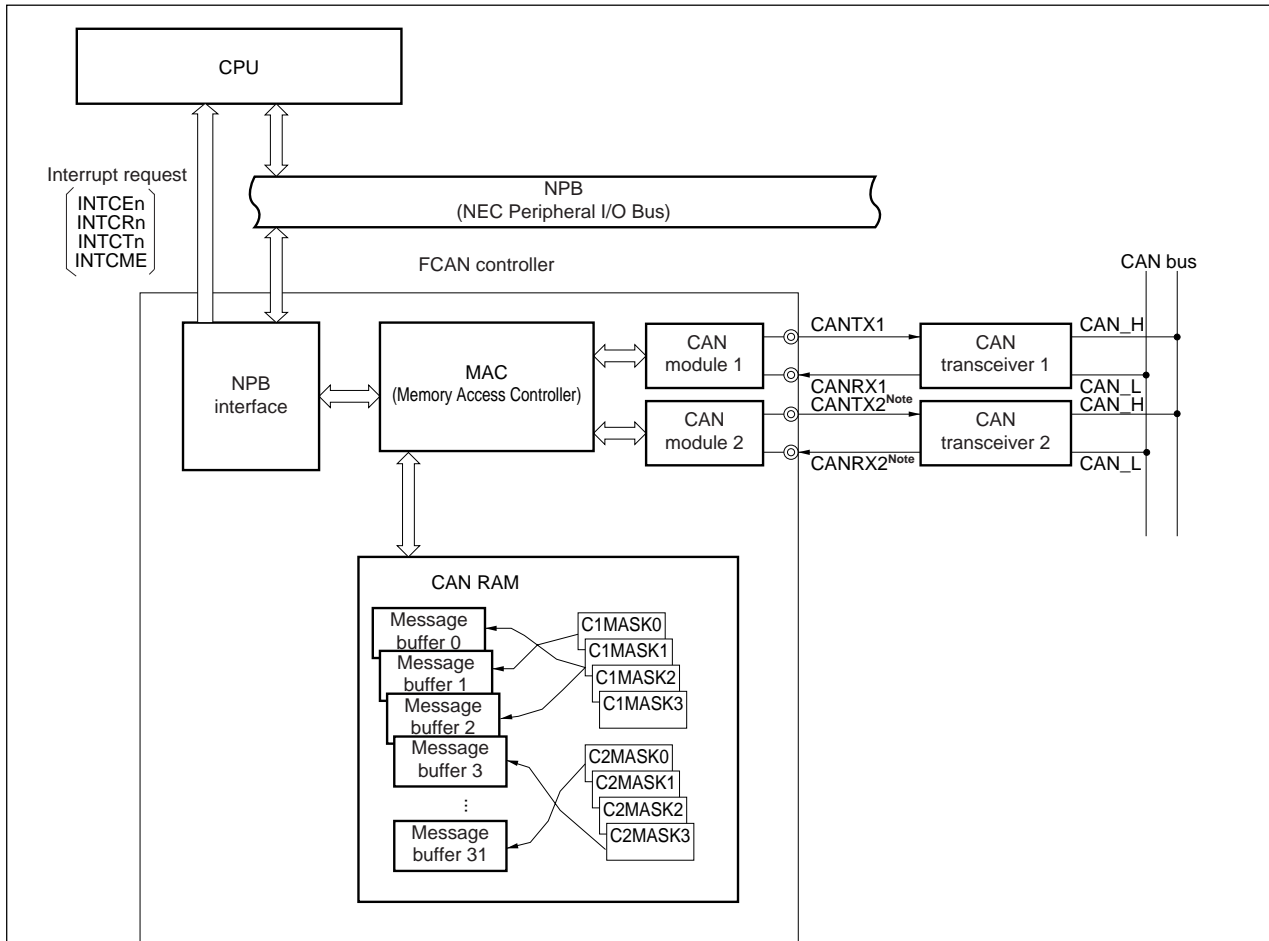
(3) CAN module

This functional block is involved in the operation of the CAN protocol layer and its related settings.

(4) CAN RAM

This is the CAN memory functional block, which is used to store message IDs, message data, etc.

Figure 18-1. Block Diagram of FCAN



Note μ PD703079Y and 70F3079Y only

- Cautions**
1. When P114/CANTX1, P115/CANRX1, P116/CANTX2, P117/CANRX2 are used during FCAN transmission/reception, they can be used as FCAN pin functions (CANTX1, CANRX1, CANTX2, CANRX2) by setting the port alternate-function control register (PAC) (refer to 5.2.10 (2) (b) Port alternate-function control register (PAC)).
 2. When the P114/CANTX1 and P116/CANTX2 pins are used as CANTX1, CANTX2, set both the P11 and PM11 registers to 0 (refer to 5.3 Setting When Port Pin Is Used for Alternate Function).
 3. When the P115/CANRX1 and P117/CANRX2 pins are used as CANRX1 and CANRX2, set the P11 register to 0 and the PM11 register to 1.
 4. If the FCAN register is read/written when the external bus interface function is used, an address/data control signal is output to the external expansion pins (ports 4, 5, 6, 9), so read/write of $0xFF800H$ to $0xFFFFFH$ ($m = 3, 7, B$), which is the FCAN address area, should not be performed for the external devices connected to the external expansion pins.
 5. If the wait function and idle function are set when the external bus interface function is used, these functions are enabled even reading/writing the FCAN register.
 6. Since no clock is supplied from the subsystem clock to FCAN, when stopping the main clock and setting the subsystem clock operation, do not read/write the FCAN register.

Remark $n = 1, 2$

18.3 Internal Registers of FCAN Controller

18.3.1 Configuration of message buffers

Table 18-2. Configuration of Message Buffers

Address	Register Name
xxnFF800H to xxnFF81FH	Message buffer 0 field
xxnFF820H to xxnFF83FH	Message buffer 1 field
xxnFF840H to xxnFF85FH	Message buffer 2 field
xxnFF860H to xxnFF87FH	Message buffer 3 field
xxnFF880H to xxnFF89FH	Message buffer 4 field
xxnFF8A0H to xxnFF8BFH	Message buffer 5 field
xxnFF8C0H to xxnFF8DFH	Message buffer 6 field
xxnFF8E0H to xxnFF8FFH	Message buffer 7 field
xxnFF900H to xxnFF91FH	Message buffer 8 field
xxnFF920H to xxnFF93FH	Message buffer 9 field
xxnFF940H to xxnFF95FH	Message buffer 10 field
xxnFF960H to xxnFF97FH	Message buffer 11 field
xxnFF980H to xxnFF99FH	Message buffer 12 field
xxnFF9A0H to xxnFF9BFH	Message buffer 13 field
xxnFF9C0H to xxnFF9DFH	Message buffer 14 field
xxnFF9E0H to xxnFF9FFH	Message buffer 15 field
xxnFFA00H to xxnFFA1FH	Message buffer 16 field
xxnFFA20H to xxnFFA3FH	Message buffer 17 field
xxnFFA40H to xxnFFA5FH	Message buffer 18 field
xxnFFA60H to xxnFFA7FH	Message buffer 19 field
xxnFFA80H to xxnFFA9FH	Message buffer 20 field
xxnFFAA0H to xxnFFABFH	Message buffer 21 field
xxnFFAC0H to xxnFFADFH	Message buffer 22 field
xxnFFAE0H to xxnFFAFFH	Message buffer 23 field
xxnFFB00H to xxnFFB1FH	Message buffer 24 field
xxnFFB20H to xxnFFB3FH	Message buffer 25 field
xxnFFB40H to xxnFFB5FH	Message buffer 26 field
xxnFFB60H to xxnFFB7FH	Message buffer 27 field
xxnFFB80H to xxnFFB9FH	Message buffer 28 field
xxnFFBA0H to xxnFFBBFH	Message buffer 29 field
xxnFFBC0H to xxnFFBDFH	Message buffer 30 field
xxnFFBE0H to xxnFFBFFH	Message buffer 31 field

- Remarks**
1. For details of message buffers, see **18.3.2 List of FCAN registers**.
 2. n = 3, 7, B

18.3.2 List of FCAN registers

(1/14)

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bit	16 Bit	32 Bit	
xxnFF804H	CAN message data length register 00	M_DLC00	R/W		√			Undefined
xxnFF805H	CAN message control register 00	M_CTRL00			√			
xxnFF806H	CAN message time stamp register 00	M_TIME00				√		
xxnFF808H	CAN message data register 000	M_DATA000			√			
xxnFF809H	CAN message data register 001	M_DATA001			√			
xxnFF80AH	CAN message data register 002	M_DATA002			√			
xxnFF80BH	CAN message data register 003	M_DATA003			√			
xxnFF80CH	CAN message data register 004	M_DATA004			√			
xxnFF80DH	CAN message data register 005	M_DATA005			√			
xxnFF80EH	CAN message data register 006	M_DATA006			√			
xxnFF80FH	CAN message data register 007	M_DATA007			√			
xxnFF810H	CAN message ID register L00	M_IDL00				√		
xxnFF812H	CAN message ID register H00	M_IDH00				√		
xxnFF814H	CAN message configuration register 00	M_CONF00			√			
xxnFF815H	CAN message status register 00	M_STAT00	R		√			
xxnFF816H	CAN status set/clear register 00	SC_STAT00	W			√	0000H	
xxnFF824H	CAN message data length register 01	M_DLC01	R/W		√		Undefined	
xxnFF825H	CAN message control register 01	M_CTRL01			√			
xxnFF826H	CAN message time stamp register 01	M_TIME01				√		
xxnFF828H	CAN message data register 010	M_DATA010			√			
xxnFF829H	CAN message data register 011	M_DATA011			√			
xxnFF82AH	CAN message data register 012	M_DATA012			√			
xxnFF82BH	CAN message data register 013	M_DATA013			√			
xxnFF82CH	CAN message data register 014	M_DATA014			√			
xxnFF82DH	CAN message data register 015	M_DATA015			√			
xxnFF82EH	CAN message data register 016	M_DATA016			√			
xxnFF82FH	CAN message data register 017	M_DATA017			√			
xxnFF830H	CAN message ID register L01	M_IDL01				√		
xxnFF832H	CAN message ID register H01	M_IDH01				√		
xxnFF834H	CAN message configuration register 01	M_CONF01			√			
xxnFF835H	CAN message status register 01	M_STAT01	R		√			
xxnFF836H	CAN status set/clear register 01	SC_STAT01	W			√	0000H	
xxnFF844H	CAN message data length register 02	M_DLC02	R/W		√		Undefined	
xxnFF845H	CAN message control register 02	M_CTRL02			√			
xxnFF846H	CAN message time stamp register 02	M_TIME02				√		
xxnFF848H	CAN message data register 020	M_DATA020			√			
xxnFF849H	CAN message data register 021	M_DATA021			√			
xxnFF84AH	CAN message data register 022	M_DATA022			√			
xxnFF84BH	CAN message data register 023	M_DATA023			√			
xxnFF84CH	CAN message data register 024	M_DATA024			√			

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset		
				1 Bit	8 Bit	16 Bit	32 Bit			
xxnFF84DH	CAN message data register 025	M_DATA025	R/W		√			Undefined		
xxnFF84EH	CAN message data register 026	M_DATA026			√					
xxnFF84FH	CAN message data register 027	M_DATA027			√					
xxnFF850H	CAN message ID register L02	M_IDL02				√				
xxnFF852H	CAN message ID register H02	M_IDH02				√				
xxnFF854H	CAN message configuration register 02	M_CONF02			√					
xxnFF855H	CAN message status register 02	M_STAT02	R		√					
xxnFF856H	CAN status set/clear register 02	SC_STAT02	W			√	0000H			
xxnFF864H	CAN message data length register 03	M_DLC03	R/W		√			Undefined		
xxnFF865H	CAN message control register 03	M_CTRL03			√					
xxnFF866H	CAN message time stamp register 03	M_TIME03				√				
xxnFF868H	CAN message data register 030	M_DATA030			√					
xxnFF869H	CAN message data register 031	M_DATA031			√					
xxnFF86AH	CAN message data register 032	M_DATA032			√					
xxnFF86BH	CAN message data register 033	M_DATA033			√					
xxnFF86CH	CAN message data register 034	M_DATA034			√					
xxnFF86DH	CAN message data register 035	M_DATA035			√					
xxnFF86EH	CAN message data register 036	M_DATA036			√					
xxnFF86FH	CAN message data register 037	M_DATA037			√					
xxnFF870H	CAN message ID register L03	M_IDL03				√				
xxnFF872H	CAN message ID register H03	M_IDH03				√				
xxnFF874H	CAN message configuration register 03	M_CONF03			√					
xxnFF875H	CAN message status register 03	M_STAT03		R		√				
xxnFF876H	CAN status set/clear register 03	SC_STAT03		W			√		0000H	
xxnFF884H	CAN message data length register 04	M_DLC04		R/W		√				Undefined
xxnFF885H	CAN message control register 04	M_CTRL04				√				
xxnFF886H	CAN message time stamp register 04	M_TIME04				√				
xxnFF888H	CAN message data register 040	M_DATA040			√					
xxnFF889H	CAN message data register 041	M_DATA041			√					
xxnFF88AH	CAN message data register 042	M_DATA042			√					
xxnFF88BH	CAN message data register 043	M_DATA043			√					
xxnFF88CH	CAN message data register 044	M_DATA044			√					
xxnFF88DH	CAN message data register 045	M_DATA045			√					
xxnFF88EH	CAN message data register 046	M_DATA046			√					
xxnFF88FH	CAN message data register 047	M_DATA047			√					
xxnFF890H	CAN message ID register L04	M_IDL04				√				
xxnFF892H	CAN message ID register H04	M_IDH04				√				
xxnFF894H	CAN message configuration register 04	M_CONF04			√					
xxnFF895H	CAN message status register 04	M_STAT04	R			√				
xxnFF896H	CAN status set/clear register 04	SC_STAT04	W				√	0000H		
xxnFF8A4H	CAN message data length register 05	M_DLC05	R/W			√			Undefined	
xxnFF8A5H	CAN message control register 05	M_CTRL05				√				

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bit	16 Bit	32 Bit	
xxnFF8A6H	CAN message time stamp register 05	M_TIME05	R/W			√		Undefined
xxnFF8A8H	CAN message data register 050	M_DATA050			√			
xxnFF8A9H	CAN message data register 051	M_DATA051			√			
xxnFF8AAH	CAN message data register 052	M_DATA052			√			
xxnFF8ABH	CAN message data register 053	M_DATA053			√			
xxnFF8ACH	CAN message data register 054	M_DATA054			√			
xxnFF8ADH	CAN message data register 055	M_DATA055			√			
xxnFF8AEH	CAN message data register 056	M_DATA056			√			
xxnFF8AFH	CAN message data register 057	M_DATA057			√			
xxnFF8B0H	CAN message ID register L05	M_IDL05				√		
xxnFF8B2H	CAN message ID register H05	M_IDH05				√		
xxnFF8B4H	CAN message configuration register 05	M_CONF05			√			
xxnFF8B5H	CAN message status register 05	M_STAT05	R		√			
xxnFF8B6H	CAN status set/clear register 05	SC_STAT05	W			√	0000H	
xxnFF8C4H	CAN message data length register 06	M_DLC06	R/W		√			Undefined
xxnFF8C5H	CAN message control register 06	M_CTRL06			√			
xxnFF8C6H	CAN message time stamp register 06	M_TIME06				√		
xxnFF8C8H	CAN message data register 060	M_DATA060			√			
xxnFF8C9H	CAN message data register 061	M_DATA061			√			
xxnFF8CAH	CAN message data register 062	M_DATA062			√			
xxnFF8CBH	CAN message data register 063	M_DATA063			√			
xxnFF8CCH	CAN message data register 064	M_DATA064			√			
xxnFF8CDH	CAN message data register 065	M_DATA065			√			
xxnFF8CEH	CAN message data register 066	M_DATA066			√			
xxnFF8CFH	CAN message data register 067	M_DATA067			√			
xxnFF8D0H	CAN message ID register L06	M_IDL06				√		
xxnFF8D2H	CAN message ID register H06	M_IDH06			√			
xxnFF8D4H	CAN message configuration register 06	M_CONF06		√				
xxnFF8D5H	CAN message status register 06	M_STAT06	R		√			
xxnFF8D6H	CAN status set/clear register 06	SC_STAT06	W			√	0000H	
xxnFF8E4H	CAN message data length register 07	M_DLC07	R/W		√			Undefined
xxnFF8E5H	CAN message control register 07	M_CTRL07			√			
xxnFF8E6H	CAN message time stamp register 07	M_TIME07				√		
xxnFF8E8H	CAN message data register 070	M_DATA070			√			
xxnFF8E9H	CAN message data register 071	M_DATA071			√			
xxnFF8EAH	CAN message data register 072	M_DATA072			√			
xxnFF8EBH	CAN message data register 073	M_DATA073			√			
xxnFF8ECH	CAN message data register 074	M_DATA074			√			
xxnFF8EDH	CAN message data register 075	M_DATA075			√			
xxnFF8EEH	CAN message data register 076	M_DATA076			√			
xxnFF8EFH	CAN message data register 077	M_DATA077			√			
xxnFF8F0H	CAN message ID register L07	M_IDL07				√		
xxnFF8F2H	CAN message ID register H07	M_IDH07			√			

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bit	16 Bit	32 Bit	
xxnFF8F4H	CAN message configuration register 07	M_CONF07	R/W		√			Undefined
xxnFF8F5H	CAN message status register 07	M_STAT07	R		√			
xxnFF8F6H	CAN status set/clear register 07	SC_STAT07	W			√	0000H	
xxnFF904H	CAN message data length register 08	M_DLC08	R/W		√			Undefined
xxnFF905H	CAN message control register 08	M_CTRL08			√			
xxnFF906H	CAN message time stamp register 08	M_TIME08				√		
xxnFF908H	CAN message data register 080	M_DATA080			√			
xxnFF909H	CAN message data register 081	M_DATA081			√			
xxnFF90AH	CAN message data register 082	M_DATA082			√			
xxnFF90BH	CAN message data register 083	M_DATA083			√			
xxnFF90CH	CAN message data register 084	M_DATA084			√			
xxnFF90DH	CAN message data register 085	M_DATA085			√			
xxnFF90EH	CAN message data register 086	M_DATA086			√			
xxnFF90FH	CAN message data register 087	M_DATA087			√			
xxnFF910H	CAN message ID register L08	M_IDL08				√		
xxnFF912H	CAN message ID register H08	M_IDH08				√		
xxnFF914H	CAN message configuration register 08	M_CONF08			√			
xxnFF915H	CAN message status register 08	M_STAT08		R	√			
xxnFF916H	CAN status set/clear register 08	SC_STAT08	W			√	0000H	
xxnFF924H	CAN message data length register 09	M_DLC09	R/W		√			Undefined
xxnFF925H	CAN message control register 09	M_CTRL09			√			
xxnFF926H	CAN message time stamp register 09	M_TIME09				√		
xxnFF928H	CAN message data register 090	M_DATA090			√			
xxnFF929H	CAN message data register 091	M_DATA091			√			
xxnFF92AH	CAN message data register 092	M_DATA092			√			
xxnFF92BH	CAN message data register 093	M_DATA093			√			
xxnFF92CH	CAN message data register 094	M_DATA094			√			
xxnFF92DH	CAN message data register 095	M_DATA095			√			
xxnFF92EH	CAN message data register 096	M_DATA096			√			
xxnFF92FH	CAN message data register 097	M_DATA097			√			
xxnFF930H	CAN message ID register L09	M_IDL09				√		
xxnFF932H	CAN message ID register H09	M_IDH09				√		
xxnFF934H	CAN message configuration register 09	M_CONF09			√			
xxnFF935H	CAN message status register 09	M_STAT09		R	√			
xxnFF936H	CAN status set/clear register 09	SC_STAT09	W			√	0000H	
xxnFF944H	CAN message data length register 10	M_DLC10	R/W		√			Undefined
xxnFF945H	CAN message control register 10	M_CTRL10			√			
xxnFF946H	CAN message time stamp register 10	M_TIME10				√		
xxnFF948H	CAN message data register 100	M_DATA100			√			
xxnFF949H	CAN message data register 101	M_DATA101			√			
xxnFF94AH	CAN message data register 102	M_DATA102			√			
xxnFF94BH	CAN message data register 103	M_DATA103			√			

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset	
				1 Bit	8 Bit	16 Bit	32 Bit		
xxnFF94CH	CAN message data register 104	M_DATA104	R/W		√			Undefined	
xxnFF94DH	CAN message data register 105	M_DATA105			√				
xxnFF94EH	CAN message data register 106	M_DATA106			√				
xxnFF94FH	CAN message data register 107	M_DATA107			√				
xxnFF950H	CAN message ID register L10	M_IDL10				√			
xxnFF952H	CAN message ID register H10	M_IDH10				√			
xxnFF954H	CAN message configuration register 10	M_CONF10			√				
xxnFF955H	CAN message status register 10	M_STAT10	R		√				
xxnFF956H	CAN status set/clear register 10	SC_STAT10	W			√	0000H		
xxnFF964H	CAN message data length register 11	M_DLC11	R/W		√			Undefined	
xxnFF965H	CAN message control register 11	M_CTRL11			√				
xxnFF966H	CAN message time stamp register 11	M_TIME11				√			
xxnFF968H	CAN message data register 110	M_DATA110			√				
xxnFF969H	CAN message data register 111	M_DATA111			√				
xxnFF96AH	CAN message data register 112	M_DATA112			√				
xxnFF96BH	CAN message data register 113	M_DATA113			√				
xxnFF96CH	CAN message data register 114	M_DATA114			√				
xxnFF96DH	CAN message data register 115	M_DATA115			√				
xxnFF96EH	CAN message data register 116	M_DATA116			√				
xxnFF96FH	CAN message data register 117	M_DATA117			√				
xxnFF970H	CAN message ID register L11	M_IDL11				√			
xxnFF972H	CAN message ID register H11	M_IDH11				√			
xxnFF974H	CAN message configuration register 11	M_CONF11			√				
xxnFF975H	CAN message status register 11	M_STAT11		R		√			
xxnFF976H	CAN status set/clear register 11	SC_STAT11		W			√		0000H
xxnFF984H	CAN message data length register 12	M_DLC12		R/W		√			
xxnFF985H	CAN message control register 12	M_CTRL12			√				
xxnFF986H	CAN message time stamp register 12	M_TIME12				√			
xxnFF988H	CAN message data register 120	M_DATA120			√				
xxnFF989H	CAN message data register 121	M_DATA121			√				
xxnFF98AH	CAN message data register 122	M_DATA122			√				
xxnFF98BH	CAN message data register 123	M_DATA123			√				
xxnFF98CH	CAN message data register 124	M_DATA124			√				
xxnFF98DH	CAN message data register 125	M_DATA125			√				
xxnFF98EH	CAN message data register 126	M_DATA126			√				
xxnFF98FH	CAN message data register 127	M_DATA127			√				
xxnFF990H	CAN message ID register L12	M_IDL12				√			
xxnFF992H	CAN message ID register H12	M_IDH12				√			
xxnFF994H	CAN message configuration register 12	M_CONF12			√				
xxnFF995H	CAN message status register 12	M_STAT12	R			√			
xxnFF996H	CAN status set/clear register 12	SC_STAT12	W			√	0000H		
xxnFF9A4H	CAN message data length register 13	M_DLC13	R/W		√		Undefined		

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bit	16 Bit	32 Bit	
xxnFF9A5H	CAN message control register 13	M_CTRL13	R/W		√			Undefined
xxnFF9A6H	CAN message time stamp register 13	M_TIME13				√		
xxnFF9A8H	CAN message data register 130	M_DATA130			√			
xxnFF9A9H	CAN message data register 131	M_DATA131			√			
xxnFF9AAH	CAN message data register 132	M_DATA132			√			
xxnFF9ABH	CAN message data register 133	M_DATA133			√			
xxnFF9ACH	CAN message data register 134	M_DATA134			√			
xxnFF9ADH	CAN message data register 135	M_DATA135			√			
xxnFF9AEH	CAN message data register 136	M_DATA136			√			
xxnFF9AFH	CAN message data register 137	M_DATA137			√			
xxnFF9B0H	CAN message ID register L13	M_IDL13				√		
xxnFF9B2H	CAN message ID register H13	M_IDH13				√		
xxnFF9B4H	CAN message configuration register 13	M_CONF13			√			
xxnFF9B5H	CAN message status register 13	M_STAT13		R	√			
xxnFF9B6H	CAN status set/clear register 13	SC_STAT13	W		√		0000H	
xxnFF9C4H	CAN message data length register 14	M_DLC14	R/W		√			Undefined
xxnFF9C5H	CAN message control register 14	M_CTRL14			√			
xxnFF9C6H	CAN message time stamp register 14	M_TIME14				√		
xxnFF9C8H	CAN message data register 140	M_DATA140			√			
xxnFF9C9H	CAN message data register 141	M_DATA141			√			
xxnFF9CAH	CAN message data register 142	M_DATA142			√			
xxnFF9CBH	CAN message data register 143	M_DATA143			√			
xxnFF9CCH	CAN message data register 144	M_DATA144			√			
xxnFF9CDH	CAN message data register 145	M_DATA145			√			
xxnFF9CEH	CAN message data register 146	M_DATA146			√			
xxnFF9CFH	CAN message data register 147	M_DATA147			√			
xxnFF9D0H	CAN message ID register L14	M_IDL14				√		
xxnFF9D2H	CAN message ID register H14	M_IDH14				√		
xxnFF9D4H	CAN message configuration register 14	M_CONF14			√			
xxnFF9D5H	CAN message status register 14	M_STAT14	R	√				
xxnFF9D6H	CAN status set/clear register 14	SC_STAT14	W		√		0000H	
xxnFF9E4H	CAN message data length register 15	M_DLC15	R/W		√			Undefined
xxnFF9E5H	CAN message control register 15	M_CTRL15			√			
xxnFF9E6H	CAN message time stamp register 15	M_TIME15				√		
xxnFF9E8H	CAN message data register 150	M_DATA150			√			
xxnFF9E9H	CAN message data register 151	M_DATA151			√			
xxnFF9EAH	CAN message data register 152	M_DATA152			√			
xxnFF9EBH	CAN message data register 153	M_DATA153			√			
xxnFF9ECH	CAN message data register 154	M_DATA154			√			
xxnFF9EDH	CAN message data register 155	M_DATA155			√			
xxnFF9EEH	CAN message data register 156	M_DATA156			√			
xxnFF9EFH	CAN message data register 157	M_DATA157			√			

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset	
				1 Bit	8 Bit	16 Bit	32 Bit		
xxnFF9F0H	CAN message ID register L15	M_IDL15	R/W			√		Undefined	
xxnFF9F2H	CAN message ID register H15	M_IDH15				√			
xxnFF9F4H	CAN message configuration register 15	M_CONF15			√				
xxnFF9F5H	CAN message status register 15	M_STAT15	R		√				
xxnFF9F6H	CAN status set/clear register 15	SC_STAT15	W			√		0000H	
xxnFFA04H	CAN message data length register 16	M_DLC16	R/W		√			Undefined	
xxnFFA05H	CAN message control register 16	M_CTRL16			√				
xxnFFA06H	CAN message time stamp register 16	M_TIME16				√			
xxnFFA08H	CAN message data register 160	M_DATA160			√				
xxnFFA09H	CAN message data register 161	M_DATA161			√				
xxnFFA0AH	CAN message data register 162	M_DATA162			√				
xxnFFA0BH	CAN message data register 163	M_DATA163			√				
xxnFFA0CH	CAN message data register 164	M_DATA164			√				
xxnFFA0DH	CAN message data register 165	M_DATA165			√				
xxnFFA0EH	CAN message data register 166	M_DATA166			√				
xxnFFA0FH	CAN message data register 167	M_DATA167			√				
xxnFFA10H	CAN message ID register L16	M_IDL16				√			
xxnFFA12H	CAN message ID register H16	M_IDH16				√			
xxnFFA14H	CAN message configuration register 16	M_CONF16			√				
xxnFFA15H	CAN message status register 16	M_STAT16		R		√			
xxnFFA16H	CAN status set/clear register 16	SC_STAT16		W			√		0000H
xxnFFA24H	CAN message data length register 17	M_DLC17		R/W		√			
xxnFFA25H	CAN message control register 17	M_CTRL17			√				
xxnFFA26H	CAN message time stamp register 17	M_TIME17				√			
xxnFFA28H	CAN message data register 170	M_DATA170			√				
xxnFFA29H	CAN message data register 171	M_DATA171			√				
xxnFFA2AH	CAN message data register 172	M_DATA172			√				
xxnFFA2BH	CAN message data register 173	M_DATA173			√				
xxnFFA2CH	CAN message data register 174	M_DATA174			√				
xxnFFA2DH	CAN message data register 175	M_DATA175			√				
xxnFFA2EH	CAN message data register 176	M_DATA176			√				
xxnFFA2FH	CAN message data register 177	M_DATA177			√				
xxnFFA30H	CAN message ID register L17	M_IDL17				√			
xxnFFA32H	CAN message ID register H17	M_IDH17				√			
xxnFFA34H	CAN message configuration register 17	M_CONF17			√				
xxnFFA35H	CAN message status register 17	M_STAT17	R			√			
xxnFFA36H	CAN status set/clear register 17	SC_STAT17	W				√	0000H	
xxnFFA44H	CAN message data length register 18	M_DLC18	R/W			√			Undefined
xxnFFA45H	CAN message control register 18	M_CTRL18			√				
xxnFFA46H	CAN message time stamp register 18	M_TIME18				√			
xxnFFA48H	CAN message data register 180	M_DATA180			√				
xxnFFA49H	CAN message data register 181	M_DATA181			√				

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bit	16 Bit	32 Bit	
xxnFFA4AH	CAN message data register 182	M_DATA182	R/W		√			Undefined
xxnFFA4BH	CAN message data register 183	M_DATA183			√			
xxnFFA4CH	CAN message data register 184	M_DATA184			√			
xxnFFA4DH	CAN message data register 185	M_DATA185			√			
xxnFFA4EH	CAN message data register 186	M_DATA186			√			
xxnFFA4FH	CAN message data register 187	M_DATA187			√			
xxnFFA50H	CAN message ID register L18	M_IDL18				√		
xxnFFA52H	CAN message ID register H18	M_IDH18				√		
xxnFFA54H	CAN message configuration register 18	M_CONF18			√			
xxnFFA55H	CAN message status register 18	M_STAT18		R		√		
xxnFFA56H	CAN status set/clear register 18	SC_STAT18	W			√	0000H	
xxnFFA64H	CAN message data length register 19	M_DLC19	R/W		√			Undefined
xxnFFA65H	CAN message control register 19	M_CTRL19			√			
xxnFFA66H	CAN message time stamp register 19	M_TIME19				√		
xxnFFA68H	CAN message data register 190	M_DATA190			√			
xxnFFA69H	CAN message data register 191	M_DATA191			√			
xxnFFA6AH	CAN message data register 192	M_DATA192			√			
xxnFFA6BH	CAN message data register 193	M_DATA193			√			
xxnFFA6CH	CAN message data register 194	M_DATA194			√			
xxnFFA6DH	CAN message data register 195	M_DATA195			√			
xxnFFA6EH	CAN message data register 196	M_DATA196			√			
xxnFFA6FH	CAN message data register 197	M_DATA197		√				
xxnFFA70H	CAN message ID register L19	M_IDL19			√			
xxnFFA72H	CAN message ID register H19	M_IDH19			√			
xxnFFA74H	CAN message configuration register 19	M_CONF19		√				
xxnFFA75H	CAN message status register 19	M_STAT19	R		√			
xxnFFA76H	CAN status set/clear register 19	SC_STAT19	W			√	0000H	
xxnFFA84H	CAN message data length register 20	M_DLC20	R/W		√			Undefined
xxnFFA85H	CAN message control register 20	M_CTRL20			√			
xxnFFA86H	CAN message time stamp register 20	M_TIME20				√		
xxnFFA88H	CAN message data register 200	M_DATA200			√			
xxnFFA89H	CAN message data register 201	M_DATA201			√			
xxnFFA8AH	CAN message data register 202	M_DATA202			√			
xxnFFA8BH	CAN message data register 203	M_DATA203			√			
xxnFFA8CH	CAN message data register 204	M_DATA204			√			
xxnFFA8DH	CAN message data register 205	M_DATA205			√			
xxnFFA8EH	CAN message data register 206	M_DATA206			√			
xxnFFA8FH	CAN message data register 207	M_DATA207		√				
xxnFFA90H	CAN message ID register L20	M_IDL20			√			
xxnFFA92H	CAN message ID register H20	M_IDH20			√			
xxnFFA94H	CAN message configuration register 20	M_CONF20		√				
xxnFFA95H	CAN message status register 20	M_STAT20	R		√			

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bit	16 Bit	32 Bit	
xxnFFA96H	CAN status set/clear register 20	SC_STAT20	W			√		0000H
xxnFFAA4H	CAN message data length register 21	M_DLC21	R/W		√			Undefined
xxnFFAA5H	CAN message control register 21	M_CTRL21			√			
xxnFFAA6H	CAN message time stamp register 21	M_TIME21				√		
xxnFFAA8H	CAN message data register 210	M_DATA210			√			
xxnFFAA9H	CAN message data register 211	M_DATA211			√			
xxnFFAAAH	CAN message data register 212	M_DATA212			√			
xxnFFAABH	CAN message data register 213	M_DATA213			√			
xxnFFAACH	CAN message data register 214	M_DATA214			√			
xxnFFAADH	CAN message data register 215	M_DATA215			√			
xxnFFAAEH	CAN message data register 216	M_DATA216			√			
xxnFFAAFH	CAN message data register 217	M_DATA217			√			
xxnFFAB0H	CAN message ID register L21	M_IDL21				√		
xxnFFAB2H	CAN message ID register H21	M_IDH21				√		
xxnFFAB4H	CAN message configuration register 21	M_CONF21		√				
xxnFFAB5H	CAN message status register 21	M_STAT21	R		√			
xxnFFAB6H	CAN status set/clear register 21	SC_STAT21	W			√	0000H	
xxnFFAC4H	CAN message data length register 22	M_DLC22	R/W		√			Undefined
xxnFFAC5H	CAN message control register 22	M_CTRL22			√			
xxnFFAC6H	CAN message time stamp register 22	M_TIME22				√		
xxnFFAC8H	CAN message data register 220	M_DATA220			√			
xxnFFAC9H	CAN message data register 221	M_DATA221			√			
xxnFFACAH	CAN message data register 222	M_DATA222			√			
xxnFFACBH	CAN message data register 223	M_DATA223			√			
xxnFFACCH	CAN message data register 224	M_DATA224			√			
xxnFFACDH	CAN message data register 225	M_DATA225			√			
xxnFFACEH	CAN message data register 226	M_DATA226			√			
xxnFFACFH	CAN message data register 227	M_DATA227			√			
xxnFFAD0H	CAN message ID register L22	M_IDL22				√		
xxnFFAD2H	CAN message ID register H22	M_IDH22				√		
xxnFFAD4H	CAN message configuration register 22	M_CONF22		√				
xxnFFAD5H	CAN message status register 22	M_STAT22	R		√			
xxnFFAD6H	CAN status set/clear register 22	SC_STAT22	W			√	0000H	
xxnFFAE4H	CAN message data length register 23	M_DLC23	R/W		√			Undefined
xxnFFAE5H	CAN message control register 23	M_CTRL23			√			
xxnFFAE6H	CAN message time stamp register 23	M_TIME23				√		
xxnFFAE8H	CAN message data register 230	M_DATA230			√			
xxnFFAE9H	CAN message data register 231	M_DATA231			√			
xxnFFAEAH	CAN message data register 232	M_DATA232			√			
xxnFFAEBH	CAN message data register 233	M_DATA233			√			
xxnFFAECH	CAN message data register 234	M_DATA234			√			
xxnFFAEDH	CAN message data register 235	M_DATA235			√			

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset		
				1 Bit	8 Bit	16 Bit	32 Bit			
xxnFFAEEH	CAN message data register 236	M_DATA236	R/W		√			Undefined		
xxnFFAEFH	CAN message data register 237	M_DATA237			√					
xxnFFAF0H	CAN message ID register L23	M_IDL23				√				
xxnFFAF2H	CAN message ID register H23	M_IDH23				√				
xxnFFAF4H	CAN message configuration register 23	M_CONF23			√					
xxnFFAF5H	CAN message status register 23	M_STAT23	R		√					
xxnFFAF6H	CAN status set/clear register 23	SC_STAT23	W			√		0000H		
xxnFFB04H	CAN message data length register 24	M_DLC24	R/W		√			Undefined		
xxnFFB05H	CAN message control register 24	M_CTRL24			√					
xxnFFB06H	CAN message time stamp register 24	M_TIME24				√				
xxnFFB08H	CAN message data register 240	M_DATA240			√					
xxnFFB09H	CAN message data register 241	M_DATA241			√					
xxnFFB0AH	CAN message data register 242	M_DATA242			√					
xxnFFB0BH	CAN message data register 243	M_DATA243			√					
xxnFFB0CH	CAN message data register 244	M_DATA244			√					
xxnFFB0DH	CAN message data register 245	M_DATA245			√					
xxnFFB0EH	CAN message data register 246	M_DATA246			√					
xxnFFB0FH	CAN message data register 247	M_DATA247			√					
xxnFFB10H	CAN message ID register L24	M_IDL24				√				
xxnFFB12H	CAN message ID register H24	M_IDH24				√				
xxnFFB14H	CAN message configuration register 24	M_CONF24			√					
xxnFFB15H	CAN message status register 24	M_STAT24		R		√				
xxnFFB16H	CAN status set/clear register 24	SC_STAT24		W			√			0000H
xxnFFB24H	CAN message data length register 25	M_DLC25		R/W		√				Undefined
xxnFFB25H	CAN message control register 25	M_CTRL25				√				
xxnFFB26H	CAN message time stamp register 25	M_TIME25					√			
xxnFFB28H	CAN message data register 250	M_DATA250			√					
xxnFFB29H	CAN message data register 251	M_DATA251			√					
xxnFFB2AH	CAN message data register 252	M_DATA252			√					
xxnFFB2BH	CAN message data register 253	M_DATA253			√					
xxnFFB2CH	CAN message data register 254	M_DATA254			√					
xxnFFB2DH	CAN message data register 255	M_DATA255			√					
xxnFFB2EH	CAN message data register 256	M_DATA256			√					
xxnFFB2FH	CAN message data register 257	M_DATA257			√					
xxnFFB30H	CAN message ID register L25	M_IDL25				√				
xxnFFB32H	CAN message ID register H25	M_IDH25				√				
xxnFFB34H	CAN message configuration register 25	M_CONF25			√					
xxnFFB35H	CAN message status register 25	M_STAT25	R			√				
xxnFFB36H	CAN status set/clear register 25	SC_STAT25	W			√		0000H		
xxnFFB44H	CAN message data length register 26	M_DLC26	R/W		√			Undefined		
xxnFFB45H	CAN message control register 26	M_CTRL26			√					
xxnFFB46H	CAN message time stamp register 26	M_TIME26				√				
xxnFFB48H	CAN message data register 260	M_DATA260			√					

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset	
				1 Bit	8 Bit	16 Bit	32 Bit		
xxnFFB49H	CAN message data register 261	M_DATA261	R/W		√			Undefined	
xxnFFB4AH	CAN message data register 262	M_DATA262			√				
xxnFFB4BH	CAN message data register 263	M_DATA263			√				
xxnFFB4CH	CAN message data register 264	M_DATA264			√				
xxnFFB4DH	CAN message data register 265	M_DATA265			√				
xxnFFB4EH	CAN message data register 266	M_DATA266			√				
xxnFFB4FH	CAN message data register 267	M_DATA267			√				
xxnFFB50H	CAN message ID register L26	M_IDL26				√			
xxnFFB52H	CAN message ID register H26	M_IDH26				√			
xxnFFB54H	CAN message configuration register 26	M_CONF26			√				
xxnFFB55H	CAN message status register 26	M_STAT26		R	√				
xxnFFB56H	CAN status set/clear register 26	SC_STAT26	W		√		0000H		
xxnFFB64H	CAN message data length register 27	M_DLC27	R/W		√			Undefined	
xxnFFB65H	CAN message control register 27	M_CTRL27			√				
xxnFFB66H	CAN message time stamp register 27	M_TIME27				√			
xxnFFB68H	CAN message data register 270	M_DATA270			√				
xxnFFB69H	CAN message data register 271	M_DATA271			√				
xxnFFB6AH	CAN message data register 272	M_DATA272			√				
xxnFFB6BH	CAN message data register 273	M_DATA273			√				
xxnFFB6CH	CAN message data register 274	M_DATA274			√				
xxnFFB6DH	CAN message data register 275	M_DATA275			√				
xxnFFB6EH	CAN message data register 276	M_DATA276			√				
xxnFFB6FH	CAN message data register 277	M_DATA277			√				
xxnFFB70H	CAN message ID register L27	M_IDL27				√			
xxnFFB72H	CAN message ID register H27	M_IDH27				√			
xxnFFB74H	CAN message configuration register 27	M_CONF27			√				
xxnFFB75H	CAN message status register 27	M_STAT27		R	√				
xxnFFB76H	CAN status set/clear register 27	SC_STAT27		W		√			0000H
xxnFFB84H	CAN message data length register 28	M_DLC28		R/W		√			
xxnFFB85H	CAN message control register 28	M_CTRL28			√				
xxnFFB86H	CAN message time stamp register 28	M_TIME28				√			
xxnFFB88H	CAN message data register 280	M_DATA280			√				
xxnFFB89H	CAN message data register 281	M_DATA281			√				
xxnFFB8AH	CAN message data register 282	M_DATA282			√				
xxnFFB8BH	CAN message data register 283	M_DATA283			√				
xxnFFB8CH	CAN message data register 284	M_DATA284			√				
xxnFFB8DH	CAN message data register 285	M_DATA285			√				
xxnFFB8EH	CAN message data register 286	M_DATA286			√				
xxnFFB8FH	CAN message data register 287	M_DATA287			√				
xxnFFB90H	CAN message ID register L28	M_IDL28				√			
xxnFFB92H	CAN message ID register H28	M_IDH28				√			
xxnFFB94H	CAN message configuration register 28	M_CONF28			√				

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bit	16 Bit	32 Bit	
xxnFFB95H	CAN message status register 28	M_STAT28	R		√			Undefined
xxnFFB96H	CAN status set/clear register 28	SC_STAT28	W			√		0000H
xxnFFBA4H	CAN message data length register 29	M_DLC29	R/W		√			Undefined
xxnFFBA5H	CAN message control register 29	M_CTRL29			√			
xxnFFBA6H	CAN message time stamp register 29	M_TIME29				√		
xxnFFBA8H	CAN message data register 290	M_DATA290			√			
xxnFFBA9H	CAN message data register 291	M_DATA291			√			
xxnFFBAAH	CAN message data register 292	M_DATA292			√			
xxnFFBABH	CAN message data register 293	M_DATA293			√			
xxnFFBACH	CAN message data register 294	M_DATA294			√			
xxnFFBADH	CAN message data register 295	M_DATA295			√			
xxnFFBAEH	CAN message data register 296	M_DATA296			√			
xxnFFBAFH	CAN message data register 297	M_DATA297			√			
xxnFFBB0H	CAN message ID register L29	M_IDL29				√		
xxnFFBB2H	CAN message ID register H29	M_IDH29				√		
xxnFFBB4H	CAN message configuration register 29	M_CONF29			√			
xxnFFBB5H	CAN message status register 29	M_STAT29	R		√			
xxnFFBB6H	CAN status set/clear register 29	SC_STAT29	W			√	0000H	
xxnFFBC4H	CAN message data length register 30	M_DLC30	R/W		√			Undefined
xxnFFBC5H	CAN message control register 30	M_CTRL30			√			
xxnFFBC6H	CAN message time stamp register 30	M_TIME30				√		
xxnFFBC8H	CAN message data register 300	M_DATA300			√			
xxnFFBC9H	CAN message data register 301	M_DATA301			√			
xxnFFBCAH	CAN message data register 302	M_DATA302			√			
xxnFFBCBH	CAN message data register 303	M_DATA303			√			
xxnFFBCCH	CAN message data register 304	M_DATA304			√			
xxnFFBCDH	CAN message data register 305	M_DATA305			√			
xxnFFBCEH	CAN message data register 306	M_DATA306			√			
xxnFFBCFH	CAN message data register 307	M_DATA307			√			
xxnFFBD0H	CAN message ID register L30	M_IDL30				√		
xxnFFBD2H	CAN message ID register H30	M_IDH30				√		
xxnFFBD4H	CAN message configuration register 30	M_CONF30			√			
xxnFFBD5H	CAN message status register 30	M_STAT30	R		√			
xxnFFBD6H	CAN status set/clear register 30	SC_STAT30	W			√	0000H	
xxnFFBE4H	CAN message data length register 31	M_DLC31	R/W		√			Undefined
xxnFFBE5H	CAN message control register 31	M_CTRL31			√			
xxnFFBE6H	CAN message time stamp register 31	M_TIME31				√		
xxnFFBE8H	CAN message data register 310	M_DATA310			√			
xxnFFBE9H	CAN message data register 311	M_DATA311			√			
xxnFFBEAH	CAN message data register 312	M_DATA312			√			
xxnFFBEBH	CAN message data register 313	M_DATA313			√			
xxnFFBECH	CAN message data register 314	M_DATA314			√			

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset	
				1 Bit	8 Bit	16 Bit	32 Bit		
xxnFFBEDH	CAN message data register 315	M_DATA315	R/W		√			Undefined	
xxnFFBEEH	CAN message data register 316	M_DATA316			√				
xxnFFBEFH	CAN message data register 317	M_DATA317			√				
xxnFFBF0H	CAN message ID register L31	M_IDL31				√			
xxnFFBF2H	CAN message ID register H31	M_IDH31				√			
xxnFFBF4H	CAN message configuration register 31	M_CONF31			√				
xxnFFBF5H	CAN message status register 31	M_STAT31	R		√				
xxnFFBF6H	CAN status set/clear register 31	SC_STAT31	W			√	0000H		
xxnFFC00H	CAN interrupt pending register	CCINTP	R			√			
xxnFFC02H	CAN global interrupt pending register	CGINTP	R/W		√			00H	
xxnFFC04H	CAN1 interrupt pending register	C1INTP			√				
xxnFFC06H	CAN2 interrupt pending register ^{Note 1}	C2INTP			√				
xxnFFC0CH	CAN stop register	CSTOP				√			0000H
xxnFFC10H	CAN global status register	CGST			√ ^{Note 2}	√			0100H
xxnFFC12H	CAN global interrupt enable register	CGIE			√ ^{Note 2}	√			0A00H
xxnFFC14H	CAN main clock selection register	CGCS			√		7F05H		
xxnFFC18H	CAN time stamp count register	CGTSC	R		√		0000H		
xxnFFC1AH	CAN message search start register	CGMSS	W		√				
	CAN message search result register	CGMSR	R		√				
xxnFFC40H	CAN1 address mask 0 register L	C1MASKL0	R/W			√		Undefined	
xxnFFC42H	CAN1 address mask 0 register H	C1MASKH0				√			
xxnFFC44H	CAN1 address mask 1 register L	C1MASKL1				√			
xxnFFC46H	CAN1 address mask 1 register H	C1MASKH1				√			
xxnFFC48H	CAN1 address mask 2 register L	C1MASKL2				√			
xxnFFC4AH	CAN1 address mask 2 register H	C1MASKH2				√			
xxnFFC4CH	CAN1 address mask 3 register L	C1MASKL3				√			
xxnFFC4EH	CAN1 address mask 3 register H	C1MASKH3				√			
xxnFFC50H	CAN1 control register	C1CTRL				√			0101H
xxnFFC52H	CAN1 definition register	C1DEF			√ ^{Note 2}	√			0000H
xxnFFC54H	CAN1 information register	C1LAST	R		√		00FFH		
xxnFFC56H	CAN1 error count register	C1ERC			√		0000H		
xxnFFC58H	CAN1 interrupt enable register	C1IE	R/W	√ ^{Note 2}	√		0900H		
xxnFFC5AH	CAN1 bus active register	C1BA	R		√		00FFH		
xxnFFC5CH	CAN1 bit rate prescaler register	C1BRP	R/W			√	0000H		
	CAN1 bus diagnostic information register	C1DINF	R			√			
xxnFFC5EH	CAN1 synchronization control register	C1SYNC	R/W			√	0218H		
xxnFFC80H	CAN2 address mask 0 register L ^{Note 1}	C2MASKL0	R/W			√	Undefined		
xxnFFC82H	CAN2 address mask 0 register H ^{Note 1}	C2MASKH0				√			
xxnFFC84H	CAN2 address mask 1 register L ^{Note 1}	C2MASKL1				√			
xxnFFC86H	CAN2 address mask 1 register H ^{Note 1}	C2MASKH1				√			
xxnFFC88H	CAN2 address mask 2 register L ^{Note 1}	C2MASKL2				√			

Notes 1. μ PD703079Y and 70F3079Y only

2. Read only

Remark n = 3, 7, B

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation				After Reset
				1 Bit	8 Bit	16 Bit	32 Bit	
xxnFFC8AH	CAN2 address mask 2 register H ^{Note 1}	C2MASKH2	R/W			√		Undefined
xxnFFC8CH	CAN2 address mask 3 register L ^{Note 1}	C2MASKL3				√		
xxnFFC8EH	CAN2 address mask 3 register H ^{Note 1}	C2MASKH3				√		
xxnFFC90H	CAN2 control register ^{Note 1}	C2CTRL				√		0101H
xxnFFC92H	CAN2 definition register ^{Note 1}	C2DEF			√ ^{Note 2}	√		0000H
xxnFFC94H	CAN2 information register ^{Note 1}	C2LAST	R			√		00FFH
xxnFFC96H	CAN2 error count register ^{Note 1}	C2ERC				√		0000H
xxnFFC98H	CAN2 interrupt enable register ^{Note 1}	C2IE	R/W		√ ^{Note 2}	√		0900H
xxnFFC9AH	CAN2 bus active register ^{Note 1}	C2BA	R			√		00FFH
xxnFFC9CH	CAN2 bit rate prescaler register ^{Note 1}	C2BRP	R/W			√		0000H
	CAN2 bus diagnostic information register ^{Note 1}	C2DINF	R			√		
xxnFFC9EH	CAN2 synchronization control register ^{Note 1}	C2SYNC	R/W			√		0218H

Notes 1. μ PD703079Y and 70F3079Y only

2. Read only

Remark n = 3, 7, B

18.4 Control Registers

18.4.1 CAN message data length registers 00 to 31 (M_DLC00 to M_DLC31)

The M_DLCn register sets the byte count in the data field of CAN message buffer n (n = 00 to 31). When receiving, the receive data field's byte count is set (1).

These registers can be read/written in 8-bit units.

After reset: Undefined		R/W		Address: See Table 18-3				
	7	6	5	4	3	2	1	0
M_DLCn	0	0	0	0	DLC3	DLC2	DLC1	DLC0
(n = 00 to 31)								
	DLC3	DLC2	DLC1	DLC0	Data length code of transmit/receive message			
	0	0	0	0	0 bytes			
	0	0	0	1	1 byte			
	0	0	1	0	2 bytes			
	0	0	1	1	3 bytes			
	0	1	0	0	4 bytes			
	0	1	0	1	5 bytes			
	0	1	1	0	6 bytes			
	0	1	1	1	7 bytes			
	1	0	0	0	8 bytes			
★	Other than above				8 bytes regardless of DLC3 to DLC0 values			

Table 18-3. Addresses of M_DLCn (n = 00 to 31)

Register Name	Address (m = 3, 7, B)	Register Name	Address (m = 3, 7, B)
M_DLC00	xxmFF804H	M_DLC16	xxmFFA04H
M_DLC01	xxmFF824H	M_DLC17	xxmFFA24H
M_DLC02	xxmFF844H	M_DLC18	xxmFFA44H
M_DLC03	xxmFF864H	M_DLC19	xxmFFA64H
M_DLC04	xxmFF884H	M_DLC20	xxmFFA84H
M_DLC05	xxmFF8A4H	M_DLC21	xxmFFAA4H
M_DLC06	xxmFF8C4H	M_DLC22	xxmFFAC4H
M_DLC07	xxmFF8E4H	M_DLC23	xxmFFAE4H
M_DLC08	xxmFF904H	M_DLC24	xxmFFB04H
M_DLC09	xxmFF924H	M_DLC25	xxmFFB24H
M_DLC10	xxmFF944H	M_DLC26	xxmFFB44H
M_DLC11	xxmFF964H	M_DLC27	xxmFFB64H
M_DLC12	xxmFF984H	M_DLC28	xxmFFB84H
M_DLC13	xxmFF9A4H	M_DLC29	xxmFFBA4H
M_DLC14	xxmFF9C4H	M_DLC30	xxmFFBC4H
M_DLC15	xxmFF9E4H	M_DLC31	xxmFFBE4H

18.4.2 CAN message control registers 00 to 31 (M_CTRL00 to M_CTRL31)

The M_CTRLn register is used to set the frame format of the data field in messages stored in CAN message buffer n (n = 00 to 31).

These registers can be read/written in 8-bit units.

(1/2)

★

After reset: Undefined	R/W	Address: See Table 18-4						
	7	6	5	4	3	2	1	0
M_CTRLn (n = 00 to 31)	RMDE1	RMDE0	ATS	IE	MOVR	Undefined ^{Note 1}	Undefined ^{Note 2}	RTR

RMDE1	Specifies operation of DN flag when remote frame is received on a transmit message buffer
0	DN flag not set when remote frame is received
1	DN flag set when remote frame is received
<ul style="list-style-type: none"> • When the RMDE1 bit is set, the setting of the RMDE0 bit is irrelevant. • If a remote frame is received at the transmit message buffer when the RMDE1 bit has not been set, the CPU is not notified, nor are other operations performed. 	

RMDE0	Specification of set/clear status of remote frame auto acknowledge function
0	Remote frame auto acknowledge function cleared
1	Remote frame auto acknowledge function set
<ul style="list-style-type: none"> • The RMDE0 bit's setting is used only for transmit messages. • When the RTR bit has been set (1) (when the receive message or transmit message has a remote frame), the RMDE0 bit is processed as RMDE0 = 0. This prevents a worst-case scenario (in which transmission of a remote frame draws a 100% bus load due to reception of the same remote frame). 	

ATS	Specifies whether or not to add a time stamp when transmitting
0	Time stamp not added when transmitting
1	Time stamp added when transmitting
<ul style="list-style-type: none"> • The ATS bit is used only for transmit messages. • When the ATS bit has been set (1) and the data length code specifies at least two bytes, the last two bytes are replaced by a time stamp (see Table 18-12). The added time stamp counter value is sent to the bus via the message's SOF. When this occurs, the last two bytes (which are defined as a data field) are ignored. 	

- Notes**
1. The value of the r1 bit on the CAN bus is set during reception.
 2. The value of the r0 bit on the CAN bus is set during reception.

Remark DN: Bit 2 of M_STATm (see **18.4.7 CAN message status registers 00 to 31 (M_STAT00 to M_STAT31)**)

IE	Specifies the enable/disable setting for interrupt requests
0	Interrupt requests disabled
1	Interrupt requests enabled

- An interrupt request occurs when an interrupt is enabled under the following conditions.
 - When a message is sent from the transmit message buffer
 - When a message is received by the receive message buffer
 - When a remote frame is received by the transmit message buffer when the auto acknowledge function has not been set (RMDE0 bit = 0).
- An interrupt request does not occur when an interrupt is enabled under the following conditions.
 - When a remote frame is received by the transmit message buffer when the auto acknowledge function has been set (RMDE0 bit = 1)
 - When a remote frame has been transmitted from the receive message buffer

MOVR	Message buffer overwrite
0	Overwrite does not occur after DN bit is cleared
1	Overwrite occurs at least once after DN bit is cleared

- An overwrite of the message buffer occurs when the CAN module writes new data to the message buffer or when the DN bit has already been set (1). The MOVR bit is updated each time new data is stored in the message buffer.

RTR	Specification of frame type
0	Data frame transmit/receive
1	Remote frame transmit/receive

- When the RTR bit has been set (1) for a transmit message, a remote frame is transmitted instead of a data frame.

Remark DN: Bit 2 of M_STATm (see 18.4.7 CAN message status registers 00 to 31 (M_STAT00 to M_STAT31))

Table 18-4. Addresses of M_CTRLn (n = 00 to 31)

Register Name	Address (m = 3, 7, B)	Register Name	Address (m = 3, 7, B)
M_CTRL00	xxmFF805H	M_CTRL16	xxmFFA05H
M_CTRL01	xxmFF825H	M_CTRL17	xxmFFA25H
M_CTRL02	xxmFF845H	M_CTRL18	xxmFFA45H
M_CTRL03	xxmFF865H	M_CTRL19	xxmFFA65H
M_CTRL04	xxmFF885H	M_CTRL20	xxmFFA85H
M_CTRL05	xxmFF8A5H	M_CTRL21	xxmFFAA5H
M_CTRL06	xxmFF8C5H	M_CTRL22	xxmFFAC5H
M_CTRL07	xxmFF8E5H	M_CTRL23	xxmFFAE5H
M_CTRL08	xxmFF905H	M_CTRL24	xxmFFB05H
M_CTRL09	xxmFF925H	M_CTRL25	xxmFFB25H
M_CTRL10	xxmFF945H	M_CTRL26	xxmFFB45H
M_CTRL11	xxmFF965H	M_CTRL27	xxmFFB65H
M_CTRL12	xxmFF985H	M_CTRL28	xxmFFB85H
M_CTRL13	xxmFF9A5H	M_CTRL29	xxmFFBA5H
M_CTRL14	xxmFF9C5H	M_CTRL30	xxmFFBC5H
M_CTRL15	xxmFF9E5H	M_CTRL31	xxmFFBE5H

18.4.3 CAN message time stamp registers 00 to 31 (M_TIME00 to M_TIME31)

The M_TIME_n register is the register where the time stamp counter value is written upon completion of data reception (n = 00 to 31).

These registers can be read/written in 16-bit units.

When a data frame or remote frame is received in the receive message buffer, the new data is stored in the message buffer and a 16-bit time tag (time stamp counter value) is stored in the M_TIME_n register. This time tag is set according to the FCAN's time stamp setting, which is either the time stamp counter value that was captured when the SOF was sent on the CAN bus or the value captured when the CAN module writes data to the message buffer.

After reset: Undefined		R/W		Address: See Table 18-5												
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M_TIME _n	TS	TS	TS	TS	TS	TS	TS	TS	TS	TS	TS	TS	TS	TS	TS	TS
(n = 00 to 31)	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00

Table 18-5. Addresses of M_TIME_n (n = 00 to 31)

Register Name	Address (m = 3, 7, B)	Register Name	Address (m = 3, 7, B)
M_TIME00	xxmFF806H	M_TIME16	xxmFFA06H
M_TIME01	xxmFF826H	M_TIME17	xxmFFA26H
M_TIME02	xxmFF846H	M_TIME18	xxmFFA46H
M_TIME03	xxmFF866H	M_TIME19	xxmFFA66H
M_TIME04	xxmFF886H	M_TIME20	xxmFFA86H
M_TIME05	xxmFF8A6H	M_TIME21	xxmFFAA6H
M_TIME06	xxmFF8C6H	M_TIME22	xxmFFAC6H
M_TIME07	xxmFF8E6H	M_TIME23	xxmFFAE6H
M_TIME08	xxmFF906H	M_TIME24	xxmFFB06H
M_TIME09	xxmFF926H	M_TIME25	xxmFFB26H
M_TIME10	xxmFF946H	M_TIME26	xxmFFB46H
M_TIME11	xxmFF966H	M_TIME27	xxmFFB66H
M_TIME12	xxmFF986H	M_TIME28	xxmFFB86H
M_TIME13	xxmFF9A6H	M_TIME29	xxmFFBA6H
M_TIME14	xxmFF9C6H	M_TIME30	xxmFFBC6H
M_TIME15	xxmFF9E6H	M_TIME31	xxmFFBE6H

18.4.4 CAN message data registers n0 to n7 (M_DATAn0 to M_DATAn7)

The M_DATAn0 to M_DATAn7 registers are areas where up to 8 bytes of transmit or receive data is stored.

These registers can be read/written in 8-bit units.

The M_DATAn0 to M_DATAn7 registers are fields used to hold receive data and transmit data. When data is transmitted, the number of messages defined by the DLC3 to DLC0 bits in the M_DLCn register are transmitted via the CAN bus.

When the M_CTRLn register's ATS bit has been set (1) and the value of the DLC3 to DLC0 bits in the M_DLCn register is at least two bytes, the last two bytes that are sent normally via the CAN bus are ignored and the time stamp value is sent.

When a new message is received, all data fields are updated, even when the value of the DLC3 to DLC0 bits in the M_DLCn register is less than 8 bytes. The values of data bytes that have not been received may be updated, but they are ignored.

Remark n = 00 to 31, x = 0 to 7

M_DATAn0	7	6	5	4	3	2	1	0	Address	After reset
	D0_7	D0_6	D0_5	D0_4	D0_3	D0_2	D0_1	D0_0	See Table 18-6	Undefined
(n = 00 to 31)										
M_DATAn1	7	6	5	4	3	2	1	0	Address	After reset
	D1_7	D1_6	D1_5	D1_4	D1_3	D1_2	D1_1	D1_0	See Table 18-6	Undefined
(n = 00 to 31)										
M_DATAn2	7	6	5	4	3	2	1	0	Address	After reset
	D2_7	D2_6	D2_5	D2_4	D2_3	D2_2	D2_1	D2_0	See Table 18-6	Undefined
(n = 00 to 31)										
M_DATAn3	7	6	5	4	3	2	1	0	Address	After reset
	D3_7	D3_6	D3_5	D3_4	D3_3	D3_2	D3_1	D3_0	See Table 18-6	Undefined
(n = 00 to 31)										
M_DATAn4	7	6	5	4	3	2	1	0	Address	After reset
	D4_7	D4_6	D4_5	D4_4	D4_3	D4_2	D4_1	D4_0	See Table 18-6	Undefined
(n = 00 to 31)										
M_DATAn5	7	6	5	4	3	2	1	0	Address	After reset
	D5_7	D5_6	D5_5	D5_4	D5_3	D5_2	D5_1	D5_0	See Table 18-6	Undefined
(n = 00 to 31)										
M_DATAn6	7	6	5	4	3	2	1	0	Address	After reset
	D6_7	D6_6	D6_5	D6_4	D6_3	D6_2	D6_1	D6_0	See Table 18-6	Undefined
(n = 00 to 31)										
M_DATAn7	7	6	5	4	3	2	1	0	Address	After reset
	D7_7	D7_6	D7_5	D7_4	D7_3	D7_2	D7_1	D7_0	See Table 18-6	Undefined
(n = 00 to 31)										

Table 18-6. Addresses of M_DATAnx (n = 00 to 31, x = 0 to 7)

Register Name n	M_DATAn0 (m = 3, 7, B)	M_DATAn1 (m = 3, 7, B)	M_DATAn2 (m = 3, 7, B)	M_DATAn3 (m = 3, 7, B)	M_DATAn4 (m = 3, 7, B)	M_DATAn5 (m = 3, 7, B)	M_DATAn6 (m = 3, 7, B)	M_DATAn7 (m = 3, 7, B)
00	xxmFF808H	xxmFF809H	xxmFF80AH	xxmFF80BH	xxmFF80CH	xxmFF80DH	xxmFF80EH	xxmFF80FH
01	xxmFF828H	xxmFF829H	xxmFF82AH	xxmFF82BH	xxmFF82CH	xxmFF82DH	xxmFF82EH	xxmFF82FH
02	xxmFF848H	xxmFF849H	xxmFF84AH	xxmFF84BH	xxmFF84CH	xxmFF84DH	xxmFF84EH	xxmFF84FH
03	xxmFF868H	xxmFF869H	xxmFF86AH	xxmFF86BH	xxmFF86CH	xxmFF86DH	xxmFF86EH	xxmFF86FH
04	xxmFF888H	xxmFF889H	xxmFF88AH	xxmFF88BH	xxmFF88CH	xxmFF88DH	xxmFF88EH	xxmFF88FH
05	xxmFF8A8H	xxmFF8A9H	xxmFF8AAH	xxmFF8ABH	xxmFF8ACH	xxmFF8ADH	xxmFF8AEH	xxmFF8AFH
06	xxmFF8C8H	xxmFF8C9H	xxmFF8CAH	xxmFF8CBH	xxmFF8CCH	xxmFF8CDH	xxmFF8CEH	xxmFF8CFH
07	xxmFF8E8H	xxmFF8E9H	xxmFF8EAH	xxmFF8EBH	xxmFF8ECH	xxmFF8EDH	xxmFF8EEH	xxmFF8EFH
08	xxmFF908H	xxmFF909H	xxmFF90AH	xxmFF90BH	xxmFF90CH	xxmFF90DH	xxmFF90EH	xxmFF90FH
09	xxmFF928H	xxmFF929H	xxmFF92AH	xxmFF92BH	xxmFF92CH	xxmFF92DH	xxmFF92EH	xxmFF92FH
10	xxmFF948H	xxmFF949H	xxmFF94AH	xxmFF94BH	xxmFF94CH	xxmFF94DH	xxmFF94EH	xxmFF94FH
11	xxmFF968H	xxmFF969H	xxmFF96AH	xxmFF96BH	xxmFF96CH	xxmFF96DH	xxmFF96EH	xxmFF96FH
12	xxmFF988H	xxmFF989H	xxmFF98AH	xxmFF98BH	xxmFF98CH	xxmFF98DH	xxmFF98EH	xxmFF98FH
13	xxmFF9A8H	xxmFF9A9H	xxmFF9AAH	xxmFF9ABH	xxmFF9ACH	xxmFF9ADH	xxmFF9AEH	xxmFF9AFH
14	xxmFF9C8H	xxmFF9C9H	xxmFF9CAH	xxmFF9CBH	xxmFF9CCH	xxmFF9CDH	xxmFF9CEH	xxmFF9CFH
15	xxmFF9E8H	xxmFF9E9H	xxmFF9EAH	xxmFF9EBH	xxmFF9ECH	xxmFF9EDH	xxmFF9EEH	xxmFF9EFH
16	xxmFFA08H	xxmFFA09H	xxmFFA0AH	xxmFFA0BH	xxmFFA0CH	xxmFFA0DH	xxmFFA0EH	xxmFFA0FH
17	xxmFFA28H	xxmFFA29H	xxmFFA2AH	xxmFFA2BH	xxmFFA2CH	xxmFFA2DH	xxmFFA2EH	xxmFFA2FH
18	xxmFFA48H	xxmFFA49H	xxmFFA4AH	xxmFFA4BH	xxmFFA4CH	xxmFFA4DH	xxmFFA4EH	xxmFFA4FH
19	xxmFFA68H	xxmFFA69H	xxmFFA6AH	xxmFFA6BH	xxmFFA6CH	xxmFFA6DH	xxmFFA6EH	xxmFFA6FH
20	xxmFFA88H	xxmFFA89H	xxmFFA8AH	xxmFFA8BH	xxmFFA8CH	xxmFFA8DH	xxmFFA8EH	xxmFFA8FH
21	xxmFFAA8H	xxmFFAA9H	xxmFFAAAH	xxmFFAABH	xxmFFAACH	xxmFFAADH	xxmFFAAEH	xxmFFAAFH
22	xxmFFAC8H	xxmFFAC9H	xxmFFACAH	xxmFFACBH	xxmFFACCH	xxmFFACDH	xxmFFACEH	xxmFFACFH
23	xxmFFAE8H	xxmFFAE9H	xxmFFAEAH	xxmFFAEBH	xxmFFAECH	xxmFFAEDH	xxmFFAEEH	xxmFFAEFH
24	xxmFFB08H	xxmFFB09H	xxmFFB0AH	xxmFFB0BH	xxmFFB0CH	xxmFFB0DH	xxmFFB0EH	xxmFFB0FH
25	xxmFFB28H	xxmFFB29H	xxmFFB2AH	xxmFFB2BH	xxmFFB2CH	xxmFFB2DH	xxmFFB2EH	xxmFFB2FH
26	xxmFFB48H	xxmFFB49H	xxmFFB4AH	xxmFFB4BH	xxmFFB4CH	xxmFFB4DH	xxmFFB4EH	xxmFFB4FH
27	xxmFFB68H	xxmFFB69H	xxmFFB6AH	xxmFFB6BH	xxmFFB6CH	xxmFFB6DH	xxmFFB6EH	xxmFFB6FH
28	xxmFFB88H	xxmFFB89H	xxmFFB8AH	xxmFFB8BH	xxmFFB8CH	xxmFFB8DH	xxmFFB8EH	xxmFFB8FH
29	xxmFFBA8H	xxmFFBA9H	xxmFFBAAH	xxmFFBABH	xxmFFBACH	xxmFFBADH	xxmFFBAEH	xxmFFBAFH
30	xxmFFBC8H	xxmFFBC9H	xxmFFBCAH	xxmFFBCBH	xxmFFBCCH	xxmFFBCDH	xxmFFBCEH	xxmFFBCFH
31	xxmFFBE8H	xxmFFBE9H	xxmFFBEAH	xxmFFBEBH	xxmFFBECH	xxmFFBEDH	xxmFFBEEH	xxmFFBEFH

18.4.5 CAN message ID registers L00 to L31 and H00 to H31 (M_IDL00 to M_IDL31 and M_IDH00 to M_IDH31)

The M_IDLn and M_IDHn registers are areas used to set identifiers (n = 00 to 31).

These registers can be read/written in 16-bit units.

When in standard format mode, any data can be stored in the following areas.

★

ID17 to ID10: First byte of receive data^{Note} is stored.

ID9 to ID2: Second byte of receive data^{Note} is stored.

ID1, ID0: Third byte (higher two bits) of receive data^{Note} is stored.

Note See 18.4.4 CAN message data registers n0 to n7 (M_DATAn0 to M_DATAn7).

After reset: Undefined		R/W		Address: See Table 18-7												
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M_IDHn (n = 00 to 31)	IDE	0	0	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID
				28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M_IDLn (n = 00 to 31)	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID	ID
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Specification of format setting mode															
	IDE															
	0	Standard format mode (ID28 to ID18: 11 bits)														
	1	Extended format mode (ID28 to ID0: 29 bits)														

Table 18-7. Addresses of M_IDLn and M_IDHn (n = 00 to 31)

Register Name	Address (m = 3, 7, B)	Register Name	Address (m = 3, 7, B)
M_IDL00	xxmFF810H	M_IDL16	xxmFFA10H
M_IDH00	xxmFF812H	M_IDH16	xxmFFA12H
M_IDL01	xxmFF830H	M_IDL17	xxmFFA30H
M_IDH01	xxmFF832H	M_IDH17	xxmFFA32H
M_IDL02	xxmFF850H	M_IDL18	xxmFFA50H
M_IDH02	xxmFF852H	M_IDH18	xxmFFA52H
M_IDL03	xxmFF870H	M_IDL19	xxmFFA70H
M_IDH03	xxmFF872H	M_IDH19	xxmFFA72H
M_IDL04	xxmFF890H	M_IDL20	xxmFFA90H
M_IDH04	xxmFF892H	M_IDH20	xxmFFA92H
M_IDL05	xxmFF8B0H	M_IDL21	xxmFFAB0H
M_IDH05	xxmFF8B2H	M_IDH21	xxmFFAB2H
M_IDL06	xxmFF8D0H	M_IDL22	xxmFFAD0H
M_IDH06	xxmFF8D2H	M_IDH22	xxmFFAD2H
M_IDL07	xxmFF8F0H	M_IDL23	xxmFFAF0H
M_IDH07	xxmFF8F2H	M_IDH23	xxmFFAF2H
M_IDL08	xxmFF910H	M_IDL24	xxmFFB10H
M_IDH08	xxmFF912H	M_IDH24	xxmFFB12H
M_IDL09	xxmFF930H	M_IDL25	xxmFFB30H
M_IDH09	xxmFF932H	M_IDH25	xxmFFB32H
M_IDL10	xxmFF950H	M_IDL26	xxmFFB50H
M_IDH10	xxmFF952H	M_IDH26	xxmFFB52H
M_IDL11	xxmFF970H	M_IDL27	xxmFFB70H
M_IDH11	xxmFF972H	M_IDH27	xxmFFB72H
M_IDL12	xxmFF990H	M_IDL28	xxmFFB90H
M_IDH12	xxmFF992H	M_IDH28	xxmFFB92H
M_IDL13	xxmFF9B0H	M_IDL29	xxmFFBB0H
M_IDH13	xxmFF9B2H	M_IDH29	xxmFFBB2H
M_IDL14	xxmFF9D0H	M_IDL30	xxmFFBD0H
M_IDH14	xxmFF9D2H	M_IDH30	xxmFFBD2H
M_IDL15	xxmFF9F0H	M_IDL31	xxmFFBF0H
M_IDH15	xxmFF9F2H	M_IDH31	xxmFFBF2H

18.4.6 CAN message configuration registers 00 to 31 (M_CONF00 to M_CONF31)

The M_CONF_n register is used to specify the message buffer type and mask setting (n = 00 to 31). These registers can be read/written in 8-bit units.

After reset: Undefined	R/W			Address: See Table 18-8				
	7	6	5	4	3	2	1	0
M_CONF _n (n = 00 to 31)	0	0	MT2	MT1	MT0	MA2	MA1	MA0

MT2	MT1	MT0	Specification of message type and mask setting
0	0	0	Transmit message
0	0	1	Receive message (no mask setting)
0	1	0	Receive message (mask 0 is set)
0	1	1	Receive message (mask 1 is set)
1	0	0	Receive message (mask 2 is set)
1	0	1	Receive message (mask 3 is set)
1	1	0	Setting prohibited
1	1	1	Receive message (used in diagnostic processing mode)

- When bits MT2 to MT0 have been set as “111”, processing can be performed only when the FCAN has been set to diagnostic processing mode. In such cases, all messages received are stored regardless of the following conditions.
 - Storage to other message buffer
 - Identifier type (standard frame or extended frame)
 - Data frame or remote frame

MA2	MA1	MA0	Message buffer's address specification
0	0	0	Message buffer is not used
0	0	1	Used as message buffer of CAN module 1
0	1	0	Used as message buffer of CAN module 2 ^{Note}
Other than above			Setting prohibited

- When bits MA2, MA1, and MA0 have been set to “000”, message buffer area is used for application RAM or for event processing as a temporary buffer.

Note μ PD703079Y and 70F3079Y only

Table 18-8. Addresses of M_CONF_n (n = 00 to 31)

Register Name	Address (m = 3, 7, B)	Register Name	Address (m = 3, 7, B)
M_CONF00	xxmFF814H	M_CONF16	xxmFFA14H
M_CONF01	xxmFF834H	M_CONF17	xxmFFA34H
M_CONF02	xxmFF854H	M_CONF18	xxmFFA54H
M_CONF03	xxmFF874H	M_CONF19	xxmFFA74H
M_CONF04	xxmFF894H	M_CONF20	xxmFFA94H
M_CONF05	xxmFF8B4H	M_CONF21	xxmFFAB4H
M_CONF06	xxmFF8D4H	M_CONF22	xxmFFAD4H
M_CONF07	xxmFF8F4H	M_CONF23	xxmFFAF4H
M_CONF08	xxmFF914H	M_CONF24	xxmFFB14H
M_CONF09	xxmFF934H	M_CONF25	xxmFFB34H
M_CONF10	xxmFF954H	M_CONF26	xxmFFB54H
M_CONF11	xxmFF974H	M_CONF27	xxmFFB74H
M_CONF12	xxmFF994H	M_CONF28	xxmFFB94H
M_CONF13	xxmFF9B4H	M_CONF29	xxmFFBB4H
M_CONF14	xxmFF9D4H	M_CONF30	xxmFFBD4H
M_CONF15	xxmFF9F4H	M_CONF31	xxmFFBF4H

18.4.7 CAN message status registers 00 to 31 (M_STAT00 to M_STAT31)

The M_STATn register indicates the transmit/receive status information of each message buffer (n = 00 to 31). These registers are read-only, in 8-bit units.

- Cautions**
1. **Writing directly to M_STATn register cannot be performed. Writing must be performed using CAN status set/clear register n (SC_STATn).**
 2. **Messages are transmitted only when the M_STATn register’s TRQ and RDY bits have been set (1).**

After reset: Undefined	R	Address: See Table 18-9						
	7	6	5	4	3	2	1	0
M_STATn (n = 00 to 31)	0	0	0	0	0	DN	TRQ	RDY ^{Note}

DN	Message update flag
0	No message was received after DN bit was cleared
1	At least one message was received after DN bit was cleared
<ul style="list-style-type: none"> When the DN bit has been set (1) by the transmit message buffer, it indicates that the message buffer has received a remote frame. When this message is sent, the DN bit is automatically cleared (0). When a frame is again received in the message buffer for which the DN bit has been set (1), an overwrite condition occurs and the M_CTRLn register's MOVR bit is set (1). 	

TRQ	Transmit request flag
0	Message transmission prohibited
1	Message transmission enabled

RDY	Transmit message ready flag
0	Transmit message is not ready
1	Transmit message is ready
<ul style="list-style-type: none"> A receive operation is performed only for a message buffer in which the RDY bit is set to 1 during reception. A transmit operation is performed only for a message buffer in which the RDY bit is set to 1 and the TRQ bit is set to 1 during transmission. 	

★ **Note** The FCAN controller incorporated in the V850/SF1 can perform reception even if the RDY bit is not set. However, in products other than the V850/SF1, the RDY bit must be set for reception. In order to maintain software compatibility, be sure to set the RDY bit even for the FCAN controller of the V850/SF1 prior to reception.

Table 18-9. Addresses of M_STATn (n = 00 to 31)

Register Name	Address (m = 3, 7, B)	Register Name	Address (m = 3, 7, B)
M_STAT00	xxmFF815H	M_STAT16	xxmFFA15H
M_STAT01	xxmFF835H	M_STAT17	xxmFFA35H
M_STAT02	xxmFF855H	M_STAT18	xxmFFA55H
M_STAT03	xxmFF875H	M_STAT19	xxmFFA75H
M_STAT04	xxmFF895H	M_STAT20	xxmFFA95H
M_STAT05	xxmFF8B5H	M_STAT21	xxmFFAB5H
M_STAT06	xxmFF8D5H	M_STAT22	xxmFFAD5H
M_STAT07	xxmFF8F5H	M_STAT23	xxmFFAF5H
M_STAT08	xxmFF915H	M_STAT24	xxmFFB15H
M_STAT09	xxmFF935H	M_STAT25	xxmFFB35H
M_STAT10	xxmFF955H	M_STAT26	xxmFFB55H
M_STAT11	xxmFF975H	M_STAT27	xxmFFB75H
M_STAT12	xxmFF995H	M_STAT28	xxmFFB95H
M_STAT13	xxmFF9B5H	M_STAT29	xxmFFBB5H
M_STAT14	xxmFF9D5H	M_STAT30	xxmFFBD5H
M_STAT15	xxmFF9F5H	M_STAT31	xxmFFBF5H

18.4.8 CAN status set/clear registers 00 to 31 (SC_STAT00 to SC_STAT31)

The SC_STATn register is used to set/clear the transmit/receive status information (n = 00 to 31).

These registers are write-only, in 16-bit units.

After reset: 0000H W Address: See **Table 18-10**

	15	14	13	12	11	10	9	8
SC_STATn	0	0	0	0	0	set DN	set TRQ	set RDY

(n = 00 to 31)

	7	6	5	4	3	2	1	0
	0	0	0	0	0	clear DN	clear TRQ	clear RDY

set DN	clear DN	Message update flag setting
0	1	Clear (Clear DN bit)
1	0	Set (Set DN bit)
Other than above		No change in DN bit value

set TRQ	clear TRQ	Transmit request flag setting
0	1	Clear (Clear TRQ bit)
1	0	Set (Set TRQ bit)
Other than above		No change in TRQ bit value

set RDY	clear RDY	Message ready flag setting
0	1	Clear (Clear RDY bit)
1	0	Set (Set RDY bit)
Other than above		No change in RDY bit value

Remark DN: Bit 2 of CAN message status register n (M_STATn)
 TRQ: Bit 1 of CAN message status register n (M_STATn)
 RDY: Bit 0 of CAN message status register n (M_STATn)

Table 18-10. Addresses of SC_STATn (n = 00 to 31)

Register name	Address (m = 3, 7, B)	Register name	Address (m = 3, 7, B)
SC_STAT00	xxmFF816H	SC_STAT16	xxmFFA16H
SC_STAT01	xxmFF836H	SC_STAT17	xxmFFA36H
SC_STAT02	xxmFF856H	SC_STAT18	xxmFFA56H
SC_STAT03	xxmFF876H	SC_STAT19	xxmFFA76H
SC_STAT04	xxmFF896H	SC_STAT20	xxmFFA96H
SC_STAT05	xxmFF8B6H	SC_STAT21	xxmFFAB6H
SC_STAT06	xxmFF8D6H	SC_STAT22	xxmFFAD6H
SC_STAT07	xxmFF8F6H	SC_STAT23	xxmFFAF6H
SC_STAT08	xxmFF916H	SC_STAT24	xxmFFB16H
SC_STAT09	xxmFF936H	SC_STAT25	xxmFFB36H
SC_STAT10	xxmFF956H	SC_STAT26	xxmFFB56H
SC_STAT11	xxmFF976H	SC_STAT27	xxmFFB76H
SC_STAT12	xxmFF996H	SC_STAT28	xxmFFB96H
SC_STAT13	xxmFF9B6H	SC_STAT29	xxmFFBB6H
SC_STAT14	xxmFF9D6H	SC_STAT30	xxmFFBD6H
SC_STAT15	xxmFF9F6H	SC_STAT31	xxmFFBF6H

18.4.9 CAN interrupt pending register (CCINTP)

The CCINTP register is used to confirm the pending status of various interrupts.

This register is read-only in 16-bit units.

After reset: 0000H R Address: xxmFFC00H (m = 3, 7, B)

	15	14	13	12	11	10	9	8
CCINTP	0	INTMAC	0	0	0	0	0	0
	7	6	5	4	3	2	1	0
	0	0	CAN2ERR	CAN2REC	CAN2TRX	CAN1ERR	CAN1REC	CAN1TRX

INTMAC	Pending status of MAC error ^{Note 1} interrupts (GINT3 to GINT1)
0	Not pending
1	Pending

CAN2ERR ^{Note 2}	Pending status of CAN2 access error interrupt (C2INT6 to C2INT2)
0	Not pending
1	Pending

CAN2REC ^{Note 2}	Pending status of CAN2 receive completion interrupt (C2INT1)
0	Not pending
1	Pending

CAN2TRX ^{Note 2}	Pending status of CAN2 transmit completion interrupt (C2INT0)
0	Not pending
1	Pending

CAN1ERR	Pending status of CAN1 access error interrupt (C1INT6 to C1INT2)
0	Not pending
1	Pending

CAN1REC	Pending status of CAN1 receive completion interrupt (C1INT1)
0	Not pending
1	Pending

CAN1TRX	Pending status of CAN1 transmit completion interrupt (C1INT0)
0	Not pending
1	Pending

- Notes 1.** MAC (Message Access Control) errors are errors that are set only when an interrupt source has occurred for the CAN global interrupt pending register (CGINTP).
- 2.** μ PD703079Y and 70F3079Y only

Remark GINT3 to GINT1: Bits 3 to 1 of the CAN global interrupt pending register (CGINTP)
 CnINT6 to CnINT0 (n = 1, 2): Bits 6 to 0 of the CANn interrupt pending register (CnINTP)

18.4.10 CAN global interrupt pending register (CGINTP)

The CGINTP register is used to confirm the pending status of MAC access error interrupts. This register can be read/written in 8-bit units.

Cautions 1. When “1” is written to a bit in the CGINTP register, that bit is cleared (0). When “0” is written to it, the bit’s value does not change.

2. An interrupt occurs when the corresponding interrupt request is enabled and when no interrupt pending bit has been set (1) for a new interrupt.

The correct or incorrect timing of setting the interrupt pending bit (1) is controlled by an interrupt service routine. The earlier that the interrupt service routine clears the interrupt pending bit (0), the more quickly the interrupt occurs without losing any new interrupts of the same type.

The interrupt pending bit can be set (1) only when the interrupt enable bit has been set (1). However, the interrupt pending bit is not automatically cleared (0) just because the interrupt enable bit has been cleared (0).

Use software processing to clear the interrupt pending bit (0).

Remark For details of invalid write access error interrupts and unavailable memory address access error interrupts, see **18.15.2 Interrupts that occur for global CAN interface**.

After reset: 00H	R/W	Address: xxmFFC02H (m = 3, 7, B)							
		7	6	5	4	3	2	1	0
CGINTP		0	0	0	0	GINT3	GINT2	GINT1	0
GINT3	Pending status of wake-up interrupt (from CAN sleep mode with stopped clock supply to FCAN)								
0	Not pending								
1	Pending								
GINT2	Pending status of invalid write access error interrupt								
0	Not pending								
1	Pending								
GINT1	Pending status of unavailable memory address access error interrupt								
0	Not pending								
1	Pending								

18.4.11 CANn interrupt pending register (CnINTP)

The CnINTP register is used to confirm the pending status of interrupts issued to the CAN.

This register can be read/written in 8-bit units.

The CAN2 interrupt pending register (C2INTP) is valid only in models μ PD703079Y and 70F3079Y.

Cautions 1. When “1” is written to a bit in the CnINTP register, that bit is cleared (0). When “0” is written to it, the bit’s value does not change.

2. An interrupt occurs when the corresponding interrupt request is enabled and when no interrupt pending bit has been set (1) for a new interrupt.

The correct or incorrect timing of setting the interrupt pending bit (1) is controlled by an interrupt service routine. The earlier that the interrupt service routine clears the interrupt pending bit (0), the more quickly the interrupt occurs without losing any new interrupts of the same type.

The interrupt pending bit can be set (1) only when the interrupt ready bit has been set (1). However, the interrupt pending bit is not automatically cleared (0) just because the interrupt enable bit has been cleared (0). Use software processing to clear the interrupt pending bit (0).

Remark n = 1, 2

After reset: 00H

R/W

Addresses: C1INTP: xxFFC04H (m = 3, 7, B)

C2INTP: xxFFC06H (m = 3, 7, B)

	7	6	5	4	3	2	1	0
CnINTP	0	CnINT6	CnINT5	CnINT4	CnINT3	CnINT2	CnINT1	CnINT0
(n = 1, 2)								

CnINT6	Pending status of CAN error interrupt
0	Not pending
1	Pending

CnINT5	Pending status of CAN bus error interrupt
0	Not pending
1	Pending

CnINT4	Pending status of wake-up interrupt (from CAN sleep mode)
0	Not pending
1	Pending

CnINT3	Pending status of CAN receive error passive status interrupt
0	Not pending
1	Pending

CnINT2	Pending status of CAN transmit error passive or bus off status interrupt
0	Not pending
1	Pending

CnINT1	Pending status of CAN receive completion interrupt
0	Not pending
1	Pending

CnINT0	Pending status of CAN transmit completion interrupt
0	Not pending
1	Pending

18.4.12 CAN stop register (CSTOP)

The CSTOP register controls clock supply to the entire CAN system.

This register can be read/written in 16-bit units.

Cautions 1. Be sure to set the CSTOP bit (1) if the FCAN function will not be used.

2. When the CSTOP bit is set (1), access to FCAN registers other than the CSTOP register is prohibited. Access to FCAN (other than the CSTOP register) is possible only when the CSTOP bit is not set (1).

3. When a change occurs on the CAN bus via a CSTOP setting while the clock supply to the CPU or peripheral functions is stopped, the CPU can be woken up.

★

After reset: 0000H		R/W		Address: xxmFFC0CH (m = 3, 7, B)												
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSTOP	CSTP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	CSTP	Controls clock supply to FCAN														
	0	FCAN in operation (supplies clock to FCAN blocks)														
	1	FCAN is stopped (access to FCAN blocks is not possible)														

18.4.13 CAN global status register (CGST)

The CGST register indicates global status information.

This register can be read in 16- or 8-bit units and written in 16-bit units.

- Cautions**
- Both bitwise writing and direct writing to the CGST register are prohibited. Attempts to write directly to this register may result in operation faults, so be sure to follow the sequence described in 18.5 Cautions Regarding Bit Set/Clear Function.
 - When writing to the CGST register, set or clear bits according to the register configuration shown in part (b) Write of the following figure.

(1/2)

After reset: 0100H R/W Address: xxmFFC10H (m = 3, 7, B)

(a) Read

	15	14	13	12	11	10	9	8
CGST	0	0	0	0	0	0	0	1
	7	6	5	4	3	2	1	0
	MERR	0	0	0	EFSD	TSM	0	GOM

(b) Write

	15	14	13	12	11	10	9	8
CGST	0	0	0	0	set EFSD	set TSM	0	set GOM
	7	6	5	4	3	2	1	0
	clear MERR	0	0	0	clear EFSD	clear TSM	0	clear GOM

(a) Read

MERR	MAC error status flag
0	Error does not occur after the MERR bit has been cleared
1	Error occurs at least once after MERR bit has been cleared
<ul style="list-style-type: none"> MAC errors occur when an access to an invalid RAM address is attempted. 	

EFSD	Shutdown request
0	Shutdown prohibited
1	Shutdown enabled
<ul style="list-style-type: none"> Be sure to set the EFSD bit (1) before clearing the GOM bit (0) (must be accessed twice). The EFSD bit will be cleared (0) automatically when the CGST register is accessed again. 	

TSM	Operation status of time stamp counter ^{Note}
0	Time stamp counter is stopped
1	Time stamp counter is operating

Note See 18.4.16 CAN time stamp count register (CGTSC).

★

(a) Read

GOM	Status of global operation mode
0	Access to CAN module register ^{Note} is prohibited
1	Access to CAN module register ^{Note} is enabled

- The GOM bit controls the method the memory is accessed by the MAC.
 - When GOM bit = 0
 - Access to CAN module register disabled (if accessed, MAC error interrupt occurs)
 - Read/write access to temporary buffer enabled
 - Access to message buffer area enabled
 - When GOM bit = 1
 - Access to CAN module register enabled
 - Only read access to temporary buffer enabled (if write access is attempted, MAC error interrupt occurs)
 - Access to message buffer area enabled
- The GOM bit is cleared (0) only when all the CAN modules are in the initial status (the ISTAT bit of the CnCTR register is 1). Even if the GOM bit is cleared when there is a CAN module not in the initial status, the GOM bit remains set (1).

(b) Write

set EFSD	clear EFSD	EFSD bit enable
0	1	EFSD bit cleared
1	0	EFSD bit set
Other than above		No change in value of EFSD bit

set TSM	clear TSM	TSM bit enable
0	1	TSM bit cleared
1	0	TSM bit set
Other than above		No change in value of TSM bit

set GOM	clear GOM	GOM bit enable
0	1	GOM bit cleared
1	0	GOM bit set
Other than above		No change in value of GOM bit

clear MERR	MERR bit enable
0	No change in value of MERR bit
1	MERR bit cleared

Note Register with a name starting with “Cn” (n = 1, 2)

18.4.14 CAN global interrupt enable register (CGIE)

The CGIE register is used to issue interrupt requests for global interrupts. This register can be read in 16- or 8-bit units and written in 16-bit units.

- Cautions**
- Both bitwise writing and direct writing to the CGIE register are prohibited. Attempts to write directly to this register may result in operation faults, so be sure to follow the sequence described in 18.5 Cautions Regarding Bit Set/Clear Function.
 - When writing to the CGIE register, set or clear bits according to the register configuration shown in part (b) Write of the following figure.

After reset: 0A00H		R/W		Address: xxmFFC12H (m = 3, 7, B)				
(a) Read	15	14	13	12	11	10	9	8
CGIE	0	0	0	0	1	0	1	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	G_IE2	G_IE1	0
(b) Write	15	14	13	12	11	10	9	8
CGIE	0	0	0	0	0	set G_IE2	set G_IE1	0
	7	6	5	4	3	2	1	0
	0	0	0	0	0	clear G_IE2	clear G_IE1	0
(a) Read								
	G_IE2	Interrupt enable status for invalid write access (to temporary buffer, etc.)						
	0	Interrupt disabled						
	1	Interrupt enabled						
	G_IE1	Interrupt enable status for memory access to reserved address						
	0	Interrupt disabled						
	1	Interrupt enabled						
(b) Write								
	set G_IEn	clear G_IEn	Setting of G_IEn bit					
	0	1	Clear G_IEn bit					
	1	0	Set G_IEn bit					
	Other than above		No change					
Remark	n = 1, 2							

18.4.15 CAN main clock select register (CGCS)

The CGCS register is used to select the main clock.

This register can be read/written in 16-bit units.

After reset: 7F05H R/W Address: xxFFC14H (m = 3, 7, B)

	15	14	13	12	11	10	9	8
CGCS	CGTS7	CGTS6	CGTS5	CGTS4	CGTS3	CGTS2	CGTS1	CGTS0
	7	6	5	4	3	2	1	0
	GTCS1	GTCS0	0	0	MCP3	MCP2	MCP1	MCP0

n	CGTS 7	CGTS 6	CGTS 5	CGTS 4	CGTS 3	CGTS 2	CGTS 1	CGTS 0	System timer prescaler selection $f_{GTS} = f_{GTS1} / (n + 1)$
0	0	0	0	0	0	0	0	0	$f_{GTS} = f_{GTS1}/1$
1	0	0	0	0	0	0	0	1	$f_{GTS} = f_{GTS1}/2$
:									$f_{GTS} = f_{GTS1}/(n + 1)$
127	0	1	1	1	1	1	1	1	$f_{GTS} = f_{GTS1}/128$ (after reset)
:									$f_{GTS} = f_{GTS1}/(n + 1)$
254	1	1	1	1	1	1	1	0	$f_{GTS} = f_{GTS1}/255$
255	1	1	1	1	1	1	1	1	$f_{GTS} = f_{GTS1}/256$

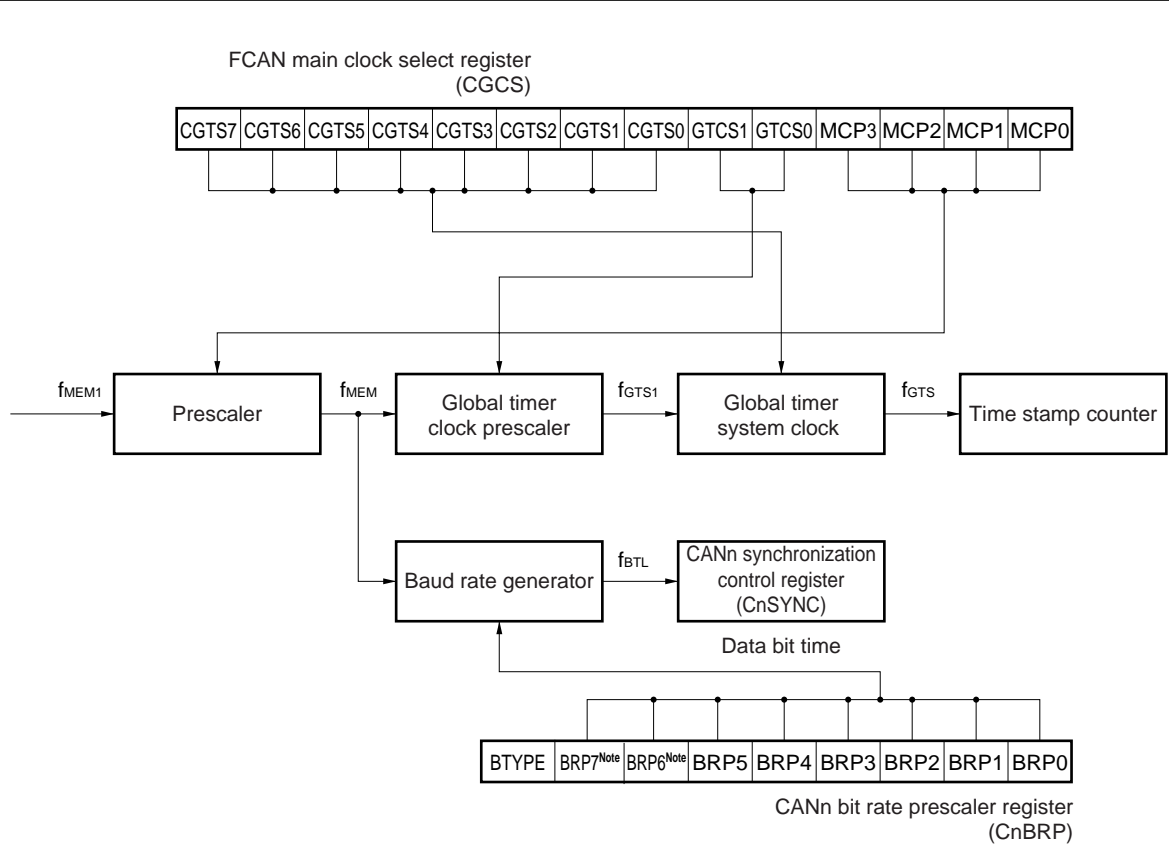
The global time system clock (f_{GTS}) is the source clock for the time stamp counter^{Note} that is used for the time stamp function.

GTCS1	GTCS0	Global timer clock selection (f_{GTS1})
0	0	$f_{MEM}/2$
0	1	$f_{MEM}/4$
1	0	$f_{MEM}/8$
1	1	$f_{MEM}/16$

n	MCP3	MCP2	MCP1	MCP0	Selection of clock to memory access controller (f_{MEM})
0	0	0	0	0	f_{MEM1}
1	0	0	0	1	$f_{MEM1}/2$
2	0	0	1	0	$f_{MEM1}/3$
:					
14	1	1	1	0	$f_{MEM1}/15$
15	1	1	1	1	$f_{MEM1}/16$

Note See 18.4.16 CAN time stamp count register (CGTSC).

Figure 18-2. FCAN Clocks



Note When the TLM bit of the CANn bit rate prescaler register (CnBRP) is 1.

Caution When using a 1 Mbps transfer rate for the CPU, input f_{MEM1} as a 16 MHz clock signal. If input at another frequency, subsequent operation is not guaranteed.

Remark $f_{MEM1} = f_{XX} =$ Main clock frequency

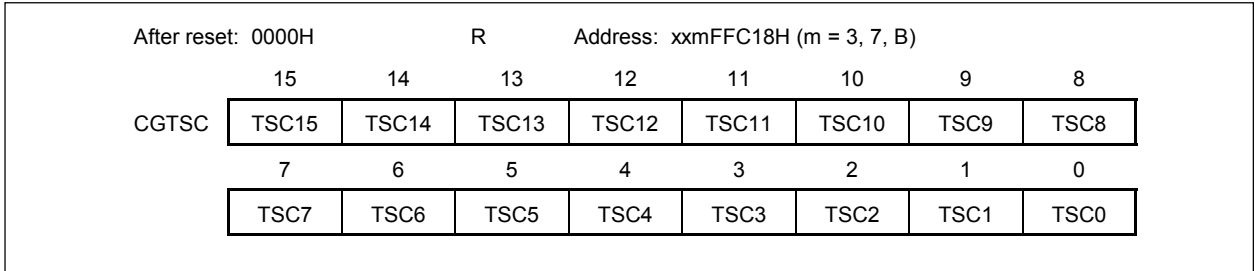
18.4.16 CAN time stamp count register (CGTSC)

The CGTSC register indicates the contents of the time stamp counter.

This register can be read at any time.

This register can be written to only when clearing bits. The clear function writes 0 to all bits in the CGTSC register.

This register is read-only, in 16-bit units.



18.4.17 CAN message search start/result register (CGMSS/CGMSR)

The CGMSS/CGMSR register indicates the message search start/result status. Messages in the message buffer that match the specified search criteria can be searched quickly.

These registers can be read/written in 16-bit units.

(1/2)

After reset: 0000H		R/W		Address: xmFFC1AH (m = 3, 7, B)				
(a) Read	15	14	13	12	11	10	9	8
CGMSR	0	0	0	0	0	0	MM	AM
	7	6	5	4	3	2	1	0
	0	0	0	MFND4	MFND3	MFND2	MFND1	MFND0
(b) Write	15	14	13	12	11	10	9	8
CGMSS	CIDE	0	CTRQ	CMSK	CDN	0	SMNO1	SMNO0
	7	6	5	4	3	2	1	0
	0	0	0	STRT4	STRT3	STRT2	STRT1	STRT0
(a) Read								
MM	Confirmation of multiple hits from message search							
0	No messages or only one message meets the search criteria							
1	Several messages meet the search criteria							
If several message buffers that meet the search criteria are detected, the MM bit is set.								
AM	Confirmation of hits from message search							
0	No messages meet the search criteria							
1	At least one message meets the search criteria							
MFND4 to MFND0	Searched message number							
<p>This indicates the number (0 to 31) of the searched message.</p> <ul style="list-style-type: none"> When multiple message buffer numbers match as a result of a search (MM = 1), the return value of bits MFND4 to MFND0 is the lowest message buffer number. When no message buffer numbers match (AM = 0), the return value of bits MFND4 to MFND0 is "message buffer number -1". 								

★

(b) Write

CIDE	Message identifier (ID) format flag check
0	Message identifier format flag not checked
1	Message with standard format identifier checked

CTRQ	Transmit request and message ready flag check
0	Transmit request and message ready flags not checked
1	Transmit request and message ready flags checked

CMSK	Masked message check
0	Masked messages not checked
1	Only masked messages checked

CDN	Status check of M_STATn register's DN flag (n = 00 to 31)
0	Status of M_STATn register's DN flag not checked
1	Status of M_STATn register's DN flag checked

SMNO1	SMNO0	Search module setting
0	0	No search module setting
0	1	CAN module 1 is set as the searched target
1	0	CAN module 2 is set as the searched target
1	1	Setting prohibited

STRTn	Message search start position (n = 0 to 4)
0 to 31	Message search start position (message number)
<ul style="list-style-type: none"> Search starts from the message number defined by bits STRT4 to STRT0. Search continues until it reaches the message buffer having the highest number among the usable message buffers. If the search results include several message buffer numbers among the matching messages, the message buffer with the lowest message buffer number is selected. To fetch the next message buffer number without changing the search criteria, "(MFND4 to MFND0) + 1" must be set as the values of bits STRT4 to STRT0. 	

18.4.18 CANn address mask a registers L and H (CnMASKLa and CnMASKHa)

The CnMASKLa and CnMASKHa registers are used to extend the number of receivable messages by masking part of the message's identifier (ID) and then ignoring the masked parts.

These registers can be read/written in 16-bit units.

The C2MASKLa and C2MASKHa registers are valid only in models μ PD703079Y and 70F3079Y.

- ★ **Cautions** 1. When the receive message buffer is linked to the CnMASKLa and CnMASKHa registers, regardless of whether the ID in the receive message buffer is a standard ID (11 bits) or an extended ID (29 bits), set all the 32-bit values of the CnMASKLa and CnMASKHa registers (a = 0 to 3, n = 1, 2).
- 2. When the CnMASKLa and CnMASKHa registers are linked to the message buffer for standard IDs, the lower 18 bits of the data field in the data frame are also automatically compared. Therefore, if it is not necessary to compare the lower 18 bits (masking), set (1) the CMID17 to CMID0 bits (a = 0 to 3, n = 1, 2).

After reset: Undefined		R/W		Address: See Table 18-11					
★	CnMASKHa	15	14	13	12	11	10	9	8
		CMIDE	0	0	CMID28	CMID27	CMID26	CMID25	CMID24
(a = 0 to 3, n = 1, 2)		7	6	5	4	3	2	1	0
		CMID23	CMID22	CMID21	CMID20	CMID19	CMID18	CMID17	CMID16
★	CnMASKLa	15	14	13	12	11	10	9	8
		CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8
(a = 0 to 3, n = 1, 2)		7	6	5	4	3	2	1	0
		CMID7	CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0
		CMIDE	Mask setting for identifier (ID) format						
		0	ID format (standard or extended) checked						
		1	ID format (standard or extended) not checked						
		CMID0 to CMID28	Mask setting for identifier (ID) bits						
		0	ID bit in message buffer linked to bits CMID28 to CMID0 compared with received ID bit.						
		1	ID bit in message buffer linked to bits CMID28 to CMID0 not compared with received ID bit (i.e., masked).						

Table 18-11. Addresses of CnMASKLa and CnMASKHa (a = 0 to 3, n = 1, 2)

Register Name	Address (m = 3, 7, B)	Register Name	Address (m = 3, 7, B)
C1MASKL0	xxmFFC40H	C2MASKL0	xxmFFC80H
C1MASKH0	xxmFFC42H	C2MASKH0	xxmFFC82H
C1MASKL1	xxmFFC44H	C2MASKL1	xxmFFC84H
C1MASKH1	xxmFFC46H	C2MASKH1	xxmFFC86H
C1MASKL2	xxmFFC48H	C2MASKL2	xxmFFC88H
C1MASKH2	xxmFFC4AH	C2MASKH2	xxmFFC8AH
C1MASKL3	xxmFFC4CH	C2MASKL3	xxmFFC8CH
C1MASKH3	xxmFFC4EH	C2MASKH3	xxmFFC8EH

18.4.19 CANn control register (CnCTRL)

The CnCTRL register is used to control the operation of the CAN module.

This register can be read/written in 16-bit units.

The C2CTRL register is valid only in models μ PD703079Y and 70F3079Y.

- Cautions**
- Both bitwise writing and direct writing to the CnCTRL register are prohibited. Attempts to write directly to this register may result in operation faults, so be sure to follow the sequence described in 18.5 Cautions Regarding Bit Set/Clear Function.
 - When writing to the CnCTRL register, set or clear bits according to the register configuration shown in part (b) Write of the following figure.
 - When canceling CAN stop mode, CAN sleep mode must be canceled at the same time.

(1/4)

After reset: 0101H		R/W	Addresses: C1CTRL: xxFFC50H (m = 3, 7, B)					
			C2CTRL: xxFFC90H (m = 3, 7, B)					
(a) Read	15	14	13	12	11	10	9	8
CnCTRL	TECS1	TECS0	RECS1	RECS0	BOFF	TSTAT	RSTAT	ISTAT
(n = 1, 2)	7	6	5	4	3	2	1	0
	0	DLEVR	DLEVT	OVM	TMR	STOP	SLEEP	INIT
(b) Write	15	14	13	12	11	10	9	8
CnCTRL	0	set DLEVR	set DLEVT	set OVM	set TMR	set STOP	set SLEEP	set INIT
(n = 1, 2)	7	6	5	4	3	2	1	0
	0	clear DLEVR	clear DLEVT	clear OVM	clear TMR	clear STOP	clear SLEEP	clear INIT
(a) Read	TECS1	TECS0	Status of transmit error counter					
	0	0	Transmit error counter value < 96					
	0	1	Transmit error counter value = 96 to 127 (warning level)					
	1	0	Not used					
	1	1	Transmit error counter value \geq 128 (error passive)					
	RECS1	RECS0	Status of receive error counter					
	0	0	Receive error counter value < 96					
	0	1	Receive error counter value = 96 to 127 (warning level)					
	1	0	Not used					
	1	1	Receive error counter value \geq 128 (error passive)					
	BOFF	Bus off flag						
	0	Transmit error counter value < 256 (not bus off status)						
	1	Transmit error counter value \geq 256 (bus off status)						

(a) Read

TSTAT	Transmit status flag
0	Transmit stop status
1	Transmit operating status

RSTAT	Receive status flag
0	Receive stop status
1	Receive operating status

ISTAT	Initialization status flag
0	Normal operating status
1	FCAN is stopped and initialized

- The ISTAT bit is set (1) when the CAN protocol layer acknowledges the setting of the INIT bit and STOP bit. The ISTAT bit is automatically cleared (0) after the INIT bit and STOP bit are cleared (0).
- When the ISTAT bit has been set (1): “receive” is output via the CANTXn pin during initialization mode.
- The CnSYNC and CnBRP registers can be written only during initialization mode.
- In the initialization status, the error counter (see **18.4.22 CANn error count register (CnERC)**) is cleared (0) and the error status (bit TECS1, TECS0, RECS1, and RECS0) is reset.

DLEVR	Dominant level control bit for receive pin
0	A low level to a receive pin is acknowledged as dominant
1	A high level to a receive pin is acknowledged as dominant

DLEVT	Dominant level control bit for transmit pin
0	A low level is transmitted from transmit pin as dominant
1	A high level is transmitted from transmit pin as dominant

OVM	Overwrite mode control bit
0	New messages stored in message buffer in which DN bit of M_STATa register is set (a = 00 to 31)
1	New messages in message buffer in which DN bit is set (a = 00 to 31) discarded

TMR	Time stamp control bit for reception
0	When the SOF is detected on the CAN bus, the value of the time stamp counter is captured.
1	When the EOF is detected on the CAN bus (a valid message is confirmed), the value of the time stamp counter is captured.

(a) Read

STOP	CAN stop mode control bit
0	No CAN stop mode setting
1	CAN stop mode

- CAN stop mode can be selected only when the CAN module has been set to CAN sleep mode, i.e., when the SLEEP bit has been set (1). CAN stop mode can be canceled only by the CPU by clearing the STOP bit (0).

SLEEP	CAN sleep mode control bit
0	Normal operating mode
1	Switch to CAN sleep mode (change in CAN bus performs wake-up)

- CAN sleep mode is canceled under the following conditions.
 - When the CPU has cleared the SLEEP bit (0)
 - When the CAN bus changes (this occurs only when CAN stop mode has not been set)
- The WAKE bit^{Note} is set (1) only when CAN sleep mode is canceled by the change of the CAN bus.

INIT	Initialization request bit
0	Normal operating mode
1	Initialization mode

- Be sure to confirm that the CAN module has entered the initialization mode using the ISTAT bit (ISTAT bit = 1) after setting the INIT bit (1). When the ISTAT bit = 0, set the INIT bit again.
- If the INIT bit is set (1) when the CAN module is in the bus off status (BOFF bit = 1), the CAN module enters initialization mode (ISTAT bit = 1) after returning from the bus off status (BOFF bit = 0).

Note See 18.4.20 CANn definition register (CnDEF).

(b) Write

set DLEVR	clear DLEVR	DLEVR bit setting
0	1	DLEVR bit cleared
1	0	DLEVR bit set
Other than above		DLEVR bit not changed

set DLEVT	clear DLEVT	DLEVT bit setting
0	1	DLEVT bit cleared
1	0	DLEVT bit set
Other than above		DLEVT bit not changed

set OVM	clear OVM	OVM bit setting
0	1	OVM bit cleared
1	0	OVM bit set
Other than above		OVM bit not changed

set TMR	clear TMR	TMR bit setting
0	1	TMR bit cleared
1	0	TMR bit set
Other than above		TMR bit not changed

set STOP	clear STOP	STOP bit setting
0	1	STOP bit cleared
1	0	STOP bit set
Other than above		STOP bit not changed

set SLEEP	clear SLEEP	SLEEP bit setting
0	1	SLEEP bit cleared
1	0	SLEEP bit set
Other than above		SLEEP bit not changed

set INIT	clear INIT	INIT bit setting
0	1	INIT bit cleared
1	0	INIT bit set
Other than above		INIT bit not changed

(a) Read

MOM	Specification of CAN module's operating mode
0	Normal operating mode
1	Diagnostic processing mode

- When in diagnostic processing mode (MOM bit = 1), the CnBRP register can be accessed only when the CAN module has been set to initialization mode (i.e., when the CnCTRL register's ISTAT bit = INIT bit = 1).
When the CAN module is operating (i.e., when the CnCTRL register's ISTAT bit = 0) the CnBRP register cannot be used, and the CANn bus diagnostic information register (CnDINF) register can be used instead.

SSHT	Specification of single shot mode
0	Normal operating mode
1	Single shot mode

- During single shot mode, the CAN module can transmit a message only one time. The M_STATa (a = 00 to 31) register's TRQ bit is then cleared (0) regardless of whether or not there are any pending normal transmit operations.
Also, if a bus error has occurred due to a transmission, it is handled as an incomplete transmission.
- During single shot mode, even if the CAN lost in arbitration, it is handled as a completed message transmission.
When in this mode, the BERR bit is set (1) but the error counter value does not change since there are no CAN bus errors.
- During the time when the CAN module is active, the CPU switches between normal operating mode and single shot mode without causing any errors to occur on the CAN bus.

PBB	Specification of priority control for transmission
0	Identifier (ID) based priority control
1	Message number based priority control

- Ordinarily, priority for transmission is defined based on message IDs, but when the PBB bit has been set (1) priority becomes based instead on the position of messages, so that messages with lower message numbers have higher priority.

BERR	CAN bus error status
0	CAN bus error was not detected
1	CAN bus error was detected at least once after bit was cleared

VALID	Valid message detection status
0	Valid message was not detected
1	Valid message was detected at least once after bit was cleared

WAKE	CAN sleep mode cancellation status
0	Normal operation
1	Cancel CAN sleep mode

(a) Read

OVR	Overrun error status
0	Normal operation
1	Overwrite occurred during RAM access
<ul style="list-style-type: none"> When an overrun error has occurred, the OVR bit is set (1) and an error interrupt occurs at the same time. The source of the overrun error may be that the RAM access clock is slower than the selected CAN baud rate. 	

(b) Write

set DGM	clear DGM	DGM bit setting
0	1	DGM bit cleared
1	0	DGM bit set
Other than above		DGM bit not changed

set MOM	clear MOM	MOM bit setting
0	1	MOM bit cleared
1	0	MOM bit set
Other than above		MOM bit not changed

set SSHT	clear SSHT	SSHT bit setting
0	1	SSHT bit cleared
1	0	SSHT bit set
Other than above		SSHT bit not changed

set PBB	clear PBB	PBB bit setting
0	1	PBB bit cleared
1	0	PBB bit set
Other than above		PBB bit not changed

clear BERR	BERR bit setting
1	BERR bit cleared
0	BERR bit not changed

clear VALID	VALID bit setting
1	VALID bit cleared
0	VALID bit not changed

clear WAKE	WAKE bit setting
1	WAKE bit cleared
0	WAKE bit not changed

clear OVR	OVR bit setting
1	OVR bit cleared
0	OVR bit not changed

18.4.21 CAn information register (CnLAST)

The CnLAST register indicates the CAn module's error information and the number of the message buffer received last.

This register is read-only, in 16-bit units.

The C2LAST register is valid only in models μ PD703079Y and 70F3079Y.

After reset: 00FFH	R	Addresses: C1LAST: xxFFC54H (m = 3, 7, B) C2LAST: xxFFC94H (m = 3, 7, B)						
	15	14	13	12	11	10	9	8
CnLAST	0	0	0	0	LERR3	LERR2	LERR1	LERR0
(n = 1, 2)	7	6	5	4	3	2	1	0
	LREC7	LREC6	LREC5	LREC4	LREC3	LREC2	LREC1	LREC0
	LERR3	LERR2	LERR1	LERR0	Last error information			
	0	0	0	0	Error not detected			
	0	0	0	1	Bit error			
	0	0	1	0	Stuff error			
	0	0	1	1	CRC error			
	0	1	0	0	Form error			
	0	1	0	1	ACK error			
	0	1	1	0	Arbitration lost (only during single shot mode) (CnDEF: SSHT = 1)			
	0	1	1	1	CAN overrun error			
	1	0	0	0	Wake-up from CAN bus			
	Other than above				Setting prohibited			
	<ul style="list-style-type: none"> Since bits LERR3 to LERR0 cannot be cleared, the current status is retained until the next error occurs. 							
	LREC7 to LREC0		Number of last receive message					
	0 to 31		Message number of message last received					
	32 to 255		Not used					

18.4.22 CANn error count register (CnERC)

The CnERC register indicates the count values of the transmission/reception error counters.

This register is read-only in 16-bit units.

The C2ERC register is valid only in models μ PD703079Y and 70F3079Y.

After reset: 0000H		R		Addresses: C1ERC: xxFFC56H (m = 3, 7, B)				C2ERC: xxFFC96H (m = 3, 7, B)	
	15	14	13	12	11	10	9	8	
CnERC	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	
(n = 1, 2)	7	6	5	4	3	2	1	0	
	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	
REC7 to REC0		Reception error counter							
0 to 255		Number of reception error counts							
		<ul style="list-style-type: none"> • This reflects the current status of the reception error counter. • The count value is defined by the CAN protocol. 							
TEC7 to TEC0		Transmission error counter							
0 to 255		Number of transmission error counts							
		<ul style="list-style-type: none"> • This reflects the current status of the transmission error counter. • The number of counts is defined by the CAN protocol. 							

18.4.23 CANn interrupt enable register (CnIE)

The CnIE register is used to enable/disable the CAN module's interrupts.
 This register can be read in 16- or 8-bit units and written in 16-bit units.
 The C2IE register is valid only in models μ PD703079Y and 70F3079Y.

- Cautions 1. Both bitwise writing and direct writing to the CnIE register are prohibited. Attempts to write directly to this register may result in operation faults, so be sure to follow the sequence described in 18.5 Cautions Regarding Bit Set/Clear Function.**
- 2. When writing to the CnIE register, set or clear bits according to the register configuration shown in part (b) Write of the following figure.**

(1/2)

After reset: 0900H		R/W		Addresses: C1IE: xxmFFC58H (m = 3, 7, B)				
				C2IE: xxmFFC98H (m = 3, 7, B)				
(a) Read	15	14	13	12	11	10	9	8
CnIE	0	0	0	0	1	0	0	1
(n = 1, 2)	7	6	5	4	3	2	1	0
	0	E_INT6	E_INT5	E_INT4	E_INT3	E_INT2	E_INT1	E_INT0
(b) Write	15	14	13	12	11	10	9	8
CnIE	0	set E_INT6	set E_INT5	set E_INT4	set E_INT3	set E_INT2	set E_INT1	set E_INT0
(n = 1, 2)	7	6	5	4	3	2	1	0
	0	clear E_INT6	clear E_INT5	clear E_INT4	clear E_INT3	clear E_INT2	clear E_INT1	clear E_INT0
(a) Read								
E_INT6	CAN module error interrupt enable flag							
0	Interrupt disabled							
1	Interrupt enabled							
E_INT5	CAN bus error interrupt enable flag							
0	Interrupt disabled							
1	Interrupt enabled							
E_INT4	Wake up from CAN sleep mode interrupt enable flag							
0	Interrupt disabled							
1	Interrupt enabled							
E_INT3	Receive error passive interrupt enable flag							
0	Interrupt disabled							
1	Interrupt enabled							
E_INT2	Receive error passive or bus off interrupt enable flag							
0	Interrupt disabled							
1	Interrupt enabled							

(a) Read

E_INT1	Receive completion interrupt enable flag
0	Interrupt disabled
1	Interrupt enabled

E_INT0	Transmit completion interrupt enable flag
0	Interrupt disabled
1	Interrupt enabled

(b) Write

set E_INT6	clear E_INT6	E_INT6 bit setting
0	1	E_INT6 interrupt cleared
1	0	E_INT6 interrupt set
Other than above		E_INT6 interrupt not changed

set E_INT5	clear E_INT5	E_INT5 bit setting
0	1	E_INT5 interrupt cleared
1	0	E_INT5 interrupt set
Other than above		E_INT5 interrupt not changed

set E_INT4	clear E_INT4	E_INT4 bit setting
0	1	E_INT4 interrupt cleared
1	0	E_INT4 interrupt set
Other than above		E_INT4 interrupt not changed

set E_INT3	clear E_INT3	E_INT3 bit setting
0	1	E_INT3 interrupt cleared
1	0	E_INT3 interrupt set
Other than above		E_INT3 interrupt not changed

set E_INT2	clear E_INT2	E_INT2 bit setting
0	1	E_INT2 interrupt cleared
1	0	E_INT2 interrupt set
Other than above		E_INT2 interrupt not changed

set E_INT1	clear E_INT1	E_INT1 bit setting
0	1	E_INT1 interrupt cleared
1	0	E_INT1 interrupt set
Other than above		E_INT1 interrupt not changed

set E_INT0	clear E_INT0	E_INT0 bit setting
0	1	E_INT0 interrupt cleared
1	0	E_INT0 interrupt set
Other than above		E_INT0 interrupt not changed

18.4.24 CANn bus active register (CnBA)

The CnBA register indicates frame information output via the CAN bus.

This register is read-only, in 16-bit units.

The C2BA register is valid only in models μ PD703079Y and 70F3079Y.

After reset: 00FFH	R	Addresses: C1BA: xxFFC5AH (m = 3, 7, B) C2BA: xxFFC9AH (m = 3, 7, B)						
	15	14	13	12	11	10	9	8
CnBA	0	0	0	CACT4	CACT3	CACT2	CACT1	CACT0
(n = 1, 2)	7	6	5	4	3	2	1	0
	TMNO7	TMNO6	TMNO5	TMNO4	TMNO3	TMNO2	TMNO1	TMNO0
	CACT4	CACT3	CACT2	CACT1	CACT0	CAN module status		
	0	0	0	0	0	Reset state		
	0	0	0	0	1	Bus idle wait		
	0	0	0	1	0	Bus idle state		
	0	0	0	1	1	Start of frame		
	0	0	1	0	0	Standard identifier area		
	0	0	1	0	1	Data length code area		
	0	0	1	1	0	Data field area		
	0	0	1	1	1	CRC field area		
	0	1	0	0	0	CRC delimiter		
	0	1	0	0	1	ACK slot		
	0	1	0	1	0	ACK delimiter		
	0	1	0	1	1	End of frame area		
	0	1	1	0	0	Intermission state		
	0	1	1	0	1	Suspend transmission		
	0	1	1	1	0	Error frame		
	0	1	1	1	1	Error delimiter wait		
	1	0	0	0	0	Error delimiter		
	1	0	0	0	1	Bus off error		
	1	0	0	1	0	Extended identifier area		
	Other than above					Setting prohibited		
	TMNO7 to TMNO0	Transmission message counter						
	0 to 31	Message number of message awaiting transmission or being transmitted						
	32 to 254	Not used						
	255	No messages awaiting transmission or being transmitted						

18.4.25 CANn bit rate prescaler register (CnBRP)

The CnBRP register is used to set the transmission baud rate for the CAN module.

Use the CnBRP register to select the CAN protocol layer main system clock (f_{BTL}). The baud rate is determined by the value set to the CnSYNC register.

While in normal operating mode (CnDEF register's MOM bit = 0), writing to the CnBRP register is enabled only when the initialization mode has been set (CnCTRL register's INIT bit = 1).

This register can be read/written in 16-bit units.

The C2BRP register is valid only in models μ PD703079Y and 70F3079Y.

Caution While in diagnostic processing mode (CnDEF register's MOM bit = 1), the CnBRP register can be accessed only when the initialization mode has been set.

After reset: 0000H R/W Addresses: C1BRP: xxFFC5CH (m = 3, 7, B)
C2BRP: xxFFC9CH (m = 3, 7, B)

★

(a) When TLM = 0

	15	14	13	12	11	10	9	8
CnBRP	TLM	0	0	0	0	0	0	0
(n = 1, 2)	7	6	5	4	3	2	1	0
	0	BTYPE	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0

★

(b) When TLM = 1

	15	14	13	12	11	10	9	8
CnBRP	TLM	0	0	0	0	0	0	BTYPE
(n = 1, 2)	7	6	5	4	3	2	1	0
	BRP7	BRP6	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0

(a) When TLM = 0

TLM	Transfer layer mode specification
0	6-bit prescaler mode

BTYPE	CAN bus type specification
0	Low speed (≤ 125 Kbps)
1	High speed (> 125 Kbps)

a	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	CAN protocol layer basic system clock (f_{BTL})
0	0	0	0	0	0	0	$f_{MEM}/2$
1	0	0	0	0	0	1	$f_{MEM}/4$
2	0	0	0	0	1	0	$f_{MEM}/6$
3	0	0	0	0	1	1	$f_{MEM}/8$
:							$f_{MEM}/\{(a + 1) \times 2\}$
60	1	1	1	1	0	0	$f_{MEM}/122$
61	1	1	1	1	0	1	$f_{MEM}/124$
62	1	1	1	1	1	0	$f_{MEM}/126$
63	1	1	1	1	1	1	$f_{MEM}/128$

Remark $f_{BTL} = f_{MEM}/\{(a + 1) \times 2\}$: CAN protocol layer basic system clock
 $a = 0$ to 63 (set by bits BRP5 to BRP0)
 f_{MEM} = CAN module clock

(b) When TLM = 1

TLM	Transfer layer mode specification
1	8-bit prescaler mode

BTYPE	CAN bus type specification
0	Low speed (≤ 125 Kbps)
1	High speed (> 125 Kbps)

a	BRP7	BRP6	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	CAN protocol layer basic system clock (f_{BTL})
0	0	0	0	0	0	0	0	0	Setting prohibited
1	0	0	0	0	0	0	0	1	$f_{MEM}/2$
2	0	0	0	0	0	0	1	0	$f_{MEM}/3$
3	0	0	0	0	0	0	1	1	$f_{MEM}/4$
:									$f_{MEM}/(a + 1)$
252	1	1	1	1	1	1	0	0	$f_{MEM}/253$
253	1	1	1	1	1	1	0	1	$f_{MEM}/254$
254	1	1	1	1	1	1	1	0	$f_{MEM}/255$
255	1	1	1	1	1	1	1	1	$f_{MEM}/256$

Remark $f_{BTL} = f_{MEM}/(a + 1)$: CAN protocol layer basic system clock
 $a = 0$ to 255 (set by bits BRP7 to BRP0)
 f_{MEM} = CAN module clock

18.4.27 CANn synchronization control register (CnSYNC)

The CnSYNC register controls the data bit time for transmission speed.

This register can be read/written in 16-bit units.

The C2SYNC register is valid only in models μ PD703079Y and 70F3079Y.

- Cautions**
1. The CPU is able to read the CnSYNC register at any time.
 2. Writing to the CnSYNC register is enabled in initialization mode (when CnCTRL register's INIT bit = 1).

- ★
3. The limit values when setting the SPTa bit and DBTa bit are as follows (a = 0 to 4).
 - $5 \times \text{BTL} \leq \text{SPT (sampling point)} \leq 17 \times \text{BTL}$ [$4 \leq \text{SPT4 to SPT0 set values} \leq 16$]
 - $8 \times \text{BTL} \leq \text{DBT (data bit time)} \leq 25 \times \text{BTL}$ [$7 \leq \text{DBT4 to DBT0 set values} \leq 24$]
 - $\text{SJW (Synchronization jump width)} < \text{DBT} - \text{SPT}$
 - $2 \leq (\text{DBT} - \text{SPT}) \leq 8$

Remark BTL = $1/f_{\text{BTL}}$ (f_{BTL} : CAN protocol layer basic system clock)

(1/2)

After reset: 0218H		R/W	Addresses: C1SYNC: xxFFC5EH (m = 3, 7, B) C2SYNC: xxFFC9EH (m = 3, 7, B)					
	15	14	13	12	11	10	9	8
CnSYNC	0	0	0	SAMP	SJW1	SJW0	SPT4	SPT3
(n = 1, 2)	7	6	5	4	3	2	1	0
	SPT2	SPT1	SPT0	DBT4	DBT3	DBT2	DBT1	DBT0
SAMP	Bit sampling specification							
0	Sample data received at the sampling point once							
1	Sample received data three times and majority value used as sampled value							
SJW1	SJW0	Synchronization jump width ^{Note}						
0	0	BTL						
0	1	$\text{BTL} \times 2$						
1	0	$\text{BTL} \times 3$						
1	1	$\text{BTL} \times 4$						

Note As stipulated in CAN protocol specification Ver. 2.0, Part B.

Remark BTL = $1/f_{\text{BTL}}$ (f_{BTL} : CAN protocol layer basic system clock)

SPT4	SPT3	SPT2	SPT1	SPT0	Position of sampling point
0	0	1	0	0	BTL × 5
0	0	1	0	1	BTL × 6
0	0	1	1	0	BTL × 7
0	0	1	1	1	BTL × 8
0	1	0	0	0	BTL × 9
0	1	0	0	1	BTL × 10
0	1	0	1	0	BTL × 11
0	1	0	1	1	BTL × 12
0	1	1	0	0	BTL × 13
0	1	1	0	1	BTL × 14
0	1	1	1	0	BTL × 15
0	1	1	1	1	BTL × 16
1	0	0	0	0	BTL × 17
Other than above					Setting prohibited
Sampling point within bit timing is selected.					

DBT4	DBT3	DBT2	DBT1	DBT0	Data bit time
0	0	1	1	1	BTL × 8
0	1	0	0	0	BTL × 9
0	1	0	0	1	BTL × 10
0	1	0	1	0	BTL × 11
0	1	0	1	1	BTL × 12
0	1	1	0	0	BTL × 13
0	1	1	0	1	BTL × 14
0	1	1	1	0	BTL × 15
0	1	1	1	1	BTL × 16
1	0	0	0	0	BTL × 17
1	0	0	0	1	BTL × 18
1	0	0	1	0	BTL × 19
1	0	0	1	1	BTL × 20
1	0	1	0	0	BTL × 21
1	0	1	0	1	BTL × 22
1	0	1	1	0	BTL × 23
1	0	1	1	1	BTL × 24
1	1	0	0	0	BTL × 25
Other than above					Setting prohibited
1-bit data length is set for CAN bus					

Remark BTL = $1/f_{BTL}$ (f_{BTL} : CAN protocol layer basic system clock)

18.5 Cautions Regarding Bit Set/Clear Function

The FCAN control registers include registers whose bits can be set or cleared via the CPU and via the CAN interface. An operation error occurs if values are written directly to these registers, so do not directly write (via bit manipulation, read/modify/write, or direct writing of target values) values to them.

- CAN global status register (CGST)
- CAN global interrupt enable register (CGIE)
- CANn control register (CnCTRL)
- CANn definition register (CnDEF)
- CANn interrupt enable register (CnIE)

Remark n = 1, 2

All 16 bits in the above registers can be read via the usual method. Use the procedure described in Figure 18-3 to set or clear the lower 8 bits in these registers.

Setting or clearing of lower 8 bits in the above registers is performed in combination with the higher 8 bits (see **Figure 18-4**). Figure 18-3 shows how the values of set bits or clear bits relate to set/clear/no change operations in the corresponding register.

Figure 18-3. Example of Bit Setting/Clearing Operations

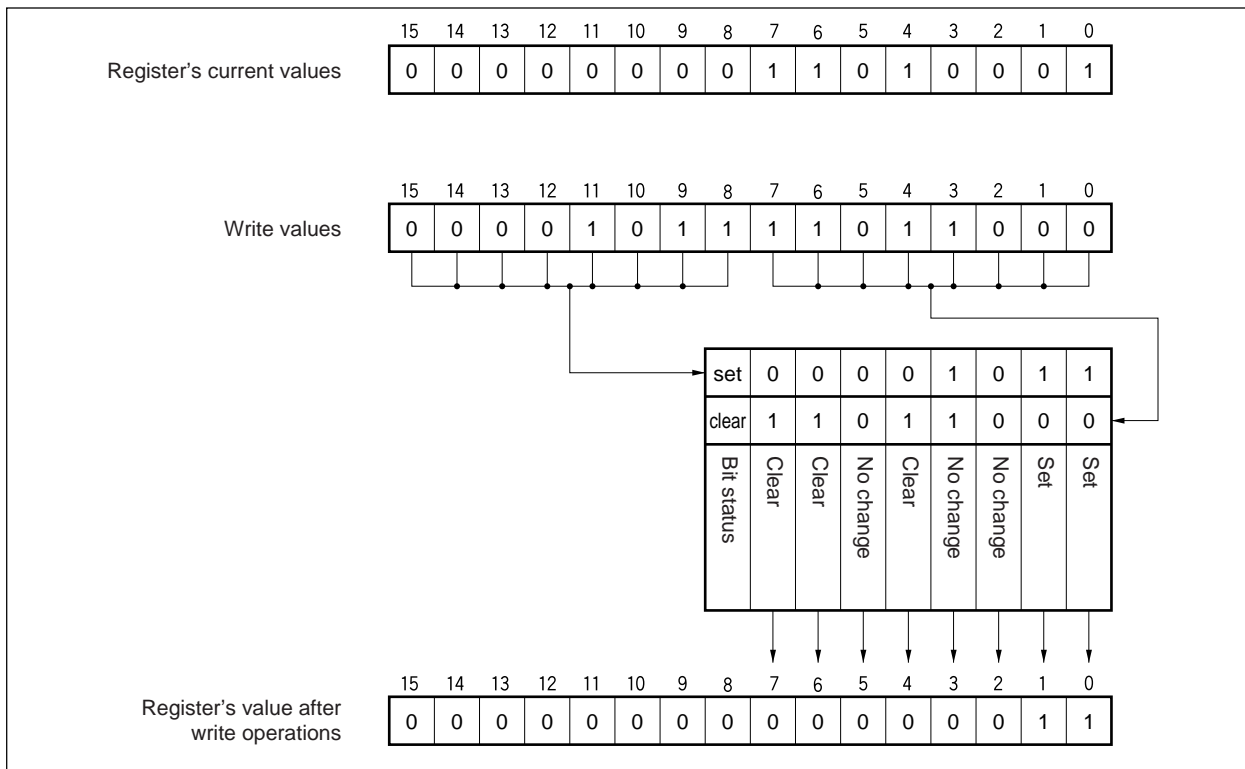


Figure 18-4. 16-Bit Data During Write Operation

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
set 7	set 6	set 5	set 4	set 3	set 2	set 1	set 0	clear 7	clear 6	clear 5	clear 4	clear 3	clear 2	clear 1	clear 0

set n	clear n	Status of clear n bit after bit set/clear operation
0	0	No change
0	1	0
1	0	1
1	1	No change

Remark n = 0 to 7

★ 18.6 Time Stamp Function

The FCAN controller supports a time stamp function. This function is needed to build a global time system.

The time stamp function is implemented using a 16-bit free-running time stamp counter.

Two types of time stamp function can be selected for message reception in the FCAN controller. Use bit 3 (TMR) of the CANx control register (CxCTRL) to set the desired time stamp function (x = 1, 2). When the TMR bit is 0, the time stamp counter value is captured after the SOF is detected on the CAN bus (see **Figure 18-5**) and when the TMR bit is 1, the time stamp counter value is captured after the EOF is detected on the CAN bus (a valid message is confirmed) (see **Figure 18-6**).

Figure 18-5. Time Stamp Function Setting for Message Reception (When CxCTRL Register's TMR Bit = 0)

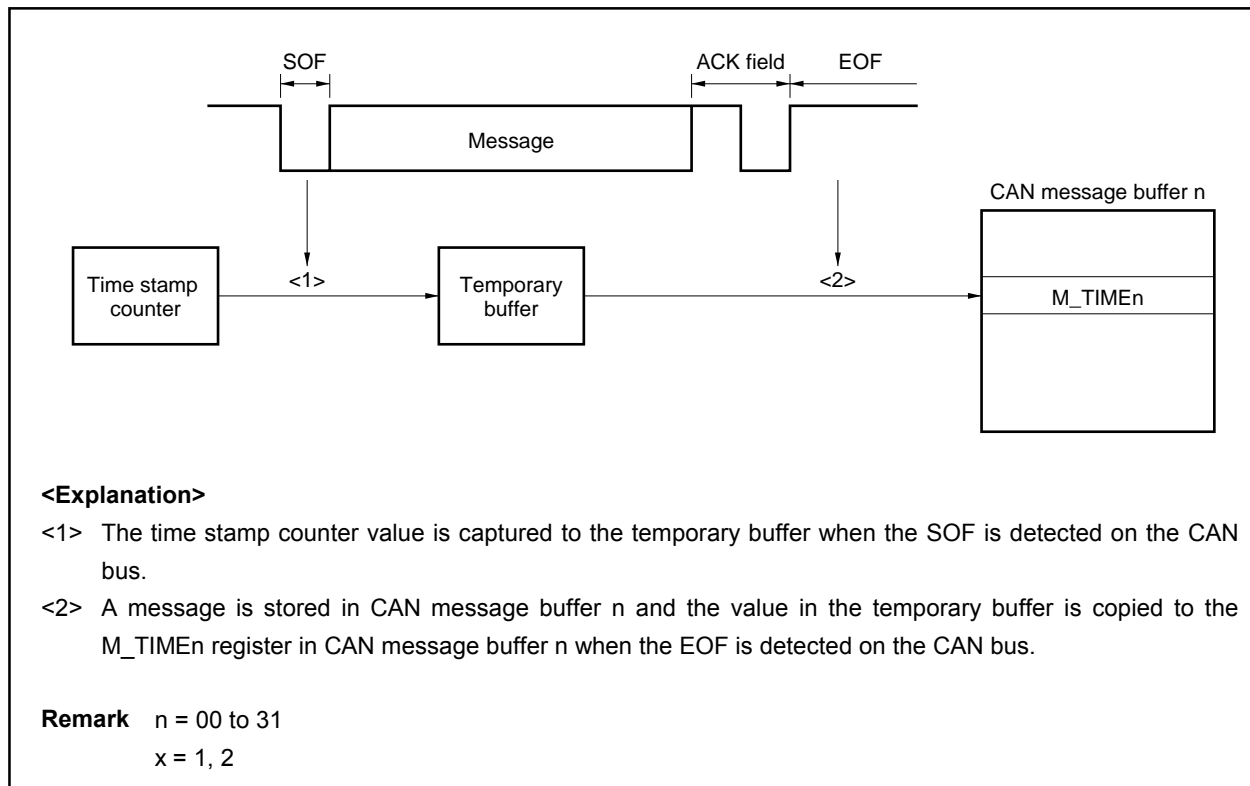
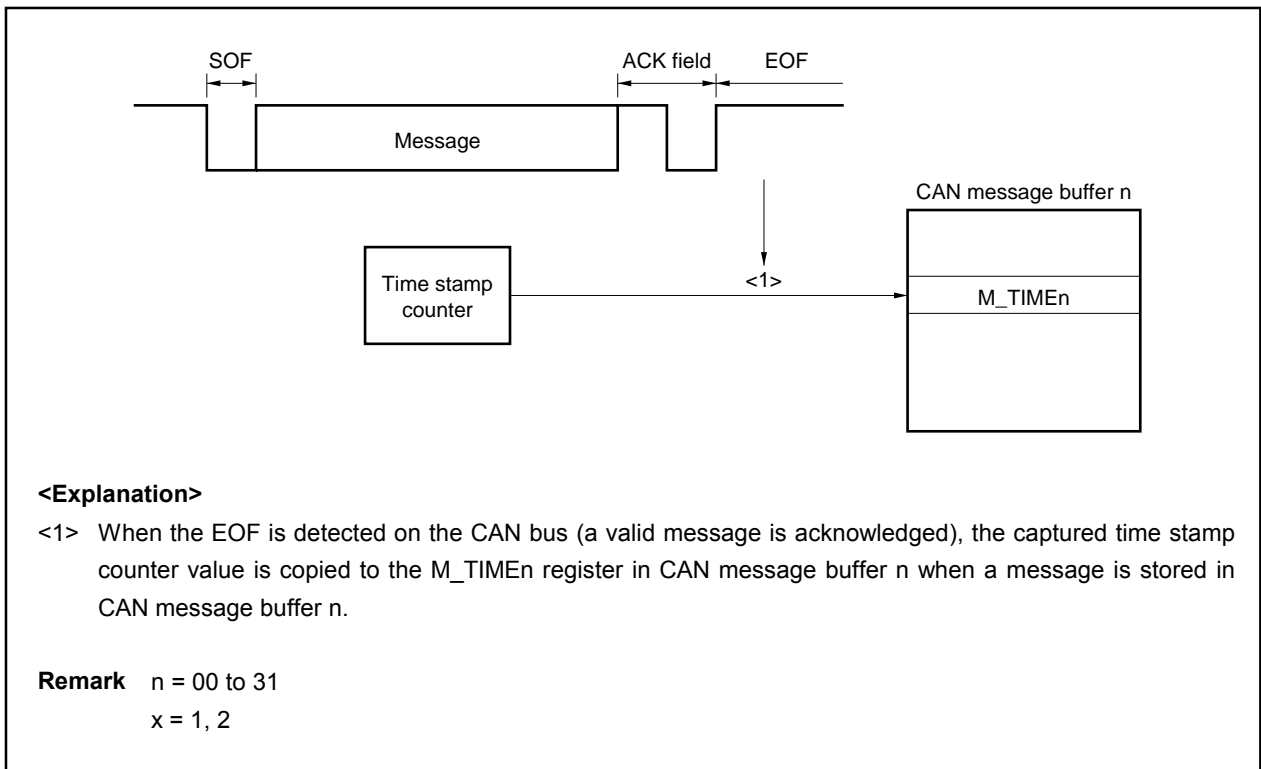


Figure 18-6. Time Stamp Function Setting for Message Reception (When CxCTRL Register's TMR Bit = 1)



In a global time system, the time value must be captured using the SOF.

In addition, the ability to capture the time stamp counter value when message is stored in CAN message buffer n is useful for evaluating the FCAN controller's performance.

The captured time stamp counter value is stored in CAN message buffer n, so CAN message buffer n has its own time stamp function (n = 00 to 31).

When the SOF is detected on the CAN bus while transmitting a message, there is an option to replace the last two bytes of the message with the captured time stamp counter value by setting bit 5 (ATS) of CAN message control register n (M_CTRLn). This function can be selected for CAN message buffer n on a buffer by buffer basis. Figure 18-7 shows the time stamp setting when the ATS bit = 1.

**Figure 18-7. Time Stamp Function Setting for Message Transmission
(When M_CTRL Register's ATS Bit = 1)**

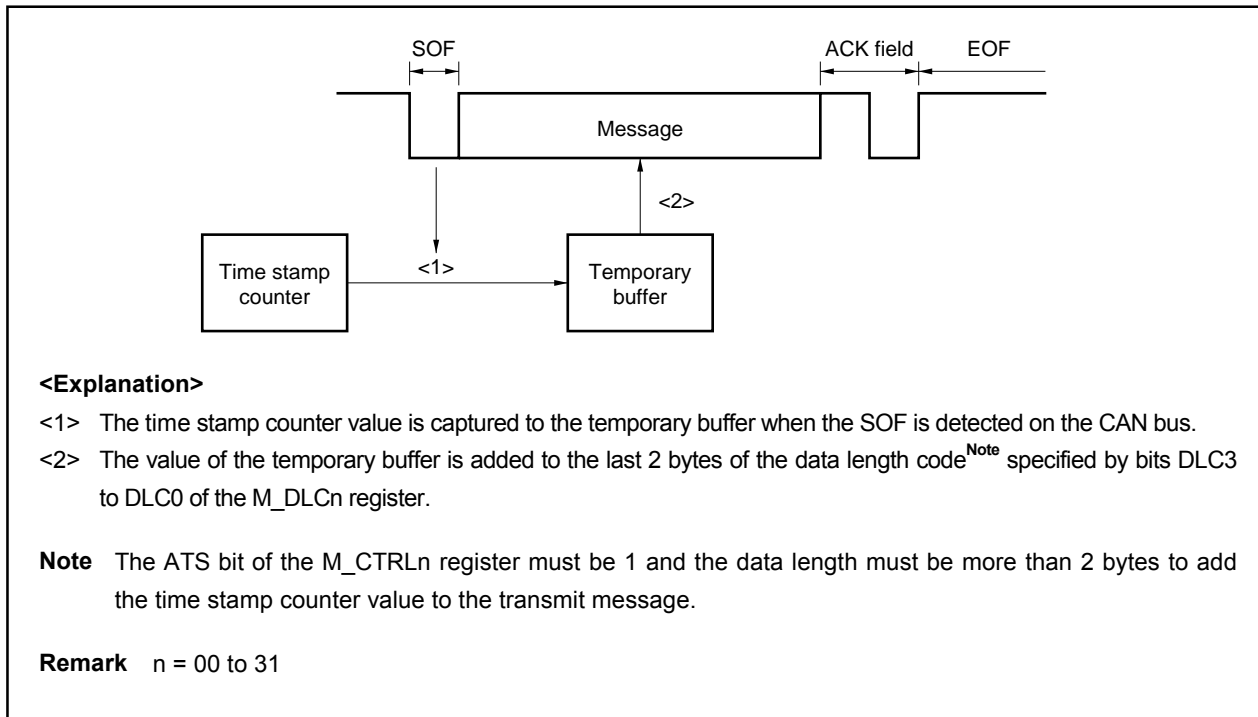


Table 18-12. Example When Adding Captured Time Stamp Counter Value to Last 2 Bytes of Transmit Message

Data Field DLC Bit Value ^{Note 1}	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8
1	M_DATAn0 register value	–	–	–	–	–	–	–
2	Note 2	Note 3	–	–	–	–	–	–
3	M_DATAn0 register value	Note 2	Note 3	–	–	–	–	–
4	M_DATAn0 register value	M_DATAn1 register value	Note 2	Note 3	–	–	–	–
5	M_DATAn0 register value	M_DATAn1 register value	M_DATAn2 register value	Note 2	Note 3	–	–	–
6	M_DATAn0 register value	M_DATAn1 register value	M_DATAn2 register value	M_DATAn3 register value	Note 2	Note 3	–	–
7	M_DATAn0 register value	M_DATAn1 register value	M_DATAn2 register value	M_DATAn3 register value	M_DATAn4 register value	Note 2	Note 3	–
8	M_DATAn0 register value	M_DATAn1 register value	M_DATAn2 register value	M_DATAn3 register value	M_DATAn4 register value	M_DATAn5 register value	Note 2	Note 3
9 to 15	M_DATAn0 register value	M_DATAn1 register value	M_DATAn2 register value	M_DATAn3 register value	M_DATAn4 register value	M_DATAn5 register value	Note 2	Note 3

- Notes**
1. See 18.4.1 CAN message data length registers 00 to 31 (M_DLC00 to M_DLC31).
 2. The lower 8 bits of the time stamp counter value when the SOF is detected on the CAN bus
 3. The higher 8 bits of the time stamp counter value when the SOF is detected on the CAN bus

Remark n = 00 to 31

18.7 Message Processing

A modular system is used for the FCAN controller. Consequently, messages can be placed at any location within the message area.

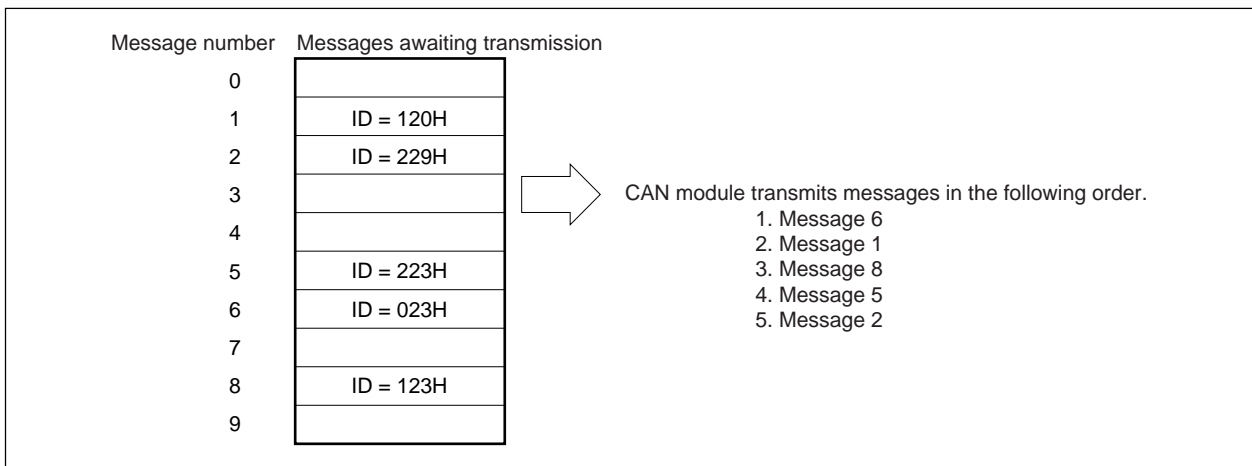
The messages can be linked to mask functions that are in turn linked to CAN modules.

The FCAN system is a multiplexed communication system. The priority of message transmission within this system is determined based on message identifiers (IDs).

To facilitate communication processing by application software when there are several messages awaiting transmission, the CAN module uses hardware to check the message IDs and automatically determine whether or not linked messages are prioritized.

This eliminates the need for software-based priority control.

Figure 18-8. Example of Message Processing



This auto priority control function can be canceled. When it is canceled, the priority among messages is determined based on the locations of the messages in memory (the message that has the lowest message number among the messages in the message buffer has the highest priority).

★ When several messages have been received in the CAN module's message buffers, priority control for storing received messages is as follows.

Priority rank 1: Remote frame reception in the transmit message buffer.

Priority rank 2: Non-masked message, for which M_STATn register's DN bit has not been set (1)

Priority rank 3: Non-masked message, for which M_STATn register's DN bit has been set (1)

Priority rank 4: Masked 0 message, for which M_STATn register's DN bit has not been set (1)

Priority rank 5: Masked 0 message, for which M_STATn register's DN bit has been set (1)

Priority rank 6: Masked 1 message, for which M_STATn register's DN bit has not been set (1)

Priority rank 7: Masked 1 message, for which M_STATn register's DN bit has been set (1)

Priority rank 8: Masked 2 message, for which M_STATn register's DN bit has not been set (1)

Priority rank 9: Masked 2 message, for which M_STATn register's DN bit has been set (1)

Priority rank 10: Masked 3 message, for which M_STATn register's DN bit has not been set (1)

Priority rank 11: Masked 3 message, for which M_STATn register's DN bit has been set (1)

Remark n = 00 to 31

If priority based on location in memory is the same for multiple messages, the message buffer having the lowest message number is selected.

18.8 Mask Function

A mask linkage function can be defined for each received message.

This means that there is no need to distinguish between local masks and global masks.

When the mask function is used, the received message's identifier is compared with the message buffer's identifier and the message can be stored in the defined message buffer regardless of whether the mask sets "0" or "1" as a result of the comparison.

When the mask function is operating, a bit whose value is defined as "1" by masking is not subject to the abovementioned comparison between the received message's identifier and the message buffer's identifier.

However, this comparison is performed for any bit whose value is defined as "0" by masking.

For example, let us assume that all messages that have a standard-format ID in which bit ID27 to ID25 = 0 and bits ID24 and ID22 = 1 are to be stored in message buffer 14 (which is linked by CAN module 1 or mask 1 as was explained in 18.4.6). The procedure for this example is shown below.

<1> Identifier bits to be stored in message buffer

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x

x = don't care

Messages with ID in which bits ID27 to ID25 = 0 and bits ID24 and ID22 = 1 are registered (initialized) in message buffer 14 (see 18.4.5).

★ <2> Identifier bits set to message buffer 14 (example)

(Using CAN message ID registers L14 and H14 (M_IDL14 and M_IDH14))

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
0	0	0	0	1	0	1	0	0	0	0
ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
0	0	0	0	0	0	0	0	0	0	0
ID6	ID5	ID4	ID3	ID2	ID1	ID0				
0	0	0	0	0	0	0				

Message buffer 14 is set as a standard-format identifier linked to mask 1 (see 18.4.6).

★ <3> **Mask setting for CAN module 1 (mask 1) (example)**

(Using CAN1 address mask 1 registers L and H (C1MASKL1 and C1MASKH1))

CMID28	CMID27	CMID26	CMID25	CMID24	CMID23	CMID22	CMID21	CMID20	CMID19	CMID18
1	0	0	0	0	1	0	1	1	1	1
CMID17	CMID16	CMID15	CMID14	CMID13	CMID12	CMID11	CMID10	CMID9	CMID8	CMID7
1	1	1	1	1	1	1	1	1	1	1
CMID6	CMID5	CMID4	CMID3	CMID2	CMID1	CMID0				
1	1	1	1	1	1	1				

1: Do not compare (mask)

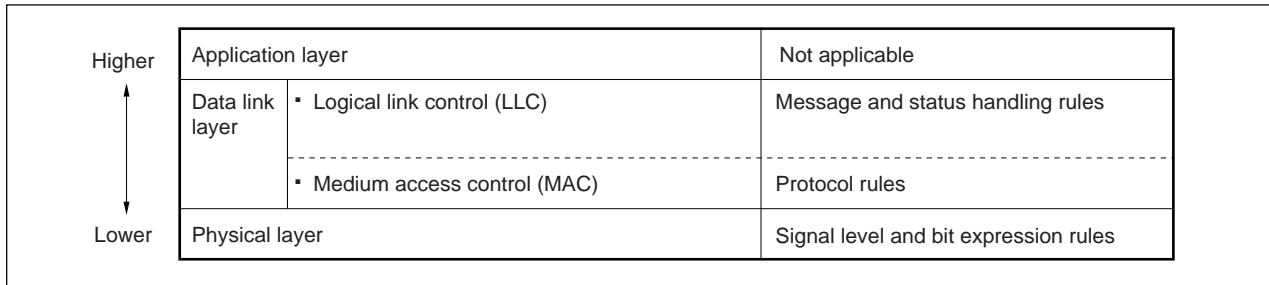
0: Compare

Values are written to mask 1 (see **18.4.18**), bits CMID27 to CMID24 and CMID22 = 0 and bits CMID28, CMID23, and CMID21 to CMID0 = 1.

18.9 Protocol

FCAN is a high-speed multiplex communication protocol designed to enable real-time communications in automotive applications. The CAN specification is generally divided into two layers (physical layer and data link layer). In turn, the data link layer includes logical link control and medium access control. The composition of these layers is illustrated in Figure 18-9 below.

Figure 18-9. Composition of Layers



18.9.1 Protocol mode function

(1) Standard format mode

In this mode 2032 different identifiers can be set.

The standard format mode uses 11-bit identifiers, which means that it can handle up to 2032 messages.

(2) Extended format mode

This mode is used to extend the number of identifiers that can be set.

- While the standard format mode uses 11-bit identifiers, the extended format mode uses 29-bit (11 bits + 18 bits) identifiers which increases the number of messages that can be handled to 2032×2^{18} messages.
- Extended format mode is set when “recessive (R): recessive in wired OR” is set for both the SRR and IDE bits in the arbitration field.
- When an extended format mode message and a standard format mode remote frame are transmitted at the same time, the node that transmitted the extended format mode message is set to receive mode.

18.9.2 Message formats

Four types of frames are used in CAN protocol messages. The output conditions for each type of frame are as follows.

- Data frame: Frame used for transmit data
- Remote frame: Frame used for transmit requests from receiving side
- Error frame: Frame that is output when an error has been detected
- Overload frame: Frame that is output when receiving side is not ready

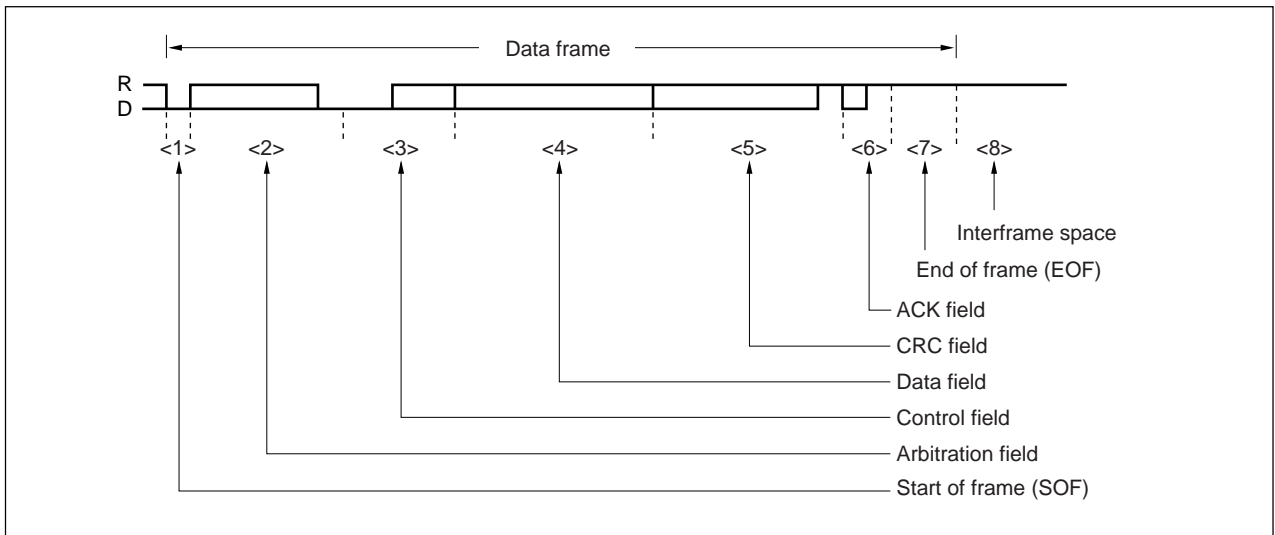
Remark Dominant (D): Dominant in wired OR
 Recessive (R): Recessive in wired OR
 In the figure shown below, (D) = 0 and (R) = 1.

(1) Data frame and remote frame

<1> Data frame

A data frame is the frame used for transmit data.
 This frame is composed of seven fields.

Figure 18-10. Data Frame

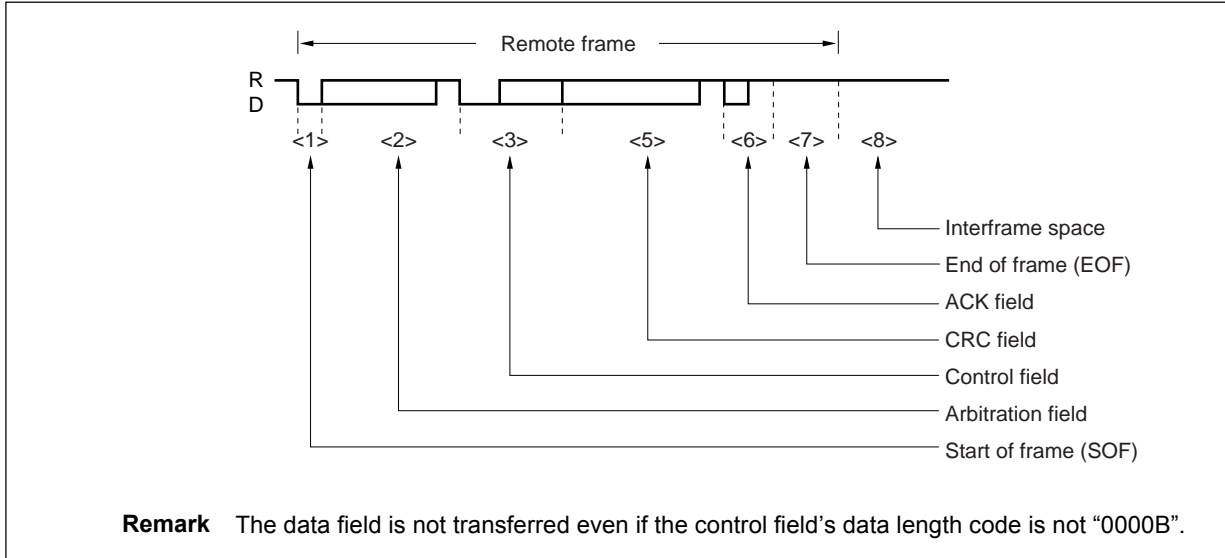


<2> Remote frame

A remote frame is transmitted when the receiving node issues a transmit request.

A remote frame is similar to a data frame, except that the “data field” is deleted and the RTR bit of the “arbitration field” is recessive.

Figure 18-11. Remote Frame

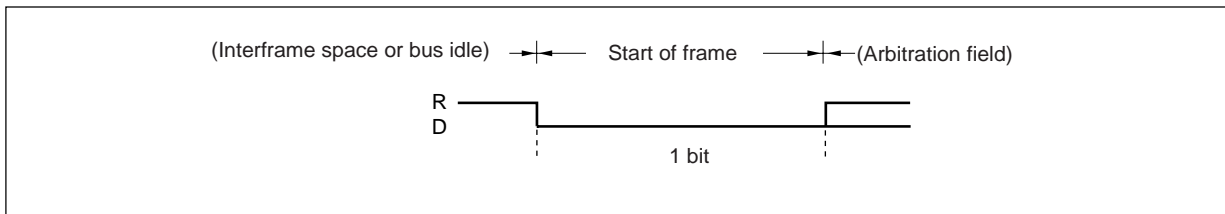


(2) Description of fields

<1> Start of frame (SOF)

The start of frame field is a 1-bit dominant (D) field that is located at the start of a data frame or remote frame.

Figure 18-12. Start of Frame (SOF)



- The start of frame field starts when the bus line level changes.
- When “dominant (D)” is detected at the sample point, reception continues.
- When “recessive (R)” is detected at the sample point, bus idle mode is set.

<2> Arbitration field

The arbitration field is used to set the priority, data frame or remote frame, and protocol mode. This field includes an identifier, frame setting (RTR bit), and protocol mode setting bit.

Figure 18-13. Arbitration Field (in Standard Format Mode)

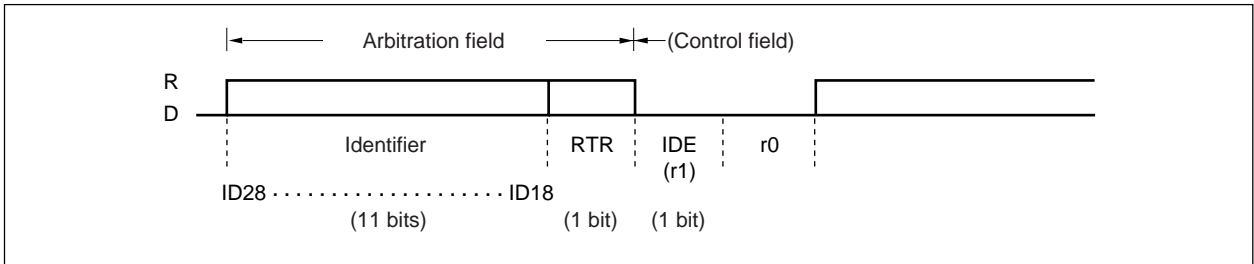
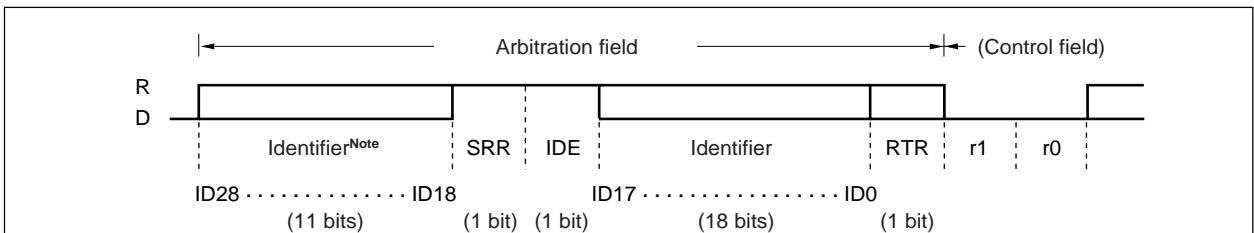


Figure 18-14. Arbitration Field (in Extended Format Mode)



Note Setting the higher 7 bits of the identifier as 1111111B is prohibited.

- Cautions**
1. ID28 to ID0 are identifier bits.
 2. Identifier bits are transferred in MSB-first order.

Table 18-13. RTR Frame Settings

Frame Type	RTR Bit
Data frame	Dominant
Remote frame	Recessive

Table 18-14. Protocol Mode Setting and Number of Identifier (ID) Bits

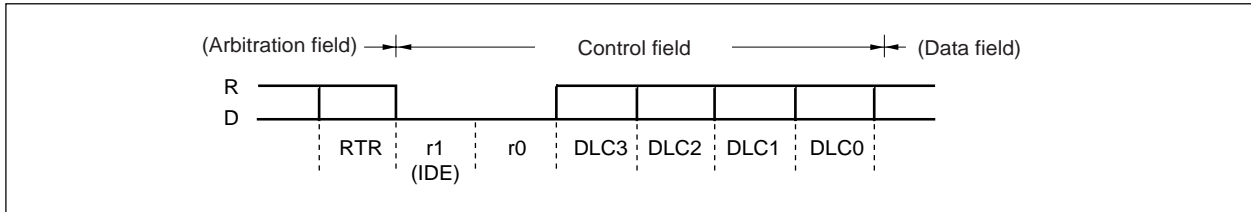
Protocol Mode	SRR Bit	IDE Bit	No. of Bits
Standard format mode	None	Dominant (D)	11 bits
Extended format mode	Recessive (R)	Recessive (R)	29 bits

<3> Control field

The control field sets “N” as the number of data bytes in the data field (N = 0 to 8). r1 and r0 are fixed as dominant (D). The data length code bits (DLC3 to DLC0) set the byte count.

Remark DLC3 to DLC0: Bits 3 to 0 in CAN message data length registers 00 to 31 (M_DLC00 to M_DLC31) (See 18.4.1)

Figure 18-15. Control Field



In standard format mode, the arbitration field's IDE bit is the same bit as the r1 bit.

Table 18-15. Data Length Code Settings

Data Length Code				Data Byte Count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0 bytes
0	0	0	1	1 byte
0	0	1	0	2 bytes
0	0	1	1	3 bytes
0	1	0	0	4 bytes
0	1	0	1	5 bytes
0	1	1	0	6 bytes
0	1	1	1	7 bytes
1	0	0	0	8 bytes
Other than above				8 bytes regardless of the value of DLC3 to DLC0

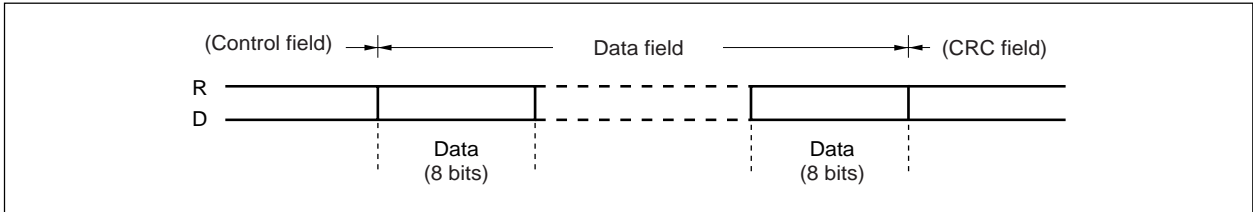
Caution In the remote frame, there is no data field even if the data length code is not 0000B.

<4> Data field

The data field contains the amount of data set by the control field. Up to 8 units of data can be set.

Remark Data units in the data field are each 8 bits long and are ordered MSB first.

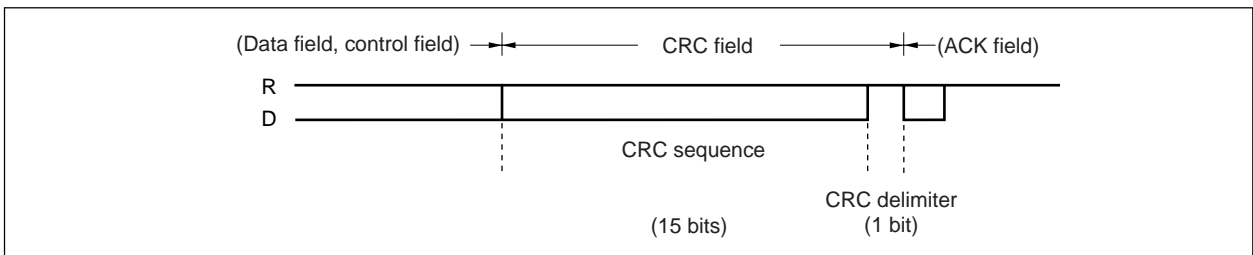
Figure 18-16. Data Field



<5> CRC field

The CRC field is a 16-bit field that is used to check for errors in transmit data. It includes a 15-bit CRC sequence and a 1-bit CRC delimiter.

Figure 18-17. CRC Field

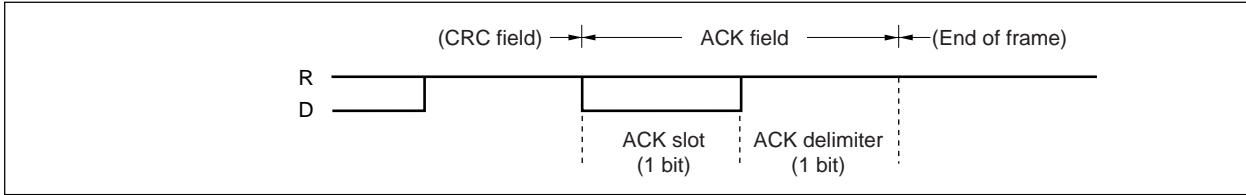


- The polynomial $P(X)$ used to generate the 15-bit CRC sequence is expressed as: $X^{15} + X^{14} + X^{10} + X^8 + X^7 + X^4 + X^3 + 1$
- Transmitting node: No bit stuffing in start of frame, arbitration field, control field, or data field: the transferred CRC sequence is calculated entirely from basic data bits.
- Receiving node: The CRC sequence calculated using data bits that exclude the stuffing bits in the receive data is compared with the CRC sequence in the CRC field. If the two CRC sequences do not match, the node is passed to an error frame.

<6> ACK field

The ACK field is used to confirm normal reception.
It includes a 1-bit ACK slot and a 1-bit ACK delimiter.

Figure 18-18. ACK Field

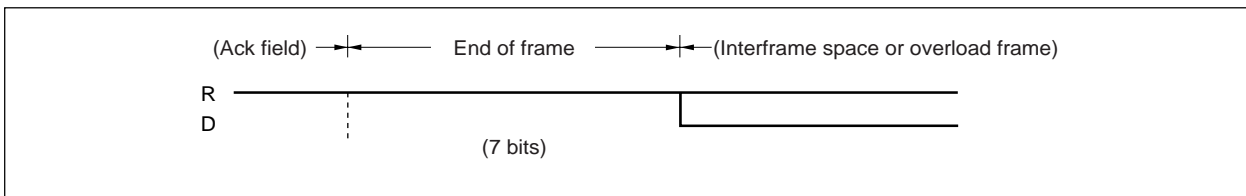


- The receiving node outputs the following depending on whether or not an error is detected between the start of frame field and the CRC field.
 - If an error is detected: ACK slot = Recessive (R)
 - If no error is detected: ACK slot = Dominant (D)
- The transmitting node outputs two “recessive(R)” bits and confirms the receiving node’s receive status.

<7> End of frame (EOF)

The end of frame field indicates the end of transmission or reception.
It includes 7 “recessive(R)” bits.

Figure 18-19. End of Frame (EOF)



<8> Interframe space

The interframe space is inserted after the data frame, remote frame, error frame, and overload frame to separate one frame from the next one.

- Error active node

When the bus is idle, transmit enable mode is set for each node. Transmission then starts from a node that has received a transmit request.

If the node is an error active node, the interframe space is composed of a 3- or 2-bit intermission field and bus idle field.

- Error passive node

After an 8-bit bus idle field, transmit enable mode is set. Receive mode is set if a transmission starts from a different node during bus idle mode.

The error passive node is composed of an intermission field, suspend transmission field, and bus idle field.

Figure 18-20. Interframe Space

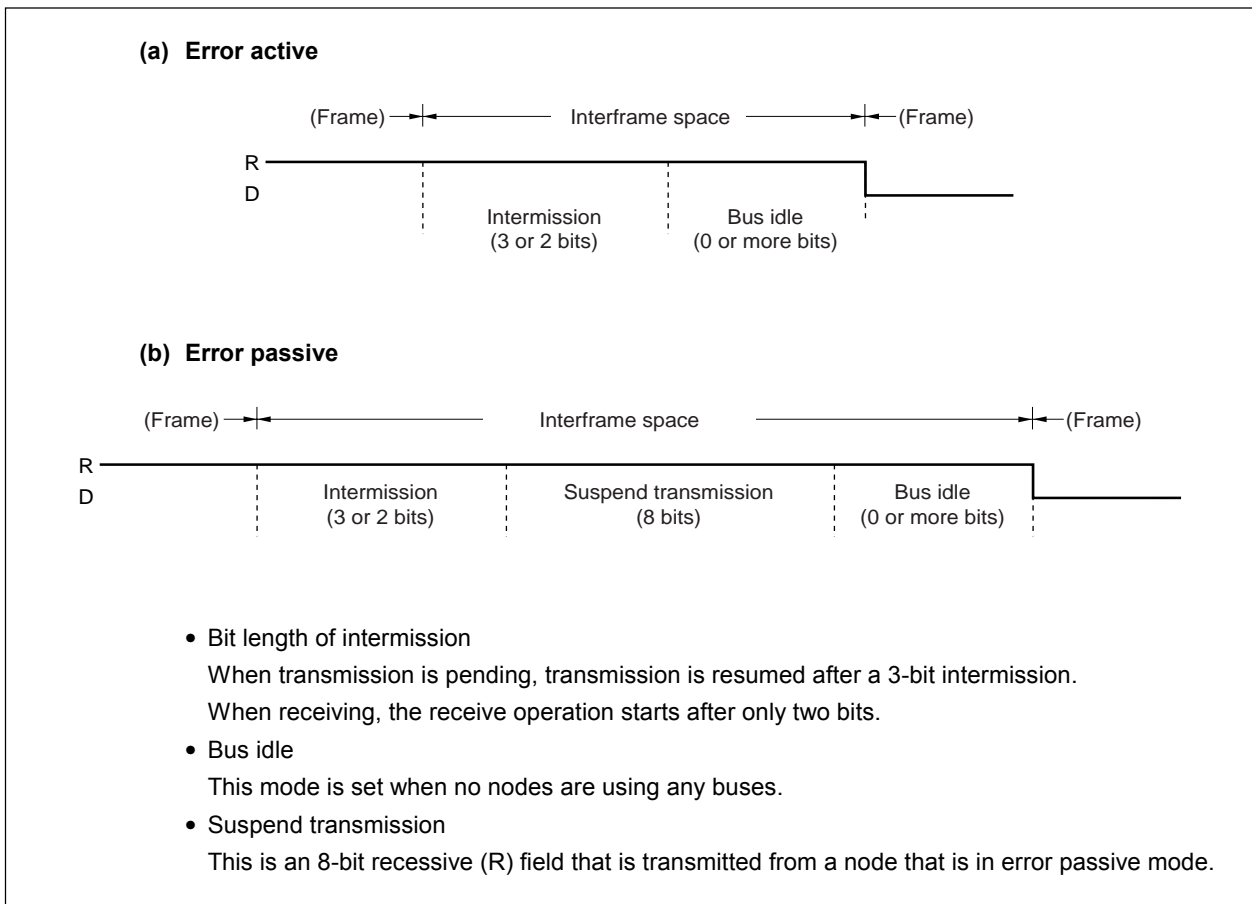


Table 18-16. Operation When Third Bit of Intermission Is “Dominant (D)”

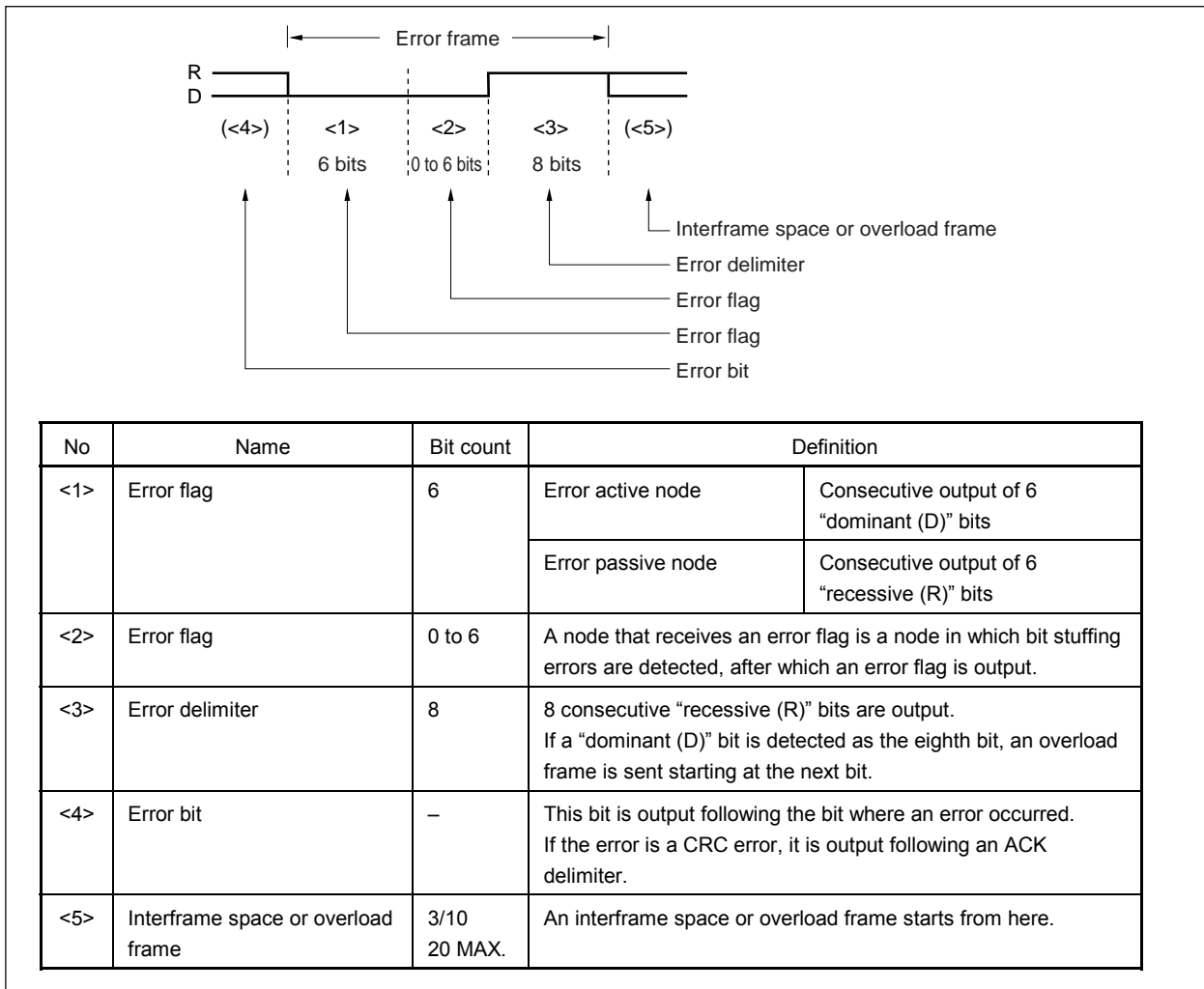
Transmit Status	Operation
No pending transmissions	A receive operation is performed when start of frame output by other node is detected.
Pending transmission exists	The identifier is transmitted when start of frame output by local node is detected.

<9> Error frame

An error frame is used to output from a node in which an error has been detected.

When a passive error flag is being output, if there is “dominant (D)” output from another node, the passive error flag does not end until 6 consecutive bits are detected on the same level. If the bit following the 6 consecutive “recessive (R)” bits is “dominant (D)”, the error frame ends when the next “recessive (R)” bit is detected.

Figure 18-21. Error Frame

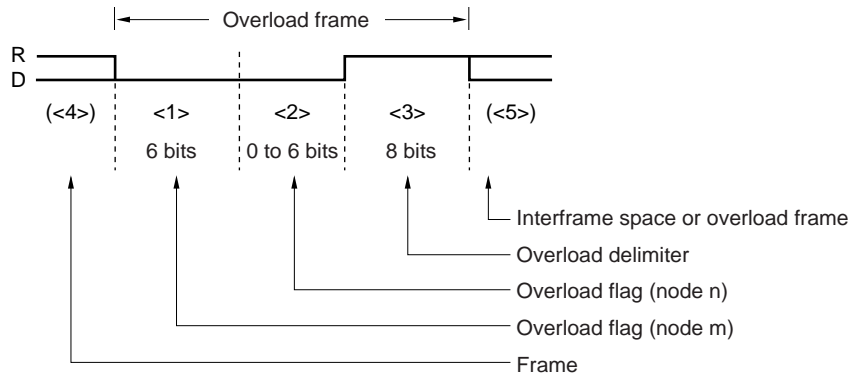


<10> Overload frame

An overload frame is output starting from the first bit in an intermission in cases where the receiving node is not yet ready to receive.

If a bit error is detected during intermission mode, it is output starting from the bit following the bit where the bit error was detected.

Figure 18-22. Overload Frame



No	Name	Bit count	Definition
<1>	Overload flag starting from node m	6	Consecutive output of 6 “dominant (D)” bits. Output when node m is not ready to receive.
<2>	Overload flag starting from node n	0 to 6	Node n, which has received an overload flag in the interframe space, outputs an overload flag
<3>	Overload delimiter	8	8 consecutive “recessive (R)” bits are output. If a “dominant (D)” bit is detected as the eighth bit, an overload frame is sent starting at the next bit.
<4>	Frame	–	Output following an end of frame, error delimiter, or overload delimiter.
<5>	Interframe space or overload frame	3/10 20 MAX.	An interframe space or overload frame starts from here.

Remark $n \neq m$

18.10 Functions

18.10.1 Determination of bus priority

(1) When one node has starting transmitting

- In bus idle mode, the node that outputs data first starts transmitting.

(2) When several nodes have started transmitting

- The node that has the longest string of consecutive “dominant (D)” bits starting from the first bit in the arbitration field has top priority for bus access (“dominant (D)” bits take precedence due to wired OR bus arbitration).
- The transmitting node compares the arbitration field which it has output and the bus data level

Table 18-17. Determination of Bus Priority

Matched levels	Transmission continues
Mismatched levels	When a mismatch is detected, data output stops at the next bit, and the operation switches to reception.

(3) Priority between data frame and remote frame

- If a bus conflict occurs between a data frame and a remote frame, the data frame takes priority because its last bit (RTR) is “dominant (D)”.

18.10.2 Bit stuffing

Bit stuffing is when one bit of inverted data is added for resynchronization to prevent burst errors when the same level is maintained for at least five consecutive bits.

Table 18-18. Bit Stuffing

Transmit	When transmitting data frames and remote frames, if the same level is maintained for at least five bits between the start of frame and CRC fields, one bit of data whose level is inverted from the previous level is inserted before the next bit.
Receive	When receiving data frames and remote frames, if the same level is maintained for at least five bits between the start of frame and CRC fields, the next bit of data is deleted before reception is resumed.

18.10.3 Multiple masters

Since bus priority is determined based on the identifier, any node can be used as the bus master.

18.10.4 Multi-cast

Even when there is only one transmitting node, the same identifier can be set for several nodes, so that the same data can be received by several nodes at the same time.

18.10.5 CAN sleep mode/CAN stop mode function

The CAN sleep mode/CAN stop mode function can be used to set the FCAN controller to sleep (standby) mode to reduce power consumption.

The CAN sleep mode is set via the procedure stipulated in the CAN specifications. The CAN sleep mode can be set to wake up by the bus operation, however the CAN stop mode cannot be set to wake up by bus operation (this is controlled via CPU access).

18.10.6 Error control function

(1) Types of errors

Table 18-19. Types of Errors

Error Type	Description of Error		Detected Status	
	Detection Method	Detection Condition	Transmit/Receive	Field/Frame
Bit error	Comparison of output level and bus level (excludes stuff bits)	Mismatch between levels	Transmitting/receiving nodes	Bits outputting data on bus in start of frame to end of frame, error frame, or overload frame
Stuff error	Use stuff bits to check receive data	Six consecutive bits of same-level data	Transmitting/receiving nodes	Start of frame to CRC sequence
CRC error	Comparison of CRC generated from receive data and received CRC sequence	CRC mismatch	Receiving node	Start of frame to data field
Form error	Check fixed-format field/frame	Detection of inverted fixed format	Receiving node	<ul style="list-style-type: none"> • CRC delimiter • ACK field • End of frame • Error frame • Overload frame
ACK error	Use transmitting node to check ACK slot	Use ACK slot to detect recessive	Transmitting node	ACK slot

(2) Error frame output timing

Table 18-20. Error Frame Output Timing

Error Type	Output Timing
Bit error, stuff error, form error, ACK error	Error frame is output at the next bit following the bit where error was detected
CRC error	Error frame is output at the next bit following the ACK delimiter

(3) Handling of errors

The transmitting node retransmits the data frame or remote frame after the error frame has been transmitted.

(4) Error statuses

(a) Types of error statuses

The three types of error statuses are listed below.

- Error active
- Error passive
- Bus off

- The error status is controlled by the transmit error counter and receive error counter (see **18.4.22 CANn error count register (CnERC)**).
- The various error statuses are categorized according to their error counter values.
- The error flags used for error statuses differ between transmit and receive operations.
- When the error counter value reaches 96 or more, the bus status must be tested since the bus may become seriously damaged.
- During start-up, if only one node is active, the error frame and data are repeatedly re-sent because no ACK is returned even data has been transmitted.

In such cases, bus off mode cannot be set. Even if the transmitting node that is sending the transmit message repeatedly experiences an error status, bus off mode cannot be set.

Table 18-21. Types of Error Statuses

Error Status Type	Operation	Error Counter Value	Type of Output Error Flag
Error active	Transmit/ receive	0 to 127	Active error flag (6 consecutive “dominant (D)” bits)
Error passive	Transmit	128 to 255	Passive error flag (6 consecutive “recessive (R)” bits)
	Receive	128 or more	
Bus off	Transmit	256 or more	Transfer is not possible. When a string of at least 11 consecutive “recessive (R)” bits occurs 128 times, the error counter is zero-cleared and error active status can be resumed.

(b) Error counter

The error counter value is incremented each time an error occurs and is decremented when a transmit or receive operation ends normally. The count up/count down timing occurs at the first bit of the error delimiter.

Table 18-22. Error Counter

Status	Transmit Error Counter (TEC7 to TEC0)	Receive Error Counter (REC7 to REC0)
When receiving node has detected an error (except for bit errors that occur in an active error flag or overload flag)	No change	+1
When “dominant (D)” is detected following error frame’s overload flag output by the receiving node	No change	+8
When transmitting node has sent an error flag [When error counter = ±0] <1> When an ACK error was detected in error passive status and a “dominant (D)” was not detected during error flag output <2> When a stuff error occurs in the arbitration field	+8	No change
Detection of bit error during output of active error flag or overload flag (transmitting node with error active status)	+8	No change
Detection of bit error during output of active error flag or overload flag (receiving node with error active status)	No change	+8
When 14 consecutive “dominant (D)” bits were detected from the start of each node’s active error flag or overload flag, followed by detection of eight consecutive dominant bits. Each node has detected eight consecutive dominant bits after a passive error flag.	+8	+8
The transmitting node has completed a transmit operation without any errors (±0 if error counter value is 0).	-1	No change
The receiving node has completed a receive operation without any errors.	No change	<ul style="list-style-type: none"> • -1 (1 ≤ REC7 to REC0 ≤ 127) • ±0 (REC7 to REC0 = 0) • 127 is set (REC7 to REC0 > 127)

(c) Occurrence of bit error during intermission

In this case, an overload frame occurs.

Caution When an error occurs, error control is performed according to the contents of the transmitting and receiving error counters as they existed prior to the error’s occurrence. The error counter value is incremented only after an error flag has been output.

18.10.7 Baud rate control function

★ (1) Prescaler

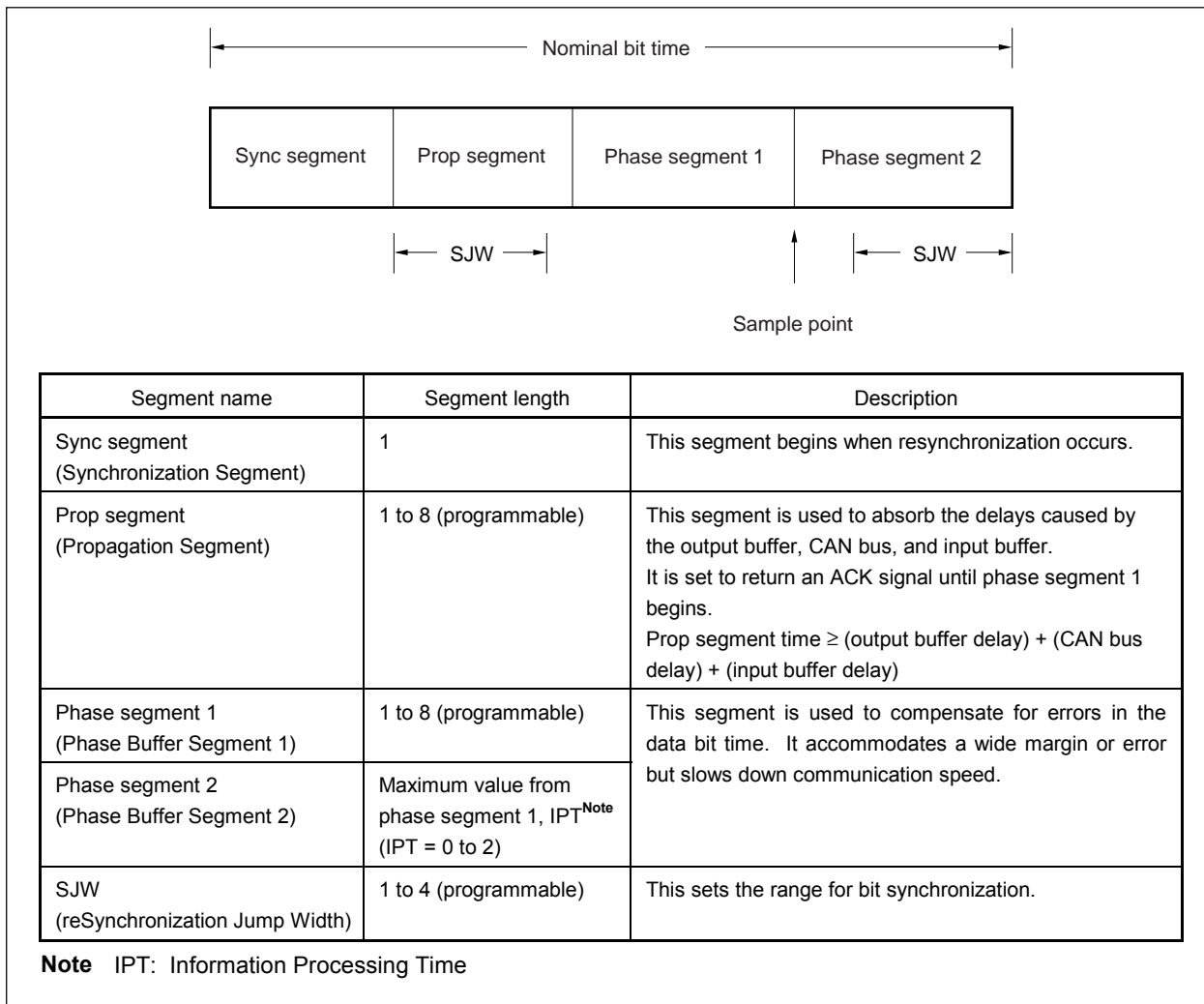
The V850/SF1 includes a prescaler for dividing the CAN module clock (f_{MEM}). This prescaler generates a clock (f_{BTL}) that is based on a division ratio ranging from 2 to 128 applied to the system clock when the CnBRP register's TLM bit = 0, and from 2 to 256 when the TLM bit = 1 (see 18.4.25 CANn bit rate prescaler register (CnBRP)).

★ (2) Nominal bit time (8 to 25 time quantum)

The definition of 1 data bit time is shown below.

Remark 1 time quantum = $1/f_{BTL}$

Figure 18-23. Nominal Bit Time



(3) Data bit synchronization

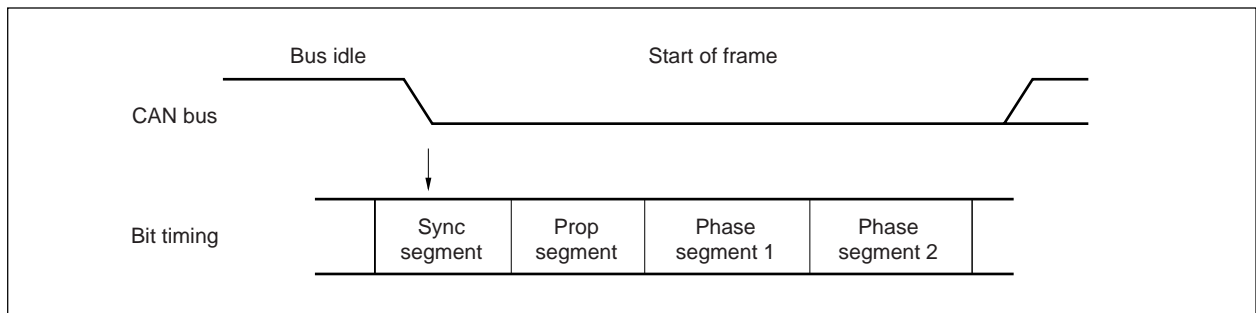
- Since the receiving node has no synchronization signal, synchronization is performed using level changes that occur on the bus.
- As for the transmitting node, data is transmitted in sync with the transmitting node's bit timing.

(a) Hardware synchronization

This is bit synchronization that is performed when the receiving node has detected a start of frame in bus idle mode.

- When a falling edge is detected on the bus, the current bit is assigned to the sync segment and the next bit is assigned to the prop segment. In such cases, synchronization is performed regardless of the SJW.
- Since bit synchronization must be established after a reset or after a wake-up, hardware synchronization is performed only at the first level change that occurs on the bus (for the second and subsequent level changes, bit synchronization is performed as shown below).

Figure 18-24. Coordination of Data Bit Synchronization



(b) Resynchronization

Resynchronization is performed when a level change is detected on the bus during a receive operation.

- The edge's phase error is produced by the relative positions of the detected edge and sync segment.

<Phase error symbols>

- 0: When edge is within sync segment
- Positive: Edge is before sample point (phase error)
- Negative: Edge is after sample point (phase error)

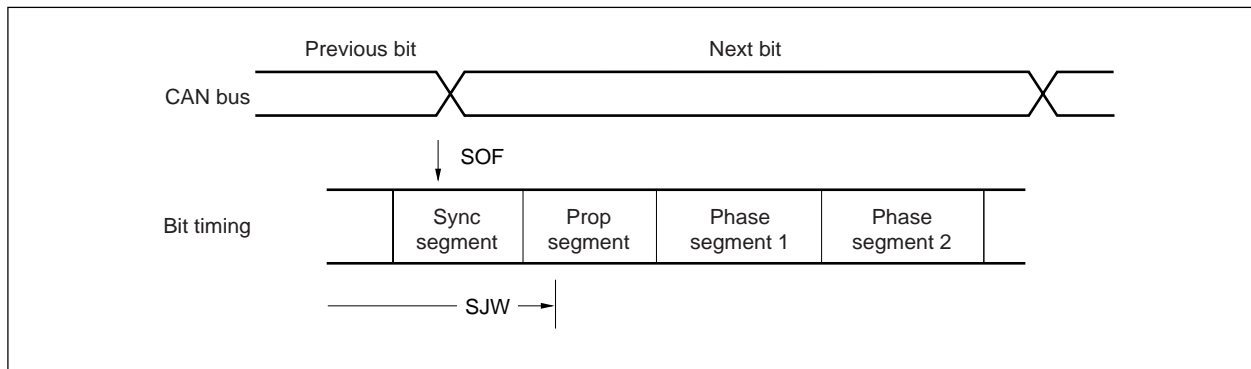
- When the edge is detected as within the bit timing specified by the SJW, synchronization is performed in the same way as hardware synchronization.
- When the edge is detected as extending beyond the bit timing specified by the SJW, synchronization is performed on the following basis.

When phase error is positive: Phase segment 1 is lengthened to equal the SJW

When phase error is negative: Phase segment 2 is shortened to equal the SJW

- A "shifting" of the baud rate for the transmitting and receiving nodes moves the relative position of the sample point for data on the receiving node.

Figure 18-25. Resynchronization



18.11 Operations

18.11.1 Initialization processing

Figure 18-26 shows a flowchart of initialization processing. The register setting flow is shown in Figures 18-27 to 18-38.

Figure 18-26. Initialization Processing

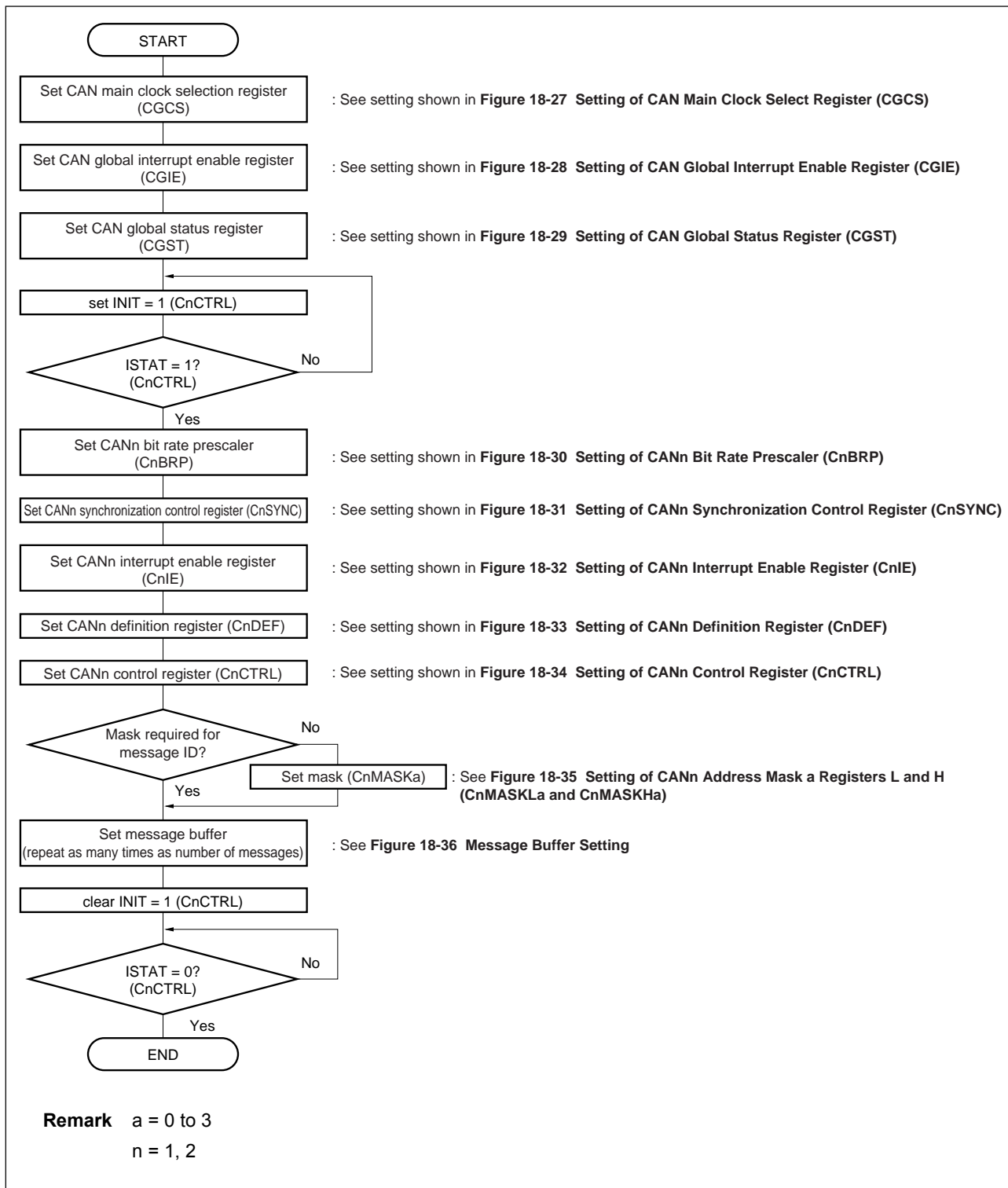
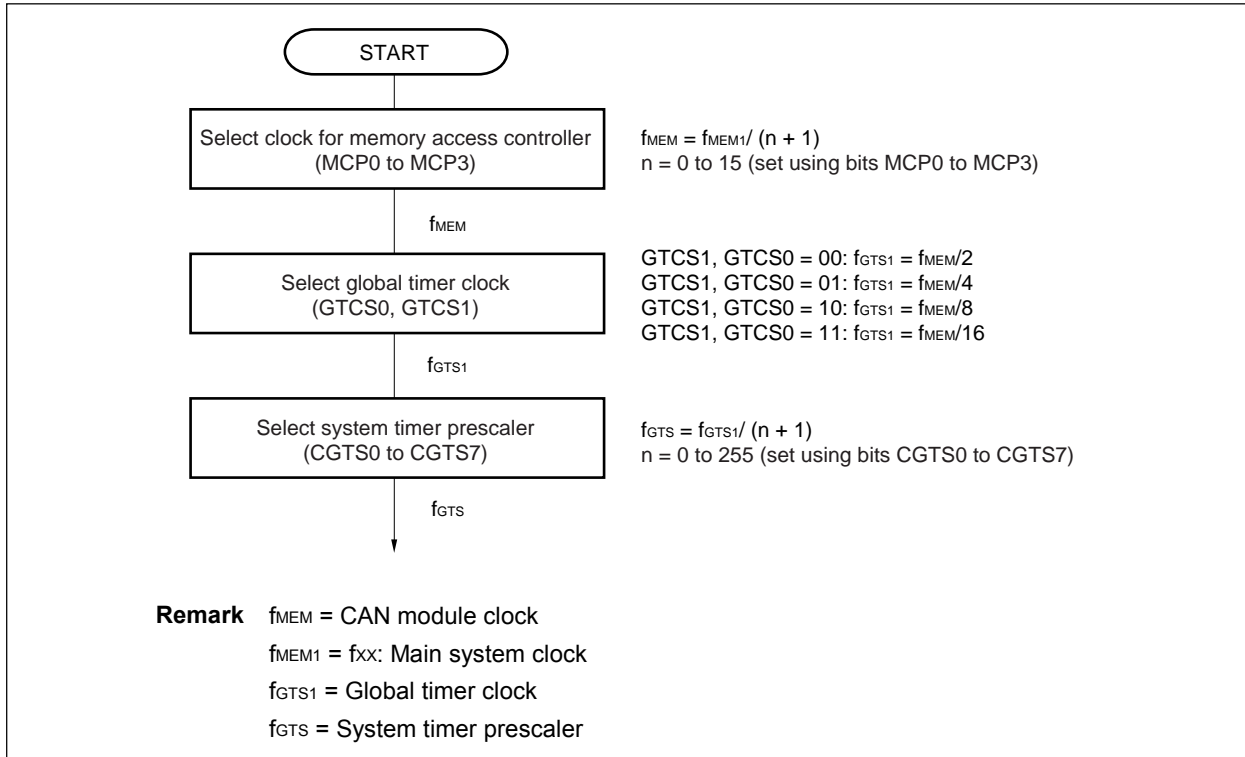


Figure 18-27. Setting of CAN Main Clock Select Register (CGCS)



★ Figure 18-28. Setting of CAN Global Interrupt Enable Register (CGIE)

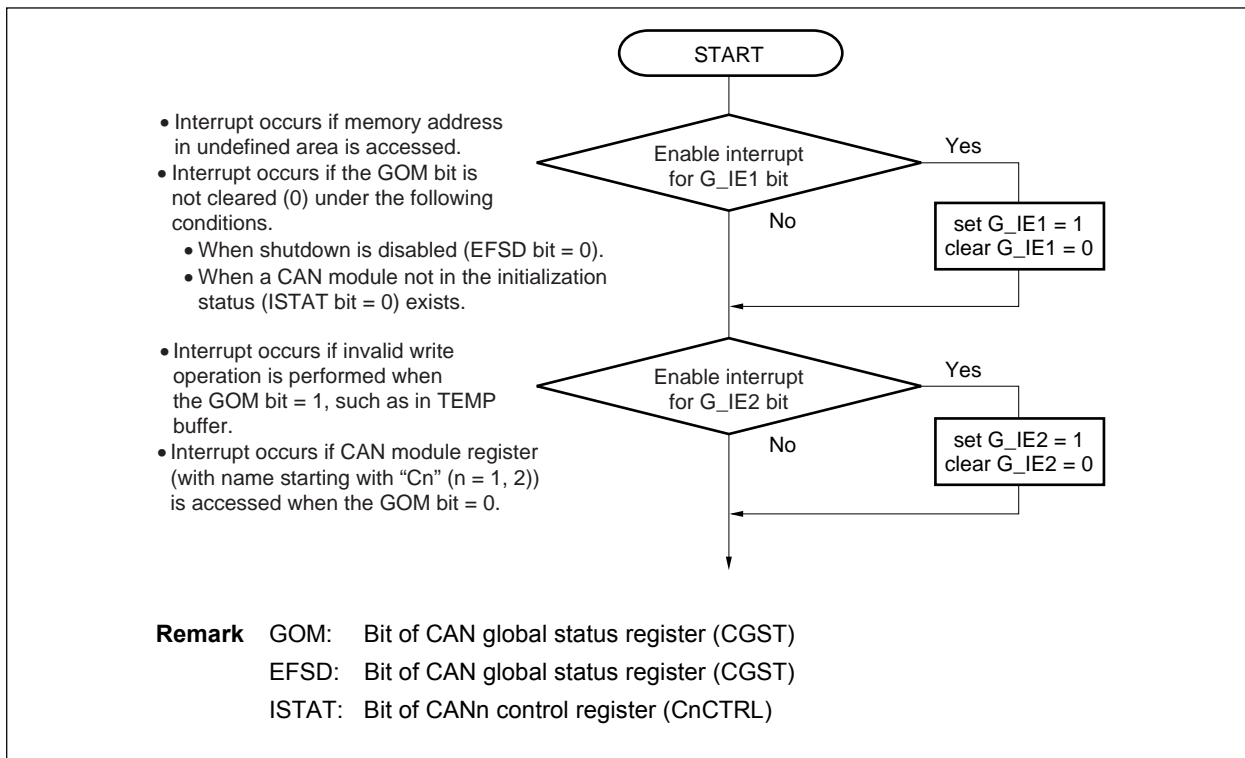
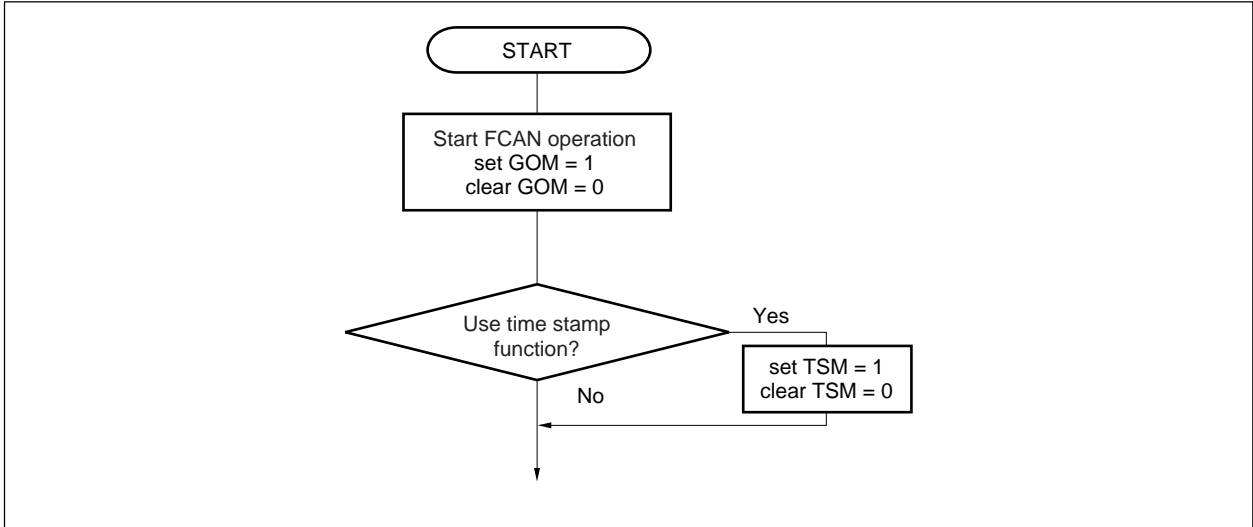


Figure 18-29. Setting of CAN Global Status Register (CGST)



★

Figure 18-30. Setting of CANn Bit Rate Prescaler Register (CnBRP)

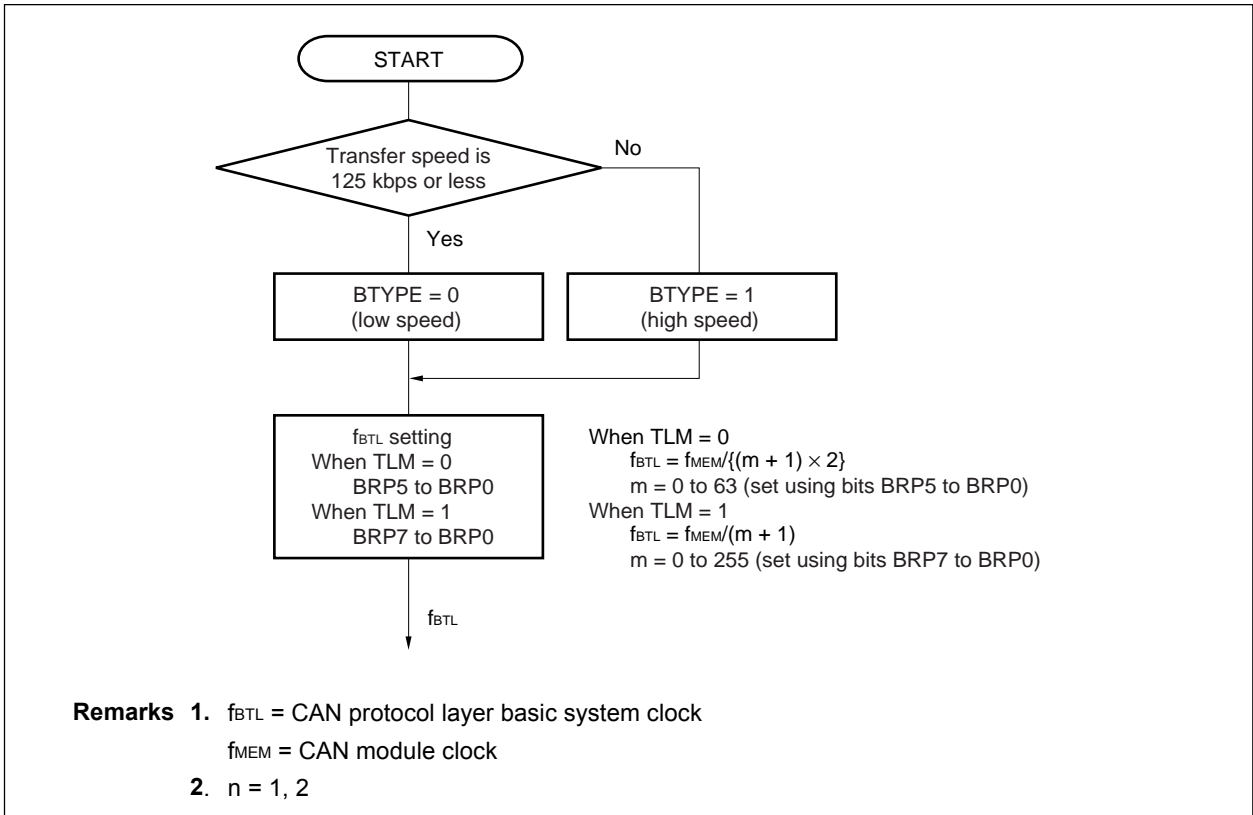


Figure 18-31. Setting of CANn Synchronization Control Register (CnSYNC)

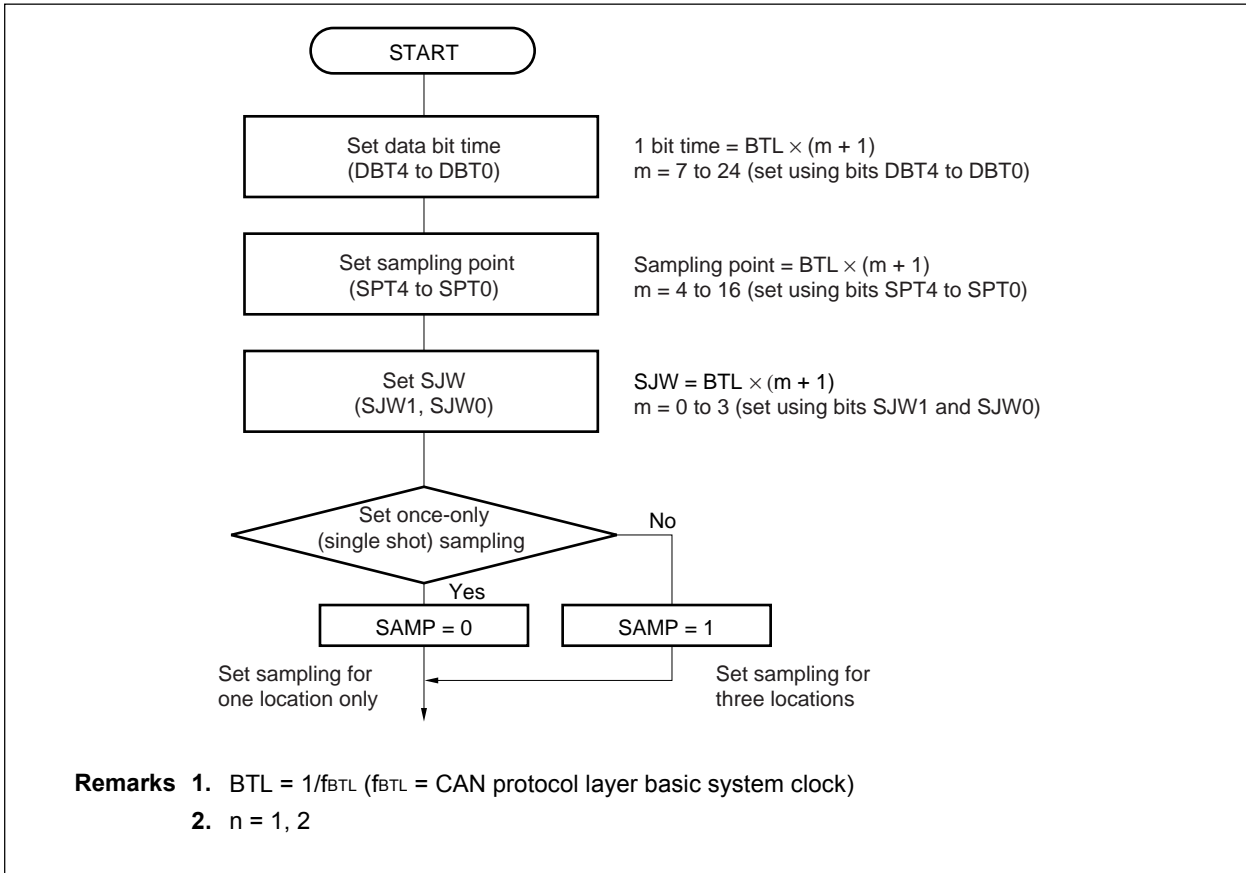


Figure 18-32. Setting of CANn Interrupt Enable Register (CnIE)

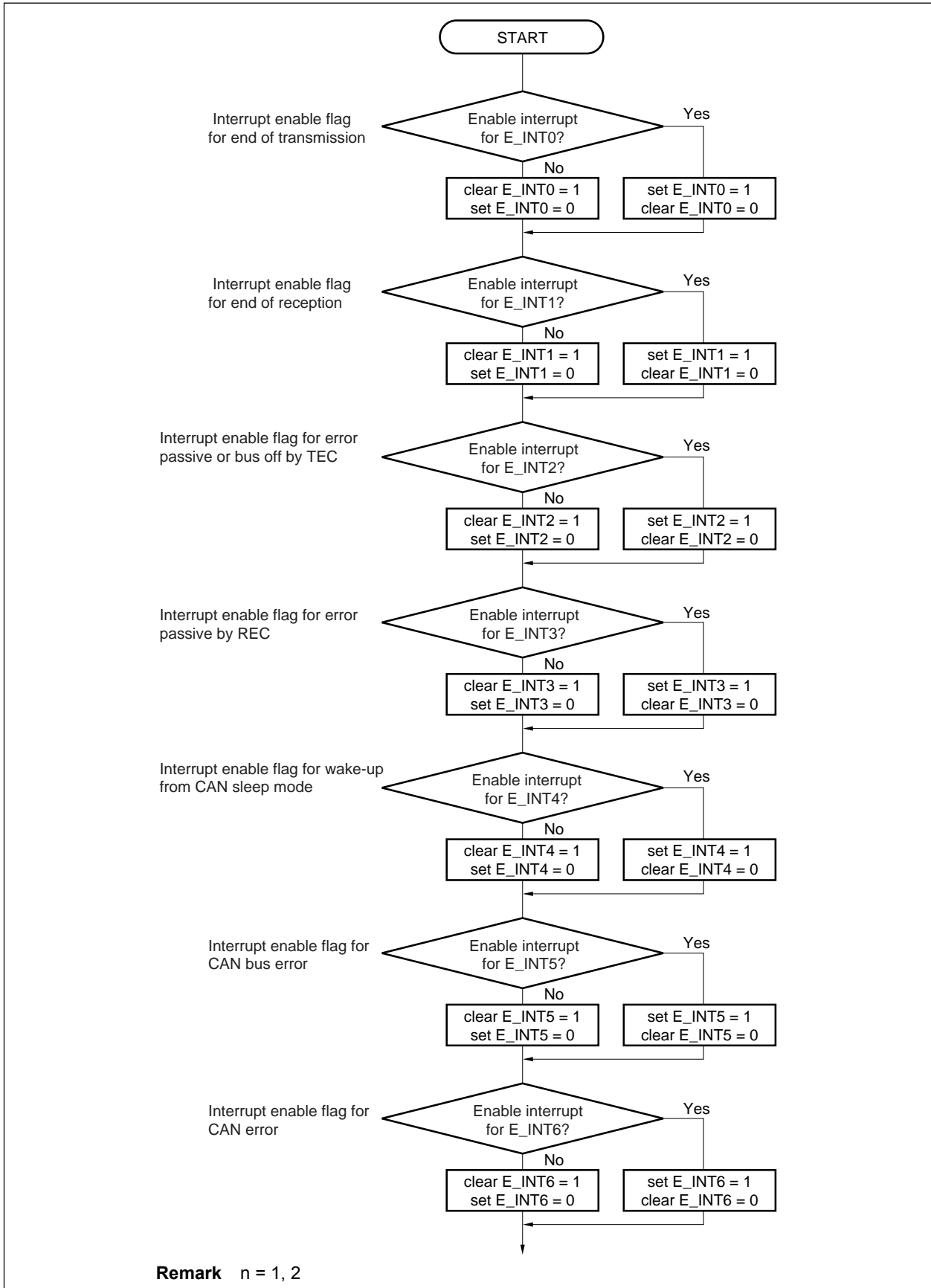


Figure 18-33. Setting of CANn Definition Register (CnDEF)

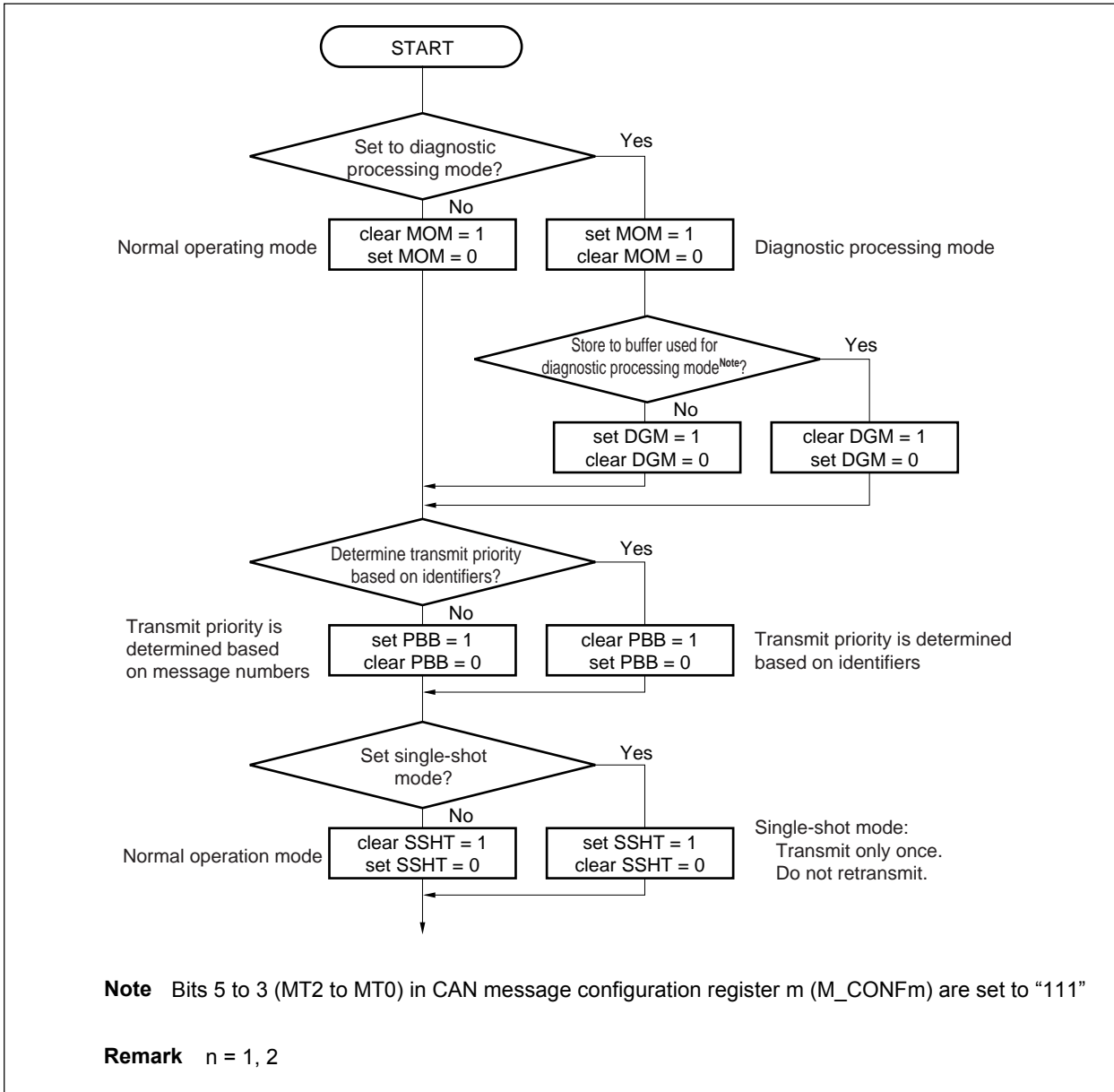


Figure 18-34. Setting of CANn Control Register (CnCTRL)

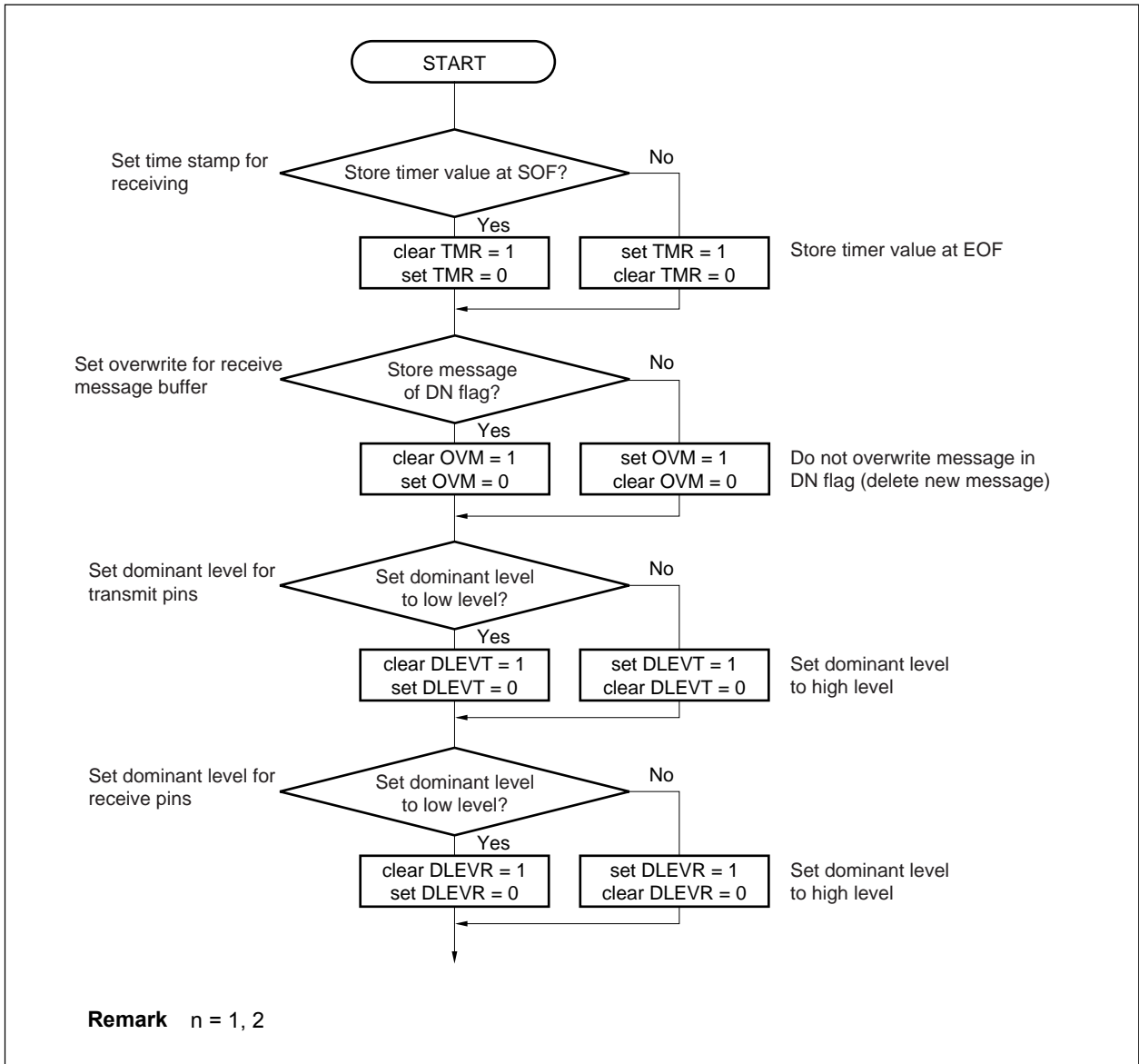


Figure 18-35. Setting of CANn Address Mask a Registers L and H (CnMASKLa and CnMASKHa)

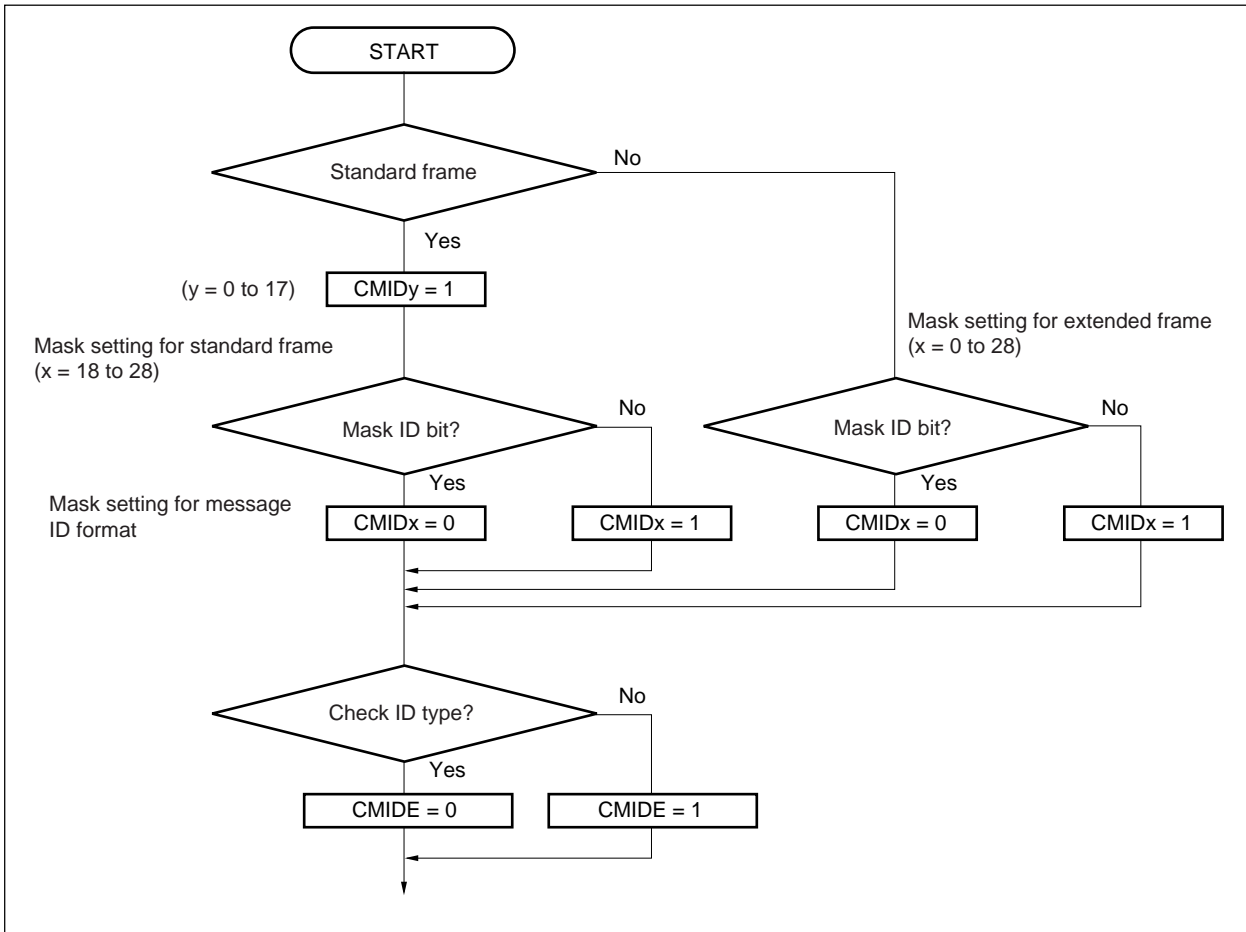


Figure 18-36. Message Buffer Setting

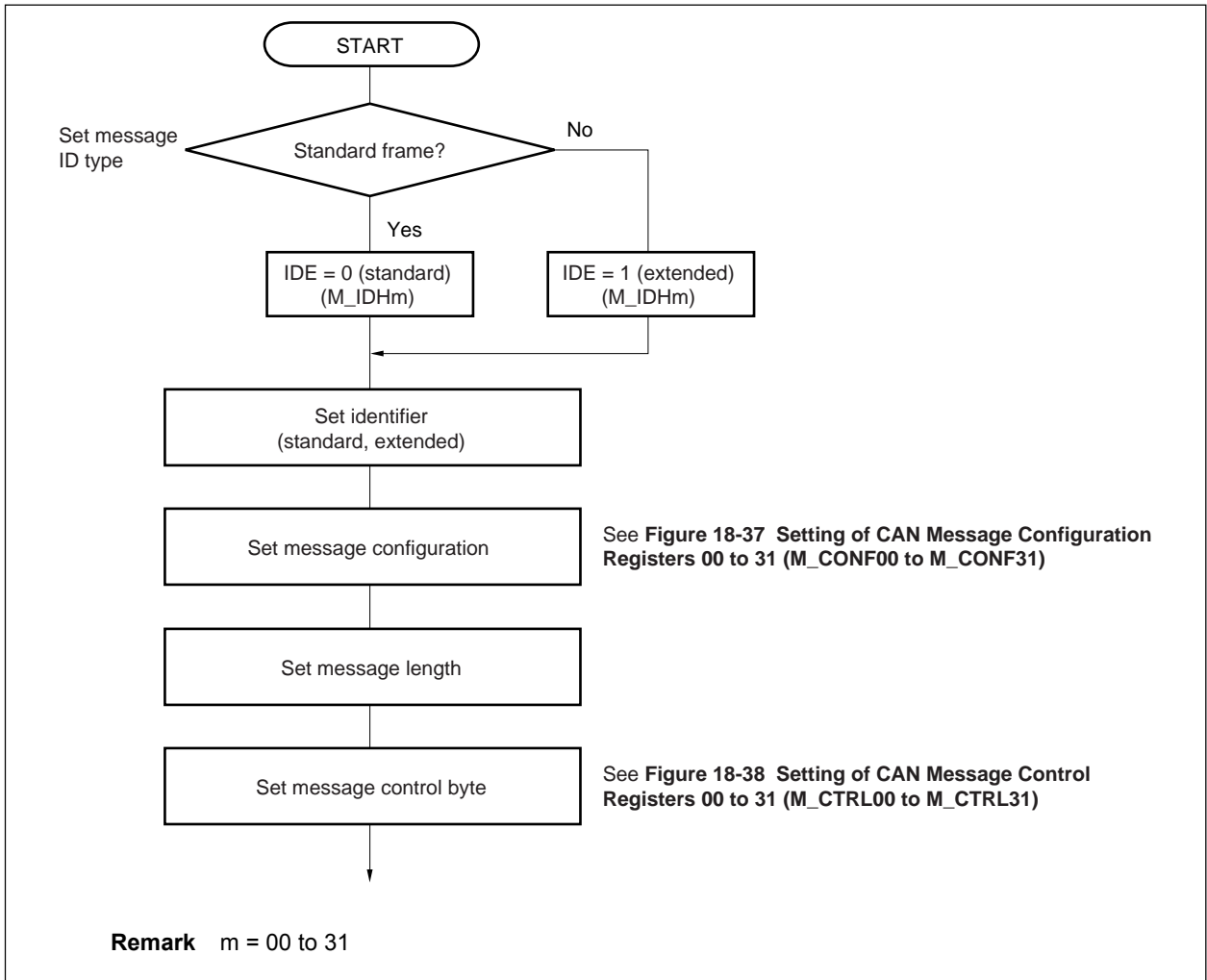


Figure 18-37. Setting of CAN Message Configuration Registers 00 to 31 (M_CONF00 to M_CONF31)

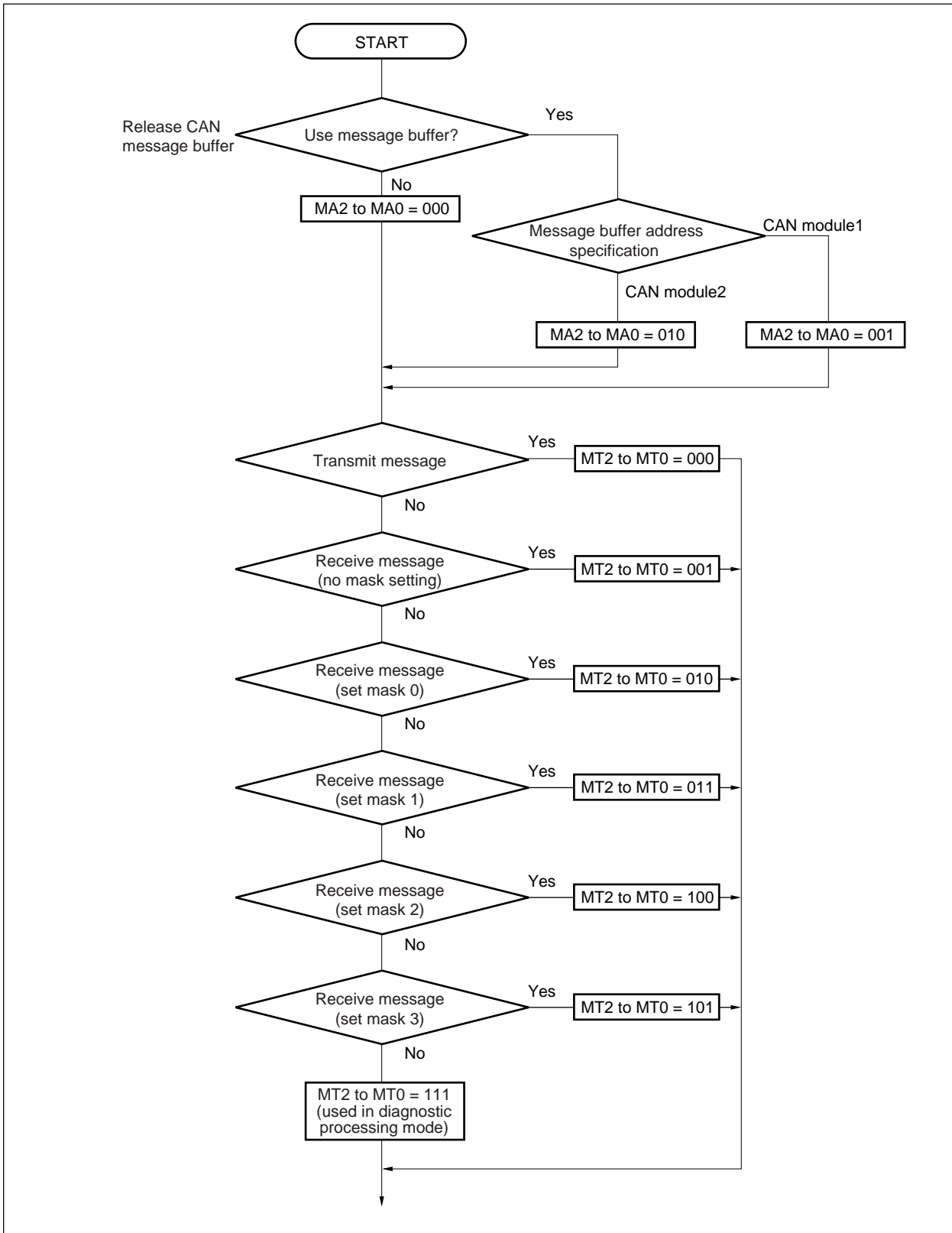
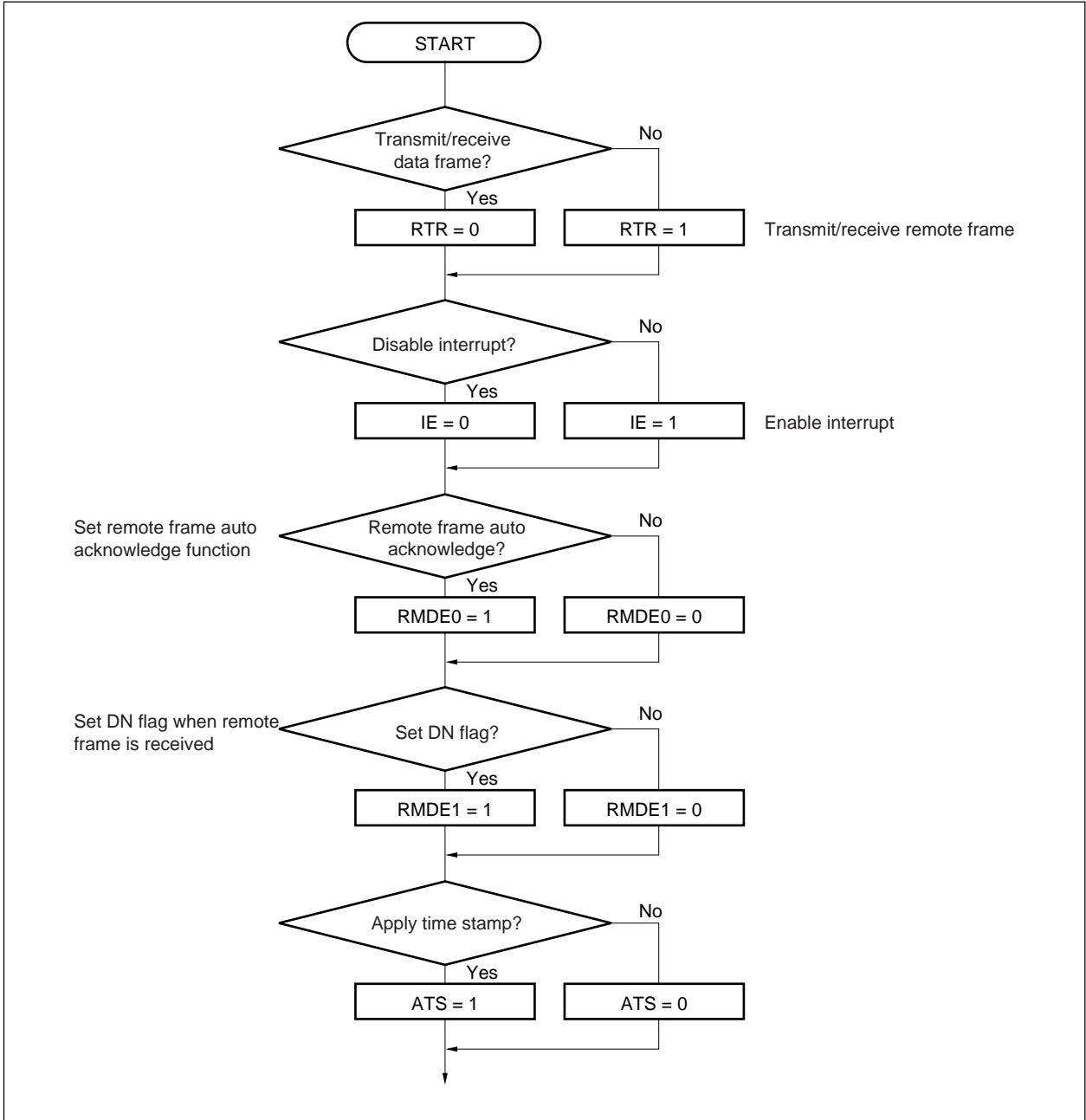


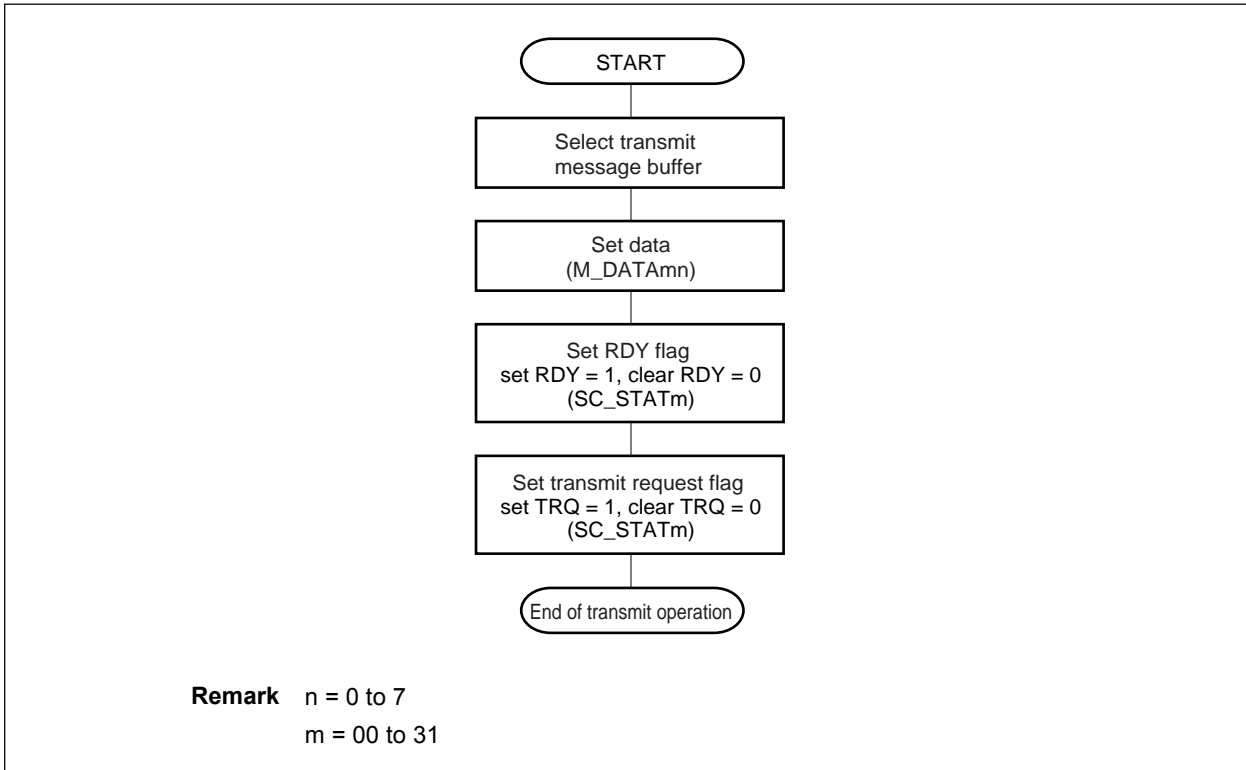
Figure 18-38. Setting of CAN Message Control Registers 00 to 31 (M_CTRL00 to M_CTRL31)



18.11.2 Transmit setting

Transmit messages are output from the target message buffer.

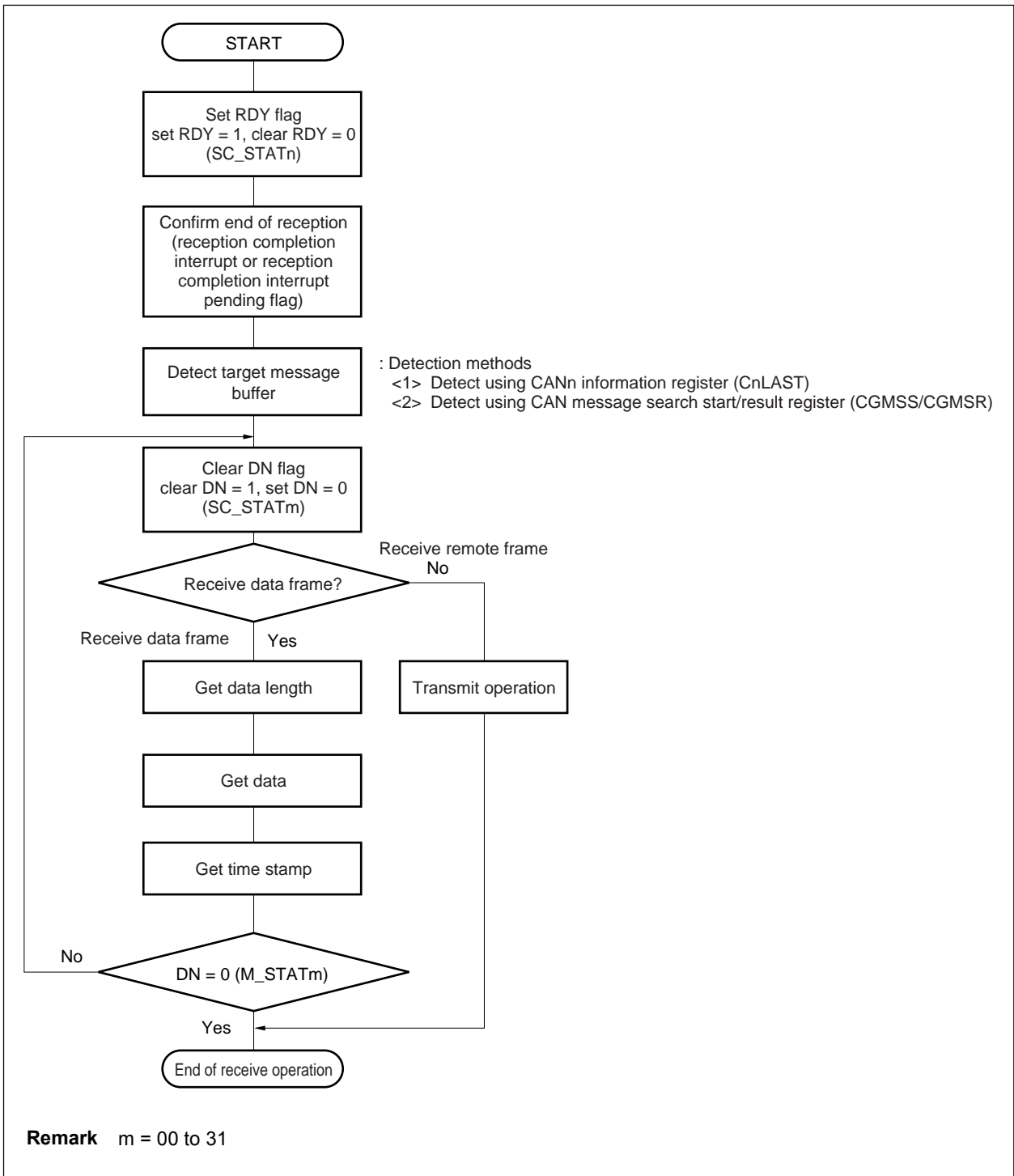
Figure 18-39. Transmit Setting



★ 18.11.3 Receive setting

Receive messages are retrieved from the target message buffer.

Figure 18-40. Receive Setting



18.11.4 CAN sleep mode

In CAN sleep mode, the FCAN controller can be set to standby mode. A wake-up occurs when there is a bus operation.

★

Figure 18-41. CAN Sleep Mode Setting

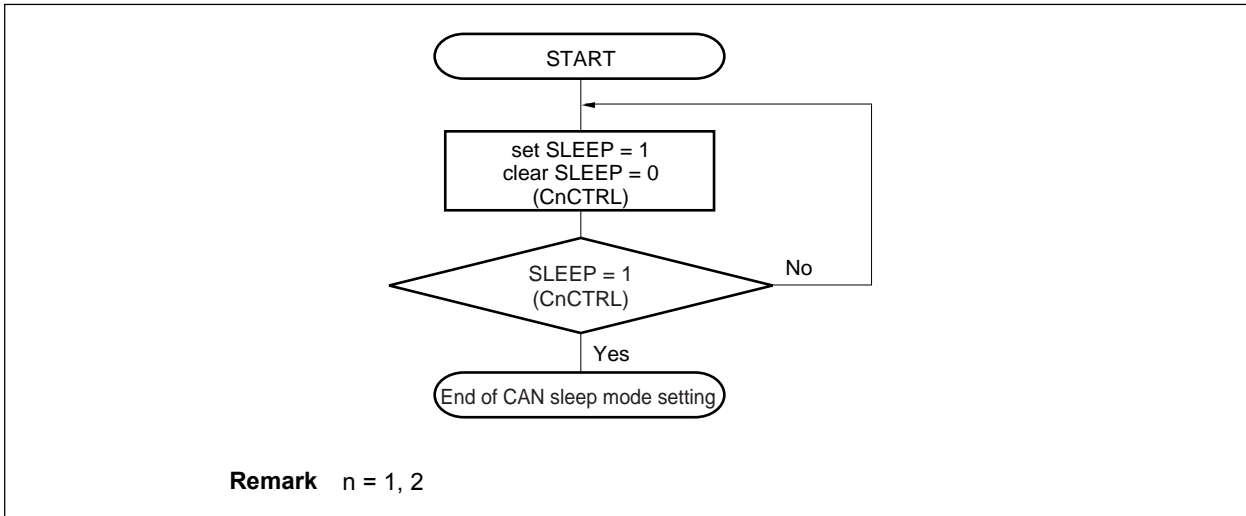


Figure 18-42. Clearing of CAN Sleep Mode by CAN Bus Active Status

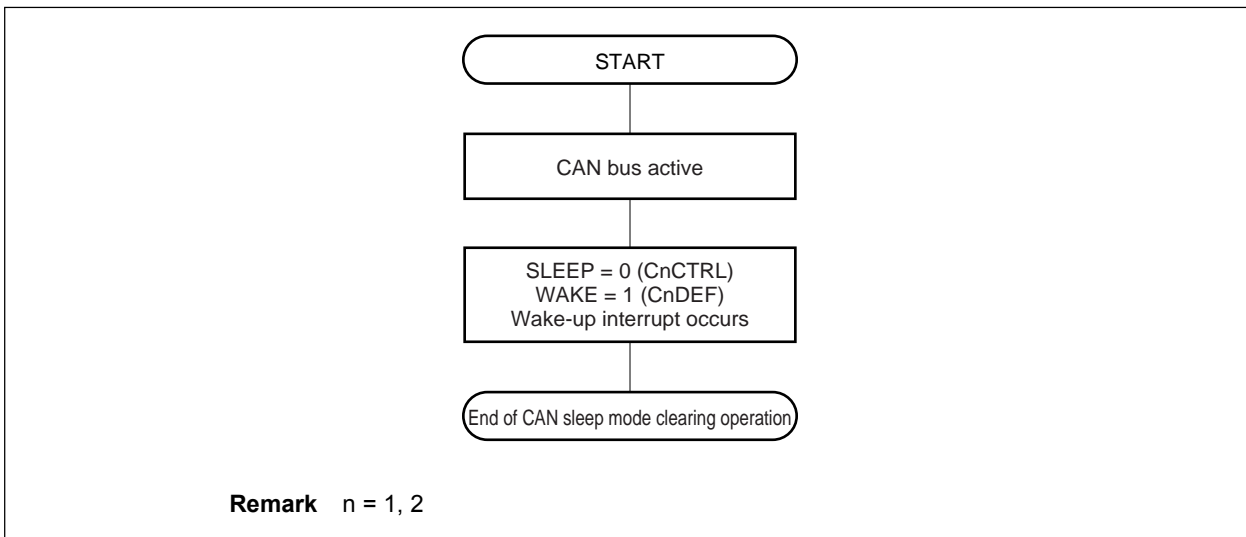
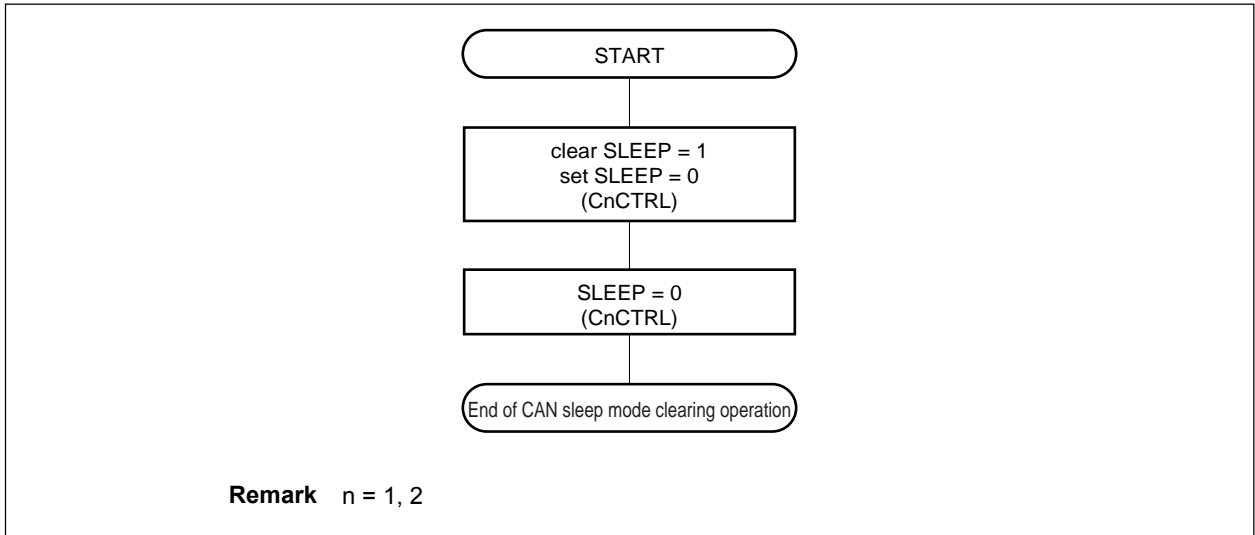


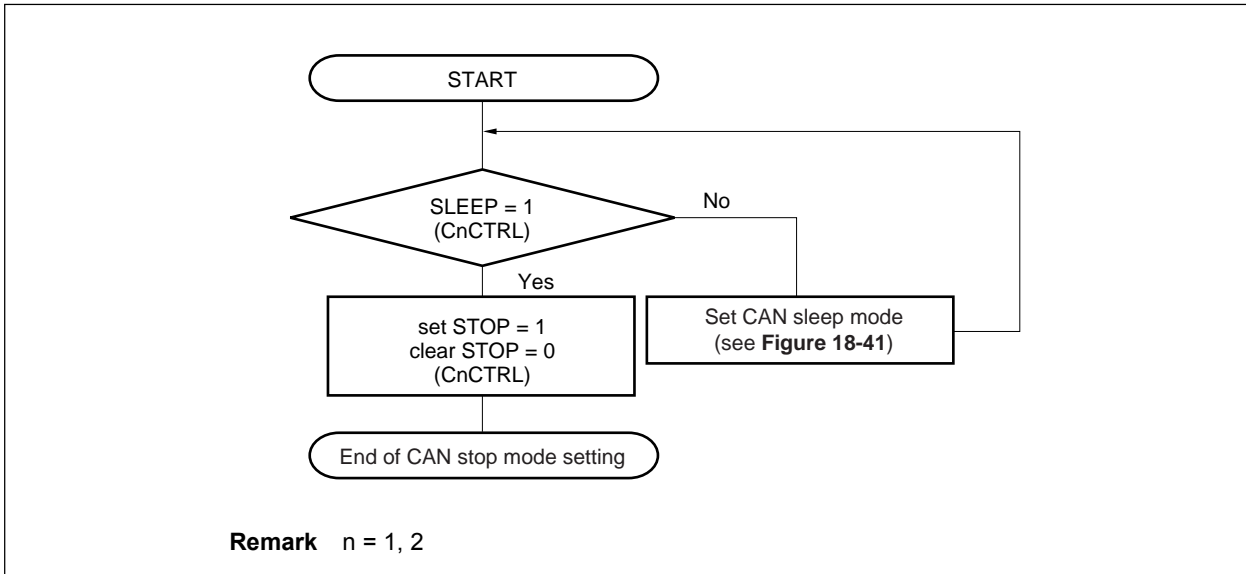
Figure 18-43. Clearing of CAN Sleep Mode by CPU



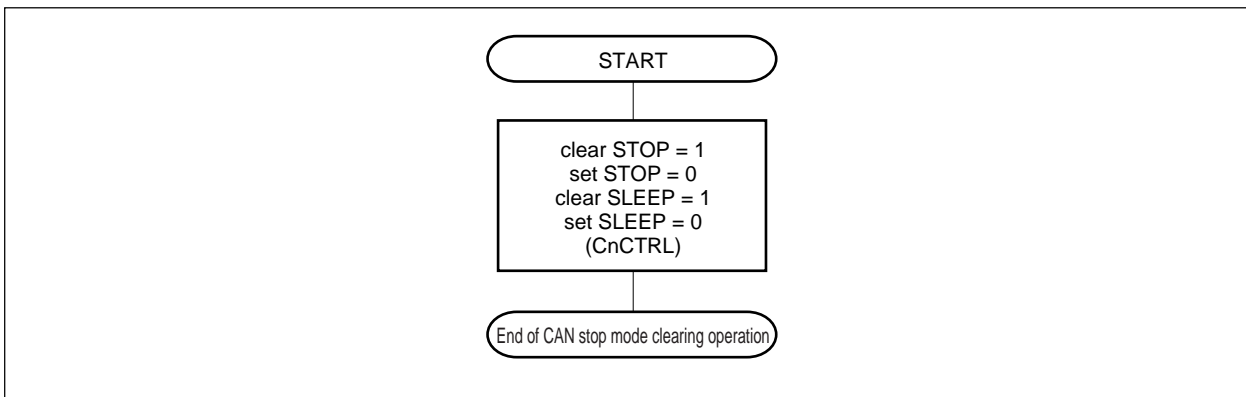
18.11.5 CAN stop mode

In CAN stop mode, the FCAN controller can be set to standby mode. No wake-up occurs when there is a bus operation (stop mode is controlled by CPU access only).

★ **Figure 18-44. CAN Stop Mode Setting**



★ **Figure 18-45. Clearing of CAN Stop Mode**



★ **18.12 Rules for Correct Setting of Baud Rate**

The CAN protocol limit values for ensuring correct operation of FCAN are described below. If these limit values are exceeded, a CAN protocol violation may occur, which can result in operation faults. Always make sure that settings are within the range of limit values.

- (a) $5 \times \text{BTL} \leq \text{SPT (sampling point)} \leq 17 \times \text{BTL}$ [$4 \leq \text{SPT4 to SPT0 set values} \leq 16$]
- (b) $8 \times \text{BTL} \leq \text{DBT (data bit time)} \leq 25 \times \text{BTL}$ [$7 \leq \text{DBT4 to DBT0 set values} \leq 24$]
- (c) $\text{SJW (Synchronization Jump Width)} < \text{DBT} - \text{SPT}$
- (d) $2 \leq (\text{DBT} - \text{SPT}) \leq 8$

Remark BTL = $1/f_{\text{BTL}}$ (f_{BTL} : CAN protocol layer basic system clock)
 SPT4 to SPT0 (bits 9 to 5 of CANn synchronization control register (CnSYNC))
 DBT4 to DBT0 (bits 4 to 0 of CANn synchronization control register (CnSYNC))

(1) Example of FCAN baud rate setting (when CnBRP register's TLM bit = 0)

The following is an example of how correct settings for the CnBRP register and CnSYNC register can be calculated.

Conditions from CAN bus:

- <1> CAN module frequency (f_{MEM}): 16 MHz
- <2> CAN bus baud rate: 83 kbps
- <3> Sampling point: 75% or more
- <4> SJW: 3 BTL

First, calculate the ratio between the CAN module frequency and the CAN bus baud rate frequency as shown below.

$$f_{\text{MEM}} / \text{CAN bus baud rate} = 16 \text{ MHz} / 83 \text{ kHz} \approx 192.77 \approx 2^6 \times 3$$

Set an even number between 2 and 128 to the CnBRP register's bits BRP5 to BRP0 as the setting for the prescaler (CAN protocol layer basic system clock: f_{BTL}), then set a value between 8 and 25 to the CnSYNC register's bits DTB4 to DBT0 as the data bit time.

Since it is assumed that the SJW value is 3, the maximum setting value for SPT (sampling point) is 4 less than the data bit time setting and is less than 17.

$$(\text{SPT} \leq \text{DBT} - 4 \text{ and } \text{SPT} \leq 17)$$

Given the above limit values, the following 3 settings are possible.

Prescaler	DBT	SPT (MAX.)	Calculated SPT
16	12	8	8/12 = 67%
12	16	12	12/16 = 75%
8	24	17	17/24 = 71%

16 MHz/83 kbps \cong 192	= 64 \times 3	<1>
	= 48 \times 4	<2>
	= 32 \times 6	<3>
	= 24 \times 8	<4>
	= 16 \times 12	<5>
	= 12 \times 16	<6>
	= 8 \times 24	<7>
	= 6 \times 32	<8>
	= 4 \times 48	<9>
	= 3 \times 64	<10>

The settings that can actually be made for the V850/SF1 are in the range from <5> to <7> above (the section enclosed in broken lines).

Among these options in the range from <5> to <7> above, option <6> is the ideal setting for used when actually setting the register.

(i) Prescaler (CAN protocol layer basic system clock: f_{BTL}) setting

f _{BTL} is calculated as below.
• f _{BTL} = f _{MEM} /{(a + 1) \times 2} : [0 \leq a \leq 63]
Value a is set using bits 5 to 0 (BRP5 to BRP0) of the CnBRP register.
f _{BTL} = 16 MHz/12
= 16 MHz/{(5 + 1) \times 2}
thus a = 5
Therefore, CnBRP register = 0005H

(ii) DBT (data bit time) setting

DBT is calculated as below.

- $DBT = BTL \times (a + 1) : [7 \leq a \leq 24]$

Value a is set using bits 4 to 0 (DBT4 to DBT0) of the CnSYNC register.

$$\begin{aligned} DBT &= BTL \times 16 \\ &= BTL \times (a + 1) \\ \text{thus } a &= 15 \end{aligned}$$

Therefore, CnBRP register's bits DBT4 to DBT0 = 01111B

Note that $1/DBT = f_{BTL}/16$

$$\cong 1333 \text{ kHz}/16$$

$$\cong 83 \text{ kbps (nearly equal to the CAN bus baud rate)}$$

(iii) SPT (sampling point) setting

Given SJW = 3:

$$SJW < DBT - SPT$$

$$3 < 15 - SPT$$

$$SPT < 12$$

Therefore, SPT is set as 11 (max.)

SPT is calculated as below.

- $SPT = BTL \times (a + 1) : [4 \leq a \leq 16]$

Value a is set using bits 9 to 5 (SPT4 to SPT0) of the CnSYNC register.

$$\begin{aligned} SPT &= BTL \times 12 \\ &= BTL \times (11 + 1) \\ \text{thus } a &= 11 \end{aligned}$$

(iv) SJW (Synchronization Jump Width) setting

SJW is calculated as below.

- $SJW = BTL \times (a + 1) : [0 \leq a \leq 3]$

Value a is set using bits 11 and 10 (SJW1, SJW0) of the CnSYNC register.

CnSYNC register's bits SJW1 and SJW0 = $BTL \times 3$

$$= BTL \times (2 + 1)$$

$$\text{thus } a = 2$$

The CnSYNC register settings based on these results are shown in Figure 18-46 below.

Figure 18-46. CnSYNC Register Settings

	15	14	13	12	11	10	9	8
CnSYNC	0	0	0	SAMP	SJW1	SJW0	SPT4	SPT3
Setting	0	0	0	0	1	0	0	1
	7	6	5	4	3	2	1	0
	SPT2	SPT1	SPT0	DBT4	DBT3	DBT2	DBT1	DBT0
Setting	0	1	1	0	1	1	1	1

18.13 Prioritization of Message Buffers During Receive Comparison

Several message buffers can be built for receiving messages. In such cases, a system of prioritization in receiving messages must be established for these message buffers.

18.13.1 Reception of data frames

Table 18-23. Prioritization of Message Buffers When Receiving Data Frames

Priority Ranking	Description
1 (High)	<p>Receive messages can be stored in receive message buffers when the receive messages meet the following conditions.</p> <ul style="list-style-type: none"> <1> Messages are not linked to masks <2> M_STATn register's DN bit has not been set (1) (n = 00 to 31) <p>If several messages have met the above conditions, the message buffer having the lowest message number is selected.</p>
2	<p>Receive messages can be stored in receive message buffers when the receive messages meet the following conditions.</p> <ul style="list-style-type: none"> <1> Messages are not linked to masks <2> M_STATn register's DN bit has been set (1) (n = 00 to 31) <p>If several messages have met the above conditions, the message buffer having the lowest message number is selected.</p>
3 (Low)	<p>Receive messages can be stored in receive message buffers that are linked to masks. However, the following conditions must be met.</p> <ul style="list-style-type: none"> <1> If several receive message buffers are linked to masks, the message buffer will be selected based on the mask number assigned to it. The lowest mask number takes the highest priority. <2> If several message buffers meet the conditions mentioned in "<1>" above, the message buffer is selected based on its DN bit (in the M_STATn register). A message buffer in which this bit is not set (1) takes priority over a message buffer in which this bit is set (1). <3> If several message buffers meet the conditions mentioned in "<1>" and "<2>" above, the message buffer with the lowest message number is selected.

18.13.2 Reception of remote frames

Table 18-24. Prioritization of Message Buffers When Receiving Remote Frames

Priority Ranking	Description
1 (High)	<p>Receive messages can be stored in transmit message buffers when the receive messages meet the following conditions.</p> <ul style="list-style-type: none"> • Messages are not linked to masks <p>If several transmit message buffers have met the above conditions, the message buffer having the lowest message number is selected.</p>
2	<p>Receive messages can be stored in receive message buffers when the receive messages meet the following conditions.</p> <ul style="list-style-type: none"> • Messages are not linked to masks • M_STATn register's DN bit has not been set (1) (n = 00 to 31) <p>If several receive message buffers have met the above conditions, the message buffer having the lowest message number is selected.</p>
3	<p>Receive messages can be stored in receive message buffers when the receive messages meet the following conditions.</p> <ul style="list-style-type: none"> • Messages are not linked to masks • M_STATn register's DN bit has been set (1) (n = 00 to 31) <p>If several receive message buffers have met the above conditions, the message buffer having the lowest message number is selected.</p>
4 (Low)	<p>Receive messages can be stored in receive message buffers that are linked to masks. However, the following conditions must be met.</p> <p><1> If several receive message buffers are linked to masks, the message buffer will be selected based on the mask number assigned to it. The lowest mask number takes the highest priority.</p> <p><2> If several message buffers meet the conditions mentioned in "<1>" above, the message buffer is selected based on its DN bit (in the M_STATn register). (n = 00 to 31) A message buffer in which this bit is not set (1) takes priority over a message buffer in which this bit is set (1).</p> <p><3> If several message buffers meet the conditions mentioned in "<1>" and "<2>" above, the message buffer with the lowest message number is selected.</p>

18.14 Ensuring Data Consistency

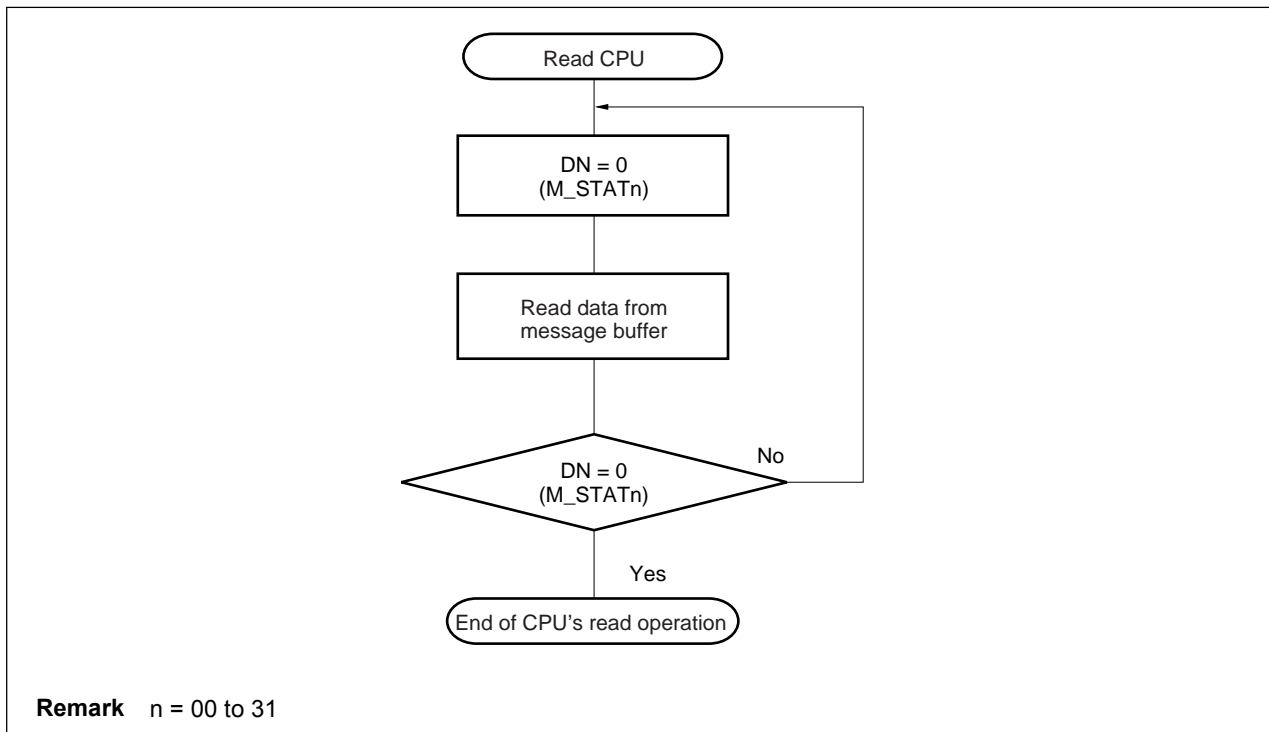
When the CPU reads data from CAN message buffers, it is essential for the read data to be consistent. Two methods are used to ensure data consistency: sequential data read and burst mode data read.

18.14.1 Sequential data read

When the CPU performs sequential access of a message buffer, data is read from the buffer in the order shown in Figure 18-47 below.

Only the FCAN can set the M_STATn register's DN bit (1) and only the CPU can clear it (0), so during the read operation the CPU must be able to check whether or not any new data has been stored in the message buffer.

Figure 18-47. Sequential Data Read



★ 18.14.2 Burst read mode

Burst read mode is implemented in the FCAN to enable faster access to complete messages and secure the synchrony of data.

Burst read mode starts up automatically each time the CPU reads the M_DLCn register and data is then copied from the message buffer area to a temporary read buffer.

Data continues to be read from the temporary buffer as long as the CPU keeps directly incrementing (+1) the read address (in other words, when data is read in the following order: M_DLCn register → M_CTRLn register → M_TIMEn register → M_DATAn0 to M_DATAn7 registers → M_IDLn, M_IDHn register).

If these linear address rules are not followed or if access is attempted to an address that is lower than the M_IDHn register's address (such as the M_CONFn register or M_STATn register), burst read mode becomes invalid.

Caution 16-bit read access is required for the entire message buffer area when using the burst read mode. If 8-bit access (byte read operation) is attempted, burst read mode does not start up even if the address is linearly incremented (+1) as described above.

Remark n = 00 to 31

18.15 Interrupt Conditions

18.15.1 Interrupts that occur for FCAN controller

When interrupts are enabled (condition <1>: the M_CTRLm register's IE bit = 1, conditions other than <1>: C_IE register's interrupt enable flag = 1), interrupts will occur under the following conditions (m = 00 to 31).

- <1> Message-related operation has succeeded
 - When a message has been received in the receive message buffer
 - When a remote frame has been received in the transmit message buffer (only when auto acknowledge mode has not been set, i.e., when the M_CTRLm register's RMDE0 bit = 0)
 - When a message has been transmitted from the transmit message buffer

- <2> When a CAN bus error has been detected
 - Bit error
 - Bit stuff error
 - Form error
 - CRC error
 - ACK error

- <3> When the CAN bus mode has been changed
 - Error passive status elapsed while FCAN was transmitting
 - Bus off status was set while FCAN was transmitting
 - Error passive status elapsed while FCAN was receiving

- <4> Internal error
 - Overrun error

★ 18.15.2 Interrupts that occur for global CAN interface

Interrupts occur for the global CAN interface under the following conditions.

- Access to undefined area
- When clearing (0) of the GOM bit is attempted with the EFSD bit of the CGST register = 0, when there is even one CAN module not initialized (ISTAT bit of CnCTRL register = 0)
- Access to the CAN module register (register name starting with "Cn" (n = 1, 2)), when the GOM bit of the CGST register = 0
- Access to a temporary buffer when the GOM bit of the CGST register = 1 (area after address of CnSYNC register)

18.16 How to Shut down FCAN Controller

The following procedure should be used to stop CAN bus operations in order to stop the clock supply to the CAN interface (to set low power mode).

- <1> Set FCAN controller initialization mode
 - Set initialization mode (INIT bit = 1 in CnCTRL register (set INIT bit = 1, clear INIT bit = 0)) (n = 1, 2)
- <2> Stop time stamp counter
 - Set TSM bit = 0 in CGST register (set TSM bit = 0, clear TSM bit = 1)
- <3> Stop CAN interface
 - Set GOM bit = 0 in CGST register (set GOM bit = 0, clear GOM bit = 1)
 - Stop CAN clock

- Cautions**
1. If the above procedure is not performed correctly, the CAN interface (in active status) can cause operation faults.
 2. Perform only steps <1> and <2> above when setting the CPU itself to low power mode and operating the CAN interface using a CPU-supplied clock.
In this case, stopping the CAN interface clock (step <3> above) is automatically performed when the CPU clock is stopped, so there is no need to perform step <3>.

★ 18.17 Cautions on Use

- <1> Bit manipulation is prohibited for all FCAN controller registers.
- <2> Be sure to properly clear (0) all interrupt request flags^{Note} in the interrupt routine. If these flags are not cleared (0), subsequent interrupt requests may not be generated. Note also that if an interrupt is generated at the same time as a CPU clear operation, that interrupt request flag will not be cleared (0). It is therefore important to confirm that interrupt request flags have been properly cleared (0).

Note See 18.4.9 CAN interrupt pending register (CCINTP), 18.4.10 CAN global interrupt pending register (CGINTP), and 18.4.11 CANn interrupt pending register (CnINTP).

- <3> When a change occurs on the CAN bus via a setting of the CSTOP bit in the CSTOP register while the clock supply to the CPU or peripheral functions is stopped, the CPU can be woken up.

APPENDIX A REGISTER INDEX

(1/8)

Symbol	Name	Unit	Page
ADCR	A/D conversion result register	ADC	362
ADCRH	A/D conversion result register H (higher 8 bits)	ADC	362
ADIC	Interrupt control register	ADC	174
ADM1	A/D converter mode register 1	ADC	364
ADM2	A/D converter mode register 2	ADC	366
ADS	Analog input channel specification register	ADC	366
ASIM0	Asynchronous serial interface mode register 0	UART	331
ASIM1	Asynchronous serial interface mode register 1	UART	331
ASIS0	Asynchronous serial interface status register 0	UART	332
ASIS1	Asynchronous serial interface status register 1	UART	332
BCC	Bus cycle control register	BCU	146
BRGC0	Baud rate generator control register 0	BRG	333
BRGC1	Baud rate generator control register 1	BRG	333
BRGCK4	Baud rate generator output clock selection register 4	BRG	354
BRGCN4	Baud rate generator source clock selection register 4	BRG	353
BRGMC00	Baud rate generator mode control register 00	BRG	334
BRGMC01	Baud rate generator mode control register 01	BRG	334
BRGMC10	Baud rate generator mode control register 10	BRG	334
BRGMC11	Baud rate generator mode control register 11	BRG	334
C1BA	CAN1 bus active register	FCAN	459
C1BRP	CAN1 bit rate prescaler register	FCAN	460
C1CTRL	CAN1 control register	FCAN	448
C1DEF	CAN1 definition register	FCAN	452
C1DINF	CAN1 bus diagnostic information register	FCAN	463
C1ERC	CAN1 error count register	FCAN	456
C1IE	CAN1 interrupt enable register	FCAN	457
C1INTP	CAN1 interrupt pending register	FCAN	435
C1LAST	CAN1 information register	FCAN	455
C1MASKH0	CAN1 address mask 0 register H	FCAN	446
C1MASKH1	CAN1 address mask 1 register H	FCAN	446
C1MASKH2	CAN1 address mask 2 register H	FCAN	446
C1MASKH3	CAN1 address mask 3 register H	FCAN	446
C1MASKL0	CAN1 address mask 0 register L	FCAN	446
C1MASKL1	CAN1 address mask 1 register L	FCAN	446
C1MASKL2	CAN1 address mask 2 register L	FCAN	446
C1MASKL3	CAN1 address mask 3 register L	FCAN	446

Symbol	Name	Unit	Page
C1SYNC	CAN1 synchronization control register	FCAN	464
C2BA	CAN2 bus active register	FCAN	459
C2BRP	CAN2 bit rate prescaler register	FCAN	460
C2CTRL	CAN2 control register	FCAN	448
C2DEF	CAN2 definition register	FCAN	452
C2DINF	CAN2 bus diagnostic information register	FCAN	463
C2ERC	CAN2 error count register	FCAN	456
C2IE	CAN2 interrupt enable register	FCAN	457
C2INTP	CAN2 interrupt pending register	FCAN	435
C2LAST	CAN2 information register	FCAN	455
C2MASKH0	CAN2 address mask 0 register H	FCAN	446
C2MASKH1	CAN2 address mask 1 register H	FCAN	446
C2MASKH2	CAN2 address mask 2 register H	FCAN	446
C2MASKH3	CAN2 address mask 3 register H	FCAN	446
C2MASKL0	CAN2 address mask 0 register L	FCAN	446
C2MASKL1	CAN2 address mask 1 register L	FCAN	446
C2MASKL2	CAN2 address mask 2 register L	FCAN	446
C2MASKL3	CAN2 address mask 3 register L	FCAN	446
C2SYNC	CAN2 synchronization control register	FCAN	464
CANIC1	Interrupt control register	INTC	174
CANIC2	Interrupt control register	INTC	174
CANIC3	Interrupt control register	INTC	174
CANIC4	Interrupt control register	INTC	174
CANIC5	Interrupt control register	INTC	174
CANIC6	Interrupt control register	INTC	174
CANIC7	Interrupt control register	INTC	174
CCINTP	CAN interrupt pending register	FCAN	433
CGCS	CAN main clock select register	FCAN	441
CGIE	CAN global interrupt enable register	FCAN	440
CGINTP	CAN global interrupt pending register	FCAN	434
CGMSR	CAN message search result register	FCAN	444
CGMSS	CAN message search start register	FCAN	444
CGST	CAN global status register	FCAN	438
CGTSC	CAN time stamp count register	FCAN	443
CORAD0	Correction address register 0	CPU	388
CORAD1	Correction address register 1	CPU	388
CORAD2	Correction address register 2	CPU	388
CORAD3	Correction address register 3	CPU	388

Symbol	Name	Unit	Page
CORCN	Correction control register	CPU	387
CORRQ	Correction request register	CPU	388
CR00	16-bit capture/compare register 00	RPU	195
CR01	16-bit capture/compare register 01	RPU	196
CR10	16-bit capture/compare register 10	RPU	195
CR11	16-bit capture/compare register 11	RPU	196
CR2	16-bit compare register 2	RPU	231
CR3	16-bit compare register 3	RPU	231
CR4	16-bit compare register 4	RPU	231
CR5	16-bit compare register 5	RPU	231
CR6	16-bit compare register 6	RPU	231
CR70	16-bit capture/compare register 70	RPU	195
CR71	16-bit capture/compare register 71	RPU	196
CRC0	Capture/compare control register 0	RPU	200
CRC1	Capture/compare control register 1	RPU	200
CRC7	Capture/compare control register 7	RPU	200
CSIB4	Variable-length serial setting register	CSI	352
CSIC0	Interrupt control register	INTC	174
CSIC1	Interrupt control register	INTC	174
CSIC3	Interrupt control register	INTC	174
CSIC4	Interrupt control register	INTC	174
CSIM0	Serial operation mode register 0	CSI	262
CSIM1	Serial operation mode register 1	CSI	262
CSIM3	Serial operation mode register 3	CSI	262
CSIM4	Variable-length serial control register	CSI	351
CSIS0	Serial clock selection register 0	CSI	263
CSIS1	Serial clock selection register 1	CSI	263
CSIS3	Serial clock selection register 3	CSI	263
CSTOP	CAN stop register	FCAN	437
DBC0	DMA byte count register 0	DMAC	379
DBC1	DMA byte count register 1	DMAC	379
DBC2	DMA byte count register 2	DMAC	379
DBC3	DMA byte count register 3	DMAC	379
DBC4	DMA byte count register 4	DMAC	379
DBC5	DMA byte count register 5	DMAC	379
DCHC0	DMA channel control register 0	DMAC	380
DCHC1	DMA channel control register 1	DMAC	380

APPENDIX A REGISTER INDEX

(4/8)

Symbol	Name	Unit	Page
DCHC2	DMA channel control register 2	DMAC	380
DCHC3	DMA channel control register 3	DMAC	380
DCHC4	DMA channel control register 4	DMAC	380
DCHC5	DMA channel control register 5	DMAC	380
DIOA0	DMA peripheral I/O address register 0	DMAC	377
DIOA1	DMA peripheral I/O address register 1	DMAC	377
DIOA2	DMA peripheral I/O address register 2	DMAC	377
DIOA3	DMA peripheral I/O address register 3	DMAC	377
DIOA4	DMA peripheral I/O address register 4	DMAC	377
DIOA5	DMA peripheral I/O address register 5	DMAC	377
DMAIC0	Interrupt control register	INTC	174
DMAIC1	Interrupt control register	INTC	174
DMAIC2	Interrupt control register	INTC	174
DMAIC3	Interrupt control register	INTC	174
DMAIC4	Interrupt control register	INTC	174
DMAIC5	Interrupt control register	INTC	174
DMAS	DMA start factor expansion register	DMAC	379
DRA0	DMA internal RAM address register 0	DMAC	378
DRA1	DMA internal RAM address register 1	DMAC	378
DRA2	DMA internal RAM address register 2	DMAC	378
DRA3	DMA internal RAM address register 3	DMAC	378
DRA4	DMA internal RAM address register 4	DMAC	378
DRA5	DMA internal RAM address register 5	DMAC	378
DWC	Data wait control register	BCU	144
ECR	Interrupt source register	CPU	61
EGN0	Falling edge specification register 0	INTC	107, 165
EGP0	Rising edge specification register 0	INTC	106, 165
IIC0	IIC shift register 0	I ² C	269, 282
IICC0	IIC control register 0	I ² C	271
IICCE0	IIC clock expansion register 0	I ² C	280
IICCL0	IIC clock selection register 0	I ² C	279
IICCS0	IIC status register 0	I ² C	276
IICX0	IIC function expansion register 0	I ² C	280
ISPR	In-service priority register	INTC	175
KRIC	Interrupt control register	INTC	174
KRM	Key return mode register	KR	190
MM	Memory expansion mode register	Port	74

APPENDIX A REGISTER INDEX

(5/8)

Symbol	Name	Unit	Page
M_CONF00 to M_CONF31	CAN message configuration registers 00 to 31	FCAN	427
M_CTRL00 to M_CTRL31	CAN message control registers 00 to 31	FCAN	419
M_DATA000 to M_DATA317	CAN message data registers 000 to 317	FCAN	423
M_DLC00 to M_DLC31	CAN message data length registers 00 to 31	FCAN	418
M_IDH00 to M_IDH31	CAN message ID registers H00 to H31	FCAN	425
M_IDL00 to M_IDL31	CAN message ID registers L00 to L31	FCAN	425
M_STAT00 to M_STAT31	CAN message status registers 00 to 31	FCAN	429
M_TIME00 to M_TIME31	CAN message time stamp registers 00 to 31	FCAN	421
NCC	Noise elimination control register	INTC	177
OSTS	Oscillation stabilization time selection register	WDT	92, 253
P0	Port 0	Port	104
P1	Port 1	Port	108
P10	Port 10	Port	129
P11	Port 11	Port	132
P2	Port 2	Port	112
P3	Port 3	Port	115
P4	Port 4	Port	118
P5	Port 5	Port	118
P6	Port 6	Port	121
P7	Port 7	Port	124
P8	Port 8	Port	124
P9	Port 9	Port	126
PAC	Port alternate-function control register	Port	133
PCC	Processor clock control register	CG	89
PF1	Port 1 function register	Port	109
PIC0	Interrupt control register	INTC	174
PIC1	Interrupt control register	INTC	174

Symbol	Name	Unit	Page
PIC2	Interrupt control register	INTC	174
PIC3	Interrupt control register	INTC	174
PIC4	Interrupt control register	INYC	174
PIC5	Interrupt control register	INTC	174
PIC6	Interrupt control register	INTC	174
PM0	Port 0 mode register	Port	106
PM1	Port 1 mode register	Port	109
PM10	Port 10 mode register	Port	130
PM11	Port 11 mode register	Port	133
PM2	Port 2 mode register	Port	113
PM3	Port 3 mode register	Port	116
PM4	Port 4 mode register	Port	119
PM5	Port 5 mode register	Port	119
PM6	Port 6 mode register	Port	122
PM9	Port 9 mode register	Port	127
POCS	POC status register	Reset	383
PRCMD	Command register	CG	87
PRM00	Prescaler mode register 00	RPU	203
PRM01	Prescaler mode register 01	RPU	203
PRM10	Prescaler mode register 10	RPU	205
PRM11	Prescaler mode register 11	RPU	205
PRM70	Prescaler mode register 70	RPU	205
PRM71	Prescaler mode register 71	RPU	205
PSC	Power save control register	CG	91
PSW	Program status word	CPU	62
PU10	Pull-up resistor option register 10	Port	130
RXB0	Receive buffer register 0	UART	330
RXB1	Receive buffer register 1	UART	330
SC_STAT00 to SC_STAT31	CAN status set/clear registers 00 to 31	FCAN	431
SERIC0	Interrupt control register	INTC	174
SERIC1	Interrupt control register	INTC	174
SIO0	Serial I/O shift register 0	CSI	260
SIO1	Serial I/O shift register 1	CSI	260
SIO3	Serial I/O shift register 3	CSI	260
SIO4	Variable-length serial I/O shift register	CSI	349
STIC0	Interrupt control register	INTC	174
STIC1	Interrupt control register	INTC	174

APPENDIX A REGISTER INDEX

(7/8)

Symbol	Name	Unit	Page
SVA0	Slave address register 0	I ² C	269, 282
SYS	System status register	CG	87
TCL20	Timer clock selection register 20	RPU	232
TCL21	Timer clock selection register 21	RPU	232
TCL30	Timer clock selection register 30	RPU	232
TCL31	Timer clock selection register 31	RPU	232
TCL40	Timer clock selection register 40	RPU	232
TCL41	Timer clock selection register 41	RPU	232
TCL50	Timer clock selection register 50	RPU	232
TCL51	Timer clock selection register 51	RPU	232
TCL60	Timer clock selection register 60	RPU	232
TCL61	Timer clock selection register 61	RPU	232
TM0	16-bit timer register 0	RPU	194
TM1	16-bit timer register 1	RPU	194
TM2	16-bit counter 2	RPU	231
TM3	16-bit counter 3	RPU	231
TM4	16-bit counter 4	RPU	231
TM5	16-bit counter 5	RPU	231
TM6	16-bit counter 6	RPU	231
TM7	16-bit timer register 7	RPU	194
TMC0	16-bit timer mode control register 0	RPU	197
TMC1	16-bit timer mode control register 1	RPU	197
TMC20	Timer mode control register 20	RPU	235
TMC30	Timer mode control register 30	RPU	235
TMC40	Timer mode control register 40	RPU	235
TMC50	Timer mode control register 50	RPU	235
TMC60	Timer mode control register 60	RPU	235
TMC7	16-bit timer mode control register 7	RPU	197
TMIC00	Interrupt control register	INTC	174
TMIC01	Interrupt control register	INTC	174
TMIC10	Interrupt control register	INTC	174
TMIC11	Interrupt control register	INTC	174
TMIC2	Interrupt control register	INTC	174
TMIC3	Interrupt control register	INTC	174
TMIC4	Interrupt control register	INTC	174
TMIC5	Interrupt control register	INTC	174
TMIC6	Interrupt control register	INTC	174
TMIC70	Interrupt control register	INTC	174

APPENDIX A REGISTER INDEX

(8/8)

Symbol	Name	Unit	Page
TMIC71	Interrupt control register	INTC	174
TOC0	Timer output control register 0	RPU	201
TOC1	Timer output control register 1	RPU	201
TOC7	Timer output control register 7	RPU	201
TXS0	Transmit shift register 0	UART	329
TXS1	Transmit shift register 1	UART	329
VM45C	VM45 control register	Reset	384
WDCS	Watchdog timer clock selection register	WDT	254
WDTIC	Interrupt control register	INTC	174
WDTM	Watchdog timer mode register	WDT	176, 255
WTNCS	Watch timer clock selection register	WT	247
WTNIC	Interrupt control register	INTC	174
WTNIIC	Interrupt control register	INTC	174
WTNM	Watch timer mode control register	WT	246

APPENDIX B INSTRUCTION SET LIST

- How to read instruction set list

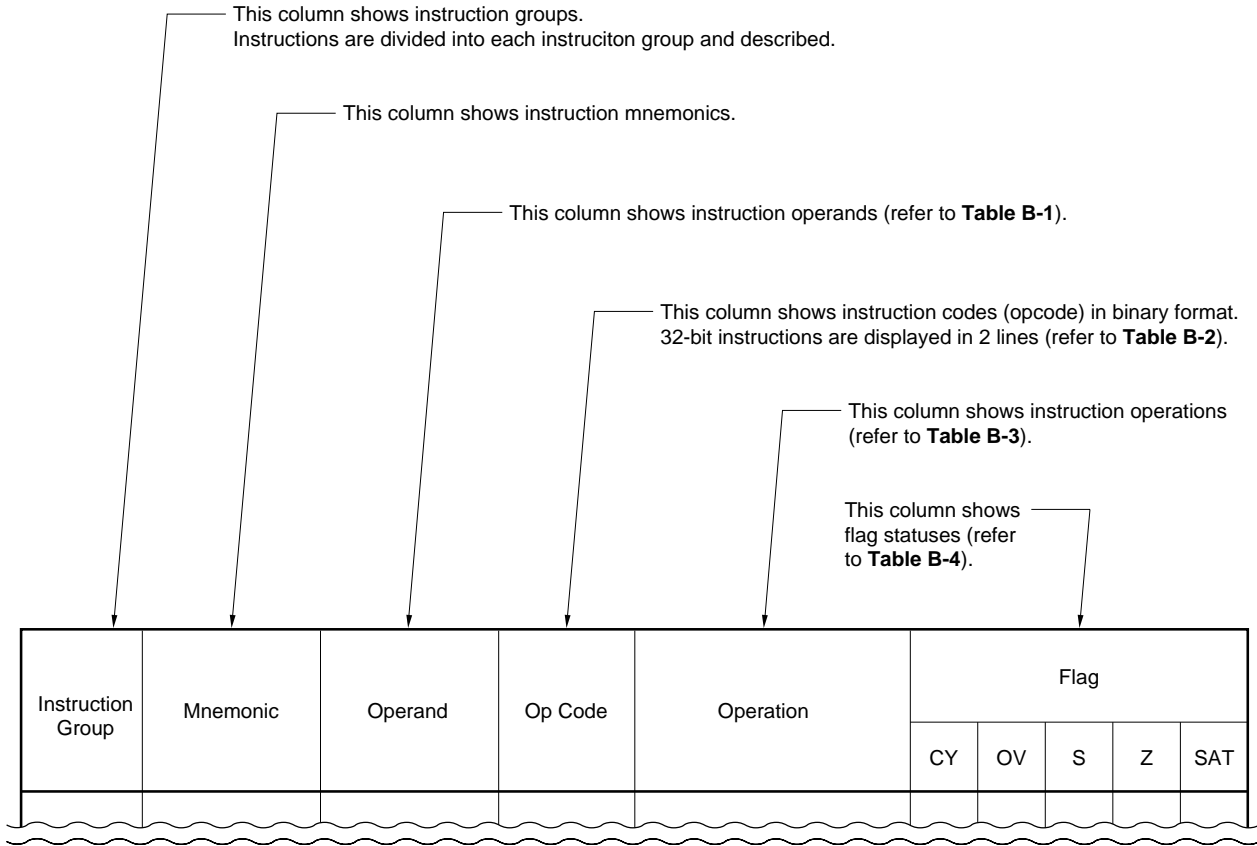


Table B-1. Symbols in Operand Description

Symbol	Description
reg1	General-purpose register (r0 to r31): Used as source register
reg2	General-purpose register (r0 to r31): Mainly used as destination register
ep	Element pointer (r30)
bit#3	3-bit data for bit number specification
imm×	×-bit immediate data
disp×	×-bit displacement
regID	System register number
vector	5-bit data that specifies trap vector number (00H to 1FH)
cccc	4-bit data that indicates condition code

Table B-2. Symbols Used for Op Code

Symbol	Description
R	1-bit data of code that specifies reg1 or regID
r	1-bit data of code that specifies reg2
d	1-bit data of displacement
i	1-bit data of immediate data
cccc	4-bit data that indicates condition code
bbb	3-bit data that specifies bit number

Table B-3. Symbols Used for Operation Description

Symbol	Description
←	Assignment
GR[]	General-purpose register
SR[]	System register
zero-extend (n)	Zero-extends n to word length.
sign-extend (n)	Sign-extends n to word length.
load-memory (a,b)	Reads data of size b from address a.
store-memory (a,b,c)	Writes data b of size c to address a.
load-memory-bit (a,b)	Reads bit b from address a.
store-memory-bit (a,b,c)	Writes c to bit b of address a
saturated (n)	Performs saturated processing of n. (n is 2's complements). Result of calculation of n: If n is $n \geq 7FFFFFFFH$ as result of calculation, $7FFFFFFFH$. If n is $n \leq 80000000H$ as result of calculation, $80000000H$.
result	Reflects result to a flag.
Byte	Byte (8 bits)
Halfword	Halfword (16 bits)
Word	Word (32 bits)
+	Add
-	Subtract
	Bit concatenation
×	Multiply
÷	Divide
AND	Logical product
OR	Logical sum
XOR	Exclusive logical sum
NOT	Logical negate
logically shift left by	Logical left shift
logically shift right by	Logical right shift
arithmetically shift right by	Arithmetic right shift

Table B-4. Symbols Used for Flag Operation

Symbol	Description
(blank)	Not affected
0	Cleared to 0
×	Set of cleared according to result
R	Previously saved value is restored

Table B-5. Condition Codes

Condition Name (cond)	Condition Code (cccc)	Conditional Expression	Description
V	0000	$OV = 1$	Overflow
NV	1000	$OV = 0$	No overflow
C/L	0001	$CY = 1$	Carry Lower (Less than)
NC/NL	1001	$CY = 0$	No carry No lower (Greater than or equal)
Z/E	0010	$Z = 1$	Zero Equal
NZ/NE	1010	$Z = 0$	Not zero Not equal
NH	0011	$(CY \text{ OR } Z) = 1$	Not higher (Less than or equal)
H	1011	$(CY \text{ OR } Z) = 0$	Higher (Greater than)
N	0100	$S = 1$	Negative
P	1100	$S = 0$	Positive
T	0101	–	Always (unconditional)
SA	1101	$SAT = 1$	Saturated
LT	0110	$(S \text{ XOR } OV) = 1$	Less than signed
GE	1110	$(S \text{ XOR } OV) = 0$	Greater than or equal signed
LE	0111	$((S \text{ XOR } OV) \text{ OR } Z) = 1$	Less than or equal signed
GT	1111	$((S \text{ XOR } OV) \text{ OR } Z) = 0$	Greater than signed

Instruction Set List (1/4)

Instruction Group	Mnemonic	Operand	Op Code	Operation	Flag				
					CY	OV	S	Z	SAT
Load/store	SLD.B	disp7 [ep], reg2	rrrrr0110ddddddd	adr ← ep + zero-extend (disp7) GR [reg2] ← sign-extend (Load-memory (adr, Byte))					
	SLD.H	disp8 [ep], reg2	rrrrr1000ddddddd Note 1	adr ← ep + zero-extend (disp8) GR [reg2] ← sign-extend (Load-memory (adr, Halfword))					
	SLD.W	disp8 [ep], reg2	rrrrr1010dddddd0 Note 2	adr ← ep + zero-extend (disp8) GR [reg2] ← Load-memory (adr, Word)					
	LD.B	disp16 [reg1], reg2	rrrrr111000RRRRR ddddddddddddddd	adr ← GR [reg1] + sign-extend (disp16) GR [reg2] ← sign-extend (Load-memory (adr, Byte))					
	LD.H	disp16 [reg1], reg2	rrrrr111001RRRRR ddddddddddddddd0 Note 3	adr ← GR [reg1] + sign-extend (disp16) GR [reg2] ← sign-extend (Load-memory (adr, Halfword))					
	LD.W	disp16 [reg1], reg2	rrrrr111001RRRRR ddddddddddddddd1 Note 3	adr ← GR [reg1] + sign-extend (disp16) GR [reg2] ← Load-memory (adr, Word)					
	SST.B	reg2, disp7 [ep]	rrrrr0111ddddddd	adr ← ep + zero-extend (disp7) Store-memory (adr, GR [reg2], Byte)					
	SST.H	reg2, disp8 [ep]	rrrrr1001ddddddd Note 1	adr ← ep + zero-extend (disp8) Store-memory (adr, GR [reg2], Halfword)					
	SST.W	reg2, disp8 [ep]	rrrrr1010dddddd1 Note 2	adr ← ep + zero-extend (disp8) Store-memory (adr, GR [reg2], Word)					
	ST.B	reg2, disp16 [reg1]	rrrrr111010RRRRR ddddddddddddddd	adr ← GR [reg1] + sign-extend (disp16) Store-memory (adr, GR [reg2], Byte)					
	ST.H	reg2, disp16 [reg1]	rrrrr111011RRRRR ddddddddddddddd0 Note 3	adr ← GR [reg1] + sign-extend (disp16) Store-memory (adr, GR [reg2], Halfword)					
	ST.W	reg2, disp16 [reg1]	rrrrr111011RRRRR ddddddddddddddd1 Note 3	adr ← GR [reg1] + sign-extend (disp16) Store-memory (adr, GR [reg2], Word)					
Arithmetic operation	MOV	reg1, reg2	rrrrr000000RRRRR	GR [reg2] ← GR [reg1]					
	MOV	imm5, reg2	rrrrr010000iiii	GR [reg2] ← sign-extend (imm5)					
	MOVHI	imm16, reg1, reg2	rrrrr110010RRRRR iiiiiiiiiiiiiii	GR [reg2] ← GR [reg1] + (imm16 0 ¹⁶)					
	MOVEA	imm16, reg1, reg2	rrrrr110001RRRRR iiiiiiiiiiiiiii	GR [reg2] ← GR [reg1] + sign-extend (imm16)					

- Notes**
1. ddddddd is the higher 7 bits of disp8.
 2. ddddddd is the higher 6 bits of disp8.
 3. ddddddddddddddd is the higher 15 bits of disp16.

Instruction Set List (2/4)

Instruction Group	Mnemonic	Operand	Op Code	Operation	Flag				
					CY	OV	S	Z	SAT
Arithmetic operation	ADD	reg1, reg2	rrrr001110RRRRR	$GR [reg2] \leftarrow GR [reg2] + GR [reg1]$	x	x	x	x	
	ADD	imm5, reg2	rrrr010010iiii	$GR [reg2] \leftarrow GR [reg2] + \text{sign-extend} (imm5)$	x	x	x	x	
	ADDI	imm16, reg1, reg2	rrrr110000RRRRR iiiiiiiiiiii	$GR [reg2] \leftarrow GR [reg1] + \text{sign-extend} (imm16)$	x	x	x	x	
	SUB	reg1, reg2	rrrr001101RRRRR	$GR [reg2] \leftarrow GR [reg2] - GR [reg1]$	x	x	x	x	
	SUBR	reg1, reg2	rrrr001100RRRRR	$GR [reg2] \leftarrow GR [reg1] - GR [reg2]$	x	x	x	x	
	MULH	reg1, reg2	rrrr000111RRRRR	$GR [reg2] \leftarrow GR [reg2] \overset{\text{Note}}{\times} GR [reg1] \overset{\text{Note}}{}$ (Signed multiplication)					
	MULH	imm5, reg2	rrrr010111iiii	$GR [reg2] \leftarrow GR [reg2] \overset{\text{Note}}{\times} \text{sign-extend} (imm5) \overset{\text{Note}}{}$ (Signed multiplication)					
	MULHI	imm16, reg1, reg2	rrrr110111RRRRR iiiiiiiiiiii	$GR [reg2] \leftarrow GR [reg1] \overset{\text{Note}}{\times} imm16 \overset{\text{Note}}{}$ (Signed multiplication)					
	DIVH	reg1, reg2	rrrr000010RRRRR	$GR [reg2] \leftarrow GR [reg2] \div GR [reg1] \overset{\text{Note}}{}$ (Signed division)		x	x	x	
	CMP	reg1, reg2	rrrr001111RRRRR	$result \leftarrow GR [reg2] - GR [reg1]$	x	x	x	x	
	CMP	imm5, reg2	rrrr010011iiii	$result \leftarrow GR [reg2] - \text{sign-extend} (imm5)$	x	x	x	x	
	SETF	cccc, reg2	rrrr111110cccc 0000000000000000	if conditions are satisfied then $GR [reg2] \leftarrow 00000001H$ else $GR [reg2] \leftarrow 00000000H$					
Saturated operation	SATADD	reg1, reg2	rrrr000110RRRRR	$GR [reg2] \leftarrow \text{saturated} (GR [reg2] + GR [reg1])$	x	x	x	x	x
	SATADD	imm5, reg2	rrrr010001iiii	$GR [reg2] \leftarrow \text{saturated} (GR [reg2] + \text{sign-extend} (imm5))$	x	x	x	x	x
	SATSUB	reg1, reg2	rrrr000101RRRRR	$GR [reg2] \leftarrow \text{saturated} (GR [reg2] - GR [reg1])$	x	x	x	x	x
	SATSUBI	imm16, reg1, reg2	rrrr110011RRRRR iiiiiiiiiiii	$GR [reg2] \leftarrow \text{saturated} (GR [reg1] - \text{sign-extend} (imm16))$	x	x	x	x	x
	SATSUBR	reg1, reg2	rrrr000100RRRRR	$GR [reg2] \leftarrow \text{saturated} (GR [reg1] - GR [reg2])$	x	x	x	x	x
Logic operation	TST	reg1, reg2	rrrr001011RRRRR	$result \leftarrow GR [reg2] \text{ AND } GR [reg1]$		0	x	x	
	OR	reg1, reg2	rrrr001000RRRRR	$GR [reg2] \leftarrow GR [reg2] \text{ OR } GR [reg1]$		0	x	x	
	ORI	imm16, reg1, reg2	rrrr110100RRRRR iiiiiiiiiiii	$GR [reg2] \leftarrow GR [reg1] \text{ OR zero-extend} (imm16)$		0	x	x	
	AND	reg1, reg2	rrrr001010RRRRR	$GR [reg2] \leftarrow GR [reg2] \text{ AND } GR [reg1]$		0	x	x	
	ANDI	imm16, reg1, reg2	rrrr110110RRRRR iiiiiiiiiiii	$GR [reg2] \leftarrow GR [reg1] \text{ AND zero-extend} (imm16)$		0	0	x	

Note Only the lower halfword data is valid.

Instruction Set List (3/4)

Instruction Group	Mnemonic	Operand	Op Code	Operation	Flag				
					CY	OV	S	Z	SAT
Logic operation	XOR	reg1, reg2	rrrr001001RRRRR	GR [reg2] ← GR [reg2] XOR GR [reg1]		0	×	×	
	XORI	imm16, reg1, reg2	rrrr110101RRRRR iiiiiiiiiiiiiii	GR [reg2] ← GR [reg1] XOR zero-extend (imm16)		0	×	×	
	NOT	reg1, reg2	rrrr000001RRRRR	GR [reg2] ← NOT (GR [reg1])		0	×	×	
	SHL	reg1, reg2	rrrr111111RRRRR 0000000011000000	GR [reg2] ← GR [reg2] logically shift left by GR [reg1]	×	0	×	×	
	SHL	imm5, reg2	rrrr010110iiii	GR [reg2] ← GR [reg2] logically shift left by zero-extend (imm5)	×	0	×	×	
	SHR	reg1, reg2	rrrr111111cccc 0000000010000000	GR [reg2] ← GR [reg2] logically shift right by GR [reg1]	×	0	×	×	
	SHR	imm5, reg2	rrrr010100iiii	GR [reg2] ← GR [reg2] logically shift right by zero-extend (imm5)	×	0	×	×	
	SAR	reg1, reg2	rrrr111111RRRRR 0000000010100000	GR [reg2] ← GR [reg2] arithmetically shift right by GR [reg1]	×	0	×	×	
	SAR	imm5, reg2	rrrr010101iiii	GR [reg2] ← GR [reg2] arithmetically shift right by zero-extend (imm5)	×	0	×	×	
Jump	JMP	[reg1]	0000000011RRRRR	PC ← GR [reg1]					
	JR	disp22	000011110ddddd dddddddddddddd0 Note 1	PC ← PC + sign-extend (disp22)					
	JARL	disp22, reg2	rrrr11110ddddd dddddddddddddd0 Note 1	GR [reg2] ← PC + 4 PC ← PC + sign-extend (disp22)					
	Bcond	disp9	dddd1011ddcccc Note 2	if conditions are satisfied then PC ← PC + sign-extend (disp9)					
Bit manipulate	SET1	bit#3, disp16 [reg1]	00bbb11110RRRRR ddddddddddddddd	adr ← GR [reg1] + sign-extend (disp16) Z flag ← Not (Load-memory-bit (adr, bit#3)) Store memory-bit (adr, bit#3, 1)				×	
	CLR1	bit#3, disp16 [reg1]	10bbb11110RRRRR ddddddddddddddd	adr ← GR [reg1] + sign-extend (disp16) Z flag ← Not (Load-memory-bit (adr, bit#3)) Store memory-bit (adr, bit#3, 0)				×	
	NOT1	bit#3, disp16 [reg1]	01bbb11110RRRRR ddddddddddddddd	adr ← GR [reg1] + sign-extend (disp16) Z flag ← Not (Load-memory-bit (adr, bit#3)) Store-memory-bit (adr, bit#3, Z flag)				×	
	TST1	bit#3, disp16 [reg1]	11bbb11110RRRRR ddddddddddddddd	adr ← GR [reg1] + sign-extend (disp16) Z flag ← Not (Load-memory-bit (adr, bit#3))				×	

- Notes 1.** ddddddddddddddddddd is the higher 21 bits of dip22.
2. ddddddd is the higher 8 bits of disp9.

Instruction Set List (4/4)

Instruction Group	Mnemonic	Operand	Op Code	Operation	Flag					
					CY	OV	S	Z	SAT	
Special	LDSR	reg2, regID	rrrrr11111RRRRR 0000000000100000 Note	SR [regID] ← GR [reg2]	regID = EIPC, FEPC					
					regID = EIPSW, FEPSW					
					regID = PSW	×	×	×	×	×
	STSR	regID, reg2	rrrrr11111RRRRR 0000000001000000	GR [reg2] ← SR [regID]						
	TRAP	vector	0000011111iiii 0000000100000000	EIPC ← PC + 4 (Restored PC) EIPSW ← PSW ECR.EICC ← Interrupt code PSW.EP ← 1 PSW.ID ← 1 PC ← 00000040H (vector = 00H to 0FH) 00000050H (vector = 10H to 1FH)						
	RETI		000001111100000 0000000101000000	if PSW.EP = 1 then PC ← EIPC PSW ← EIPSW else if PSW.NP = 1 then PC ← FEPC PSW ← FEPSW else PC ← EIPC PSW ← EIPSW	R	R	R	R	R	
	HALT		000001111100000 0000000100100000	Stops						
	DI		000001111100000 0000000101100000	PSW.ID ← 1 (Maskable interrupt disabled)						
	EI		100001111100000 0000000101100000	PSW.ID ← 0 (Maskable interrupt enabled)						
NOP		0000000000000000	Uses 1 clock cycle without doing anything							

Note The op code of this instruction uses the field of reg1 even though the source register is shown as reg2 in the above table. Therefore, the meaning of register specification for mnemonic description and op code is different from that of the other instructions.

rrrrr = regID specification

RRRRR = reg2 specification

APPENDIX C INDEX

[Number]

16-bit compare registers 2 to 6	231
16-bit counters 2 to 6	231
16-bit timer (TM0, TM1, TM7)	192
16-bit timer (TM2 to TM6)	230
16-bit timer mode control registers 0, 1, 7	197
16-bit timer mode control registers 20 to 60	235
16-bit timer output control registers 0, 1, 7	201
16-bit timer registers 0, 1, 7	194
3-wire serial I/O	259
3-wire variable-length serial I/O	348

[A]

A/D conversion result register	362
A/D conversion result register H	362
A/D converter	360
A/D converter mode register 1	364
A/D converter mode register 2	366
A16 to A21	49
AD0 to AD7	48
AD8 to AD15	49
ADCGND	53
ADCR	362
ADCRH	362
ADCV _{DD}	53
Address match detection method	310
Address space	64
ADIC	174
ADM1	364
ADM2	366
ADS	366
ADTRG	46
Analog input channel specification register	366
ANI0 to ANI11	50
Arbitration	311
Area	68
ASCK0	47
ASCK1	47
ASIM0, ASIM1	331
ASIS0, ASIS1	332
ASTB	51
Asynchronous serial interface	328
Asynchronous serial interface mode	
registers 0, 1	331

Asynchronous serial interface status	
registers 0, 1	332

[B]

Baud rate control function	488
Baud rate generator control registers 0, 1	333
Baud rate generator mode control	
registers n0, n1	334
Baud rate generator output clock selection	
register 4	354
Baud rate generator source clock selection	
register 4	353
BCC	146
BCU	36
Bit stuffing	484
BRGC0, BRGC1	333
BRGCK4	354
BRGCN4	353
BRGMCn0, BRGMCn1	334
Burst read mode	513
Bus control function	140
Bus control pins	140
Bus control unit	36
Bus cycle control register	146
Bus hold function	147
Bus priority	156
Bus timing	149
Bus width	142
Byte access	142

[C]

CAN global interrupt enable register	440
CAN global interrupt pending register	434
CAN global status register	438
CAN interrupt pending register	433
CAN main clock select register	441
CAN message configuration registers 00 to 31	427
CAN message control registers 00 to 31	419
CAN message data length registers 00 to 31	418
CAN message data registers n0 to n7	423
CAN message ID registers L00 to L31 and	
H00 to H31	425
CAN message search start/result register	444
CAN message status registers 00 to 31	429
CAN message time stamp registers 00 to 31	421

CAN module	401	Command register	87
CAN RAM	401	Communication command	398
CAN sleep mode	504	Communication mode	392
CAN status set/clear registers 00 to 31	431	Communication reservation	314
CAN stop mode	506	Configuration of message buffers	403
CAN stop register	437	Connection of unused pins	54
CAN time stamp count register	443	CORAD0 to CORAD3	388
CANIC1 to CANIC7	174	CORCN	387
CANn address mask a registers L and H	446	Correction address registers 0 to 3	388
CANn bit rate prescaler register	460	Correction control register	387
CANn bus active register	459	Correction request register	388
CANn bus diagnostic information register	463	CORRQ	388
CANn control register	448	CPU	36
CANn definition register	452	CPU address space	64
CANn error count register	456	CPU register set	59
CANn information register	455	CPUREG	53
CANn interrupt enable register	457	CR2 to CR6	231
CANn interrupt pending register	435	CRC0, CRC1, CRC7	200
CANn synchronization control register	464	CRn0	195
CANRX1, CANRX2	52	CRn1	196
CANTX1, CANTX2	52	CSI0, CSI1, CSI3	259
Capture/compare control registers 0, 1, 7	200	CSI4	348
Capture/compare register n0	195	CSIB4	352
Capture/compare register n1	196	CSIC0, CSIC1, CSIC3, CSIC4	174
Cautions regarding bit set/clear function	466	CSIM4	351
CCINTP	433	CSIMn	262
CG	36	CSISn	263
CGCS	441	CSTOP	437
CGIE	440		
CGINTP	434	[D]	
CGMSS/CGMSR	444	Data wait control register	144
CGST	438	DBC0 to DBC5	379
CGTSC	443	DCHC0 to DCHC5	380
CLKOUT	53	Determination of bus priority	484
Clock generation function	88	DIOA0 to DIOA5	377
Clock generator (CG)	36	DMA functions	377
Clock output function	89	DMA byte count registers 0 to 5	379
CnBA	459	DMA channel control registers 0 to 5	380
CnBRP	460	DMA internal RAM address registers 0 to 5	378
CnCTRL	448	DMA peripheral I/O address registers 0 to 5	377
CnDEF	452	DMAIC0 to DMAIC5	174
CnDINF	463	DMA start factor expansion register	379
CnERC	456	DMAS	379
CnIE	457	DRA0 to DRA5	378
CnINTP	435	DSTB	51
CnLAST	455	DWC	144
CnMASKLa and CnMASKHa	446		
CnSYNC	464	[E]	

ECR -----	61	IIC shift register 0 -----	269, 282
Edge detection function -----	178	IIC status register 0 -----	276
EGN0 -----	107, 165	IIC0 -----	269, 282
EGP0 -----	106, 165	IICC0 -----	271
EIPC -----	61	IICCE0 -----	280
EIPSW -----	61	IICCL0 -----	279
Ensuring data consistency -----	512	IICS0 -----	276
EP flag -----	181	IICX0 -----	280
Error control function -----	485	Illegal op code -----	181
Error detection -----	310	Images -----	65
Exception trap -----	181	In-service priority register -----	175
Extension code -----	310	Initialization processing -----	491
External expansion mode -----	74	INTC -----	36
External memory -----	72	Internal peripheral I/O area -----	71
External wait function -----	145	Internal RAM area -----	70
		Internal registers of FCAN controller -----	403
[F]		Internal ROM/flash memory area -----	68
FCAN controller -----	400	Interrupt conditions -----	514
Falling edge specification register 0 -----	107, 165	Interrupt control register -----	172
FEPC -----	61	Interrupt controller -----	36
FEPSW -----	61	Interrupt disable flag -----	175
Flash memory -----	390	Interrupt request signal generator -----	270
Flash memory control -----	397	Interrupt source register -----	61
Flash memory programming mode -----	63, 397	Interrupt status saving register -----	61
		Interrupt/exception processing function -----	157
[G]		Interrupts that occur for FCAN controller -----	514
General-purpose registers -----	60	Interrupts that occur for global CAN interface -----	514
GND0 to GND2 -----	53	Interval timer mode -----	252
		INTP0 to INTP6 -----	46
[H]		ISPR -----	175
Halfword access -----	142		
HALT mode -----	94	[K]	
Hardware start -----	360	Key interrupt function -----	190
$\overline{\text{HLDAK}}$ -----	51	Key return mode register -----	190
$\overline{\text{HLDRQ}}$ -----	51	KR0 to KR7 -----	52
		KRIC -----	174
[I]		KRM -----	190
I ² C bus -----	266		
I ² C bus mode -----	266	[L]	
I ² C interrupt request -----	291	$\overline{\text{LBEN}}$ -----	50
IC -----	53	List of FCAN registers -----	404
ID -----	175	Low power consumption mode -----	373
IDLE mode -----	97		
Idle state insertion function -----	146	[M]	
IIC clock expansion register 0 -----	280	M_CONF00 to M_CONF31 -----	427
IIC clock selection register 0 -----	279	M_CTRL00 to M_CTRL31 -----	419
IIC control register 0 -----	271	M_DATAn0 to M_DATAn7 -----	423
IIC function expansion register 0 -----	280	M_DLC00 to M_DLC31 -----	418

M_IDL00 to M_IDL31, M_IDH00 to M_IDH31	425	P20 to P27	47
M_STAT00 to M_STAT31	429	P3	115
M_TIME00 to M_TIME31	421	P30 to P34	48
MAC	401	P4	118
Main system clock oscillator	88	P40 to P47	48
Mask function	472	P5	118
Maskable interrupts	166	P50 to P57	49
Memory block function	143	P6	121
Memory boundary operation condition	156	P60 to P65	49
Memory expansion mode register	74	P7	124
Memory map	67	P70 to P77	50
Message formats	475	P8	124
Message processing	471	P80 to P83	50
MM	74	P9	126
Multi-cast	484	P90 to P96	50
Multiple interrupt servicing	184	PAC	133
Multiple masters	484	PC	60
		PCC	89
[N]		Periods in which interrupts are not acknowledged-	187
NCC	177	Peripheral I/O registers	78
NMI	46	PF1	109
NMI status saving register	61	PIC0 to PIC6	174
Noise elimination	176	Pin functions	38
Noise elimination control register	177	Pin I/O buffer power supply	54
Non-maskable interrupt	160	Pin I/O circuit type	54
Normal operation mode	63	PM0	106
NPB interface	401	PM1	109
Number of access clocks	141	PM10	130
		PM11	133
[O]		PM2	113
Off-board programming	391	PM3	116
On-board programming	391	PM4	119
Operation modes	63	PM5	119
Oscillation stabilization time	101	PM6	122
Oscillation stabilization time		PM9	127
selection register	92, 253	POC status register	383
OSTS	92, 253	POCS	383
		Ports	37
[P]		Port 0	104
P0	104	Port 0 mode register	106
P00 to P07	46	Port 1	108
P1	108	Port 1 function register	109
P10	129	Port 1 mode register	109
P10 to P15	46	Port 10	129
P100 to P107	52	Port 10 mode register	130
P11	132	Port 11	132
P110 to P117	52	Port 11 mode register	133
P2	112	Port 2	112

Port 2 mode register -----	113	Reception of data frames -----	510
Port 3 -----	115	Reception of remote frames-----	511
Port 3 mode register -----	116	Recommended use of address space -----	75
Port 4 -----	118	Regulator -----	385
Port 4 mode register -----	119	$\overline{\text{RESET}}$ -----	53
Port 5 -----	118	Reset function -----	382
Port 5 mode register -----	119	Response time-----	187
Port 6 -----	121	Rising edge specification register 0 -----	106, 165
Port 6 mode register -----	122	ROM -----	36
Port 7 -----	124	ROM correction function -----	386
Port 8 -----	124	Rules for correct setting of baud rate -----	507
Port 9 -----	126	RX0, RX1 -----	329
Port 9 mode register -----	127	RXB0, RXB1 -----	330
Port alternate-function control register -----	133	RXD0 -----	47
PORTGND-----	53	RXD1 -----	47
PORTV _{DD} -----	53		
Power save control register -----	91	[S]	
Power save functions -----	93	SAR -----	362
Power-on-clear operation-----	383	$\overline{\text{SCK0}}$, $\overline{\text{SCK1}}$ -----	46
PRCMD -----	87	$\overline{\text{SCK3}}$, $\overline{\text{SCK4}}$ -----	47
Prescaler mode registers 00, 01 -----	203	SCL0 -----	47
Prescaler mode registers m0, m1 -----	205	SDA0 -----	46
Prioritization of message buffers during receive comparison-----	510	SC_STAT00 to SC_STAT31 -----	431
Priorities of interrupts and exceptions-----	184	Sequential data read -----	512
Priority control -----	184	Serial clock counter -----	270
PRM00, PRM01 -----	203	Serial clock selection register n -----	263
PRMm0, PRMm1 -----	205	Serial I/O shift register n-----	260
Processor clock control register -----	89	Serial interface function -----	259
Program counter -----	60	Serial operation mode register n -----	262
Program register set -----	60	SERIC0, SERIC1 -----	174
Program status word -----	62	SI0, SI1 -----	46
Programmable wait function -----	144	SI3, SI4 -----	47
Programming environment -----	391	Single-chip mode -----	63
Programming method -----	397	SIO _n -----	260
Protocol -----	474	SIO4 -----	349
Protocol mode function -----	474	Slave address register 0 -----	269, 282
PSC -----	91	SO latch -----	269
PSW -----	62	SO0, SO1 -----	46
PU10 -----	130	SO3, SO4 -----	47
Pull-up resistor option register 10 -----	130	Software exception -----	179
		Software start -----	360
[R]		Software STOP mode -----	99
R $\overline{\text{W}}$ -----	51	Specific registers -----	85
RAM -----	36	Standby function -----	347
Receive buffer registers 0, 1 -----	330	Start condition -----	284
Receive setting-----	503	Start factor settings-----	381
Receive shift registers 0, 1 -----	329	STIC0, STIC1 -----	174
		Stop condition -----	288

Subsystem clock oscillator ----- 88
 Successive approximation register -----362
 SVA0 -----269, 282
 SYS ----- 87
 System register set ----- 61
 System status register ----- 87

[T]

TCL20 to TCL60 -----232
 TCL21 to TCL61 -----232
 TI00, TI01, TI10, TI11, TI70 ----- 52
 TI2, TI3, TI4, TI5, TI71 ----- 48
 Time stamp function -----468
 Timer clock selection registers 20 to 60 -----232
 Timer clock selection registers 21 to 61 -----232
 Timer/counter function -----192
 TM0, TM1, TM7 -----192, 194
 TM2 to TM6 -----230, 231
 TMC0, TMC1, TMC7 -----197
 TMC20 to TMC60 -----235
 TMIC00 -----174
 TMIC01 -----174
 TMIC10 -----174
 TMIC11 -----174
 TMIC2 to TMIC6 -----174
 TMIC70, TMIC71 -----174
 TO0, TO1, TO7 ----- 52
 TO2, TO3, TO4, TO5 ----- 48
 TOC0, TOC1, TOC7 -----201
 Transfer completion interrupt request -----377
 Transfer direction specification -----286
 Transmit setting -----502
 Transmit shift registers 0, 1 -----329
 TXD0 ----- 47
 TXD1 ----- 47
 TXS0, TXS1 -----329

[U]

UART0, UART1 -----328
 \overline{UBEN} ----- 51

[V]

Variable-length serial control register 4 ----- 351
 Variable-length serial I/O shift register 4 ----- 349
 Variable-length serial setting register 4 ----- 352
 V_{DD0} -----53
 VM45-----48
 VM45 control register-----384
 VM45C-----384
 V_{PP} -----53

[W]

WAIT -----52
 Wait function ----- 144
 Wait signal-----289
 Wake-up controller -----269
 Wakeup function ----- 313
 Watch timer clock selection register -----247
 Watch timer function -----244
 Watch timer mode control register -----246
 Watchdog timer clock selection register -----254
 Watchdog timer function -----251
 Watchdog timer mode -----252
 Watchdog timer mode register ----- 176, 255
 WDCS -----254
 WDTIC -----174
 WDTM ----- 176, 255
 Word access -----142
 Wraparound of CPU address space -----66
 Write/read time-----391
 Writing with flash programmer -----391
 WTNCS -----247
 WTNIC -----174
 WTNIIC-----174
 WTNM-----246

[X]

X1 -----53
 X2 -----53
 XT1 -----53
 XT2 -----53

Facsimile Message

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.

From:

Name

Company

Tel.

FAX

Address

Thank you for your kind support.

North America

NEC Electronics Inc.
Corporate Communications Dept.
Fax: +1-800-729-9288
+1-408-588-6130

Hong Kong, Philippines, Oceania

NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

Asian Nations except Philippines

NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Europe

NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

Korea

NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: +82-2-528-4411

Japan

NEC Semiconductor Technical Hotline
Fax: +81- 44-435-9608

South America

NEC do Brasil S.A.
Fax: +55-11-6462-6829

Taiwan

NEC Electronics Taiwan Ltd.
Fax: +886-2-2719-5951

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	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Technical Accuracy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>